General and Specific Combining Abilities

Facundo Muñoz

2015-11-18 breedR version: 0.10.17

Contents

In a factorial design (either complete or not) the *General Combining Ability* (GCA) of a parent is its **Breeding Value**, while the *Specific Combinig Ability* (SCA) of a mating is the additional genetic value due to the **interaction** between those particular genotypes.

We propose **two alternative ways** of getting BLUPs for the GCAs. While for the SCAs we will simply use an unstructured random effect with one level for each observed mating.

We illustrate the methods with the following simulated data. Note that in this example the males and females are different individuals (and thus, they have different codes). However, monoic species can be set up as diallels, which simply means that some or all of the codes (and GCAs) will be shared. Otherwise, the methods still apply.

Note also that while the GCAs are sampled using the base population's additive-genetic variance, the intra-family breeding values are sampled with half-that variance. This is standard theory.

```
## Setup
library(breedR)
library(ggplot2)
set.seed(123)

## Simulation parameters
n.parents <- c(male = 15L,
               female = 10L)
n.matings <- 100
n.replicates <- 40
mu = 10        # Intercept
sigma2_g <- 6  # Genetic variance of the base population
sigma2_s <- 1  # Variance of the SCA
sigma2_e <- 1  # Residual variance
## Generate all crosses and sample a subset
parents.codes <- list(male = seq.int(n.parents['male']),
                      female = n.parents['male'] + seq.int(n.parents['female']))
matings <- expand.grid(parents.codes)
matings <- matings[sample(prod(n.parents), n.matings),]
rownames(matings) <- with(matings, paste(male, female, sep = 'x'))

## Simulated values
GCA = sapply(do.call('c', parents.codes),
             function(x) rnorm(1, mean = 0, sd = sqrt(sigma2_g)))
SCA = sapply(rownames(matings),
             function(x) rnorm(1, mean = 0, sd = sqrt(sigma2_s)))

## Expected phenotype per family
eta.family <- mu + SCA + (GCA[matings$male] + GCA[matings$female])/2

## Realised Breeding Values in the progeny
```

```
## (intra-family variance = half genetic variance)
n.progeny <- n.replicates*n.matings
eta.realised <- eta.family + rnorm(n.progeny, sd = sqrt(sigma2_g/2))

dat <- data.frame(Id = max(sapply(parents.codes, max)) + seq.int(n.progeny),
                  rep = rep(seq.int(n.replicates), each = n.matings),
                  matings,
                  eta.realised,
                  y = eta.realised + rnorm(n.progeny, sd = sqrt(sigma2_e)))

## Define variable for the non-additive SCA
dat <- transform(dat,
                 SCA   = factor(paste(male, female, sep = 'x'),
                                levels = rownames(matings)))

## Printing simulated setting
print(table(dat[, c('male', 'female')]), zero.print = "")
```

```
##     female
## male 16 17 18 19 20 21 22 23 24 25
##   1  40 40    40 40       40 40
##   2     40    40    40 40    40 40
##   3  40 40 40 40       40 40 40 40
##   4        40    40 40 40    40 40
##   5  40 40 40 40 40 40 40
##   6  40    40    40       40 40
##   7  40    40       40 40 40 40 40
##   8  40    40 40 40 40    40 40 40
##   9  40 40 40 40 40       40
##   10    40 40 40 40 40 40 40 40 40
##   11 40 40    40 40 40    40 40 40
##   12 40 40    40    40       40 40
##   13    40 40             40    40
##   14 40 40 40    40 40 40 40 40
##   15 40    40 40    40       40 40
```

```
str(dat)
```

```
## 'data.frame':    4000 obs. of  7 variables:
##  $ Id          : int  26 27 28 29 30 31 32 33 34 35 ...
##  $ rep         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ male        : int  14 13 1 10 3 7 2 8 4 5 ...
##  $ female      : int  18 23 20 24 25 16 21 24 21 20 ...
##  $ eta.realised: num  7.94 10.6 12.14 10.33 8.17 ...
##  $ y           : num  8.73 11.6 12.52 11.44 7.2 ...
##  $ SCA         : Factor w/ 100 levels "14x18","13x23",..: 1 2 3 4 5 6 7 8 9 10 ...
```

Method 1: using unstructured random effects

The first method uses two independent unstructured random effects for the GCAs of the mother and the father trees respectively.

Note that `remlf90` will estimate two independent variances for these effects, while in reality they are the same. However, we currently do not have a way to specify that in **breedR**. It will be possible soon, when we implement the generic model. Therefore, **this approach is currently sub-efficient**.

Furthermore, the `female` and `male` effects represent actually **half** of the Breeding Value contributed by both parents. So their variance is a **quarter** of the base population's additive-genetic variance. We will then use four times the mean of both estimates as an estimate of the additive-genetic variance.

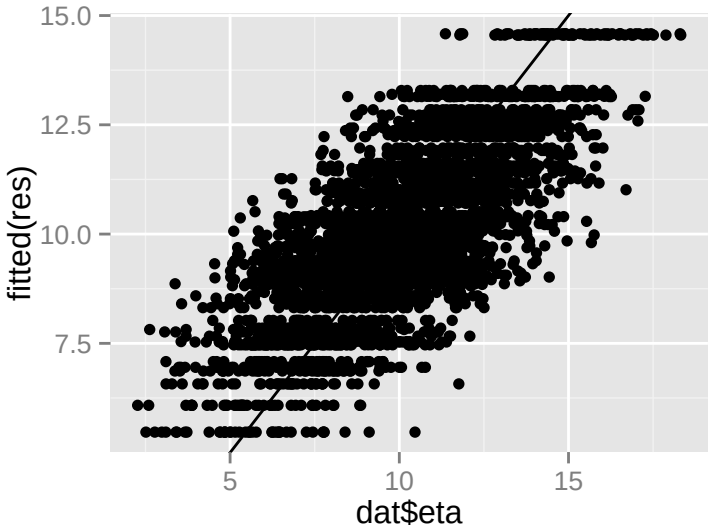
```
## Note that I would like to estimate only **one** GCA effect
## However, currently I need to specify two independent random effects with
## two independent variances, which account in reality for the same thing
res <- remlf90(y ~ 1,
               random = ~ male + female + SCA,
               dat = transform(dat,
                               male   = factor(male),
                               female = factor(female)))
```

```
## No specification of initial variances.
##      Using default value of 1 for all variance components.
##      See ?breedR.getOption.
```

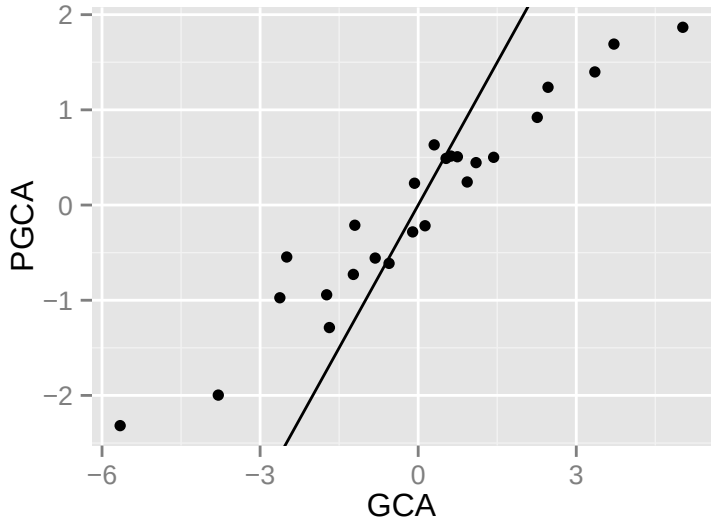
```
## Here, the effects 'female' and 'male' are both estimating GCA/2
## therefore, their variances are Var(GCA)/4 = sigma_g/4
## So, a point estimator for sigma_g would be:
(sigma_g.est <- 4 * mean(res$var[c('female', 'male'), 1]))
```

```
## [1] 5.5054
```

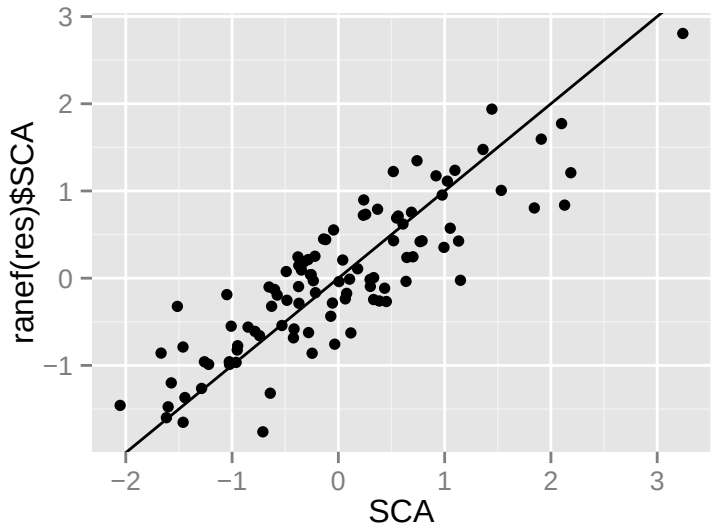
```
## while the BLUPs
PGCA <- c(ranef(res)$male, ranef(res)$female)

## Check fit
qplot(dat$eta, fitted(res)) + geom_abline(int=0, sl=1)
```



```
qplot(GCA, PGCA) + geom_abline(int=0, sl=1)
```




```
qplot(SCA, ranef(res)$SCA) + geom_abline(int=0, sl=1)
```




```
summary(res)
```

```
## Linear Mixed Model with pedigree and spatial effects fit by AI-REMLF90 ver. 1.110
##    Data: transform(dat, male = factor(male), female = factor(female))
##    AIC   BIC logLik
##  17152 17177  -8572
##
## Parameters of special components:
##
##
## Variance components:
##          Estimated variances    S.E.
## male                  1.0514 0.46682
## female                1.7013 0.86324
```

```
## SCA                    0.9908 0.17669
## Residual               3.9535 0.08952
##
## Fixed effects:
##           value   s.e.
## Intercept 10.05 0.5018
```

Method 2: using the implicit pedigree

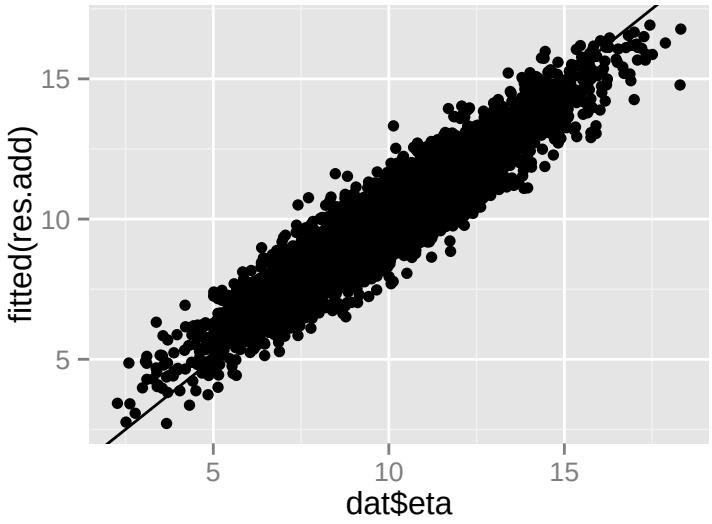
With this approach we estimate **directly** the genetic variance of the base population, and predict the Breeding Values of all individuals, including the parents (i.e. the GCAs).

The SCAs are again fitted as an unstructured random effect.

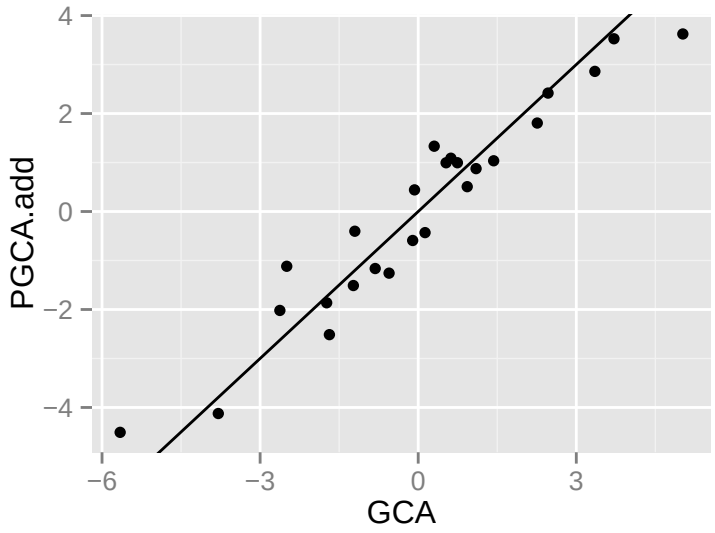
```
res.add <- remlf90(y ~ 1,
                   random  = ~ SCA,
                   genetic = list(model    = 'add_animal',
                                  pedigree = dat[, c('Id', 'male', 'female')],
                                  id       = 'Id'),
                   dat = dat)
```

```
## No specification of initial variances.
##      Using default value of 1 for all variance components.
##      See ?breedR.getOption.
```

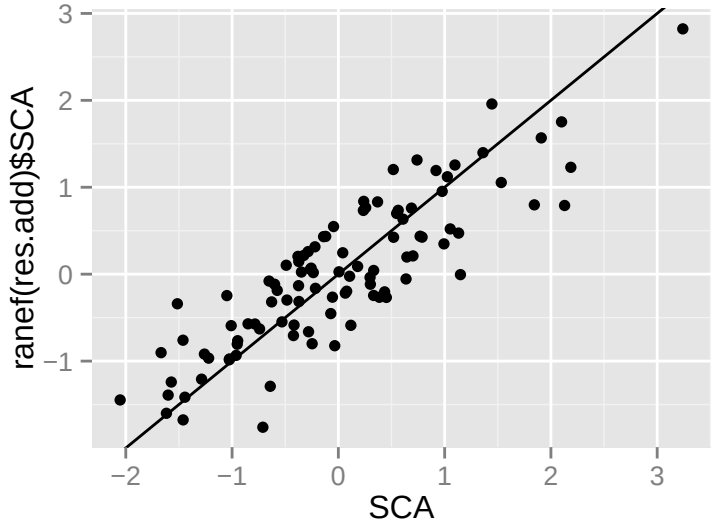
```
# Check fit
qplot(dat$eta, fitted(res.add)) + geom_abline(int=0, sl=1)
```




```
# Predicted GCAs for the parents
PGCA.add <- ranef(res.add)$genetic[do.call('c', parents.codes)]
qplot(GCA, PGCA.add) + geom_abline(int=0, sl=1)
```




```
# Predicted SCAs for the families
qplot(SCA, ranef(res.add)$SCA) + geom_abline(int=0, sl=1)
```




```
summary(res)
```

```
## Linear Mixed Model with pedigree and spatial effects fit by AI-REMLF90 ver. 1.110
##    Data: transform(dat, male = factor(male), female = factor(female))
##    AIC   BIC logLik
##  17152 17177  -8572
##
## Parameters of special components:
##
##
## Variance components:
##          Estimated variances    S.E.
## male                  1.0514 0.46682
## female                1.7013 0.86324
## SCA                   0.9908 0.17669
```

```
## Residual              3.9535 0.08952
##
## Fixed effects:
##           value   s.e.
## Intercept 10.05 0.5018
```

Final remarks

- You can derive point estimates of Heritabilities using the resulting variance estimates
- The GCA and SCA BLUPs can be extracted with the `ranef` expressions above
- Note that the log-likelihood of both models is exactly the same, while AIC penalizes slightly the first approach because it has one extra parameter.