# Assignment: Spatial Diversity

*Venus Kuo; Z620: Quantitative Biodiversity, Indiana University*

*06 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

```r
rm(list = ls())
getwd()
setwd("C:/Users/Venus/Github/QB2017_Kuo/Week3-Beta/")
#setwd("/Users/vkuo/GitHub/QB2017_Kuo/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`

```
# Require or install packages #
package.list <- c('vegan', 'sp', 'gstat', 'raster', 'RgoogleMaps', 'maptools', 'rgdal', 'simba', 'gplots
for (package in package.list) {
  if (!require(package, character.only=T, quietly=T)) {
    install.packages(package)
    library(package, character.only=T)
} }
```

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

> ***Answer 1***:
> simba is used to calculate similarity measurements for binary data. sp is a package providing classes and methods for spatial data such as points, lines, polygons, and grids. rgdal allows bindings 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)
dim(OTUs)
S.obs <- function(x=""){
  rowSums(x>0) *1
}
max(S.obs(OTUs))
```

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

> ***Answer 2a***: There are 51 sites and 16383 species.

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

> ***Answer 2b***: 3659 species

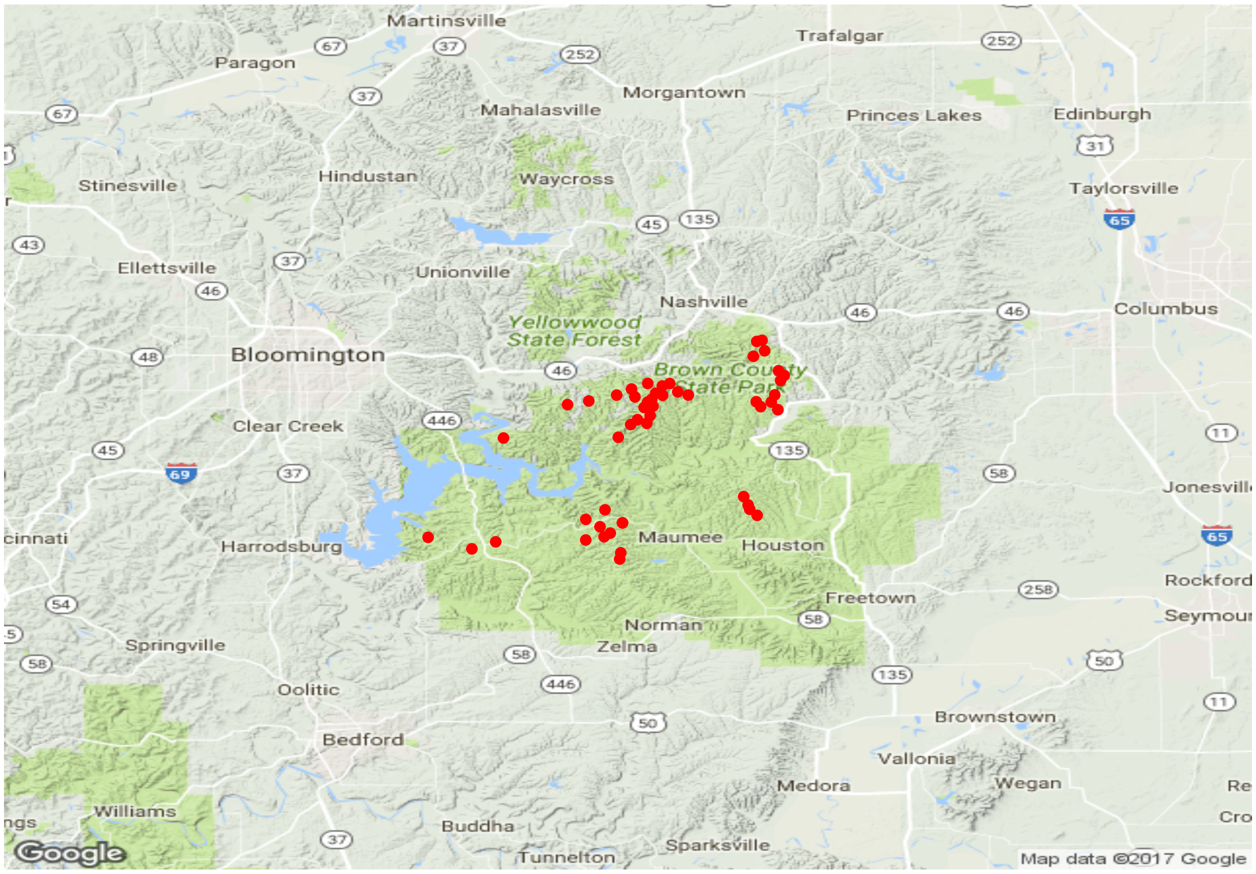
## 4) GENERATE MAPS

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])
lons <- as.numeric(Ponds[, 4])

newmap <- GetMap(center = c(39.1,-86.3), zoom = 10,
                  destfile = "PondsMap.png", maptype="terrain")
PlotOnStaticMap(newmap, zoom = 10, cex = 2, col = 'blue')
PlotOnStaticMap(newmap, lats, lons, cex = 1, pch = 20, col = 'red', add = TRUE)
```



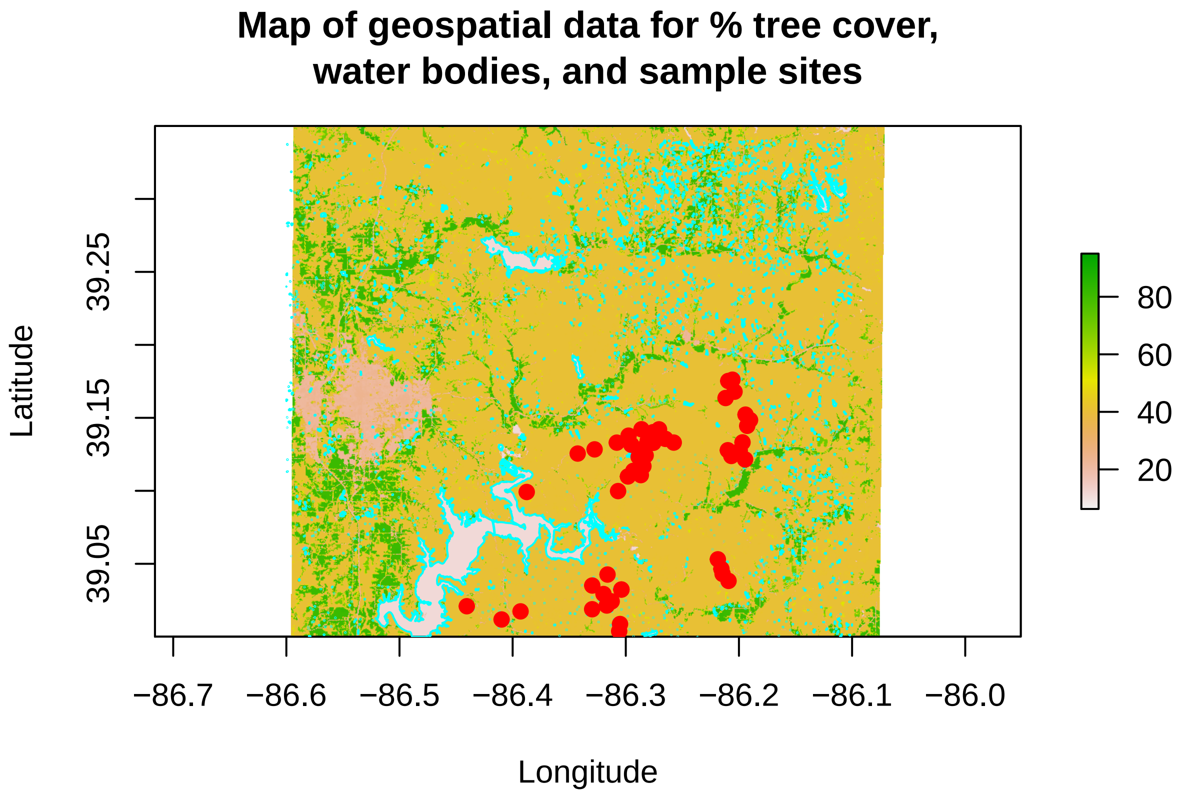
***Question 3***: Briefly describe the geographical layout of our sites.

> ***Answer 3***: The get map function retrieved and displayed the map of Brown County State Park form google maps. All points are located within Brown County State park but clustered into 5 or 6 groups around Lake Monroe, with some sites closer and some farther away.

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")
# 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')
# 3. Import water bodies as a shapefile.
Water.Bodies <- readShapeSpatial("water/water.shp")
# 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))
# 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***: Datums are model for Earth's shape (elicpse) and projections are the way in which coordinates on a sphere are projected onto a 2-D surface (a map).

## 5) UNDERSTANDING SPATIAL AUTOCORRELATION

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

> ***Answer 5***: "Related things tend to be closer to each other than to distant things". If something is more related and closer in space then that is positive autocorrelation but if related things are farther apart than expected then that is negative autocorrelation.

## 6) EXAMINING DISTANCE-DECAY

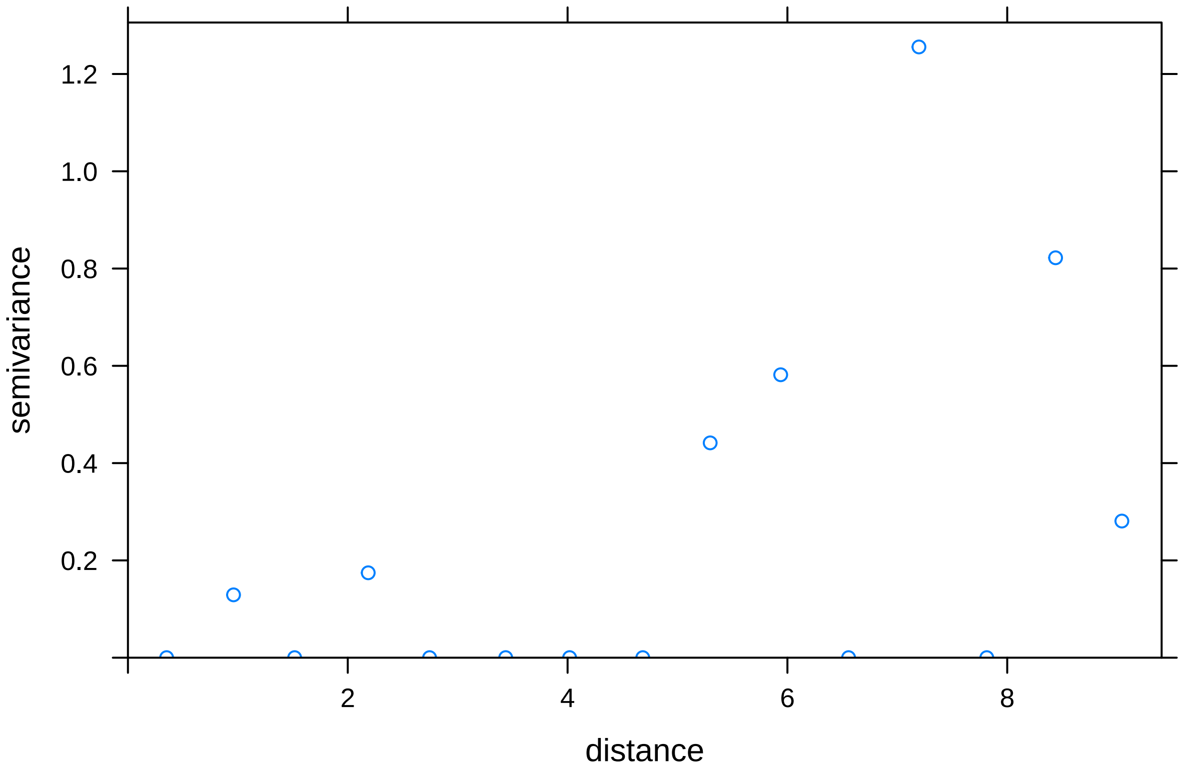
***Question 6***: In your own words, explain what a distance decay pattern is and what it reveals.

> ***Answer 6***: The distance decay pattern is the pattern of spatial autocorrelation (species and enviromental similarity) across geographic distance.

In the R code chunk below, do the following:

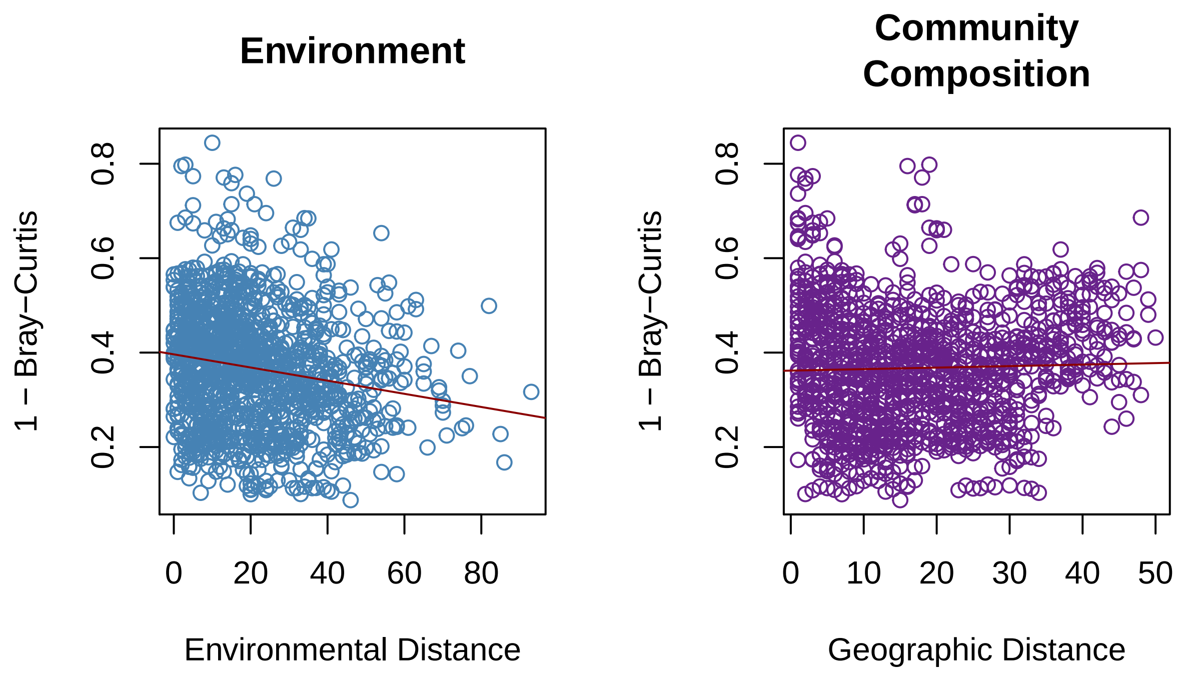
1. Generate the distance decay relationship for bacterial communities of our refuge ponds and for some of the environmental variables that were measured. Note: You will need to use some of the data transformations within the *semivariogram* section of the handout.

```r
# Transforming data set: Construct new dataframe
xy <- data.frame(env = Ponds$TDS, pond.name = Ponds$Sample_ID, lats = Ponds$lat, lons = Ponds$long)
coordinates(xy) <- ~lats+lons
proj4string(xy) <- CRS("+proj=longlat +datum=NAD83")
UTM <- spTransform(xy, CRS("+proj=utm +zone=51 +ellps=WGS84"))
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)
```



```r
# 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)) # geographical distance between plots
# 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")
```

```r
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")
# 9) After setting the plot parameters, plot the distance-decay relationships, with regression lines in
attach(df)
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
abline(OLS, col="red4")
plot(dist, struc, xlab="Geographic Distance", ylab="1 - Bray-Curtis",
     main="Community\nComposition", col='darkorchid4')
OLS <- lm(struc ~ dist)
OLS
abline(OLS, col="red4")
```



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

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

***Answer 7***: Enviromental sites become more dissimilar as they increase in geographic distance.

Species community composition seems to stay the same or increase slightly in similarity with increase geographic distance.

## 7) EXAMINING SPECIES SPATIAL ABUNDANCE DISTRIBUTIONS

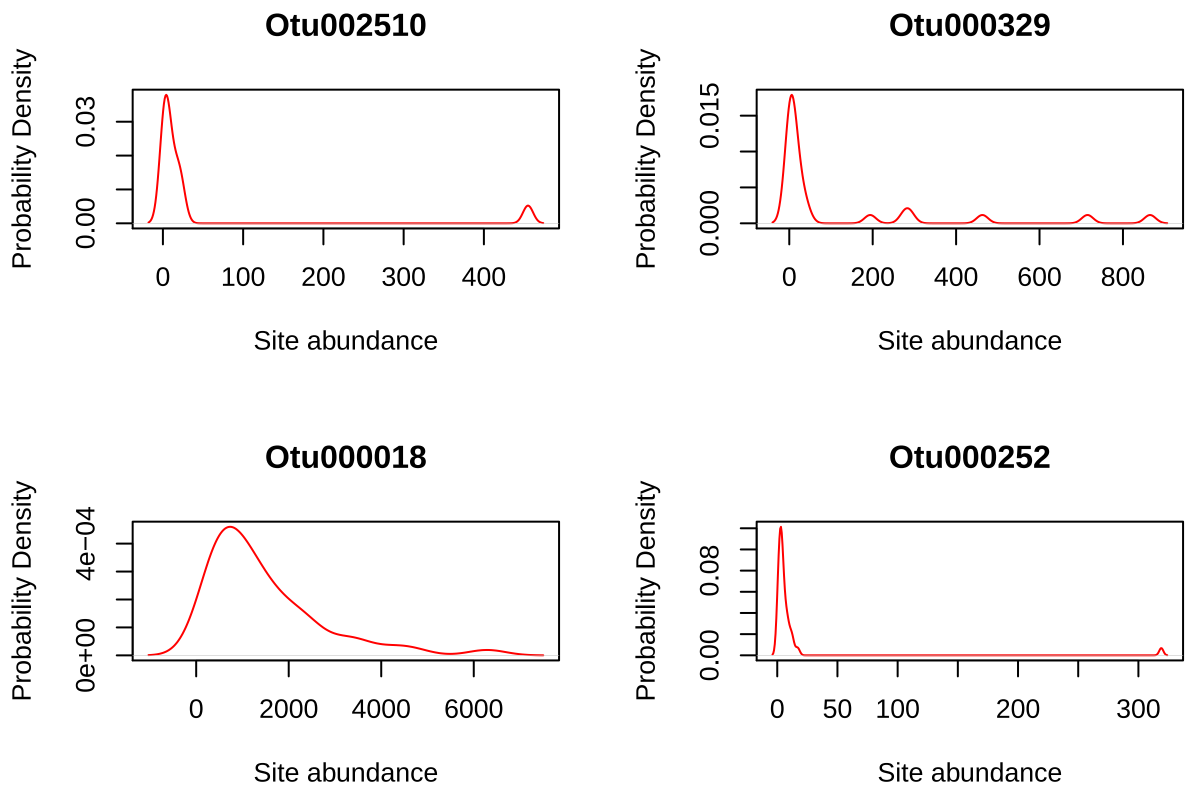
***Question 8***: In your own words, explain the species spatial abundance distribution and what it reveals.

> ***Answer 8***: SSAD is shows a pattern of commonness and rarity distributed across sites. The SSAD will generally show that a few sites will have many individuals while many sites will have few individuals, moreover a few species will be abundant while many species will appear only a few times.

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
# 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 # a counter variable
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 sit
  ad <- ssad(otu) # find the OTU's SSAD
# 4. Write a while loop to plot the SSADs of six species chosen at random
if (length(ad) > 10 & sum(ad > 100)) {
  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***: Extent and Grain are two main components of spatial scale. Extent is the greatest distance considered in a study, while grain is the smallest scale or unit measured in the same study. For example, what is the distance of the UK shore? . . . the answer depends on the grain size used in the measurement. Another example would be deciding how individuals of a population is aggregated, up-close the answer would be randomly aggregated, far away the answer might be that individuals aggregate into one spot.

## 9) CONSTRUCTING THE SPECIES-AREA RELATIONSHIP

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

> ***Answer 10***: The SAR is a scaling law that is used to predict species diversity across scales, because generally the more sites you sample the more species you encounter up to a certain point.

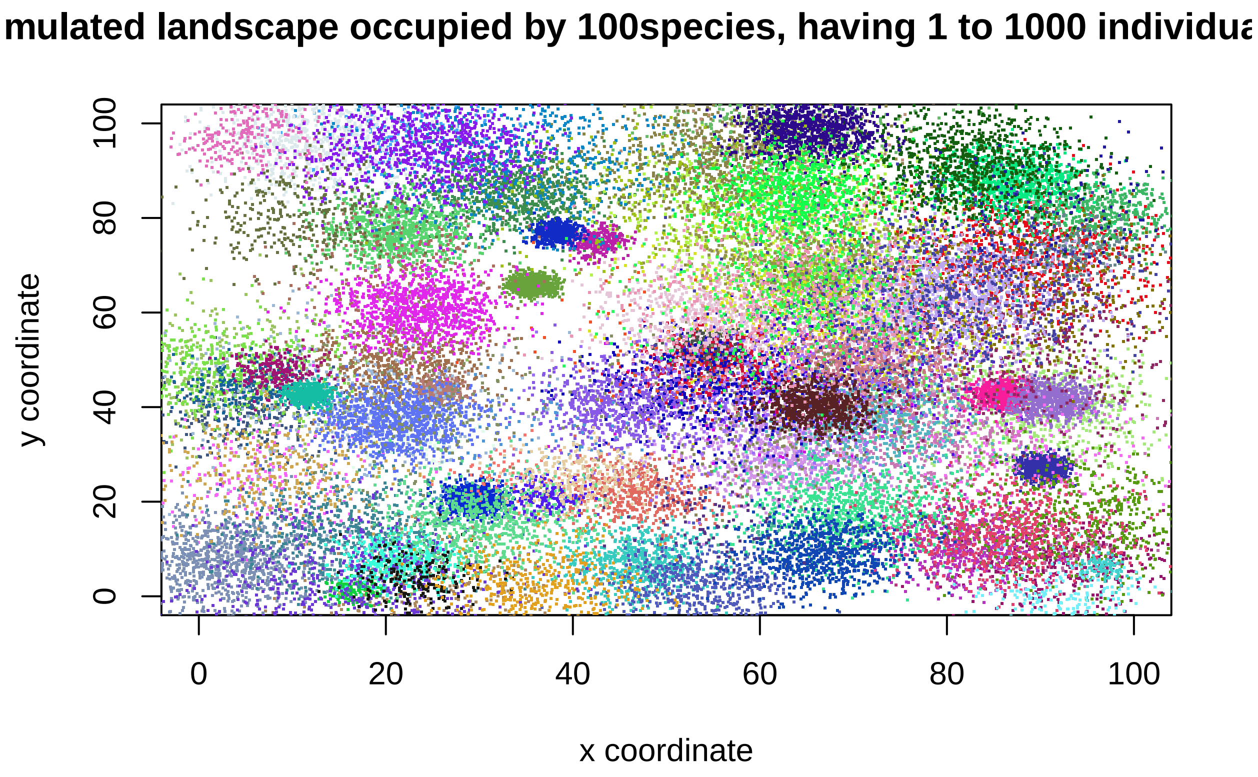
In the R code chunk below, provide the code to:

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()
```

```r
species <- c()
# 2. Populate the simulated landscape
plot(0, 0, col='white', xlim = c(0, 100), ylim = c(0, 100),
     xlab='x coordinate', ylab='y coordinate',
     main='A simulated landscape occupied by 100species, having 1 to 1000 individuals each.')

while (length(community) < 100){
  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.

```r
# 1. Declare the spatial extent and lists to hold species richness and area data
lim <- 10
S.list <- c()
A.list <- c()
# 2. Construct a 'while' loop and 'for' loop combination to quantify the numbers of species for progres
```

```
while(lim <= 100) {
  S <- 0
  for (sp in community) {
    xs <- sp[[1]]
    ys <- sp[[2]]
    sp.name <- sp[[3]]
    xy.coords <- cbind(xs, ys)
    for (xy in xy.coords) {
      if (max(xy) <= lim) {
        S <- S + 1
          break
      }
    }
  }
  S.list <- c(S.list, log10(S))
  A.list <- c(A.list, log10(lim^2))
  lim <- lim*2
}
# 3. Be sure to log10-transform the richness and area data
```

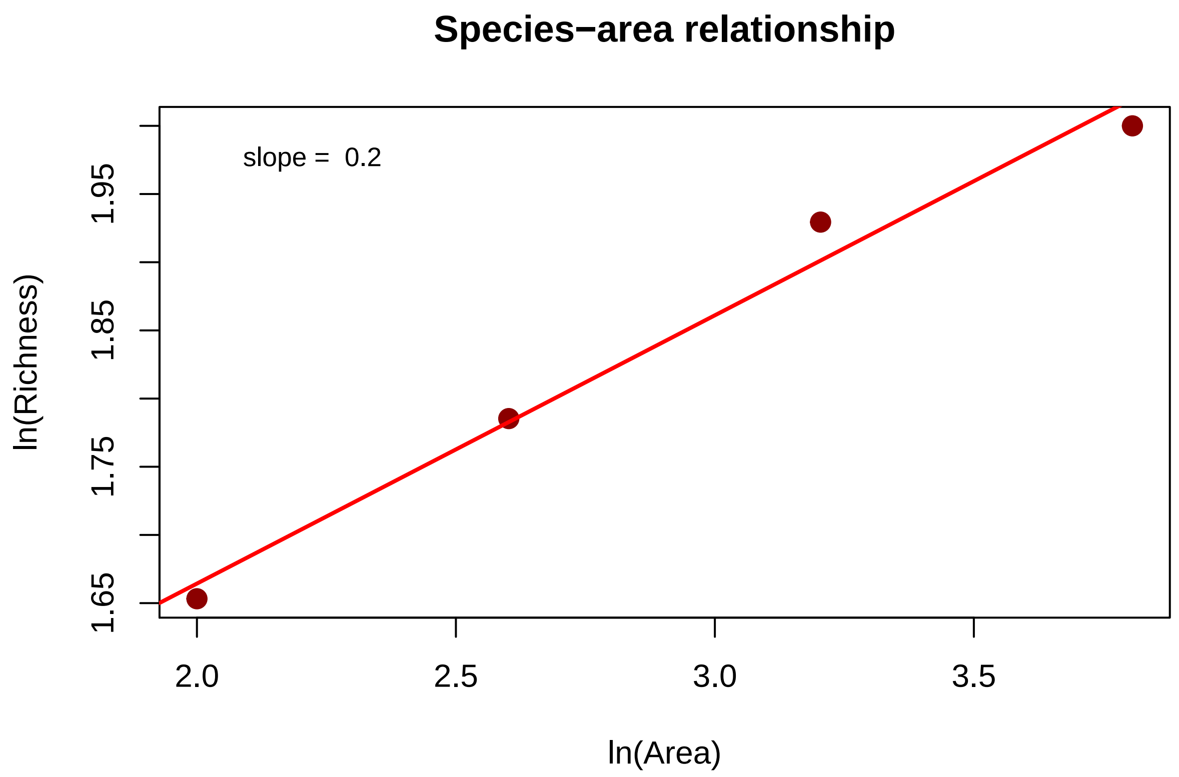
In the R code chunk below, provide the code to:

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)", ylab="ln(Richness)")

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***: The slope is 0.22, this means that as area increases, so does the species richness.

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

> ***Answer 10b***: The y-axis represents the log normal of species number (richness) so the y axis would be the alpha diversity of a site or a close cluster of sites?

## 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).

```r
# Set working directory #
rm(list = ls())
setwd("C:/Users/Venus/Github/QB2017BenavidezKuo/data")

# Load packages #
require("vegan")
require("tidyr")
```

```
## Loading required package: tidyr
```

```
##
## Attaching package: 'tidyr'
## The following object is masked from 'package:raster':
##
##     extract
```

```r
require(dplyr)
```

```
## Loading required package: dplyr

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:rgeos':
##
##     intersect, setdiff, union

## The following objects are masked from 'package:raster':
##
##     intersect, select, union

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

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

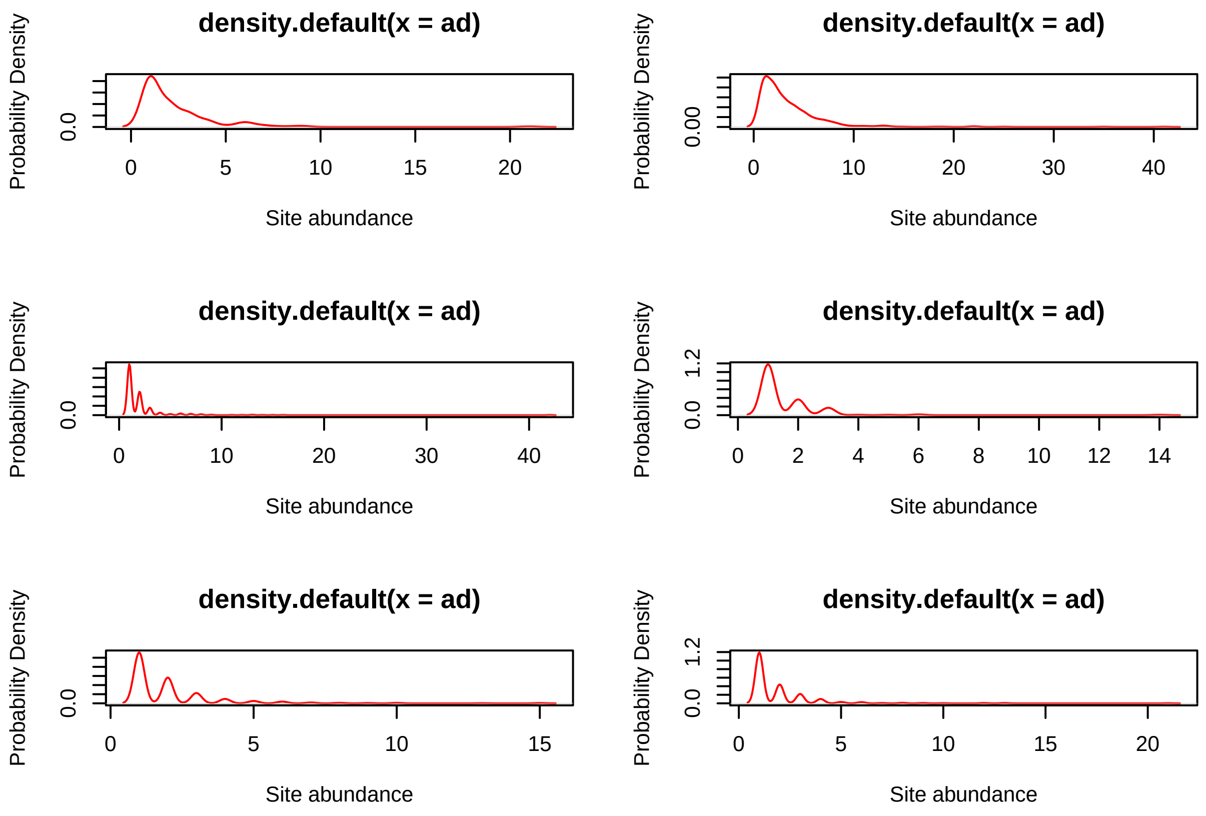
```r
# Load BCI datasets #
BCI2010 <- read.delim("BCI2010.txt", header=T)
BCI2005 <- read.delim("BCI2005.txt", header=T)
BCI2000 <- read.delim("BCI2000.txt", header=T)
BCI1995 <- read.delim("BCI1995.txt", header=T)
BCI1990 <- read.delim("BCI1990.txt", header=T)
BCI1985 <- read.delim("BCI1985.txt", header=T)
BCI1982 <- read.delim("BCI1982.txt", header=T)

# Formatting datasets into SbyS matrcies #
BCI.2010.SbyS <- group_by(BCI2010, Quadrat) %>% count(Latin) %>% spread(key=Latin, value=n , fill=0)
BCI.2005.SbyS <- group_by(BCI2005, Quadrat) %>% count(Latin) %>% spread(key=Latin, value=n , fill=0)
BCI.2000.SbyS <- group_by(BCI2000, Quadrat) %>% count(Latin) %>% spread(key=Latin, value=n , fill=0)
BCI.1995.SbyS <- group_by(BCI1995, Quadrat) %>% count(Latin) %>% spread(key=Latin, value=n , fill=0)
BCI.1990.SbyS <- group_by(BCI1990, Quadrat) %>% count(Latin) %>% spread(key=Latin, value=n , fill=0)
BCI.1985.SbyS <- group_by(BCI1985, Quadrat) %>% count(Latin) %>% spread(key=Latin, value=n , fill=0)
BCI.1982.SbyS <- group_by(BCI1982, Quadrat) %>% count(Latin) %>% spread(key=Latin, value=n , fill=0)

# Removing quadrat row
BCI.1982.SBS=BCI.1982.SbyS[,2:307]
BCI.1985.SBS=BCI.1985.SbyS[,2:308]
BCI.1990.SBS=BCI.1990.SbyS[,2:306]
BCI.1995.SBS=BCI.1995.SbyS[,2:304]
BCI.2000.SBS=BCI.2000.SbyS[,2:303]
BCI.2005.SBS=BCI.2005.SbyS[,2:300]
BCI.2010.SBS=BCI.2010.SbyS[,2:298]

# SADD for 4 species #
# 1. Define an SSAD function
ssad <- function(x){
  ad <- c(2, 2)
  ad <- BCI.1982.SbyS[, species]
  ad = as.vector(t(x = ad))
  ad = ad[ad > 0]
}
# 2. Set plot parameters
par(mfrow=c(3, 2))

# 3. Declare a counter variable
ct <- 0 # a counter variable
while (ct < 6){
  species <- sample(1:length(BCI.1982.SbyS), 1)
  ad <- ssad(species)
# 4. Write a while loop to plot the SSADs of six species chosen at random
if (length(ad) > 2 & sum(ad > 10)) {
  ct <- ct + 1
  plot(density(ad), col = 'red' , xlab = 'Site abundance' ,
       ylab = 'Probability Density')
  }
}
```



Discusion: I had a hard time figuring out how to add the species name to the top of the graphs. But it appears that from the six tree species selected from the 1982 BCI census dataset shows that generally, species will typically be abundnant in only a fraction of the sites that it occurs (somewhere between 1-5 depending on the species).