Spatial Diversity

Z620: Quantitative Biodiversity, Indiana University

February 3, 2017

OVERVIEW

In this exercise, we will consider alpha (α) and beta (β) diversity in a spatial context. We will introduce Geographical Information Systems (GIS) to map and spatially examine environmental and biodiversity data. This will allow us to explore core concepts like spatial autocorrelation, aggregation, and scale dependence.

After completing this exercise you will be able to:
1. Identify primary concepts and patterns of spatial diversity.
2. Examine effects of geographic distance on environmental and ecological similarity.
3. Characterize aggregation of abundance across space.
4. Examine the extent to which patterns of diversity depend on spatial scale.
5. Use geospatial data and packages to conduct GIS operations in R.
6. Use control structures such as `loops` to control how R operates on variables.

1.) SETUP

A. Retrieve and Set Your Working Directory

```r
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 a suite of packages developed in R for geographical information systems (GIS).

```r
require(vegan)
#install.packages('sp') # Classes and methods for handling spatial data
require(sp)
#install.packages('gstat') # Methods for geostatistical analyses
require(gstat)
#install.packages('raster') # Methods to create a RasterLayer object
require(raster)
#install.packages('RgoogleMaps') # For querying the Google server for static maps.
require(RgoogleMaps)
#install.packages('maptools') # Tools for manipulating and reading geospatial data
require(maptools)
#install.packages('rgdal') # Geospatial Data Abstraction Library
require(rgdal)
#install.packages('simba')
require("simba")
#install.packages('gplots')
require("gplots")
```

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 to increase wildlife habitat. 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 console of RStudio. You should see there are 16,384 operational taxonomic units (OTUs) distributed across 51 sites, for which we have 19 environmental and geographic variables recorded. These variables include elevation (m), geographical coordinates (lat-long), temperature (C), pond diameter(m), pond depth (m), redox potential (ORP), specific conductivity or SpC (S/cm), dissolved oxygen (mg/L), total dissolved solids (g/L), salinity (p.s.u.= ppm), color - measured at absorbance = 660; an estimate of carbon in the water sample, chlorophyll *a* (g/ml), dissolved organic carbon (mg/L), dissolved organic nitrogen (mg/L), and total phosphorus (g/L).

In addition to this wealth of environmental data, let's add four diversity-related columns of data to `Ponds` data set. These will provide basic diversity-related variables to explore with respect to geograpy and environmental conditions. There will be a column for richness (S), total abundance (N), Shannon's Diversity (H), and Simpson's evenness (De).

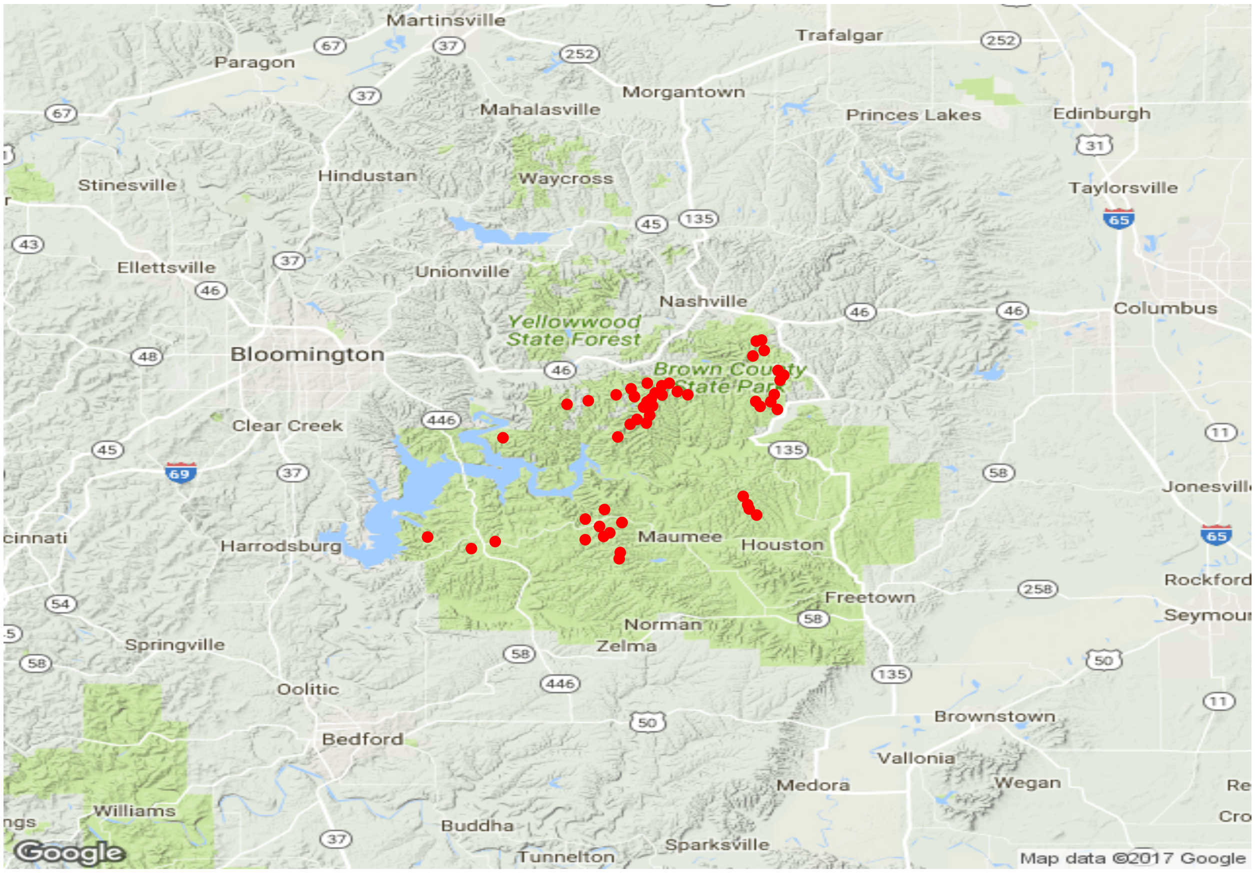
```
# To get S, we sum the number of non-zero abundances for a site
S.obs <- function(x = ""){ rowSums(x > 0) * 1}

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

Ponds$N <- as.vector(rowSums(OTUs)) # numbers of reads
Ponds$S <- S.obs(OTUs) # number of species OTUs
Ponds$H <- as.vector(diversity(OTUs, index = "shannon"))

# To get De at each site, we divide Simpson's Diversity by OTU site richness
Ponds$D <- as.vector(diversity(OTUs, index = "invsimpson")/Ponds$S)
```

Now we have a large compilation of geographical, environmental, and biodiversity data. Let's do some spatial diversity!

2.) MAP SAMPLES AND DATA

First, let's visualize the spatial distribution of our samples with a basic map in RStudio. Let's generate 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)
```

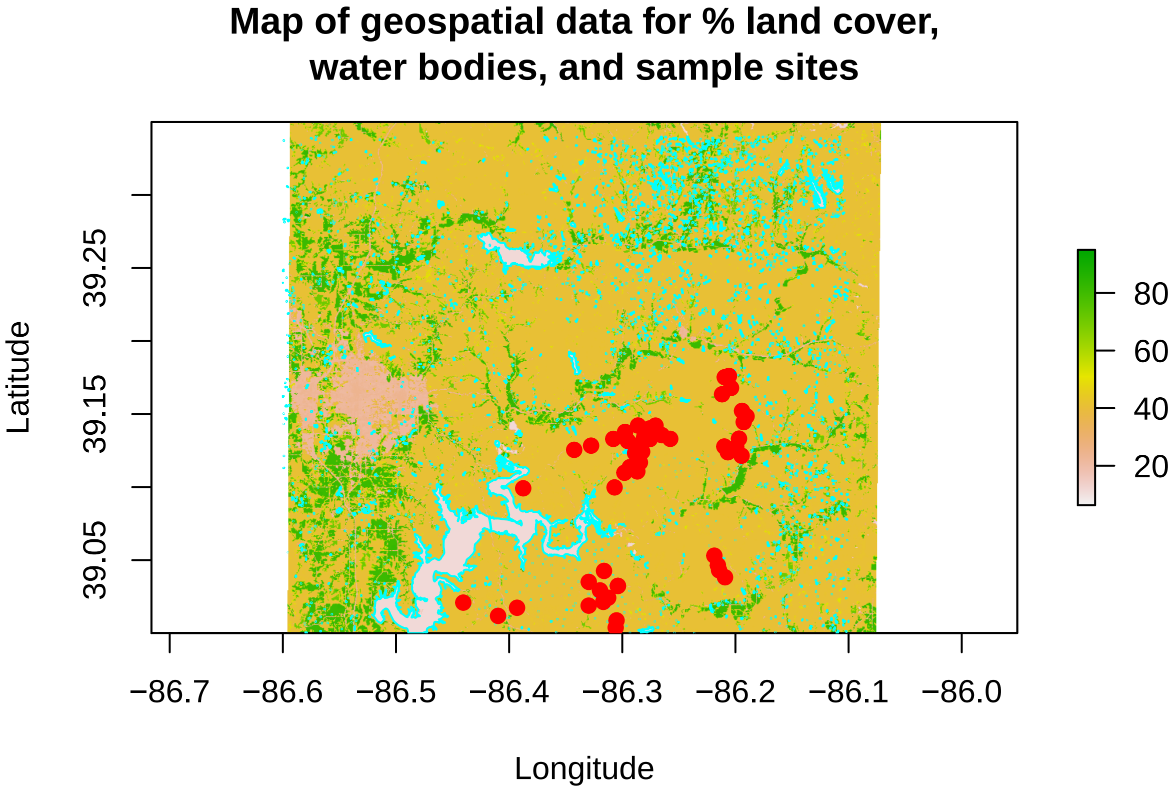


This map displays a lot of useful information that we otherwise, would not have been aware of. For example, all points are on State or National Forest land. Likewise, the sampled ponds appear to be aggregated in four or five small groups and distributed across a topographically complex area.

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, because the Google map is only an image, it does not contain any extractable environmental or geographic data.

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*.

See the GLOSSARY at the end of this document for basic definitions of these and other GIS terms.

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 a 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 a few: 1) USGS: http://viewer.nationalmap.gov/viewer/ 2) State organizations: http://www.indianamap.org/resources.php 3) USDA: http://datagateway.nrcs.usda.gov/

3. PRIMARY CONCEPTS AND PATTERNS

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.

A. 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).

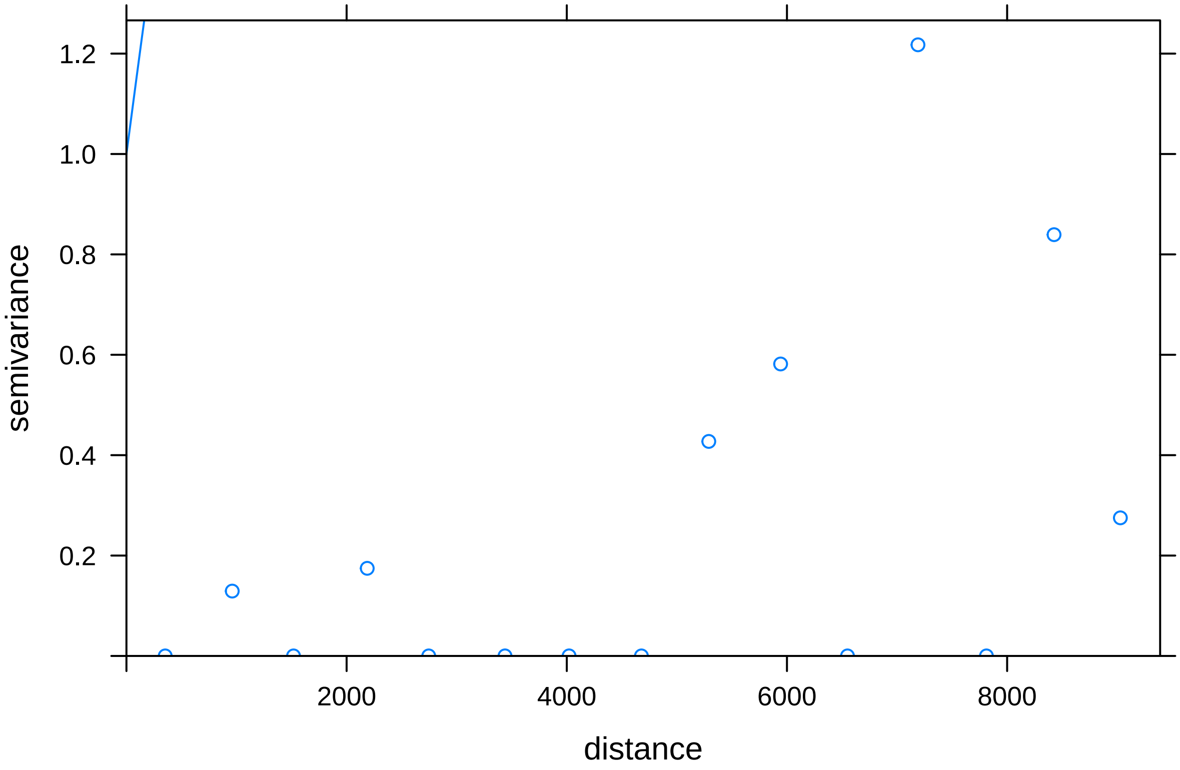
This law is a formulation of the concept of spatial autocorrelation. In short, spatial autocorrelation is the degree to which spatial variables are either clustered in space (positive autocorrelation) or over-dispersed (negative autocorrelation).

When examing spatial data, it is important to check for autocorrelation not just among variables but across distance. We want to know more than whether variables are generally auto-correlated, but how greatly that autocorrelation changes (increases or decreases) with increasing distance. For example, variables that are highly autocorrelated at the meter scale may or may not be less autocorrelated at the kilometer scale. Importantly, if we want to use these variables in our analyses, we should not use the scales over which the variables are autocorrelated.

Here, we reveal a 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 (measured as the **semivariance**) changes over distance.

The semivariance is a measure of dispersion that is related to the variance, but which only considers observations below the mean. In this case, higher semivariance means implies greater dispersion (lower autocorrelation) of values at a given spatial scale. Let's plot the variogram for one of our environmental variables.

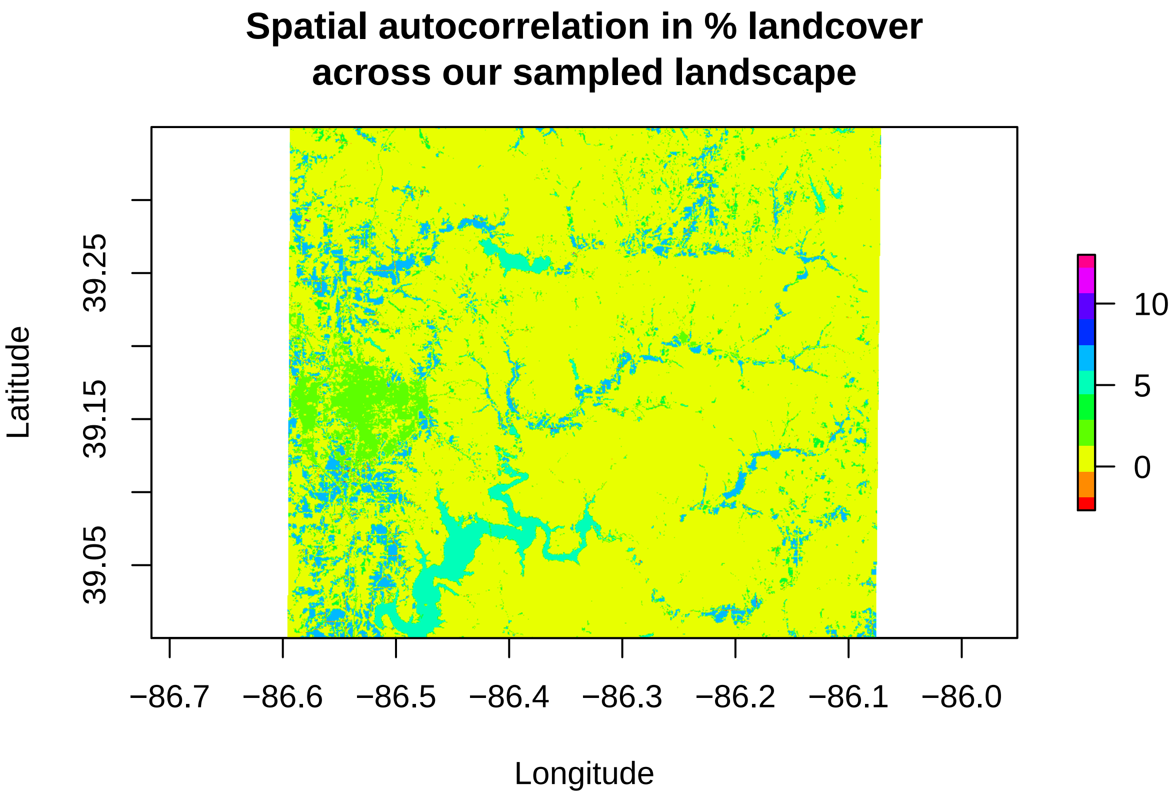
```r
# Construct a new dataframe for coordinates
xy <- data.frame(pond.name = Ponds$Sample_ID, lats = Ponds$lat, lons = Ponds$long)
# Transform 'xy' into a spatial points dataframe
coordinates(xy) <- c("lats", "lons")

# 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")

# Transform the projection and data, so we can get meaningful distances. In this case, we will use the
UTM <- spTransform(xy, CRS("+proj=utm + zone=51 ellps=WGS84"))

UTM <- as.data.frame(UTM)
Ponds$lats_utm <- UTM[,2] # lattitude data
Ponds$lons_utm <- UTM[,3] # longitude data

coordinates(Ponds) = ~lats_utm+lons_utm
vgm <- variogram(TDS~1, data=Ponds)
vgm.fit = fit.variogram(vgm, model = vgm(1, "Sph", 900, 1,fit.sill=F, fit.range=F))
plot(vgm, vgm.fit)
```



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. The function we will use also assigns a global Moran index value, a z-score and p-value.

Take for example, our land cover data represented in a raster file. Raster files are one of two types of digital graphic files, the other being vector files. Images in vector files are made of many lines and curves (or paths) that create the image. In contrast, raster images are composed of pixels and for our purposes, encode the values of the environment at a precise point. 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))
```

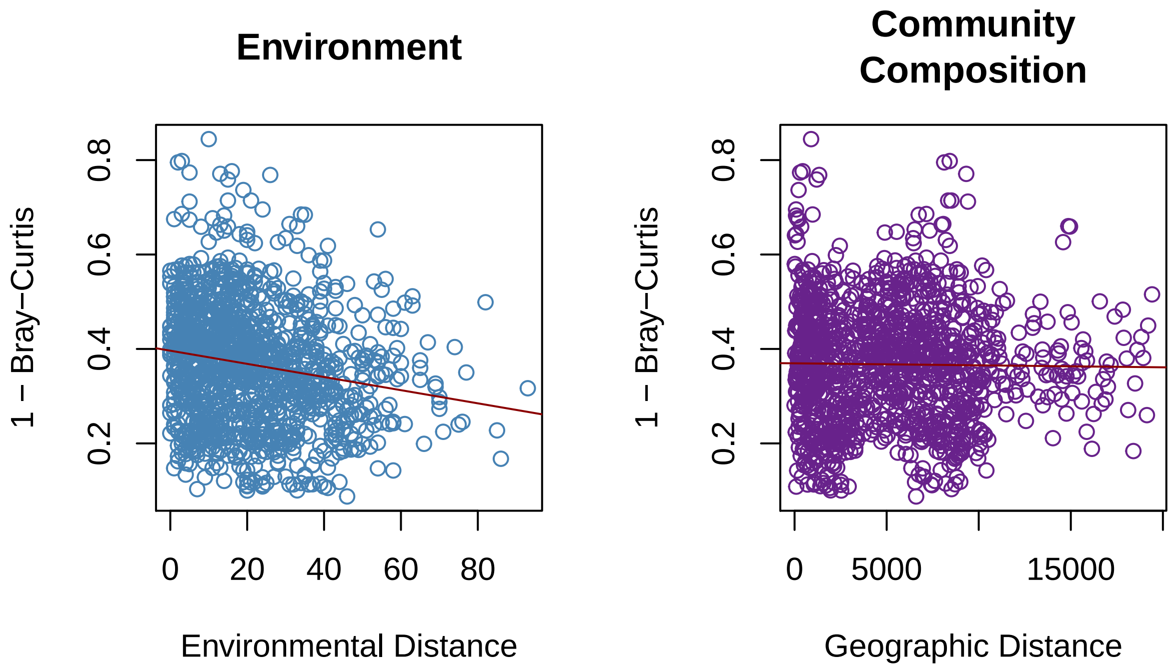


Pattern 1: Distance-decay relationship

The geograhpic distance-decay relationship is a pattern of spatial autocorrelation that captures the rate of decreasing similarity with increasing geographic distance. This pattern addresses whether communities close to one another are more similar than communities that are farther away. The distance-decay pattern can also be used to address whether near environments have greater similarity than far ones. We will use the `simba` package to generate distance decay relationships for bacterial communities of our refuge ponds and for some of the environmental variables we measured.

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

lats <- as.numeric(Ponds$lats_utm) # lattitude data
lons <- as.numeric(Ponds$lons_utm) # longitude data
coord.dist <- dist(as.matrix(lats, lons)) # geographical distance between plots

# transform environmental data to numeric types
x1 <- as.numeric(Ponds$"SpC")
# calculate the distance (Euclidean) between the plots regarding environmental variables
env.dist <- vegdist(x1, "euclidean")

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

```
#Now, 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])
names(df)[4:5] <- c("env", "struc")
attach(df)

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



Let's, examine the slope of the regression lines, asking whether they are significantly different from one another.

```
diffslope(env, struc, dist, struc)
```

```
##
```

```
## Is difference in slope significant?
## Significance is based on 1000 permutations
##
## Call:
## diffslope(x1 = env, y1 = struc, x2 = dist, y2 = struc)
##
## Difference in Slope: -0.001393
## Significance: 0.001
##
## Empirical upper confidence limits of r:
##      90%      95%    97.5%      99%
## 1.53e-06 1.94e-06 2.37e-06 2.69e-06
```

Concept 2: Spatial Aggregation

Tobler made a general observation that occurs in nearly all systems, i.e., spatial autocorrelation. A related observation is that natural phenomena are generally clustered, i.e., spatially aggregated. That is, individuals, conditions, and events often occur in patches, clusters, pulses, etc.

Pattern 2: Spatial abundance distributions

One of the primary patterns of spatial aggregation is the distribution of individuals within a landscape. We can refer to this pattern as the spatial abundance distribution and can examine in regards to all individuals of a community, to all individuals of a particular species, or any other subset of particular interest (e.g., rare species, common species). The spatial abundance distribution reveals the frequency at which we find individuals at particular abundances.

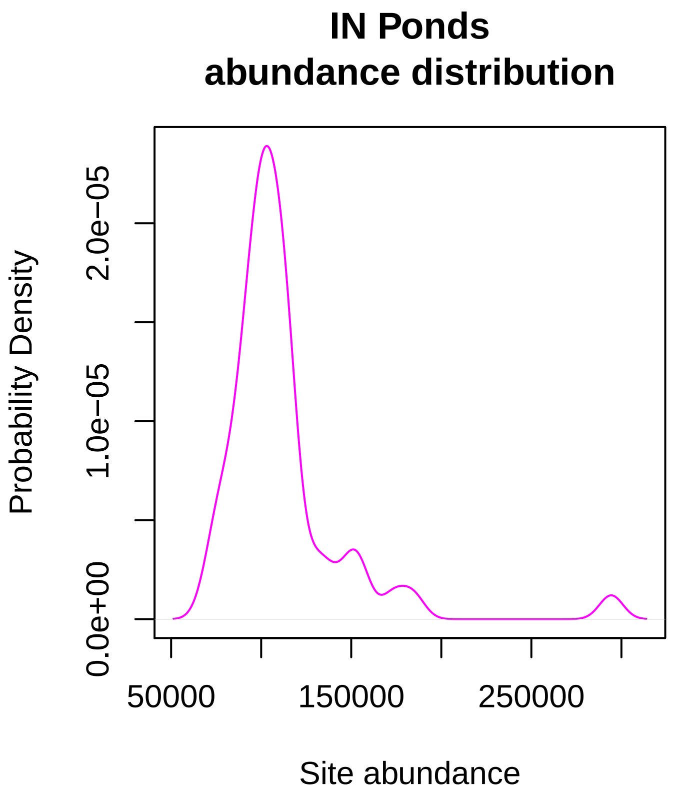
For example, perhaps we are interested in understanding how total abundance 16S rRNA genes differed among our ponds:

```
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
```

As we can see, we recovered between 60,000 and 300,000 sequences per site. As a more convient and insightful way to show how abundance varies among plots, let's plot our data 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')
```



Here, we plotted our data as a **kernel-density curve**, which is a particular way of representing a distribution of abundances, frequencies, etc. Kernel density curves are similar to histograms but avoid the need for discrete bins. In interpreting kernel density curves, we say that a randomly drawn data point will take a value within a particular range, e.g., between 50,000 and 150,000.

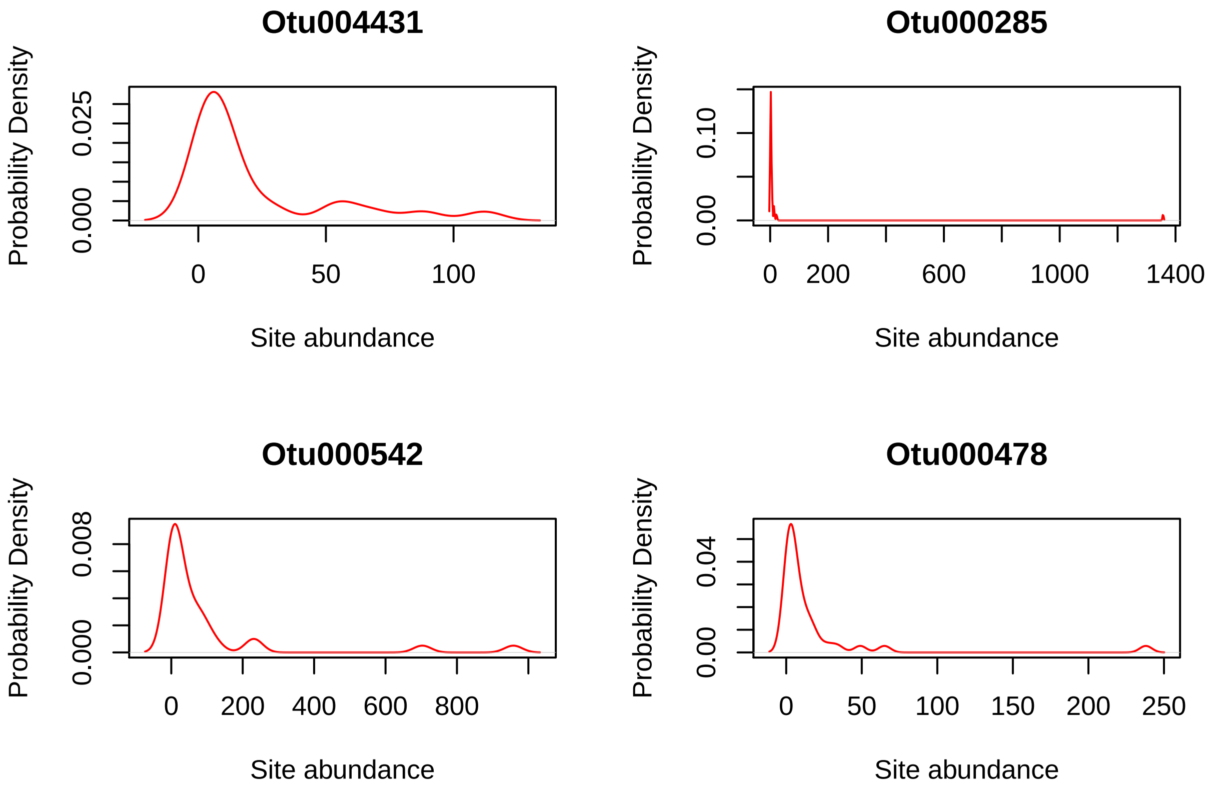
Below, we will examine the abundances distributions for individual species or OTUs, i.e., the species spatial abundance distribution (SSAD).

Let's first defining a function that will generate the SSAD for a given OTU.

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

Next, let's draw 4 OTUs at random from the IN ponds dataset and and plot their SSADs as kernel density curves. We will use **while loops** and **if** statements to accomplish this. The while loop is a type of **control flow** structure that allows us to tell a program to perform an operation *while* some condition is satisfied. And **if** statement is a conditional statement that tells a program to do something if a particular condition is true or false.

```
par(mfrow=c(2, 2))

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
  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])
    }
  }
```



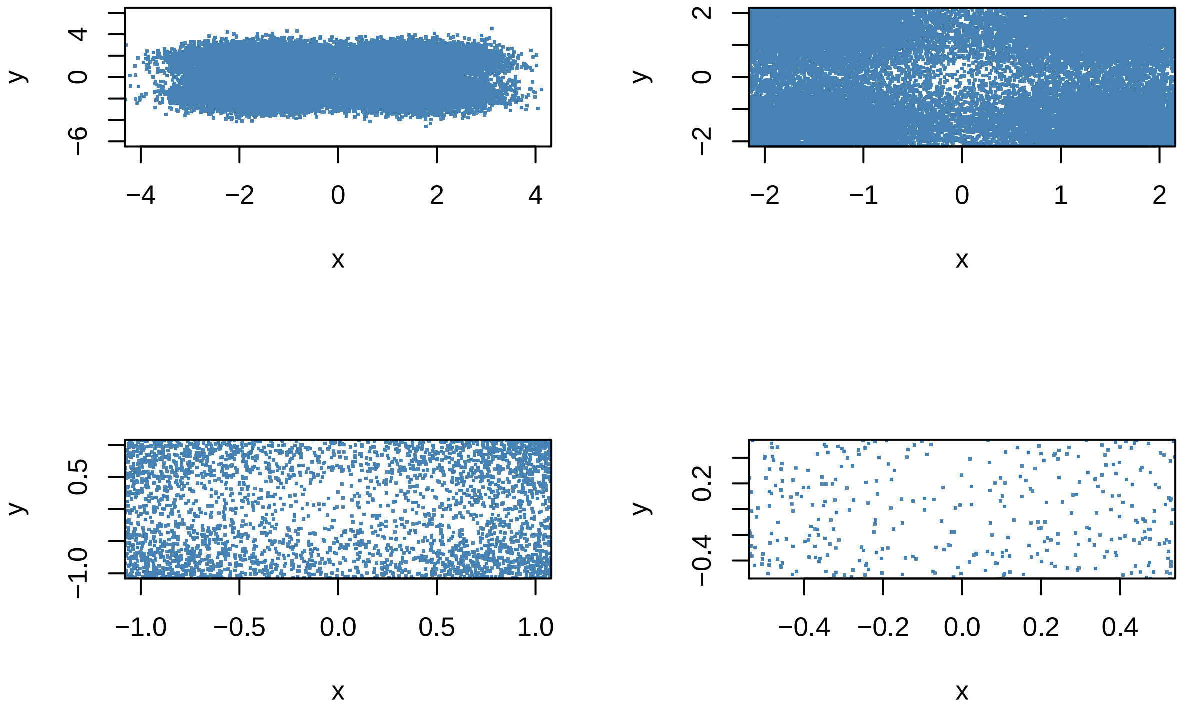
Concept 3: Spatial scale

Many patterns of biodiversity relate to spatial scale. The two main components of spatial scale are extent and grain. **Extent** is the greatest distance considered in an observation or study, e.g., a 50ha study area. **Grain** is the smallest or primary unit by which the extent is measured, e.g., 1ha subplots. Whenever conducting a scientific study with a spatial component, we aim to for a large enough extent to capture meaningful dynamics (e.g., dispersal limitation) and a small enough grain to capture a sufficiently resolved pattern.

As an example of considering the importance of spatial extent in your own studies, consider how a landscape of points can appear to be aggregated, uniform, or random, depending on how far we zoom in, e.g., from regional to local scales:

```
par(mfrow=c(2, 2))

# Draw random samples from a Normal distribution https://en.wikipedia.org/wiki/Normal_distribution
# Inputs to the Normal distribution are the expected standard deviation and the mean
sd <- 0.75 # standard deviation
mn <- 1.5 # mean
```

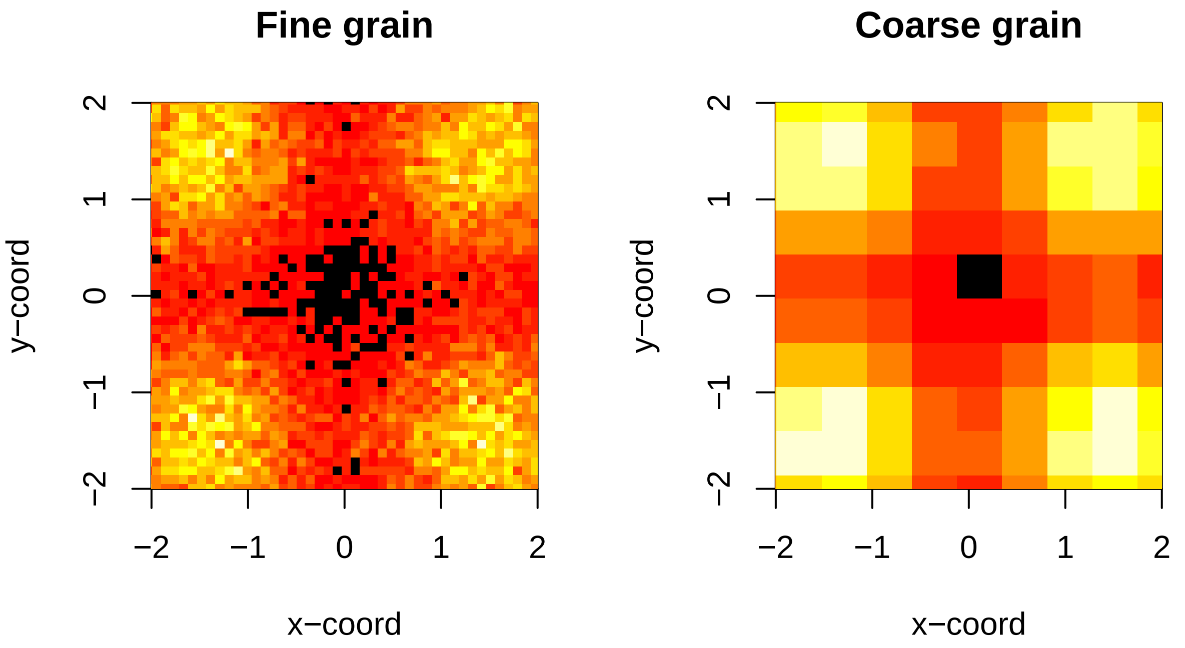
```r
x1 <- rnorm(10000, mean = mn, sd =sd) # x-coordinates
y1 <- rnorm(10000, mean = mn, sd =sd) # y-coordinates

x2 <- rnorm(10000, mean = -mn, sd =sd) # x-coordinates
y2 <- rnorm(10000, mean = -mn, sd =sd) # y-coordinates

x3 <- rnorm(10000, mean = -mn, sd =sd) # x-coordinates
y3 <- rnorm(10000, mean = mn, sd =sd) # y-coordinates

x4 <- rnorm(10000, mean = mn, sd =sd) # x-coordinates
y4 <- rnorm(10000, mean = -mn, sd =sd) # y-coordinates

y <- c(y1, y2, y3, y4)
x <- c(x1, x2, x3, x4)

plot(x,y, xlim=c(-4, 4), ylim=c(-6, 6), pch=".", cex=2, col='Steelblue')
plot(x,y, xlim=c(-2, 2), ylim=c(-2, 2), pch=".", cex=2,col='Steelblue')
plot(x,y, xlim=c(-1, 1), ylim=c(-1, 1), pch=".", cex=2,col='Steelblue')
plot(x,y, xlim=c(-0.5, 0.5), ylim=c(-0.5, 0.5), pch=".", cex=2, col='Steelblue')
```



All of these data points were randomly drawn from Normal distributions. The spatial properties of the simulated data clearly change with extent.

Spatial `grain` also influences our notion of how abundance and diversity are distributed across space. Holding the extent constant, heat maps (i.e. 2D histograms) reveal how the density of individuals across the landscape can appear more even if spatial grain is large enough. In general, the choice of spatial extent and grain should match resolution needed to answer your question or study your system.

```
par(mfrow=c(1, 2), pty="s")

df <- data.frame(x,y)

hist2d(df, nbins=100, show=TRUE, xlim=c(-2,2), ylim=c(-2,2),
             xlab='x-coord', ylab='y-coord', main = "Fine grain" )
hist2d(df, nbins=20, show=TRUE, xlim=c(-2,2), ylim=c(-2,2),
             xlab='x-coord', ylab='y-coord', main = "Coarse grain" )
```



Pattern 3: Species-area relationship (SAR)

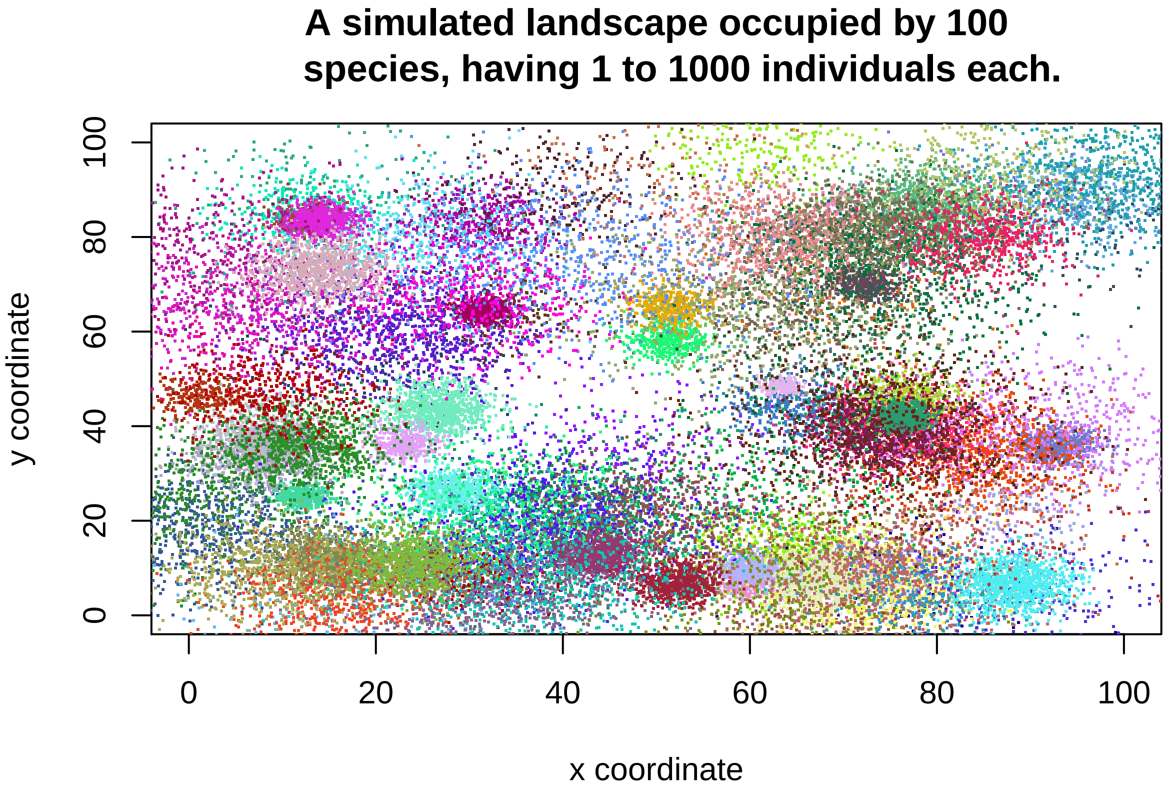
So far, we have discussed spatial autocorrelation and aggregation as core concepts of spatial diversity. Likewise, we have introduced and examined primary patterns for both of those concepts. Here, we introduce a primary pattern of diversity related to spatial scale. We are going to construct one of ecology's oldest and most intensively studied patterns, i.e., the **Species-Area Relationship (SAR)**.

The SAR reveals the rate at which we discover species with greater sampling area (i.e., extent). It may seem obvious that, if starting from a very small local scale and increasing our sample area, that we will inevitably encounter more species. Howevever, SARs often take a particular form. Arrhenius (1921) first described the general form of the *species-area relationship (SAR)* as a power-law: $S = cA^z$ where S is species richness and A is area. Arrhenius's power-law predicts a rate of increase in scales of richness that is approximately linear across scales of space. That is, $log(S) = log(c) + zlog(A)$, where z is the scaling exponent.

Power-laws like the SAR can be used to make predictions of diversity across scales, and also point to mechanisms and processes that operate in similar ways across scales.

Here, we'll use the following code to reinforce the concept of the SAR and to demonstrate how one would generate and analyze this pattern. Suppose we simulate the spatial distribution of a community with 100 species, letting each species have between 1 and 1,000 individuals.

```
community <- c() # an initiall empty community
species <- c()   # with zero species

# 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',
     main='A simulated landscape occupied by 100
     species, having 1 to 1000 individuals each.')

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)
  ab <- sample(1000, 1)
  x <- rnorm(ab, mean = runif(1, 0, 100), sd = std)
  y <- rnorm(ab, mean = runif(1, 0, 100), sd = std)
  color <- c(rgb(runif(1),runif(1),runif(1)))

  points(x, y, pch=".", col=color)
  species <- list(x, y, color)
  community[[length(community)+1]] <- species
  }
```

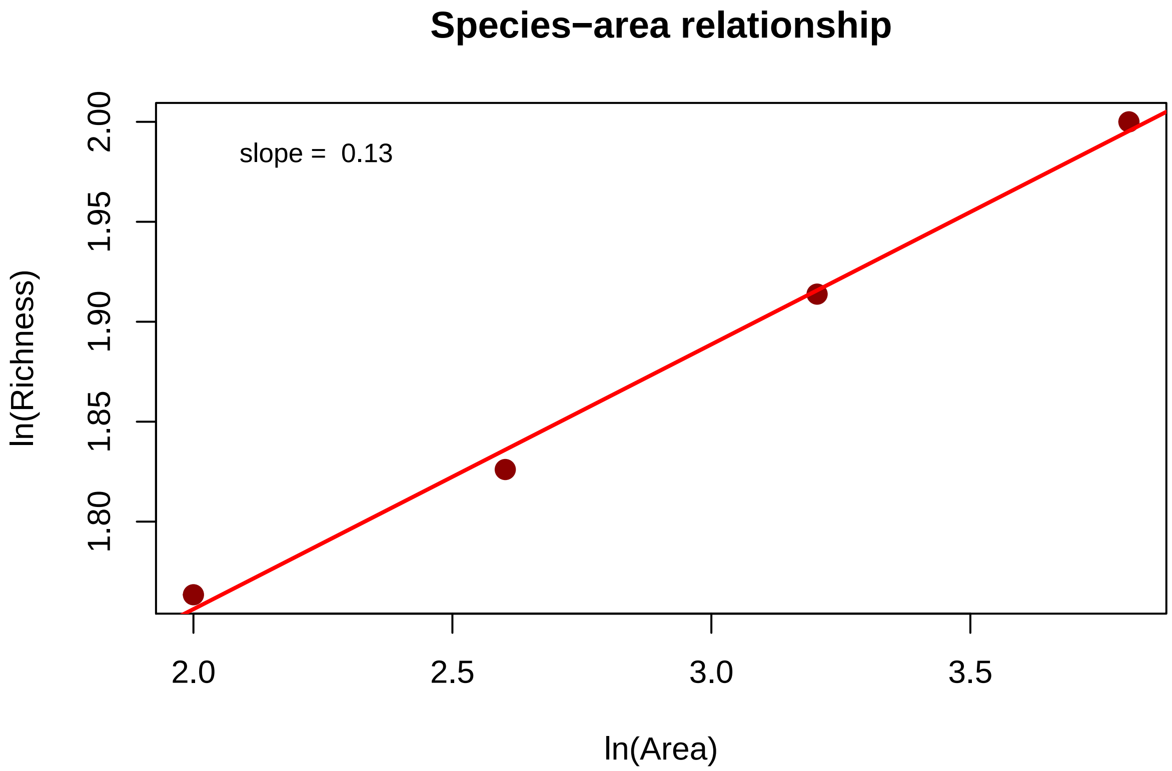


Having generated a simulated landscape occupied by 100 species, we can now examine how richness scales with increasing area (i.e., extent).

```
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

while (lim <= 100){ # while the spatial extent on the x and y 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 samp
        }
      }
    }
  S.list <- c(S.list, log10(S))
  A.list <- c(A.list, log10(lim^2))
  lim <- lim * 2  # increase the extent multiplicately
}
```

Having populated our lists for species richness (i.e., S.list) and areas (i.e., A.list), let's examine our 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)
```



In our simulation, the slope or z-value was near 0.2. This means that when area increases from say 10 to 100, the richness increases from say, 5 to 13.

The fact that we accumulate species with increasing area is not necessarily interesting. In fact, we just showed that we can expect a positive SAR as a result of random sampling. However, for more than a decade it was believed that the SAR of bacteria and other microorganisms was essentially flat.

GLOSSARY

Because you may not be familiar with geographical systems and spatial analyses, we provide a few key terms to help you quickly get up to speed.

datum:

projection:

spatial: Relating to one or more dimensions of space resulting in aspects of distance, area, and volume

geographical: Relating to physical, environmental, and biological features that closely follow spatial patterns

scale: The resolution that characterizes an observation or study, and that results from a relationship between extent and grain

extent: The greatest distance considered in an observation or study

grain: The smallest or primary unit by which extent the is measured

GIS: Geographical Information Systems.

shapefile: One of two primary file formats for working with geographical data. In short, shapefiles generally store information that is displayed as polygons, e.g. habitat patches.

raster file: The other primary geographical file format. Raster files contain highly pixelated information on specific points or precise locations, e.g., elevation data.

land cover: The physical material at the surface of the earth and includes grass, asphalt, trees, bare ground, water, etc. Landcover gives us some of the most precise accounting for what general environments occupy large expanses of geographic areas.

REFERENCES

1) Tobler W., (1970) “A computer movie simulating urban growth in the Detroit region”. Economic Geography, 46(2): 234-240.

2) Arrhenius (1921)

3) Lomolino (