

Quick start for the sommer package

Giovanny Covarrubias-Pazarán

2018-12-06

The sommer package was developed to provide R users a powerful and reliable multivariate mixed model solver. The package is focused in problems of the type $p > n$ (more effects to estimate than observations) and its core algorithm is coded in C++ using the Armadillo library. This package allows the user to fit mixed models with the advantage of specifying the variance-covariance structure for the random effects, and specify heterogeneous variances, and obtain other parameters such as BLUPs, BLUEs, residuals, fitted values, variances for fixed and random effects, etc.

The purpose of this quick start guide is to show the flexibility of the package under certain common scenarios:

B1) Background on mixed models

B2) Background on covariance structures

- 1) Univariate homogeneous variance models
- 2) Univariate heterogeneous variance models
- 3) Univariate unstructured variance models
- 4) Multivariate homogeneous variance models
- 5) Multivariate heterogeneous variance models
- 6) Multivariate unstructured variance models
- 7) Random regression models
- 8) GWAS models
- 9) Including special functions
 - the major `vs()` function for special variance models and its auxiliars:
 - `at()` specific levels structure
 - `ds()` diagonal structure
 - `us()` unstructured
 - `cs()` customized structure
 - `overlay()` overlaid models
 - `spl2D()` two dimensional spline models
- 10) The specification of constraints
- 11) Final remarks

B1) Background on mixed models

The core of the package is 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 algorithms (Tunnicliffe 1989; Gilmour et al. 1995; Lee et al. 2016). From 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:

$$y_1 = X_1\beta_1 + Z_1u_1 + \epsilon_1$$

$$y_2 = X_2\beta_2 + Z_2u_2 + \epsilon_2$$

...

$$y_i = X_i\beta_i + Z_iu_i + \epsilon_i$$

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:

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

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

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

$$\mathbf{X} = \begin{bmatrix} X_1 & 0 & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & X_t \end{bmatrix}$$

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

where K is the relationship or covariance matrix for the k th random effect ($u=1, \dots, k$), and $H=I$ is an identity matrix or a partial 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.

B2) Background on covariance structures

One of the major strenghts of linear mixed models is the flexibility to specify variance-covariance structures at all levels. In general, variance structures of mixed models can be seen as tensor (kronecker) products of multiple variance-covariance stuctures. For example, a multi-response model (i.e. 2 traits) where “ g ”

individuals (i.e. 100 genotypes) are tested in “e” treatments (i.e. 3 environments), the variance-covariance for the random effect “individuals” can be seen as the following multiplicative model:

$$\mathbf{T} \otimes \mathbf{G} \otimes \mathbf{A}$$

where:

$$\mathbf{T} = \begin{bmatrix} \sigma_{g_{t1,t1}}^2 & \sigma_{g_{t1,t2}} \\ \sigma_{g_{t2,t1}} & \sigma_{g_{t2,t2}}^2 \end{bmatrix}$$

is the covariance structure for individuals among traits.

$$\mathbf{G} = \begin{bmatrix} \sigma_{g_{e1,e1}}^2 & \sigma_{g_{e1,e2}} & \sigma_{g_{e1,e3}} \\ \sigma_{g_{e2,e1}} & \sigma_{g_{e2,e2}}^2 & \sigma_{g_{e2,e3}} \\ \sigma_{g_{e3,e1}} & \sigma_{g_{e3,e2}} & \sigma_{g_{e3,e3}}^2 \end{bmatrix}$$

is the covariance structure for individuals among environments.

and A is a square matrix representing the covariance among the levels of the individuals (any known relationship matrix).

The T and G covariance structures shown above are unknown matrices to be estimated whereas A is known. The T and G matrices shown above are called as unstructured (US) covariance matrices, although this type is just one example from several covariance structures that the linear mixed models enable. For example, other popular covariance structures are:

Diagonal (DIAG) covariance structures

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_{g_{e1,e1}}^2 & 0 & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & \sigma_{g_{ei,ei}}^2 \end{bmatrix}$$

Compound symmetry (CS) covariance structures

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_g^2 + \sigma_{ge}^2 & \sigma_g^2 & \sigma_g^2 & \sigma_g^2 & \sigma_g^2 \\ \sigma_g^2 & \sigma_g^2 + \sigma_{ge}^2 & \sigma_g^2 & \sigma_g^2 & \sigma_g^2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma_g^2 & \sigma_g^2 & \sigma_g^2 & \sigma_g^2 + \sigma_{ge}^2 & \sigma_g^2 \end{bmatrix}$$

First order autoregressive (AR1) covariance structures

$$\mathbf{\Sigma} = \sigma^2 \begin{bmatrix} 1 & \rho & \rho^2 & \rho^3 \\ \rho & 1 & \rho & \rho^2 \\ \rho^2 & \rho & 1 & \rho \\ \rho^3 & \rho^2 & \rho & 1 \end{bmatrix}$$

or the already mentioned Unstructured (US) covariance structures

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_{g_{e1,e1}}^2 & \sigma_{g_{e1,e2}} & \sigma_{g_{e1,e3}} \\ \vdots & \ddots & \vdots \\ \sigma_{g_{e3,e1}} & \sigma_{g_{e3,e2}} & \sigma_{g_{e3,e3}}^2 \end{bmatrix}$$

among others. Sommer has the capabilities to fit some of these covariance structures in the mixed model machinery.

forming variance structures in sommer using the vs() function

The sommer function `vs()` allows to construct very structured variance models that are passed to the `mmer()` function it's one of the most important functions in the sommer package. Its specification is:

```
random=~vs(..., Gu, Gt, Gtc)
```

The idea is that the `vs()` function reflects the special variance structure that each random effect could have:

$$T \otimes E \otimes \dots \otimes A$$

where the `...` argument in the `vs()` function is used to specify the kronecker products from all matrices that form the variance for the random effect, where the auxiliary function `ds()`, `us()`, `cs()`, `at()`, can be used to define such structure. The idea is that a variance model for a random effect `x` (i.e. individuals) might require a more flexible model than just:

```
random=~x
```

For example, if individuals are tested in different time-points and environment, we can assume a different variance and covariance components among the individuals in the different environment-timepoint combinations. An example of variance structure of the type:

$$T \otimes E \otimes S \otimes A$$

would be specified in the `vs()` function as:

```
random=~vs(us(e),us(s),x, Gu=A, Gtc=T)
```

where the `e` would be a column vector in a data frame for the environments, `s` a vector in the dataframe for the time points, `x` is the vector in the dataframe for the identifier of individuals, `A` is a known square variance covariance matrix among individuals, and `T` is a square matrices with as many rows and columns as the number of traits.

1) Univariate homogeneous variance models

This type of models refer to single response models where a variable of interest (i.e. genotypes) needs to be analyzed as interacting with a 2nd random effect (i.e. environments), but you assume that across environments the genotypes have the same variance component. This is the so-called compound symmetry (CS) model.

```
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           CO02024-9W 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~Env,
             random= ~ Name + Env:Name,
             rcov= ~ units,
             data=DT)
```

```
## iteration    LogLik      wall    cpu(sec)   restrained
##      1      -31.2668  22:12:29      0           0
##      2      -23.2804  22:12:30      1           0
##      3      -20.4746  22:12:30      1           0
##      4      -20.1501  22:12:30      1           0
##      5      -20.1454  22:12:30      1           0
##      6      -20.1454  22:12:30      1           0
```

```
summary(ans1)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value -20.14544 46.29075 55.95182      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## Name.Yield-Yield      3.682      1.691  2.177  Positive
## Env:Name.Yield-Yield  5.173      1.495  3.460  Positive
## units.Yield-Yield     4.366      0.647  6.748  Positive
## =====
## Fixed effects:
##   Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)  16.496      0.6855  24.065
## 2 Yield EnvCA.2012   -5.777      0.7558  -7.643
## 3 Yield EnvCA.2013   -6.380      0.7960  -8.015
## =====
## Groups and observations:
##           Yield
## Name      41
## Env:Name  123
## =====
## Use the '$' sign to access results and parameters
```

2) Univariate heterogeneous variance models

Very often in multi-environment trials, the assumption that the genetic variance or the residual 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. This requires a CS+DIAG model (also called heterogeneous CS model).

```
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           CO02024-9W 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
```

```
ans2 <- mmer(Yield~Env,
             random= ~Name + vs(ds(Env),Name),
             rcov= ~ vs(ds(Env),units),
             data=DT)
```

```
## iteration    LogLik      wall    cpu(sec)   restrained
##      1      -31.2668  22:12:30      0           0
##      2     -19.8549  22:12:30      0           0
##      3     -15.9797  22:12:30      0           0
##      4     -15.4374  22:12:30      0           0
##      5     -15.43   22:12:30      0           0
##      6     -15.4298  22:12:30      0           0
```

```
summary(ans2)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value -15.42995 36.85965 46.52072      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## Name.Yield-Yield      2.963      1.496  1.980  Positive
## CA.2011:Name.Yield-Yield 10.146      4.507  2.251  Positive
## CA.2012:Name.Yield-Yield  1.878      1.870  1.004  Positive
## CA.2013:Name.Yield-Yield  6.629      2.503  2.649  Positive
## CA.2011:units.Yield-Yield 4.942      1.525  3.242  Positive
## CA.2012:units.Yield-Yield 5.725      1.312  4.363  Positive
## CA.2013:units.Yield-Yield 2.560      0.640  4.000  Positive
## =====
## Fixed effects:
## Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept) 16.508      0.8268 19.965
## 2 Yield EnvCA.2012  -5.817      0.8575  -6.783
## 3 Yield EnvCA.2013  -6.412      0.9356  -6.854
## =====
## Groups and observations:
##           Yield
## Name      41
## CA.2011:Name 41
## CA.2012:Name 41
## CA.2013:Name 41
## =====
## Use the '$' sign to access results and parameters
```

As you can see the special function `at` or `diag` can be used to indicate that there's a different variance for the genotypes in each environment. Same was done for the residual. The difference between `at` and `diag` is that the `at` function can be used to specify the levels or specific environments where the variance is different.

3) Unstructured variance models

A more relaxed assumption than the CS+DIAG model is the unstructured model (US) which assumes that among the levels of certain factor (i.e. Environments) there's a covariance structure of a second random effect (i.e. Genotypes). This can be done in sommer using the `us(.)` function:

```
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           CO02024-9W 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
```

```
ans3 <- mmer(Yield~Env,
             random=~ vs(us(Env),Name),
             rcov=~vs(us(Env),units),
             data=DT)
```

```
## iteration  LogLik      wall  cpu(sec)  restrained
##      1      -37.9059  22:12:31      0          0
##      2      -17.9745  22:12:31      0          0
##      3      -12.2427  22:12:31      0          0
##      4      -11.5121  22:12:31      0          0
##      5      -11.5001  22:12:31      0          0
##      6      -11.4997  22:12:31      0          0
```

```
summary(ans3)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value -11.50007  28.99943  38.66049      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE      Zratio Constraint
## CA.2011:Name.Yield-Yield      15.665  5.421e+00  2.890e+00  Positive
## CA.2012:CA.2011:Name.Yield-Yield      6.110  2.485e+00  2.459e+00  Unconstr
## CA.2012:Name.Yield-Yield      4.530  1.821e+00  2.488e+00  Positive
## CA.2013:CA.2011:Name.Yield-Yield      6.384  3.066e+00  2.082e+00  Unconstr
## CA.2013:CA.2012:Name.Yield-Yield      0.393  1.523e+00  2.580e-01  Unconstr
## CA.2013:Name.Yield-Yield      8.597  2.484e+00  3.461e+00  Positive
## CA.2011:units.Yield-Yield      4.970  1.532e+00  3.243e+00  Positive
## CA.2012:CA.2011:units.Yield-Yield      4.087  2.436e-16  1.678e+16  Unconstr
## CA.2012:units.Yield-Yield      5.673  1.301e+00  4.361e+00  Positive
## CA.2013:CA.2011:units.Yield-Yield      4.087  0.000e+00      Inf  Unconstr
## CA.2013:CA.2012:units.Yield-Yield      4.087  0.000e+00      Inf  Unconstr
## CA.2013:units.Yield-Yield      2.557  6.393e-01  4.000e+00  Positive
## =====
## Fixed effects:
## Trait      Effect Estimate Std.Error t.value
```

```
## 1 Yield (Intercept) 16.331 0.8137 20.070
## 2 Yield EnvCA.2012 -5.696 0.7404 -7.693
## 3 Yield EnvCA.2013 -6.271 0.8191 -7.656
## =====
## Groups and observations:
##          Yield
## CA.2011:Name      41
## CA.2012:CA.2011:Name 82
## CA.2012:Name      41
## CA.2013:CA.2011:Name 82
## CA.2013:CA.2012:Name 82
## CA.2013:Name      41
## =====
## Use the '$' sign to access results and parameters
```

As can be seen the `us(Env)` indicates that the genotypes (Name) can have a covariance structure among environments (Env).

4) Multivariate homogeneous variance models

Currently there's a great push for multi-response models. This is motivated by the correlation that certain variables hide and that could benefit in the prediction perspective. In sommer to specify multivariate models the response requires the use of the `cbind()` function in the response, and the `us(trait)`, `diag(trait)`, or `at(trait)` functions in the random part of the model.

```
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          CO02024-9W 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
```

```
DT$EnvName <- paste(DT$Env,DT$Name)
ans4 <- mmer(cbind(Yield, Weight) ~ Env,
            random= ~ vs(Name) + vs(EnvName),
            rcov= ~ vs(units),
            data=DT)
```

```
## iteration  LogLik      wall      cpu(sec)  restrained
##      1      66.0395  22:12:32      1           0
##      2     131.529  22:12:32      1           0
##      3     162.769  22:12:33      2           0
##      4     166.983  22:12:33      2           0
##      5     167.025  22:12:34      3           0
##      6     167.025  22:12:34      3           0
```

```
summary(ans4)
```

```
## =====
##          Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
```



```

##           logLik           AIC           BIC Method Converge
## Value 167.0248 -322.0505 -298.5695      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## u:Name.Yield-Yield      3.7089  1.68117  2.206  Positive
## u:Name.Yield-Weight      0.9071  0.37944  2.391  Unconstr
## u:Name.Weight-Weight      0.2243  0.08775  2.557  Positive
## u:EnvName.Yield-Yield     5.0921  1.47879  3.443  Positive
## u:EnvName.Yield-Weight     1.0269  0.30767  3.338  Unconstr
## u:EnvName.Weight-Weight    0.2101  0.06661  3.154  Positive
## u:units.Yield-Yield       4.3837  0.64941  6.750  Positive
## u:units.Yield-Weight       0.9077  0.14145  6.417  Unconstr
## u:units.Weight-Weight      0.2280  0.03377  6.751  Positive
## =====
## Fixed effects:
##   Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)  16.4093   0.6783  24.191
## 2 Weight (Intercept)  0.9806   0.1497   6.550
## 3 Yield EnvCA.2012  -5.6844   0.7474  -7.606
## 4 Weight EnvCA.2012  -1.1846   0.1593  -7.439
## 5 Yield EnvCA.2013  -6.2952   0.7850  -8.019
## 6 Weight EnvCA.2013  -1.3559   0.1681  -8.065
## =====
## Groups and observations:
##           Yield Weight
## u:Name      41      41
## u:EnvName    94      94
## =====
## Use the '$' sign to access results and parameters

```

You may notice that we have added the `us(trait)` behind the random effects. This is to indicate the structure that should be assumed in the multivariate model. The `diag(trait)` used behind a random effect (i.e. Name) indicates that for the traits modeled (Yield and Weight) there's no a covariance component and should not be estimated, whereas `us(trait)` assumes that for such random effect, there's a covariance component to be estimated (i.e. covariance between Yield and Weight for the random effect Name). Same applies for the residual part (`rcov`).

5) Multivariate heterogeneous variance models

This is just an extension of the univariate heterogeneous variance models but at the multivariate level. This would be a CS+DIAG multivariate model:

```

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           CO02024-9W 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

```

```
DT$EnvName <- paste(DT$Env,DT$Name)
ans5 <- mmer(cbind(Yield, Weight) ~ Env,
             random= ~ vs(Name) + vs(ds(Env),Name),
             rcov= ~ vs(ds(Env),units),
             data=DT)
```

```
## iteration    LogLik      wall    cpu(sec)  restrained
##      1      66.0395  22:12:36      1           0
##      2     138.617  22:12:37      2           0
##      3     172.682  22:12:38      3           0
##      4     177.662  22:12:39      4           0
##      5     177.801  22:12:40      5           0
##      6     177.813  22:12:41      6           0
##      7     177.815  22:12:42      7           0
##      8     177.815  22:12:43      8           0
```

```
summary(ans5)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value 177.8151 -343.6308 -320.1497      NR      TRUE
## =====
## Variance-Covariance components:
##
##           VarComp VarCompSE Zratio Constraint
## u:Name.Yield-Yield      3.31936  1.45269  2.2850  Positive
## u:Name.Yield-Weight      0.79393  0.32621  2.4338  Unconstr
## u:Name.Weight-Weight      0.19085  0.07503  2.5438  Positive
## CA.2011:Name.Yield-Yield  8.70657  4.01470  2.1687  Positive
## CA.2011:Name.Yield-Weight  1.77892  0.83926  2.1196  Unconstr
## CA.2011:Name.Weight-Weight 0.35966  0.17903  2.0089  Positive
## CA.2012:Name.Yield-Yield  2.57109  1.94951  1.3188  Positive
## CA.2012:Name.Yield-Weight  0.33245  0.39840  0.8345  Unconstr
## CA.2012:Name.Weight-Weight 0.03842  0.08595  0.4470  Positive
## CA.2013:Name.Yield-Yield  5.46908  2.16307  2.5284  Positive
## CA.2013:Name.Yield-Weight  1.34713  0.50479  2.6687  Unconstr
## CA.2013:Name.Weight-Weight 0.32902  0.12208  2.6952  Positive
## CA.2011:units.Yield-Yield  4.93852  1.52318  3.2422  Positive
## CA.2011:units.Yield-Weight 0.99447  0.32150  3.0932  Unconstr
## CA.2011:units.Weight-Weight 0.23982  0.07394  3.2433  Positive
## CA.2012:units.Yield-Yield  5.73887  1.31533  4.3631  Positive
## CA.2012:units.Yield-Weight  1.28009  0.30157  4.2448  Unconstr
## CA.2012:units.Weight-Weight 0.31806  0.07286  4.3652  Positive
## CA.2013:units.Yield-Yield  2.56127  0.63993  4.0024  Positive
## CA.2013:units.Yield-Weight  0.44569  0.12645  3.5246  Unconstr
## CA.2013:units.Weight-Weight 0.12232  0.03057  4.0009  Positive
## =====
## Fixed effects:
##      Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)  16.4243    0.7891  20.815
## 2 Weight (Intercept)   0.9866    0.1683   5.863
## 3 Yield EnvCA.2012  -5.7339    0.8266  -6.937
```

```
## 4 Weight EnvCA.2012 -1.1998 0.1698 -7.066
## 5 Yield EnvCA.2013 -6.3128 0.8757 -7.209
## 6 Weight EnvCA.2013 -1.3621 0.1915 -7.114
## =====
## Groups and observations:
##           Yield Weight
## u:Name      41      41
## CA.2011:Name 41      41
## CA.2012:Name 41      41
## CA.2013:Name 41      41
## =====
## Use the '$' sign to access results and parameters
```

6) Multivariate unstructured variance models

This is just an extension of the univariate unstructured variance models but at the multivariate level. This would be a US multivariate model:

```
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           C002024-9W 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
```

```
DT$EnvName <- paste(DT$Env,DT$Name)
ans6 <- mmer(cbind(Yield, Weight) ~ Env,
             random= ~ vs(us(Env),Name),
             rcov= ~ vs(ds(Env),units),
             data=DT)
```

```
## iteration      LogLik      wall      cpu(sec)      restrained
##      1      53.5859  22:12:45      1      0
##      2     140.385  22:12:46      2      0
##      3     176.108  22:12:48      4      0
##      4     181.454  22:12:49      5      0
##      5     181.696  22:12:50      6      0
##      6     181.755  22:12:53      9      0
##      7     181.777  22:12:54     10      0
##      8     181.787  22:12:55     11      0
##      9     181.791  22:12:58     14      0
##     10     181.793  22:12:59     15      0
##     11     181.794  22:13:1      17      0
##     12     181.795  22:13:2     18      0
```

```
summary(ans6)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
```

```

##          logLik          AIC          BIC Method Converge
## Value 181.7941 -351.5895 -328.1085      NR      TRUE
## =====
## Variance-Covariance components:
##
##          VarComp VarCompSE Zratio Constraint
## CA.2011:Name.Yield-Yield      15.6405  5.35495  2.921  Positive
## CA.2011:Name.Yield-Weight      3.3577  1.14603  2.930  Unconstr
## CA.2011:Name.Weight-Weight      0.7180  0.24867  2.887  Positive
## CA.2012:CA.2011:Name.Yield-Yield  6.5289  2.48598  2.626  Unconstr
## CA.2012:CA.2011:Name.Yield-Weight  1.3505  0.52388  2.578  Unconstr
## CA.2012:CA.2011:Name.Weight-Weight  0.2842  0.11259  2.524  Unconstr
## CA.2012:Name.Yield-Yield      4.7896  1.86200  2.572  Positive
## CA.2012:Name.Yield-Weight      0.8641  0.38382  2.251  Unconstr
## CA.2012:Name.Weight-Weight      0.1693  0.08355  2.027  Positive
## CA.2013:CA.2011:Name.Yield-Yield  5.9941  2.93825  2.040  Unconstr
## CA.2013:CA.2011:Name.Yield-Weight  1.4235  0.64974  2.191  Unconstr
## CA.2013:CA.2011:Name.Weight-Weight  0.3379  0.14681  2.302  Unconstr
## CA.2013:CA.2012:Name.Yield-Yield  2.0970  1.44043  1.456  Unconstr
## CA.2013:CA.2012:Name.Yield-Weight  0.5232  0.32355  1.617  Unconstr
## CA.2013:CA.2012:Name.Weight-Weight  0.1339  0.07571  1.769  Unconstr
## CA.2013:Name.Yield-Yield      8.6264  2.47808  3.481  Positive
## CA.2013:Name.Yield-Weight      2.1046  0.58737  3.583  Unconstr
## CA.2013:Name.Weight-Weight      0.5124  0.14279  3.588  Positive
## CA.2011:units.Yield-Yield      4.9516  1.52693  3.243  Positive
## CA.2011:units.Yield-Weight      0.9993  0.32286  3.095  Unconstr
## CA.2011:units.Weight-Weight      0.2411  0.07432  3.244  Positive
## CA.2012:units.Yield-Yield      5.7783  1.32398  4.364  Positive
## CA.2012:units.Yield-Weight      1.2912  0.30401  4.247  Unconstr
## CA.2012:units.Weight-Weight      0.3211  0.07354  4.367  Positive
## CA.2013:units.Yield-Yield      2.5567  0.63882  4.002  Positive
## CA.2013:units.Yield-Weight      0.4452  0.12631  3.524  Unconstr
## CA.2013:units.Weight-Weight      0.1223  0.03056  4.001  Positive
## =====
## Fixed effects:
##      Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)  16.3339    0.8252  19.793
## 2 Weight (Intercept)  0.9677    0.1770   5.467
## 3 Yield EnvCA.2012  -5.6635    0.7447  -7.605
## 4 Weight EnvCA.2012  -1.1855    0.1604  -7.391
## 5 Yield EnvCA.2013  -6.2152    0.8338  -7.454
## 6 Weight EnvCA.2013  -1.3406    0.1805  -7.426
## =====
## Groups and observations:
##          Yield Weight
## CA.2011:Name      41   41
## CA.2012:CA.2011:Name  82   82
## CA.2012:Name      41   41
## CA.2013:CA.2011:Name  82   82
## CA.2013:CA.2012:Name  82   82
## CA.2013:Name      41   41
## =====
## Use the '$' sign to access results and parameters

```

Any number of random effects can be specified with different structures.

7) Random regression models

In order to fit random regression models the user can use the `leg()` function to fit Legendre polynomials. This can be combined with other special covariance structures such as `ds()`, `us()`, etc.

```
library(orthopolynom)
```

```
## Loading required package: polynom
```

```
data(DT_legendre)
```

```
head(DT)
```

```
##      SUBJECT X      Y Xf
## 1.1      1 1 -0.7432795  1
## 2.1      2 1 -0.6669945  1
## 3.1      3 1 -4.2802751  1
## 4.1      4 1  4.1092149  1
## 5.1      5 1 -3.0317213  1
## 6.1      6 1  1.3506577  1
```

```
mRR2<-mmer(Y~ 1 + Xf
           , random=~vs(us(leg(X,1)),SUBJECT)
           , rcov=~vs(units)
           , data=DT)
```

```
## iteration   LogLik      wall   cpu(sec)  restrained
##      1      -145.279  22:13:3      1          0
##      2      -138.353  22:13:3      1          0
##      3      -136.403  22:13:3      1          0
##      4      -136.224  22:13:3      1          0
##      5      -136.222  22:13:3      1          0
##      6      -136.222  22:13:4      2          0
```

```
summary(mRR2)$varcomp
```

```
##              VarComp VarCompSE  Zratio Constraint
## leg0:SUBJECT.Y-Y    2.5782969 0.6717242 3.838326   Positive
## leg1:leg0:SUBJECT.Y-Y 0.4765431 0.2394975 1.989763   Unconstr
## leg1:SUBJECT.Y-Y    0.3497299 0.2183229 1.601893   Positive
## u:units.Y-Y         2.6912226 0.3825197 7.035513   Positive
```

Here, a numeric covariate X is used to explain the trajectory of the SUBJECT's and combined with an unstructured covariance matrix. The details can be found in the theory.

8) GWAS models

Although genome wide association studies can be conducted through a variety of approaches, the use of mixed models to find association between markers and phenotypes still one of the most popular approaches. Two of the most classical and popular approaches is to test marker by marker through mixed modeling (1 model by marker) to obtain the marker effect and an statistic reflecting the level of association usually provided as the $-\log_{10}$ p-value. The second most popular approach is to assume that the genetic variance component is similar for all markers and therefore the variance components are only estimated once (1 model for all markers) and use the inverse of the phenotypic variance matrix (V^{-1}) to test all markers in the generalized linear model $b=(XV-X)^{-1}XV^{-1}y$. This makes the GWAS much faster and efficient without major losses. Given the straight forward extension, sommer provides the `GWAS` function which can fit both type of approaches (be aware that these are 2 among many existant in the literature) in univariate and multivariate models, that way genetically correlated traits can be tested together to increase the power of detection.

Here we show a simple GWAS model for an univariate example.

```
data(DT_cpdata)
#### create the variance-covariance matrix
A <- A.mat(GT) # additive relationship matrix
#### look at the data and fit the model
head(DT,3)

##          id Row Col Year      color  Yield FruitAver Firmness Rowf Colf
## P003 P003   3   1 2014 0.10075269 154.67    41.93  588.917    3   1
## P004 P004   4   1 2014 0.13891940 186.77    58.79  640.031    4   1
## P005 P005   5   1 2014 0.08681502  80.21    48.16  671.523    5   1

head(MP,3)

##          Locus Position Chrom
## 1 scaffold_77830_839          0    1
## 2 scaffold_39187_895          0    1
## 3 scaffold_50439_2379         0    1

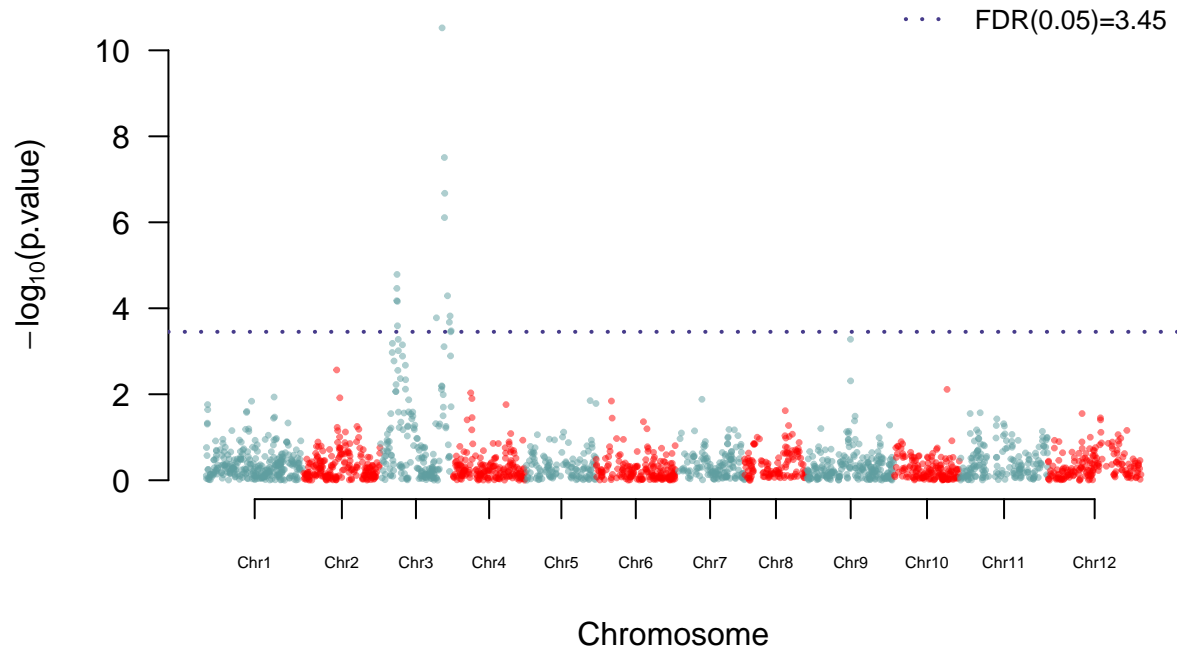
GT[1:3,1:4]

##          scaffold_50439_2381 scaffold_39344_153 uneak_3436043 uneak_2632033
## P003                0                0                0                1
## P004                0                0                0                1
## P005                0                -1                0                1

mix1 <- GWAS(color~1,
             random=~vs(id,Gu=A)
             + Rowf + Colf,
             rcov=~units,
             data=DT,
             M=GT, gTerm = "u:id")

## iteration   LogLik      wall   cpu(sec)  restrained
##      1      -143.207  22:13:5      1          0
##      2      -117.977  22:13:5      1          0
##      3      -109.877  22:13:5      1          1
##      4      -108.178  22:13:6      2          1
##      5      -108.123  22:13:6      2          1
##      6      -108.12   22:13:6      2          1
##      7      -108.12   22:13:7      3          1
## Performing GWAS evaluation

ms <- as.data.frame(t(mix1$scores))
ms$Locus <- rownames(ms)
MP2 <- merge(MP,ms,by="Locus",all.x = TRUE);
manhattan(MP2, pch=20,cex=.5, PVCN = "color score")
```



Be aware that the marker matrix M has to be imputed (no missing data allowed) and make sure that the number of rows in the M matrix is equivalent to the levels of the $gTerm$ specified (i.e. if the $gTerm$ is “id” and has 300 levels or in other words 300 individuals, then M has dimensions $300 \times p$, being p the number of markers).

9) Including special functions

Including special functions + the major $vs()$ function for special variance models + $at()$ specific levels structure + $ds()$ diagonal structure + $us()$ unstructured + $cs()$ customized structure + $overlay()$ overlaid models + $spl2D()$ two dimensional spline models

In a mixed model framework there’s two types of covariance structures, the unknown and known. An example of a known covariance structure is the relationship matrix among individuals commonly present in plant and animal breeding programs. On the other hand, an example of an unknown covariance structure is in a multi-environment trial the covariance among genotypes in these environments, can be assumed diagonal, compound symmetry or unstructured but any needs to be estimated. In the following section we show how to specify unknown and known covariance structured for the random effects.

the $vs()$ function and its auxiliars $ds()$, $us()$, $at()$ and $cs()$

The $vs()$ function allows to fit different types of variance models (please take the time to read the documentation of this function). As explained in the introduction to covariance structures section in this document, the terms in the $vs()$ function define the kronecker products that will be performed to define the variance and covariance components to be estimated. For example:

```
fixed=cbind(Y1,Y2,Y3)1 random=~vs(ds(Env),us(Time),Geno, Gu=A, Gtc=unsm(3)) rcov=~vs(ds(Env),us(Time),units)
```

defines a very complex model for the Geno random effect, where assumes that genotypes in different environments will be independent (diagonal structure using ds() function), but within each environment the different time points hold an unstructured variance-covariance structure (using the us() function), and at the same time a known covariance structure for Geno is specified in the Gu argument (here A is a square matrix provided by the user).

the Gtc argument for constraints

At the same time all these is embedded in a multivariate model and the var-cov model is specified in the Gtc argument, here a full unstructured multivariate model is used by putting a 3x3 matrix in the Gtc argument with the following format:

$$\mathbf{Gtc} = \begin{bmatrix} 1 & 2 & 2 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

By default, sommer assumes an unstructured model if the Gtc argument is not provided. If the user wanted a DIAG model for the multivariate structure the argument would be Gtc=diag(3) which is again a 3x3 matrix but of a diagonal form:

$$\mathbf{Gtc} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Remember that the numbers of the Gtc argument define the constraint applied in the model (1 positive, 2 unconstrained, 3 fixed)

Estimating a DIAG unknown covariance structure among genotypes in different environments (using the ds() function), same for residuals, and using a known covariance structure among genotypes (additive relationship matrix A applied in the Gu argument of the vs function).

```
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           CO02024-9W 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
```

```
ans2 <- mmer(Yield~Env,
             random= ~ vs(ds(Env),Name, Gu=A),
             rcov= ~ vs(ds(Env),units),
             data=DT)
```

```
## iteration   LogLik      wall   cpu(sec)  restrained
##      1      -42.26  22:13:24      0          0
##      2      -25.3744  22:13:24      0          0
##      3      -19.1877  22:13:24      0          0
##      4      -18.3538  22:13:24      0          0
##      5      -18.3432  22:13:24      0          0
##      6      -18.343  22:13:24      0          0
```



```
summary(ans2)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value -18.34316 42.68598 52.34705      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## CA.2011:Name.Yield-Yield 17.214 6.1570 2.796 Positive
## CA.2012:Name.Yield-Yield 4.597 1.8361 2.503 Positive
## CA.2013:Name.Yield-Yield 8.790 2.5463 3.452 Positive
## CA.2011:units.Yield-Yield 4.954 1.5284 3.241 Positive
## CA.2012:units.Yield-Yield 5.663 1.2984 4.362 Positive
## CA.2013:units.Yield-Yield 2.557 0.6393 4.000 Positive
## =====
## Fixed effects:
##   Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept) 16.622 0.9485 17.525
## 2 Yield EnvCA.2012 -5.969 1.0447 -5.713
## 3 Yield EnvCA.2013 -6.659 1.0981 -6.064
## =====
## Groups and observations:
##           Yield
## CA.2011:Name 41
## CA.2012:Name 41
## CA.2013:Name 41
## =====
## Use the '$' sign to access results and parameters
```

and for multivariate models:

```
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 CO02024-9W 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
```

```
ans2 <- mmer(cbind(Yield,Weight)~Env,
             random= ~ vs(ds(Env),Name, Gu=A, Gtc=unsm(2)),
             rcov= ~ vs(ds(Env),units, Gtc=diag(2)),
             data=DT)
```

```
## iteration      LogLik      wall      cpu(sec)      restrained
## 1 -62.7426 22:13:25 1 0
## 2 26.0621 22:13:26 2 0
## 3 76.3155 22:13:27 3 0
## 4 92.1779 22:13:27 3 0
## 5 92.4933 22:13:29 5 0
```

```
##      6      92.4963  22:13:30      6      0
##      7      92.4963  22:13:31      7      0
```

```
summary(ans2)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value 92.4963 -172.9927 -149.5116      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## CA.2011:Name.Yield-Yield  17.2101  6.12630  2.809  Positive
## CA.2011:Name.Yield-Weight   4.1997  1.30601  3.216  Unconstr
## CA.2011:Name.Weight-Weight   0.7979  0.28585  2.791  Positive
## CA.2012:Name.Yield-Yield   4.9114  1.87421  2.620  Positive
## CA.2012:Name.Yield-Weight   1.5623  0.36980  4.225  Unconstr
## CA.2012:Name.Weight-Weight   0.2031  0.08881  2.286  Positive
## CA.2013:Name.Yield-Yield   8.7891  2.53798  3.463  Positive
## CA.2013:Name.Yield-Weight   2.3723  0.60100  3.947  Unconstr
## CA.2013:Name.Weight-Weight   0.5259  0.14763  3.562  Positive
## CA.2011:units.Yield-Yield   4.8687  1.49433  3.258  Positive
## CA.2011:units.Weight-Weight  0.2363  0.07249  3.259  Positive
## CA.2012:units.Yield-Yield   5.4932  1.25629  4.373  Positive
## CA.2012:units.Weight-Weight  0.3031  0.06925  4.377  Positive
## CA.2013:units.Yield-Yield   2.5280  0.62975  4.014  Positive
## CA.2013:units.Weight-Weight  0.1209  0.03014  4.010  Positive
## =====
## Fixed effects:
##   Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)  16.623    0.9463  17.566
## 2 Weight (Intercept)   1.035    0.2044   5.065
## 3 Yield EnvCA.2012   -5.949    1.0454  -5.691
## 4 Weight EnvCA.2012  -1.251    0.2256  -5.547
## 5 Yield EnvCA.2013   -6.661    1.0958  -6.078
## 6 Weight EnvCA.2013  -1.445    0.2440  -5.923
## =====
## Groups and observations:
##           Yield Weight
## CA.2011:Name    41    41
## CA.2012:Name    41    41
## CA.2013:Name    41    41
## =====
## Use the '$' sign to access results and parameters
```

customized random effects

One of the most powerful features of sommer is the ability to provide any customized matrix and estimate any random effect. For example:

```
data(DT_cpdata)
GT[1:4,1:4]
```

```
##      scaffold_50439_2381 scaffold_39344_153 uneak_3436043 uneak_2632033
## P003                0                0                0                1
## P004                0                0                0                1
## P005                0                -1                0                1
## P006               -1                -1               -1                0
```

```
#### look at the data and fit the model
```

```
mix1 <- mmer(Yield~1,
             random=~vs(list(GT)),
             rcov=~units,
             data=DT)
```

```
## iteration   LogLik      wall   cpu(sec)   restrained
##      1      -286.365  22:13:32     1           0
##      2      -236.78   22:13:33     2           0
##      3      -200.635  22:13:33     2           0
##      4      -180.045  22:13:33     2           0
##      5      -176.4    22:13:33     2           0
##      6      -176.211  22:13:34     3           0
##      7      -176.207  22:13:34     3           0
##      8      -176.207  22:13:34     3           0
```

the matrix GT is provided as a random effect by encapsulating the matrix in a list and provided in the vs() function.

the overlay() function

Another very useful function is the overlay function, which allows to overlay matrices of different random effects and estimate a single variance component for the overlaid terms.

```
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(DT$femalef,DT$malef))
             + genof,
             data=DT)
```

```
## iteration   LogLik      wall   cpu(sec)   restrained
##      1      -10.425  22:13:43     0           0
##      2       -6.487  22:13:43     0           0
##      3       -5.732  22:13:43     0           0
##      4      -5.67494  22:13:43     0           0
##      5      -5.67441  22:13:43     0           0
```

here the femalef and malef random effects are overlayed becoming a single random effect that has the same variance component.

the spl2D() function (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 necessity 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	22:13:45	1	0
## 2	-152.064	22:13:45	1	0
## 3	-151.265	22:13:45	1	0
## 4	-151.202	22:13:46	2	0
## 5	-151.201	22:13:46	2	0

```
summary(mix)

## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value -151.2016 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.

10) The specification of constraints

One of the major strengths of sommer is its extreme flexibility to specify variance-covariance structures in the multi-trait framework. Since sommer 3.7 this is easily achieved by the use of the `vs()` function and its argument `Gtc`. The idea behind how to specify the constraints has been explained in section 7) and here we will only show some examples.

Some useful function to create constrained matrices quickly are `unsm()` for unstructured, `uncm` for unconstrained, `fixm()` for fixed constraint, and `fcm()` for fixed effect constrains and its use is very easy:

```
unsm(4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    2    2
## [2,]    2    1    2    2
## [3,]    2    2    1    2
## [4,]    2    2    2    1
```

can be used in `vs(x,Gtc=unsm(4))` to specify unstructured model for RE x

```
uncm(4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    2    2    2
## [2,]    2    2    2    2
## [3,]    2    2    2    2
## [4,]    2    2    2    2
```

can be used in `vs(x,Gtc=uncm(4))` to specify unconstrained model for RE x

```
fixm(4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    3    3    3    3
## [2,]    0    3    3    3
## [3,]    0    0    3    3
## [4,]    0    0    0    3
```

can be used in `vs(x,Gtc=fixm(4),Gt=mm)` to specify a fixed var-cov model for RE x and Gt needs to be provided

```
fcm(c(1,0,1,0))
```

```
##      [,1] [,2]
## [1,]    1    0
```

```
## [2,] 0 0
## [3,] 0 1
## [4,] 0 0
```

can be used in `vs(xf,Gtc=fcm(c(1,0,1,0)))` to specify that the fixed effect `xf` should be only estimated for traits 1 and 3

A matrix can combine the different constraints (0: not estimated, 1: positive, 2:unconstrained, 3:fixed) as desired.

First we show how to fit fixed effects for an specific trait. Here we assume two traits (Yield and Weight) and a fixed effect called “Env”, which we only want to fit for the trait number 2 and defaults for the random effects.

```
data(DT_example)
ansf <- mmer(cbind(Yield,Weight)~vs(Env,Gtc=fcm(c(0,1))),
            random= ~ vs(ds(Env),Name),
            rcov= ~ vs(ds(Env),units),
            data=DT)
```

```
## iteration    LogLik      wall    cpu(sec)  restrained
##      1      -15.613  22:13:47      0          0
##      2       96.0781  22:13:48      1          0
##      3      146.282  22:13:49      2          0
##      4      156.447  22:13:50      3          0
##      5      158.156  22:13:51      4          0
##      6      158.93   22:13:52      5          0
##      7      159.285  22:13:53      6          0
##      8      159.445  22:13:54      7          0
##      9      159.516  22:13:55      8          0
##     10      159.548  22:13:56      9          0
##     11      159.562  22:13:57     10          0
##     12      159.568  22:13:58     11          0
##     13      159.571  22:13:59     12          0
##     14      159.572  22:14:00     13          0
##     15      159.572  22:14:01     14          0
```

```
summary(ansf)
```

```
## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value 159.5719 -311.1449 -295.4909      NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## CA.2011:Name.Yield-Yield    51.6351  16.40157  3.148  Positive
## CA.2011:Name.Yield-Weight    11.0591   3.50882  3.152  Unconstr
## CA.2011:Name.Weight-Weight   2.3664   0.75367  3.140  Positive
## CA.2012:Name.Yield-Yield     4.5017   1.81385  2.482  Positive
## CA.2012:Name.Yield-Weight     0.8621   0.38456  2.242  Unconstr
## CA.2012:Name.Weight-Weight    0.1824   0.08619  2.117  Positive
## CA.2013:Name.Yield-Yield     9.1116   2.60811  3.494  Positive
## CA.2013:Name.Yield-Weight     2.2282   0.62058  3.590  Unconstr
## CA.2013:Name.Weight-Weight    0.5432   0.15137  3.588  Positive
```

```

## CA.2011:units.Yield-Yield    4.9660    1.53246    3.241    Positive
## CA.2011:units.Yield-Weight    1.0013    0.32371    3.093    Unconstr
## CA.2011:units.Weight-Weight    0.2413    0.07445    3.241    Positive
## CA.2012:units.Yield-Yield    5.6712    1.29969    4.364    Positive
## CA.2012:units.Yield-Weight    1.2616    0.29732    4.243    Unconstr
## CA.2012:units.Weight-Weight    0.3131    0.07174    4.365    Positive
## CA.2013:units.Yield-Yield    2.5508    0.63739    4.002    Positive
## CA.2013:units.Yield-Weight    0.4438    0.12600    3.522    Unconstr
## CA.2013:units.Weight-Weight    0.1220    0.03050    4.001    Positive
## =====
## Fixed effects:
##   Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept) 10.678099    0.33613 31.7678
## 2 Weight (Intercept) -0.233747    0.07737 -3.0213
## 3 Weight EnvCA.2012  0.017286    0.04188  0.4127
## 4 Weight EnvCA.2013 -0.008647    0.04116 -0.2101
## =====
## Groups and observations:
##           Yield Weight
## CA.2011:Name    41    41
## CA.2012:Name    41    41
## CA.2013:Name    41    41
## =====
## Use the '$' sign to access results and parameters

```

Now we specify an unstructured model for the random effect Name and the residuals and after a diagonal for both.

```

data(DT_example)
ans.uns <- mmer(cbind(Yield,Weight)~Env,
               random= ~ vs(Name,Gtc=unsm(2)),
               rcov= ~ vs(units,Gtc=unsm(2)),
               data=DT)

```

```

## iteration    LogLik      wall    cpu(sec)    restrained
##      1      56.6189    22:14:2      0          0
##      2     117.266    22:14:2      0          0
##      3     149.82    22:14:3      1          0
##      4     154.605    22:14:3      1          0
##      5     154.655    22:14:4      2          0
##      6     154.655    22:14:4      2          0

```

```
summary(ans.uns)
```

```

## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value 154.6549 -297.3108 -273.8298    NR    TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## u:Name.Yield-Yield    4.8592    1.52160    3.193    Positive
## u:Name.Yield-Weight    1.1432    0.34851    3.280    Unconstr
## u:Name.Weight-Weight    0.2737    0.08163    3.353    Positive

```

```

## u:units.Yield-Yield      8.1015    0.96013  8.438   Positive
## u:units.Yield-Weight    1.6523    0.20192  8.183   Unconstr
## u:units.Weight-Weight   0.3792    0.04496  8.434   Positive
## =====
## Fixed effects:
##   Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)  16.3396    0.5824  28.058
## 2 Weight (Intercept)  0.9641    0.1312   7.345
## 3 Yield EnvCA.2012  -5.6429    0.5712  -9.878
## 4 Weight EnvCA.2012  -1.1739    0.1245  -9.429
## 5 Yield EnvCA.2013  -6.1768    0.6064 -10.186
## 6 Weight EnvCA.2013  -1.3292    0.1327 -10.019
## =====
## Groups and observations:
##      Yield Weight
## u:Name   41     41
## =====
## Use the '$' sign to access results and parameters

```

```

ans.diag <- mmer(cbind(Yield,Weight)~Env,
  random= ~ vs(Name,Gtc=diag(2)),
  rcov= ~ vs(units,Gtc=diag(2)),
  data=DT)

```

```

## iteration   LogLik      wall    cpu(sec)  restrained
##      1      -74.2545  22:14:4      0           0
##      2      -65.3278  22:14:5      1           0
##      3      -63.8888  22:14:5      1           0
##      4      -63.8151  22:14:5      1           0
##      5      -63.815   22:14:6      2           0

```

```
summary(ans.diag)
```

```

## =====
##           Multivariate Linear Mixed Model fit by REML
## ***** sommer 3.7 *****
## =====
##           logLik      AIC      BIC Method Converge
## Value -63.81513 139.6301 163.1111   NR      TRUE
## =====
## Variance-Covariance components:
##           VarComp VarCompSE Zratio Constraint
## u:Name.Yield-Yield  4.8559  1.52330  3.188   Positive
## u:Name.Weight-Weight  0.2733  0.08158  3.351   Positive
## u:units.Yield-Yield  8.1086  0.96145  8.434   Positive
## u:units.Weight-Weight  0.3793  0.04499  8.432   Positive
## =====
## Fixed effects:
##   Trait      Effect Estimate Std.Error t.value
## 1 Yield (Intercept)  16.3850    0.5849  28.012
## 2 Weight (Intercept)  0.9661    0.1313   7.359
## 3 Yield EnvCA.2012  -5.6880    0.5741  -9.908
## 4 Weight EnvCA.2012  -1.1756    0.1246  -9.437
## 5 Yield EnvCA.2013  -6.2183    0.6107 -10.182
## 6 Weight EnvCA.2013  -1.3304    0.1328 -10.021

```



```
## =====
## Groups and observations:
##      Yield Weight
## u:Name   41     41
## =====
## Use the '$' sign to access results and parameters
```

As a final example we will fit a multivariate model to deal with separate sexes which is a common problem in animal genetics.

```
# Generate some fake data:
# 100 males and 100 females
# Two traits are measured on each male, and two traits on each female
# 20 individuals per sex are measured for each of 5 different genotypes
set.seed(3434)
df <- data.frame(
  sex = rep(c("female", "male"), each = 100),
  female_trait_1 = c(rnorm(100), rep(NA, 100)),
  female_trait_2 = c(rnorm(100), rep(NA, 100)),
  male_trait_1 = c(rep(NA, 100), rnorm(100)),
  male_trait_2 = c(rep(NA, 100), rnorm(100)),
  genotype = rep(rep(1:5, each = 20), 2),
  individual = 1:200
)
df$genotype <- as.factor(df$genotype)
df$individual <- as.factor(df$individual)

mm <- adiagl(unsm(2),unsm(2));mm
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    0    0
## [2,]    2    1    0    0
## [3,]    0    0    1    2
## [4,]    0    0    2    1
```

```
# mix <- mmer(cbind(female_trait_1,
#                   female_trait_2,
#                   male_trait_1,
#                   male_trait_2) ~ 1,
#                   random=~vs(genotype,Gtc=unsm(4)) + vs(individual,Gtc=mm),
#                   rcov=~vs(units), na.method.Y = "include",
#                   data=df)
# summary(mix)
```

I have silenced this code because data is not meaningful but this must show the way.

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

Covarrubias-Pazaran 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.