We will start from a multi-environment full diallel experiment with 5 parental lines, in four blocks and 10 environments. The dataset is factitious and it was generated by Monte Carlo simulation, starting from the results reported in Zhang, et al. (2005; see here). The following box shows how we can load the data, after installing (if necessary) and loading the 'lmDiallel' package. In the same box, we use 'dplyr' to transform the explanatory variables into factors.

```
# library(devtools) # Install if necessary
# install_github("OnofriAndreaPG/lmDiallel")
library(lmDiallel)
library(dplyr)

dataset <- read.csv("https://www.casaonofri.it/_datasets/diallelMET.csv", header =
T)
dataset <- dataset %>%
  dplyr::mutate(across(c(Env, Block, Par1, Par2), .fns = factor))
head(dataset)
##   Env Block Par1 Par2 Yield
## 1   1     1    1    1 10.78
## 2   1     1    1    2 12.78
## 3   1     1    1    3 15.23
## 4   1     1    1    4 10.66
## 5   1     1    1    5 14.42
## 6   1     1    2    1 11.84
```

Fixed model fitting

For this full diallel experiment (withe selfs and reciprocals) we can fit the Griffing's model 1, including the General combining Abilities (GCA), total Specific Combining Abilities (tSCA) and reciprocal effects (REC). We also include the environment effect, the block within environment effect and all interactions between genetical effects and the environment. If we regard all effects as fixed, we can code the final model either by using the `lm()` function, or by using the `lm.diallel()` wrapper in the 'lmDiallel' package, as shown in the box below. The two parameterisations are slightly different, although variance partitioning is totally equivalent.

```
dMod <- lm(Yield ~ Env/Block + GCA(Par1, Par2) + tSCA(Par1, Par2) +
            REC(Par1, Par2) + GCA(Par1, Par2):Env +
             tSCA(Par1, Par2):Env + REC(Par1, Par2):Env,
           data = dataset)
# anova(dMod)

dMod2 <- lm.diallel(Yield ~ Par1 + Par2,
                    data = dataset, fct = "GRIFFING1",
                    Env = Env, Block = Block)
anova(dMod2)
## Analysis of Variance Table
##
## Response: Yield
##                    Df Sum Sq Mean Sq  F value Pr(>F)
## Environment         9   10.6    1.17   0.1550 0.9978
## Env:Block          30 3195.1  106.50  14.0554 <2e-16 ***
```

```
## GCA               4 8673.7 2168.42 286.1657 <2e-16 ***
## GCA:Env          36  187.7    5.21   0.6882 0.9175
## tSCA             10 3021.3  302.13  39.8725 <2e-16 ***
## tSCA:Env         90  108.8    1.21   0.1596 1.0000
## Reciprocals      10 4379.7  437.97  57.7989 <2e-16 ***
## Reciprocals:Env  90  352.6    3.92   0.5170 0.9999
## Residuals       720 5455.8    7.58
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Considering the 'diallel' object 'dMod2, the full list of genetical parameters (in each environment) is retrieved by using the `glht()` function in the 'multcomp' package. In the box below we show an excerpt of the output.

```
library(multcomp)
gh <- glht(linfct = diallel.eff(dMod2))
# summary(gh, test = adjusted(type = "none"))
#                Estimate Std. Error t value Pr(>|t|)
# g_1:1 == 0     -3.67200    0.38929  -9.432 7.06e-10 ***
# g_2:1 == 0     -0.63850    0.38929  -1.640 0.113019
# g_3:1 == 0      1.28950    0.38929   3.312 0.002723 **
# g_4:1 == 0      0.03025    0.38929   0.078 0.938658
# g_5:1 == 0      2.99075    0.38929   7.682 3.75e-08 ***
# ts_1:1:1 == 0   1.73500    1.10109   1.576 0.127184
# ts_1:2:1 == 0   0.29150    0.80255   0.363 0.719379
# ts_1:3:1 == 0  -1.43150    0.80255  -1.784 0.086153 .
# ts_1:4:1 == 0  -2.10225    0.80255  -2.619 0.014504 *
# ts_1:5:1 == 0   1.50725    0.80255   1.878 0.071631 .
# ts_2:1:1 == 0   0.29150    0.80255   0.363 0.719379
# ts_2:2:1 == 0   1.03050    1.10109   0.936 0.357942
```

The ANOVA table above shows that genetical effects did not significantly interact with the environment and, therefore, we might be interested in getting estimates of average genetical effects, which can be done by using the `glht()` function and passing the `type = "means"` argument in the `diallel.eff()` call. An excerpt of the result is given in the box below.

```
gh <- glht(linfct = diallel.eff(dMod2, type = "means"))
# summary(gh, test = adjusted(type = "none"))
#    Simultaneous Tests for General Linear Hypotheses
#
# Linear Hypotheses:
#               Estimate Std. Error t value Pr(>|t|)
# g_1:1 == 0    -3.46259    0.12311 -28.127  < 2e-16 ***
# g_2:1 == 0    -0.55351    0.12311  -4.496 0.000127 ***
# g_3:1 == 0     0.64251    0.12311   5.219 1.89e-05 ***
# g_4:1 == 0     0.40521    0.12311   3.292 0.002868 **
# g_5:1 == 0     2.96838    0.12311  24.112  < 2e-16 ***
# ts_1:1:1 == 0  1.53104    0.34820   4.397 0.000165 ***
# ts_1:2:1 == 0  0.21247    0.25379   0.837 0.410125
# ts_1:3:1 == 0 -1.39756    0.25379  -5.507 8.87e-06 ***
# ....
# ....
# r_1:2:1 == 0  -0.20075    0.30776  -0.652 0.519943
```

```
# r_1:3:1 == 0   2.06700    0.30776   6.716 3.98e-07 ***
# r_1:4:1 == 0  -1.33550    0.30776  -4.339 0.000192 ***
# r_1:5:1 == 0  -3.97250    0.30776 -12.908 8.19e-13 ***
# ....
# ...
```

Mixed model fitting

In most cases we might be willing to regard the environment effect as random, so that all the interactions between genetical effects and the environment are random as well. A mixed model can be fitted by using the `mmer.diallel()` wrapper, that is is available in the 'ImDiallel' package. The call is very similar to a `lm.diallel()` call, as shown in the box above; we can use the 'type = "environment"' argument to specify that we want to include random environment effects. The fit can take quite a few seconds (or minutes, depending on your device…). Fixed effects and variance components can be easily retrieved from the model object, as shown in the box below.

```
dMod3 <- mmer.diallel(Yield ~ Par1 + Par2,
                      data = dataset, fct = "GRIFFING1",
                      Env = Env, Block = Block, type = "environment")
dMod3$beta #fixed effects
##    Trait                  Effect  Estimate Std.Error     t.value
## 1      Y             (Intercept) 16.272390 0.2800709  58.1009636
## 2      Y     GCA(Par1, Par2)g_1 -3.462590 0.1005471 -34.4374961
## 3      Y     GCA(Par1, Par2)g_2 -0.553515 0.1005471  -5.5050326
## 4      Y     GCA(Par1, Par2)g_3  0.642510 0.1005471   6.3901402
## 5      Y     GCA(Par1, Par2)g_4  0.405210 0.1005471   4.0300520
## 6      Y tSCA(Par1, Par2)ts_1:1  1.531040 0.3246046   4.7166305
## 7      Y tSCA(Par1, Par2)ts_1:2  0.212465 0.2365942   0.8980143
## 8      Y tSCA(Par1, Par2)ts_1:3 -1.397560 0.2365942  -5.9069910
## 9      Y tSCA(Par1, Par2)ts_1:4 -1.866010 0.2365942  -7.8869632
## 10     Y tSCA(Par1, Par2)ts_2:2  1.946890 0.3246046   5.9977275
## 11     Y tSCA(Par1, Par2)ts_2:3  0.896115 0.2365942   3.7875606
## 12     Y tSCA(Par1, Par2)ts_2:4 -3.674085 0.2365942 -15.5290556
## 13     Y tSCA(Par1, Par2)ts_3:3 -0.187160 0.3246046  -0.5765784
## 14     Y tSCA(Par1, Par2)ts_3:4  1.134515 0.2365942   4.7951930
## 15     Y tSCA(Par1, Par2)ts_4:4  4.228940 0.3246046  13.0279727
## 16     Y   REC(Par1, Par2)r_1:2 -0.200750 0.2869127  -0.6996903
## 17     Y   REC(Par1, Par2)r_1:3  2.067000 0.2869127   7.2042832
## 18     Y   REC(Par1, Par2)r_1:4 -1.335500 0.2869127  -4.6547268
## 19     Y   REC(Par1, Par2)r_1:5 -3.972500 0.2869127 -13.8456774
## 20     Y   REC(Par1, Par2)r_2:3  1.345250 0.2869127   4.6887092
## 21     Y   REC(Par1, Par2)r_2:4 -1.277750 0.2869127  -4.4534460
## 22     Y   REC(Par1, Par2)r_2:5 -2.285625 0.2869127  -7.9662747
## 23     Y   REC(Par1, Par2)r_3:4  4.424875 0.2869127  15.4223768
## 24     Y   REC(Par1, Par2)r_3:5 -0.038625 0.2869127  -0.1346229
## 25     Y   REC(Par1, Par2)r_4:5 -2.149875 0.2869127  -7.4931342
dMod3$varcomp #variance components
##               VarComp  VarCompSE
## Env        0.00000000 0.42671998
## Env:Block  2.99662037 0.84276821
## GCA:Env   -0.03826627 0.03707895
```

```
## tSCA:Env   0.00000000 0.14493517
## REC:Env    0.00000000 0.13004754
## Residuals  6.58550954 0.34464357
```

Random model fitting

In other instances, our aim is to estimate variance components for all genetical effects and, therefore, we might like to regard all effects as random. This is easily done by changing the call to `mmer.diallel()` and replacing `type = "environment"` with `type = "all"`.

```
dMod4 <- mmer.diallel(Yield ~ Par1 + Par2,
                      data = dataset, fct = "GRIFFING1",
                      Env = Env, Block = Block, type = "all")
dMod4$beta #fixed effects
##   Trait      Effect Estimate Std.Error  t.value
## 1     Y (Intercept) 16.45538  1.920185 8.569682
dMod4$varcomp #variance components
##                VarComp  VarCompSE
## Env         0.00000000 0.42621427
## Env:Block   2.99608118 0.84185447
## GCA         4.11923254 3.41043097
## tSCA        4.68884763 2.14244233
## REC         5.39232126 2.44832836
## GCA:Env    -0.03828058 0.03706453
## tSCA:Env    0.00000000 0.14494177
## REC:Env     0.00000000 0.13005751
## Residuals   6.58560305 0.34466898
```

Obviously, a similar procedure can be used to fit all other diallel models to multi-environment diallel experiments.