Geographical Ecology

Z620: Quantitative Biodiversity, Indiana University

February 13, 2015

OVERVIEW

In this exercise, we will add a geographical context to alpha (α) and beta (β) diversity. We will explore core concepts like spatial autocorrelation, aggregation, and scale dependence. We will also make use of Geographical Information Systems (GIS) to map and explore environmental, spatial, and biodiversity data.

After completing this exercise you will be able to:
1. Quantify effects of geographic distance on environmental and ecological similarity.
2. Examine the extent to which patterns of diversity depend on spatial scale.
3. Use control structures such as `loops` to control how R operates on variables.
4. Find and use geospatial data and conduct basic GIS operations in R.

1.) SETUP

A. Retrieve and Set Your Working Directory

```
rm(list=ls())
getwd()
setwd("~/GitHub/QuantitativeBiodiversity/Assignments/GeographicalEcology")
```

B. Load Packages

We will use the `vegan` package for biodiversity estimators and related functions. We will also use packages developed in R for geographical information systems (GIS). These will allow us to generate maps, project diversity data onto environmental layers, and conduct geographical analyses. Be sure to run **install.packages(PackageName)**, if not previously installed, where PackageName is the name of the package you want installed. Or, run **install.packages(PackageName, type="source", dependencies=TRUE)**, if the previous command doesn't work.

```
require("vegan")

require("sp") # classes and methods for spatial data: points, lines, polygons, grids
require("rgdal")  # Geospatial Data Abstraction Library
require("raster") # Methods to create a RasterLayer object

require("RgoogleMaps") # For querying the Google server for static maps.
require("maptools")    # Tools for manipulating and reading geospatial data
require("geoR")        # Methods for geostatistical analyses
```

C. Load and Compile a Large Dataset

We will analyze environmental and bacterial community data from a survey of shallow ponds found east of Bloomington, IN. These ponds are scattered throughout Brown County State Park, Yellowood State Forest, and Hoosier National Forest. In the 1940s, Maury Reeves of the Indiana Department of Natural Resources began constructing refuge ponds for wildlife. In the summer of 2013, we visited approximately 50 of these ponds and recorded their geographic locations using a GPS unit; 'GPS' is the acronym for Global Positioning System. We sampled aspects of water chemistry, physical properties, and bacterial community composition. Let's load the environmental and site-by-species data for the refuge ponds.

```
Ponds <- read.table(file="BrownCoData/20130801_PondDataMod.csv", head=TRUE, sep=",")
lats <- as.numeric(Ponds[, 3]) # latitudes (north and south)
lons <- as.numeric(Ponds[, 4]) # longitudes (east and west)
OTUs <- read.csv(file="BrownCoData/SiteBySpecies.csv", head=TRUE, sep=",")
```

Take a look at the `Environment` tab in the upper right corner. You should see there are 16384 OTUs distributed across 52 sites, for which, we have 21 variables recorded. These variables include elevation (in meters) and geographical coordinates (lat-long), temperature (Celcius), dissolved oxygen (DO), among others. **This would be a great time to do some due diligence and look at the environmental metadata in the README.md file.**

As you can see, we have a lot of data environmental and geographical data associated with the bacterial communities at our 52 refuge ponds. But, let's not stop there. Let's add four additional columns of data to `Ponds`. There will be a column for richness (S), total abundance (N), Shannon's Diversity (H), and Simpson's evenness (De). These will provide quick-and-easy diversity related variables to explore with respect to distance and environment.

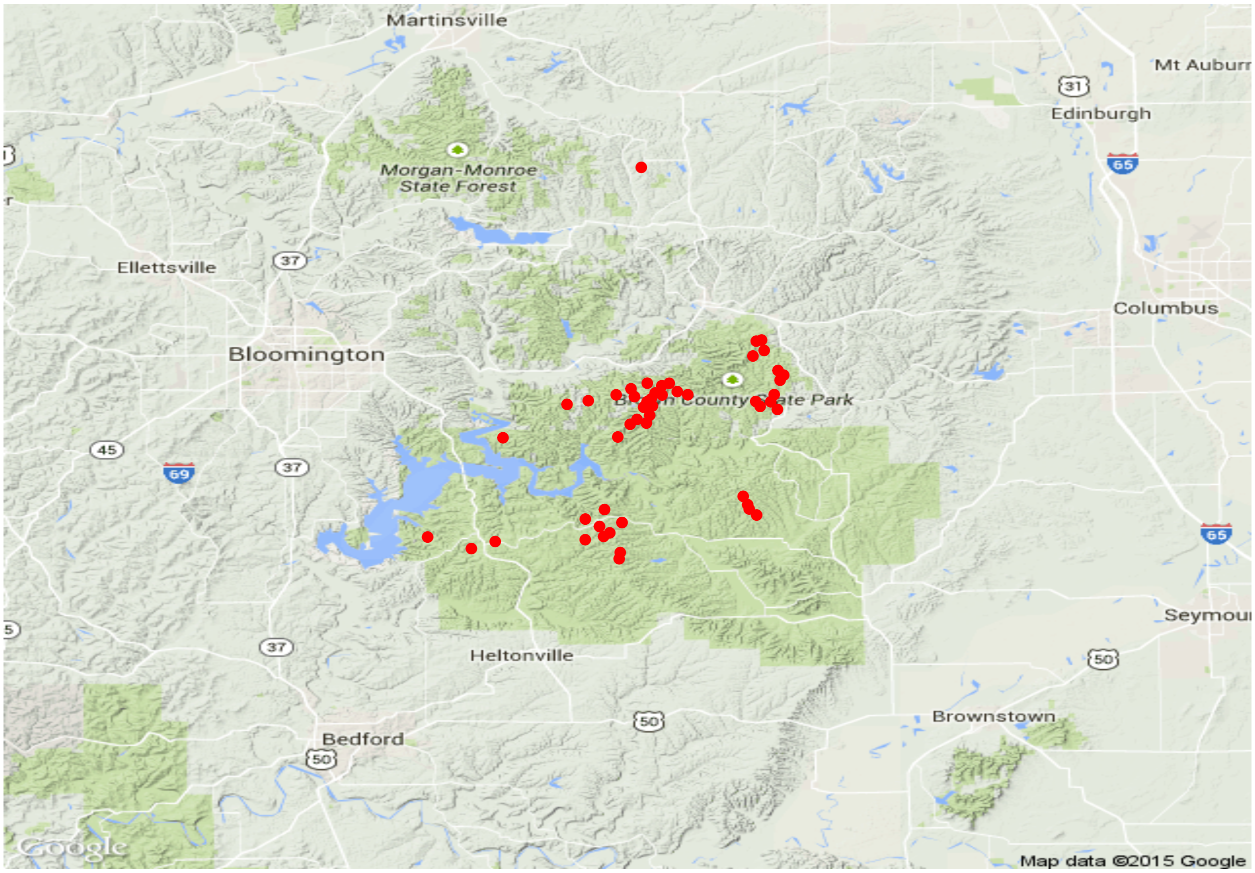
```
otu.names <- names(OTUs)
OTUs <- as.data.frame(OTUs[-1]) # remove first column (site names)

Ponds$N <- as.vector(rowSums(OTUs)) # no. reads
Ponds$S <- as.vector(rowSums(OTUs > 0) * 1)
Ponds$H <- as.vector(diversity(OTUs, index = "shannon"))
Ponds$De <- as.vector(diversity(OTUs, index = "invsimpson")/Ponds$S)
# Note, for De at each site, we divided Simpson's Diversity by OTU site richness
```

Now we have a large compilation of geographical, environmental, taxonomic, richness, abundance, diversity, and evenness data! Let's do some Geographical Ecology!

2.) MAP SAMPLES AND DATA

Let's visualize the spatial distribution of our samples with a basic map in RStudio by generating a map of the refuge ponds using the `GetMap` function in the package `RgoogleMaps`. This map will be centered on Brown County, Indiana (39.1 latitude, -86.3 longitude).

```
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)
```

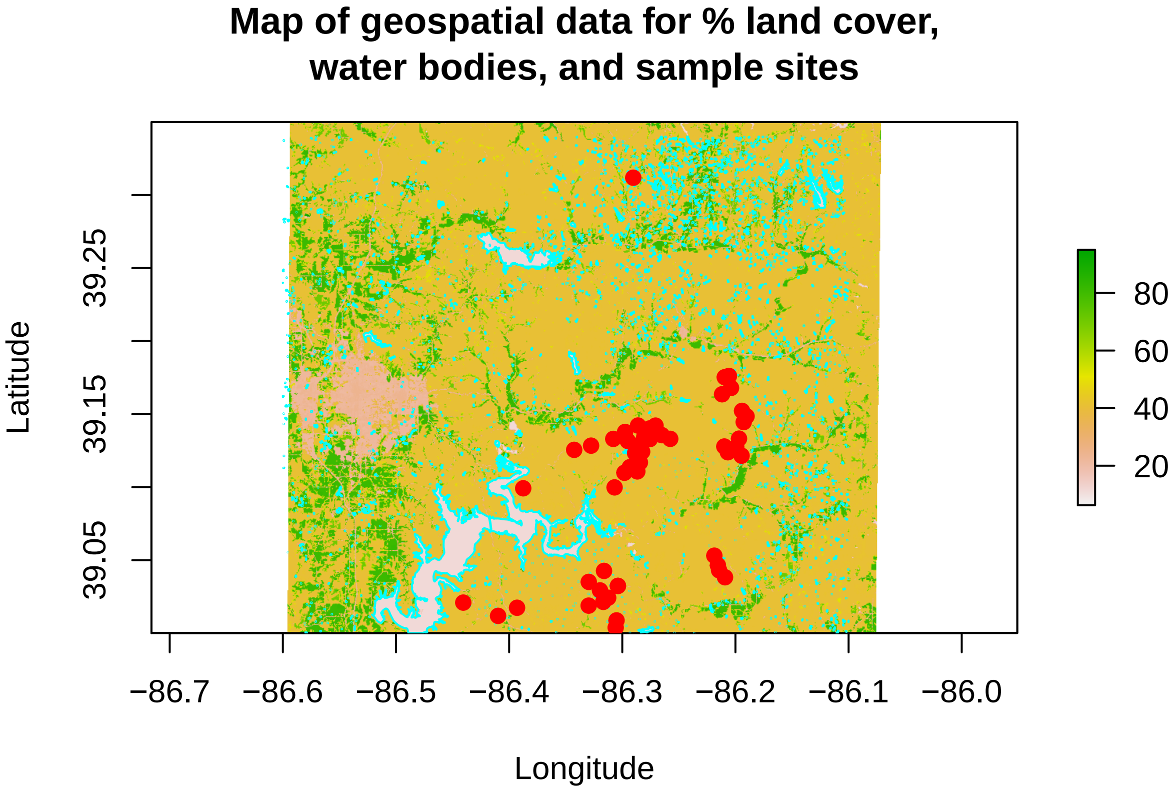


This map displays a lot of useful information that we otherwise, would not have been aware of. For example, all points are on National Forest land except for one that is north of all others. **Perhaps, a natural outgroup?** Likewise, the sample ponds appear to be aggregated in four or five small groups and distributed across a topographically complex area.

Despite being a fast way to contextualize our sample ponds within the broader landscape, the Google map misses a lot of information that would otherwise help us to understand the environmental and geographical factors that may coincide with our observations on diversity. Likewise, the geographical data contained within the map cannot be extracted.

For spatially explicit data on environmental and geographic features, i.e. geospatial data, we can turn to one of the many freely accessible online GIS databases and warehouses. Here, we will use the high quality geospatial data on Indiana water bodies and percent landcover. We obtained these data ‘layers’ from the **IndianaMap** geographical layer gallery: http://maps.indiana.edu/layerGallery.html.

```
Land.Cover <- raster("LandCover/LandCover.tif")
plot(Land.Cover, xlab='Longitude', ylab='Latitude',
main='Map of geospatial data for % land cover,\nwater bodies, and sample sites')

Water.Bodies <- readShapeSpatial("water/water.shp")
plot(Water.Bodies, border='cyan', axes=TRUE, add = TRUE)

Refuge.Ponds <- SpatialPoints(cbind(lons, lats))
plot(Refuge.Ponds, line='r', col='red', pch = 20, cex=1.5, add=TRUE)
```



Note, that the percent land cover, water bodies, and points for refuge ponds are in spatial agreement, i.e., there is no obvious mis-alignment. That is because we have previously modified each layer to have the same **datum**, **projection**, and nearly the same **extent**.
Working with geospatial data can be challenging because there is so much information involved with correctly identifying where on Earth something occurs and because there are many ways to represent points on a globe with 2-dimensional surface, i.e., a map. But, whether it's data on temperature, elevation, soils, geology, human demographics, ecoregions, etc., diverse data can be found among the many GIS warehouses. Here are some: 1.) USGS: http://viewer.nationalmap.gov/viewer/ 2.) State organizations: http://www.indianamap.org/resources.php 3.) USDA: http://datagateway.nrcs.usda.gov/

PRIMARY CONCEPTS OF GEOGRAPHICAL ECOLOGY

Having imported our primary community and environmental data from the refuge ponds, as well as having obtained a wealth of geospatial data from online sources, we are now ready to make a data intensive exploration into primary concepts and patterns of geographical ecology!

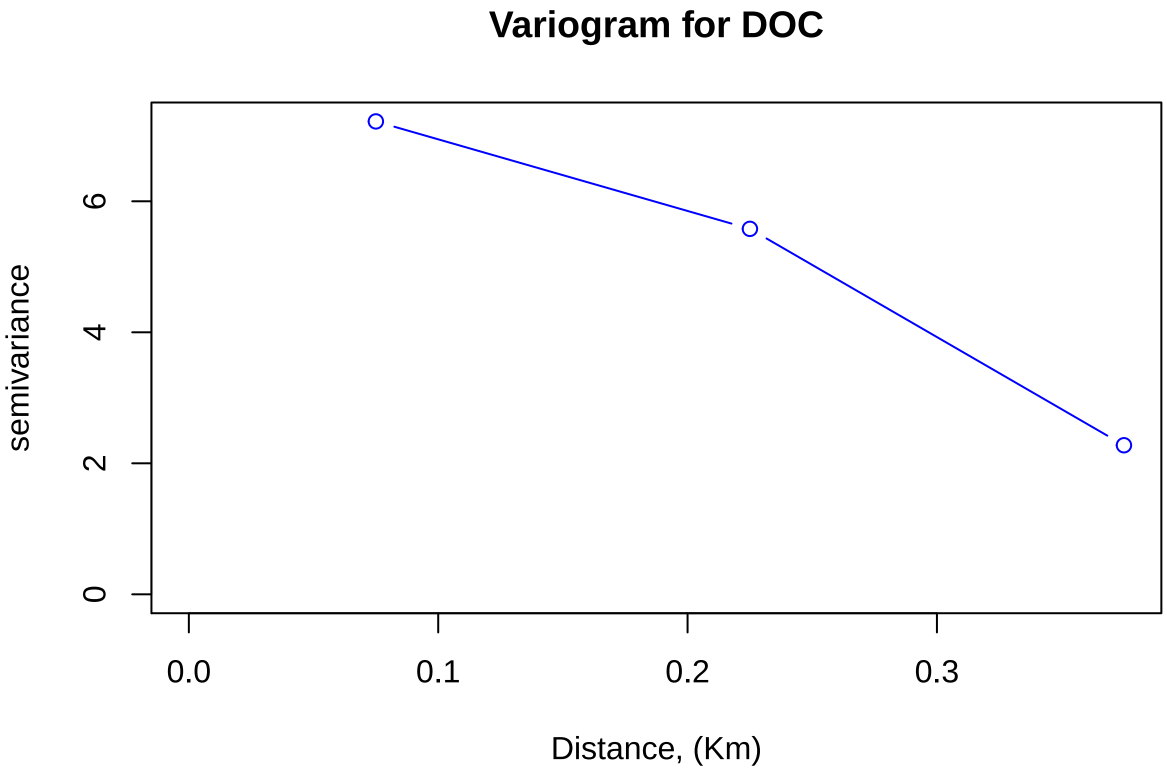
Concept 1: Spatial Autocorrelation

Tobler's first law of geography states that "Everything is related to everything else, but near things are more related than distant things" (Tobler 1970). This law is a formulation of the concept of spatial autocorrelation. In short, spatial autocorrelation is the degree to which spatial variables are either clustered in space (positive autocorrelation) or over-dispersed (negative autocorrelation). **When examing spatial data, it is important to check for autocorrelation not just among variables but across scales.**

Last week, you used the Mantel test to detect autocorrelated changes in environmental variables. This week, we reveal another way of detecting autocorrelation with respect to scale, that is, by using a **variogram**.

Variograms are frequently used in spatial analyses and reveal the degree of spatial autocorrelation in sample data and how the autocorrelation changes over scales of distance.

```
dists <- dist(cbind(lats, lons))
breaks = seq(0, 1.5, l = 11)

v1 <- variog(coords = cbind(lats,lons), data = Ponds$DOC, breaks = breaks)
v1.summary <- cbind(c(1:10), v1$v, v1$n)
colnames(v1.summary) <- c("lag", "semi-variance", "# of pairs")
plot(v1, type = "b", main = "Variogram for DOC", xlab='Distance, (Km)', col='blue')
```

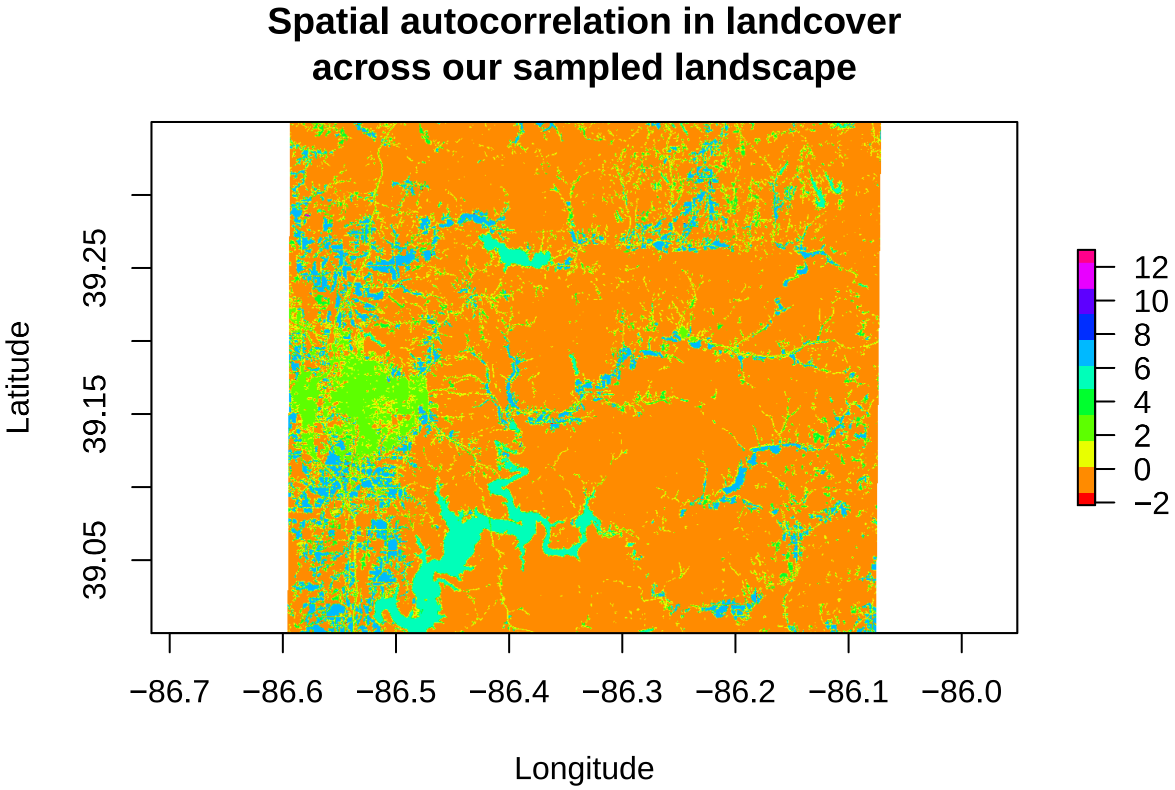


As you can see, the **semivariance** decreases with distances, where at a distance near 0.4 kilometers the semi-variance approaches 2. But, what's the semivariance? Well, if you were to find the differences between all possible points spaced a constant distance apart. And then, if you were to find the variance among the differences, and then divide that variance in half, you would have the semivariance! Consequently, a variance of 4 or semivariance of 2 might be considerd small.

For a more visually informative picture, we can visualize autocorrelation across the landscape, by calculating **Moran's I**, a correlational statistic that measures autocorrelation based on feature locations and feature values. Moran's I evaluates whether the pattern expressed is clustered, dispersed, or random and assigns a Moran's I Index value, a z-score and p-value. Take for example, our land cover data represented in a raster file. Using R's `raster` package, we can calculate **global** (across the landscape) and **local** (in comparison to neighbors) measures of **Moran's I**.

```
Moran(Land.Cover)
LC.Moran <- MoranLocal(Land.Cover)
```

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



Looking at the ‘landscape’ of Moran’s I, we can see that spatial autocorrelation, at least in land cover, is generally low.

Primary Pattern 1: Distance-decay relationship

The distance-decay relationship is the primary pattern of spatial autocorrelation, and captures the rate of decreasing similarity with increasing distance. This patterns address the question of whether near communities have greater similarity than far communities. Likewise, it can also be used to address whether near environments have greater similarity than far ones.

Let’s load the `simba` package and generate distance decay relationships for the environment and community composition. We’ll only examine a few environmental variables and leave it to you to explore the rest!

```
require("simba")

struc.dist <- 1 - vegdist(OTUs) # Bray-Curtis similarity between the plots
coord.dist <- dist(as.matrix(lats, lons)) # geographical distance between plots

# transform environmental data to numeric types
temp <- as.numeric(Ponds$"Salinity")
elev <- as.numeric(Ponds$"ORP")
```

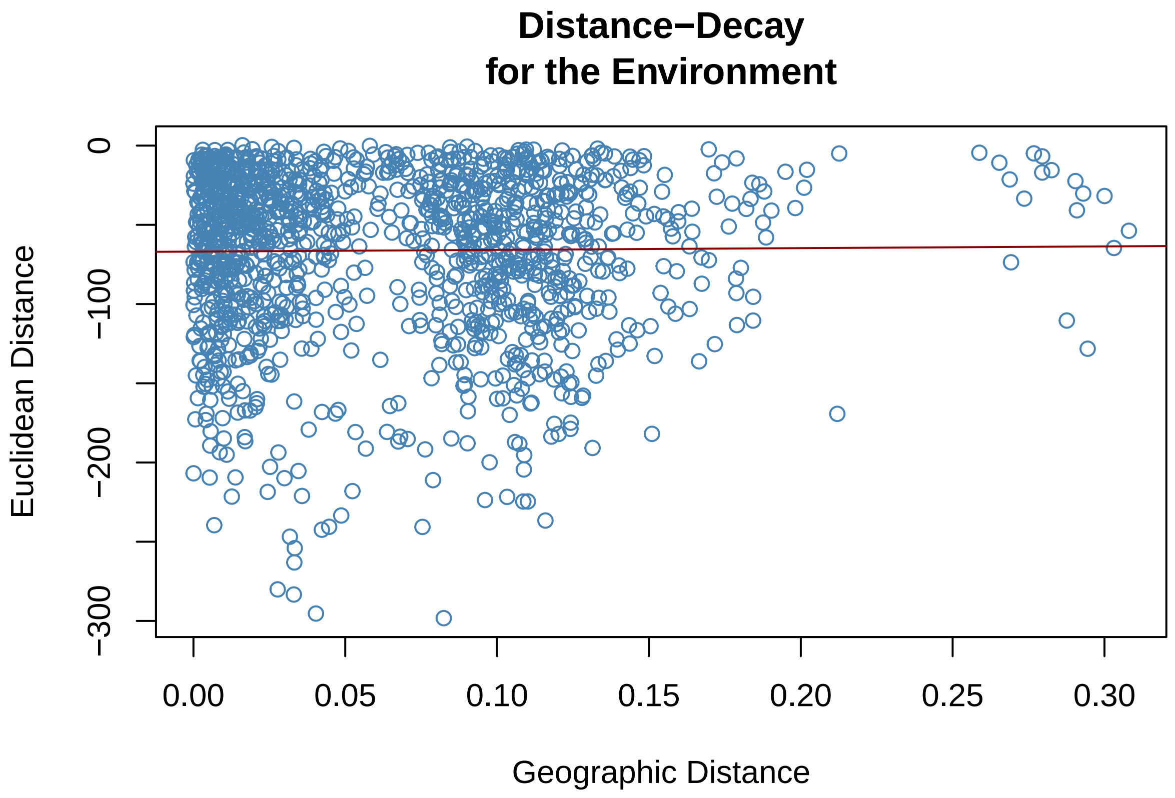
```
depth <- as.numeric(Ponds$"Depth")
doc <- as.numeric(Ponds$"DOC")

# calculate the distance (Euclidean) between the plots regarding environmental variables
env.dist <- 1 - vegdist(cbind(temp, elev, depth, doc), "euclidean")

# transform all distance matrices into list format:
struc.dist.ls <- liste(struc.dist, entry="struc")
env.dist.ls <- liste(env.dist, entry="env")
coord.dist.ls <- liste(coord.dist, entry="dist")
```

Now, create a data frame containg plot information, geographical distance, similarity of the environment, and similarity of community.

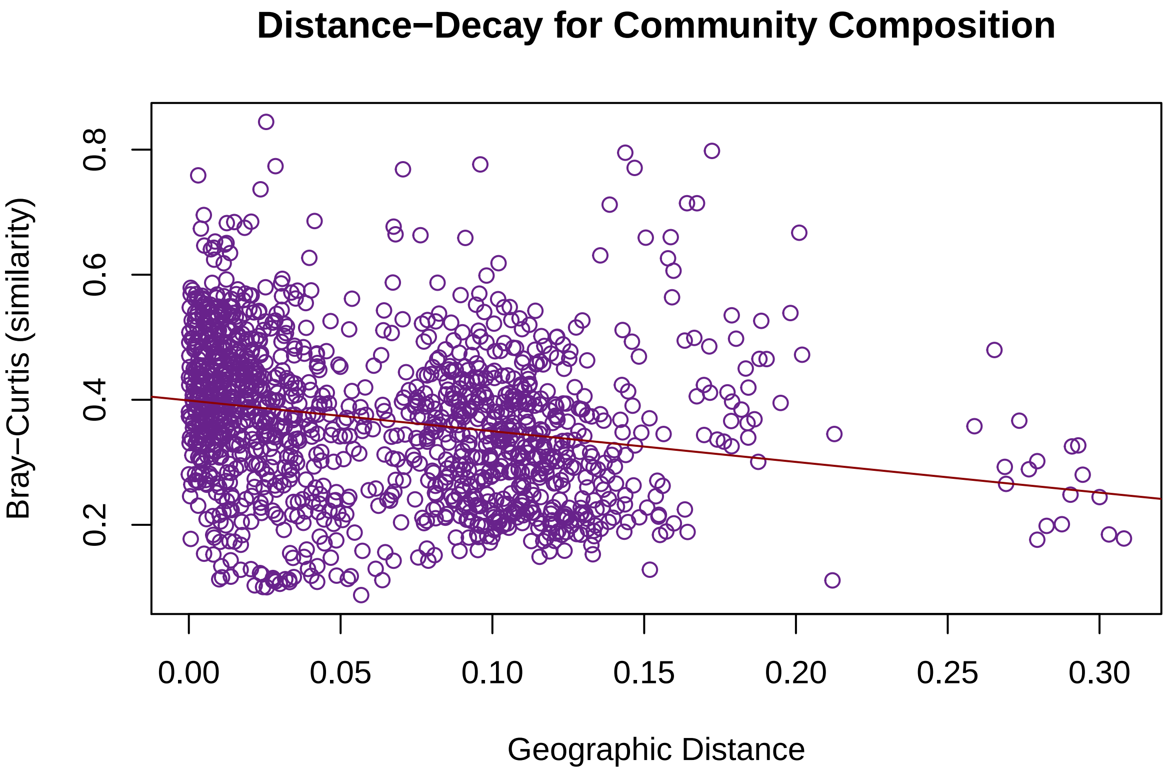
```
df <- data.frame(coord.dist.ls, env.dist.ls[,3], struc.dist.ls[,3])
names(df)[4:5] <- c("env", "struc")
attach(df) #df <- subset(df, struc != 0)
```

Finally, let's plot the Distance-decay relationships, with regression lines in red.

```
par(mfrow=c(1, 1))
plot(dist, env, xlab="Geographic Distance", ylab="Euclidean Distance",
     main = "Distance-Decay\nfor the Environment", col='SteelBlue')
abline(lm(env ~ dist), col="red4")
```



```
par(mfrow=c(1, 1))
plot(dist, struc, xlab="Geographic Distance", ylab="Bray-Curtis (similarity)",
     main="Distance-Decay for Community Composition", col='darkorchid4')
abline(lm(struc ~ dist), col="red4")
```



```
# Are the slopes significantly different?
diffslope(dist, env, dist, struc)
```

```
##
## Is difference in slope significant?
## Significance is based on 1000 permutations
##
## Call:
## diffslope(x1 = dist, y1 = env, x2 = dist, y2 = struc)
##
## Difference in Slope:  11.4
## Significance: 0.313
##
## Empirical upper confidence limits of r:
##   90%   95% 97.5%   99%
##  27.5  36.5  47.6  58.0
```

It seems that microbial communities that are closer in geographic distance are also closer in compositional similarity. However, this does not seem to be the case for environmental similarity, at least, in the variable we either measured directly or were able to obtain data for.

So, rather than being spatially autocorrelated perhaps, environmental conditions are extremely patchy. This might make natural sense considering that we're examining aquatic habitat patches, i.e., refuge ponds. But then why would closer communities be more similar than far one? Perhaps, an examination of spatial aggregation in can help.

Concept 2: Aggregation

Tobler made a very general observation that occurs in nearly all systems. However, a related and seemingly conflicting observation is that natural phenomena are generally aggregated. In short, this means that similar conditions, organisms, and events are often encountered in patches, pulses, runs, and between certain periods of time. As you saw from the GIS map, you wouldn't have to walk long before encountering a pond.

Reconciling the concepts of spatial autocorrelation and aggregation we might say that, **"Getting futher away means getting more different. . . until you encounter a new patch, pulse, or period of previous conditions."**

Here, we will examine aggregation with respect to the relative abundance of bacterial taxa sampled among the refuge ponds.

Spatial aggregation increases the chance that ecological differences between samples will reflect differences owing to geographical similarities and distance between points. Looking at the map of refuge ponds, we can see that some points belong to distinct clusters.

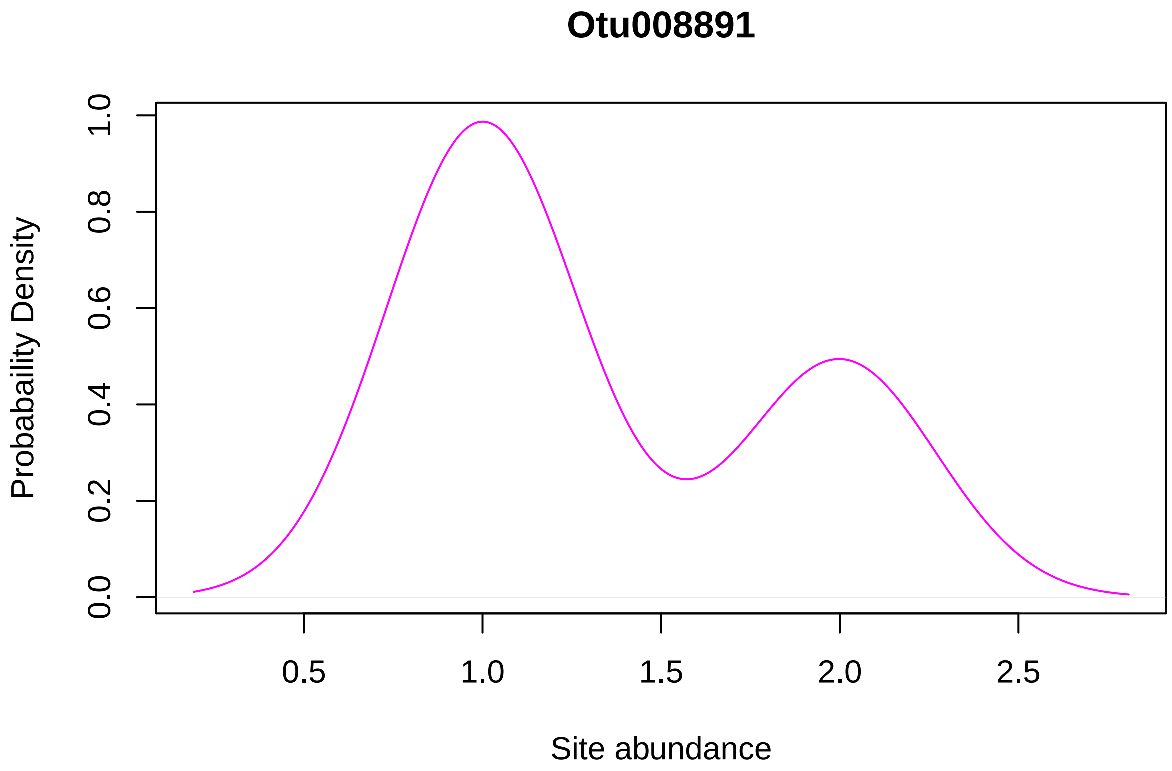
Primary Pattern 2: Spatial abundance distribution

One of the most primary patterns of spatial aggregation is the distribution of a species or OTU abundance (or relative abundance) across space. We can capture the degree of 'intraspecific' variation with a frequency distribution revealing the frequency at which we find a species at a particular abundance, whether relative or not.

Here, we will construct **kernel-density curves** for the distribution of abundance among individual OTU's across the refuge pond dataset. Kernel density curves are analogous to histograms, but avoid the arbitrary creation of bins (i.e. bars). In constructing kernel density curves, we attempt to account for uncertainty and sampling error by focusing on the probability that a randomly drawn data point will take a value within a particular range, instead of the exact frequencies we observed.

Let's draw an OTU at random and plot its spatial abundance distribution.

```r
ad <- c(2, 2)
otu <- sample(1:length(OTUs), 1) # a random OTU
ad <- OTUs[, otu]
ad = as.vector(t(x = ad))
ad = ad[ad > 0]

plot(density(ad), col = 'magenta', xlab='Site abundance',
     ylab='Probabaility Density',  main = otu.names[otu])
```



Feel free to run this chunk as many time as you like. As you will likely see, the sampled abundance for a given OTU is often aggregated, revealing many sites where the OTU is relatively rare and many where it is relatively more common. This type of uneven spatial distribution of abundance is exceptionally common in ecological systems, and is often referred to as the **species spatial abundance distribution (SSAD)**.

Question 2: In the site-by-species matrix, each row represents a site and each column represents an OTU. If the SSAD is generated by considering all rows for a single column, then what do we obtain when we consider all columns for a given row? Have we examined this sort of data structure before? If so, elaborate?

> ***Answer 2***:

Concept 3: Scale-Dependence

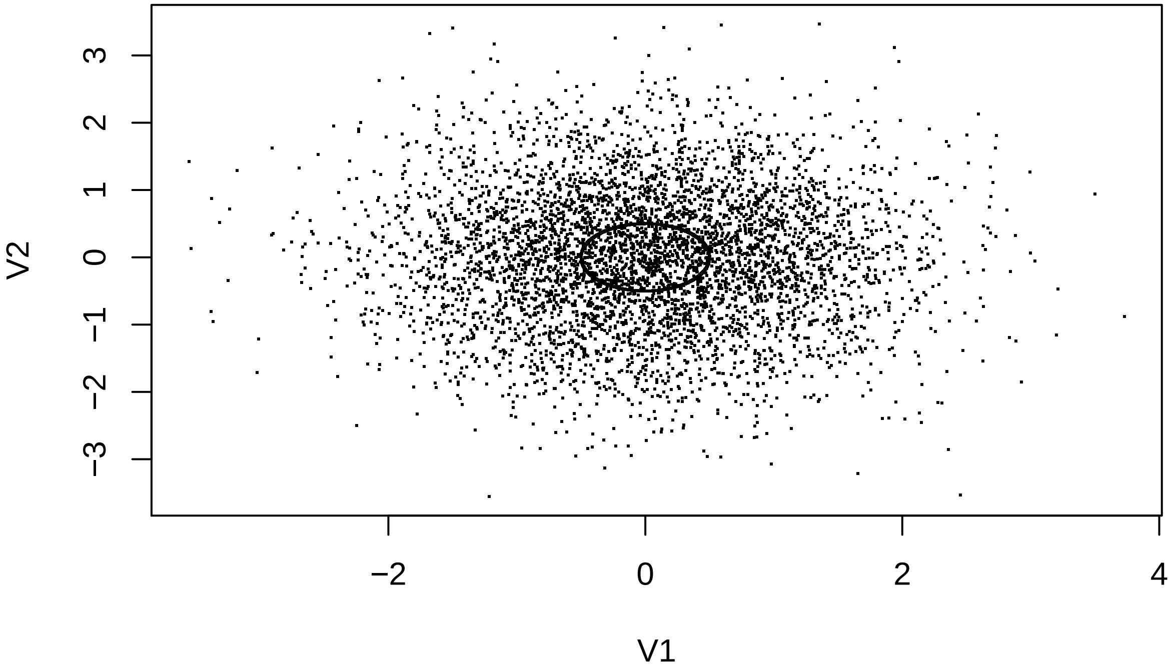
i. extent and grain

ii. How interpretation can change with extent

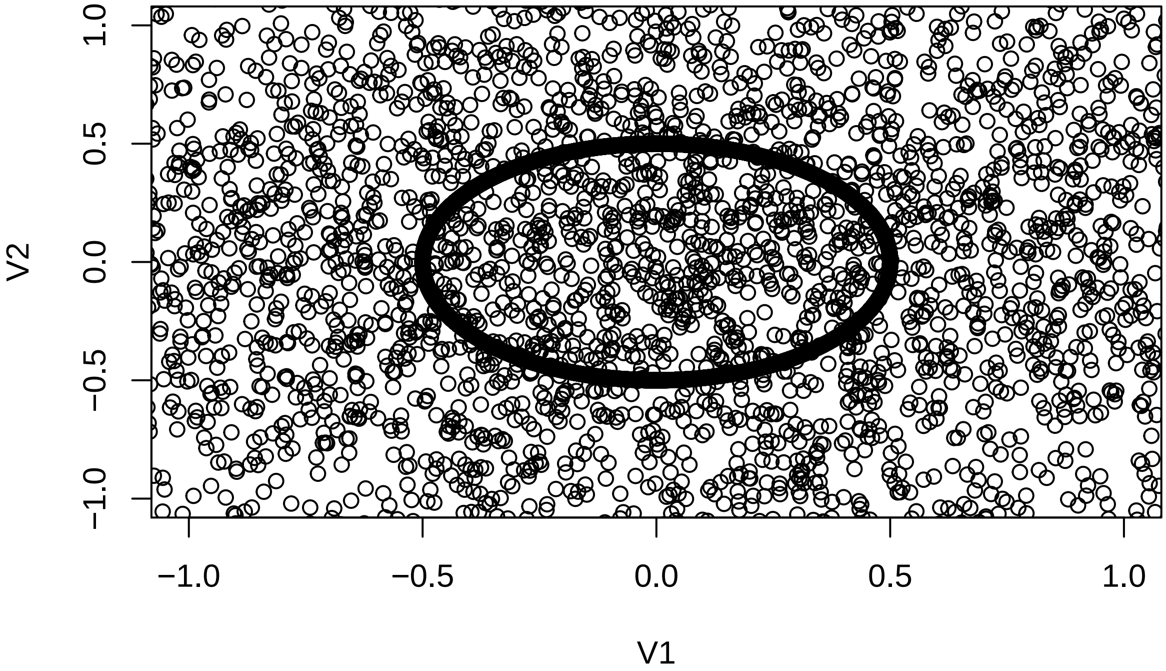
Here, we generate a single sample of data based on a random draw from a normal distribution (a.k.a., Gaussian distribution, bell curve). While all the points were randomly drawn from the same distribution, zooming in and out to different extents reveals what we would very likely call different patterns of aggregation.

```r
# generate the data
set.seed(20111105)
x = rbind(matrix(rnorm(10000), ncol = 2), local({
  r = runif(10000, 0, 2 * pi)
  0.5 * cbind(sin(r), cos(r))
}))
```

```
x = as.data.frame(x[sample(nrow(x)), ])

plot(x, pch = ".") # The distribution of points is clearly aggregated around the center of the graph (i
```



```
plot(x, xlim = c(-1, 1), ylim = c(-1, 1)) # Zooming in, our first guess would not be that these points 
```



Primary Pattern 3: Species-area relationship (SAR)

The species-area relationship (SAR) describes the rate at which we discover or accumulate species with increasing area. The general relationship of the SAR … and is of the form S=c*A^z … which was first predicted by Arrhenius (1921)…wow, that's old.

Random accmulation of sites

Do we accumulate species with greater area just because there is more area or because greater area means more niches?

Accumulation of sites by proximity