This lab introduces a new set of techniques that are based on distance matrices that indicate similarity between observations. All these techniques require a two step process: (1) create a matrix of similarity between observations, choosing among dozens of ecological, genetic, or mathematical distance measures, and (2) using this matrix to ordinate or cluster the observations, choosing again among dozens of ordination and clustering techniques. This leaves you with \( (\text{dozens})^2 \) possible ways to view your data.

There is no perfectly objective way to guide the choice of distance matrix and ordination and clustering techniques. Generally, Euclidean and Mahalanobis distances work well for normally distributed environmental data. The Bray-Curtis distance is a favorite for community ecologists to describe similarity in species composition in sample plots. This is a very robust distance measure: spot-check similar and dis-similar plots in your dataset. The measure just seems to work on anything.

Once your choice of distance measure is made, it is good to explore several techniques: if they give you completely different answers, then likely your observations do not fall into clear groups. You can test for significant differences among your groups (next lab) to confirm the validity of your classification.

### 5.1. Creating and importing distance matrices

Download the dataset `AB_Climate_Means.csv` from the course website. These are means of climate variables for ecosystem (natural subregions of Alberta). We want to see how similar these ecosystems with respect to multiple climate variables (Note: variable explanations on last page!).

- To set yourself up in R as usual, starting R from an empty workspace in a working directory, and import the dataset `AB_Climate_Means.csv`. Check if the data imported correctly:

  ```r
  dat1 = read.csv("AB_Climate_Means.csv")
  head(dat1)
  ```

- Next, we have to make a slight modification to this data table. To properly label our observation, we have to assign row names to our table and delete the data column “ECOSYS”. We have to do this because otherwise R thinks that ECOSYS is numerical data rather than a label to keep track of our observations:

  ```r
  rownames(dat1) = dat1$ECOSYS; # creates rownames
  ecolabels = dat1[,1:2] # creates labels that we need later
  dat1 = dat1[,3:10] # keeps columns 3-10 and drops column 1-2
  head(dat1) # check it
  head(ecolabels) # check it
  ```

- Next, we use the distance function to create distance matrices. Instead of “euclidean”, you can also use these distance measures of the `dist()` function "maximum", "manhattan", "canberra", "binary" or "minkowski". Further, you can exaggerate or reduce the relative importance of the largest distances with transformations of the distance matrix. Squares or the square root are commonly used:

  ```r
  euclid = dist(dat1, method ="euclidean")
  euclid_sq = euclid^2
  euclid_sqrt = sqrt(euclid)
  ```
• For ecologists, there are other important distance measures. For this, you have to install the R package “ecodist”, which allows you to calculate Bray-Curtis and Mahalanobis distances. Install the package by choosing the menu “Packages” > “Install Packages”. Then, select a download location nearby and pick the package “ecodist”.

• The `distance()` function of “ecodist” actually calculates the squared Mahalanobis, so you have to take the square root to get the original.

```r
library(ecodist)
braycurtis=bcdist(dat1)
mahal_sq=distance(dat1, "mahal")mahal=sqrt(mahal_sq)
```

• You may also need to calculate distance matrices outside R and import them for analysis. There are, for example, specialized distance measures for geneticists to describe similarities in the DNA sequences, etc. In order to use a matrix that you generate outside of R as distance matrix, you have to define it as a distance matrix in R before you can work with it after import. This works according to the following sample code if you ever have to do this:

```r
custom1=read.csv("Custom_Distance_Matrix.csv")
head(custom1) # check it
rownames(custom1)= custom1$ID; # row name modification as above custom1=custom1[,-1] # drop ID column custom2=as.dist(custom1) # conversion to distance matrix head(custom1) # check it
```

5.2. Cluster Analysis

OK, now that you know everything about generating and importing distance matrices, let’s use them for building dendrograms. Below, I call the distance matrix uniformly “dm”. You have to replace this with the distance matrix of your choice that you generated above.

• The code below builds dendrograms. `hclust` stands for hierarchical cluster analysis. There are many methods. Ward’s method and the “centroid” method are widely used and usually yields reasonable results, but you may try: “single”, “complete”, “average”, “mcquitty”, “median” or “centroid”.

```r
tree=hclust(dm, method="ward")plot(tree, hang=-1, main="Alberta Natural Subregions")
```

• Get a feel for the robustness of cluster analysis by trying various distances, transformations, and clustering methods. Save a few dendrograms and make up your mind if they make biological sense. For reference, the Alberta Natural Subregion system is given on the last page. For this dataset, the Mahalanobis distance is likely appropriate because some of the climate variables are correlated.
5.3. Cluster Analysis with significance testing

You can run a bootstrap-version of cluster analysis that evaluates how consistently the same clusters appear over hundreds or thousands of runs with a sub-sampled dataset.

- Install the package `pvclust` and try out the following code.

```r
dat1 = read.csv("AB_Climate_Means.csv")
head(dat1)
rownames(dat1) = dat1$ECOSYS; # creates rownames
dat1 = dat1[,3:10] # keeps columns 3-10 and drops column 1-2
head(dat1) # check it

dat1t = t(dat1) # transpose dataset
head(dat1t) # check it

# Install package "pvclust"
library(pvclust)

# Be patient the next step takes some time
tree = pvclust(dat1t, method.hclust="ward", nboot=1000,
method.dist="euclidean")

# Create the dendrogram with p values
plot(tree, hang=-1, main="Alberta Natural Subregions")

# add rectangles around groups highly supported by the data
pvrect(tree, alpha=.95)
```

- Clusters with high AU values, for example >0.95, are strongly supported by the data. This means they really are very similar units that form a natural cluster. AU means “Approximately Unbiased P-value” whereas BP refers to raw “Bootstrap Probabilities” before statistical adjustments.

For interpretation check what the ecosystem abbreviations stand for on the last page.
5.4. Nonmetric multidimensional scaling (NMDS)

Instead of using a dendrogram, we can also use ordination techniques. NMDS is a very robust technique for all kinds of normally and non-normally distributed data, including presence/absence data. Similarity is implied by proximity:

```r
library(ecodist)
dm = distance(dat1, "mahal") # calculates squared Mahalanobis

nmds_out = nmds(dm, mindim=2, maxdim=2) # runs NMDS
scores = nmds.min(nmds_out) # generate scores
nmds_out$stress # the last value indicates the final stress
```

- The stress value by itself is not informative, but it should be stable (i.e. for the last permutations, and you should look at the stress values of 1, 2, 3, 4, 5, 6 or so dimensions (modify maxdim). A scree-plot of stress values over the number of dimensions will tell you how many dimensions you need to consider. Let’s look at the first 2 dimensions.

- In the plot below, we color by biome, guided by the first command that shows the order of biomes. Then create a color legend in the same order, and plot with ecosystem labels. The +0.8 offsets the labels, so you can still see the points. Vectors are fitted as usual.

```r
sort(unique(ecolabels$BIOME))
mycol = c("green", "yellow", "purple", "orange")
plot(scores, pch=19, col=mycol[ecolabels$BIOME])
text(scores+0.8, labels=ecolabels$ECOSYS)
vectors = vf(scores, dat1, nperm=10)
plot(vectors, len=0.1, col="red")
```

- The overlay of vectors indicate how the original variables are correlated with observations. For example the “DMG” ecosystem (Dry Mixed Grass) is associated with high mean annual temperature (MAT) and dryness (AHM).

- You can guide the reader by adding ellipses that represent groups. Install the vegan library if it’s not already on the lab computers for this. The basic command is quite simple: `ordiellipse(scores, ecolabels$BIOME)`, but you need a loop to color the ellipses, drawing one at a time and repeating the process four times for each biome. The option `conf` allows to scale the size of the ellipses to your liking:

```r
library(vegan)
mygroups = sort(unique(ecolabels$BIOME))
mycol = c("green", "yellow", "purple", "orange")
ordiellipse(scores, ecolabels$BIOME, conf=0.6, col=mycol, show.groups=mygroups)
```
If you run the NMDS code multiple times, you will notice that the plots differ substantially in appearance. A big part of this is that the canvas is randomly rotated to a new angle every time. As a consequence, loadings and variance explained by X1 and X2 are not particularly informative. Variance explained by X1 and X2 is not informative for another reason: NMDS is not an orthogonal rotation, and because X1 and X2 are not at right angles (uncorrelated), only the cumulative variance explained should be reported.

Besides the random rotation, you will also see that the sample points shift relative to each other and relative to the vectors by small amounts (hopefully). Thus, NMDS really provides a different ordination every time you run it. There is nothing inherently wrong with this. It is equivalent to looking at your dataset from slightly different angles as in PCA versus FACTOR analysis. If you have a run that has substantially lower stress than all others, you should probably pick that one. Otherwise, you are free to choose a run that you like for esthetic reasons.

5.5. MetaMDS

If you feel uncomfortable with a subjective decision what run you like best or what run is easiest to interpret, the vegan package offers a metaMDS function that executes multiple runs and looks for stable configurations. One advantage is that this package produces repeatable results in that you get exactly the same ordination output every time.

- In addition to the number of dimensions (k), you can also specify the number of random start configurations (trymax) that metaMDS executes in search of a stable solution. If it does not find a stable solution, it will use PCA as starting point for consistency. Check ?metaMDS for all options in this package. You can see that metaMDS will also calculate distance matrices for you, but for better control and transparency let’s keep doing it manually:

```r
dat1 = read.csv("AB_Climate_Means.csv")
rownames(dat1) = dat1$ECOSYS
ecolabels = dat1[,1:2]
dat1 = dat1[,3:10]

library(ecodist)
dm = sqrt(distance(dat1, "mahal"))

library(vegan)
out1 = metaMDS(dm, k=2, trymax=500)
out1

mycol = c("green", "yellow", "purple", "orange")
plot(out1$points, pch=19, col=mycol[ecolabels$BIOME])
text(out1$points-0.1, labels=ecolabels$ECOSYS)

vectors = vf(out1$points, dat1, nperm=10)
plot(vectors, len=0.1, col="red")

mygroups = sort(unique(ecolabels$BIOME))
for (i in 1:4){
  ordiellipse(out1$points, ecolabels$BIOME, conf=0.6, col=mycol[i],
             show.groups=mygroups[i])
}
```
5.6. Principal Coordinate Analysis (PCoA)

Another well regarded and well behaved ordination technique is Principal Coordinate Analysis (PCoA). Like NMDS it is an ordination that is based on a distance matrix of your choice, so you are free to use a distance measure that is suits your data, i.e. Bray-Curtis for species community data, which I am using in the example below (just because I can, and because I want to use something that's non-Euclidean, which is actually the point of this analysis).

The difference to NMDS is that I do not constrain this ordination to just few or two dimensions. Instead, I allow as many dimensions as required to honor the relative position of all points in my dataset, and that is (to be mathematically 100% on the safe side), n-1 dimensions. So, if you have a dataset with n=50 observations (rows) and 10 variables (columns), we allow 49 dimensions for this ordination.

This makes the ordination procedure a no-brainer because I don’t need to make any compromises. There is zero stress in my ordination and no complicated algorithms to minimize stress are required. That said, if your objective is to reduce complexity, we have just made things substantially worse, going from 10 original dimensions (or variables) to 49.

However, now comes the party trick in the second step: I run a regular PCA on that 49 dimensional ordination. So you can think of PCoA as a representation of non-Euclidean data (i.e. since you normally would not use Euclidean distances for your distance matrix) in a Euclidean space.

- Here is the full code to run a PCoA with all customizations that we got used to, rather than using the quick and dirty biplot functions. Sometimes, you may run into trouble with negative Eigenvalues when trying to rotate your n-1 dimensional distance matrix, and there are some corrections available for that (see the ?pcoa help file for references and details):

```r
dat1 = read.csv("AB_Climate_Means.csv")
rownames(dat1) = dat1$ECOSYS
ecolabels = dat1[,1:2]
dat1 = dat1[,3:10]

library(ecodist)
dm = bcdist(dat1)

library(ape)
out1 = pcoa(dm, correction="none")
out1

mycol = c("green", "yellow", "purple", "orange")
plot(out1$vectors[,1:2], pch=19, col=mycol[ecolabels$BIOME])
text(out1$vectors[,1:2]-0.003, labels=ecolabels$ECOSYS)

vectors = vf(out1$vectors[,1:2], dat1, nperm=10)
plot(vectors, len=0.1, col="red")

library(vegan)
mygroups = sort(unique(ecolabels$BIOME))
mycol = c("green", "yellow", "purple", "orange")
for (i in 1:4){
  ordiellipse(out1$vectors[,1:2], ecolabels$BIOME,
              conf=0.6, col=mycol[i],
              show.groups=mygroups[i])
}
```
Some Reference information, so that you can interpret the results more easily:

**Abbreviations of Ecosystems:**

**Mountains**
- Alpine: A
- Subalpine: SA
- Montane: M
- Upper Foothills: UF
- Lower Foothills: LF

**Grasslands**
- Dry Mixedgrass: DMG
- Mixedgrass: MG
- Northern Fescue: NF
- Foothills Fescue: FF

**Parklands**
- Foothills Parkland: FP
- Central Parkland: CP
- Peace River Parkland: PRP

**Boreal Forest**
- Dry Mixedwood: DM
- Central Mixedwood: CM
- Lower Boreal Highlands: LBH
- Upper Boreal Highlands: UBH
- Athabasca Plain: AP
- Peace–Athabasca Delta: Peac
- Northern Mixedwood: NM
- Boreal Subarctic: BSA

**Abbreviations of Climate Variables:**

- MAT: mean annual temperature (°C)
- MWMT: mean warmest month temperature (°C)
- MCMT: mean coldest month temperature (°C)
- TD: temperature difference between MWMT and MCMT, or continentality (°C)
- MAP: mean annual precipitation (mm)
- MSP: mean annual summer (May to Sept.) precipitation (mm)
- AHM: annual heat:moisture index \((\text{MAT}+10)/(\text{MAP}/1000))\)
- SHM: summer heat:moisture index \((\text{MWMT})/(\text{MSP}/1000))\)
- DD0: degree-days below 0°C, chilling degree-days
- DD5: degree-days above 5°C, growing degree-days
- NFFD: the number of frost-free days
- FFP: frost-free period
- BFFP: the Julian date on which FFP begins
- EFP: the Julian date on which FFP ends
- PAS: precipitation as snow (mm)