Tests on multivariate REML using dmm().

Neville Jackson

18 Aug 2021

0.1 Introduction

We want to test multivariate REML estimation using dmeopt="fgls" in dmm(). The code for multivariate *fgls* in *dmm()* uses the conventional approach to multivariate analysis of stacking all traits in succesive blocks in the data vector so that the analysis treats the data vector as if it were all one trait. This requires appropriate blocking of the associated matrices. In the case of "fgls" on the dyadic model, this is not simple; the dyadic model errors have a covariance structure which is specified by a commutation matrix combined with the covriance matrix of fixed model residuals.

0.2 Methods

Make up a small dataset, 4 traits with various levels of correlation between the traits.

```
> wxyz.df
  Id y x   w z
1  1 1 2 2.5 3
2  2 2 3 2.7 2
3  3 3 2 0.6 3
4  4 2 3 2.1 2

> cor(wxyz.df)
           Id          y          x          w          z
Id  1.0000000  0.6324555  0.4472136 -0.4484507 -0.4472136
y   0.6324555  1.0000000  0.0000000 -0.8164966  0.0000000
x   0.4472136  0.0000000  1.0000000  0.5165766 -1.0000000
w  -0.4484507 -0.8164966  0.5165766  1.0000000 -0.5165766
z  -0.4472136  0.0000000 -1.0000000 -0.5165766  1.0000000
```

We can use this probe the multivariate code for sanity.

0.3 Results

The data has no genetic variance component, only VarE(I).

0.3.1 Two uncorrelated traits

Traits y and x have zero correlation.

Solve DME's by OLS

```
> junk <- dmm(wxyz.df,I(cbind(x,y)) ~ 1, components=c("VarE(I)"))
Dyadic mixed model fit for datafile: wxyz.df
```

```
  ....
> summary(junk)
Call:
summary.dmm(object = junk)

Coefficients fitted by OLS for fixed effects:

  Trait Estimate StdErr CI95lo CI95hi
1     x      2.5  0.289   1.93   3.07

  Trait Estimate StdErr CI95lo CI95hi
1     y        2  0.408    1.2    2.8


Components partitioned by DME from residual var/covariance after OLS-fixed-effects fit:

        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       x:x    0.333  0.122 0.0948  0.572

        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       x:y -1.6e-17  0.211 -0.413  0.413

        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       y:x -1.6e-17  0.211 -0.413  0.413

        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       y:y    0.667  0.243   0.19   1.14

>
```

So OLS correctly finds

```
> var(wxyz.df$x)
[1] 0.3333333
> var(wxyz.df$y)
[1] 0.6666667
```

because these aer the variance estimates assuming zero correlation of x with y.

Solve DME's by feasable GLS - multivariate case

```
junk <- dmm(wxyz.df,I(cbind(x,y))  ~ 1, components=c("VarE(I)"),dmeopt="fgls")
..............
        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       x:x      0.5  0.353 -0.192   1.19

        Traitpair Estimate   StdErr    CI95lo   CI95hi
```

```
VarE(I)       x:y 3.22e-05 5.35e-05 -7.26e-05 0.000137

        Traitpair Estimate   StdErr    CI95lo   CI95hi
VarE(I)       y:x 3.22e-05 5.35e-05 -7.26e-05 0.000137

        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       y:y 4.56e-09  0.353 -0.692  0.692
```

So all the variance goes to the first trait (x). Is that right? Well lets look at the variances of y and x separately, and then combined

```
attach(wxyz.df)
> var(y)
[1] 0.6666667
> var(x)
[1] 0.3333333
> var(c(x,y))
[1] 0.5
```

So if we "stack" x and y we get the figure estimated by REML. But why did REML assign all the "stacked" variance to x rather than to y? I dont know. This does not seem correct !!

Solve DME's by feasable GLS - univariate case

If we model single traits with "fgls" fit we get

```
> junk <- dmm(wxyz.df,x ~ 1, components=c("VarE(I)"),dmeopt="fgls")
Dyadic mixed model fit for datafile: wxyz.df
  ....
> summary(junk)
Call:
summary.dmm(object = junk)

Coefficients fitted by OLS for fixed effects:

  Trait Estimate StdErr CI95lo CI95hi
1     x      2.5  0.289   1.93   3.07


Components partitioned by DME from residual var/covariance after OLS-fixed-effects fit:

        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       x:x    0.333  0.272   -0.2  0.867

>
```

which is correct and the same as obtained with OLS. OLS will always give the same estimates whether multi-trait or single-trait, whereas FGLS will give different estimates for multitrait vs univariate. That is because FGLS uses the entire multi-trait matrix of covariance of residuals so it adjusts for correlated errors within a trait and for correlated errors across traits. Maybe that is the problem... maybe FGLS should only adjust for within trait error covariances?

0.3.2 Swap the order of traits

One test of blocking is to put traits in reverse order

```
junk <- dmm(wxyz.df,I(cbind(y,x))  ~ 1, components=c("VarE(I)"),dmeopt="fgls")
..................
        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       y:y 4.56e-09  0.353 -0.692  0.692

        Traitpair Estimate   StdErr    CI95lo   CI95hi
VarE(I)       y:x 3.22e-05 5.35e-05 -7.26e-05 0.000137

        Traitpair Estimate   StdErr    CI95lo   CI95hi
VarE(I)       x:y 3.22e-05 5.35e-05 -7.26e-05 0.000137

        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       x:x      0.5  0.353 -0.192   1.19
```

Yes, the estimates come out in reverse order, and are the same. So the blocking passes this test. The stacked variance of both traits is being assigned to the trait with the largest variance.

0.3.3 Correlated traits

We can look at $r = 1$ by putting the same trait in twice

```
junk <- dmm(wxyz.df,I(cbind(x,x))  ~ 1, components=c("VarE(I)"),dmeopt="fgls")
......................
        Traitpair Estimate   StdErr    CI95lo   CI95hi
VarE(I)       x:x 1.72e-05 0.000192 -0.000358 0.000393

        Traitpair Estimate   StdErr    CI95lo   CI95hi
VarE(I)       x:x 1.72e-05 0.000192 -0.000358 0.000393

        Traitpair Estimate   StdErr    CI95lo   CI95hi
VarE(I)       x:x 1.72e-05 0.000192 -0.000358 0.000393

        Traitpair Estimate   StdErr    CI95lo   CI95hi
VarE(I)       x:x 1.72e-05 0.000192 -0.000358 0.000393
```

All the estimates are the same, which is correct, but why are they effectively zero. The "stacked" variance is not zero but it is less than the univariate variance of x

```
> var(c(x,x))
[1] 0.2857143
```

Perhaps we can explain the above of we look at other correlations. Try $r = -1$

```
junk <- dmm(wxyz.df,I(cbind(x,z))  ~ 1, components=c("VarE(I)"),dmeopt="fgls")
................
        Traitpair Estimate   StdErr   CI95lo  CI95hi
VarE(I)       x:x 0.000107 0.000754 -0.00137 0.00158

        Traitpair  Estimate   StdErr   CI95lo  CI95hi
VarE(I)       x:z -0.000107 0.000754 -0.00158 0.00137

        Traitpair  Estimate   StdErr   CI95lo  CI95hi
VarE(I)       z:x -0.000107 0.000754 -0.00158 0.00137

        Traitpair Estimate   StdErr   CI95lo  CI95hi
VarE(I)       z:z 0.000107 0.000754 -0.00137 0.00158

> var(c(x,z))
[1] 0.2857143
```

So the components do correspond to a -1 correlation, but I expected stacking two negatively correlated traits would increase the variance. Pooling negatively correlated things increases the information content more than double!

Finally lets try $r = 0.5$

```
junk <- dmm(wxyz.df,I(cbind(x,w))  ~ 1, components=c("VarE(I)"),dmeopt="fgls")
............
        Traitpair Estimate StdErr CI95lo CI95hi
VarE(I)       x:x 3.56e-08  0.133  -0.26   0.26

        Traitpair Estimate   StdErr    CI95lo   CI95hi
VarE(I)       x:w 7.95e-05 9.98e-05 -0.000116 0.000275

        Traitpair Estimate   StdErr    CI95lo   CI95hi
VarE(I)       w:x 7.95e-05 9.98e-05 -0.000116 0.000275

        Traitpair Estimate StdErr  CI95lo CI95hi
VarE(I)       w:w    0.187  0.133 -0.0735  0.449

> var(c(x,w))
[1] 0.6083929
```

So we have some variance (.187) and it seems to have arbitrarily assigned it to w, but it is not as large as the "stacked" variance (.608). That seems OK, I would expect positively correlated pooling to achieve less than twice the information.

0.4 Conclusion

Some of these results look dramatically wrong. I am not happy with the somewhat arbitrary allocation of variance to one trait, and I am not happy with the magnitude of the multivariate variances.