Multi-Environment Trial (MET) Analysis

Guilherme Oliveira, Mandeep Singh, Subash Thapa

2024-11-05

Multi-Environment Trial Analysis - PS 756 project

The following document was developed by Guilherme Oliveira, Mandeep Singh, Subash Thapa, students at South Dakota State University, as a requisite for the course PS 746 - Quantitative Genetics.

The main objective of this project was to conduct Multi Trial Analysis, covering i) GxE analysis (+ GGE analysis) ii) AMMI analysis, iii) GGI analysis, iv) FW analysis and v) genomic selection approach which explores in some way Genotype x Environment interaction (G x E).

For the parts i, ii, iii, iv, we are exploring the open dataset made available by Dias et al. (2018) which contains phenotypic data of five drought tolerance traits, measured in in 308 hybrids along eight environments contrasting for water availability. For practical purposes, we are using a subset of 202 hybrids. The traits analyzed were grain yield (GY), ears per plot (EPP), female and male flowering times (FFT and MFT), and anthesis-silking interval (ASI). The source code used is from the metan (Olivoto and Lucio, 2020) and statgenGxE reference manuals.

For the part v, we are exploring the open dataset made available by Crossa et al. 2013 and Montesinos-Lopez et al. 2016, 2017 which contains the data of 309 double-haploid maize lines conducted in 3 environments and each environments having 3 reps of each line. The traits analyzed were grain yield (yield), anthesis-silking interval (ASI) and plant height (PH). The source code used is from Montesinos-Lopez et al. 2019 using the BMTME R package, where we did just small modifications, however, bringing more interpretations and extra analyses.

Directory

```
setwd("C:/Users/Guilherme.Oliveira/Documents/PS 756 - Final Project")
getwd()
```

```
## [1] "C:/Users/Guilherme.Oliveira/Documents/PS 756 - Final Project"
```

Libraries

The main libraries used for this project to run the Multi-Environment Trial Analysis.

```
library(statgenGxE)
library(metan)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
## |=======================================================|
```

```
## | Multi-Environment Trial Analysis (metan) v1.18.0      |
```

```
## | Author: Tiago Olivoto                                 |
```

```
## | Type 'citation('metan')' to know how to cite metan    |
```

```
## | Type 'vignette('metan_start')' for a short tutorial   |
```

```
## | Visit 'https://bit.ly/pkgmetan' for a complete tutorial |
```

```
## |=======================================================|
```

```
library(tibble)
```

```
##
## Attaching package: 'tibble'
```

```
## The following objects are masked from 'package:metan':
##
##     column_to_rownames, remove_rownames, rownames_to_column
```

```
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.2     v tidyr     1.3.1
## v purrr     1.0.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x forcats::as_factor()         masks metan::as_factor()
## x tibble::column_to_rownames() masks metan::column_to_rownames()
## x dplyr::filter()              masks stats::filter()
## x dplyr::lag()                 masks stats::lag()
## x dplyr::recode_factor()       masks metan::recode_factor()
## x tibble::remove_rownames()    masks metan::remove_rownames()
## x tidyr::replace_na()          masks metan::replace_na()
## x tibble::rownames_to_column() masks metan::rownames_to_column()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(BMTME)
library(lme4)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

Data

The "pheno" is the dataset which will be used for the parts i, ii, iii, iv while the "G.Maize" and "Data.Maize" will be used for the part v.

```
pheno <- read.table("pheno_corn_MET.txt", header = TRUE)
load("C:/Users/Guilherme.Oliveira/Documents/PS 756 - Final Project/G.Maize.RData")
load("C:/Users/Guilherme.Oliveira/Documents/PS 756 - Final Project/Data.Maize.RDATa")
```

Data preparation and exploration

Before running the parts i, ii, iii, iv, we need to check some very important points. First, we need to check our "pheno" structure, normality, distribution, descriptive statistics etc.

To complete these tasks, we are using some R basic functions + some functions inside the metan package.

Data strcture

In this analysis the ENV, GEN, REP should be factors, while the traits should be numeric. Also we need, to change our phenotypic data frame to a tibble, being able to run the metan functions.

```
str(pheno)
```

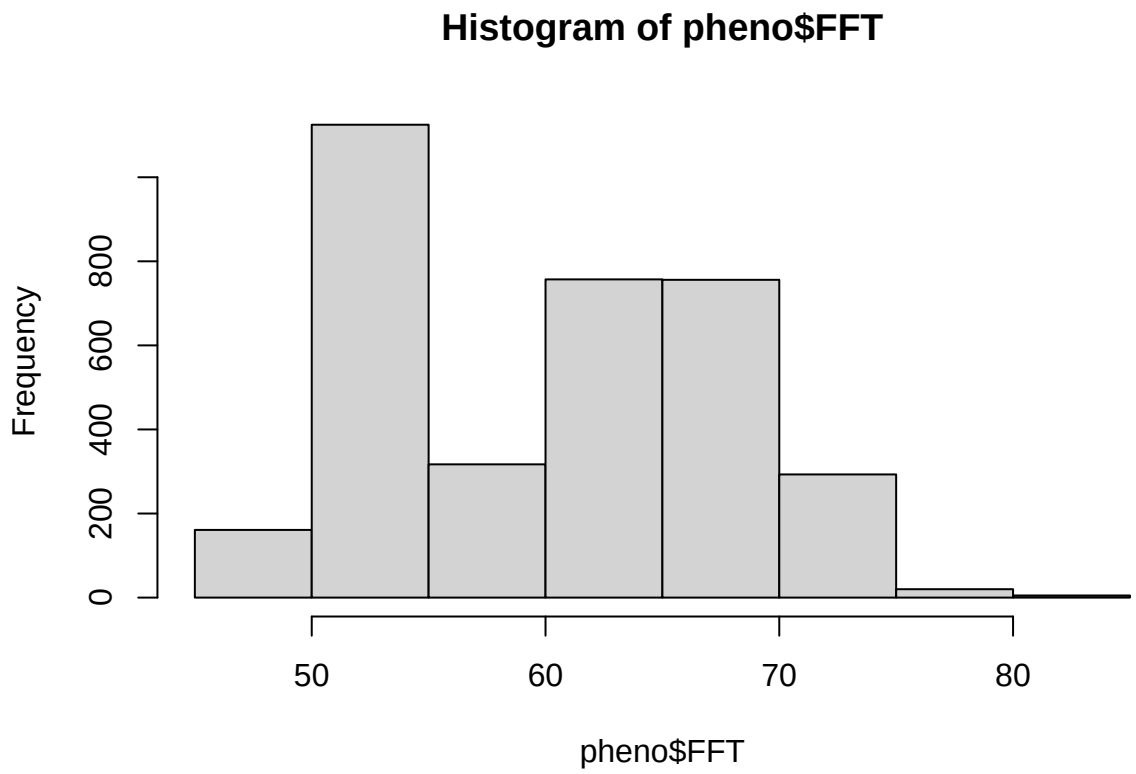
```
## 'data.frame':    3434 obs. of  8 variables:
##  $ ENV: chr  "E1" "E2" "E3" "E4" ...
##  $ REP: int  1 1 1 1 1 1 1 2 2 ...
##  $ GEN: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ FFT: int  66 62 69 70 63 55 54 56 65 63 ...
##  $ MFT: int  62 61 66 69 50 52 51 53 63 62 ...
##  $ ASI: int  4 1 3 1 13 3 3 3 2 1 ...
##  $ EPP: int  12 18 13 15 7 15 13 7 12 16 ...
##  $ GY : num  1.91 6.63 3.63 5.6 0.97 4.64 4.5 2.55 3.6 5.53 ...
```

```
pheno$ENV <- as.factor(pheno$ENV)
pheno$GEN <- as.factor(pheno$GEN)
pheno$REP <- as.factor(pheno$REP)
pheno$FFT <- as.numeric(pheno$FFT)
pheno$MFT <- as.numeric(pheno$MFT)
pheno$ASI <- as.numeric(pheno$ASI)
pheno$EPP <- as.numeric(pheno$EPP)
pheno <- as_tibble(pheno)

str(pheno)
```

```
## tibble [3,434 x 8] (S3: tbl_df/tbl/data.frame)
##  $ ENV: Factor w/ 8 levels "E1","E2","E3",..: 1 2 3 4 5 6 7 8 1 2 ...
##  $ REP: Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 2 2 ...
##  $ GEN: Factor w/ 202 levels "1","3","4","8",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ FFT: num [1:3434] 66 62 69 70 63 55 54 56 65 63 ...
##  $ MFT: num [1:3434] 62 61 66 69 50 52 51 53 63 62 ...
##  $ ASI: num [1:3434] 4 1 3 1 13 3 3 3 2 1 ...
##  $ EPP: num [1:3434] 12 18 13 15 7 15 13 7 12 16 ...
##  $ GY : num [1:3434] 1.91 6.63 3.63 5.6 0.97 4.64 4.5 2.55 3.6 5.53 ...
```

Data normality

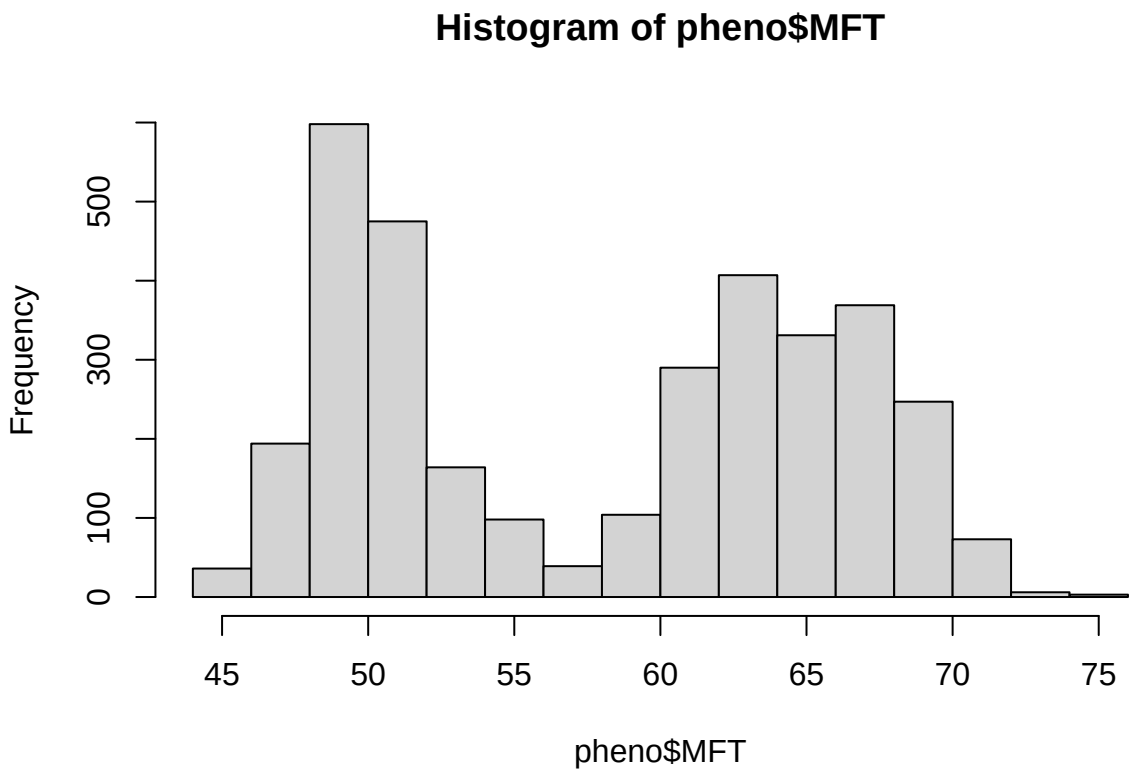
```
hist(pheno$FFT)
```




```
shapiro.test(pheno$FFT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  pheno$FFT
## W = 0.93568, p-value < 2.2e-16
```

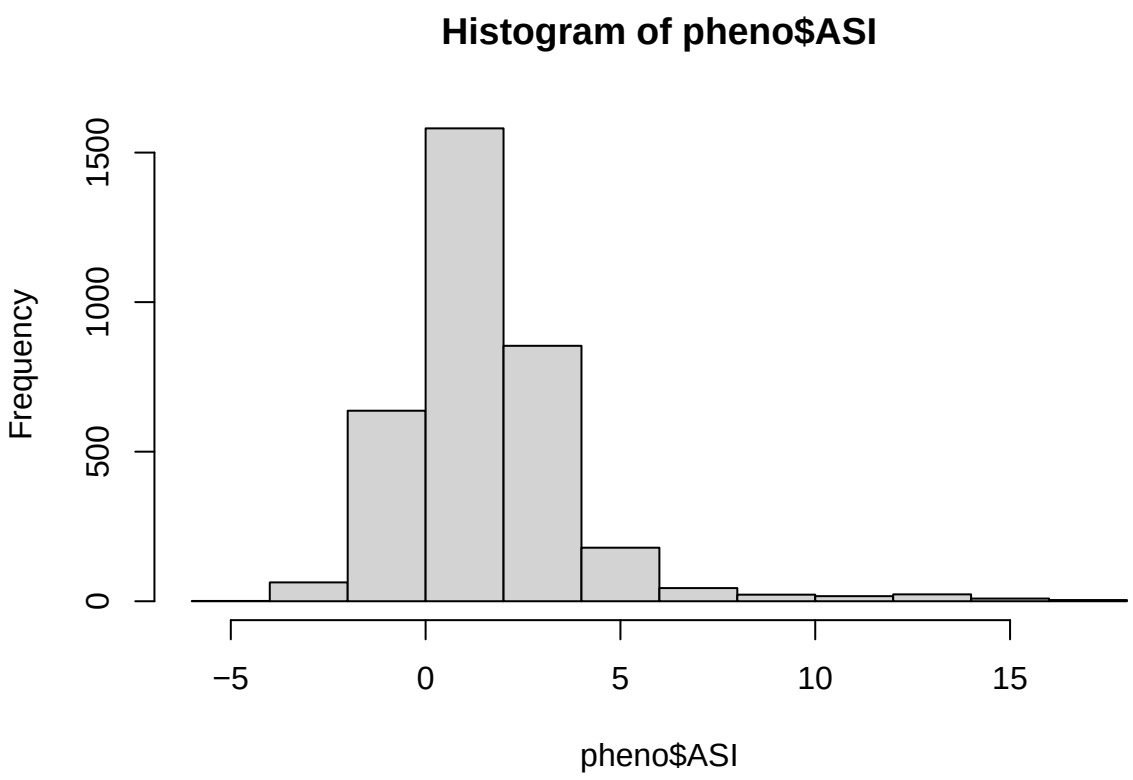
```r
hist(pheno$MFT)
```




```r
shapiro.test(pheno$MFT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  pheno$MFT
## W = 0.89781, p-value < 2.2e-16
```

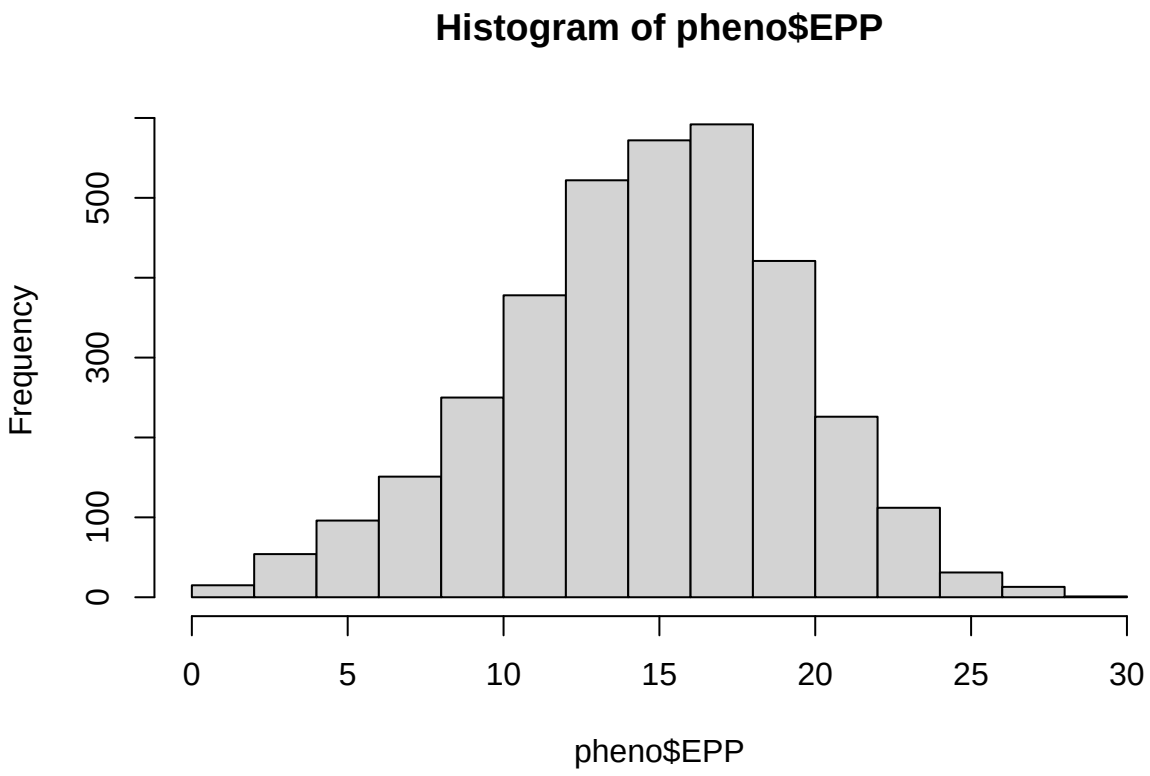
```r
hist(pheno$ASI)
```




```
shapiro.test(pheno$ASI)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  pheno$ASI
## W = 0.82689, p-value < 2.2e-16
```

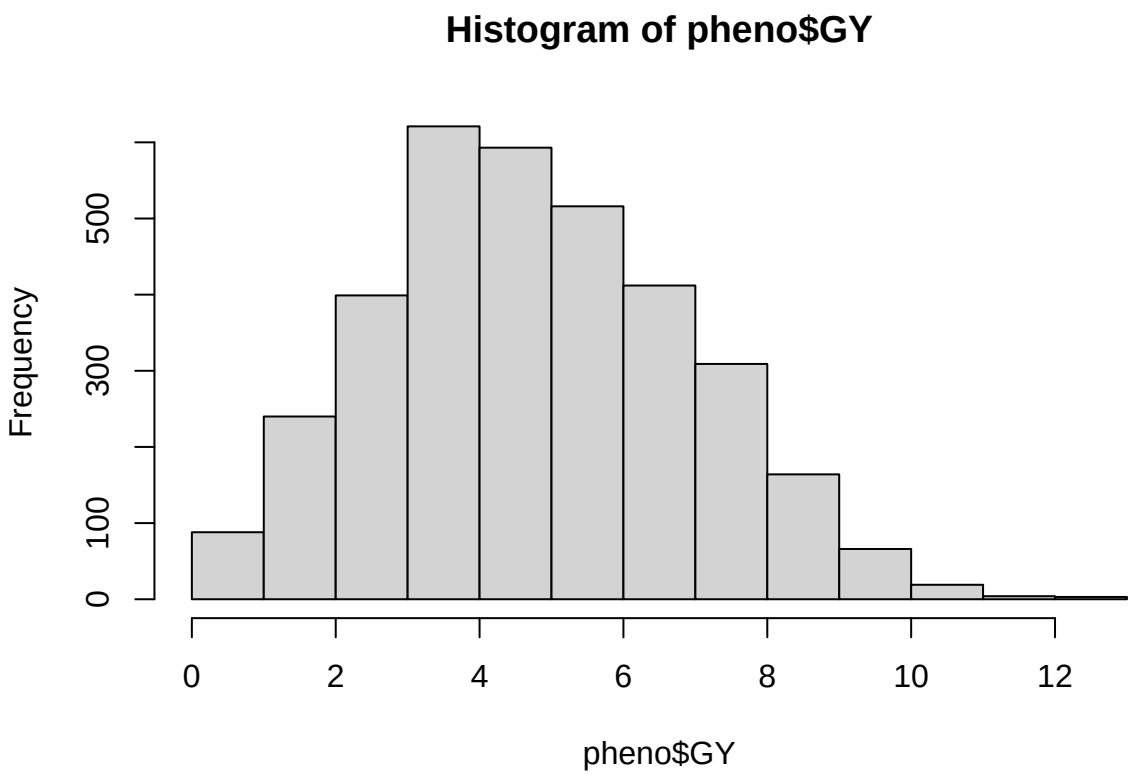
```
hist(pheno$EPP)
```




```
shapiro.test(pheno$EPP)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  pheno$EPP
## W = 0.9893, p-value = 2.005e-15
```

```
hist(pheno$GY)
```




```
shapiro.test(pheno$GY)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  pheno$GY
## W = 0.99075, p-value = 3.805e-14
```

Based on the histograms and Shapiro-Wilks, all the traits does not have a normal distribution. There are different ways to get distributions more close to normality, as example, removing outliers or transforming the data. In our case we tested both approaches, however, we are still unable to get higher values than 0.05 in our Shapiro-Wilk analyses.

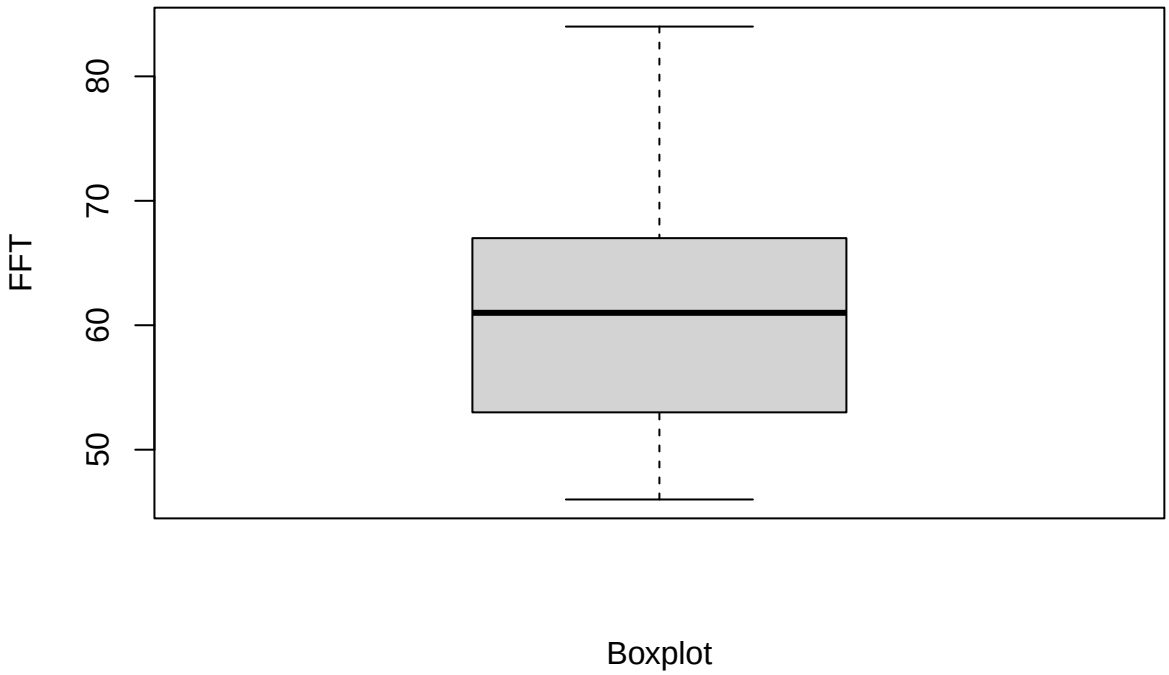
The main reason for this is that because we have in our dataset environments with high drought stress affecting the plants, as well as, good environments. In the moment that we pull together the data, we obtain some extremes values (caused by the drought stress), resulting in some normality deviations.

In this sense, we decide to proceed the analysis with our raw values, despite then being completely normal, once, that the main objective of this project is to understand and interpret the MET analysis and develop and/or analyze some previous developed code. Nevertheless, we strongly suggest to pay attention in the data normality and understand how (or not) it can affect the analysis results.

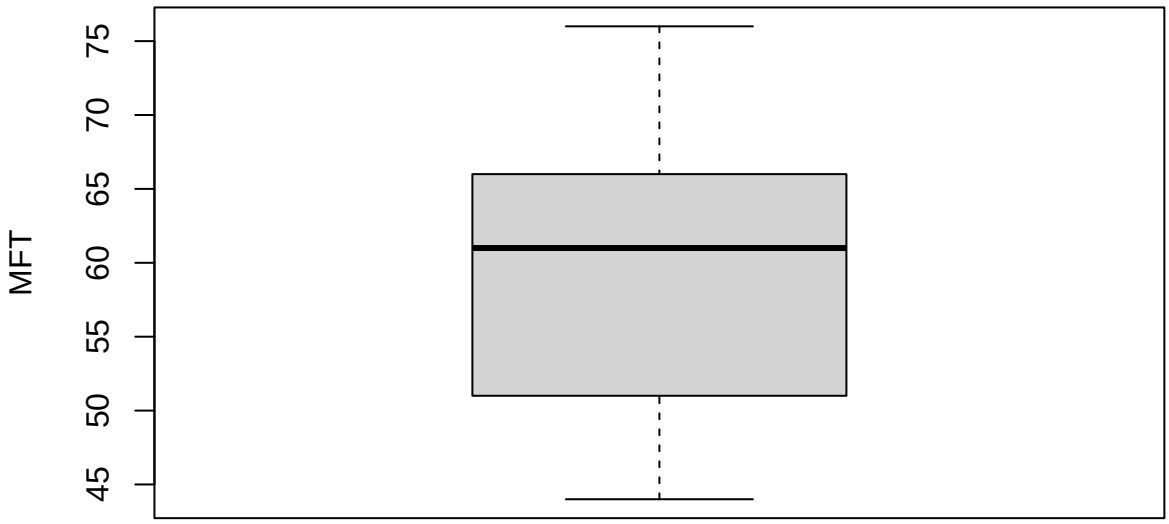
However, in the next codes chunks, we are still providing our code to remove outliers based on the interquartile range (IQR) equations to remove outliers (Q1 - 1.5 x IQR and Q3 + 1.5 x IQR).

Code for removing outilers

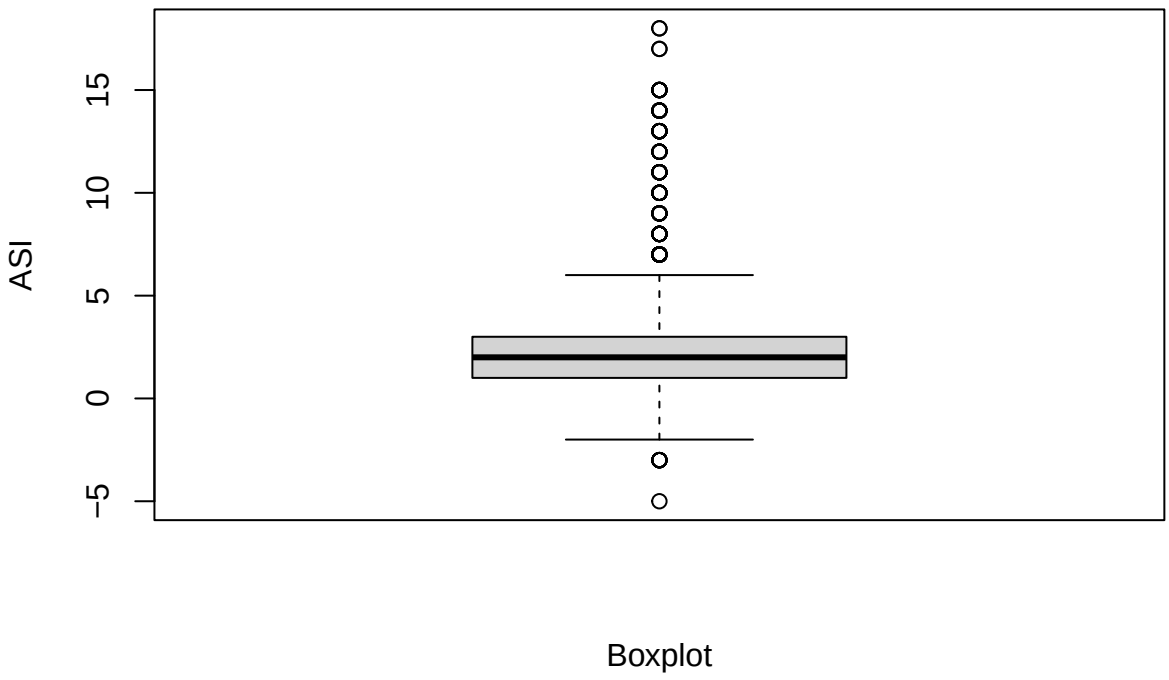
```
# Code to get and see the boxplots

boxplot.FFT <- boxplot(pheno$FFT, xlab="Boxplot",ylab="FFT")
```



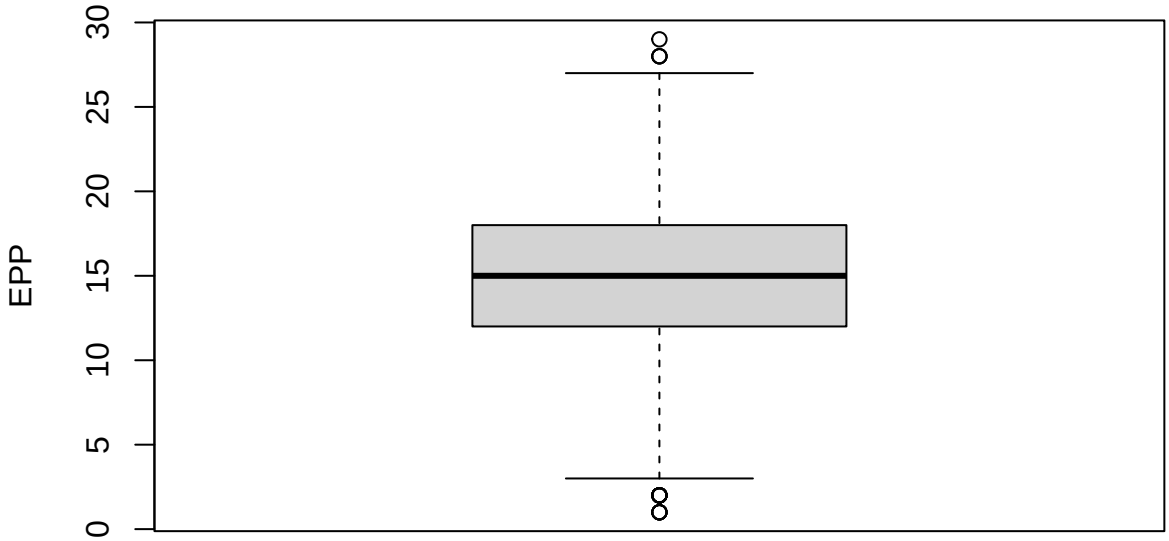

```
boxplot.MFT <- boxplot(pheno$MFT, xlab="Boxplot",ylab="MFT")
```




```
boxplot.ASI <- boxplot(pheno$ASI, xlab="Boxplot",ylab="ASI")
```

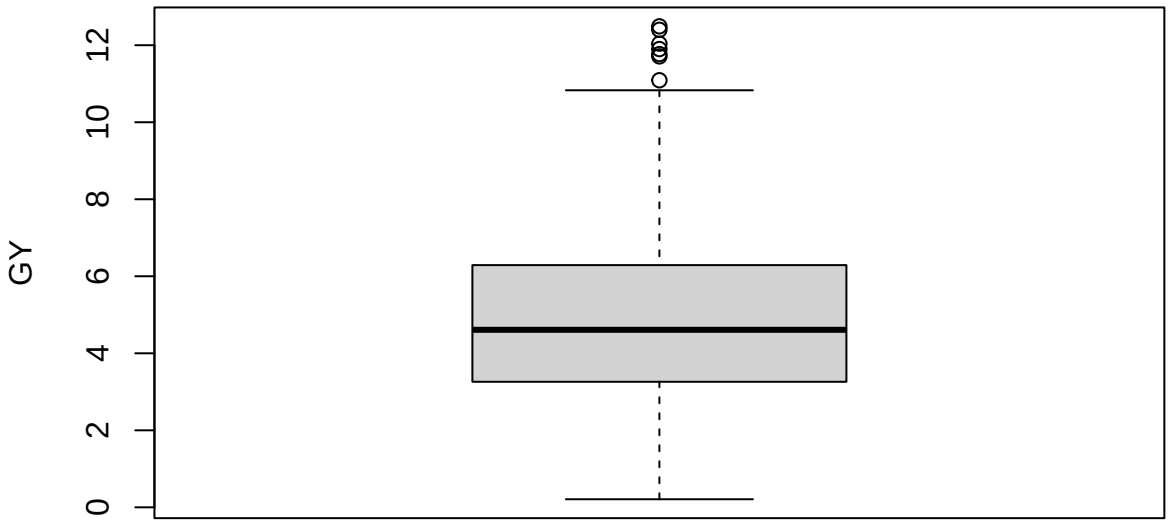



```
boxplot.EPP <- boxplot(pheno$EPP, xlab="Boxplot",ylab="EPP")
```



Boxplot

```
boxplot.GY <- boxplot(pheno$GY, xlab="Boxplot",ylab="GY")
```



Boxplot

```
# Example to remove outliers for a specific trait

#outliers <- boxplot.GY$out; outliers
#pheno <- pheno[-which(pheno$GY%in%outliers),]
#shapiro.test(pheno$GY)
```

Information of the genotypes

The metan package has a bunch of different functions which provide valuable information to check, manipulate, and summarize data. In the next code chunks we are going to show just a small portion of them, before starting the main analyzes required by this project. However, if someone is interested in learning more about these preliminary steps, here is the link for the paper: *https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/2041-210x.13384* that give the access to supplementary information and R codes.

First, the metan package has functions that permits to check means and coefficient of variation (CV) for each genotype across the environments. It can be useful, as a first way to understand about data quality, finding the best and worst ones, as well, based on the CV, having some idea about genotype by environment interaction.

Also, using the metan functions, we can collect the descriptive analysis by trait.

```
desc_stat(pheno, stats = "all")
```

```
## # A tibble: 5 x 34
##   variable av.dev   ci.t   ci.z    cv gmean hmean   iqr    kurt   mad   max
```

```
##   <chr>     <dbl>  <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl> <dbl> <dbl>
## 1 ASI        1.48 0.0769 0.0769 107.   1.97  0     2     9.06    1.48  18
## 2 EPP        3.72 0.156  0.156   31.0 14.1  12.7   6    -0.0772  4.45  29
## 3 FFT        6.70 0.251  0.251   12.4 60.2  59.7  14    -1.24   10.4   84
## 4 GY         1.74 0.0713 0.0713  44.5  4.21  3.39  3.03 -0.336   2.21  12.5
## 5 MFT        7.26 0.261  0.261   13.3 58.0  57.4  14.8  -1.54   11.9   76
## # i 23 more variables: mean <dbl>, median <dbl>, min <dbl>, n <dbl>,
## #   n.valid <dbl>, n.missing <dbl>, n.unique <dbl>, ps <dbl>, q2.5 <dbl>,
## #   q25 <dbl>, q75 <dbl>, q97.5 <dbl>, range <dbl>, sd.amo <dbl>, sd.pop <dbl>,
## #   se <dbl>, skew <dbl>, sum <dbl>, sum.dev <dbl>, ave.dev <dbl>,
## #   sum.sq.dev <dbl>, var.amo <dbl>, var.pop <dbl>
```

In this analysis, we can observe some very extreme values for CV, as example the ASI CV. In this case, the CV is higher than 100%, meaning that our standard deviation is larger than our mean, in other words, there is a high variability for this trait. Also, we got a high CV for GY, close to 40%.

Again, the main reason to obtain these results is the nature of the environments where the data was collected, having some of them, extreme drought effects. Traits as ASI and GY are extremely affected by drought, so high CVs to these traits, in our analysis conditions are expected.

If necessary, the same function provides information about single locations, where we can confirm our hypothesis that some environments (drought ones) are pushing the CVs to high values. This code lines can be useful for removing traits or environments with high coefficient of variation.

```
desc_stat(pheno,
          stats = ("mean, se, cv, max, min"),
          by = ENV)
```

```
## # A tibble: 40 x 7
##    ENV   variable   mean     se     cv   max   min
##    <fct> <chr>     <dbl>  <dbl>  <dbl> <dbl> <dbl>
##  1 E1    ASI       0.932 0.0701 185.    8    -5
##  2 E1    EPP      12.3   0.152   30.4  23     2
##  3 E1    FFT      64.0   0.105    4.03 73    59
##  4 E1    GY        3.53  0.053   37.0   8.36  0.24
##  5 E1    MFT      63.1   0.0785   3.06 70    58
##  6 E2    ASI       0.404 0.0581 289.    4    -3
##  7 E2    EPP      18.1   0.172   19.1  28     5
##  8 E2    FFT      64.6   0.155    4.81 73    58
##  9 E2    GY        6.70  0.078   23.4  10.7   1.31
## 10 E2    MFT      64.2   0.154    4.81 72    57
## # i 30 more rows
```

Something very interesting that also the metan package provides is a function that shows the best lines (higher phenotypic values) in each of these environments. We can see that there is no replicate for GY, meaning that for each environment, there was a different best hybrid.

```
ge_winners(pheno, ENV, GEN, resp = everything())
```

```
## # A tibble: 8 x 6
##   ENV   FFT   MFT   ASI   EPP   GY
##   <fct> <chr> <chr> <chr> <chr> <chr>
## 1 E1    146   74    146   44    287
```

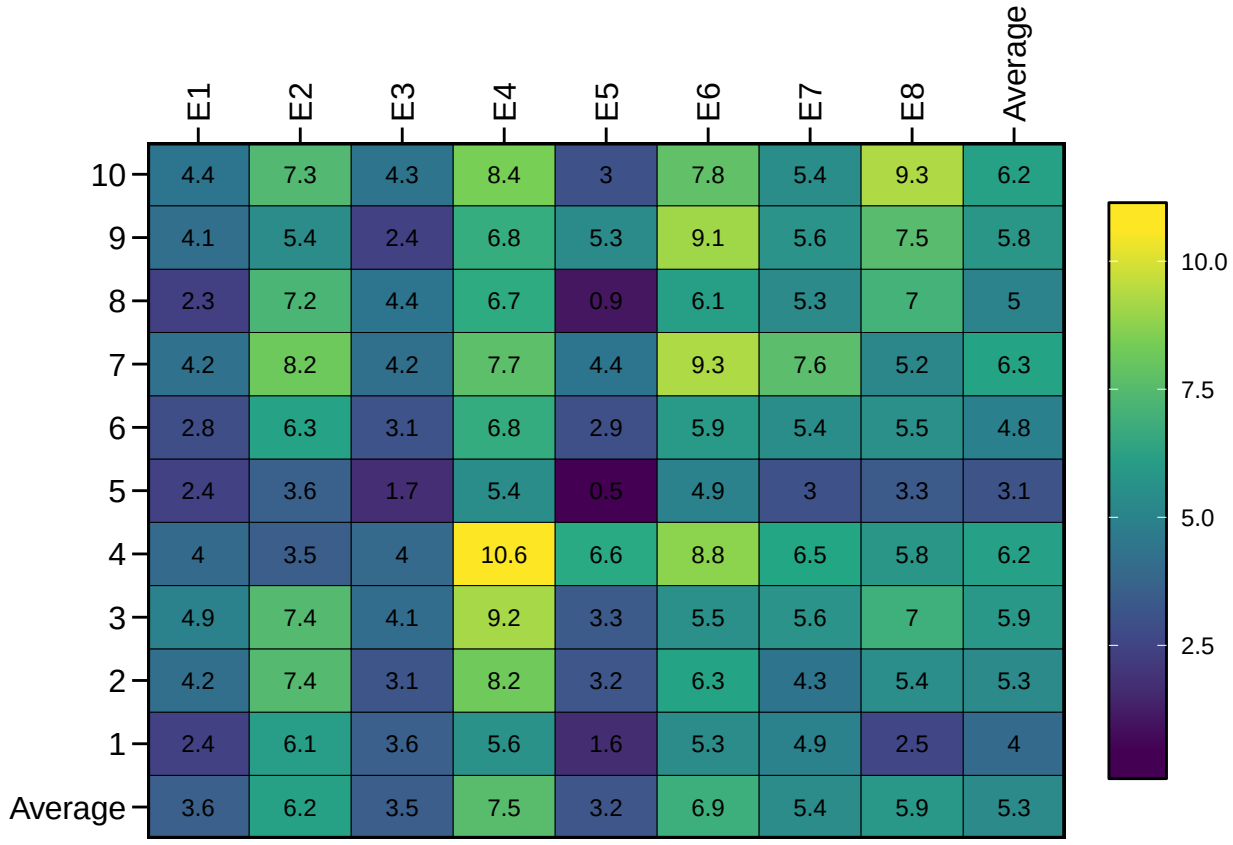
```
## 2 E2    146   236   268   53    101
## 3 E3    22    253   9     191   190
## 4 E4    253   253   224   216   156
## 5 E5    270   16    270   122   234
## 6 E6    73    253   307   289   222
## 7 E7    22    176   276   83    83
## 8 E8    158   148   158   261   69
```

In addition, sometimes numbers in tables are hard to visualize and/or to take other conclusions, which are more easy by plots.

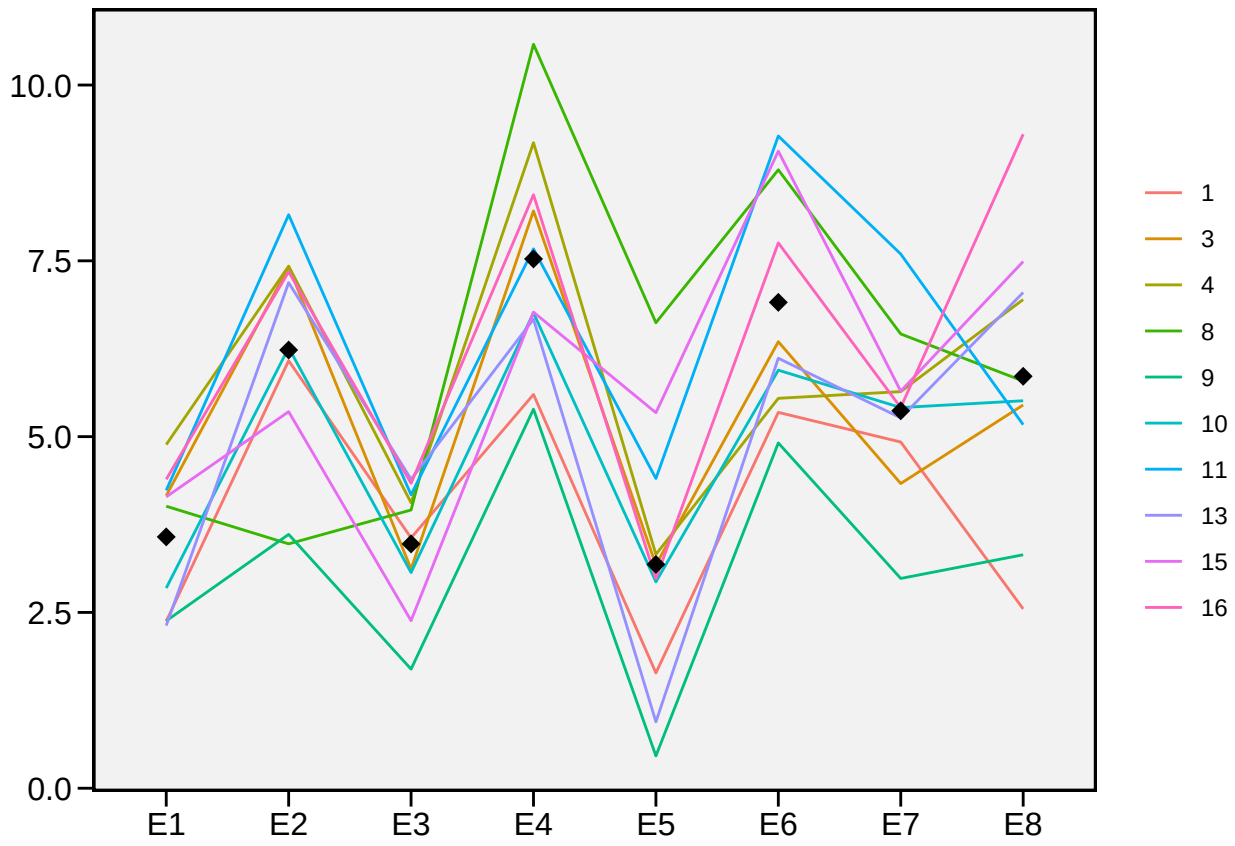
So in the next code chunk, we are showing how to do some heatmaps and line plots to collect more applied information for specific lines across the environments.

So, the first subset, shows how to collect a range of genotypes and the second subset shows how to select very specific lines.

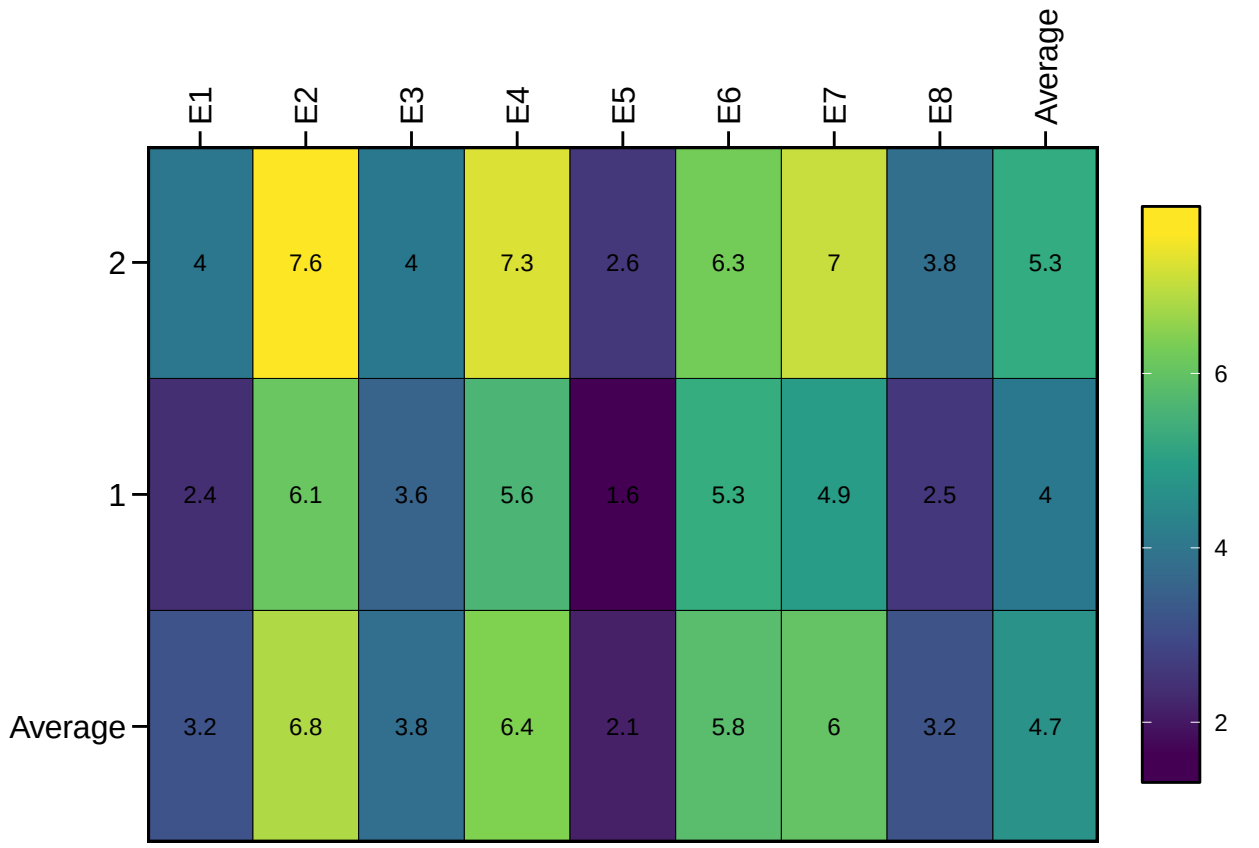
```
subset_pheno <- pheno %>%
  filter(GEN %in% levels(GEN)[1:10]) %>%
  droplevels()
a <- ge_plot(subset_pheno, ENV, GEN, GY); a
```



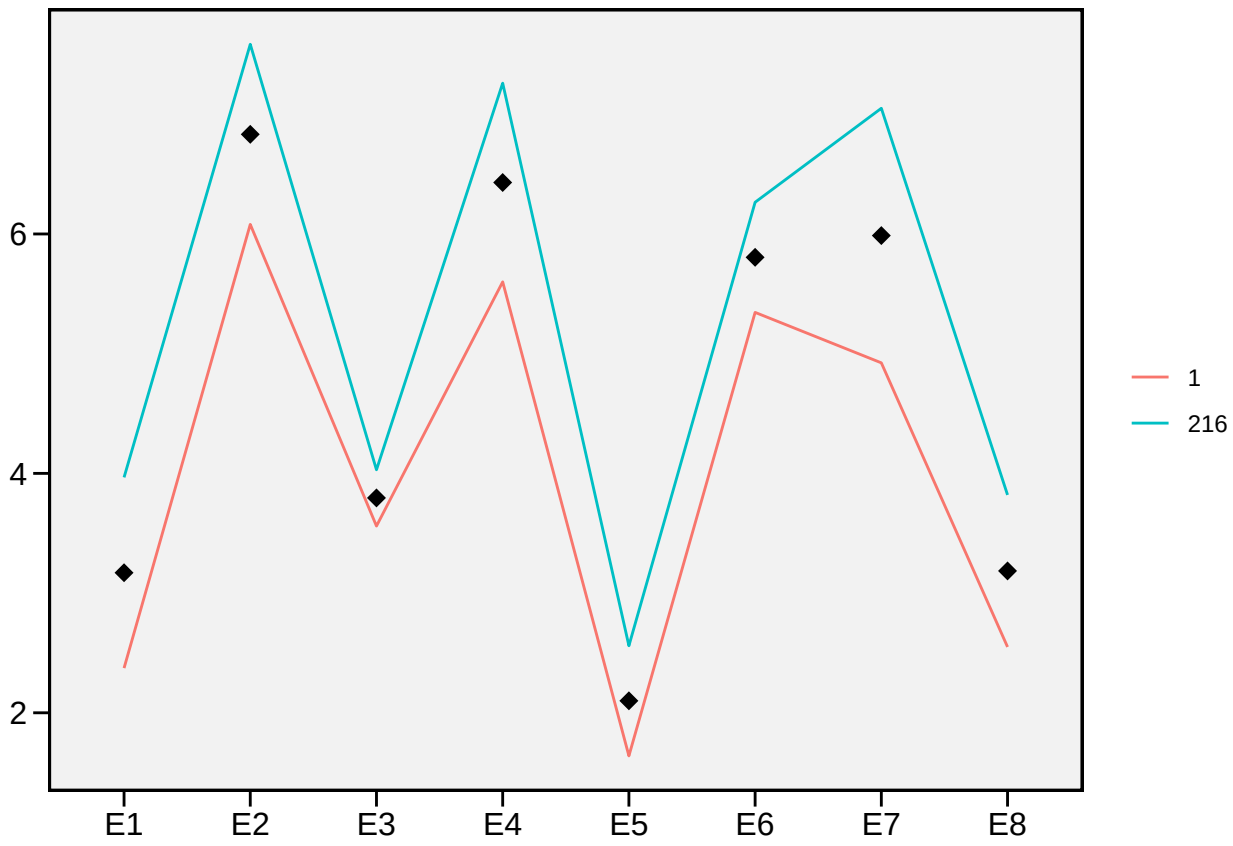

```
b <- ge_plot(subset_pheno, ENV, GEN, GY, type = 2); b
```




```
subset_pheno2 <- pheno %>% filter(GEN %in% c(levels(GEN)[1], levels(GEN)[137]))
c <- ge_plot(subset_pheno2, ENV, GEN, GY); c
```




```
d <- ge_plot(subset_pheno2, ENV, GEN, GY, type = 2); d
```



So, now, we have a good idea bout our data, we are going to do the main analyses required for this project.

Part i) GxE analysis (+GGE analysis)

In this first step, we are doing some analysis to understand more the genotype x environment interaction. In addition, we also running some GGE analysis, aiming to understand better the group of environments (mega-environments).

For the first analysis, we are using the statgenGxE package. Thus, we need to convert the data according to the package requirements. In addition, we are running the some analysis (not all) for the subset_pheno, aiming to understand how conclusions can (or not) differ when we have more data.

```
phenoTD <- statgenSTA::createTD(data = pheno, genotype = "GEN",
                                trial = "ENV")

phenoTD2 <- statgenSTA::createTD(data = subset_pheno, genotype = "GEN",
                                trial = "ENV")
```

Using a function from the statgenGxE, we are running a mixed model, where we are able to extract the variance components.

First, we are only checking yield.

```
phenoVarComp <- gxeVarComp(TD = phenoTD, trait = "GY")
summary(phenoVarComp)
```

```
## Fitted model formula final mixed model
##
##  GY ~ trial + (1 | genotype) + (1 | genotype:trial)
##
## Sources of variation for fully random model:
##  GY ~ (1 | trial) + (1 | genotype) + (1 | genotype:trial)
##
##                Component % Variance expl.
## trial               2.66          52.75 %
## genotype            0.39           7.75 %
## genotype:trial      0.57          11.32 %
## residuals           1.42          28.19 %
##
## Analysis of Variance Table for fully fixed model:
##  GY ~ trial + genotype + genotype:trial
##
##                   Df Sum Sq Mean Sq  F value    Pr(>F)
## trial              7 7579.9 1082.84 824.9253 < 2.2e-16 ***
## genotype         201 1835.0    9.13   6.9550 < 2.2e-16 ***
## genotype:trial  1407 3778.9    2.69   2.0461 < 2.2e-16 ***
## residuals       1818 2386.4    1.31
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

More than 50% of the variation observed was due the trial. It was expected, because, as explained, these trials are conducted in different conditions of water availability.

We also can use some loops to test for all the traits.

It is important that we are using a lot this vector with the traits ("GY", "FFT", "MFT", "ASI", "EPP") in our loops, so we are not calling it all the time, but, pay attention that the loops using the 'traits' are calling back this vector.

Once we have the variance components, we also can estimate the heritability for the traits. Let's check for all the traits.

```
herit(phenoVarCompGY)
```

```
## [1] 0.7091811
```

```
herit(phenoVarCompFFT)
```

```
## [1] 0.7428897
```

```
herit(phenoVarCompMFT)
```

```
## [1] 0.7875332
```

```
herit(phenoVarCompASI)
```

```
## [1] 0.6839426
```

```r
herit(phenoVarCompEPP)
```

```
## [1] 0.7213632
```

It is a little surprising to get a high heritability (0.71) for grain yield (quantitative trait), specifically for the trials conditions from this dataset. Something very interesting which this package permits us is to predict the genotype main effect based on the mixed model analysis.

```r
for (trait in traits) {
  var_name <- paste0("phenoVarComp", trait)
  phenoVarComp_obj <- get(var_name)
  predGeno <- predict(phenoVarComp_obj)
  print(head(predGeno))
  predGenoTrial <- predict(phenoVarComp_obj, predictLevel = "trial")
  print(head(predGenoTrial))
}
```

```
##   genotype predictedValue
## 1        1       4.075380
## 2        3       4.829526
## 3        4       5.336766
## 4        8       5.579351
## 5        9       3.116764
## 6       10       4.544380
##   genotype trial predictedValue
## 1        1    E1       2.594241
## 2        3    E1       3.906923
## 3        4    E1       4.489299
## 4        8    E1       4.113286
## 5        9    E1       2.259634
## 6       10    E1       3.066884
##   genotype predictedValue
## 1        1       61.35769
## 2        3       60.58507
## 3        4       58.94312
## 4        8       59.88803
## 5        9       63.51131
## 6       10       61.57250
##   genotype trial predictedValue
## 1        1    E1       65.66005
## 2        3    E1       63.03142
## 3        4    E1       62.24192
## 4        8    E1       64.09488
## 5        9    E1       66.49333
## 6       10    E1       65.51408
##   genotype predictedValue
## 1        1       58.03661
## 2        3       58.46670
## 3        4       57.51110
## 4        8       57.62900
## 5        9       59.06293
## 6       10       60.05819
```

```
##   genotype trial predictedValue
## 1        1    E1       62.51132
## 2        3    E1       62.02938
## 3        4    E1       62.51744
## 4        8    E1       61.90762
## 5        9    E1       63.40701
## 6       10    E1       64.60323
##   genotype predictedValue
## 1        1       3.247772
## 2        3       2.119967
## 3        4       1.447720
## 4        8       2.231176
## 5        9       4.341619
## 6       10       1.593752
##   genotype trial predictedValue
## 1        1    E1      2.9651082
## 2        3    E1      0.9634526
## 3        4    E1     -0.1982906
## 4        8    E1      2.0253837
## 5        9    E1      2.9951103
## 6       10    E1      0.9091673
##   genotype predictedValue
## 1        1       12.85855
## 2        3       13.47370
## 3        4       16.14316
## 4        8       18.05381
## 5        9       12.35989
## 6       10       14.87659
##   genotype trial predictedValue
## 1        1    E1       10.66674
## 2        3    E1       10.06561
## 3        4    E1       14.24057
## 4        8    E1       15.60256
## 5        9    E1       10.93304
## 6       10    E1       12.21042
```

In this example, we can see that for example based on the mixed model and evaluation across all the locations for GY, the genotype 8 is the best, and the genotype 9 is the worst, based on the predictions.

Extracting the variance components, getting the heritability, being able to do predictions are main analysis in the GxE spectrum. However, we would like to go more deep, and also finding something is very useful for some research areas, as example plant breeding, mega environments. To complete this task, we are going back to the metan package.

Initially, we need to fit a gge model. Here, we are testing with the subset_pheno.

```
gge_model <- gge(pheno, ENV, GEN, GY)
gge_model2 <- gge(subset_pheno, ENV, GEN, GY)
```

As the statgenGxE, we also can do predictions, however, because, the models work a little different, the results will be similar, but not the same results.

```
predict(gge_model)
```

```
## $GY
## # A tibble: 202 x 8
##       E1    E2    E3    E4    E5    E6    E7    E8
##   * <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  2.94  5.79  2.56  5.82  2.23  5.51  4.52  2.62
##  2  3.70  7.04  3.57  7.53  3.07  7.03  5.42  5.30
##  3  3.99  7.48  3.96  8.04  3.40  7.57  5.74  6.74
##  4  4.26  8.05  4.22  9.20  3.65  8.21  6.15  5.73
##  5  2.43  4.83  2.01  4.12  1.74  4.43  3.83  3.00
##  6  3.37  6.42  3.20  6.44  2.75  6.32  4.98  5.43
##  7  4.26  8.06  4.21  9.25  3.64  8.21  6.16  5.51
##  8  3.46  6.50  3.38  6.33  2.89  6.45  5.04  6.91
##  9  3.82  7.13  3.83  7.29  3.27  7.19  5.49  7.64
## 10  4.15  7.64  4.28  7.91  3.65  7.83  5.87  9.18
## # i 192 more rows
```

```
predict(gge_model2)
```

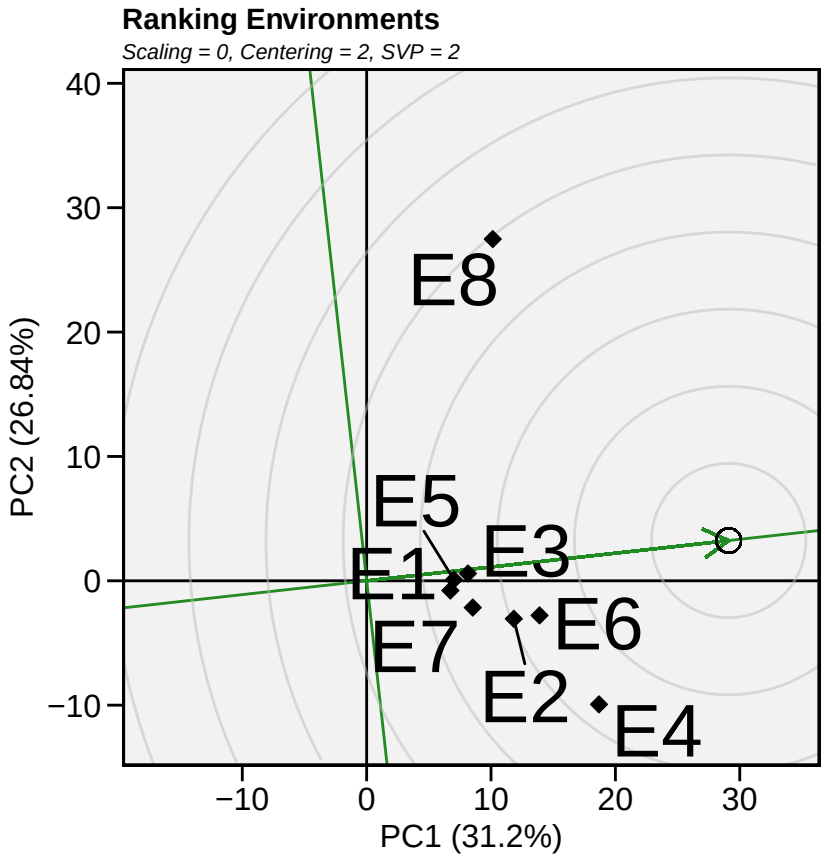
```
## $GY
## # A tibble: 10 x 8
##       E1    E2    E3    E4    E5    E6    E7    E8
##   * <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  2.49  5.28  2.71  5.87 1.34   5.26  4.14  3.72
##  2  3.55  6.63  3.58  7.40 2.79   6.68  5.30  6.08
##  3  3.99  7.52  4.03  7.95 3.11   7.11  5.74  7.25
##  4  4.37  4.09  3.21  9.41 6.94   9.46  6.57  5.66
##  5  2.00  4.36  2.23  5.23 0.938  4.75  3.63  2.45
##  6  3.27  6.14  3.31  7.01 2.51   6.36  5.00  5.36
##  7  4.23  6.60  3.88  8.58 4.47   8.01  6.14  7.03
##  8  3.35  7.81  3.83  6.78 1.31   5.74  4.93  6.52
##  9  4.11  5.61  3.53  8.60 5.01   8.24  6.09  6.23
## 10  4.37  8.28  4.42  8.44 3.41   7.50  6.14  8.27
```

The metan package has a variety of biplots that explore the GGE (Genotype plus GxE), but, for practical purposes, we are only selecting the biplot that ranks the environments, bringing, at some degree, the relationship between environments.

```
biplot_ranking <- plot(gge_model,
type = 6,
col.gen = "black",
col.env = "black",
size.text.env = 10,
axis_expand = 1.5)
biplot_ranking
```

```
## Warning in geom_segment(x = 0, y = 0, xend = xcoord, yend = ycoord, arrow = arrow(length = unit(0.15
## i Please consider using ‘annotate()‘ or provide this layer with data containing
##   a single row.
```

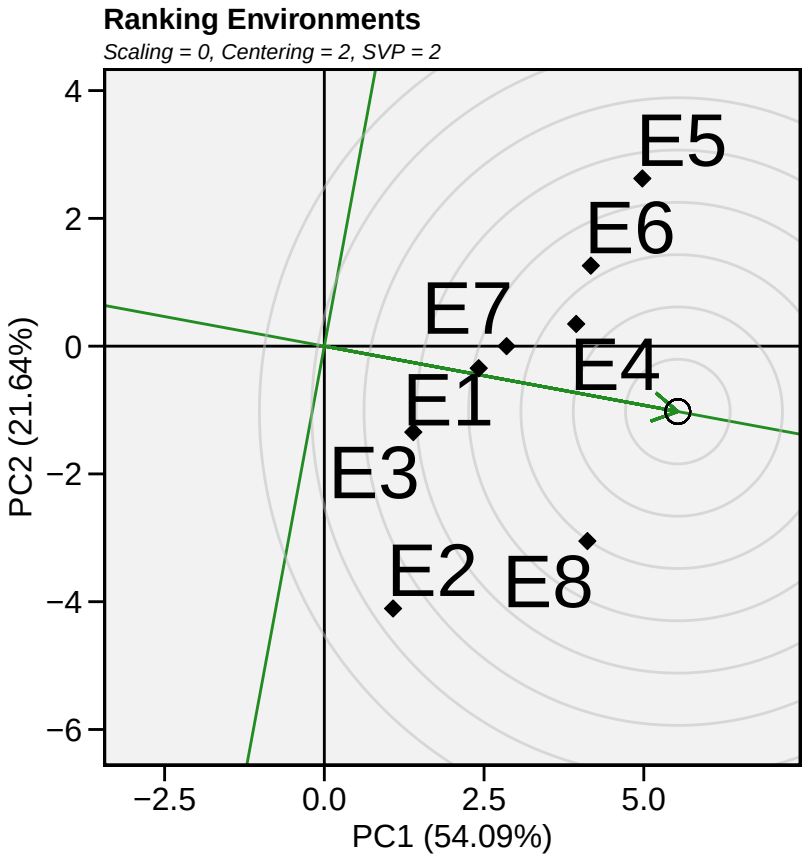
```
## Warning in geom_point(aes(xcoord, ycoord), shape = 1, size = 4, color = col.stroke): All aesthetics h
## i Please consider using ‘annotate()‘ or provide this layer with data containing
##   a single row.
```




```
biplot_ranking <- plot(gge_model2,
type = 6,
col.gen = "black",
col.env = "black",
size.text.env = 10,
axis_expand = 1.5)
biplot_ranking
```

```
## Warning in geom_segment(x = 0, y = 0, xend = xcoord, yend = ycoord, arrow = arrow(length = unit(0.15
## i Please consider using ‘annotate()‘ or provide this layer with data containing
##   a single row.
```

```
## Warning in geom_point(aes(xcoord, ycoord), shape = 1, size = 4, color = col.stroke): All aesthetics l
## i Please consider using ‘annotate()‘ or provide this layer with data containing
##   a single row.
```



Basically, in the first biplots, what is close to the black line are considered good environments, while the environments far from the line, differ from the best environments. For example the E5 looks a very good environment, according to this line, while the E8 is not so good (drought stress). It also give to us some idea, about which environments can be grouped as on unique environment, and which ones, does not belong to the "good" pool.

However, these conclusions changes drastically when we only evaluated a small subset of hybrids, being a good example of how the sample size affects some important conclusions in MET analysis.

Below is the code to rank the environments only based on GY, what makes more sense, for example, in the breeding research area. However, we also are providing some code (loop) to test for all the traits.

Part iii) AMMI analysis

The AMMI stands for the additive main effect and multiplicative interaction. In this model, the additive main effects are the genotypes and the trials, coming from the ANOVA results, where the multiplicative interaction factors are coming from a PCA analysis on the interactions residuals (genotype by environment means after adjustment for additive genotype and environment effects).

Therefore, in the code chunk below we are running this model only for GY trait, aiming to evaluate the genotypes performance across the environments, and finding the stable ones.

Also, from here, we decide, to run the analysis with a subset of genotypes, to show how the graphs look when we have a small or big number of genotypes and how it can affect our conclusions.

```
AMMI_model <- performs_ammi(pheno, ENV, GEN, REP, GY)
```

```
## variable GY
## ------------------------------------------------------------------------
## AMMI analysis table
## ------------------------------------------------------------------------
##     Source   Df Sum Sq Mean Sq F value    Pr(>F) Proportion Accumulated
##        ENV    7   7580 1082.84   90.57  1.28e-07         NA          NA
##   REP(ENV)    9    108   11.96    9.49  3.08e-14         NA          NA
##        GEN  201   1835    9.13    7.25 4.44e-128         NA          NA
##    GEN:ENV 1407   3779    2.69    2.13  5.99e-52         NA          NA
##        PC1  207   2635   12.73   10.11  0.00e+00       37.9        37.9
##        PC2  205   1303    6.36    5.04  0.00e+00       18.7        56.6
##        PC3  203   1047    5.16    4.09  0.00e+00       15.0        71.6
##        PC4  201    829    4.13    3.28  0.00e+00       11.9        83.6
##        PC5  199    561    2.82    2.24  0.00e+00        8.1        91.6
##        PC6  197    360    1.83    1.45  1.00e-04        5.2        96.8
##        PC7  195    224    1.15    0.91  8.01e-01        3.2       100.0
##  Residuals 1809   2279    1.26      NA        NA         NA          NA
##      Total 4840  22540    4.66      NA        NA         NA          NA
## ------------------------------------------------------------------------
##
## All variables with significant (p < 0.05) genotype-vs-environment interaction
## Done!
```

```
AMMI_model_2 <- performs_ammi(subset_pheno, ENV, GEN, REP, GY)
```

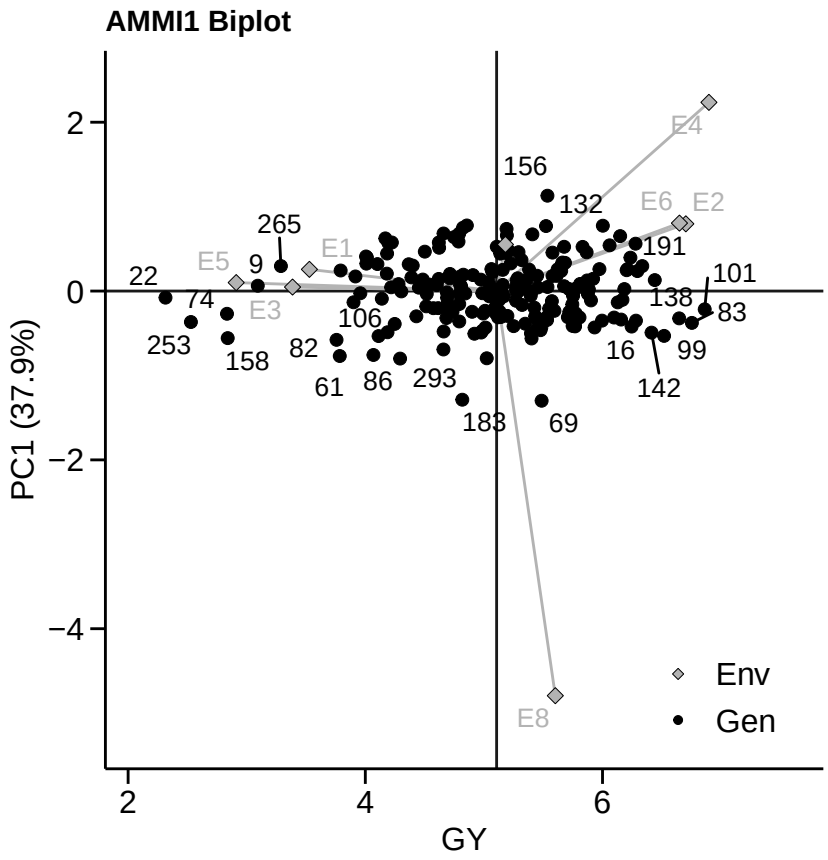
```
## variable GY
## ------------------------------------------------------------------------
## AMMI analysis table
## ------------------------------------------------------------------------
##     Source  Df  Sum Sq Mean Sq F value   Pr(>F) Proportion Accumulated
##        ENV   7  373.69  53.385  69.460 4.10e-07         NA          NA
##   REP(ENV)   9    6.92   0.769   0.599 7.94e-01         NA          NA
##        GEN   9  150.26  16.696  13.009 1.38e-12         NA          NA
##    GEN:ENV  63  159.39   2.530   1.971 2.05e-03         NA          NA
##        PC1  15  125.53   8.369   6.520 0.00e+00       47.5        47.5
##        PC2  13   64.78   4.983   3.880 1.00e-04       24.5        72.0
##        PC3  11   42.23   3.839   2.990 2.20e-03       16.0        87.9
##        PC4   9   18.15   2.016   1.570 1.38e-01        6.9        94.8
##        PC5   7    7.33   1.047   0.820 5.74e-01        2.8        97.6
##        PC6   5    4.66   0.931   0.730 6.03e-01        1.8        99.3
##        PC7   3    1.81   0.603   0.470 7.04e-01        0.7       100.0
##  Residuals  81  103.95   1.283      NA       NA         NA          NA
##      Total 232 1058.70   4.563      NA       NA         NA          NA
## ------------------------------------------------------------------------
##
## All variables with significant (p < 0.05) genotype-vs-environment interaction
## Done!
```

The values clearly showed a strong genotype x environment interaction in both scenarios.

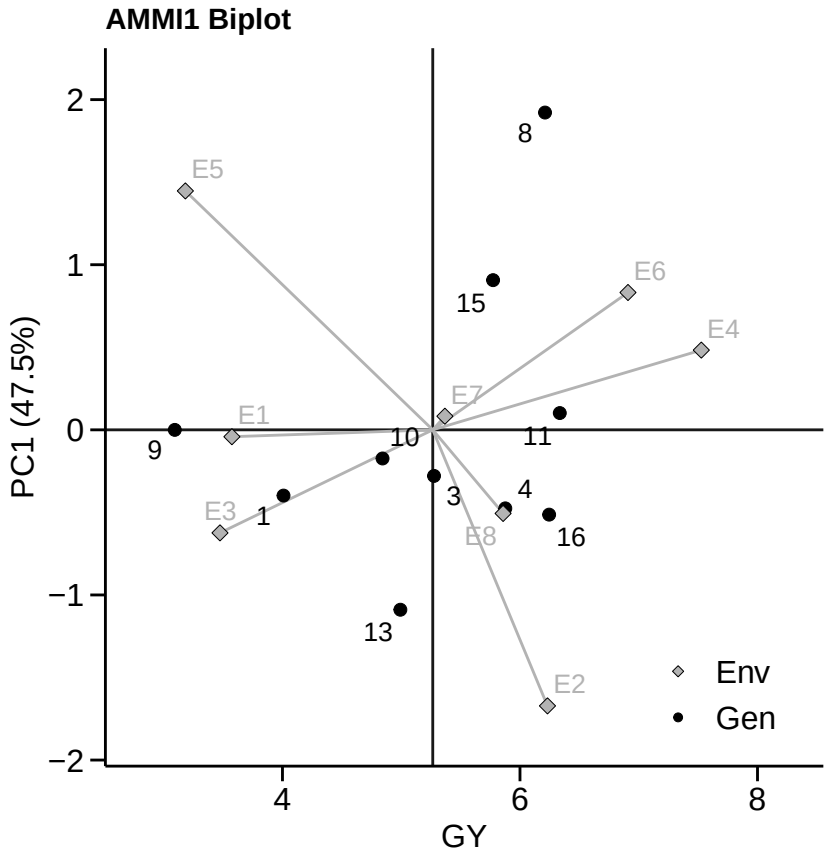
There are different ways to explore the results obtained by the AMMI model. One good way is developing a biplot, where we are using as axis our trait and the first PC.

```
AMMI_biplot_GY <- plot_scores(AMMI_model,
col.gen = "black",
col.env = "gray70",
col.segm.env = "gray70",
plot_theme = theme_metan_minimal())
AMMI_biplot_GY
```

```
## Warning: ggrepel: 181 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
AMMI_biplot_GY2 <- plot_scores(AMMI_model_2,
col.gen = "black",
col.env = "gray70",
col.segm.env = "gray70",
plot_theme = theme_metan_minimal())
AMMI_biplot_GY2
```



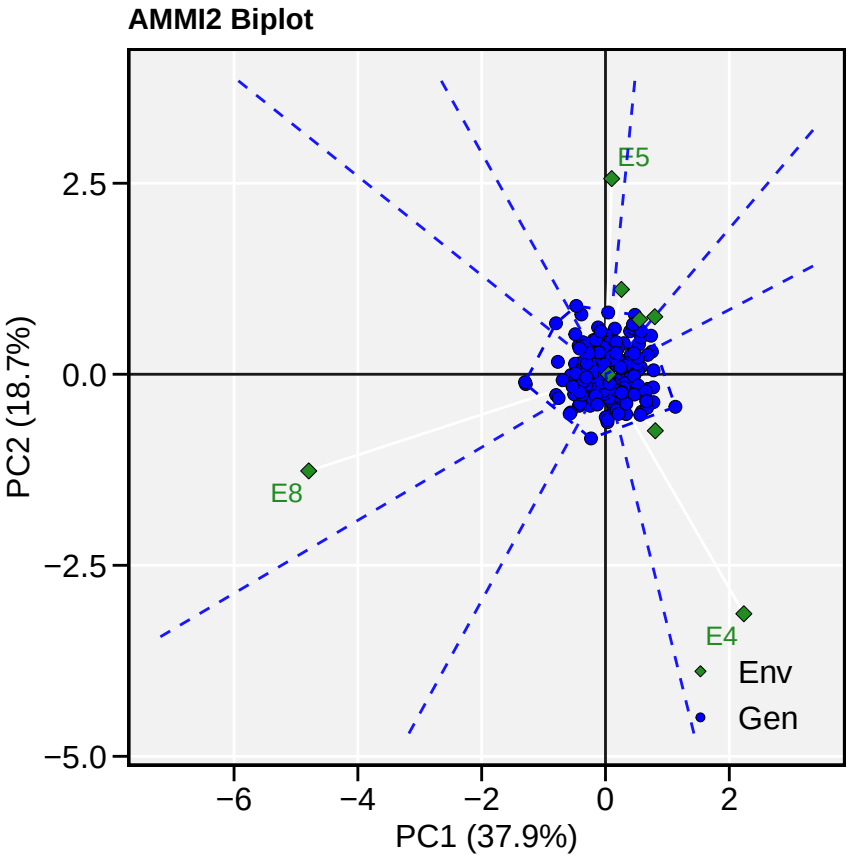
In this biplot we can take a look, from each environments are more similar, or thinking in the other side, finding the environment that have the same best lines. Also we need to pay attention in the quadrants, as well as, in the directions of our environments, because it represents the relationship between environments. For example, E6 and E2 are very similar to each other, however, they are completely different from the E5, E1, E3. The hybrid 101 and 138 apparently have high yield on the environment 6 and 2 and it is low yield in environments 5,3, and 1.

The conclusions are very similar for our small dataset, however we have some deviation about the environments correlation.

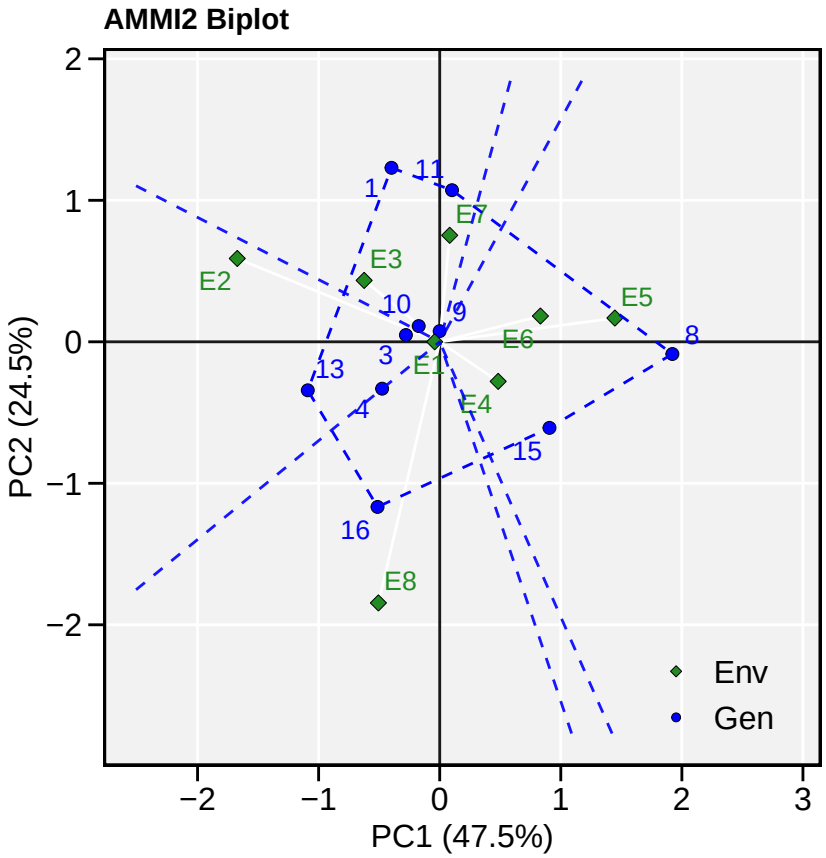
We also can check if we obtain similar results comparing PC1 and PC2.

```
AMMI_biplot_PCs <- plot_scores(AMMI_model,
type = 2,
polygon = T,
col.segm.env = "#FFFFFF00", # Transparent
axis.expand = 1.5,
plot_theme = theme_metan(grid = "both"))
AMMI_biplot_PCs
```

```
## Warning: ggrepel: 207 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
AMMI_biplot_PCs_2 <- plot_scores(AMMI_model_2,
type = 2,
polygon = T,
col.segm.env = "#FFFFFF00", # Transparent
axis.expand = 1.5,
plot_theme = theme_metan(grid = "both"))
AMMI_biplot_PCs_2
```

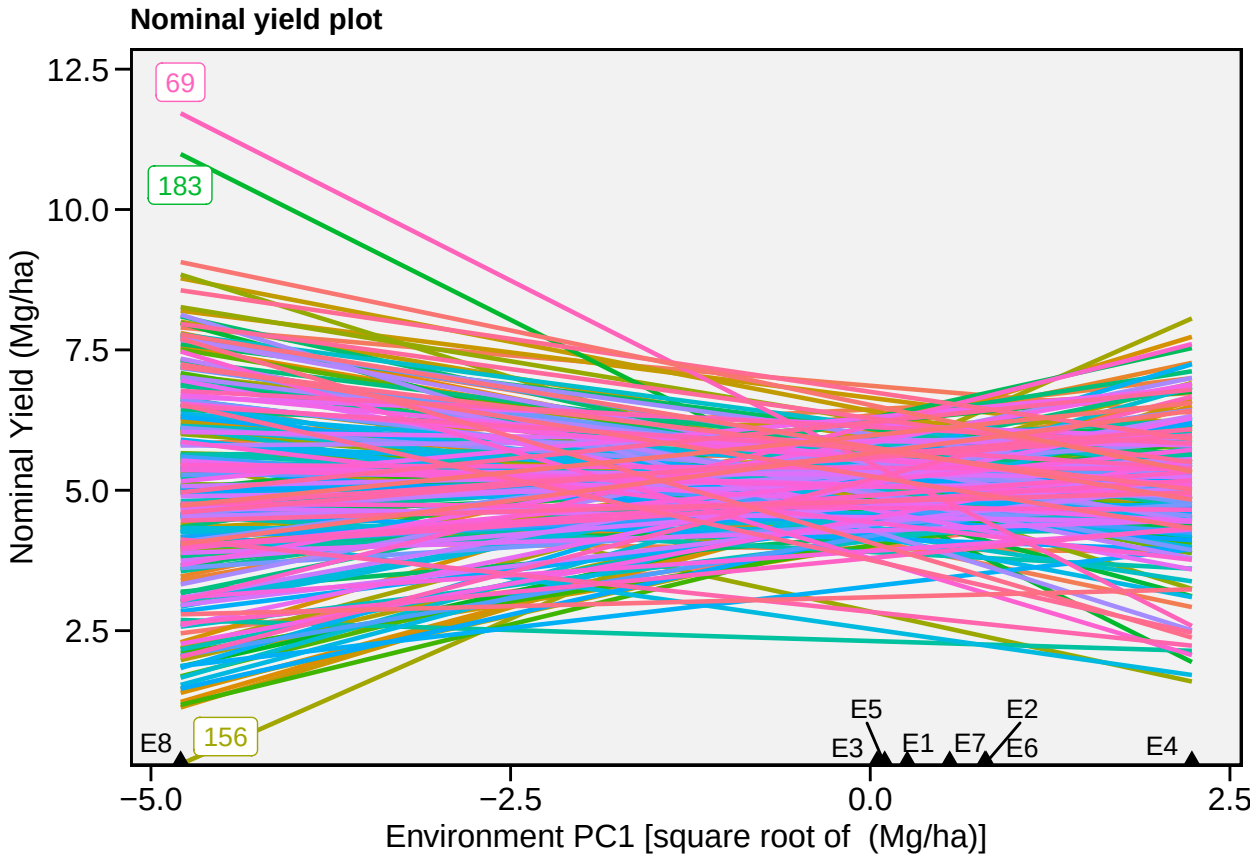


The results are pretty the same.

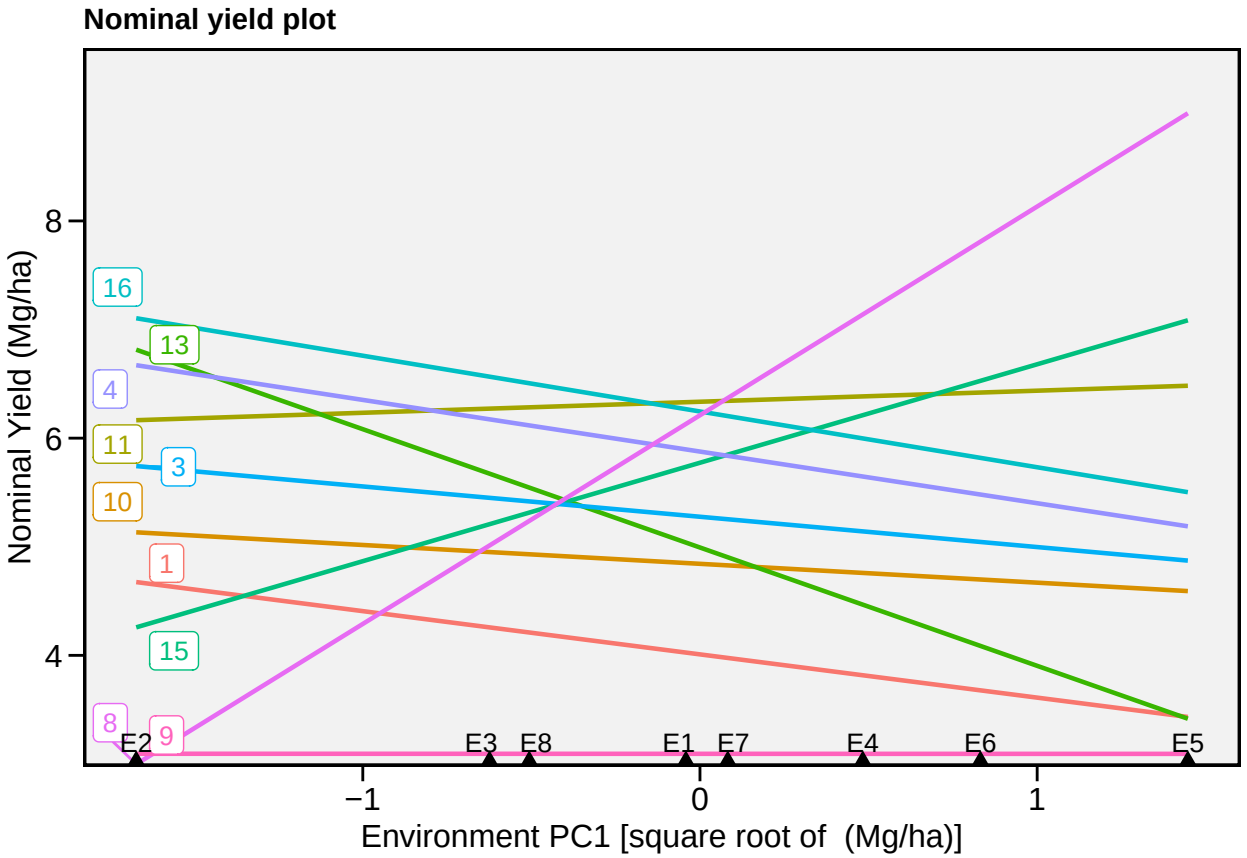
Also we can check "which-won-where" based on our AMMI model.

```
AMMI_biplot_nominal <- plot_scores(AMMI_model, type = 4)
AMMI_biplot_nominal
```

```
## Warning: ggrepel: 199 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
AMMI_biplot_nominal_2 <- plot_scores(AMMI_model_2, type = 4)
AMMI_biplot_nominal_2
```



In these plots, we can observe the change in rankings, as example, in the first plot where the line 69 is a good one in the E8, however, its rankings drops drastically across better environments.

Again, we only conducted for GY, because it the most important trait in plant breeding, and makes more sense to take conclusions of genotype stability based on this trait.

However, if someone would like to check for all the traits, the code below has a loop that conducts the graphs for all the traits.

Part iii) GGI analysis

The GGI stands for genotype by group interaction. The main differences from the previous analysis is that now, we are grouping some hybrids according to their similarities, considering all the traits, and estimating group-by-environment interactions.

So, the package uses a function to cluster genotypes based on their means for all the traits. Again, here, we are using the pheno and subset_pheno.

```
mean_gen <-
pheno %>%
means_by(GEN) %>%
column_to_rownames("GEN")
```

```
## Warning: `mean_by()` is deprecated as of metan 1.17.0
## Please use `mean_by()` instead
```

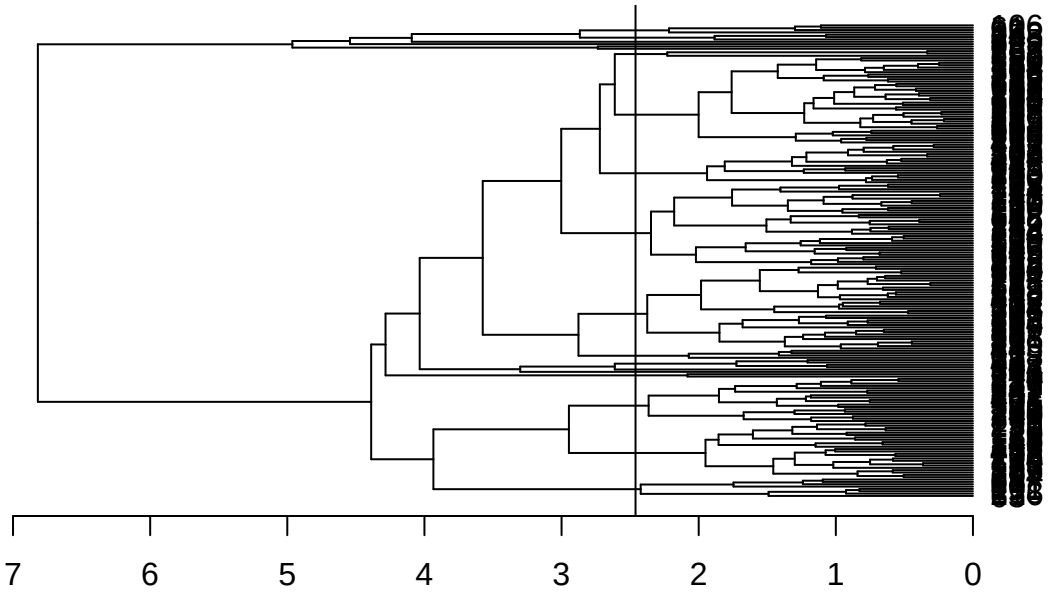
```
d2 <- clustering(mean_gen)

mean_gen_2 <-
subset_pheno %>%
means_by(GEN) %>%
column_to_rownames("GEN")
```

```
## Warning: ‘mean_by()‘ is deprecated as of metan 1.17.0
## Please use ‘mean_by()‘ instead
```

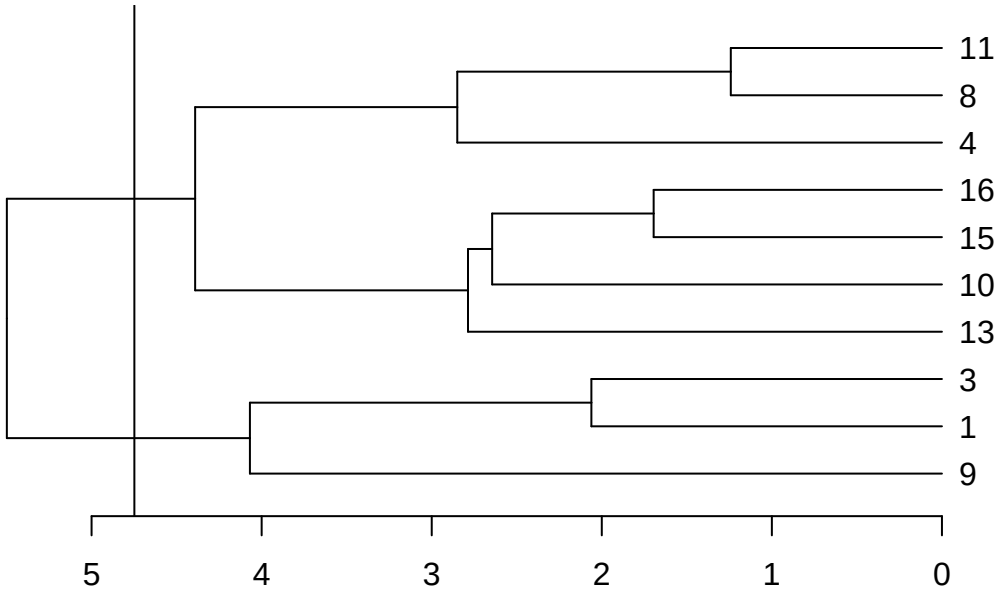
```
d2_2 <- clustering(mean_gen_2)
```

After clustering the genotypes, we can plot this cluster in a dendrogram shape, also, to identify how many cluster we got in this analysis.

```
plot(d2)
```




```
plot(d2_2)
```



For the biggest dataset, we have a lot of cluster, meaning that small number of genotypes in each cluster, while in the subset pheno, based on the threshold, we only have two cluster.

One big problem of this clustering analysis is that we do not know if all the traits are in fact contributing for the cluster building, so, we need to find each ones are important and each ones can be removed.

```
sel_var <- clustering(mean_gen, selvar = TRUE)
```

```
## ASI excluded in this step |============                                  | 25% GY excluded in this s
## ------------------------------------------------------------------------
##
## Summary of the adjusted models
## ------------------------------------------------------------------------
##    Model excluded cophenetic remaining cormantel    pvmantel
##  Model 1        -  0.6965867         5 1.0000000 0.000999001
##  Model 2      ASI  0.7051845         4 0.9894093 0.000999001
##  Model 3       GY  0.7142594         3 0.9869084 0.000999001
##  Model 4      MFT  0.7329440         2 0.9616288 0.000999001
## ------------------------------------------------------------------------
## Suggested variables to be used in the analysis
## ------------------------------------------------------------------------
## The clustering was calculated with the  Model 4
## The variables included in this model were...
##  FFT EPP
## ------------------------------------------------------------------------
```

```r
sel_var2 <- clustering(mean_gen_2, selvar = TRUE)
```

```
## ASI excluded in this step |============                                    | 25% GY excluded in this s
## ------------------------------------------------------------------------
##
## Summary of the adjusted models
## ------------------------------------------------------------------------
##    Model excluded cophenetic remaining cormantel    pvmantel
##  Model 1        -  0.6928961         5 1.0000000 0.000999001
##  Model 2      ASI  0.7164971         4 0.9864986 0.000999001
##  Model 3       GY  0.7426920         3 0.9756303 0.000999001
##  Model 4      MFT  0.7307509         2 0.9710941 0.000999001
## ------------------------------------------------------------------------
## Suggested variables to be used in the analysis
## ------------------------------------------------------------------------
## The clustering was calculated with the  Model 3
## The variables included in this model were...
##  FFT MFT EPP
## ------------------------------------------------------------------------
```

For the biggest dataset, only FFT and EPP were selected, while for the small dataset: FFT, MFT and EPP were selected. Based on these clusters, we can extract implement this information in other analysis and see how this group information can be used in favor of the researcher.

Part iv) FW analysis

The FW stands for Finlay-Wilkinson analysis, where the GxE interaction is estimated heterogeneity of the slopes of a regression of individual genotypic performance on an environmental index.

```r
phenoFW <- gxeFw(TD = phenoTD, trait = "GY")
summary(phenoFW)
```

```
## Environmental effects
## =====================
##   Trial     EnvEff  SE_EnvEff  EnvMean SE_EnvMean Rank
## 1    E1 -1.5626337 0.05086093 3.539316  0.4311904    6
## 2    E2  1.5685308 0.05996941 6.667806  0.5341492    3
## 3    E3 -1.6566421 0.05086093 3.445388  0.4449041    7
## 4    E4  1.8280238 0.08139299 6.927077  0.5805836    1
## 5    E5 -2.2546431 0.05996941 2.847898  0.5429888    8
## 6    E6  1.5963523 0.05996941 6.695603  0.5390234    2
## 7    E7  0.0621412 0.05086093 5.162703  0.3412076    5
## 8    E8  0.4188708 0.08139299 5.519128  0.3689585    4
##
## Anova
## =====
##                  Df  Sum Sq Mean Sq  F value    Pr(>F)
## Trial             7  7579.9 1082.84 581.7270 < 2.2e-16 ***
## Genotype        201  1835.0    9.13   4.9046 < 2.2e-16 ***
## Sensitivities   201   536.4    2.67   1.4336  0.000101 ***
## Residual       3024  5628.9    1.86
```

```
## Total         3433 15580.2    4.54
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Most sensitive genotypes
## ========================
##  Genotype  GenMean SE_GenMean Rank     Sens   SE_Sens MSdeviation
##        66 5.681267  0.3380492    1 1.710121 0.2224153    1.881192
##       196 5.074181  0.3380492    2 1.680427 0.2224153    1.569990
##        40 6.147283  0.3380492    3 1.618015 0.2224153    3.181711
##       222 5.626621  0.3380492    4 1.530812 0.2224153    1.975627
##        63 5.269591  0.3380492    5 1.477241 0.2224153    1.299583
```

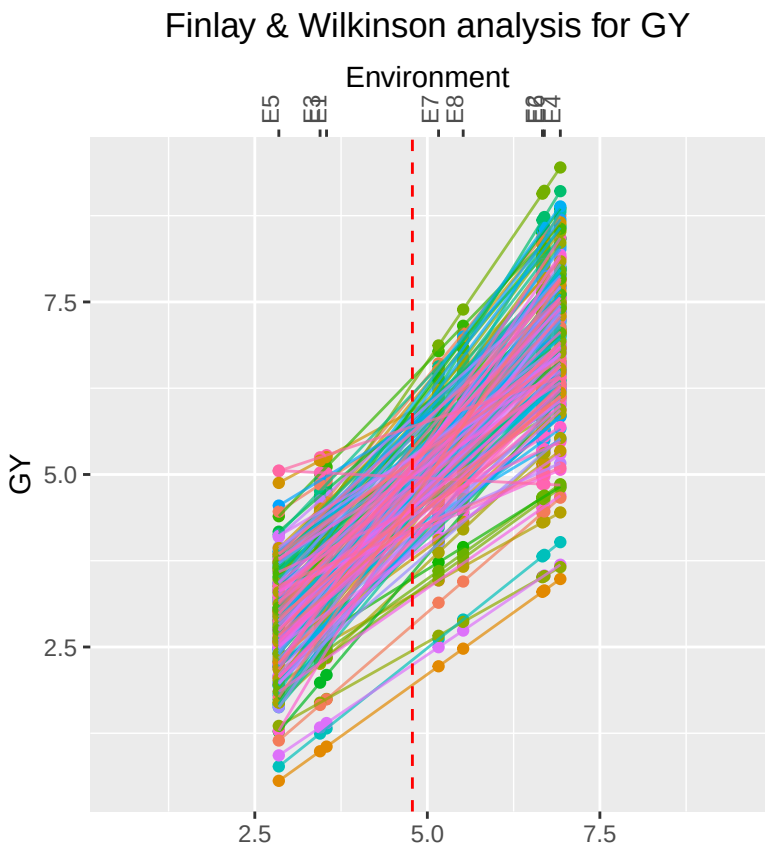
```
phenoFW2 <- gxeFw(TD = phenoTD2, trait = "GY")
summary(phenoFW2)
```

```
## Environmental effects
## =====================
##   Trial     EnvEff SE_EnvEff  EnvMean SE_EnvMean Rank
## 1    E1 -1.7015057 0.2325573 3.561427  0.4326692    6
## 2    E2  1.0360834 0.2742051 6.296761  0.4415681    3
## 3    E3 -1.7645439 0.2325573 3.498441  0.4419319    7
## 4    E4  2.1428439 0.3721626 7.402610  0.6306224    1
## 5    E5 -2.1220884 0.2742051 3.141190  0.4983030    8
## 6    E6  1.6418817 0.2742051 6.902061  0.5400925    2
## 7    E7  0.1025548 0.2325573 5.364002  0.3366600    5
## 8    E8  0.6647743 0.3721626 5.925758  0.3907917    4
##
## Anova
## =====
##                Df Sum Sq Mean Sq F value    Pr(>F)
## Trial           7 373.69  53.385 29.2191 < 2.2e-16 ***
## Genotype        9 150.26  16.696  9.1383  7.13e-11 ***
## Sensitivities   9   7.17   0.796  0.4358    0.9139
## Residual      144 263.09   1.827
## Total         169 794.21   4.699
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Most sensitive genotypes
## ========================
##  Genotype  GenMean SE_GenMean Rank     Sens   SE_Sens MSdeviation
##         4 6.586575  0.3306879    1 1.205905 0.2165685   1.5558270
##         8 5.032008  0.3306879    2 1.199806 0.2165685   1.8503382
##        10 6.089283  0.3306879    3 1.164750 0.2165685   1.3596097
##         9 5.684087  0.3306879    4 1.027283 0.2165685   2.0816706
##        11 5.208903  0.3306879    5 1.001658 0.2165685   0.9558142
```

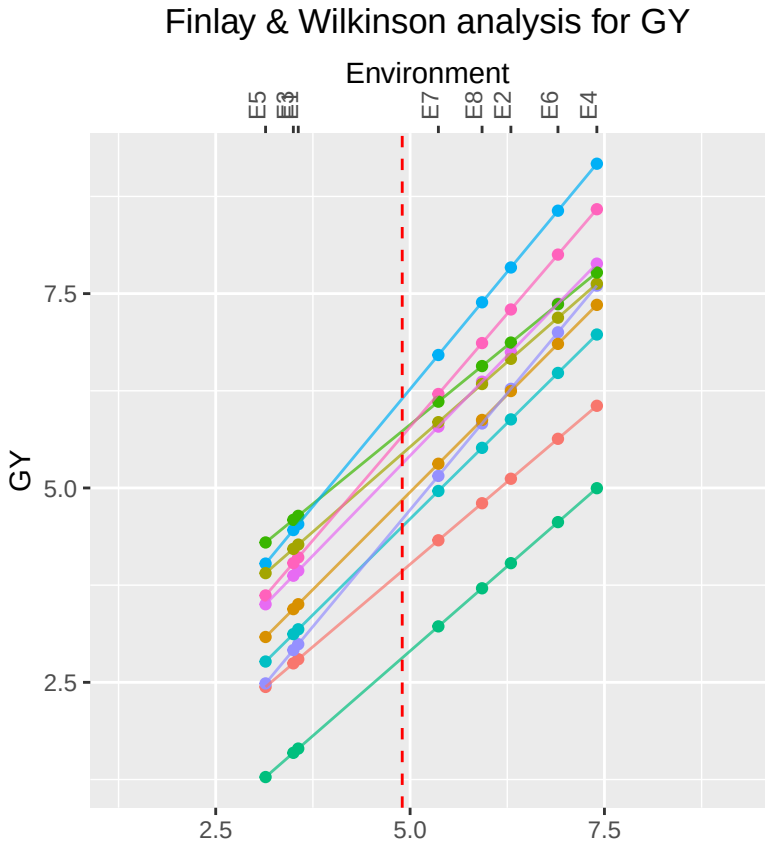
It is interesting that we in the FW output, we can obtain also environment rankings based on this model, as well as, getting some environment effects per se and in environments effects on the hybrids.

Additionally, we can also plot, to take a look how the hybrids performed across the environments and select some specific hybrids based on the rankings that we got in the previous code.

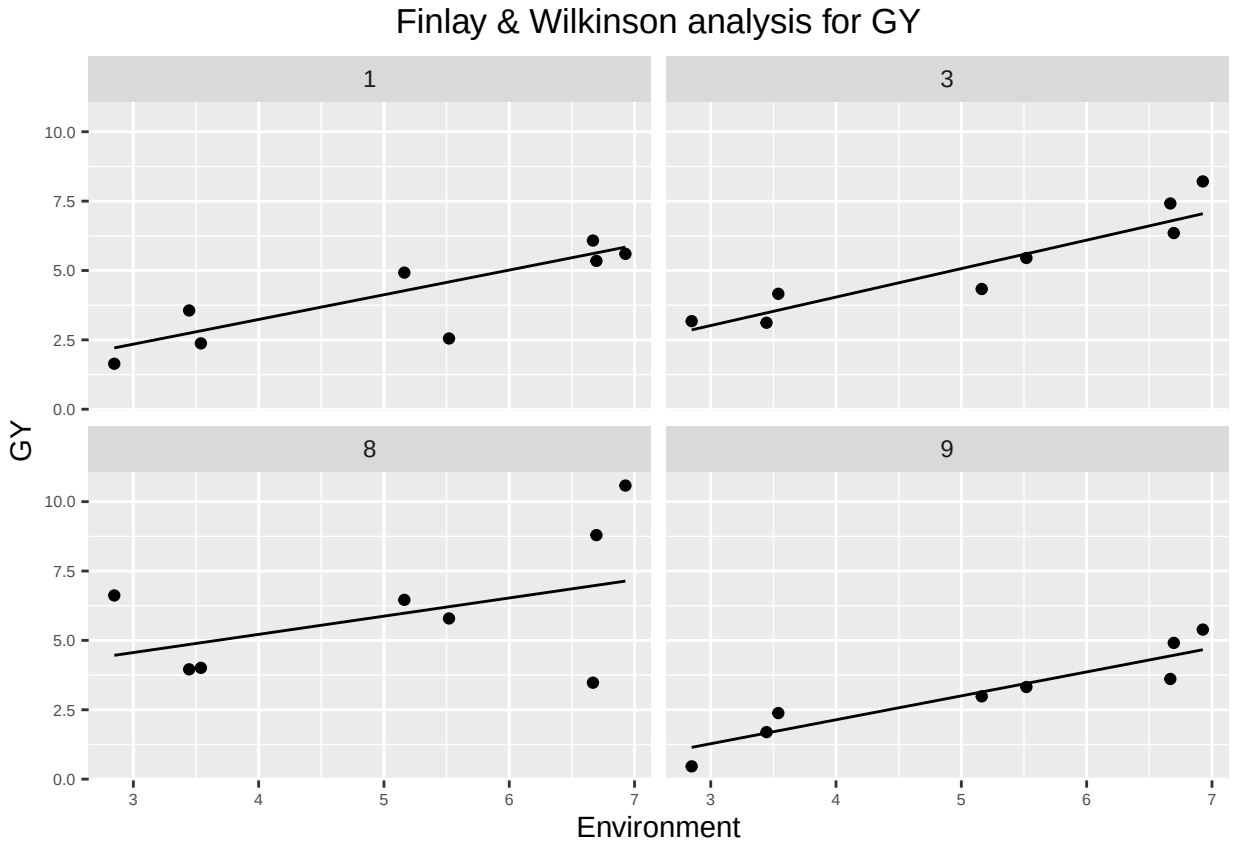
```
plot(phenoFW, plotType = "line")
```




```
plot(phenoFW2, plotType = "line")
```

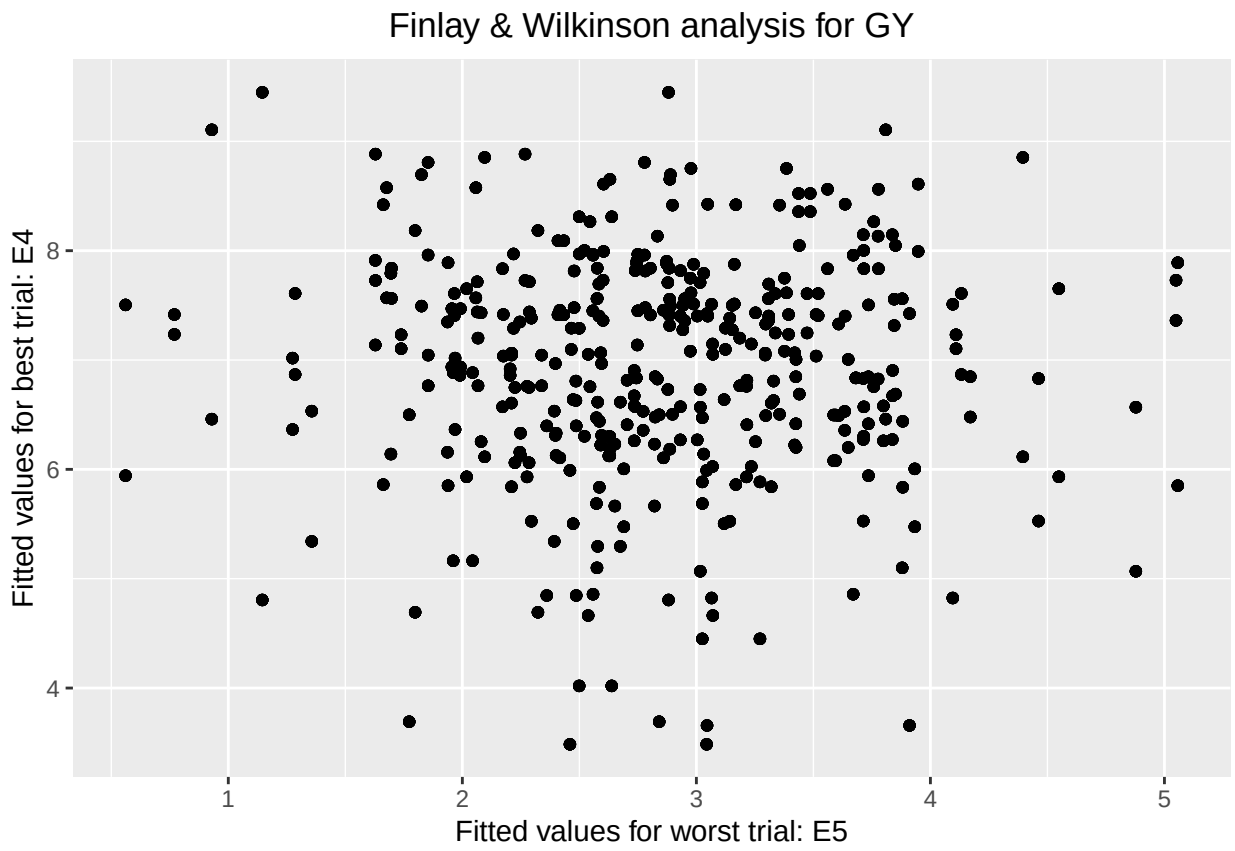



```
plot(phenoFW, plotType = "trellis", genotypes = c("1", "3", "8", "9"))
```

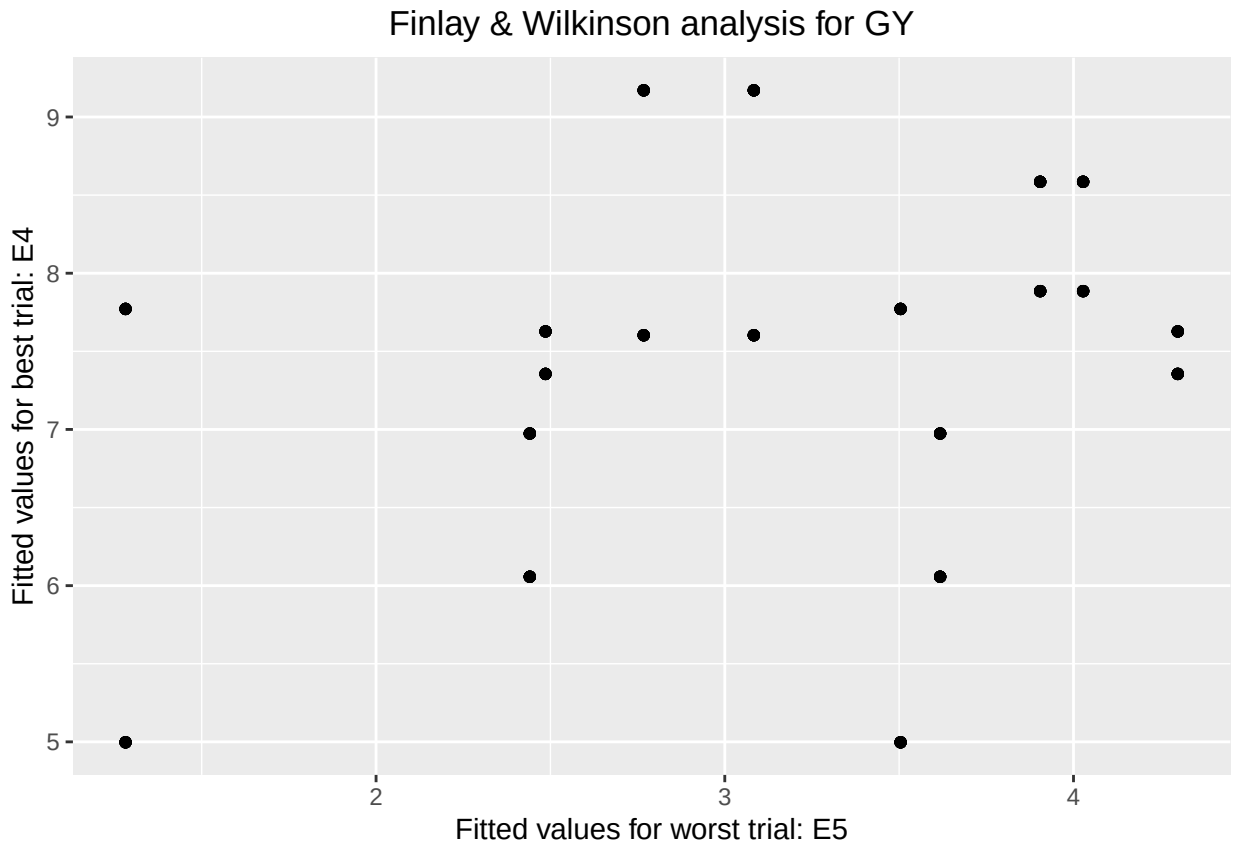


Also, it is possible check the hybrids performance through a scatter plot, comparing their distribution in the best and worst environment.

```
plot(phenoFW, plotType = "scatterFit")
```




```
plot(phenoFW2, plotType = "scatterFit")
```



As the other examples, we also are providing a code to run for all the traits if necessary or desired.

Extra - Multi-trait stability index

We also decide to bring some extra analysis which can help in the breeding practices. In this datataset, we have a lot of traits and environments. So it creates a very complex scenario, making difficult to identify which genotype based on all this information are the best. So, this analysis create a selection index based on all the traits and give to us a very graphical result to identify the best ones.

The code below is to set the models.

And now, we are obtaining the indexes considering all the traits.

```
get_model_data(model_MTSI, what = "WAASBY")
```

```
## Class of the model: waasb

## Variable extracted: WAASBY

## # A tibble: 202 x 6
##    GEN      GY   ASI   FFT   MFT   EPP
##    <chr> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 1      53.5  67.4  61.3  50.9  48.3
##  2 3      73.2  57.2  56.6  54.9  51.1
##  3 4      80.2  42.7  36.4  33.8  71.1
```

```
##  4 8      63.6  53.1  38.9  32.1  79.1
##  5 9      45.3  74.5  69.3  50.6  39.8
##  6 10     71.1  38.0  61.1  64.5  64.1
##  7 11     79.8  51.7  43.3  40.6  80.2
##  8 13     62.7  67.9  62.5  55.8  47.4
##  9 15     63.4  46.9  53.6  60.0  61.0
## 10 16     83.7  60.9  60.9  54.5  71.7
## # i 192 more rows
```

```
get_model_data(model_MTSI2, what = "WAASBY")
```

```
## Class of the model: waasb
## Variable extracted: WAASBY
```

```
## # A tibble: 10 x 6
##    GEN      GY   ASI   FFT   MFT   EPP
##    <chr> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 1      39.2  71.2  59.7  37.4  38.2
##  2 3      70.9  55.0  62.9  60.7  47.4
##  3 4      76.8  31.4  20.5  20.1  72.3
##  4 8      62.5  53.1  29.4  13.4  88.4
##  5 9      34.3  65    77.8  46.6  28.1
##  6 10     70.1  25.1  70.6  92.2  58.8
##  7 11     91.9  46.3  36.0  33.9  88.8
##  8 13     52.2  69.7  58.3  51.0  38.6
##  9 15     65.0  39.2  53.4  73.2  57.0
## 10 16     87.3  62.6  60.4  65    76.9
```

```
index <- mtsi(model_MTSI,
index = "waasby",
mineval = 0.7,
verbose = FALSE)
print(index)
```

```
## ------------------- Correlation matrix used used in factor analysis -----------------
##              GY          ASI         FFT         MFT          EPP
## GY   1.00000000 -0.024481288 -0.08908648 -0.11409874  0.795280112
## ASI -0.02448129  1.000000000  0.42764609 -0.03808517 -0.006151085
## FFT -0.08908648  0.427646085  1.00000000  0.83048715 -0.118027212
## MFT -0.11409874 -0.038085165  0.83048715  1.00000000 -0.141856292
## EPP  0.79528011 -0.006151085 -0.11802721 -0.14185629  1.000000000
##
## -------------------------- Principal component analysis ------------------------------
## # A tibble: 5 x 4
##   PC    Eigenvalues `Variance (%)` `Cum. variance (%)`
##   <chr>       <dbl>          <dbl>               <dbl>
## 1 PC1       2.092          41.85                41.85
## 2 PC2       1.628          32.55                74.40
## 3 PC3       1.026          20.53                94.93
## 4 PC4       0.2042          4.084               99.01
## 5 PC5       0.04953         0.9907             100
##
```

```
## --------------------------------- Initial loadings ---------------------------------
## # A tibble: 5 x 4
##   VAR       PC1     PC2       PC3
##   <chr>   <dbl>   <dbl>     <dbl>
## 1 GY     0.6122 -0.7197 -0.07772
## 2 ASI   -0.3084 -0.3231  0.8915
## 3 FFT   -0.8106 -0.5644  0.007941
## 4 MFT   -0.7517 -0.4374 -0.4731
## 5 EPP    0.6327 -0.7038 -0.04213
##
## ------------------------- Loadings after varimax rotation --------------------------
## # A tibble: 5 x 4
##   VAR        FA1       FA2       FA3
##   <chr>    <dbl>     <dbl>     <dbl>
## 1 GY     0.03858 -0.9471   -0.01718
## 2 ASI   -0.08267  0.005620  0.9937
## 3 FFT   -0.9161   0.05177   0.3657
## 4 MFT   -0.9783   0.07724  -0.1308
## 5 EPP    0.07570 -0.9443    0.006308
##
## --------------------------- Scores for genotypes-ideotype ----------------------------
## # A tibble: 203 x 4
##    GEN      FA1    FA2   FA3
##    <chr>  <dbl>  <dbl> <dbl>
##  1 1     -4.547 -4.512 5.422
##  2 3     -4.719 -5.484 4.405
##  3 4     -3.117 -6.475 3.330
##  4 8     -2.990 -6.062 4.280
##  5 9     -4.743 -3.841 6.144
##  6 10    -5.534 -5.946 2.731
##  7 11    -3.646 -6.845 4.047
##  8 13    -4.852 -4.894 5.366
##  9 15    -4.904 -5.455 3.426
## 10 16    -4.919 -6.765 4.790
## # i 193 more rows
##
## --------------------------- Multitrait stability index -----------------------------
## # A tibble: 202 x 2
##    Genotype  MTSI
##    <chr>    <dbl>
##  1 20       4.496
##  2 174      4.506
##  3 73       4.519
##  4 209      4.554
##  5 58       4.606
##  6 288      4.700
##  7 145      4.746
##  8 217      4.843
##  9 45       4.880
## 10 262      4.928
## # i 192 more rows
##
## ------------------------ Selection differential (variables) -------------------------
## # A tibble: 5 x 11
```

```
##   VAR   Factor     Xo     Xs     SD SDperc     h2     SG SGperc sense     goal
##   <chr> <chr>   <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl> <chr>    <dbl>
## 1 FFT   FA 1   60.22  61.80  1.583   2.628 0.7738 1.225   2.034 increase   100
## 2 MFT   FA 1   58.12  59.09  0.9676  1.665 0.8129 0.7865  1.353 increase   100
## 3 GY    FA 2    5.107  5.389 0.2819  5.520 0.7463 0.2104  4.120 increase   100
## 4 EPP   FA 2   15.20  16.18  0.9791  6.443 0.7650 0.7490  4.929 increase   100
## 5 ASI   FA 3    2.097  2.712 0.6152 29.34  0.7280 0.4478 21.36  increase   100
##
## ---------------------------------- Selected genotypes -----------------------------------
## 20 174 73 209 58 288 145 217 45 262 16 289 19 99 81 256 218 149 303 222 84 254 143 17 109 29 87 176
```

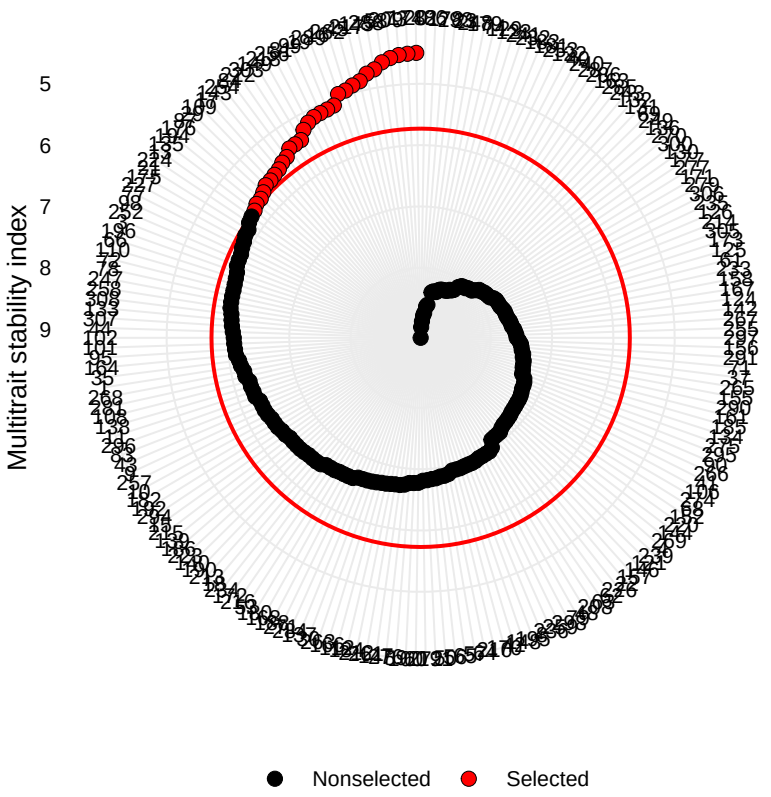
```
index2 <- mtsi(model_MTSI2,
index = "waasby",
mineval = 0.7,
verbose = FALSE)
print(index2)
```

```
## -------------------- Correlation matrix used used in factor analysis -----------------
##              GY        ASI        FFT         MFT        EPP
## GY   1.00000000 -0.5146750 -0.4709605  0.07991295  0.8088237
## ASI -0.51467500  1.0000000  0.3429066 -0.23984224 -0.4449885
## FFT -0.47096047  0.3429066  1.0000000  0.72386842 -0.7357910
## MFT  0.07991295 -0.2398422  0.7238684  1.00000000 -0.3212979
## EPP  0.80882374 -0.4449885 -0.7357910 -0.32129791  1.0000000
##
## --------------------------- Principal component analysis -----------------------------
## # A tibble: 5 x 4
##   PC    Eigenvalues `Variance (%)` `Cum. variance (%)`
##   <chr>       <dbl>          <dbl>               <dbl>
## 1 PC1       2.789          55.78                55.78
## 2 PC2       1.517          30.34                86.11
## 3 PC3       0.5337         10.67                96.79
## 4 PC4       0.1109          2.219               99.00
## 5 PC5       0.04975         0.9951             100
##
## ---------------------------------- Initial loadings -----------------------------------
## # A tibble: 5 x 3
##   VAR       PC1      PC2
##   <chr>   <dbl>    <dbl>
## 1 GY     0.8023 -0.4254
## 2 ASI   -0.5770  0.6012
## 3 FFT   -0.8736 -0.4115
## 4 MFT   -0.4089 -0.8963
## 5 EPP    0.9391 -0.04027
##
## ------------------------- Loadings after varimax rotation ---------------------------
## # A tibble: 5 x 3
##   VAR       FA1     FA2
##   <chr>   <dbl>   <dbl>
## 1 GY     0.8986  0.1310
## 2 ASI   -0.8208  0.1440
## 3 FFT   -0.4615 -0.8482
## 4 MFT    0.1998 -0.9647
## 5 EPP    0.7813  0.5225
```

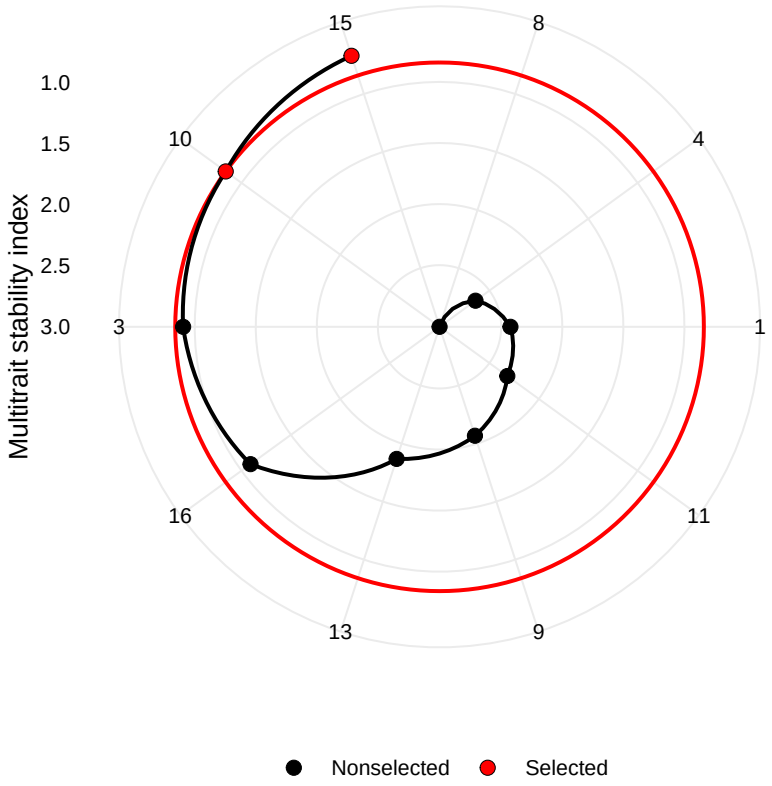
```
##
## --------------------------- Scores for genotypes-ideotype -----------------------------
## # A tibble: 11 x 3
##    GEN        FA1     FA2
##    <chr>    <dbl>   <dbl>
##  1 1     -0.3875  -1.105
##  2 3      1.014   -1.931
##  3 4      1.886   -0.1616
##  4 8      1.148    0.2398
##  5 9     -0.4733  -1.857
##  6 10     2.155   -3.102
##  7 11     2.105   -0.5496
##  8 13     0.06276 -1.440
##  9 15     1.577   -2.111
## 10 16     1.616   -1.688
## 11 ID1    1.388   -2.755
##
## --------------------------- Multitrait stability index ------------------------------
## # A tibble: 10 x 2
##    Genotype   MTSI
##    <chr>     <dbl>
##  1 15       0.6719
##  2 10       0.8410
##  3 3        0.9054
##  4 16       1.092
##  5 13       1.868
##  6 9        2.067
##  7 11       2.319
##  8 1        2.425
##  9 4        2.641
## 10 8        3.005
##
## ------------------------ Selection differential (variables) --------------------------
## # A tibble: 5 x 11
##   VAR   Factor     Xo     Xs       SD   SDperc     h2       SG   SGperc sense
##   <chr> <chr>   <dbl>  <dbl>    <dbl>    <dbl>  <dbl>    <dbl>    <dbl> <chr>
## 1 GY    FA 1    5.266  5.308  0.04256   0.8083 0.8753  0.03725   0.7075 increase
## 2 ASI   FA 1    2.433  1.646 -0.7875  -32.36   0.7585 -0.5973  -24.55   increase
## 3 EPP   FA 1   15.69  16.14   0.4438    2.828  0.8937  0.3966    2.527  increase
## 4 FFT   FA 2   60.88  61.24   0.3646    0.5989 0.7317  0.2668    0.4382 increase
## 5 MFT   FA 2   58.44  59.59   1.152     1.971  0.5924  0.6825    1.168  increase
## # i 1 more variable: goal <dbl>
##
## -------------------------------- Selected genotypes ---------------------------------
## 15 10
```

Lets plot to find the best one according our indexes

```
plot(index)
```




```
plot(index2)
```



Part v) Genomic selection

For this part, it is important to reminder that we are using a new dataset that fits better for the genomic selection analysis.

Checking data stuctures

Before running any analysis, it is important to check the data structures, to avoid mistakes and warnings in the R code.

```
str(Data.Maize)
```

```
## 'data.frame':    927 obs. of  6 variables:
##  $ Gid  : Factor w/ 309 levels "CKDHL0002","CKDHL0003",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Env  : Factor w/ 3 levels "EBU","KAK","KTI": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Rep  : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Yield: num  6.65 6.1 5.07 6.55 6.82 6.88 6.34 7.09 5.28 6.07 ...
##  $ ASI  : num  1.4 1.7 1.6 2.1 2 2.7 2 1.7 2 1.9 ...
##  $ PH   : num  2.48 2.45 2.39 2.31 2.28 2.26 2.46 2.39 2.42 2.29 ...
```

```
str(Gg)
```

```
##  num [1:309, 1:309] 0.2205 0.1318 0.1392 0.0858 0.1243 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:309] "V1" "V2" "V3" "V4" ...
##   ..$ : chr [1:309] "V1" "V2" "V3" "V4" ...
```

In this case, the REP column is not a factor, so, we need to fix it.

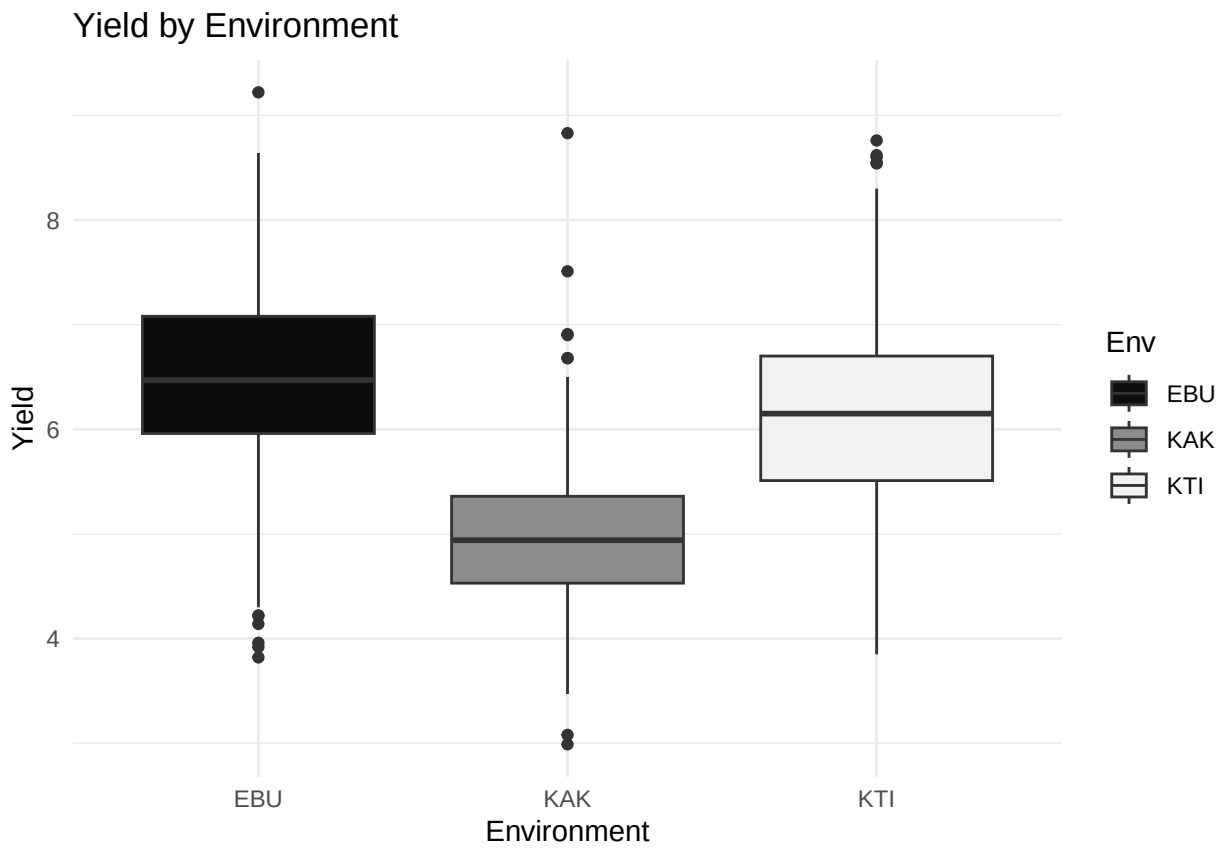
```
Data.Maize$Rep <- as.factor(Data.Maize$Rep)
str(Data.Maize)
```

```
## 'data.frame':    927 obs. of  6 variables:
##  $ Gid  : Factor w/ 309 levels "CKDHL0002","CKDHL0003",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Env  : Factor w/ 3 levels "EBU","KAK","KTI": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Rep  : Factor w/ 1 level "1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Yield: num  6.65 6.1 5.07 6.55 6.82 6.88 6.34 7.09 5.28 6.07 ...
##  $ ASI  : num  1.4 1.7 1.6 2.1 2 2.7 2 1.7 2 1.9 ...
##  $ PH   : num  2.48 2.45 2.39 2.31 2.28 2.26 2.46 2.39 2.42 2.29 ...
```

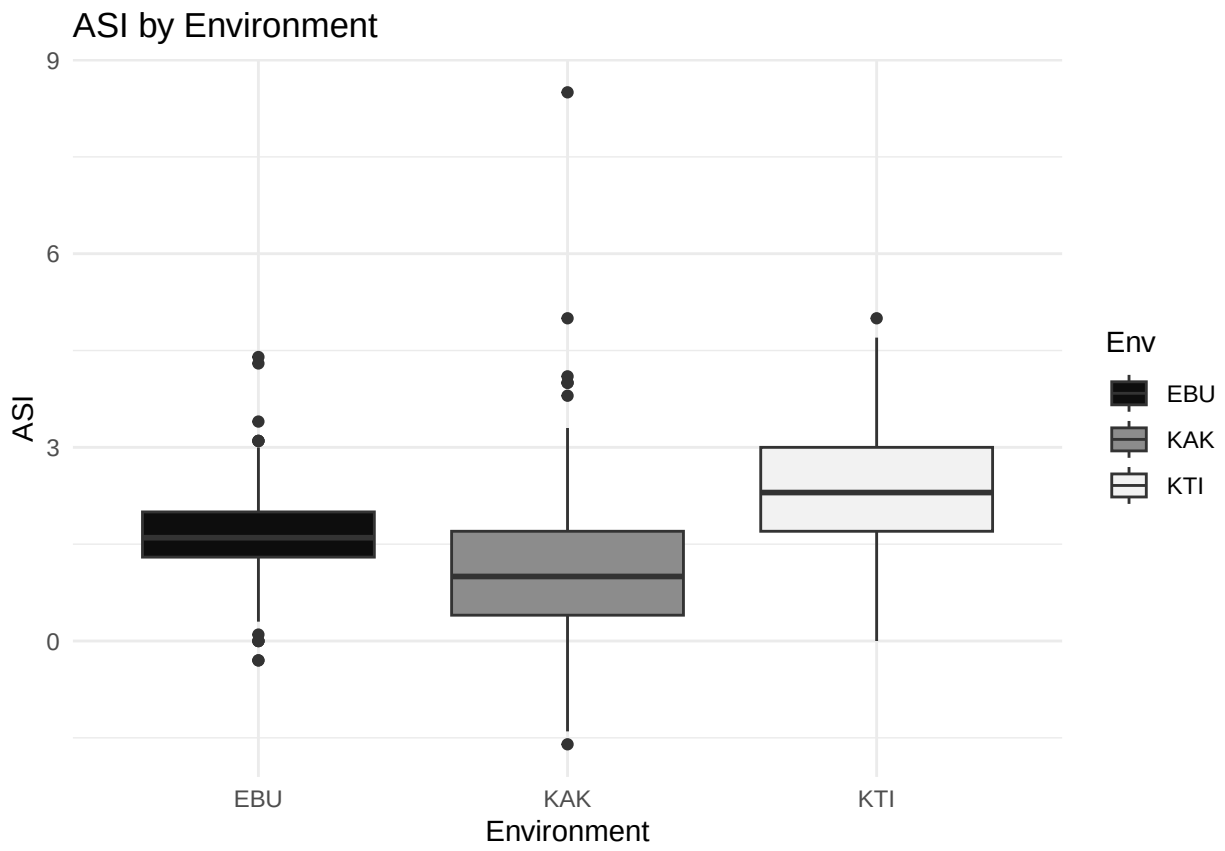
Checking data distribution

In this piece of code as we did for the other dataset, we are checking the distribution and outliers.

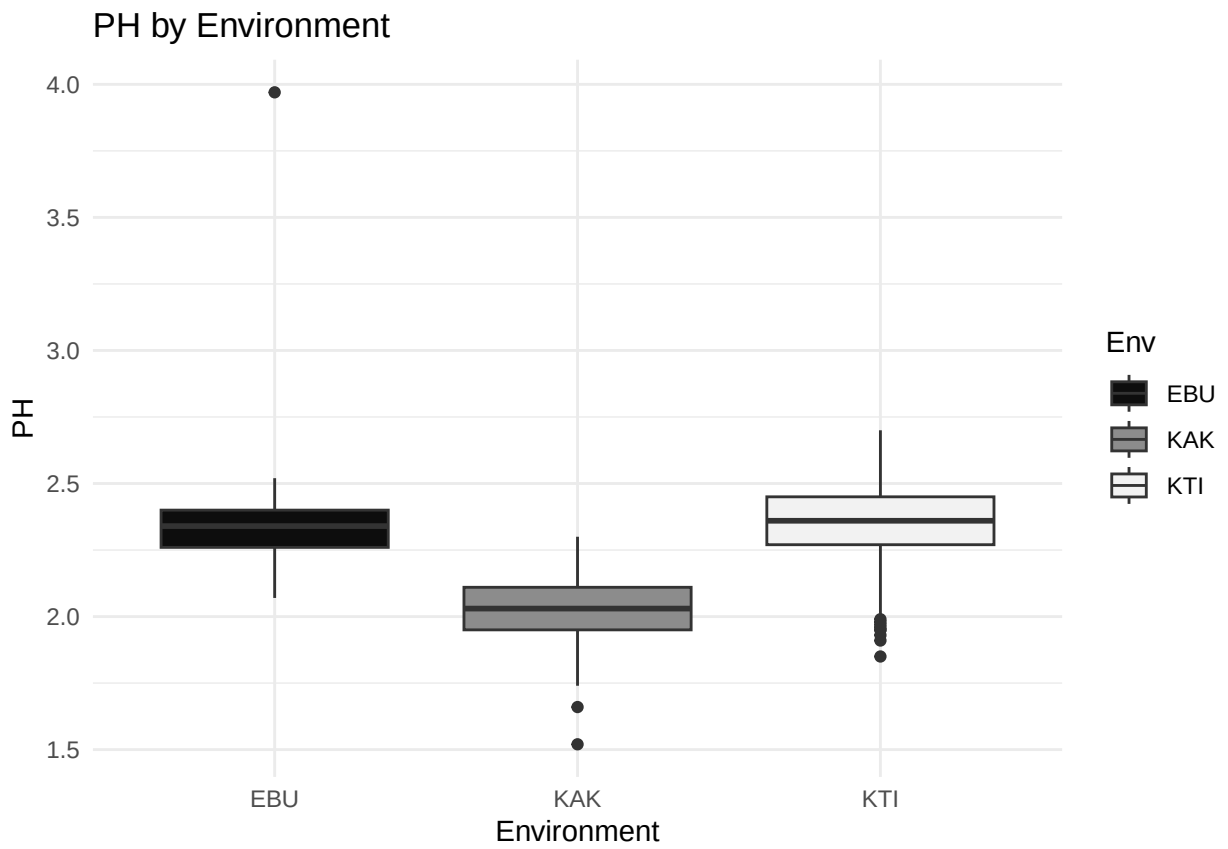
```
ggplot(Data.Maize, aes(x = Env, y = Yield, fill = Env)) +
  geom_boxplot() +
  labs(title = "Yield by Environment", x = "Environment", y = "Yield") +
  scale_fill_manual(values = c("grey5", "grey55", "grey95")) +
  theme_minimal()
```




```
ggplot(Data.Maize, aes(x = Env, y = ASI, fill = Env)) +
  geom_boxplot() +
  labs(title = "ASI by Environment", x = "Environment", y = "ASI") +
  scale_fill_manual(values = c("grey5", "grey55", "grey95")) +
  theme_minimal()
```




```
ggplot(Data.Maize, aes(x = Env, y = PH, fill = Env)) +
  geom_boxplot() +
  labs(title = "PH by Environment", x = "Environment", y = "PH") +
  scale_fill_manual(values = c("grey5", "grey55", "grey95")) +
  theme_minimal()
```



We observe some outliers which can affect our predictions, so, for this case, we are removing them from the dataset. To complete this task, we are using the previous code showed in the beginning of this project.

```
#boxplot.Yield <- boxplot(Data.Maize$Yield)
#boxplot.ASI <- boxplot(Data.Maize$ASI)
#boxplot.PH <- boxplot(Data.Maize$PH)

#outliers <- boxplot.Yield$out; outliers
#Data.Maize <- Data.Maize[-which(Data.Maize$Yield%in%outliers),]
#shapiro.test(Data.Maize$Yield)

#outliers2 <- boxplot.ASI$out; outliers2
#Data.Maize <- Data.Maize[-which(Data.Maize$ASI%in%outliers2),]
#shapiro.test(Data.Maize$ASI)

#outliers3 <- boxplot.PH$out; outliers3
#Data.Maize <- Data.Maize[-which(Data.Maize$PH%in%outliers3),]
#shapiro.test(Data.Maize$PH)
```

Correlation between enviroments

```
yield_wide <- Data.Maize %>%
  select(Gid, Env, Yield) %>%
  spread(key = Env, value = Yield)
```

```r
correlation_matrix <- yield_wide %>%
  select(-Gid) %>%
  cor(use = "complete.obs")

print(correlation_matrix)
```

```
##           EBU       KAK       KTI
## EBU 1.0000000 0.1747700 0.1809038
## KAK 0.1747700 1.0000000 0.2113454
## KTI 0.1809038 0.2113454 1.0000000
```

The correlations between the environments are positive, however, it is not a very strong correlations, meaning that using all the three environments in this analysis can be a good idea.

Correlation between traits

```r
compute_correlations <- function(env_data) {
  cor(env_data %>% select(Yield, ASI, PH), use = "complete.obs")
}

correlations_by_env <- Data.Maize %>%
  group_by(Env) %>%
  summarise(correlation_matrix = list(compute_correlations(cur_data())))
```

```
## Warning: There was 1 warning in `summarise()`.
## i In argument: `correlation_matrix = list(compute_correlations(cur_data()))`.
## i In group 1: `Env = EBU`.
## Caused by warning:
## ! `cur_data()` was deprecated in dplyr 1.1.0.
## i Please use `pick()` instead.
```

```r
correlations_by_env %>%
  filter(Env == "EBU") %>%
  pull(correlation_matrix) %>%
  .[[1]]
```

```
##            Yield        ASI         PH
## Yield  1.0000000 -0.1461431  0.1637575
## ASI   -0.1461431  1.0000000 -0.1200107
## PH     0.1637575 -0.1200107  1.0000000
```

```r
correlations_by_env %>%
  filter(Env == "KAK") %>%
  pull(correlation_matrix) %>%
  .[[1]]
```

```
##             Yield         ASI         PH
## Yield  1.00000000 -0.09744938  0.3669753
## ASI   -0.09744938  1.00000000 -0.1638430
## PH     0.36697528 -0.16384300  1.0000000
```

```
correlations_by_env %>%
  filter(Env == "KTI") %>%
  pull(correlation_matrix) %>%
  .[[1]]
```

```
##            Yield        ASI         PH
## Yield  1.0000000 -0.2200647  0.4091269
## ASI   -0.2200647  1.0000000 -0.2222470
## PH     0.4091269 -0.2222470  1.0000000
```

Interesting we have some positive and negative correlation among the traits. Yield and PH are positive correlated, while both traits are negative correlated to ASI.

Fixing data strcture (again)

```
Data.Maize <- (Data.Maize[order(Data.Maize$Env,Data.Maize$Gid),])
rownames(Data.Maize)=1:nrow(Data.Maize)
head(Data.Maize)
```

```
##         Gid Env Rep Yield ASI   PH
## 1 CKDHL0002 EBU   1  6.65 1.4 2.48
## 2 CKDHL0003 EBU   1  6.10 1.7 2.45
## 3 CKDHL0004 EBU   1  5.07 1.6 2.39
## 4 CKDHL0005 EBU   1  6.55 2.1 2.31
## 5 CKDHL0007 EBU   1  6.82 2.0 2.28
## 6 CKDHL0008 EBU   1  6.88 2.7 2.26
```

Design of matrices

Here we have one of the most important part of this genomic selection. In this step we are developing matrices for the line effects, the environment and the genotype x environment.

So, first of all, lets check our genotypic information data.

```
str(Gg)
```

```
##  num [1:309, 1:309] 0.2205 0.1318 0.1392 0.0858 0.1243 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:309] "V1" "V2" "V3" "V4" ...
##   ..$ : chr [1:309] "V1" "V2" "V3" "V4" ...
```

```
head(Gg, n = c(5,5))
```

```
##            V1        V2        V3         V4        V5
## V1 0.22048906 0.1318021 0.1391629 0.08576769 0.1243159
## V2 0.13180207 0.2086577 0.1273509 0.11743680 0.1621031
## V3 0.13916294 0.1273509 0.2119528 0.10351472 0.1093051
## V4 0.08576769 0.1174368 0.1035147 0.19857022 0.1054898
## V5 0.12431594 0.1621031 0.1093051 0.10548983 0.2038211
```

So, in this datatset, what we got and we are showing here is a G matrix (kinship matrix). Therefore, it is not the molecular markers code. There are different ways to get the G matrix, however, we are not spending much time and how to get this specific matrix, because, we are going more in detail on the other matrices for our genomic selection model.

So, the first step is to conduct a Cholesky factorization. It is a technique used in linear algebra, basically, to make easier next algebraic operations. Note, that our matrix has the same dimensions, however, we have different values.

```
LG <- cholesky(Gg)
str(LG)
```

```
##  num [1:309, 1:309] 0.166 0.138 0.163 0.168 0.138 ...
```

```
head(LG, n = c(5,5))
```

```
##           [,1]       [,2]       [,3]        [,4]        [,5]
## [1,] 0.1662624 -0.2499722 -0.1640082 -0.04246139 -0.01664452
## [2,] 0.1380494 -0.2876658 -0.2318626 -0.06067006  0.06990752
## [3,] 0.1629967 -0.2352840 -0.1536174 -0.02350467 -0.01127936
## [4,] 0.1675681 -0.2197006 -0.1442824 -0.02382445 -0.04645611
## [5,] 0.1381495 -0.2777132 -0.2202040 -0.05486124  0.04410865
```

Next, we are just creating a matrix with the name of our hybrids, where we are coding according the row number, the hybrids order. Basically to associate with our previous matrix.

```
ZG <- model.matrix(~0 + as.factor(Data.Maize$Gid))
str(ZG)
```

```
##  num [1:927, 1:309] 1 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:927] "1" "2" "3" "4" ...
##   ..$ : chr [1:309] "as.factor(Data.Maize$Gid)CKDHL0002" "as.factor(Data.Maize$Gid)CKDHL0003" "as.fa
##  - attr(*, "assign")= int [1:309] 1 1 1 1 1 1 1 1 1 1 ...
##  - attr(*, "contrasts")=List of 1
##   ..$ as.factor(Data.Maize$Gid): chr "contr.treatment"
```

```
head(ZG, n = c(5,5))
```

```
##   as.factor(Data.Maize$Gid)CKDHL0002 as.factor(Data.Maize$Gid)CKDHL0003
## 1                                  1                                  0
## 2                                  0                                  1
## 3                                  0                                  0
## 4                                  0                                  0
## 5                                  0                                  0
##   as.factor(Data.Maize$Gid)CKDHL0004 as.factor(Data.Maize$Gid)CKDHL0005
## 1                                  0                                  0
## 2                                  0                                  0
## 3                                  1                                  0
## 4                                  0                                  1
## 5                                  0                                  0
##   as.factor(Data.Maize$Gid)CKDHL0007
```

```
## 1                                  0
## 2                                  0
## 3                                  0
## 4                                  0
## 5                                  1
```

So, now we have two matrices: one is our G matrix and the other one is matrix containing the genotype order and we are multiplying them. The reasons behind is that the Z.G is basically the LG x 3. We just duplicate the genotypic information by 3, because we are using this matrix as a base matrix to account for environment effects (we have three environments).

```
Z.G <- ZG %*% LG
str(Z.G)
```

```
##  num [1:927, 1:309] 0.166 0.138 0.163 0.168 0.138 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:927] "1" "2" "3" "4" ...
##   ..$ : NULL
```

```
head(Z.G, n = c(5,5))
```

```
##        [,1]       [,2]       [,3]        [,4]        [,5]
## 1 0.1662624 -0.2499722 -0.1640082 -0.04246139 -0.01664452
## 2 0.1380494 -0.2876658 -0.2318626 -0.06067006  0.06990752
## 3 0.1629967 -0.2352840 -0.1536174 -0.02350467 -0.01127936
## 4 0.1675681 -0.2197006 -0.1442824 -0.02382445 -0.04645611
## 5 0.1381495 -0.2777132 -0.2202040 -0.05486124  0.04410865
```

The purpose of this matrix is very similar to the Z.G. But here is to code the hybrids observation to a respective environment.

```
Z.E <- model.matrix(~0 + as.factor(Data.Maize$Env))
str(Z.E)
```

```
##  num [1:927, 1:3] 1 1 1 1 1 1 1 1 1 1 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:927] "1" "2" "3" "4" ...
##   ..$ : chr [1:3] "as.factor(Data.Maize$Env)EBU" "as.factor(Data.Maize$Env)KAK" "as.factor(Data.Maiz
##  - attr(*, "assign")= int [1:3] 1 1 1
##  - attr(*, "contrasts")=List of 1
##   ..$ as.factor(Data.Maize$Env): chr "contr.treatment"
```

```
head(Z.E, n = c(5,5))
```

```
##   as.factor(Data.Maize$Env)EBU as.factor(Data.Maize$Env)KAK
## 1                            1                            0
## 2                            1                            0
## 3                            1                            0
## 4                            1                            0
## 5                            1                            0
##   as.factor(Data.Maize$Env)KTI
```

```
## 1                           0
## 2                           0
## 3                           0
## 4                           0
## 5                           0
```

This matrix is combining basically all the hybrids and environments.

```
ZEG <- model.matrix(~0 + as.factor(Data.Maize$Gid):as.factor(Data.Maize$Env))
str(ZEG)
```

```
##  num [1:927, 1:927] 1 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:927] "1" "2" "3" "4" ...
##   ..$ : chr [1:927] "as.factor(Data.Maize$Gid)CKDHL0002:as.factor(Data.Maize$Env)EBU" "as.factor(Data
##  - attr(*, "assign")= int [1:927] 1 1 1 1 1 1 1 1 1 1 ...
##  - attr(*, "contrasts")=List of 2
##   ..$ as.factor(Data.Maize$Gid): chr "contr.treatment"
##   ..$ as.factor(Data.Maize$Env): chr "contr.treatment"
```

```
head(ZEG, n = c(5,5))
```

```
##   as.factor(Data.Maize$Gid)CKDHL0002:as.factor(Data.Maize$Env)EBU
## 1                                                               1
## 2                                                               0
## 3                                                               0
## 4                                                               0
## 5                                                               0
##   as.factor(Data.Maize$Gid)CKDHL0003:as.factor(Data.Maize$Env)EBU
## 1                                                               0
## 2                                                               1
## 3                                                               0
## 4                                                               0
## 5                                                               0
##   as.factor(Data.Maize$Gid)CKDHL0004:as.factor(Data.Maize$Env)EBU
## 1                                                               0
## 2                                                               0
## 3                                                               1
## 4                                                               0
## 5                                                               0
##   as.factor(Data.Maize$Gid)CKDHL0005:as.factor(Data.Maize$Env)EBU
## 1                                                               0
## 2                                                               0
## 3                                                               0
## 4                                                               1
## 5                                                               0
##   as.factor(Data.Maize$Gid)CKDHL0007:as.factor(Data.Maize$Env)EBU
## 1                                                               0
## 2                                                               0
## 3                                                               0
## 4                                                               0
## 5                                                               1
```

In summary, now we are creating a multi-environment genomic relationship

```
G2 <- kronecker(diag(length(unique(Data.Maize$Env))),data.matrix(Gg))
LG2 <- cholesky(G2)
str(LG2)
```

```
##  num [1:927, 1:927] 0 0 0 0 0 0 0 0 0 0 ...
```

```
head(LG2, n = c(5,5))
```

```
##      [,1] [,2]      [,3]       [,4] [,5]
## [1,]    0    0 0.1662624 -0.2499722    0
## [2,]    0    0 0.1380494 -0.2876658    0
## [3,]    0    0 0.1629967 -0.2352840    0
## [4,]    0    0 0.1675681 -0.2197006    0
## [5,]    0    0 0.1381495 -0.2777132    0
```

We are transforming the genotype-environment interaction matrix according to the multi-environment genomic relationships.

```
Z.EG <- ZEG %*% LG2
str(Z.EG)
```

```
##  num [1:927, 1:927] 0 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:927] "1" "2" "3" "4" ...
##   ..$ : NULL
```

```
head(Z.EG, n = c(5,5))
```

```
##   [,1] [,2]      [,3]       [,4] [,5]
## 1    0    0 0.1662624 -0.2499722    0
## 2    0    0 0.1380494 -0.2876658    0
## 3    0    0 0.1629967 -0.2352840    0
## 4    0    0 0.1675681 -0.2197006    0
## 5    0    0 0.1381495 -0.2777132    0
```

```
Y <- as.matrix(Data.Maize[, -c(1, 2, 3)])
str(Y)
```

```
##  num [1:927, 1:3] 6.65 6.1 5.07 6.55 6.82 6.88 6.34 7.09 5.28 6.07 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:927] "1" "2" "3" "4" ...
##   ..$ : chr [1:3] "Yield" "ASI" "PH"
```

```
head(Y, n = c(5,5))
```

```
##   Yield ASI   PH
## 1  6.65 1.4 2.48
## 2  6.10 1.7 2.45
## 3  5.07 1.6 2.39
## 4  6.55 2.1 2.31
## 5  6.82 2.0 2.28
```

Fitting the model

Model interpretation: Y = Phenotypic trait data for each genotype-environment combination. X (Z.E) = Environment effects in the model Z1 (Z.G.) = Genetic effects, combining genotype information with genetic relationships Z2 (Z.EG) = Genotype-by-environment (GxE) interaction effects nIter = Total number of Markov Chain Monte Carlo iterations for sampling posterior distribution. High number is required to ensure model convergence burnIn = Number of initial iterations that will be discarded (chain stabilization and avoiding bias) Thin = Determines that every second iteration will be kept (others discarded). Thinning reduces autocorrelation in the samples. bs = Number of posterior samples to save.

```
LG <- cholesky(Gg)
ZG <- model.matrix(~0 + as.factor(Data.Maize$Gid))
Z.G <- ZG %*% LG
Z.E <- model.matrix(~0 + as.factor(Data.Maize$Env))
ZEG <- model.matrix(~0 + as.factor(Data.Maize$Gid):as.factor(Data.Maize$Env))
G2 <- kronecker(diag(length(unique(Data.Maize$Env))),data.matrix(Gg))
LG2 <- cholesky(G2)
Z.EG <- ZEG %*% LG2
Y <- as.matrix(Data.Maize[, -c(1, 2, 3)])
fm <- BMTME(Y = Y, X = Z.E, Z1 = Z.G, Z2 = Z.EG,
#nIter =15000, burnIn =10000, thin = 2,bs = 50)
nIter =150, burnIn =100, thin = 2,bs = 50)
```

Extracting covariances

```
COV_TraitGenetic <- fm$varTrait
COV_TraitGenetic
```

```
##        Yield     ASI      PH
## [1,]  0.3026 -0.0335  0.0272
## [2,] -0.0335  0.5109 -0.0015
## [3,]  0.0272 -0.0015  0.0124
```

Covariance matrix between traits

```
COR_TraitGenetic <- cov2cor(COV_TraitGenetic)
COR_TraitGenetic
```

```
##            Yield         ASI          PH
## [1,]  1.00000000 -0.08520055  0.44404154
## [2,] -0.08520055  1.00000000 -0.01884571
## [3,]  0.44404154 -0.01884571  1.00000000
```

Covariance matrix between environments

```
COV_EnvGenetic <- fm$varEnv
COV_EnvGenetic
```

```
##         EBU    KAK    KTI
## [1,] 1.4565 0.3459 0.8645
## [2,] 0.3459 0.3013 0.3095
## [3,] 0.8645 0.3095 0.7902
```

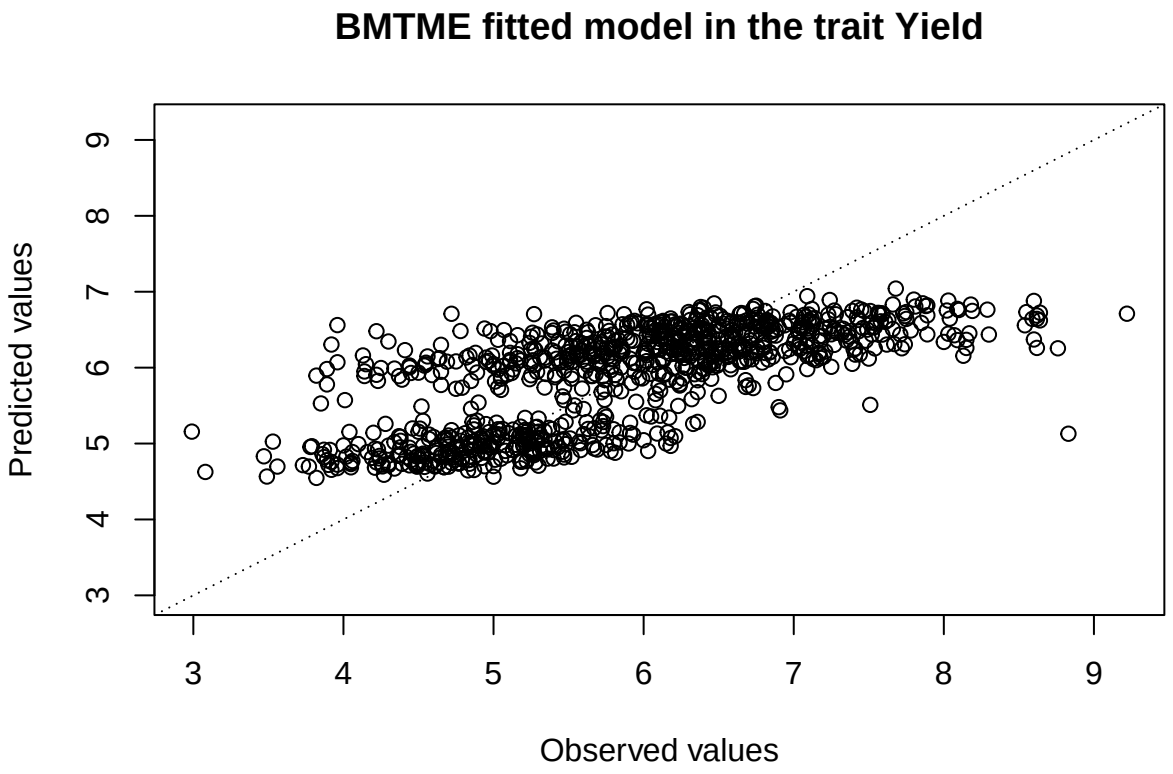
Residual covariance matrix between traits

```
COV_ResGenetic <- fm$vare
COV_ResGenetic
```

```
##        Yield     ASI      PH
## [1,]  0.6397 -0.0933  0.0262
## [2,] -0.0933  0.6045 -0.0153
## [3,]  0.0262 -0.0153  0.0144
```

Predictions

```
plot(fm, trait="Yield")
```



Cross validation

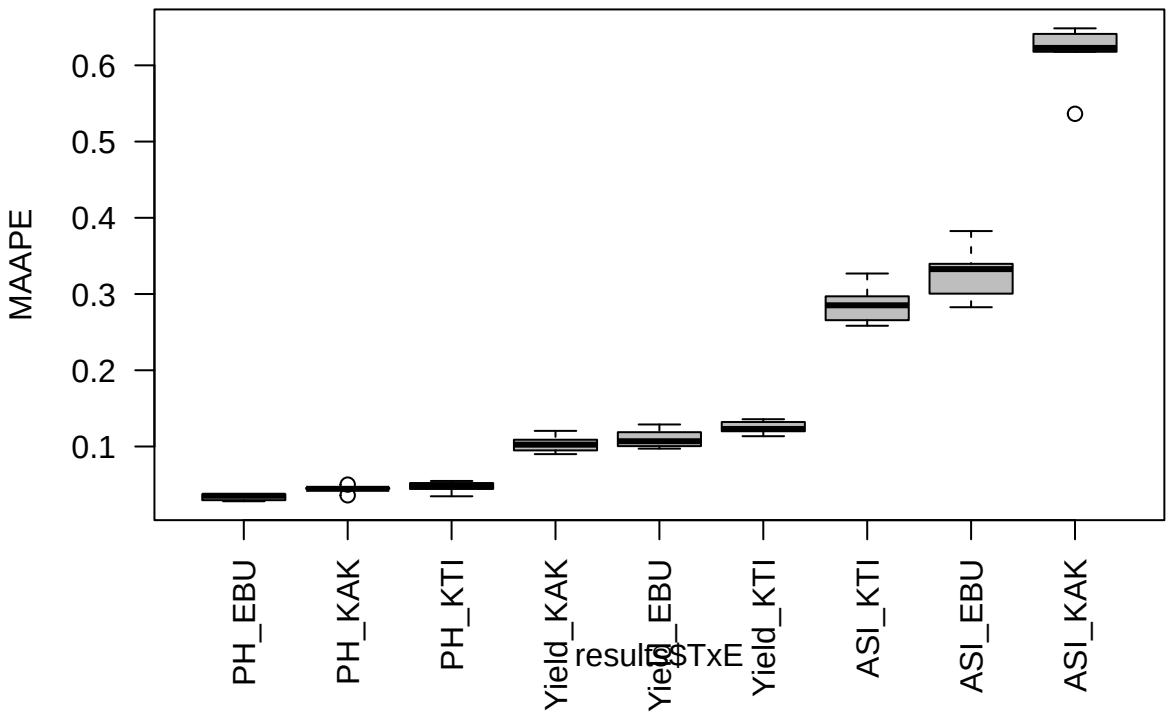
```
pheno <- data.frame(GID = Data.Maize[, 1], Env =
Data.Maize[, 2], Response = Data.Maize[, 4])
CrossV <- CV.KFold(pheno, DataSetID = "GID", K = 5,
set_seed = 123)
pm <- BMTME(Y = Y, X = Z.E, Z1 = Z.G, Z2 = Z.EG,
nIter = 250, burnIn = 50, thin = 2,bs = 50, testingSet = CrossV)
```

Final results

```
summary(pm)
```

```
##   Environment Trait Pearson SE_Pearson  MAAPE SE_MAAPE
## 1         EBU   ASI  0.4679     0.0323 0.3276   0.0172
## 2         EBU    PH  0.2678     0.0458 0.0337   0.0020
## 3         EBU Yield  0.2995     0.0438 0.1105   0.0059
## 4         KAK   ASI  0.3410     0.0681 0.6133   0.0201
## 5         KAK    PH  0.3053     0.0552 0.0441   0.0023
## 6         KAK Yield  0.3162     0.0519 0.1033   0.0054
## 7         KTI   ASI  0.1808     0.0592 0.2866   0.0122
## 8         KTI    PH  0.4273     0.0382 0.0470   0.0035
## 9         KTI Yield  0.2438     0.0437 0.1249   0.0041
```

```
boxplot(pm, select = "MAAPE", las = 2)
```



References

Crossa, J., Y. Beyene, S. Kassa, P. Pérez-Rodríguez, J. M. Hickey, et al., 2013 Genomic prediction in maize breeding populations with genotyping-by-sequencing. G3: Genes|Genomes|Genetics (Bethesda) 3, 1903–1926. https://doi.org/10.1534/g3.113.008227

Dias, K.O.D.G., Gezan, S.A., Guimarães, C.T. et al. Improving accuracies of genomic predictions for drought tolerance in maize by joint modeling of additive and dominance effects in multi-environment trials. Heredity 121, 24–37 (2018). https://doi.org/10.1038/s41437-018-0053-6

Montesinos-López, O. A., A. Montesinos-López, J. Crossa, F. Toledo,O. Pérez-Hernández et al., 2016 A Genomic Bayesian Multi-trait and Multi-environment model. G3: Genes|Genomes|Genetics (Bethesda),6:2725–2744. https://doi.org/10.1534/g3.116.032359

Montesinos-López, O. A., A. Montesinos-López, J. Crossa, J. C. Montesinos-López, F. J. Luna-Vázquez et al., 2017 A Variational Bayes Genomic-Enabled Prediction Method with Genotype · Environment Interaction. G3: Genes, Genomes. Genetics 7: 1833–1853

Osval A Montesinos-López, Abelardo Montesinos-López, Francisco Javier Luna-Vázquez, Fernando H Toledo, Paulino Pérez-Rodríguez, Morten Lillemo, José Crossa, An R Package for Bayesian Analysis of Multi-environment and Multi-trait Multi-environment Data for Genome-Based Prediction, G3 Genes|Genomes|Genetics, Volume 9, Issue 5, 1 May 2019, Pages 1355–1369, https://doi.org/10.1534/g3.119.400126

Olivoto T, Lúcio AD. metan: An R package for multi-environment trial analysis. Methods Ecol Evol. 2020; 11: 783–789. https://doi.org/10.1111/2041-210X.13384