JSS Referee Report: Kenward-Roger Approximation and Parametric Bootstrap Methods

General comments

1. The article presents its subject very clearly and is well written. The presentation is well developed with good illustative data examples that are used to motivate and explain the methodology. The detailed matrix examples using this data are very effective.

2. The package seems to have problems with stability. I used it a month ago and it worked properly in the sense that it gave results consistent with those in the article. To prepare a final version of this report, I re-installed the package today (August 15, 2012), Version: 0.3-1, Built: R 2.15.1; ; 2012-08-13 18:52:18 UTC; windows, and the examples applying 'KRmodcomp' to the 'beets' data set, gave results very different from those in the article, e.g.

```
> beet0 <- lmer( sugpct ~ block + sow + harvest + (1|block:harvest) , data = beets, REML = FALSE)
> beet_no.harv <- update(beet0, . ~ . - harvest)
> beet_no.sow  <- update(beet0, . ~ . - sow)
> (kr.h <- KRmodcomp(beet0, beet_no.harv))
F-test with Kenward-Roger approximation; computing time: 0.18 sec.
Large : sugpct ~ block + sow + harvest + (1 | block:harvest)
small : sugpct ~ block + sow + (1 | block:harvest)
Fstat df1      df2 p.value     F.scaling
 0   1 2.017977       1 -1.869422e-12
The scaling factor for the F-statistic is smaller than 0.2
The unscaled statistic might be more reliable
Results fromm the unscaled F-statistic
df1=  1, df2=    2.02, FstatU=   15.21, pvalU=     NA
Warning messages:
1: In KRmodcomp.mer(beet0, beet_no.harv) :
largeModel has been refitted with REML=TRUE
2: In mer_finalize(ans) : false convergence (8)
> (kr.s <- KRmodcomp(beet0, beet_no.sow))
F-test with Kenward-Roger approximation; computing time: 0.07 sec.
Large : sugpct ~ block + sow + harvest + (1 | block:harvest)
small : sugpct ~ block + harvest + (1 | block:harvest)
Fstat df1 df2 p.value F.scaling
101.0006   4  20       0         1
Warning messages:
1: In KRmodcomp.mer(beet0, beet_no.sow) :
largeModel has been refitted with REML=TRUE
2: In mer_finalize(ans) : false convergence (8)
```

3. I commend the authors for using the 'parallel' package for parallel generation of bootstrap samples. Note that the current help file for 'PBmodcomp' was written for the 'snow' package. The example showing a call to 'makeSOCKcluster' does not work with the 'parallel' package.

4. Since the package extends the 'lme4' package it might be a good idea to clarify the scope of the models to which pbkrtest applies. It seems to be intended only for linear models with a gaussian response. I have been able to invoke 'PBmodcomp' on a binomial model without producing an error message although I don't think that the simulated null values were drawn from a binomial. It would be desirable to make this more clear both in the article and in error or warning messages in the package.

5. The last sentence of the second paragraph following equation (1) on p. 2 reads “Notice that the structural forms of the random components of the two models are mes[sic] identical”. I presume that 'mes' is a typographical error but what I find interesting here is that there appears to be no need for the random structures to be identical. For K&R, it is only the random structure of the large model that matters. The role of the smaller model is merely to provide the fixed effects model subspace from which the L matrix can be obtained. In contrast, for the PB test, it is only the random structure of the smaller model that matters.

6. There are very few typographical errors and relatively few questions concerning usage and grammar. A professional copy editor would correct these much better than I could. On request I would be happy to forward those I believe I have found. The only 'error' I have found that could be missed by a copy editor follows Appendix A.1 equation (22):

   “elements of the lower triangular Γ_i”

   should read:

   “elements of the lower triangle of Γ_i”

7. The following are further comments that are merely suggestions for future consideration.

General Linear Hypotheses

Although KRmodcomp computes a linear hypothesis matrix to compute a Wald test, it is not possible to specify the hypothesis in the form:

$$L\beta = 0$$

or, more generally,

$$L\beta = c$$

i.e. using the 'restriction matrix' directly without requiring the user to specify and fit the null model.

This would be a very easy and, in my opinion a desirable, enhancement.

It would also be easy to allow for L matrices that are not of full rank by performing the overall test by transforming L to an equivalent semi-orthogonal hypothesis matrix whose rows span the same space as that spanned by the rows of L. The packages has all the tools are in place to do this easily. The default could be simply testing the hypotheses with $L = I$, i.e. producing standard regression output with K&R estimates for denominator degrees of freedom for each single-degree-of-freedom effect in the model. This would be highly welcome by many users – perhaps too welcome!

The output could produce tests for the hypotheses represented by each row of L, each with the appropriate degrees of freedom, as well as the joint hypothesis that all row hypotheses are satisfied.

This output would be very useful in estimating, for example, a full set of pairwise differences among levels of a categorical factor.

Algorithms

I am interested in the numerical properties of these algorithms. I believe that it would be preferable if regression packages such as 'lme4' provided an 'sd' version of 'vcov' that returned something like the eigenvectors and the square roots of the eigenvalues (or the eigenvalues themselves) of the 'vcov' matrix – provided these elements are produced in the course of the fitting algorithm. For example, greater numerical accuracy is achieved in ordinary-least-squares computation by working with the SVD decomposition of $X = UDV'$ instead of computing and then working with $(X'X)^{-1}$. If we had infinite precision, the hypothesis $L\beta = 0$ can be tested with a Wald-style test statistic:

$$W = (L\hat{\beta})'(L(X'X)^{-1}L')^{-1}L\hat{\beta}/\hat{\sigma}^2$$

If D and V were calculated with the full precision available using the SVD of X – not by using the spectral decomposition of $ $X'X = VD^2V'$ which cannot recover the accuracy lost in $X'X$ – a numerically superior test would use the SVD of $D^{-1}V'L' = P\Delta Q'$ where Δ and Q are square assuming that L is of full row rank. A Wald-style test statistic has the form:

$$W = ||P'\Delta V'\hat{\beta}||^2/\hat{\sigma}^2$$

Note that similar results can be obtained with QR decompositions.

In the context of mixed models, if one had accurate versions of V and Λ in the spectral decomposition of $\hat{V}(\hat{\beta}) = V\Lambda V'$, then the similar Wald-style test statistic can be obtained with the SVD decompositon of $\Lambda^{1/2}V'L' = P\Delta Q'$:

$$W = ||P'\Delta V'\hat{\beta}||^2$$

Unfortunately (almost all?) packages, lme4 among them, return only the 'vcov' matrix as a matrix and the greater accuracy from direct estimates of its spectral decomposition are not available. The question that comes to mind is, given that we only have 'vcov', can we do better than:

$$W = (L\hat{\beta})'(L\hat{V}(\hat{\beta})L')^{-1}L\hat{\beta}$$

I have worked on simulating various alternatives without clear results in general. The only things that is clear is that it is much bettter to code this in R as:

```
t(L %*% beta.hat) %*% solve( L %*% Vcov %*% t(L),  L %*% beta.hat)
```

instead of:

```
t(L %*% beta.hat) %*% solve( L %*% Vcov %*% t(L)) %*%  (L %*% beta.hat)
```

The code in 'pbkrtest' correctly uses the former approach.

In conclusion, I cannot suggest any improvement on the numerical approach currently used in the package. However, I think it would be very interesting to investigate the possibility of improvements in the future.

An additional note is that, as long as L is only obtained as a restriction matrix from two models, strong collinearity in the rows of L – as might occur if L were supplied directly by the user – is not a concern. The issue becomes more salient if L can be supplied as an argument.