Frequently asked questions for the sommer package

Giovanny Covarrubias-Pazaran

2019-08-24

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 vignette is to provide answers to frequently asked questions (FAQ) related to performance and possible issues:

1) I got an error similar to:

```
# iteration  LogLik   wall   cpu(sec) restrained
# 1          -224.676 18:11:23 3 0
# System is singular. Aborting the job. You could try a bigger tolparinv value.
```

This error indicates that your model is singular (phenotypic variance \( V \) matrix is not invertible) and therefore the model is stopped throwing the error message and returning an empty list. Whether you can try a simpler model or just modify the argument `tolparinv` in the `mmer` function. The default is 1e-3, which means that it will try to invert \( V \) and if it fails it will try to add a small value to the diagonal of \( V \) of 1e-3 to make it invertible and try bigger and bigger numbers. If this fails then the program will return the empty list.

Sometimes the model becomes singular when you use variance covariance matrices (i.e. genomic relationship matrices) that are not full-rank. You can try to make it full-rank and try again.

2) My model runs very slow

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.

3) Can I run rrBLUP for markers and GBLUP for individuals in sommer?

Both types of models can be fitted in sommer. The only thing that it changes is what is the random effect of interest; the marker matrix or the identifier for the individual. Here there is a complex example using multi-trait models but can be used with only one trait.

```
library(sommer)
## rrBLUP for makers
data(DT_cpdata)
```
DT <- DT_cpdata
GT <- GT_cpdata
MP <- MP_cpdata

mix.rglblup <- mmer(fixed=color ~ 1,
                      random = vs(GT, Gtc=unsm(1)) + vs(Rowf, Gtc=diag(1)),
                      rcov = vs(units, Gtc=unsm(1)), getPEV = FALSE,
                      data = DT, verbose = FALSE)

summary(mix.rglblup)

## ============================================================
## Multivariate Linear Mixed Model fit by REML
## ********************** sommer 4.0 **********************
## ============================================================
## logLik AIC BIC Method Converge
## Value -108.1202 218.2403 222.132 NR TRUE
## ============================================================
## Variance-Covariance components:
## VarComp VarCompSE Zratio Constraint
## u:GT.color-color 4.213e-06 8.581e-07 4.909 Positive
## u:Rowf.color-color 1.963e-04 1.355e-04 1.449 Positive
## u:units.color-color 2.612e-03 2.926e-04 8.928 Positive
## ============================================================
## Fixed effects:
## Trait Effect Estimate Std.Error t.value
## 1 color (Intercept) 0.1692 0.03908 4.329
## ============================================================
## Groups and observations:
## color
## u:GT 2889
## u:Rowf 13
## ============================================================
## Use the ‘$’ sign to access results and parameters

A <- A.mat(GT)

mix.gblup <- mmer(fixed=color ~ 1,
                      random = vs(id, Gu=A, Gtc=unsm(1)) + vs(Rowf, Gtc=diag(1)),
                      rcov = vs(units, Gtc=unsm(1)),
                      data = DT, verbose = FALSE)

summary(mix.gblup)

## ============================================================
## Multivariate Linear Mixed Model fit by REML
## ********************** sommer 4.0 **********************
## ============================================================
## logLik AIC BIC Method Converge
## Value -108.1201 218.2403 222.1319 NR TRUE
## ============================================================
## Variance-Covariance components:
## VarComp VarCompSE Zratio Constraint
## u:id.color-color 0.0049524 0.0010082 4.912 Positive
## u:Rowf.color-color 0.0001965 0.0001359 1.446 Positive
## u:units.color-color 0.0026123 0.0002926 8.928 Positive
## ============================================================
## Fixed effects:
4) I am missing BLUPs for individuals even when I provided them in the relationship matrix

I got this good question in the past: “when I want to fit an animal model with sommer package using additive relationship matrix(A), this A matrix would contain parents. But the random effects only contains animals in the random effect but not including parents in the A matrix. How can I get the random effects for parents?”

Answer: The easy way to do it is to make sure that even if the parents don’t show up in the dataset, you need to make sure that they are present in the levels of the column that contains the individuals (i.e. animal IDs), in addition they have to be provided in the relationship matrix and that’s it. They should be returned in the blups.

```r
library(sommer)

data(DT_cpdata)
DT <- DT_cpdata
GT <- GT_cpdata
MP <- MP_cpdata

### create the variance-covariance matrix
```
A <- A.mat  # additive relationship matrix  
### look at the data and fit the model  
set.seed(12)  
DT2 <- droplevels(DT[sample(1:nrow(DT),100),])  # we simulate a dataset with only 100 animals  
nrow(DT2); length(levels(DT2$id))

## [1] 100
## [1] 100  
# we fit a model with the reduced dataset where only 100 blups will be returned since only  
# 100 levels exist in the "id" column  
mix1 <- mmer(Yield~1,  
  random=-vs(id,Gu=A)  
  + Rowf + Colf,  
  rcov=-units,  
  data=DT2, verbose = FALSE)  
summary(mix1)

# ==============================================================  
# Multivariate Linear Mixed Model fit by REML  
# ********************** sommer 4.0  **********************  
# ==============================================================  
# logLik  AIC  BIC  Method  Converge  
# Value -45.86523 93.73045 96.32557 NR  TRUE  
# ==============================================================  
# Variance-Covariance components:  
# VarComp  VarCompSE  Zratio  Constraint  
# u:id.Yield-Yield 1655.45 990.7 1.6710  Positive  
# Rowf.Yield-Yield 747.25 529.5 1.4111  Positive  
# Colf.Yield-Yield 46.52 379.4 0.1226  Positive  
# units.Yield-Yield 2811.86 786.3 3.5759  Positive  
# ==============================================================  
# Fixed effects:  
# Trait  Effect  Estimate  Std.Error  t.value  
# 1 Yield (Intercept) 133.7 9.908 13.49  
# ==============================================================  
# Groups and observations:  
# Yield  
# u:id 100  
# Rowf 13  
# Colf 34  
# ==============================================================  
# Use the "$" sign to access results and parameters  
length(mix1$U$u:id$Yield)  # only 100 levels

## [1] 100  
# we add additional levels to the "id" column and also provide them in the relationship matrix  
levels(DT2$id) <- c(levels(DT$id), setdiff(levels(DT$id), levels(DT2$id)))  
mix2 <- mmer(Yield~1,  
  random=-vs(id,Gu=A)  
  + Rowf + Colf,  
  rcov=-units,  
  data=DT2, verbose = FALSE)
## Multivariate Linear Mixed Model fit by REML

### sommer 4.0

### logLik AIC BIC Method Converge

| Value | -45.86523 | 93.73045 | 96.32557 | NR | TRUE |

### Variance-Covariance components:

<table>
<thead>
<tr>
<th>VarComp</th>
<th>VarCompSE</th>
<th>Zratio</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>u:id.Yield-Yield</td>
<td>1655.45</td>
<td>990.7</td>
<td>1.6710</td>
</tr>
<tr>
<td>Rowf.Yield-Yield</td>
<td>747.25</td>
<td>529.5</td>
<td>1.4111</td>
</tr>
<tr>
<td>Colf.Yield-Yield</td>
<td>46.52</td>
<td>379.4</td>
<td>0.1226</td>
</tr>
<tr>
<td>units.Yield-Yield</td>
<td>2811.86</td>
<td>786.3</td>
<td>3.5759</td>
</tr>
</tbody>
</table>

### Fixed effects:

<table>
<thead>
<tr>
<th>Trait</th>
<th>Effect</th>
<th>Estimate</th>
<th>Std.Error</th>
<th>t.value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yield</td>
<td>(Intercept)</td>
<td>133.7</td>
<td>9.908</td>
<td>13.49</td>
</tr>
</tbody>
</table>

### Groups and observations:

<table>
<thead>
<tr>
<th>Yield</th>
<th>u:id</th>
<th>363</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rowf</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Colf</td>
<td>34</td>
<td></td>
</tr>
</tbody>
</table>

### Use the $' \times $ sign to access results and parameters

```r
length(mix2$U`u:id`$Yield) # now 363 levels
```

```r
# [1] 363
```

5) How can I use the AR1(), CS() and ARMA() functions

Sommer doesn’t support the estimation of additional correlation components like AR1 in the way asreml does. Still, if the user know the correlation value or can do an iterative approach to find the best value then these functions can be used to specify the variance covariance structure for a given random effect.

For example, in the DT_cpdata dataset we have a field with row and column coordinates. This allows to fit row and column as random effects:

```r
library(sommer)
data(DT_cpdata)
DT <- DT_cpdata
mix1 <- mmer(Yield ~ 1,
random = Rowf + Colf,
rcov = units,
data = DT, verbose = FALSE)
summary(mix1)$varcomp
```

```r
# VarComp VarCompSE Zratio Constraint
| Rowf.Yield-Yield | 832.2879 | 393.8951 | 2.112968 | Positive |
| Colf.Yield-Yield | 153.9201 | 126.7582 | 1.214281 | Positive |
| units.Yield-Yield | 3647.3486 | 290.4910 | 12.555804 | Positive |
```

5
If the user wants to relax the independence between rows and define an AR1 covariance structure among rows then the model could be fitted as:

```r
library(sommer)
data(DT_cpdata)
DT <- DT_cpdata
mixAR1row <- mmer(Yield ~ 1,
                   random = vs(Rowf, Gu=AR1(Rowf, rho=0.3)) + Colf,
                   rcov = units,
                   data = DT, verbose = FALSE)
summary(mixAR1row)$varcomp
```

```
# VarComp VarCompSE Zratio Constraint
## u:Rowf.Yield-Yield 791.8219 387.8695 2.041465 Positive
## Colf.Yield-Yield 154.5660 126.8094 1.218885 Positive
## units.Yield-Yield 3643.6027 290.1689 12.556834 Positive
```

Same could be done for the column random effect:

```r
library(sommer)
data(DT_cpdata)
DT <- DT_cpdata
mixAR1col <- mmer(Yield ~ 1,
                   random = Rowf + vs(Colf, Gu=AR1(Colf, rho=0.3)),
                   rcov = units,
                   data = DT, verbose = FALSE)
summary(mixAR1col)$varcomp
```

```
# VarComp VarCompSE Zratio Constraint
## Rowf.Yield-Yield 830.3623 392.8264 2.113815 Positive
## u:Colf.Yield-Yield 178.7490 134.2703 1.331262 Positive
## units.Yield-Yield 3624.6074 287.6072 12.602629 Positive
```

If on the other hand, you would like to establish the presence of correlation in row and columns at the same time the model would look like this:

```r
library(sommer)
data(DT_cpdata)
DT <- DT_cpdata
mixAR1rowcol <- mmer(Yield ~ 1,
                      random = vs(Rowf:Colf,
                                  Gu=kronecker(AR1(Rowf, rho=0.3),AR1(Colf, rho=0.3),make.dimnames = TRUE),
                                  make.dimnames = TRUE),
                      rcov = units,
                      data = DT, verbose = FALSE)
summary(mixAR1rowcol)$varcomp
```

```
# VarComp VarCompSE Zratio Constraint
## units.Yield-Yield 2025.584 622.1023 3.256030 Positive
```

Notice that if you specify a random effect that is the interaction between 2 random effects the covariance structure to be specified in the Gu argument has to be built using the `kronecker()` function. The same applies to the `ARMA()` and `CS()` functions. Please keep in mind that the correlation values (rho argument) is a fixed value not estimated by REML like asreml does but you can always follow an iterative approach.
6) Can I run GWAS in MET experiments with replicates?

Although the direct-inversion algorithm that sommer uses in the background is not the best choice to solve GWAS models for the n > p scenario it is still possible to perform GWAS in MET models.

For example, assume a MET model that has 41 lines with 1000 SNP markers, tested in 3 environments. The genetic term in a MET can be modeled as CS, DIAG or US covariance. The whole point of the GWAS function in sommer is to provide the name of the gTerm (random effect) that will match the marker data provided.

We first make up marker data for the MET data for example purposes

```r
library(sommer)
data(DT_example)
DT <- DT_example
A <- A_example
M <- matrix(rep(0, 41*1000), 1000, 41)
for (i in 1:41) {
  M[,i] <- ifelse(runif(1000) < 0.5, -1, 1)
}
tM <- t(M)
```

We then fit the MET model of the type compound symmetry (CS) and evaluate the GWAS for the main genetic term “Name”:

```r
## GWAS for main term in CS model
ansx <- GWAS(Yield ~ Env,
             random = ~ Name + Env:Name,
             rcov = ~ units,
             data = DT,
             M = tM,
             gTerm = "Name", verbose = FALSE)

## Performing GWAS evaluation
ms <- as.data.frame(t(ansx$scores))
plot(ms$`Yield score`, ylim=c(0,8))
```
We could do the same but evaluate the GWAS considering the genetic term as the interaction term “Env:Name”

```r
## GWAS for the interaction term in CS model
E <- matrix(1, nrow = length(unique(DT$Env))); E

## Performing GWAS evaluation
ms <- as.data.frame(t(ansx$scores))
plot(ms$'Yield score', ylim=c(0,8))
```
If the MET is a diagonal model, there’s a variance components and BLUPs for each environment. Therefore
we can evaluate the GWAS at any environment, here for example we evaluate the GWAS at the genetic term
in the environment ‘CA.2011’.

## GWAS for the interaction term in DIAG model
E <- matrix(1, nrow = length(unique(DT$Env))); E

## [,1]
## [1,] 1
## [2,] 1
## [3,] 1
EtM <- kronecker(E, tM)
ansx <- GWAS(Yield ~ Env, 
              random = ~ Name + vs(ds(Env), Name), 
              rcov = ~ vs(ds(Env), units), 
              data = DT, 
              M = tM, 
              gTerm = "CA.2011:Name", verbose = FALSE )

## Performing GWAS evaluation
ms <- as.data.frame(t(ansx$scores))
plot(ms$`Yield score`, ylim = c(0, 8))
If the MET is an unstructured model, there’s a variance components and BLUPs for each environment and a covariance component among the different combinations of environments. Therefore we can evaluate the GWAS at any environment as before, here for example we evaluate the GWAS at the genetic term in the environment ‘CA.2011’. The difference with the previous model is that here we expect a greater accuracy in the environment CA.2011 since it has borrowed information from the other environments given the covariance fitted among environments.

```r
## GWAS for main term in US model
ansx <- GWAS(Yield~Env,  
               random= ~vs(us(Env),Name),  
               rcov= ~ vs(us(Env),units),  
               data=DT,  
               M=tM,  
               gTerm = "CA.2011:Name", verbose = FALSE)
```

```r
## Performing GWAS evaluation
ms <- as.data.frame(t(ansx$scores))
plot(ms$`Yield score`, ylim=c(0,8))
```
Same theory applies for the multitrait model:

```r
## GWAS for main term in multitrait DIAG model
ansx <- GWAS(cbind(Weight, Yield) ~ Env,
             random = ~vs(ds(Env), Name, Gtc=unsm(2)),
             rcov = ~ vs(ds(Env), units, Gtc=diag(2)),
             data = DT,
             M = tM,
             gTerm = "CA.2011:Name", verbose = FALSE)

## Performing GWAS evaluation
ms <- as.data.frame(t(ansx$scores))
plot(ms$'Yield score', ylim=c(0,8))
```
Literature


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


