

Quantitative genetics using the sommer package

Giovanny Covarrubias-Pazaran

2019-03-25

The sommer package was developed to provide R users a powerful and reliable multivariate mixed model solver for different genetic and non-genetic analysis in diploid and polyploid organisms. This package allows the user to estimate variance components for a mixed model with the advantage of specifying the variance-covariance structure of the random effects, specify heterogeneous variances, and obtain other parameters such as BLUPs, BLUEs, residuals, fitted values, variances for fixed and random effects, etc. The core algorithms of the package are coded in C++ using the Armadillo library to optimize dense matrix operations common in the direct-inversion algorithms.

The package is focused on problems of the type $p > n$ related to genomic prediction (hybrid prediction & genomic selection) and GWAS analysis, although any general mixed model can be fitted as well. The package provides kernels to estimate additive (**A.mat**), dominance (**D.mat**), and epistatic (**E.mat**) relationship matrices that have been shown to increase prediction accuracy under certain scenarios or simply to estimate the variance components of such. The package provides flexibility to fit other genetic models such as full and half diallel models as well.

Vignettes aim to provide several examples in how to use the sommer package under different scenarios. We will spend the rest of the space providing examples for:

- 1) Heritability (h^2) calculation
- 2) Specifying heterogeneous variances in mixed models
- 3) Using the pin calculator
- 4) Half and full diallel designs (using the overlay)
- 5) Genomic selection (predicting mendelian sampling)
 - GBLUP
 - rrBLUP
- 6) Single cross prediction (hybrid prediction)
- 7) Spatial modeling (using the 2-dimensional splines)
- 8) Multivariate genetic models and genetic correlations
- 9) Final remarks

Background

The core of the package the **mmer** function which solve the mixed model equations. The functions are an interface to call the NR Direct-Inversion Newton-Raphson or Average Information (Tunnicliffe 1989; Gilmour et al. 1995; Lee et al. 2016). Since version 2.0 sommer can handle multivariate models. Following Maier et al. (2015), the multivariate (and by extension the univariate) mixed model implemented has the form:

$$y_1 = X_1\beta_1 + Z_1u_1 + \epsilon_1 \quad y_2 = X_2\beta_2 + Z_2u_2 + \epsilon_2 \quad \dots \quad y_i = X_i\beta_i + Z_iu_i + \epsilon_i$$

where y_i is a vector of trait phenotypes, β_i is a vector of fixed effects, u_i is a vector of random effects for individuals and e_i are residuals for trait 'i' ($i = 1, \dots, t$). The random effects ($u_1 \dots u_i$ and e_i) are assumed to be normally distributed with mean zero. X and Z are incidence matrices for fixed and random effects respectively. The distribution of the multivariate response and the phenotypic variance covariance (V) are:

$$Y = X\beta + ZU + \epsilon_i$$

$$Y \sim \text{MVN}(X\beta, V)$$

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_t \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} X_1 & \dots & \dots \\ \vdots & \ddots & \vdots \\ \dots & \dots & X_t \end{bmatrix}$$

$$\mathbf{V} = \begin{bmatrix} Z_1 K \sigma_{g_1}^2 Z_1' + H \sigma_{\epsilon_1}^2 & \dots & Z_1 K \sigma_{g_{1,t}} Z_t' + H \sigma_{\epsilon_{1,t}}^2 \\ \vdots & \ddots & \vdots \\ Z_1 K \sigma_{g_{1,t}} Z_t' + H \sigma_{\epsilon_{1,t}}^2 & \dots & Z_t K \sigma_{g_t}^2 Z_t' + H \sigma_{\epsilon_t}^2 \end{bmatrix}$$

where K is the relationship or covariance matrix for the k th random effect ($u=1, \dots, k$), and $R=I$ is an identity matrix for the residual term. The terms $\sigma_{g_i}^2$ and $\sigma_{\epsilon_i}^2$ denote the genetic (or any of the k th random terms) and residual variance of trait 'i', respectively and $\sigma_{g_{ij}}$ and $\sigma_{\epsilon_{ij}}$ the genetic (or any of the k th random terms) and residual covariance between traits 'i' and 'j' ($i=1, \dots, t$, and $j=1, \dots, t$). The algorithm implemented optimizes the log likelihood:

$$\log L = 1/2 * \ln(|V|) + \ln(X'V|X) + Y'PY$$

where $||$ is the determinant of a matrix. And the REML estimates are updated using a Newton optimization algorithm of the form:

$$\theta^{k+1} = \theta^k + (H^k)^{-1} * \frac{dL}{d\sigma_i^2} | \theta^k$$

Where, θ is the vector of variance components for random effects and covariance components among traits, H^{-1} is the inverse of the Hessian matrix of second derivatives for the k th cycle, $\frac{dL}{d\sigma_i^2}$ is the vector of first derivatives of the likelihood with respect to the variance-covariance components. The Eigen decomposition of the relationship matrix proposed by Lee and Van Der Werf (2016) was included in the Newton-Raphson algorithm to improve time efficiency. Additionally, the popular pin function to estimate standard errors for linear combinations of variance components (i.e. heritabilities and genetic correlations) was added to the package as well.

Please refer to the canonical papers listed in the Literature section to check how the algorithms work. We have tested widely the methods to make sure they provide the same solution when the likelihood behaves well but for complex problems they might lead to slightly different answers. If you have any concern please contact me at cova_ruber@live.com.mx.

In the following section we will go in detail over several examples on how to use mixed models in univariate and multivariate case and their use in quantitative genetics.

1) Marker and non-marker based heritability calculation

The heritability is one of the most popular parameters among the breeding and genetics community because of the insight that provides in the inheritance of the trait. The heritability is usually estimated as narrow sense (h^2 ; only additive variance in the numerator σ_A^2), and broad sense (H^2 ; all genetic variance in the numerator σ_G^2).

In a classical breeding experiment with no molecular markers, special designs are performed to estimate and dissect the additive (σ_A^2) and non-additive (i.e. dominance σ_D^2) variance along with environmental variability. Designs such as generation analysis, North Carolina designs are used to dissect σ_A^2 and σ_D^2 to estimate the narrow sense heritability (h^2). When no special design is available we can still dissect the genetic variance (σ_G^2) and estimate the broad sense heritability. In this first example we will show the broad sense estimation

which doesn't use covariance structures for the genotypic effect (i.e. genomic or additive relationship matrices). For big models with no covariance structures, sommer's direct inversion is a bad idea to use but we will show anyways how to do it, but keep in mind that for very sparse models we recommend using the lmer function from the lme4 package or any other package using MME-based algorithms (i.e. asreml-R).

The following dataset has 41 potato lines evaluated in 5 locations across 3 years in an RCBD design. We show how to fit the model and extract the variance components to calculate the h^2 .

```
library(sommer)
data(DT_example)
head(DT)
```

```
##           Name      Env Loc Year      Block Yield      Weight
## 33  Manistee(MSL292-A) CA.2013  CA 2013  CA.2013.1      4 -1.904711
## 65           CD02024-9W CA.2013  CA 2013  CA.2013.1      5 -1.446958
## 66  Manistee(MSL292-A) CA.2013  CA 2013  CA.2013.2      5 -1.516271
## 67           MSL007-B CA.2011  CA 2011  CA.2011.2      5 -1.435510
## 68           MSR169-8Y CA.2013  CA 2013  CA.2013.1      5 -1.469051
## 103          AC05153-1W CA.2013  CA 2013  CA.2013.1      6 -1.307167
```

```
ans1 <- mmer(Yield~1,
             random= ~ Name + Env + Env:Name + Env:Block,
             rcov= ~ units,
             data=DT)
```

```
## iteration      LogLik      wall      cpu(sec)      restrained
##      1      -40.765    20:13:49          0          0
##      2      -30.2657    20:13:49          0          0
##      3      -25.8227    20:13:49          0          1
##      4      -24.7277    20:13:49          0          1
##      5      -24.7203    20:13:49          0          1
##      6      -24.7202    20:13:49          0          1
```

```
summary(ans1)$varcomp
```

```
##           VarComp  VarCompSE      Zratio Constraint
## Name.Yield-Yield      3.718279  1.6959834  2.1924029   Positive
## Env.Yield-Yield      12.008450 12.2771178  0.9781164   Positive
## Env:Name.Yield-Yield      5.152643  1.4923912  3.4526091   Positive
## Env:Block.Yield-Yield      0.000000  0.1156675  0.0000000   Positive
## units.Yield-Yield      4.366189  0.6573086  6.6425245   Positive
```

```
(n.env <- length(levels(DT$Env)))
```

```
## [1] 3
```

```
pin(ans1, h2 ~ V1 / ( V1 + (V3/n.env) + (V5/(2*n.env)) ) )
```

```
##      Estimate      SE
## h2 0.6032715 0.1344582
```

Recently with markers becoming cheaper, thousand of markers can be run in the breeding materials. When markers are available, an special design is not necessary to dissect the additive genetic variance. The availability of the additive, dominance and epistatic relationship matrices allow us to estimate σ_A^2 , σ_D^2 and σ_I^2 , although given that A, D and E are not orthogonal the interpretation of models that fit more than A and D become cumbersome.

Assume you have a population (even unreplicated) in the field but in addition we have genetic markers. Now we can fit the model and estimate the genomic heritability that explains a portion of the additive genetic

variance (with high marker density $\sigma_A^2 = \sigma_g^2$)

```
data("DT_cpdata")
DT$idd <-DT$id; DT$ide <-DT$id
### look at the data
A <- A.mat(GT) # additive relationship matrix
D <- D.mat(GT) # dominance relationship matrix
E <- E.mat(GT) # epistatic relationship matrix
ans.ADE <- mmer(color~1,
  random=~vs(id,Gu=A) + vs(idd,Gu=D),
  rcov=~units,
  data=DT)

## iteration      LogLik      wall      cpu(sec)      restrained
##      1         -123    20:13:52          1           0
##      2        -107.864    20:13:52          1           0
##      3        -103.867    20:13:52          1           0
##      4        -103.315    20:13:53          2           0
##      5        -103.294    20:13:53          2           0
##      6        -103.293    20:13:53          2           0
```

```
(summary(ans.ADE)$varcomp)

##              VarComp      VarCompSE      Zratio Constraint
## u:id.color-color  0.003662202  0.0012194130  3.003250   Positive
## u:idd.color-color  0.001820079  0.0007406216  2.457502   Positive
## units.color-color  0.002106929  0.0002864724  7.354736   Positive
```

```
pin(ans.ADE, h2 ~ (V1) / ( V1+V3) )
```

```
##      Estimate      SE
## h2 0.6347926 0.08840488
```

```
pin(ans.ADE, h2 ~ (V1+V2) / ( V1+V2+V3) )
```

```
##      Estimate      SE
## h2 0.7223783 0.05563774
```

In the previous example we showed how to estimate the additive (σ_A^2), dominance (σ_D^2), and epistatic (σ_I^2) variance components based on markers and estimate broad (H^2) and narrow sense heritability (h^2). Notice that we used the `vs()` function which indicates that the random effect inside the parenthesis (i.e. id, idd or ide) has a covariance matrix (A, D, or E), that will be specified in the `Gu` argument of the `vs()` function. Please DO NOT provide the inverse but the original covariance matrix.

2) Specifying heterogeneous variances in univariate models

Very often in multi-environment trials, the assumption that genetic variance is the same across locations may be too naive. Because of that, specifying a general genetic component and a location specific genetic variance is the way to go.

We estimate variance components for GCA_2 and SCA specifying the variance structure.

```
data("DT_cornhybrids")
### fit the model
modFD <- mmer(Yield~1,
  random=~ vs(at(Location,c("3","4")),GCA2),
  rcov=~ vs(ds(Location),units),
  data=DT)
```

```
## iteration      LogLik      wall      cpu(sec)      restrained
##      1      -190.104    20:13:54         0         0
##      2      -171.543    20:13:55         1         0
##      3      -165.319    20:13:55         1         0
##      4      -164.691    20:13:56         2         0
##      5      -164.684    20:13:56         2         0
##      6      -164.684    20:13:57         3         0
```

```
summary(modFD)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.8 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value -164.6839 331.3677 335.3592      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## 3:GCA2.Yield-Yield    62.48    53.45  1.169   Positive
## 4:GCA2.Yield-Yield    97.99    79.56  1.232   Positive
## 1:units.Yield-Yield   216.82    30.77  7.047   Positive
## 2:units.Yield-Yield   216.82    30.77  7.047   Positive
## 3:units.Yield-Yield   493.05    77.27  6.381   Positive
## 4:units.Yield-Yield   711.98   111.63  6.378   Positive
## =====
## Fixed effects:
##      Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)    138.1    0.9442   146.3
## =====
## Groups and observations:
##           Yield
## 3:GCA2      20
## 4:GCA2      20
## =====
## Use the '$' sign to access results and parameters
```

In the previous example we showed how the `at()` function is used in the `mmer` solver. By using the `at` function you can specify that i.e. the GCA2 has a different variance in different Locations, in this case locations 3 and 4, but also a main GCA variance. This is considered a CS + DIAG (compound symmetry + diagonal) model.

In addition, other functions can be added on top to fit models with covariance structures, i.e. the `Gu` argument from the `vs()` function to indicate a covariance matrix (A, pedigree or genomic relationship matrix)

```
data("DT_cornhybrids")
GT[1:4,1:4]
```

```
##           A258      A634      A641      A680
## A258  2.23285528 -0.3504778 -0.04756856 -0.32239362
## A634 -0.35047780  1.4529169  0.45203869 -0.02293680
## A641 -0.04756856  0.4520387  1.96940221 -0.09896791
## A680 -0.32239362 -0.0229368 -0.09896791  1.65221984
```

```
### fit the model
modFD <- mmer(Yield~1,
```

```

random=~ vs(at(Location,c("3","4")),GCA2,Gu=GT),
rcov= ~ vs(ds(Location),units),
data=DT)

```

```

## iteration    LogLik      wall    cpu(sec)   restrained
##      1      -191.286   20:13:57         0         0
##      2      -172.247   20:13:58         1         0
##      3      -165.948   20:13:58         1         0
##      4      -165.248   20:13:59         2         0
##      5      -165.23    20:13:59         2         0
##      6      -165.229   20:13:59         2         0
##      7      -165.229   20:14:0         3         0

```

```
summary(modFD)
```

```

## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.8 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value -165.2286 332.4571 336.4486      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## 3:GCA2.Yield-Yield    26.64    26.16 1.0185   Positive
## 4:GCA2.Yield-Yield    37.51    37.78 0.9927   Positive
## 1:units.Yield-Yield   216.77    30.75 7.0489   Positive
## 2:units.Yield-Yield   216.77    30.75 7.0489   Positive
## 3:units.Yield-Yield   503.62    77.87 6.4673   Positive
## 4:units.Yield-Yield   738.86   114.17 6.4715   Positive
## =====
## Fixed effects:
##   Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)    138.1    0.9147    151
## =====
## Groups and observations:
##           Yield
## 3:GCA2      20
## 4:GCA2      20
## =====
## Use the '$' sign to access results and parameters

```

3) Using the pin calculator

Sometimes the user needs to calculate ratios or functions of specific variance-covariance components and obtain the standard error for such parameters. Examples of these are the genetic correlations, heritabilities, etc. Using the CPdata we will show how to estimate the heritability and the standard error using the pin function that uses the delta method to come up with these parameters. This can be extended for any linear combination of the variance components.

```

data("DT_cpdata")
### look at the data
A <- A.mat(GT) # additive relationship matrix
ans <- mmer(color~1,

```

```

random=~vs(id,Gu=A),
rcov=~units,
data=DT)

```

```

## iteration    LogLik      wall    cpu(sec)   restrained
##      1      -137.304   20:14:1         0         0
##      2      -115.507   20:14:1         0         0
##      3      -111.236   20:14:1         0         0
##      4      -110.755   20:14:2         1         0
##      5      -110.741   20:14:2         1         0
##      6      -110.741   20:14:2         1         0

```

```

(summary(ans.ADE)$varcomp)

```

```

##              VarComp   VarCompSE   Zratio Constraint
## u:id.color-color 0.003662202 0.0012194130 3.003250   Positive
## u:idd.color-color 0.001820079 0.0007406216 2.457502   Positive
## units.color-color 0.002106929 0.0002864724 7.354736   Positive

```

```

pin(ans, h2 ~ (V1) / ( V1+V2) )

```

```

##      Estimate      SE
## h2 0.6512157 0.06107574

```

The same can be used for multivariate models. Please check the documentation of the `pin` function to see more examples.

4) Half and full diallel designs (use of the overlay)

When breeders are looking for the best single cross combinations, diallel designs have been by far the most used design in crops like maize. There are 4 types of diallel designs depending if reciprocal and self cross (omission of parents) are performed (full diallel with parents n^2 ; full diallel without parents $n(n-1)$; half diallel with parents $1/2 * n(n+1)$; half diallel without parents $1/2 * n(n-1)$). In this example we will show a full diallel design (reciprocal crosses are performed) and half diallel designs (only one of the directions is performed).

In the first data set we show a full diallel among 40 lines from 2 heterotic groups, 20 in each. Therefore 400 possible hybrids are possible. We have phenotypic data for 100 of them across 4 locations. We use the data available to fit a model of the form:

$$y = X\beta + Zu_1 + Zu_2 + Zu_S + \epsilon$$

We estimate variance components for GCA_1 , GCA_2 and SCA and use them to estimate heritability. Additionally BLUPs for GCA and SCA effects can be used to predict crosses.

```

data("DT_cornhybrids")

modFD <- mmer(Yield~Location,
              random=~GCA1+GCA2+SCA,
              rcov=~units,
              data=DT)

```

```

## iteration    LogLik      wall    cpu(sec)   restrained
##      1      -149.436   20:14:3         1         0
##      2      -136.475   20:14:3         1         1
##      3      -132.852   20:14:4         2         1
##      4      -132.625   20:14:4         2         1

```

```
##      5      -132.596   20:14:4      2      1
##      6      -132.59   20:14:5      3      1
##      7      -132.589   20:14:5      3      1
##      8      -132.589   20:14:5      3      1
```

```
(suma <- summary(modFD)$varcomp)
```

```
##              VarComp VarCompSE      Zratio Constraint
## GCA1.Yield-Yield    0.000000  16.50337  0.0000000    Positive
## GCA2.Yield-Yield    7.412226  18.94200  0.3913116    Positive
## SCA.Yield-Yield   187.560303  41.59428  4.5092817    Positive
## units.Yield-Yield 221.142463  18.14716 12.1860656    Positive
```

```
Vgca <- sum(suma[1:2,1])
Vsca <- suma[3,1]
Ve <- suma[4,1]
Va = 4*Vgca
Vd = 4*Vsca
Vg <- Va + Vd
(H2 <- Vg / (Vg + (Ve)) )
```

```
## [1] 0.7790856
```

```
(h2 <- Va / (Vg + (Ve)) )
```

```
## [1] 0.02961832
```

Don't worry too much about the small h2 value, the data was simulated to be mainly dominance variance, therefore the Va was simulated extremely small leading to such value of narrow sense h2.

In this second data set we show a small half diallel with 7 parents crossed in one direction. $n(n-1)/2$ crosses are possible $7(6)/2 = 21$ unique crosses. Parents appear as males or females indistinctly. Each with two replications in a CRD. For a half diallel design a single GCA variance component for both males and females can be estimated and an SCA as well ($\sigma_G^2 CA$ and $\sigma_S^2 CA$ respectively), and BLUPs for GCA and SCA of the parents can be extracted. We would show first how to use it with the `mmer` function using the `overlay()` function. The specific model here is:

$$y = X\beta + Zu_g + Zu_s + \epsilon$$

```
data("DT_halfdiallel")
head(DT)
```

```
##   rep geno male female    sugar
## 1   1   12    1      2 13.950509
## 2   2   12    1      2  9.756918
## 3   1   13    1      3 13.906355
## 4   2   13    1      3  9.119455
## 5   1   14    1      4  5.174483
## 6   2   14    1      4  8.452221
```

```
DT$femalef <- as.factor(DT$female)
DT$malef <- as.factor(DT$male)
DT$genof <- as.factor(DT$geno)
#### model using overlay
modh <- mmer(sugar~1,
             random=~vs(overlay(femalef,malef))
             + genof,
             data=DT)
```



```
## iteration    LogLik      wall    cpu(sec)    restrained
##      1      -10.425    20:14:6         0          0
##      2       -6.487    20:14:6         0          0
##      3       -5.732    20:14:6         0          0
##      4      -5.67494    20:14:6         0          0
##      5      -5.67441    20:14:6         0          0
```

```
summary(modh)$varcomp
```

```
##                               VarComp VarCompSE   Zratio Constraint
## u:femalef.sugar-sugar  5.507899 3.5741151 1.541052   Positive
## genof.sugar-sugar     1.815784 1.3629575 1.332238   Positive
## units.sugar-sugar      3.117538 0.9626094 3.238632   Positive
```

Notice how the `overlay()` argument makes the overlap of incidence matrices possible making sure that male and female are joint into a single random effect.

5) Genomic selection

In this section we will use wheat data from CIMMYT to show how is genomic selection performed. This is the case of prediction of specific individuals within a population. It basically uses a similar model of the form:

$$y = X\beta + Zu + \epsilon$$

and takes advantage of the variance covariance matrix for the genotype effect known as the additive relationship matrix (A) and calculated using the `A.mat` function to establish connections among all individuals and predict the BLUPs for individuals that were not measured. The prediction accuracy depends on several factors such as the heritability (h^2), training population used (TP), size of TP, etc.

```
data("DT_wheat");
colnames(DT) <- paste0("X",1:ncol(DT))
DT <- as.data.frame(DT);DT$id <- as.factor(rownames(DT))
# select environment 1
rownames(GT) <- rownames(DT)
K <- A.mat(GT) # additive relationship matrix
colnames(K) <- rownames(K) <- rownames(DT)
# GBLUP pedigree-based approach
set.seed(12345)
y.trn <- DT
vv <- sample(rownames(DT),round(nrow(DT)/5))
y.trn[vv,"X1"] <- NA
head(y.trn)
```

```
##           X1           X2           X3           X4    id
## 775         NA -1.72746986 -1.89028479  0.0509159  775
## 2166 -0.2527028  0.40952243  0.30938553 -1.7387588 2166
## 2167  0.3418151 -0.64862633 -0.79955921 -1.0535691 2167
## 2465         NA  0.09394919  0.57046773  0.5517574 2465
## 3881         NA -0.28248062  1.61868192 -0.1142848 3881
## 3889  2.3360969  0.62647587  0.07353311  0.7195856 3889
```

```
## GBLUP
ans <- mmer(X1~1,
            random=~vs(id,Gu=K),
            rcov=~units,
            data=y.trn) # kinship based
```

```
## iteration      LogLik      wall      cpu(sec)      restrained
##      1      -202.344    20:14:7         0         0
##      2      -198.717    20:14:8         1         0
##      3      -197.634    20:14:8         1         0
##      4      -197.51    20:14:9         2         0
##      5      -197.508    20:14:9         2         0
##      6      -197.508    20:14:10        3         0
```

```
ans$U$`u:id`$X1 <- as.data.frame(ans$U$`u:id`$X1)
rownames(ans$U$`u:id`$X1) <- gsub("id","",rownames(ans$U$`u:id`$X1))
cor(ans$U$`u:id`$X1[vv,],DT[vv,"X1"], use="complete")
```

```
## [1] 0.4885674
```

```
## rrBLUP
ans2 <- mmer(X1~1,
             random=~vs(list(GT)),
             rcov=~units,
             data=y.trn) # kinship based
```

```
## iteration      LogLik      wall      cpu(sec)      restrained
##      1      -343.082    20:14:12         2         0
##      2      -243.965    20:14:12         2         0
##      3      -208.257    20:14:13         3         0
##      4      -197.982    20:14:13         3         0
##      5      -197.519    20:14:14         4         0
##      6      -197.508    20:14:14         4         0
##      7      -197.508    20:14:15         5         0
```

```
u <- GT %*% as.matrix(ans2$U$`u:GT`$X1) # BLUPs for individuals
rownames(u) <- rownames(GT)
cor(u[vv,],DT[vv,"X1"]) # same correlation
```

```
## [1] 0.4885716
```

```
# the same can be applied in multi-response models in GBLUP or rrBLUP
```

6) Single cross prediction

When doing prediction of single cross performance the phenotype can be dissected in three main components, the general combining abilities (GCA) and specific combining abilities (SCA). This can be expressed with the same model analyzed in the diallel experiment mentioned before:

$$y = X\beta + Zu_1 + Zu_2 + Zu_s + \epsilon$$

with:

$$u_1 \sim N(0, K_1\sigma_u^2 1)$$

$$u_2 \sim N(0, K_2\sigma_u^2 2)$$

$$u_s \sim N(0, K_3\sigma_u^2 s)$$

And we can specify the K matrices. The main difference between this model and the full and half diallel designs is the fact that this model will include variance covariance structures in each of the three random effects (GCA1, GCA2 and SCA) to be able to predict the crosses that have not occurred yet. We will use the data published by Technow et al. (2015) to show how to do prediction of single crosses.

```

data("DT_technow")
# RUN THE PREDICTION MODEL
y.trn <- DT
vv1 <- which(!is.na(DT$GY))
vv2 <- sample(vv1, 100)
y.trn[vv2,"GY"] <- NA
anss2 <- mmer(GY~1,
              random=~vs(dent,Gu=Ad) + vs(flint,Gu=Af),
              rcov=~units,
              data=y.trn)

## iteration      LogLik      wall      cpu(sec)    restrained
##      1      93.142    20:14:25         7           0
##      2     135.18    20:14:32        14           0
##      3     145.517    20:14:39        21           0
##      4     147.085    20:14:47        29           0
##      5     147.178    20:14:54        36           0
##      6     147.184    20:15:1         43           0
##      7     147.184    20:15:8         50           0

summary(anss2)$varcomp

##              VarComp VarCompSE      Zratio Constraint
## u:dent.GY-GY  16.93639  2.6917284   6.292012   Positive
## u:flint.GY-GY 12.47174  2.3248074   5.364634   Positive
## units.GY-GY   16.75020  0.7662471  21.860045   Positive

zu1 <- model.matrix(~dent-1,y.trn) %*% anss2$U$`u:dent`$GY
zu2 <- model.matrix(~flint-1,y.trn) %*% anss2$U$`u:flint`$GY
u <- zu1+zu2+anss2$Beta[1,"Estimate"]
cor(u[vv2,], DT$GY[vv2])

## [1] 0.8234584

```

In the previous model we only used the GCA effects (GCA1 and GCA2) for praticity, although it's been shown that the SCA effect doesn't actually help that much in increasing prediction accuracy and increase a lot the computation intensity required since the variance covariance matrix for SCA is the kronecker product of the variance covariance matrices for the GCA effects, resulting in a 10578x10578 matrix that increases in a very intensive manner the computation required.

A model without covariance structures would show that the SCA variance component is insignificant compared to the GCA effects. This is why including the third random effect doesn't increase the prediction accuracy.

7) Spatial modeling (using the 2-dimensional spline)

We will use the CPdata to show the use of 2-dimensional splines for accomodating spatial effects in field experiments. In early generation variety trials the availability of seed is low, which makes the use of unreplicated design a neccesity more than anything else. Experimental designs such as augmented designs and partially-replicated (p-rep) designs become every day more common this days.

In order to do a good job modeling the spatial trends happening in the field special covariance structures have been proposed to accomodate such spatial trends (i.e. autoregressive residuals; ar1). Unfortunately, some of these covariance structures make the modeling rather unstable. More recently other research groups have proposed the use of 2-dimensional splines to overcome such issues and have a more robust modeling of the spatial terms (Lee et al. 2013; Rodríguez-Álvarez et al. 2018).

In this example we assume an unreplicated population where row and range information is available which allows us to fit a 2 dimensional spline model.

```
data("DT_cpdata")
### mimic two fields
A <- A.mat(GT)
mix <- mmer(Yield~1,
            random=~vs(id, Gu=A) +
              vs(Rowf) +
              vs(Colf) +
              vs(spl2D(Row,Col)),
            rcov=~vs(units),
            data=DT)
```

## iteration	LogLik	wall	cpu(sec)	restrained
## 1	-154.198	20:15:11	1	0
## 2	-152.064	20:15:11	1	0
## 3	-151.265	20:15:11	1	0
## 4	-151.202	20:15:12	2	0
## 5	-151.201	20:15:12	2	0

```
summary(mix)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.8 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value -151.2011 304.4021 308.2938      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## u:id.Yield-Yield      783.4      319.3 2.4536 Positive
## u:Rowf.Yield-Yield      814.7      390.5 2.0863 Positive
## u:Colf.Yield-Yield      182.2      129.7 1.4053 Positive
## u:Row.Yield-Yield      513.6      694.7 0.7393 Positive
## u:units.Yield-Yield  2922.6      294.1 9.9368 Positive
## =====
## Fixed effects:
##   Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)      132.1      8.791  15.03
## =====
## Groups and observations:
##           Yield
## u:id      363
## u:Rowf     13
## u:Colf     36
## u:Row     168
## =====
## Use the '$' sign to access results and parameters
```

Notice that the job is done by the `spl2D()` function that takes the Row and Col information to fit a spatial kernel.

8) Multivariate genetic models and genetic correlations

Sometimes is important to estimate genetic variance-covariance among traits, multi-reponse models are very useful for such task. Let see an example with 3 traits (color, Yield, and Firmness) and a single random effect (genotype; id) although multiple effects can be modeled as well. We need to use a variance covariance structure for the random effect to be able to obtain the genetic covariance among traits.

```
data("DT_cpdata")
A <- A.mat(GT)
ans.m <- mmer(cbind(Yield,color)~1,
              random=~ vs(id, Gu=A, Gtc=unsm(2))
              + vs(Rowf,Gtc=diag(2))
              + vs(Colf,Gtc=diag(2)),
              rcov=~ vs(units, Gtc=unsm(2)),
              data=DT)
```

##	iteration	LogLik	wall	cpu(sec)	restrained
##	1	-375.872	20:15:17	4	0
##	2	-291.932	20:15:22	9	0
##	3	-258.273	20:15:26	13	0
##	4	-253.459	20:15:30	17	0
##	5	-253.291	20:15:34	21	0
##	6	-253.278	20:15:39	26	0
##	7	-253.277	20:15:43	30	0
##	8	-253.277	20:15:48	35	0

Now you can extract the BLUPs using the 'randef' function or simple accesing with the '\$' sign and pick 'u.hat'. Also, genetic correlations and heritabilities can be calculated easily.

```
cov2cor(ans.m$sigma$`u:id`)
```

```
##           Yield      color
## Yield 1.0000000 0.1234441
## color 0.1234441 1.0000000
```

9) Final remarks

Keep in mind that sommer uses direct inversion (DI) algorithm which can be very slow for large datasets. The package is focused in problems of the type $p > n$ (more random effect levels than observations) and models with dense covariance structures. For example, for experiment with dense covariance structures with low-replication (i.e. 2000 records from 1000 individuals replicated twice with a covariance structure of 1000x1000) sommer will be faster than MME-based software. Also for genomic problems with large number of random effect levels, i.e. 300 individuals (n) with 100,000 genetic markers (p). For highly replicated trials with small covariance structures or $n > p$ (i.e. 2000 records from 200 individuals replicated 10 times with covariance structure of 200x200) asreml or other MME-based algorithms will be much faster and we recommend you to opt for those software.

Literature

Covarrubias-Pazarán G. 2016. Genome assisted prediction of quantitative traits using the R package sommer. PLoS ONE 11(6):1-15.

Covarrubias-Pazarán G. 2018. Software update: Moving the R package sommer to multivariate mixed models for genome-assisted prediction. doi: <https://doi.org/10.1101/354639>

- Bernardo Rex. 2010. Breeding for quantitative traits in plants. Second edition. Stemma Press. 390 pp.
- Gilmour et al. 1995. Average Information REML: An efficient algorithm for variance parameter estimation in linear mixed models. *Biometrics* 51(4):1440-1450.
- Henderson C.R. 1975. Best Linear Unbiased Estimation and Prediction under a Selection Model. *Biometrics* vol. 31(2):423-447.
- Kang et al. 2008. Efficient control of population structure in model organism association mapping. *Genetics* 178:1709-1723.
- Lee, D.-J., Durban, M., and Eilers, P.H.C. (2013). Efficient two-dimensional smoothing with P-spline ANOVA mixed models and nested bases. *Computational Statistics and Data Analysis*, 61, 22 - 37.
- Lee et al. 2015. MTG2: An efficient algorithm for multivariate linear mixed model analysis based on genomic information. Cold Spring Harbor. doi: <http://dx.doi.org/10.1101/027201>.
- Maier et al. 2015. Joint analysis of psychiatric disorders increases accuracy of risk prediction for schizophrenia, bipolar disorder, and major depressive disorder. *Am J Hum Genet*; 96(2):283-294.
- Rodriguez-Alvarez, Maria Xose, et al. Correcting for spatial heterogeneity in plant breeding experiments with P-splines. *Spatial Statistics* 23 (2018): 52-71.
- Searle. 1993. Applying the EM algorithm to calculating ML and REML estimates of variance components. Paper invited for the 1993 American Statistical Association Meeting, San Francisco.
- Yu et al. 2006. A unified mixed-model method for association mapping that accounts for multiple levels of relatedness. *Genetics* 38:203-208.
- Abdollahi Arpanahi R, Morota G, Valente BD, Kranis A, Rosa GJM, Gianola D. 2015. Assessment of bagging GBLUP for whole genome prediction of broiler chicken traits. *Journal of Animal Breeding and Genetics* 132:218-228.
- Tunncliffe W. 1989. On the use of marginal likelihood in time series model estimation. *JRSS* 51(1):15-27.