The model we are going to talk about is used to describe the results of full (complete) diallel experiments, where we have crosses + reciprocals + selfs. If you are not sure what a diallel experiment is, we suggest you go back to one of our previous posts on this sequence, where we give some preliminary information for beginners. Otherwise, we can proceed to the motivating example.

The example

In this post we will use the same example as provided in the original Hayman's paper (Hayman, 1954), relating to a complete diallel experiment with eight parental lines. 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).

```
# library(devtools) # Install if necessary
# install_github("OnofriAndreaPG/lmDiallel")
library(lmDiallel)
data("hayman54")
head(hayman54)
##   Block Par1 Par2 Ftime
## 1     1    A    A   276
## 2     1    A    B   156
## 3     1    A    C   322
## 4     1    A    D   250
## 5     1    A    E   162
## 6     1    A    F   193
```

The Hayman's model type 2

The Hayman's model type 2 is derived from type 1 (see our previous post), by partitioning the tSCA effect in three additive components. The equation is:

$$y_{ijk} = \left\{ {\begin{array}{ll} \mu + \gamma_k + \textrm{g}_i + \textrm{g}_j + m + d_i + d_j + s_{ij} + rg^a_i + rg^b_j + rs_{ij} + \varepsilon_{ijk} & \textrm{for} \quad i \neq j\\ \mu + \gamma_k + 2\, \textrm{g}_i – (n – 1)m – (n – 2)d_i + \varepsilon_{ijk} & \textrm{for} \quad i = j \end{array}} \right.$$

where μ is the expected value (the overall mean, in the case of fully orthogonal designs), n is the number of parentals and ε_{ijk} is the residual random error term. 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 (see here).
2. The rg^a_i and rg^b_j terms are the **reciprocal general combining abilities** (RGCAs) for the i^{th} and j^{th} parents (see here).
3. The m term relates to the difference between the average value of all observations and the average values of crosses (**Mean Dominance Deviation**; MDD).
4. The d_i and df_j terms relate to the differences between the yield of each selfed parent (Y_{ij}, with $i = j$) and the average yield of all selfed parents (**dominance deviation** for the i^{th} parent; DD).
5. The term s_{ij} is the SCA effect for the combination ij.
6. The rs_{ij} term is the **reciprocal specific combining ability** (RSCA) for a specific

\(ij\) combination, that is the discrepancy between the performances of the two reciprocals (e.g, A \(\times\) B vs. B \(\times\) A)(see here).

Similarly to type 1, the Hayman's model type 2 considers the genetical effects as differences with respect to the intercept \(\mu\), that is the mean of all observations (when the design is orthogonal). However, with respect to type 1, this latter model permits the estimation of a higher number of genetic effects (GCAs, RGCAs, MDD, DDs, SCAs and RSCAs) and provides an approach to quantify heterotic effects. We should consider that, due to unbalance (the number of crosses is never equal to the number of selfs), it is necessary to introduce some coefficients (i.e. \(n – 1\) and \(n – 2\) in Equation 1), which do not have an obvious meaning. In future posts we will see that other diallel models were proposed, which account for heterotic effects in a different manner (Gardner and Eberhart, 1966).

Model fitting with R

Let's assume that all effects are fixed, apart from the residual error effect. Consequently, Equation 1 is a specific parameterisation of a general linear model, which we can fit by the usual `lm()` function and related methods. However, we need to exploit some of the facilities in our new 'lmDiallel' extension package, which consist of the `GCA()`, `MDD()`, `DD()`, `SCA()`, `RGCA()` and `RSCA()` functions (see the box below). The resulting `lm` object can be explored by the usual R methods, such as `summary()` and `anova()` (the output of the `summary()` method is partly hidden, for brevity)

```
contrasts(hayman54$Block) <- "contr.sum"
dMod <- lm(Ftime ~ Block + GCA(Par1, Par2) + MDD(Par1, Par2) +
             DD(Par1, Par2) + SCA(Par1, Par2) +
             RGCA(Par1, Par2) + RSCA(Par1, Par2), data = hayman54)
summary(dMod)$coef[1:6,]
##                       Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)         162.898437   1.804567 90.2700843 2.381071e-68
## Block1               -1.054688   1.804567 -0.5844545 5.610017e-01
## GCA(Par1, Par2)g_A   46.195312   3.376036 13.6832990 1.558468e-20
## GCA(Par1, Par2)g_B -24.585938   3.376036 -7.2824864 6.421946e-10
## GCA(Par1, Par2)g_C   49.632812   3.376036 14.7015049 4.900927e-22
## GCA(Par1, Par2)g_D   18.351563   3.376036  5.4358311 9.415231e-07
# ...
# ...
anova(dMod)
## Analysis of Variance Table
##
## Response: Ftime
##                   Df Sum Sq Mean Sq F value    Pr(>F)
## Block              1    142     142  0.3416   0.56100
## GCA(Par1, Par2)    7 277717   39674 95.1805 < 2.2e-16 ***
## MDD(Par1, Par2)    1  30797   30797 73.8840 3.259e-12 ***
## DD(Par1, Par2)     7  34153    4879 11.7050 1.957e-09 ***
## SCA(Par1, Par2)   20  37289    1864  4.4729 2.560e-06 ***
## RGCA(Par1, Par2)   7   6739     963  2.3097   0.03671 *
## RSCA(Par1, Par2)  21  12373     589  1.4135   0.14668
## Residuals         63  26260     417
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

For the sake of simplicity, we also built a wrapper function named `lm.diallel()`, which can be used in the very same fashion as `lm()`. The syntax is:

```
lm.diallel(formula, Block, Env, data, fct)
```

where 'formula' specifies the response variable and the two variables for parentals (e.g., Yield ~ Par1 + Par2) and the two arguments 'Block' and 'Env' are used to specify optional variables, coding for blocks and environments, respectively. The argument 'data' is a 'dataframe' where to look for the explanatory variables and, finally, 'fct' is a string variable coding for the selected model ("HAYMAN2", for this example; see below).

```
dMod2 <- lm.diallel(Ftime ~ Par1 + Par2, Block = Block,
                    data = hayman54, fct = "HAYMAN2")
# 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
## MDD        1  30797   30797 73.8840 3.259e-12 ***
## GCA        7 277717   39674 95.1805 < 2.2e-16 ***
## DD         7  34153    4879 11.7050 1.957e-09 ***
## SCA       20  37289    1864  4.4729 2.560e-06 ***
## RGCA       7   6739     963  2.3097   0.03671 *
## RSCA      21  12373     589  1.4135   0.14668
## Residuals 63  26260
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The above function works very much like the `lm()` function and makes use of the general purpose linear model solver `lm.fit()`. Apart from simplicity, another advantage is that the call to `lm.diallel()` returns an object of both 'lm' and 'diallel' classes. For this latter class, we built several specific S3 methods, such as the usual `anova()`, `summary()` and `model.matrix()` methods, partly shown in the box above.

Considering that diallel models are usually fitted to determine genetical parameters, we also built the `glht.diallelMod()` method and the `diallel.eff()` function, which can be used with the 'multcomp' package, to retrieve the complete list of genetical parameters. An excerpt is shown in the box 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 == 0 162.8984     1.8046  90.270  < 2e-16 ***
# m == 0          -5.8627     0.6821  -8.596 4.48e-09 ***
# g_A == 0        46.1953     3.3760  13.683 2.17e-13 ***
# g_B == 0       -24.5859     3.3760  -7.282 9.83e-08 ***
# g_C == 0        49.6328     3.3760  14.702 4.13e-14 ***
```

```
# g_D == 0        18.3516     3.3760   5.436 1.07e-05 ***
# g_E == 0       -20.9297     3.3760  -6.199 1.47e-06 ***
# g_F == 0         2.4453     3.3760   0.724 0.475340
# g_G == 0       -44.7109     3.3760 -13.244 4.57e-13 ***
# g_H == 0       -26.3984     3.3760  -7.819 2.71e-08 ***
# d_A == 0         1.2213     1.9492   0.627 0.536380
# d_B == 0        -2.6224     1.9492  -1.345 0.190113
# ...
# ...
```

In a previous post (see here) we have shown that, when diallel models are fitted to the genotype means (and thus we have no replicates), an appropriate estimate of residual mean square and degrees of freedom can be passed as arguments to the `summary()`, `anova()` and `diallel.eff()` methods.

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.

```
# Random genetic effects
mod1m <- mmer.diallel(Ftime ~ Par1 + Par2, Block = Block,
                      data = hayman54,
                      fct = "HAYMAN2")
mod1m
##               VarComp   VarCompSE
## Block        0.00000    9.188298
## MDD       1783.96081 3118.893561
## GCA       1005.92052  574.893353
## RGCA        17.97898   19.920016
## DD         659.53567  468.205470
## SCA        351.74035  144.688653
## RSCA        32.02325   46.361581
## Residuals  412.54051   73.506382
```

That's all about the Hayman's models; you may have noted that both models (type 1 and 2) were devised for full diallel experiments, which are not so widespread in 'genetical' literature. A few years later, in 1956, the Australian scientist B. Griffing made the relevant effort of creating a comprehensive set of models which can be fitted to all types of diallel experiments. We will talk about these models in a future post.

Thanks for reading (and happy 2021!)