Moving from older to newer versions of 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. 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.

Recently, I decided to code the main algorithm (Newton-Raphson & Average-Information) in C++ which encouraged me to refactor all the machinery including special functions and specification of the models. For a more in depth explanation of how the machinery works please read the "Quick start for the sommer package" vignette [i.e. typing vignette('sommer.start')]. Here I will focus in just making a translation of the old specification to the new specification.

1) The specification of multiresponse model

In past versions (depending how old your version) there was an argument called `MVM` which had to be set to TRUE if the user wanted to run a true multi-trait model since the specification

```
fixed= cbind(y1,y2)~x
```

would by default fit 2 univariate models in parallel. That is no longer the case, the `MVM` argument doesn't exist and if a model like the one above is specified it will run a tru multi-trait model.

2) The specification of multivariate unknown covariance structures

In the previous versions when I introduced the multivariate solver I decided to follow the asreml sintax to specify the unknown covariance structured that needed to be estimated. For example, a diagonal model for the multitrait model, assuming a random effect called 're' looked something like this:

```
fixed= cbind(y1,y2)~x
```

```
random= ~ diag(trait):re
```

and an unstructured multitrait model was:

```
random= ~ us(trait):re
```

Although this was easier for users adapted to asreml a good thing, it put a lot of limitations on the way constraints for variance components where specified. The same model in the new versions looks like this:

```
random= ~ vs(re, Gtc=unsm(2))
```

where the `Gtc` argument helps us to indicate what type of structure this random effect represents. Here I specified an unstructured model with the function `unsm()` with a number 2 for 2 traits. The user can use either `diag()` or `uncm()`, or any customized matrix with dimensions t x t (t being the number of traits) containing the number 0,1,2,3 that specify the constraint:

0: not to be estimated 1: estimated and constrained to be positive (i.e. variance component) 2: estimated and unconstrained (can be negative or positive, i.e. covariance component) 3: not to be estimated but fixed (value has to be provided in the Gt argument)

All this models fit a model with the following variance for `re`:

$var(u) = T \otimes A$

where:

$$\mathbf{var(u)} = \begin{bmatrix} \sigma^2_{g_{t1,t1}} & \sigma_{g_{t1,t2}} \\ \sigma_{g_{t2,t1}} & \sigma^2_{g_{t2,t2}} \end{bmatrix} \otimes A$$

By doing this change now the user has full control on the constraints applied to the estimation of variance components and can provide initial values easily using the `Gt` argument.

3) The specification of additional unknown covariance structures

If we focus for a moment in an univariate mixed model we can also have other unknown covariance structures specified.

$var(u) = E \otimes ... \otimes F \otimes A$

where:

$$\mathbf{var(u)} = \begin{bmatrix} \sigma^2_{g_{e1,e1}} & \sigma_{g_{e1,e2}} & \sigma_{g_{e1,e3}} \\ \sigma_{g_{e2,e1}} & \sigma^2_{g_{e2,e2}} & \sigma_{g_{e2,e3}} \\ \sigma_{g_{e3,e1}} & \sigma_{g_{e3,e2}} & \sigma^2_{g_{e3,e3}} \end{bmatrix} \otimes ... \otimes \begin{bmatrix} \sigma^2_{g_{f1,f1}} & \sigma_{g_{f1,f2}} \\ \sigma_{g_{f2,f1}} & \sigma^2_{g_{f2,f2}} \end{bmatrix} \otimes A$$

If we think in the multi trait model this is very similar but with an additional kroneker product for the multivariate version:

$var(u) = T \otimes E \otimes ... \otimes F \otimes A$

where:

$$\mathbf{var(u)} = \begin{bmatrix} \sigma^2_{g_{t1,t1}} & \sigma_{g_{t1,t2}} \\ \sigma_{g_{t2,t1}} & \sigma^2_{g_{t2,t2}} \end{bmatrix} \otimes \begin{bmatrix} \sigma^2_{g_{e1,e1}} & \sigma_{g_{e1,e2}} & \sigma_{g_{e1,e3}} \\ \sigma_{g_{e2,e1}} & \sigma^2_{g_{e2,e2}} & \sigma_{g_{e2,e3}} \\ \sigma_{g_{e3,e1}} & \sigma_{g_{e3,e2}} & \sigma^2_{g_{e3,e3}} \end{bmatrix} \otimes ... \otimes \begin{bmatrix} \sigma^2_{g_{f1,f1}} & \sigma_{g_{f1,f2}} \\ \sigma_{g_{f2,f1}} & \sigma^2_{g_{f2,f2}} \end{bmatrix} \otimes A$$

Going back to the point. The additional unknown covariance structures besides the multi-trait (T) before where specified with asreml sintax. For example an univariate diagonal and unstructured model, assumed a random effect called 'id' representing the treatments planted in different environments coded in a second random effect called 'env'. The model used to look like:

`fixed= y1~x`

`random= ~ diag(env):id` or `random= ~ us(env):id`

and now it would be specified as:

`fixed= y1~x`

`random= ~ vs(ds(env),id)` or `random= ~ vs(us(env),id)`

where the `ds()` and `us()` functions specifie a diagonal and unstructured models respectively. Now in addition `cs()` for a customized structure is available. The main gain from having changed the formulation is that the new specification trhough the `vs()` function allows to contruct more complex moels. For example, assume individuals specified in a column called 'id' tested in three environments in a column called 'env' measured at two different time points specified in a column called 'time'. We may want something more flexible than:

`fixed= y1~x`

`random= ~ id`

We could actually assume that individuals are correlated within the environments for the different time points but want to consider envrionments indepedent. The variance for such random effects is the following:

$$\mathbf{var}(\mathbf{u}) = \begin{bmatrix} \sigma^2_{g_{e1,e1}} & \sigma_{g_{e1,e2}} & \sigma_{g_{e1,e3}} \\ \sigma_{g_{e2,e1}} & \sigma^2_{g_{e2,e2}} & \sigma_{g_{e2,e3}} \\ \sigma_{g_{e3,e1}} & \sigma_{g_{e3,e2}} & \sigma^2_{g_{e3,e3}} \end{bmatrix} \otimes \begin{bmatrix} \sigma^2_{g_{t1,t1}} & \sigma_{g_{t1,t2}} \\ \sigma_{g_{t2,t1}} & \sigma^2_{g_{t2,t2}} \end{bmatrix} \otimes A$$

which was not possible in previous versions of sommer and now can be specified as:

`random= ~ vs(us(env),ds(time),id)`

and the same logic can be extended to as many factors interacting as desired.

4) The specification of unknown covariance structures in the residuals

Before sommer was limited to only diagonal models in the residuals (unstructured available only for multi-trait before). Now all the same applications discussed for the random term also applies for the residual term. Just keep in mind that the residual term is always called units.

Previous versions:

`random= ~ diag(trait):diag(env):units`

`random= ~ us(trait):diag(env):units` # limit

New versions (>3.7):

`random= ~ vs(ds(env),units, Gtc=mm)` ## can be extended to more interacting factors

`random= ~ vs(us(env),units, Gtc=mm)` ## can be extended to more interacting factors

`random= ~ vs(at(env),units, Gtc=mm)` ## can be extended to more interacting factors

`random= ~ vs(cs(env),units, Gtc=mm)` ## can be extended to more interacting factors

where mm can be any matrix specifying the type of multi-trait model (constraints). For example we could use `unsm()` `diag()`, `uncm()` or any other customized matrix.

5) Special models

In previous version the use of asreml formulation really limited the expansion of sommer to more sophistiated models. Now there's many more possible models

Previous versions:

Overlay models

Previous version: limited to 2 columns and only random and no covariance structures.

`random= ~  x + and(y)`

New versions (>3.7): in theory no limits. Can be extended to more interacting factors in the unknown covariance structures and can overlay as many columns as needed. Plus is fully functional with the multivariate models.

```
random=~ vs(..., overlay(x1,...,xn), Gtc=mm)
```

Random regression models

Previous version: Not available before

New versions (>3.7): in theory no limits. Can be extended to more interacting factors in the unknown covariance structures and only requires the use of the `leg()` function. Plus is fully functional with the multivariate models.

```
random=~ vs(us(leg(v,1)),x)
```

GWAS models

Previous version: Only univariate models available

New versions (>3.7): all the power of the mmer function is available plus you can fit multivariate GWAS models. See details in the sommer.start vignettes.

Spatial models

Previous version: It was called directly in the formula

```
random=~ spl2D(Row,Col,at=?)
```

New versions (>3.7): It has to be called within the `vs()` function but now it can be combined with all the unknown covariance structures available.

`random=~ vs(...,spl2D(Row,Col,at=?), Gtc=mm)` # being mm any multi-trait constraint-structure.

Customized random effects

Previous version: It was provided in the grouping argument

```
random=~ grp(x),
```

```
grouping=list(x=Z)
```

New versions (>3.7): It has to be called within the `vs()` function but now it can be combined with all the unknown covariance structures available.

`random=~vs(,..., Z, Gtc=mm)` # being mm any multi-trait constraint-structure.

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.

Tunnicliffe W. 1989. On the use of marginal likelihood in time series model estimation. JRSS 51(1):15-27.