Genetic diversity in a wheat collection

M. Humberto Reyes-Valdés

June 3, 2019

CABANA workshop

Genomic analysis of crop diversity using R

In this tutorial, we'll analyze the genetic diversity in the Mexican wheat collection. The data is a table of 7,986 wheat lines and 1,102 informative alleles. The data frame has a low missing data rate.

Import data

```
#Set working directory
setwd("~/cursos/cabanaIrapuato/lectures")
#Import data
dat<-read.csv("tables/CleanWheat.csv",head=T)
dim(dat)
```

```
## [1] 1102 7988
```

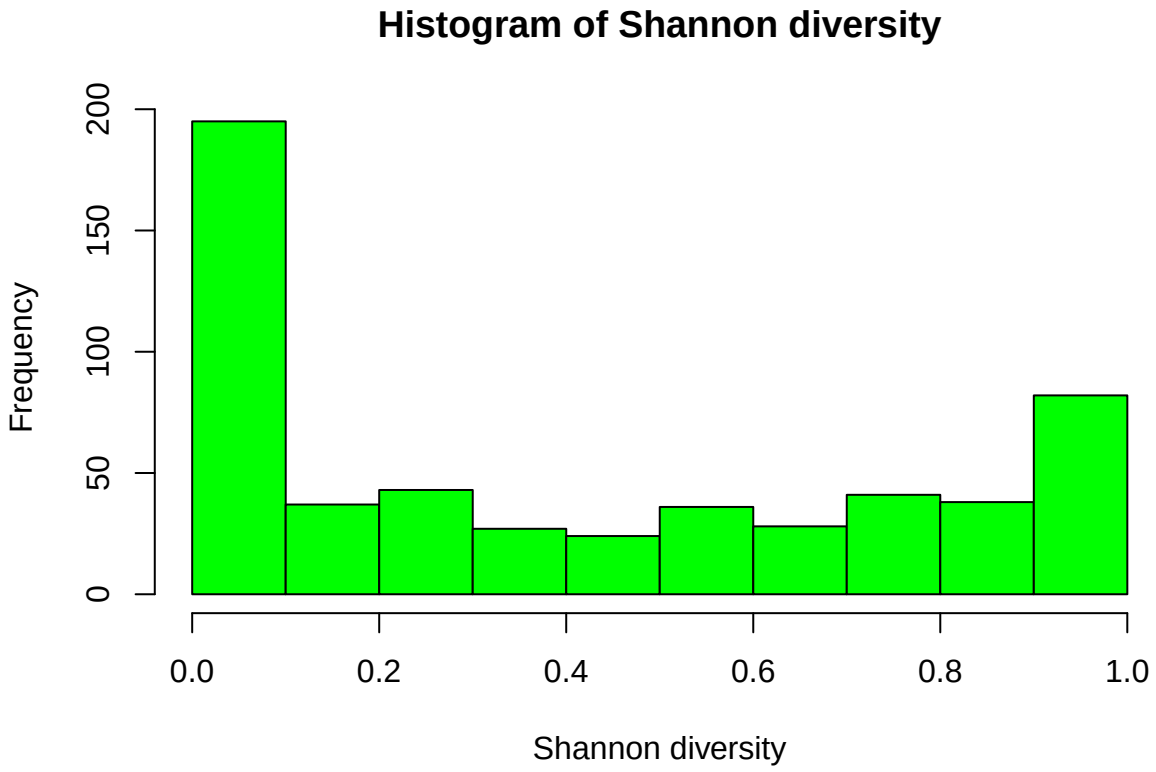
Shannon diversity

Now, we will use the Shannon entropy as a measure of allelic diversity in the population of wheat lines. Since the table comprises 551 SNP loci, an entropy per locus will be calculated, and then we will take the average, as a global measure of allelic diversity.

```
#A function for Shannon entropy
#x log x
MyLog2p<-function(x){if(x==0) 0 else x*log(x,2)}
#The entropy function
entropy<-function(x){-sum(sapply(x,MyLog2p),na.rm=T)}
#Remember that alleles are in rows
#and wheat lines are in columns
#Take the row averages as allelic frequencies
freq<-apply(dat[-c(1,2)],1,function(x) mean(x,na.rm=T))
#Per locus entropy
a<-NA;length(a)<-dim(dat)[1]/2;for(i in 1:length(a)){a[i]<-entropy(freq[c(2*i-1,2*i)])}
head(a)
```

```
## [1] 0.9999903 0.9993995 0.9934804 0.9971753 0.9915479 0.9985042
```

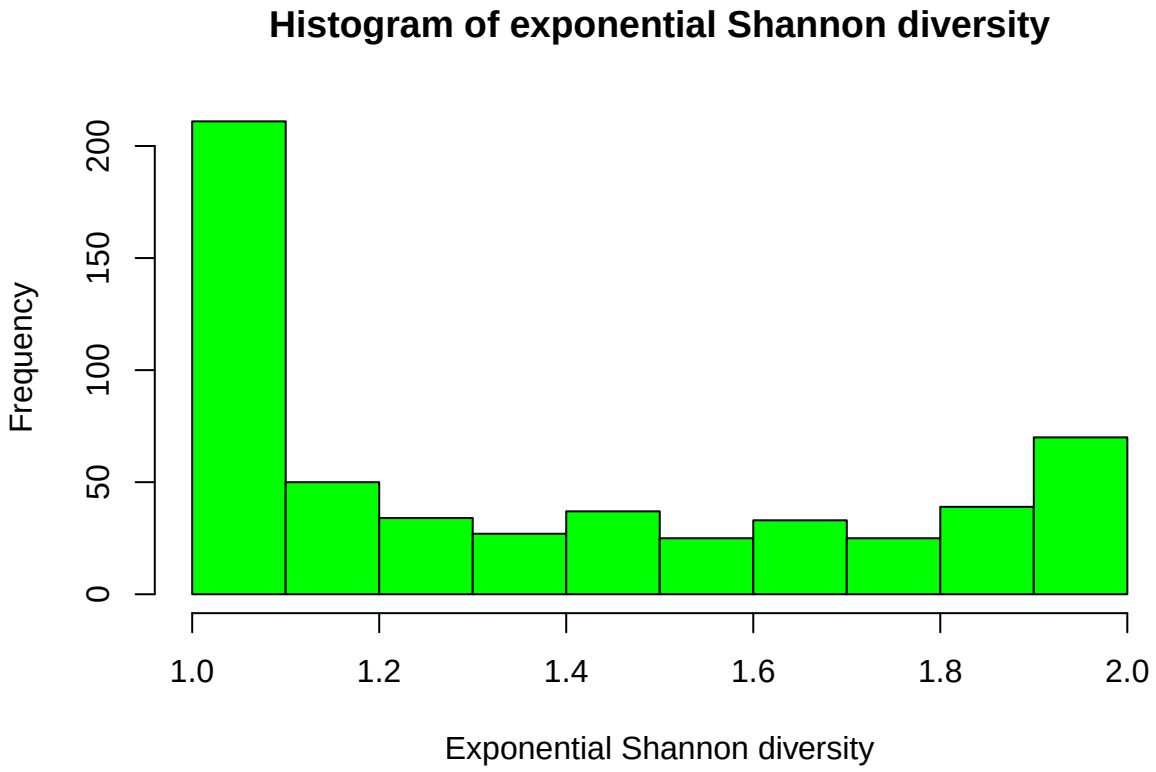
```
#Distribution of Shannon diversities across loci
hist(a,xlab="Shannon diversity",col="green",main="Histogram of Shannon diversity")
```




```
#Average entropy
entropy(freq)/(length(freq)/2)
```

```
## [1] 0.4035897
```

```
#Exponential Shannon diversity
hist(2^a,xlab="Exponential Shannon diversity",
col="green",
main="Histogram of exponential Shannon diversity")
```




```
#You can interpret this as the effective number of alleles per locus
#Average exponential Shannon diversity
mean(2^a)
```

```
## [1] 1.366351
```

Here comes an important question: is this the genetic diversity among wheat lines? In general not. It is the allelic variation in the sample. But, can we calculate the Shannon diversity among lines?

We can do that by treating data as allele present/absent (actually the original layout), then calculating the diversity of the categorical variable.

```
binary<-dat[-c(1,2)]
myf<-function(x) ifelse(x>0,1,0)
binary<-apply(binary,2,myf)
binary[1:10,1:10]
```

```
##       SEEDDIV2819 SEEDDIV2891 SEEDDIV2899 SEEDDIV2907 SEEDDIV2827
##  [1,]           0           1           1           1           0
##  [2,]           1           0           1           0           1
##  [3,]           1           1           0           0           0
##  [4,]           0           1           1           1           1
##  [5,]          NA           1           0           1           0
##  [6,]          NA           0           1           0           1
##  [7,]           0          NA          NA           1          NA
##  [8,]           1          NA          NA           0          NA
##  [9,]           0           0           0           0           0
## [10,]           1           1           1           1           1
##       SEEDDIV2835 SEEDDIV2843 SEEDDIV2851 SEEDDIV2859 SEEDDIV2867
##  [1,]           1           0           1           0           0
##  [2,]           0           1           0           1           1
##  [3,]           0           0           1           1           1
```

```
##  [4,]           1           1           0           0           0
##  [5,]           0           1           1           1           0
##  [6,]           1           0           0           0           1
##  [7,]          NA          NA          NA          NA          NA
##  [8,]          NA          NA          NA          NA          NA
##  [9,]           0           0           1           0           0
## [10,]           1           1           1           1           1
```

```
binary<-as.data.frame(binary)
#Entropy per row
row.entropy<-function(x){
    a<-table(x)
    a<-a/sum(a)
    entropy(a)
}
#Vector of entropies
r.entropies<-apply(binary,1,row.entropy)
#Average
mean(r.entropies)
```

```
## [1] 0.4018206
```

Why is it so close to the average Shannon entropy of allele frequencies? Because they are mostly homozygous lines.

Ginni-Simpson index

As we have seen, the Ginni-Simpson index is a measure of allelic diversity, that in a population in Hardy-Weinberg equilibrium represents the heterocigosity.

```
#Function for the Gini-Simpson index
gs<-function(x){1-sum(x^2,na.rm=T)}
#Per locus GS
a<-NA;length(a)<-dim(dat)[1]/2;for(i in 1:length(a)){a[i]<-gs(freq[c(2*i-1,2*i)])}
mean(a)
```

```
## [1] 0.1772055
```

```
#There is an equation for unbiased GN
#We need to count the number of observations for each locus
n<-apply(dat[-c(1,2)],1,function(x) sum(!is.na(x)))
a<-NA;length(a)<-dim(dat)[1]/2;for(i in 1:length(a)){a[i]<-gs(freq[c(2*i-1,2*i)])*n[2*i]/(n[2*i]-1)}
mean(a)
```

```
## [1] 0.177228
```

As we can see, the average obtained by the raw Ginni-Simpson index is almost identical to the average of the unbiased GS. The reason: we have a large sample of wheat lines.

Allele richness

Defined as the total number of alleles in the sample, across loci, AR can be readily obtained as follows:

```
sum(freq>0)
```

```
## [1] 1102
```

Notice that is the same as the number of rows in the table. The reason is that all alleles are informative.

Prepare a data subset

The following analyses are system demanding. For the sake of time, will work with only a subset of 100 wheat lines. The same methods apply to the whole set of lines, but they would take a long time.

If we select a subset of lines, some loci can become **noninformative**. Thus, after line selection, we select for informative alleles and with low NA rates. In the last related exercise, we did this from scratch. To make life easier, I prepared a code called selector.R, that will do this without much coding.

```
#Prepare a data subset for the workshop
sample<-dat[c(1:102)] #First 100 lines
dim(sample)
```

```
## [1] 1102  102
```

```
#Clean the sample: Only informative loci, with a maximum NA rate of 50%
#Use selector.R (by HR)
source("selector.R")
sample<-selector(sample,na.rate=0.5)
dim(sample)
```

```
## [1] 596 102
```

Average distance

Average distance is a metrics that we can use to assess diversity among lines. In this case we will use the average euclidean distance, based on allele frequencies.

```
#Average distance
sam<-sample[-c(1,2)]  #Take identifiers out
#Wheat lines are in columns
#We need to transpose
sam<-t(sam) #Important: dist calculates distance between rows
d<-dist(sam) #Distance object
mean(d) #Average distance
```

```
## [1] 11.55601
```

Principal component analysis

PCA is a technique widely used to assess the structure of diversity in sets of cultivars. The idea is to reduce the dimensionality of the data set. If we consider each allele as one characterizing variable, then we have 1102 dimensions. We cannot graphically represent the lines in a 1102-dimensional space. Thus, we will use the first two principal components, which are uncorrelated variables that capture a considerable amount of variance, to have a two-dimensional representation of the lines.

One problem with PCA is the presence of missing data cells. Thus, we will resort to imputation through a machine learning method called knn, implemented in the package **impute**.

```
library(impute)
for.pc<-sample[-c(1,2)]
imputed.for.pc<-impute.knn(as.matrix(for.pc))
```

```
imputed.for.pc<-imputed.for.pc$data
for.pc[1:10,1:10]
```

```
##    SEEDDIV2819 SEEDDIV2891 SEEDDIV2899 SEEDDIV2907 SEEDDIV2827 SEEDDIV2835
## 1            0         1.0         0.5           1           0           1
## 2            1         0.0         0.5           0           1           0
## 3            1         0.5         0.0           0           0           0
## 4            0         0.5         1.0           1           1           1
## 5           NA         1.0         0.0           1           0           0
## 6           NA         0.0         1.0           0           1           1
## 9            0         0.0         0.0           0           0           0
## 10           1         1.0         1.0           1           1           1
## 11           0         0.0         1.0           1          NA           1
## 12           1         1.0         0.0           0          NA           0
##    SEEDDIV2843 SEEDDIV2851 SEEDDIV2859 SEEDDIV2867
## 1            0         1.0           0           0
## 2            1         0.0           1           1
## 3            0         1.0           1           1
## 4            1         0.0           0           0
## 5            1         1.0           1           0
## 6            0         0.0           0           1
## 9            0         0.5           0           0
## 10           1         0.5           1           1
## 11           1         1.0           1           1
## 12           0         0.0           0           0
```

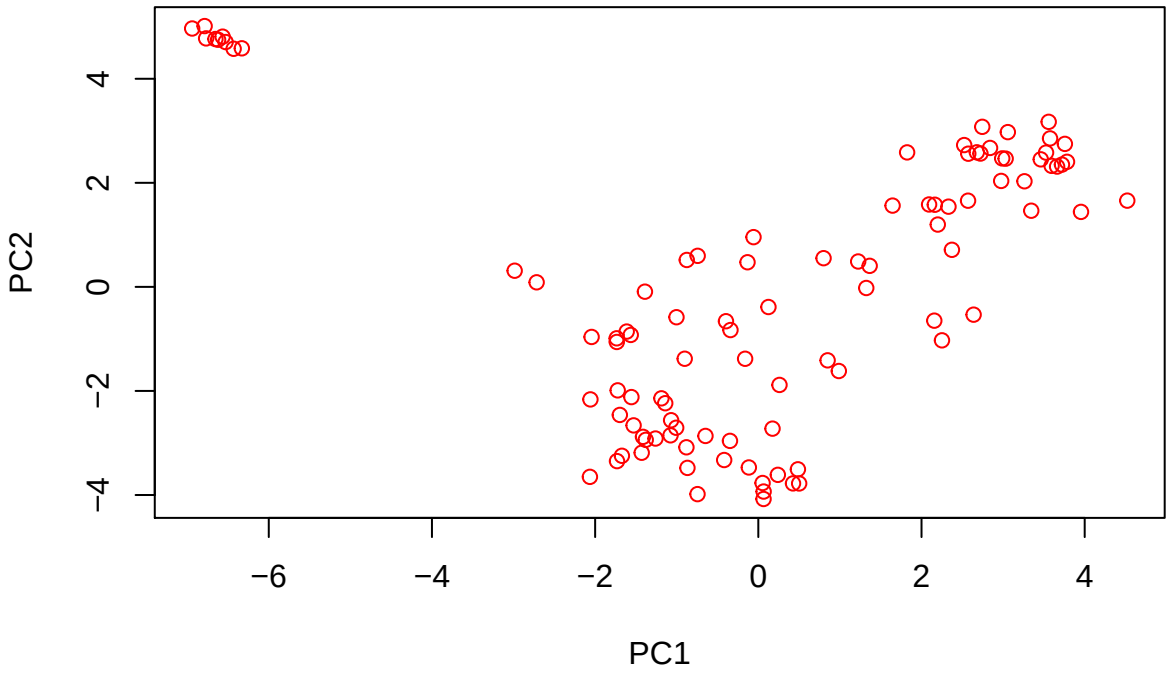
```
imputed.for.pc[1:10,1:10]
```

```
##    SEEDDIV2819 SEEDDIV2891 SEEDDIV2899 SEEDDIV2907 SEEDDIV2827 SEEDDIV2835
## 1          0.0         1.0         0.5           1         0.0           1
## 2          1.0         0.0         0.5           0         1.0           0
## 3          1.0         0.5         0.0           0         0.0           0
## 4          0.0         0.5         1.0           1         1.0           1
## 5          0.6         1.0         0.0           1         0.0           0
## 6          0.4         0.0         1.0           0         1.0           1
## 9          0.0         0.0         0.0           0         0.0           0
## 10         1.0         1.0         1.0           1         1.0           1
## 11         0.0         0.0         1.0           1         0.7           1
## 12         1.0         1.0         0.0           0         0.3           0
##    SEEDDIV2843 SEEDDIV2851 SEEDDIV2859 SEEDDIV2867
## 1            0         1.0           0           0
## 2            1         0.0           1           1
## 3            0         1.0           1           1
## 4            1         0.0           0           0
## 5            1         1.0           1           0
## 6            0         0.0           0           1
## 9            0         0.5           0           0
## 10           1         0.5           1           1
## 11           1         1.0           1           1
## 12           0         0.0           0           0
```

```
imputed.for.pc<-t(imputed.for.pc)
#Principal components
pc<-prcomp(imputed.for.pc)
names(pc)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

```
plot(pc$x[,1],pc$x[,2],col="red",xlab="PC1",ylab="PC2")
```



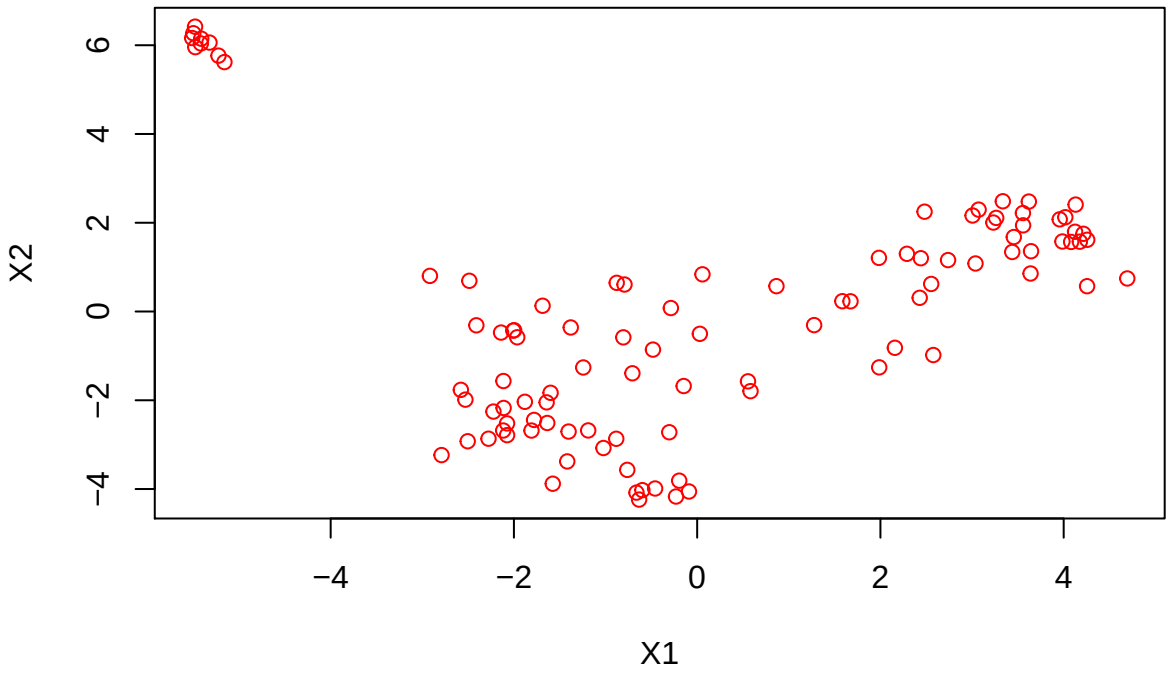

```
#Which are those isolated lines?
x<-as.data.frame(pc$x)
a<-x[x$PC1<-4 & x$PC2>4,]
row.names(a)
```

```
## [1] "SEEDDIV2899" "SEEDDIV2893" "SEEDDIV2846" "SEEDDIV2904" "SEEDDIV2848"
## [6] "SEEDDIV2898" "SEEDDIV2834" "SEEDDIV2945" "SEEDDIV2969"
```

Multidimensional scaling

Another method to reduce the dimensionality of a data representation is Multidimensional Scaling (MDS) also called Principal Coordinate Analysis (PCoA). This approach attempts to represent as better as possible multidimensional distances in a two- or three-dimensional plot. To apply this method we do not need imputation.

```
d<-dist(sam)
#were sam is the tanspose of the allele
#frequency data
#lines must be in rows
#alleles in columns
fit<-cmdscale(d,eig=T,k=2) #Object with two dimensions
plot(fit$points[,1],fit$points[,2],col="red",xlab = "X1",ylab = "X2") #Plot the two variables
```




```
#Which are those isolated lines?
x<-as.data.frame(fit$points)
a<-x[x[,1]<-4 & x[,2]>4,]
row.names(a)
```

```
## [1] "SEEDDIV2899" "SEEDDIV2893" "SEEDDIV2846" "SEEDDIV2904" "SEEDDIV2848"
## [6] "SEEDDIV2898" "SEEDDIV2834" "SEEDDIV2945" "SEEDDIV2969"
```

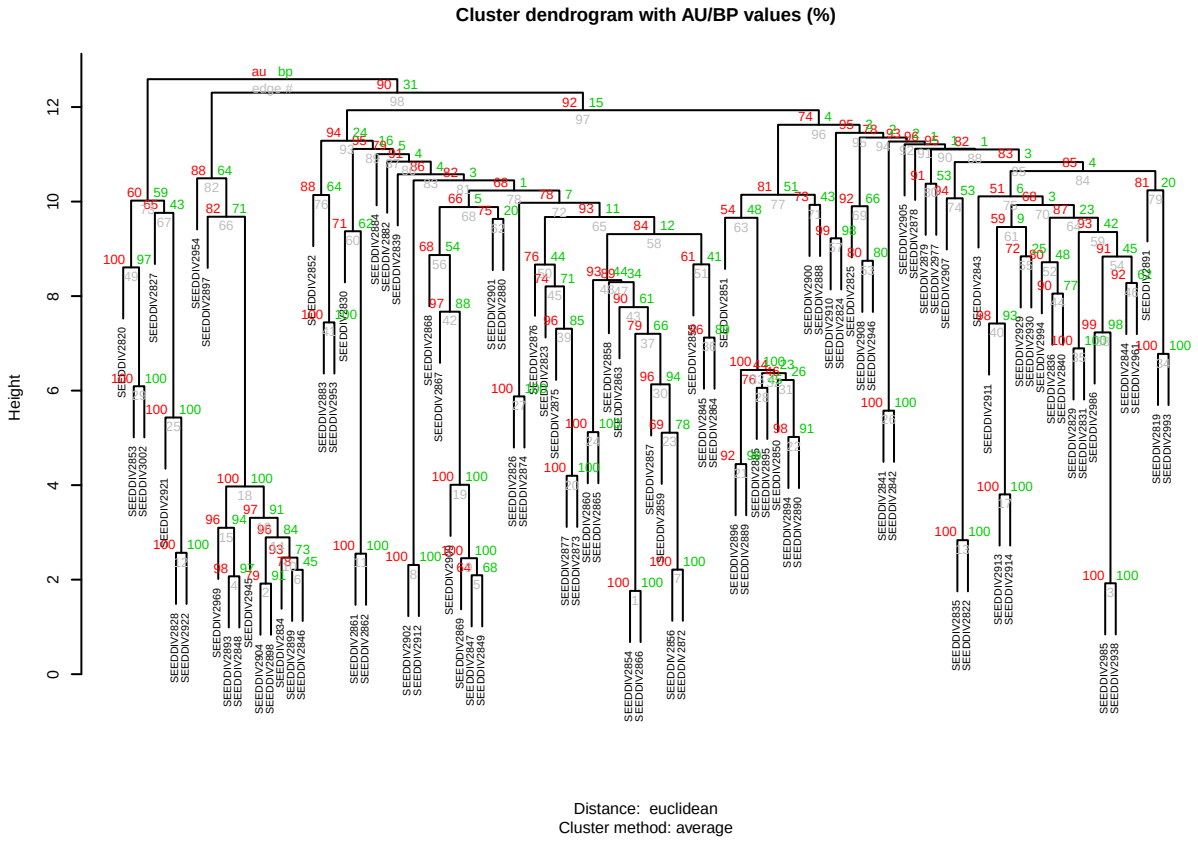
Hierarchical clustering

The hierarchical clustering approach goes beyond Principal Component Analysis and Principal Coordinate Analysis, and builds a hierarchy of clusters. In this section, we will build a hierarchical representation of the 100 wheat lines, and try to relate the findings with those obtained through PCA and PCoA.

```
#Hierarchical analysis
library(pvclust)
clust<-pvclust(sample[-c(1,2)],method.dist="euclidean",nboot=100)
```

```
## Bootstrap (r = 0.5)... Done.
## Bootstrap (r = 0.6)... Done.
## Bootstrap (r = 0.7)... Done.
## Bootstrap (r = 0.8)... Done.
## Bootstrap (r = 0.9)... Done.
## Bootstrap (r = 1.0)... Done.
## Bootstrap (r = 1.1)... Done.
## Bootstrap (r = 1.2)... Done.
## Bootstrap (r = 1.3)... Done.
## Bootstrap (r = 1.4)... Done.
```

```
par(cex=.5)
plot(clust,cex=.6)
```



Interestingly, there is a small, outstanding group, whose node is scored with 100%, for both *AUB* and *BP*. The lines in that group are the same as those detected as a separate group by PCA and PCoA. However, hierarchical analysis provides a much better resolution of the structure of the genetic diversity in the set of wheat lines. Particularly, the most isolated group appears a the left side of the plot.

$$\overbrace{\left(\hat{\mathrm{O}}_{\triangledown}\hat{\mathrm{O}}\right)}^{\frown}$$
$$\left(\left[\begin{array}{c}\mathrm{vvv}\\ \mathrm{vvv}\end{array}\right]\right)$$
$$\lambda\frown\lambda$$

Humberto Reyes