

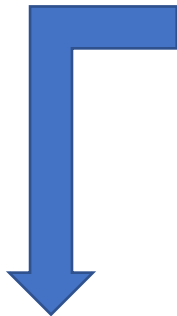
Determine species ranges

- Download Barone's data for all 3 transects
- Create a .csv file with species from Sonadora dataset
 - Columns names: Family, Genus, Species, var/spp, SpeciesCode, low, high, range
- Fill in family genus and species names based on SpeciesCode given
- Input upper and lower range limits based on data from Barone's 3 transects
- Adjust ranges based on species presence in LTER plot
 - 350 as lower limit, 400 as upper limit
- Adjust ranges based on Axelrod's book
- Adjust taxonomy to match updated taxonomy in Axelrod's book
 - Calculated range size in Excel
 - =high-low



Place species into range quantiles and make sure they are even

- `quantile(barone.ranges$range, seq(0, 1, by=.2))`
 - Input=column with ranges
 - Output=range that each 20% of the data is <= to
- Determine the number of species in each quantile
 - `sum(barone.ranges$range<=500)`
 - `sum(barone.ranges$range>500 & barone.ranges$range<=734)`
 - `sum(barone.ranges$range>734 & barone.ranges$range<=848)`
 - `sum(barone.ranges$range>848 & barone.ranges$range<=950)`
 - `sum(barone.ranges$range>950)`
 - Input=column with ranges and the range given as the division between quantiles
 - Output=number of species in each quantile to determine if they are even



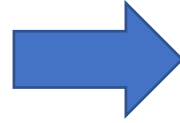
Turn data into a community matrix with plots as rows and quantiles as columns

- `cdm=matrix(data=NA, ncol=5, nrow=16)`
- `colnames(cdm)=c("Q1", "Q2", "Q3", "Q4", "Q5")`
- `rownames(cdm)=c(250, 300, 350, 400, 450, 500, 550, 600, 650, 700, 750, 800, 850, 900, 950, 1000)`
`for(i in 1:length(listy)){ elevation=subset(Sonadora, Sonadora$plotElevation==listy[i]) sp.name=subset(elevation, !duplicated(elevation$stemSpeciesCode)) cdm[i,]=table(sp.name$Quantile)`
 - Input=complete data.frame from original data
 - Output=community data matrix with the number of species from each elevation plot that are in each quantile category.



Transform trait data to approximate a normal distribution of each trait, scale the data

- `Traits.scaled=apply(log(traits), MARGIN=2, scale)`
- Input=original trait data
- Output=log transformed trait data



Perform a PCA on traits and construct traits

- `PCA=princomp(traits)`
- Input=transformed trait data
- Output=pc.scores of the axes that explain 90-95% of the variance
- Can use these pc.scores to calculate an distance matrix
 - `Pc.dist.mat=dist(pc.scores, method="Euclidean")`



Generating trait dendrograms

- `Dist.matrix=dist(pca.scores, method="euclidean")`
 - `Dendro=hclust(dist.matrix, method="average")`
- For traits separately
- `Trait.1=as.matrix(traits[,1])`
 - `Rownames(trait.1)=rownames(traits)`
 - `Dendro.trait.1=hclust(dist(trait.1, method="euclidean"), method="average")`
 - Input=distance matrix calculated from pca scores
 - Output=dendrograms for each trait and then using pca scores



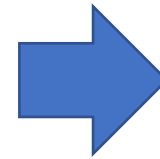
Construct trait data matrix to match species list

- `tdm=matrix(data=NA, nrow=5, ncol=5)`
- `colnames(tdm)=c("trait1", "trait2"...)`
- `rownames(tdm)=c("Q1", "Q2", "Q3", "Q4", "Q5")`
- Cells of tdm will contain trait values for species within each quantile
 - Make a list of the species code within the quantile
 - `Listy=c("sp1", "sp2",...)`
 - Subset the original trait data for the rows of only species in listy object
 - `Q1=traits[listy,]`
 - Determine the mean of the trait values for only those species
 - `Output=colMeans(Q1)`
 - Put the output of the trait means into the rows of the trait data matrix
 - `Tdm[1,]=output`
- Input=transformed trait data.csv
- Output=trait data matrix with means trait values of each trait of species in each quantile



Calculate observed values for functional diversity

- Distance matrix from dendrograms
 - `Dendro.dist.mat=cophenetic(dendro)`
- Distance matrix from raw trait data, input distance matrix from `pca.scores`
 - `Square.dist.mat=as.matrix(pca.dist.matrix)`
- `Mpd(cdm, square.dist.mat, abundance.weighted=F)`
- `Mpd(cdm, square.dist.mat, abundance.weighted=T)`
- `Mntd(comm.data, dist.mat, abundance.weighted=F)`
- `Mntd(comm.data, dist.mat, abundance.weighted=T)`
- Calculate functional richness, `pca` performed within the function
 - `FRic=dbFD(traits.scaled, cdm)$FRic`
- Input=`pca` distance matrix, dendrograms, raw trait data
- Output=observed values of `mpd`, `mntd`, and `FRic`



Null Model

- Randomize community data matrix to alter which species are in each range size quantile
- Is species richness in range size along elevation nonrandom with respect to function?
- Randomly assemble species richness in each plot, would select a quantile at random without replacement
- Constrained=fix row sums since lower elevations have higher richness and fix column sums since more species have smaller ranges
- Independent swap null models