Diallel mating designs are often used by plant breeders to compare the possible crosses between a set of genotypes. In spite of such widespread usage, the process of data analysis in R is not yet strightforward and it is not clear which tool should be routinely used. We recently gave a small contribution by publishing a paper in Plant Breeding (Onofri et al., 2020), where we advocated the idea that models for diallel crosses are just a class of general linear models, that should be fit by Ordinary Least Squares (OLS) or REstricted Maximum Likelihood methods (REML).

In that paper, we presented `lmDiallel`, a new R package to fit diallel models, which we followed up with a series of three blog posts, giving more detail about the package (see here), about the Hayman's models type 1 (see here) and type 2 (see here). These latter models can be used to describe the data from full diallel experiments.

In this fourth post we are going to talk about a very flexible family of models, that was introduced by Griffing in 1956 and it is still very used in plant breeding, to estimate General Combining Ability (GCA) and Specific Combining Ability (SCAs). The equations take different forms, to account for all possible mating schemes.

With full diallel experiments (including selfs and reciprocals; **mating scheme 1**), the model is very similar to Hayman's model type 1, except that reciprocal effects are not parted into RGCA and RSCA (Reciprocal General Combining Ability and Reciprocal Specific Combining Ability). The equation is:

$$y _{ijk} = \mu + \textrm{g}_i + \textrm{g}_j + \textrm{ts}_{ij} + r_{ij} + \varepsilon_{ijk} \quad\quad\quad (1)$$

where μ is the expected value (the overall mean, in the balanced case) and ε_{ijk} is the residual random error term for the observation in the k^{th} block and with the parentals i and j. All the other terms correspond to genetic effects, namely:

1. the $\textrm{g}_i$ and $\textrm{g}_j$ terms are the General Combining Abilities (GCAs) of the i^{th} and j^{th} parents.
2. The ts_{ij} term is the total Specific Combining Ability (SCA) for the combination ij.
3. The r_{ij} term is the reciprocal effect for a specific ij combination.

When the reciprocal crosses are not available (**mating scheme 2**), the term $\textrm{r}_{ij}$ needs to be dropped, so that the model reduces to:

$$y _{ijk} = \mu + \textrm{g}_i + \textrm{g}_j + \textrm{ts}_{ij} + \varepsilon_{ijk} \quad\quad\quad (2)$$

When the reciprocals are available, but selfs are missing (**mating scheme 3**), the model is similar to equation 1, but the term $\textrm{ts}_{ij}$ is replaced by $\textrm{s}_{ij}$ (we use a different symbol, because the design matrix is slightly different and needs a different coding):

$$y _{ijk} = \mu + \textrm{g}_i + \textrm{g}_j + \textrm{s}_{ij} + r_{ij} + \varepsilon_{ijk} \quad\quad\quad (3)$$

Finally, when neither selfs nor reciprocals are available (**mating scheme 4**), the equation reduces to:

$$y _{ijk} = \mu + \textrm{g}_i + \textrm{g}_j + \textrm{s}_{ij} + \varepsilon_{ijk} \quad\quad\quad (4)$$

Let’s see how to fit the above models by using a set of examples with different mating schemes.

Example 1: a full diallel experiment

The example in Hayman (1954) relates to a complete diallel experiment with eight parental lines, producing 64 combinations (8 selfs + 28 crosses with 2 reciprocals each). The R dataset is included in the ‘lmDiallel’ package; in the box below we load the data, after installing (if necessary) and loading the ‘lmDiallel’ package (see box below). For brevity, some R commands are shown but not executed (they are commented out)

```
# library(devtools) # Install if necessary
# install_github("OnofriAndreaPG/lmDiallel")
library(lmDiallel)
data("hayman54")
```

For this complete diallel experiment we can fit equation 1, by including GCAs, tSCAs and reciprocal effects. Please, note that excluding any of these effects results in unreliable estimates of the residual mean square. We can use either the `lm()` or the `lm.diallel()` functions, as shown in the box below.

```
contrasts(hayman54$Block) <- "contr.sum"
dMod <- lm(Ftime ~ Block + GCA(Par1, Par2) + tSCA(Par1, Par2) +
             REC(Par1, Par2), data = hayman54)
dMod2 <- lm.diallel(Ftime ~ Par1 + Par2, Block = Block,
                    data = hayman54, fct = "GRIFFING1")
# summary(dMod2)
anova(dMod2)
## Analysis of Variance Table
##
## Response: Ftime
##             Df Sum Sq Mean Sq F value    Pr(>F)
## Block        1    142     142  0.3416   0.56100
## GCA          7 277717   39674 95.1805 < 2.2e-16 ***
## SCA         28 102238    3651  8.7599 6.656e-13 ***
## Reciprocals 28  19112     683  1.6375   0.05369 .
## Residuals   63  26260
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In order to obtain the full list of genetical parameters, we can use the `glht()` function in the `multcomp` package, together with the `diallel.eff()` function in the ‘lmDiallel’ package. An excerpt of the results is shown below.

```
library(multcomp)
gh <- glht(linfct = diallel.eff(dMod2))
# summary(gh, test = adjusted(type = "none"))
#    Simultaneous Tests for General Linear Hypotheses
#
# Linear Hypotheses:
#                   Estimate Std. Error t value Pr(>|t|)
# Intercept == 0  1.629e+02  1.805e+00  90.270  < 2e-16 ***
# g_A == 0        4.620e+01  3.376e+00  13.683 2.17e-13 ***
# g_B == 0       -2.459e+01  3.376e+00  -7.282 9.83e-08 ***
# g_C == 0        4.963e+01  3.376e+00  14.702 4.13e-14 ***
# g_D == 0        1.835e+01  3.376e+00   5.436 1.07e-05 ***
# g_E == 0       -2.093e+01  3.376e+00  -6.199 1.47e-06 ***
# g_F == 0        2.445e+00  3.376e+00   0.724 0.475340
# g_G == 0       -4.471e+01  3.376e+00 -13.244 4.57e-13 ***
# g_H == 0       -2.640e+01  3.376e+00  -7.819 2.71e-08 ***
# ts_A:A == 0     3.371e+01  1.263e+01   2.669 0.012941 *
# ts_A:B == 0    -3.151e+01  9.023e+00  -3.492 0.001731 **
# ...
# ...
```

Example 2: no reciprocals

As an example of a diallel experiments with no reciprocals, we consider the data reported in Lonnquist and Gardner (1961) relating to the yield of 21 maize genotypes, obtained from six male and six female parentals. The dataset is available as `lonnquist61` in the `lmDiallel` package and the model fitting process is very similar to that shown before for the mating scheme 1, apart from the fact that we fit equation 2 instead of equation 1. In the ‘lm()’ call, we use the `GCA()` and `tSCA()` functions, while in the `lm.diallel()` call, we set the argument ‘fct’ to “GRIFFING2”.

```
rm(list=ls())
data(lonnquist61)
dMod <- lm(Yield ~ GCA(Par1, Par2) + tSCA(Par1, Par2),
           data = lonnquist61)
dMod2 <- lm.diallel(Yield ~ Par1 + Par2,
                    data = lonnquist61, fct = "GRIFFING2")
```

In this case the dataset has no replicates and, for the inferences, we need to provide an estimate of the residual mean square and degrees of freedom (see box below). If we have fitted the model by using the `lm()` function, the resulting ‘lm’ object can be explored by using the `summary.diallel()` and `anova.diallel()` functions. Otherwise, if we have fitted the model with the `lm.diallel()` function, the resulting ‘diallel’ object can be explored by using the `summary()` and `anova()` methods. See the box below for an example: the results are, obviously, the same.

```
# summary.diallel(dMod, MSE = 7.1, dfr = 60)
anova.diallel(dMod, MSE = 7.1, dfr = 60)
## Analysis of Variance Table
##
## Response: Yield
##                  Df Sum Sq Mean Sq F value    Pr(>F)
## GCA(Par1, Par2)   5 234.23  46.846  6.5980 5.923e-05 ***
## tSCA(Par1, Par2) 15 238.94  15.929  2.2436   0.01411 *
## Residuals        60          7.100
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# summary(dMod2, MSE = 7.1, dfr = 60)
anova(dMod2, MSE = 7.1, dfr = 60)
## Analysis of Variance Table
##
## Response: Yield
##           Df Sum Sq Mean Sq F value    Pr(>F)
## GCA        5 234.23  46.846  6.5980 5.923e-05 ***
## SCA       15 238.94  15.929  2.2436   0.01411 *
## Residuals 60          7.100
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Also for the diallel object, we can retrieve the full list of genetical parameters with the `glht()` function, by using the same syntax as shown above.

```
gh <- glht(linfct = diallel.eff(dMod2, MSE = 7.1, dfr = 60))
# summary(gh, test = adjusted(type = "none"))
```

Example 3: no selfs

When the experimental design includes the reciprocal crosses but not the selfs, we can fit Equation 3. As an example, we take the same dataset as before (‘hayman54’), but we remove the selfs by using ‘dplyr’. The fitting process is the same as shown above and only the model specification is changed.

```
library(dplyr)
data(hayman54)
hayman54b <- hayman54  %>%
  filter(Par1 != Par2)

dMod <- lm(Ftime ~ Block + GCA(Par1, Par2) +
             SCA.G3(Par1, Par2) + REC.G3(Par1, Par2),
           data = hayman54b)
dMod2 <- lm.diallel(Ftime ~ Par1 + Par2, Block = Block,
                    data = hayman54b, fct = "GRIFFING3")
anova(dMod2)
## Analysis of Variance Table
##
## Response: Ftime
##             Df Sum Sq Mean Sq F value    Pr(>F)
## Block        1    329   329.1  0.8367   0.36432
## GCA          7 168923 24131.9 61.3479 < 2.2e-16 ***
## SCA         20  37289  1864.4  4.7398 2.318e-06 ***
## Reciprocals 28  19112   682.6  1.7352   0.04052 *
## Residuals   55  21635
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
gh <- glht(linfct = diallel.eff(dMod2))
# summary(gh, test = adjusted(type = "none"))
```

Example 4: no reciprocals, no selfs

In this final example, we consider a mating scheme where neither the reciprocal crosses nor the selfs are included (mating scheme 4). The dataset is taken from the original Griffing’s paper (Griffing, 1956) and it is available as ‘Griffing56’ in the ‘lmDiallel’ package. The analysis proceeds in the very same fashion as above, apart from the fact that we fit Equation 4, instead of 3 and that we input the appropriate residual error term to obtain the correct inferences, as the original dataset does not contain the replicated data.

```
data("griffing56")

dMod <- lm(Yield ~ GCA(Par1, Par2) + SCA.G3(Par1, Par2),
           data = griffing56)
anova.diallel(dMod, MSE = 21.05, dfr = 2558)
## Analysis of Variance Table
##
## Response: Yield
##                      Df  Sum Sq Mean Sq F value    Pr(>F)
## GCA(Par1, Par2)       8 18606.0 2325.75 110.487 < 2.2e-16 ***
## SCA.G3(Par1, Par2)   27  9164.9  339.44  16.125 < 2.2e-16 ***
## Residuals          2558           21.05
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
dMod2 <- lm.diallel(Yield ~ Par1 + Par2,
                    data = griffing56, fct = "GRIFFING4")
anova(dMod2, MSE = 21.05, dfr = 2558)
## Analysis of Variance Table
##
## Response: Yield
##             Df  Sum Sq Mean Sq F value    Pr(>F)
## GCA          8 18606.0 2325.75 110.487 < 2.2e-16 ***
## SCA         27  9164.9  339.44  16.125 < 2.2e-16 ***
## Residuals 2558           21.05
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# summary(dMod2, MSE = 21.05, dfr = 2558)

gh <- glht(linfct = diallel.eff(dMod2, MSE = 21.05, dfr = 2558))
# summary(gh, test = adjusted(type = "none"))
```

Estimation of variance components (random genetic effects)

If we intend to regard the genetic effects as random and to estimate variance components, we can use the `mmer()` function in the 'sommer' package (Covarrubias-Pazaran, 2016), although we need to code a bunch of dummy variables. In order to make things simpler for routine experiments, we have coded the `mmer.diallel()` wrapper using the same syntax as the `lm.diallel()` function. The exemplary code is given in the box below, relating to Equation 2, although the other equations can be fitted in a similar manner.

```
# Random genetic effects
mod1m <- mmer.diallel(Yield ~ Par1 + Par2,
                      data = lonnquist61,
                      fct = "GRIFFING2")
mod1m
##        VarComp VarCompSE
## GCA   3.863695  3.769373
## tSCA 15.930144  5.819217
```