Assignment: Spatial Diversity

Alison Partee; Z620: Quantitative Biodiversity, Indiana University

08 February, 2017

OVERVIEW

This assignment will emphasize primary concepts and patterns associated with spatial diversity, while using R as a Geographic Information Systems (GIS) environment. Complete the assignment by refering to examples in the handout.

After completing this assignment you will be able to:
1. Begin using R as a geographical information systems (GIS) environment.
2. Identify primary concepts and patterns of spatial diversity.
3. Examine effects of geographic distance on community similarity.
4. Generate simulated spatial data.

Directions:

1. Change "Student Name" on line 3 (above) with your name.
2. Complete as much of the assignment as possible during class; what you do not complete in class will need to be done on your own outside of class.
3. Use the handout as a guide; it contains a more complete description of data sets along with the proper scripting needed to carry out the assignment.
4. Be sure to **answer the questions** in this assignment document. Space for your answer is provided in this document and indicated by the ">" character. If you need a second paragraph be sure to start the first line with ">".
5. Before you leave the classroom, **push** this file to your GitHub repo.
6. When you are done wit the assignment, **Knit** the text and code into an html file.
7. After Knitting, please submit the completed assignment by creating a **pull request** via GitHub. Your pull request should include this file *spatial_assignment.Rmd* and the html output of `Knitr` (*spatial_assignment.html*).

1) R SETUP

In the R code chunk below, provide the code to:

1. Clear your R environment
2. Print your current working directory,
3. Set your working directory to your "*/Week4-Spatial*" folder, and

```
rm(list=ls())
getwd()
setwd("/Users/flopsei/GitHub/QB2017_Partee/Week4-Spatial")
```

2) LOADING R PACKAGES

In the R code chunk below, do the following:

1. Install and/or load the following packages: `vegan`, `sp`, `gstat`, `raster`, `RgoogleMaps`, `maptools`, `rgdal`, `simba`, `gplots`, `rgeos`

```
# install.packages('sp') # Classes and methods for handling spatial data
# install.packages('gstat') # Methods for geostatistical analyses
# install.packages('raster') # Methods to create a RasterLayer object
# install.packages('RgoogleMaps') # For querying the Google server for static maps.
# install.packages('maptools') # Tools for manipulating and reading geospatial data
# install.packages('rgdal') # Geospatial Data Abstraction Library
# install.packages('simba') # Similarity measures for community data
# install.packages('gplots') # Programming tools for plotting data
# install.packages('rgeos') # Geostatistical package, used here for semivariograms

require(rgeos)
require(rgdal)
require(vegan)
require(gplots)
require(simba)
require(gstat)
require(maptools)
require(RgoogleMaps)
require(raster)
require(sp)
```

Question 1: What are the packages `simba`, `sp`, and `rgdal` used for?

> ***Answer 1***: 'simba' is used for its functions that do similarity analysis of vegetation data. 'sp' is a package for spatial data. 'rgdal' is for the Geospatial Data Abstraction Library.

3) LOADING DATA

In the R code chunk below, use the example in the handout to do the following:

1. Load the Site-by-Species matrix for the Indiana ponds datasets: BrownCoData/SiteBySpecies.csv
2. Load the Environmental data matrix: BrownCoData/20130801_PondDataMod.csv
3. Assign the operational taxonomic units (OTUs) to a variable 'otu.names'
4. Remove the first column (i.e., site names) from the OTU matrix.

```
Ponds <- read.table(file = "BrownCoData/20130801_PondDataMod.csv", head = TRUE, sep = ",")
OTUs <- read.csv(file = "BrownCoData/SiteBySpecies.csv", head = TRUE, sep = ",")
otu.names <- names(OTUs) # Get the names of the OTUs
OTUs <- as.data.frame(OTUs[-1]) # remove first column (site names)

S.obs <- function(x = ""){
  # input: site by species matrix
  # output: vector of named nums
  # purpose: sums the amount of cells > 0 for each row
  rowSums(x > 0)
}

siterichs <- S.obs(OTUs)
maxrich <- max(siterichs)
maxrich
maxrichsite <- Ponds$Sample_ID[siterichs == maxrich]
maxrichsite
```

Question 2a: How many sites and OTUs are in the SiteBySpecies matrix?

Answer 2a: There are 51 sites and 16,383 OTUS in the SitebySpecies matrix.

Question 2b: What is the greatest species richness found among sites?

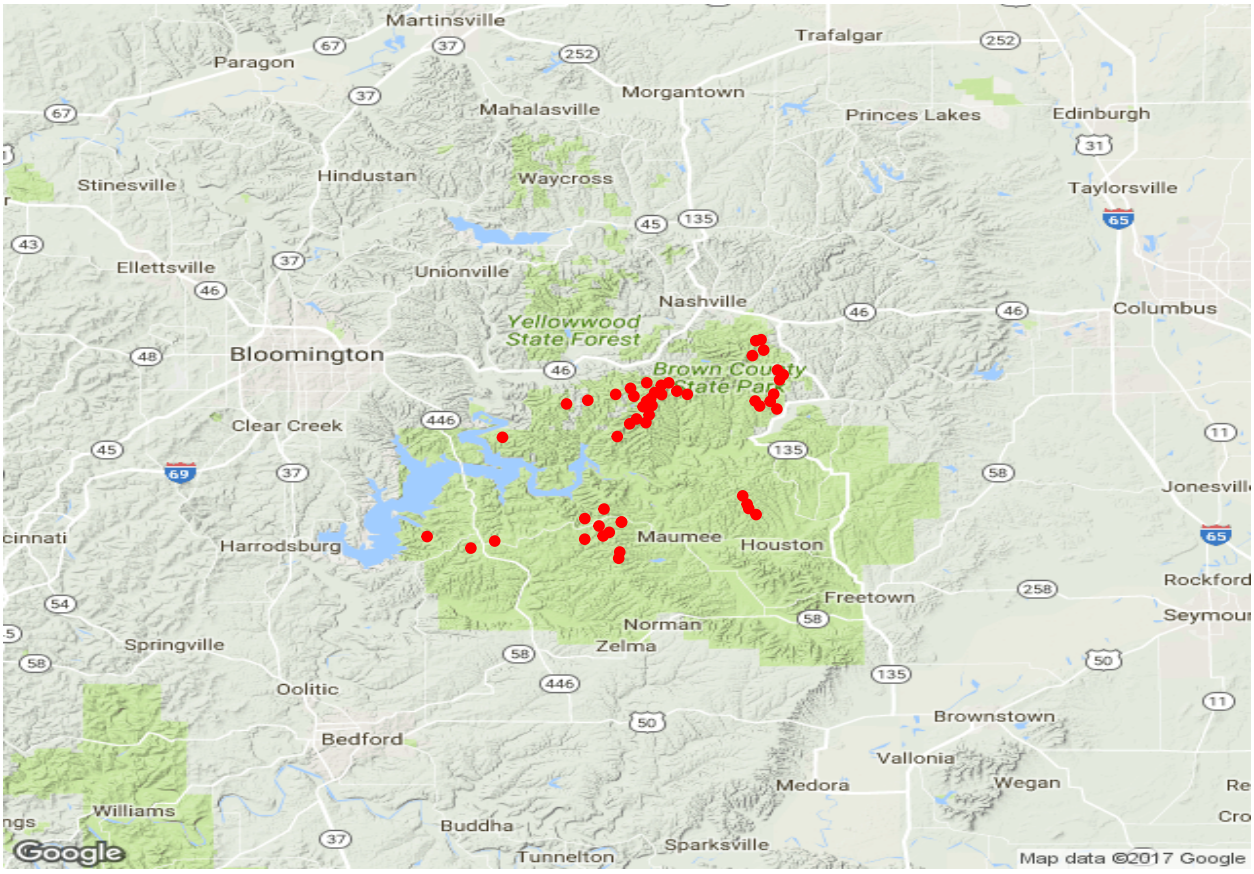
Answer 2b: The greatest richness value is 3659 in the YSF66 sample.

4) GENERATE MAPS

In the R code chunk below, do the following:

1. Using the example in the handout, visualize the spatial distribution of our samples with a basic map in RStudio using the `GetMap` function in the package `RgoogleMaps`. This map will be centered on Brown County, Indiana (39.1 latitude, -86.3 longitude).

```
lats <- as.numeric(Ponds[, 3]) # latitudes (north and south)
lons <- as.numeric(Ponds[, 4]) # longitudes (east and west)
newmap <- GetMap(center = c(39.1,-86.3), zoom = 10,
destfile = "PondsMap.png", maptype="terrain")
PlotOnStaticMap(newmap, zoom = 10, cex = 2, col = 'blue') # Plot map in RStudio
PlotOnStaticMap(newmap, lats, lons, cex = 1, pch = 20, col = 'red', add = TRUE)
```



Question 3: Briefly describe the geographical layout of our sites.

Answer 3: They tend to be grouped into a few clumps, although there are a few that are more spread out.

In the R code chunk below, do the following:

1. Using the example in the handout, build a map by combining lat-long data from our ponds with land cover data and data on the locations and shapes of surrounding water bodies.

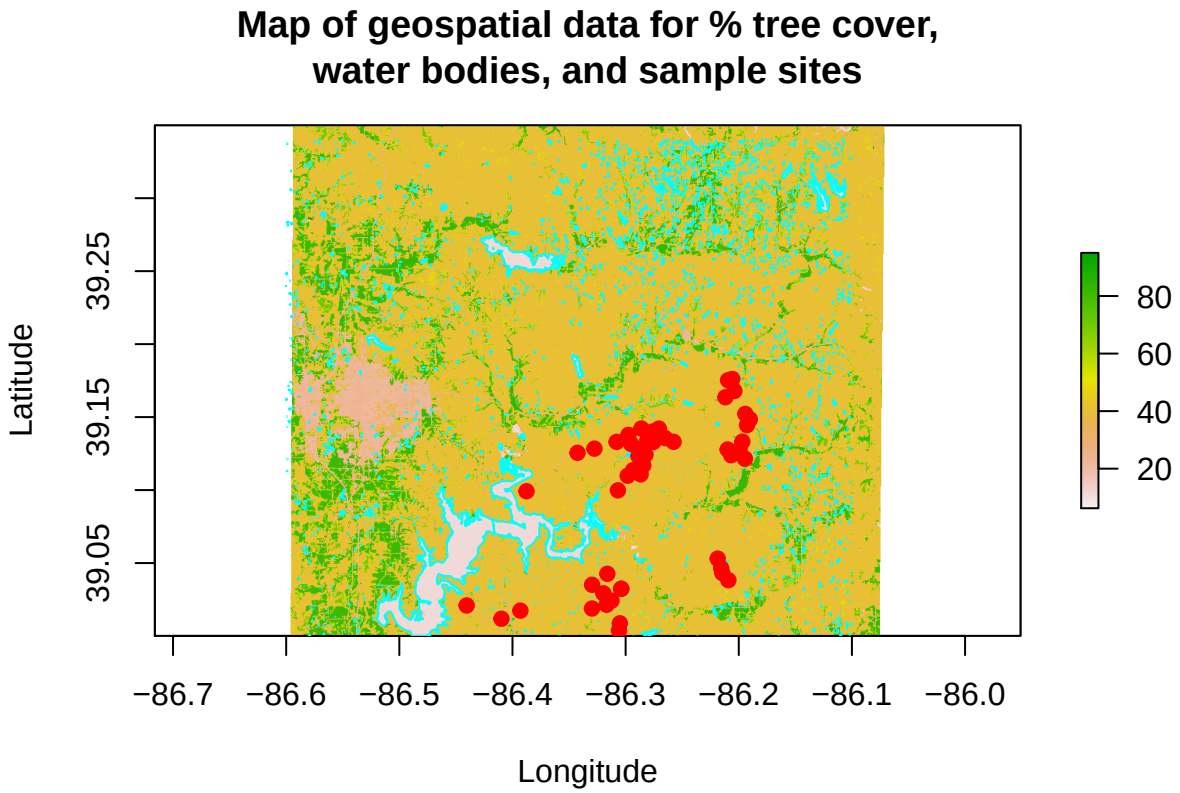
```
# 1. Import TreeCover.tif as a raster file.
Tree.Cover <- raster("TreeCover/TreeCover.tif") # import TreeCover.tif as a raster file.
```

```
## NOTE: rgdal::checkCRSArgs: no proj_defs.dat in PROJ.4 shared files
```

```
# 2. Plot the % tree cover data
plot(Tree.Cover, xlab = 'Longitude', ylab = 'Latitude',
     main = 'Map of geospatial data for % tree cover,\nwater bodies, and sample sites')
```

```
## NOTE: rgdal::checkCRSArgs: no proj_defs.dat in PROJ.4 shared files
```

```
# 3. Import water bodies as a shapefile.
Water.Bodies <- readShapeSpatial("water/water.shp") # import water bodies as a shapefile.

# 4. Plot the water bodies around our study area, i.e., Monroe County.
plot(Water.Bodies, border='cyan', axes = TRUE, add = TRUE)

# 5. Convert lat-long data for ponds to georeferenced points.
Refuge.Ponds <- SpatialPoints(cbind(lons, lats)) # convert lat-long data for ponds to georeferenced poi

# 6. Plot the refuge pond locations
plot(Refuge.Ponds, line='r', col='red', pch = 20, cex = 1.5, add = TRUE)
```



Question 4a: What are datums and projections?

> ***Answer 4a***: The datum is the model of Earth's shape. The projection is the way in which coordinates on the sphere are projected onto the 2-D surface.

5) UNDERSTANDING SPATIAL AUTOCORRELATION

Question 5: In your own words, explain the concept of spatial autocorrelation.

> ***Answer 5***: Spatial autocorrelation reveals whether variables are closer together in space which indicates positive autocorrelation, or randomly distributed in space, which indicates negative autocorrelation.

6) EXAMINING DISTANCE-DECAY

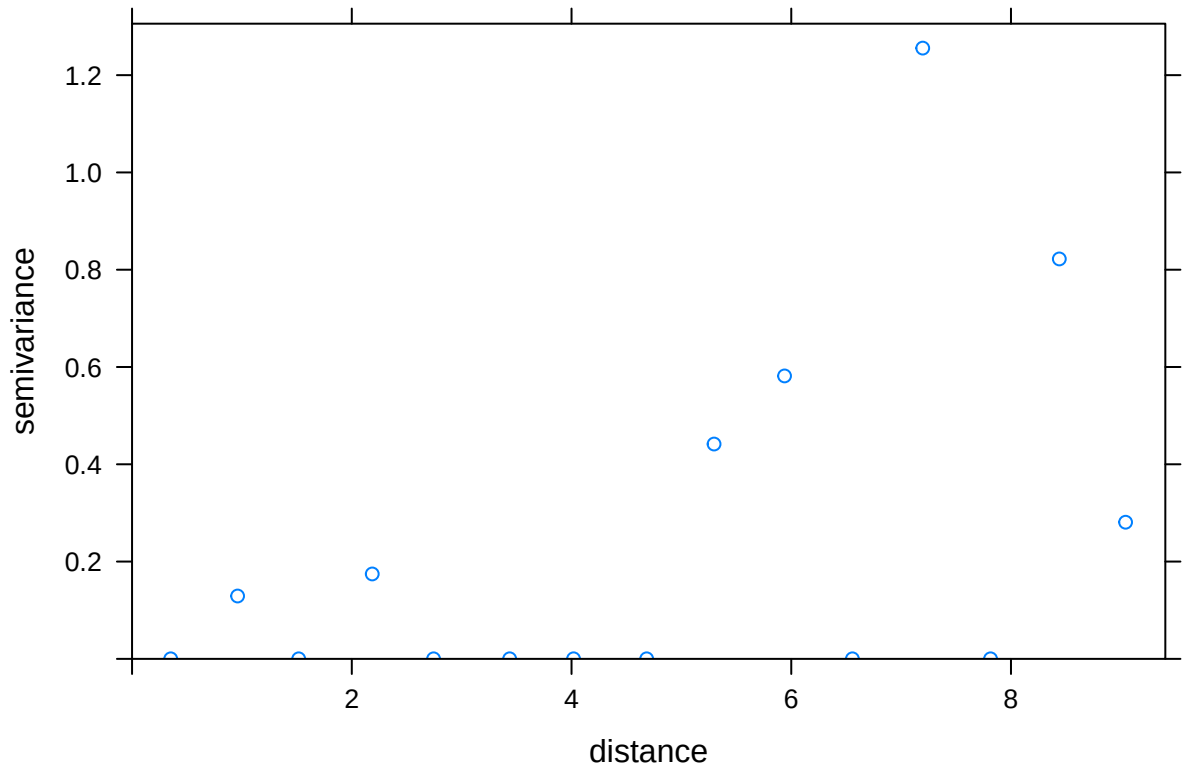
Question 6: In your own words, explain what a distance decay pattern is and what it reveals.

> ***Answer 6***: The distance decay pattern shows the relationship between increasing geographic distance and decreasing similarity between environments.

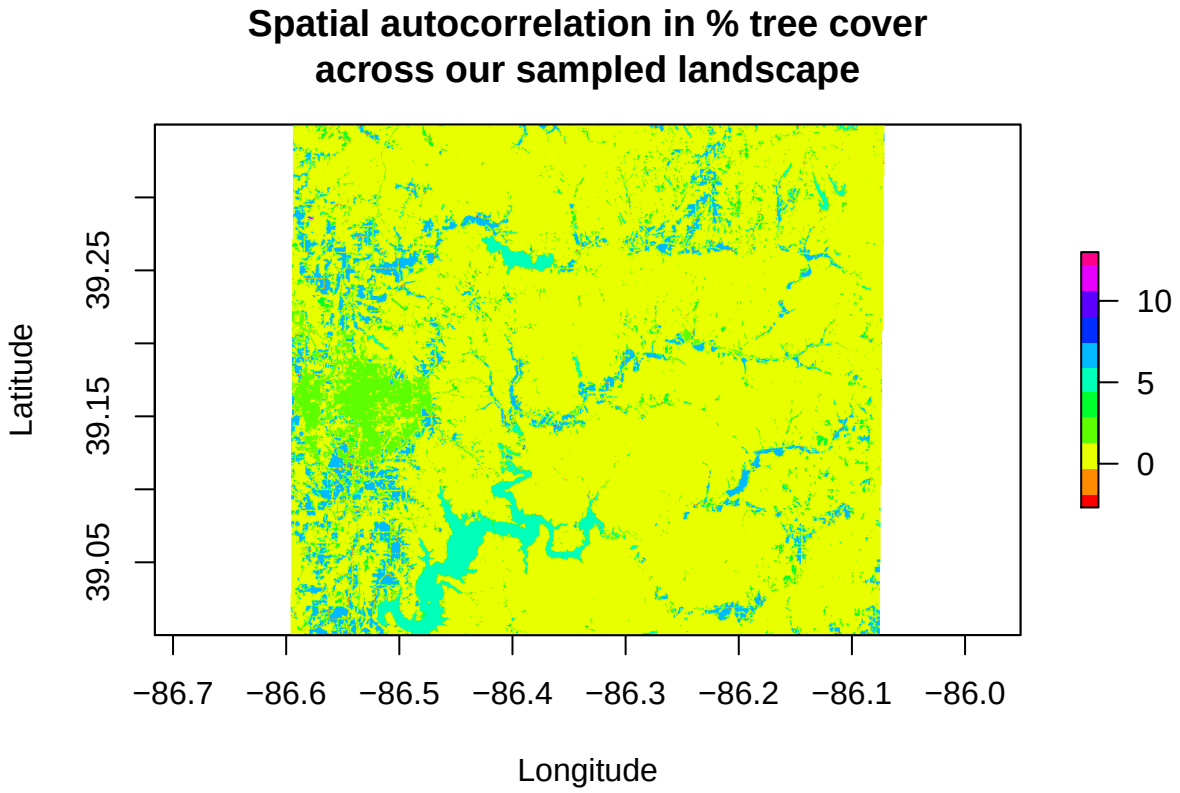
In the R code chunk below, do the following:

1. Generate the distance decay relationship for bacterial communities of our refuge ponds and for some of the environmental variables that were measured. Note: You will need to use some of the data transformations within the *semivariogram* section of the handout.

```
# Construct a new dataframe for coordinates
xy <- data.frame(env = Ponds$TDS, pond.name = Ponds$Sample_ID, lats = Ponds$lat, lons = Ponds$long)
coordinates(xy) <- ~lats+lons # Transform 'xy' into a spatial points dataframe

# Identify the current projection (i.e., lat-long) and datum (NAD83). In our case, the projection and d
proj4string(xy) <- CRS("+proj=longlat +datum=NAD83") # coordinate reference system (CRS)

# Then, transform the projection and data so we can get meaningful georeferenced distances. In this cas
UTM <- spTransform(xy, CRS("+proj=utm +zone=51 +ellps=WGS84")) # coordinate reference system (CRS)

UTM <- as.data.frame(UTM)

xy$lats_utm <- UTM[,2] # lattitude data according to UTM

xy$lons_utm <- UTM[,3] # longitude data according to UTM

#coordinates(xy) = ~lats_utm+lons_utm # Step required by the variogram function
# Examine the semivariance with regards to one of our environmental variables
env.vgm <- variogram(env~1, data=xy)
plot(env.vgm)
```

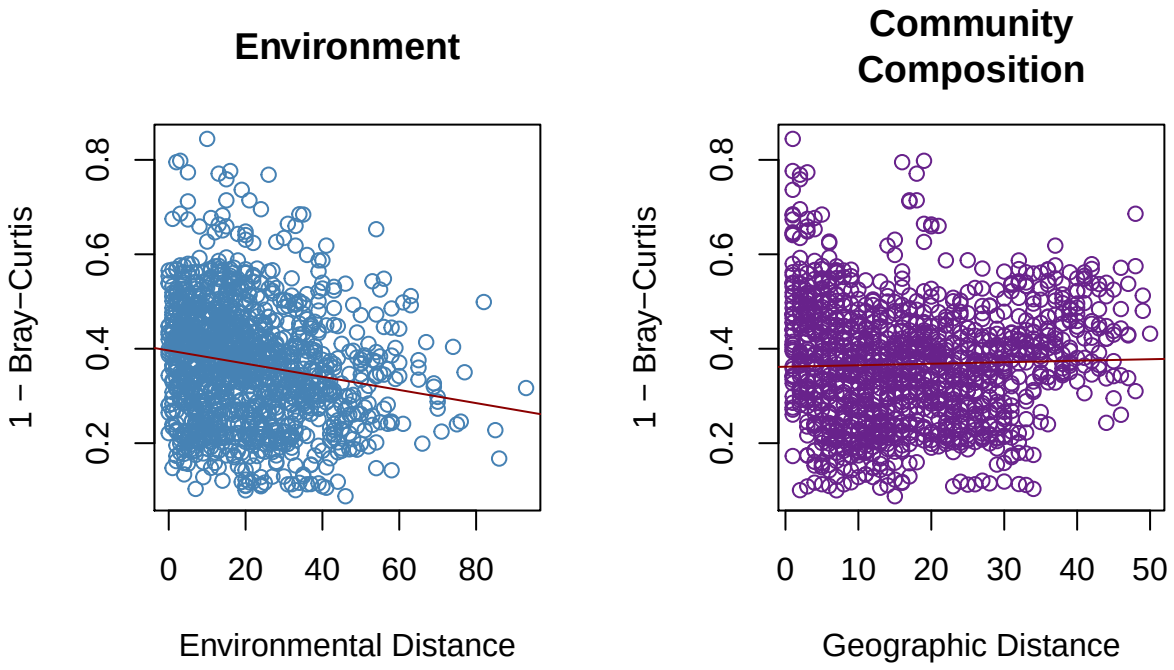



```
Moran(Tree.Cover)

TC.Moran <- MoranLocal(Tree.Cover)

plot(TC.Moran, xlab="Longitude", ylab="Latitude",
main="Spatial autocorrelation in % tree cover\nacross our sampled landscape",
col=rainbow(11, alpha=1))
```




```
# 1) Calculate Bray-Curtis similarity between plots using the `vegdist()` function
comm.dist <- 1 - vegdist(OTUs)

# 2) Assign UTM lattitude and longitude data to 'lats' and 'lons' variables
lats <- as.numeric(xy$lats_utm) # lattitude data
lons <- as.numeric(xy$lons_utm) # longitude data

# 3) Calculate geographic distance between plots and assign to the variable 'coord.dist'
coord.dist <- dist(as.matrix(lats, lons))

# 4) Transform environmental data to numeric type, and assign to variable 'x1'
x1 <- as.numeric(Ponds$"SpC")

# 5) Using the `vegdist()` function in `simba`, calculate the Euclidean distance between the plots for
env.dist <- vegdist(x1, "euclidean")

# 6) Transform all distance matrices into database format using the `liste()` function in `simba`:
comm.dist.ls <- liste(comm.dist, entry="comm")
env.dist.ls <- liste(env.dist, entry="env")
coord.dist.ls <- liste(coord.dist, entry="dist")

# 7) Create a data frame containing similarity of the environment and similarity of community.
df <- data.frame(coord.dist.ls, env.dist.ls[,3], comm.dist.ls[,3])

# 8) Attach the columns labels 'env' and 'struc' to the dataframe you just made.
names(df)[4:5] <- c("env", "struc")
attach(df)

# 9) After setting the plot parameters, plot the distance-decay relationships, with regression lines in
```

```
par(mfrow=c(1, 2), pty="s")
plot(env, struc, xlab="Environmental Distance", ylab="1 - Bray-Curtis",
main = "Environment", col='SteelBlue')
OLS <- lm(struc ~ env)
OLS # print regression results to the screen
abline(OLS, col="red4")
plot(dist, struc, xlab="Geographic Distance", ylab="1 - Bray-Curtis",
main="Community\nComposition", col='darkorchid4')
OLS <- lm(struc ~ dist)
OLS # print regression results to the screen
abline(OLS, col="red4")
```




```
# 10) Use `simba` to calculates the difference in slope or intercept of two regression lines
diffslope(env, struc, dist, struc) # a function in simba that calculates the difference in slope or int
```

Question 7: What can you conclude about community similarity with regars to environmental distance and geographic distance?

> ***Answer 7***: Species similarity decreased with both increasing environmental distance and increasing geographic distance. The relationship between similarity and environmental distance is stronger.

7) EXAMINING SPECIES SPATIAL ABUNDANCE DISTRIBUTIONS

Question 8: In your own words, explain the species spatial abundance distribution and what it reveals.

> ***Answer 8***: The species abundance distribution groups species by their abundance counts, and shows the frequency of species at each abundance level.

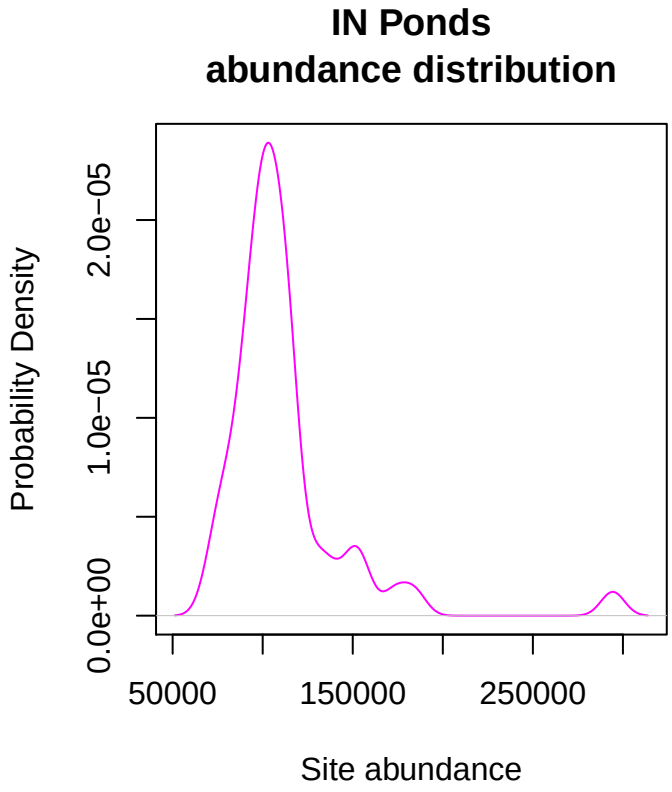
In the R code chunk below, do the following:

1. Define a function that will generate the SSAD for a given OTU.
2. Draw six OTUs at random from the IN ponds dataset and and plot their SSADs as kernel density curves. Use **while loops** and **if** statements to accomplish this.

```r
#how does total abundance 16S rRNA genes differ among our ponds
siteN <- rowSums(OTUs) # Abundances in each plot
siteN
```

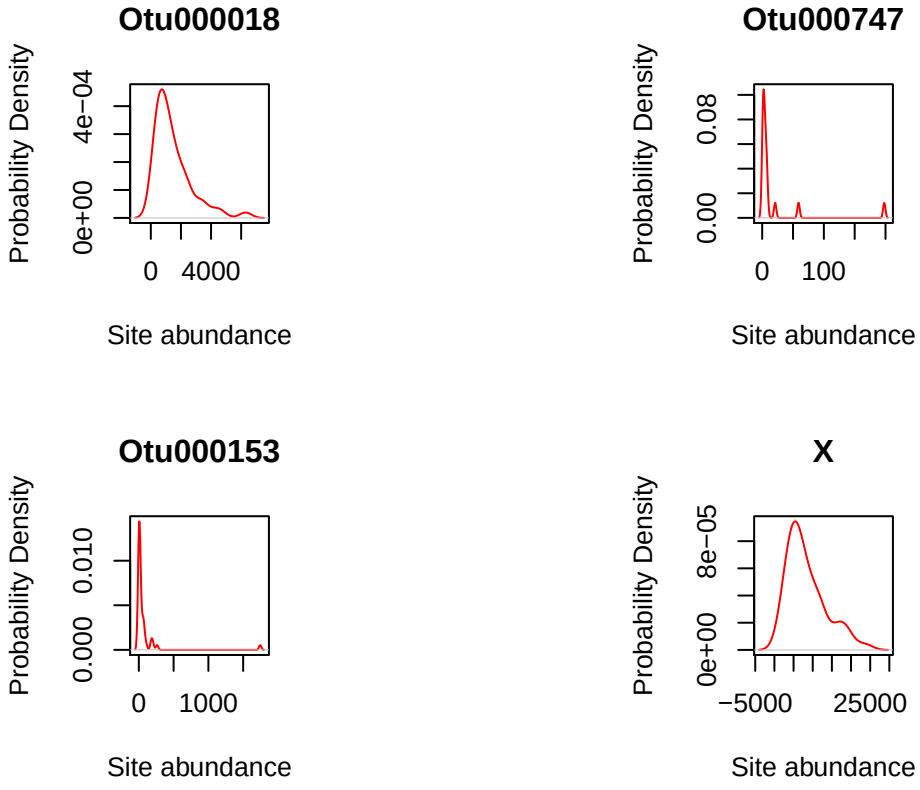
```
##  [1] 173194  91490 100595 100306 109561  94396 101579  90070 107097 114167
## [11] 101843 115108 151746  98495 109220 184225 149383  95476 108600 294346
## [21]  82508 108550  99190  78281 109876  91989 100153  85429 106245 117809
## [31] 101640  94125 115895 113251 132327 129936 156408 110889 102649  85770
## [41] 117904 139882 117278 101096  77124  70786  75233 107828 101166  93045
## [51]  89874
```

```r
# plot our data as an abundance distribution
par(mfrow=c(1, 1), pty="s")
plot(density(siteN), col = 'magenta', xlab='Site abundance',
ylab='Probability Density', main = 'IN Ponds\nabundance distribution')
```




```r
# 1. Define an SSAD function
ssad <- function(x){
  ad <- c(2, 2)
  ad <- OTUs[, otu]
  ad = as.vector(t(x = ad))
  ad = ad[ad > 0]
}

# 2. Set plot parameters
par(mfrow=c(2, 2))

# 3. Declare a counter variable
ct <- 0
```

```
# 4. Write a while loop to plot the SSADs of six species chosen at random
while (ct < 4){ # While the ct variable is less than 4, do ...
  otu <- sample(1:length(OTUs), 1) # choose 1 random OTU (i.e., a random column of the site-by-species
  ad <- ssad(otu) # find the OTU's SSAD
  if (length(ad) > 10 & sum(ad > 100)){ # if the species is present in at least 10 sites and has an ove
    ct <- ct + 1
    plot(density(ad), col = 'red', xlab='Site abundance',
         ylab='Probability Density', main = otu.names[otu])
    }
}
```



8) UNDERSTANDING SPATIAL SCALE

Many patterns of biodiversity relate to spatial scale.

Question 9: List, describe, and give examples of the two main aspects of spatial scale

> ***Answer 9***: The two main aspects of spatial scale are extent and grain. Extent is the largest distance considered in your data, and grain is the smallest unit that is measured. Extent usually refers to the size of the area you are studying, so if your study area is 10 ha, your extent is 10 ha. Grain could be the size of each plot being measured, so if you are measuring 1 ha subplots your grain would be 1 ha.

9) CONSTRUCTING THE SPECIES-AREA RELATIONSHIP

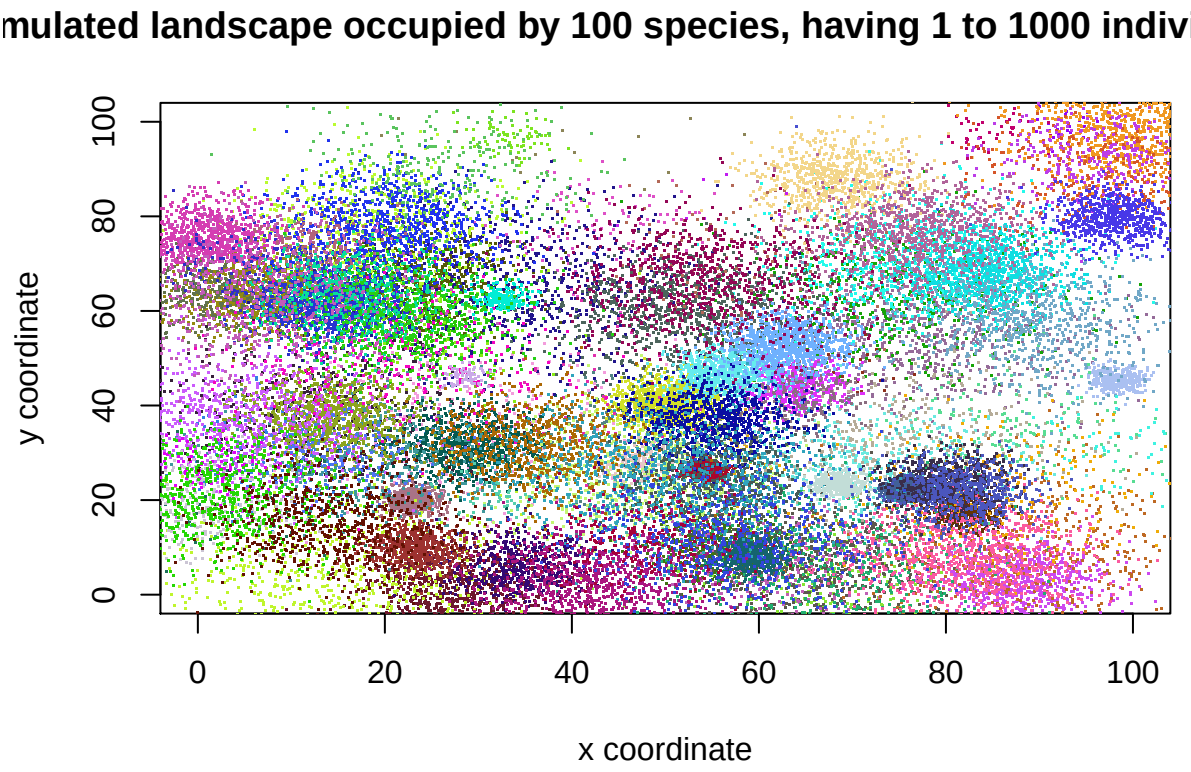
Question 10: In your own words, describe the species-area relationship.

Answer 10: The species-area relationship shows how fast we find new species as we sample more area.

In the R code chunk below, provide the code to:

1. Simulate the spatial distribution of a community with 100 species, letting each species have between 1 and 1,000 individuals.

```
# 1. Declare variables to hold simulated community and species information
community <- c() # an initiall empty community
species <- c() # with zero species

# 2. Populate the simulated landscape
# initiate the plot, i.e., landscape
plot(0, 0, col='white', xlim = c(0, 100), ylim = c(0, 100), xlab='x coordinate', ylab='y coordinate', ma

while (length(community) < 100){ # while the community has less than 100 species
  # choose the mean, standard deviation, and species color at random
  std <- runif(1, 1, 10) # random sample from a uniform distribution
  ab <- sample(1000, 1) # random number between 1 and 1000
  x <- rnorm(ab, mean = runif(1, 0, 100), sd = std) # 1000 random numbers from a Normal distribution
  y <- rnorm(ab, mean = runif(1, 0, 100), sd = std) # 1000 random numbers from a Normal distribution
  color <- c(rgb(runif(1),runif(1),runif(1))) # Let each species have a randomly chosen color
  points(x, y, pch=".", col=color) # Add points to a plot
  species <- list(x, y, color) # The species color, x-coords, and y-coords
  community[[length(community)+1]] <- species # Add the species info to the community
}
```



While consult the handout for assistance, in the R chunk below, provide the code to:

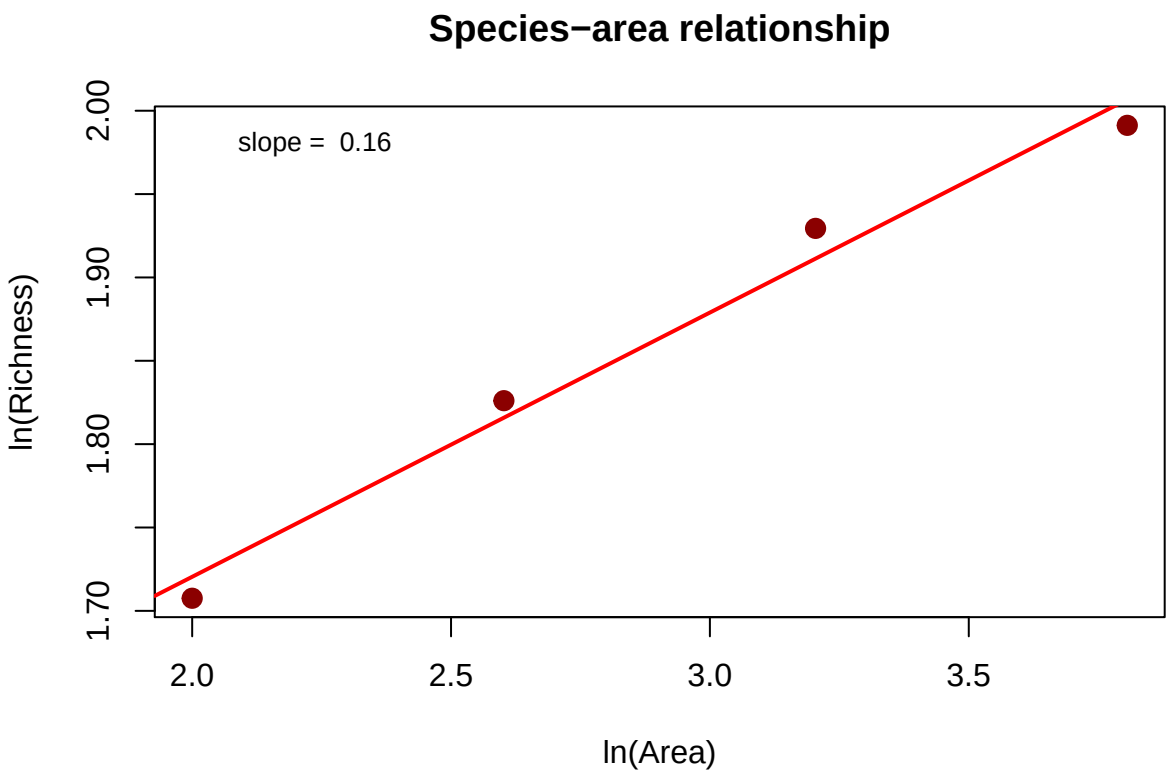
1. Use a nested design to examine the SAR of our simulated community.
2. Plot the SAR and regression line.

```
# 1. Declare the spatial extent and lists to hold species richness and area data
lim <- 10 # smallest spatial extent. This also equals the spatial grain.
S.list <- c() # holds the number of species
A.list <- c() # holds the spatial scales

# 2. Construct a 'while' loop and 'for' loop combination to quantify the numbers of species for progres
while (lim <= 100){ # while the spatial extent is less than or equal to 100...
  S <- 0 # initiate richness at zero
  for (sp in community){ # for each species in the community
    xs <- sp[[1]] # assign the x coordinates
    ys <- sp[[2]] # assign the y coordinates
    sp.name <- sp[[3]] # assign the species name
    xy.coords <- cbind(xs, ys) # combine the columns for x and y coordinates
    for (xy in xy.coords){ # for each pair of xy coordinates
      if (max(xy) <= lim){ # if the individual is within our current spatial extent...
        S <- S + 1 # then the species occurs there
        break # break out of the last for loop because we now know the species occurs inside our sampli
        }
      }
    }
    S.list <- c(S.list, log10(S))
    A.list <- c(A.list, log10(lim^2))
    lim <- lim * 2 # increase the extent multiplicatively

}
```

In the R code chunk below, provide the code to:

1. Plot the richness and area data as a scatter plot.
2. Calculate and plot the regression line
3. Add a legend for the z-value (i.e., slope of the SAR)

```
results <- lm(S.list ~ A.list)
plot(A.list, S.list, col="dark red", pch=20, cex=2, main="Species-area relationship", xlab='ln(Area)', y

abline(results, col="red", lwd=2)
int <- round(results[[1]][[1]],2)
z <- round(results[[1]][[2]],2)
legend(x=2, y=2, paste(c('slope = ', z), collapse = " "), cex=0.8, box.lty=0)
```



Question 10a: Describe how richness relates to area in our simulated data by interpreting the slope of the SAR.

> ***Answer 10a***: Since the slope of our SAR was .22, this means that increasing the sampling area from 10 to 100 would increase the richness from about 5 to about 13.

Question 10b: What does the y-intercept of the SAR represent?

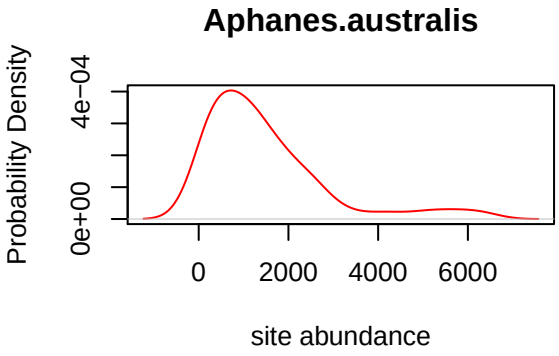
> ***Answer 10b***: It represents the scale at which the (un-log-transformed) SAR curve is stretched vertically, indicating a scalar increase or decrease on the rate new species are discovered with increasing sampling area.

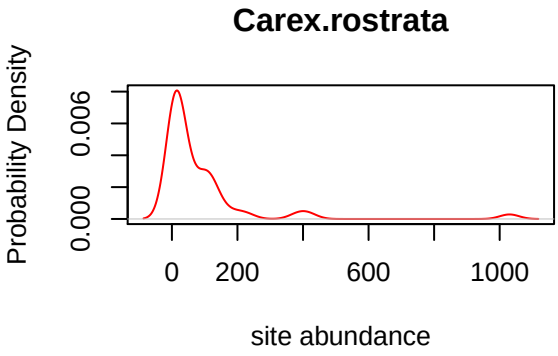
SYNTHESIS

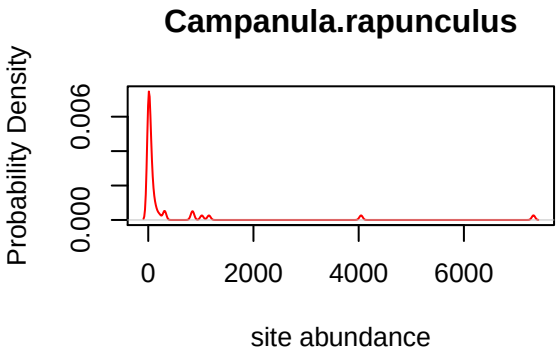
Load the dataset you are using for your project. Plot and discuss either the geogrpahic Distance-Decay relationship, the SSADs for at least four species, or any variant of the SAR (e.g., random accumulation of plots or areas, accumulation of contiguous plots or areas, nested design).

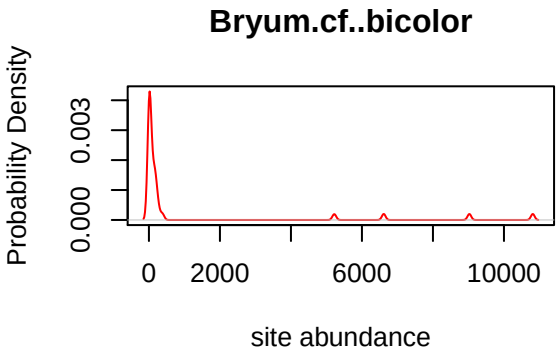
```
# i will show the ssad's for 4 species in our data set

#load and transform data
speciesdata <- read.table("/Users/flopsei/GitHub/QB2017_Partee/speciesdata_clean.csv", sep = ",", heade
envdata <- read.table("/Users/flopsei/GitHub/QB2017_Partee/environmentaldata.csv", sep = ",", header = 
envdata <- envdata[1:153,]

#data frames
speciesdatafr <- speciesdata[,5:dim(speciesdata)[2]]
row.names(speciesdatafr) <- speciesdata$Site.number
speciesdatafr <- speciesdatafr[order(speciesdata$Site.number),]

envdatafr <- envdata[,8:24]
```

```
row.names(envdatafr) <- envdata$Site.no.
envdatafr <- envdatafr[order(envdata$Site.no.),]

#as matrix
speciesdatamat <- as.matrix(speciesdatafr)
envdatamat <- as.matrix(envdatafr)

speciesnames <- colnames(speciesdatamat)

#to find 4 random species within our data set

par(mfrow = c(2,2))

ct <- 0
while (ct < 4) {
  otu <- sample(1:dim(speciesdatamat)[2], 1) #choose a random species
  ad <- ssad(otu)
  if (length(ad) > 10 & sum(ad > 100)) { #if the species is in at least 10 sites with more than 100 ind
    ct <- ct + 1
    plot(density(ad), col = 'red', xlab = 'site abundance', ylab = 'Probability Density', main = specie
  }
}
```










```
# The SSADs for the 4 randomly chosen species in our data set follow the expected
# abundance distribution for species, where most sites that contain a given species
# will have very low abundances for that species, and only a few sites contain high
# abundances of species (at least this time when I run it, it will likely pick
# different species when I knit this to html).
```