Using gemmR

Jeffrey S. Chrabaszcz and Joe W. Tidwell

September 9, 2014

Motivation

In the social sciences, we often ask locational questions, such as:

- Do people who train on certain computer tasks have *higher* cognitive ability? [4]
- Are there *more* murders per capita in more honor-focused cultures? [3]
- Does the native language *change* acquisition of definite articles in a second language? [1]

These questions make no mention of the specific distance between relative groups and instead focus on the order of outcome magnitudes. While the statistics applied to these questions are usually variants of the general linear model, there is no reason to impose the assumption of linearity on the reality underlying these tests. One alternative is to apply the general monotone model (`GeMM`) as proposed by Dougherty & Thomas, 2012 [2].

`GeMM` uses a search and scale procedure to find the optimal relative weights for a set of predictors and scale these weights to minimize the order-constrained squared error. This first, computationally-intensive step is accomplished by using a genetic algorithm to optimize some fit criterion (e.g., Kendall's τ) between an observed outcome and a weighted set of predictors. Use of τ in this case assures relative weights that maximally reflect the monotone relationship between the outcome and model predictions. Other fit criteria penalize for complexity, but are based on transformations of τ. We then regress the original outcome onto the relative-weighted model predictions to compute an intercept and scaling factor that minimizes squared error conditioned on this ordinal constraint.

Fitting a `gemm` model

We implement `GeMM` with the `gemmR` package, which uses `Rcpp` to speed up repeated calculation of Kendall's τ for use in the genetic search process. As `GeMM` serves as a functional replacement for the linear model, a similar syntax is used to fit a `GeMM` model.

```r
library(gemmR)
data(culture)
mod <- gemm(murder.rate ~ pasture + gini + gnp, data = culture)
```

This produces a `gemm` object, which is modeled after the `lm` object.

Helper functions

The `gemmR` package includes a number of S3 methods and a few novel functions to help extract information from `gemm` objects.

Summary

`summary` displays some helpful information about the fitted `gemm` object.

```
summary(mod)

## Call:
## gemm.formula(formula = murder.rate ~ pasture + gini + gnp, data = culture)
##
## Coefficients:
##      intercept pasture   gini        gnp
## [1,]    0.4286  0.2184 0.2500 -0.0001911
## [2,]    0.3280  0.2063 0.2524 -0.0001903
## [3,]    0.4391  0.2102 0.2499 -0.0001912
## [4,]    0.4263  0.2183 0.2501 -0.0001911
##
## bic
## [1] -41.64 -41.64 -41.64 -41.64
```

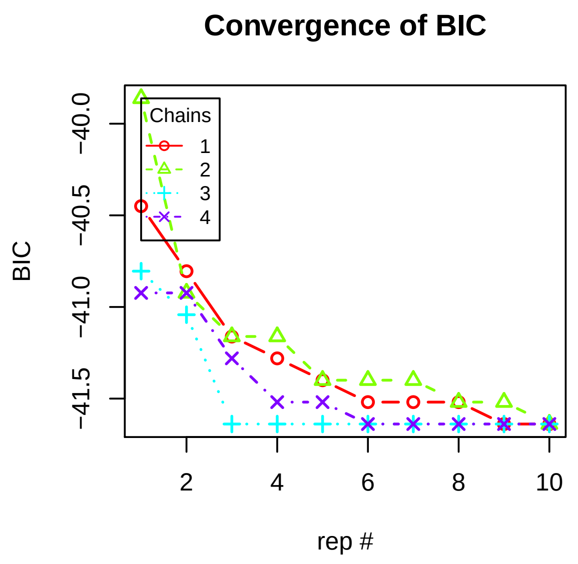
`GeMM` is a stochastic process, so multiple replications are advisable to ensure stability of parameter estimates. `gemm` runs four replications by default, all of which are displayed by descending value on the fit criterion.

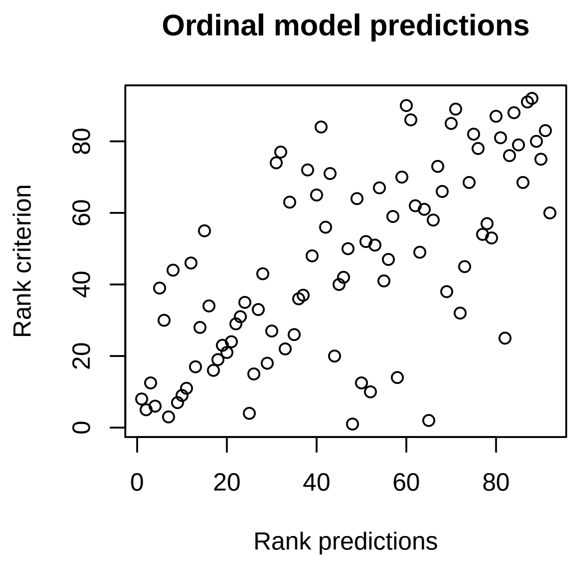
Below the four chains are the corresponding values of the optimized fit criterion. While all fit criteria are calculated and contained in the `gemm` object, only the criterion used for selection is displayed with `summary`.

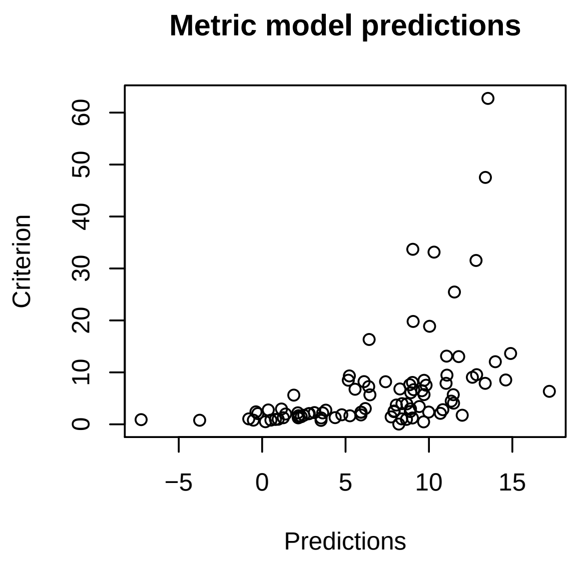
Diagnostics

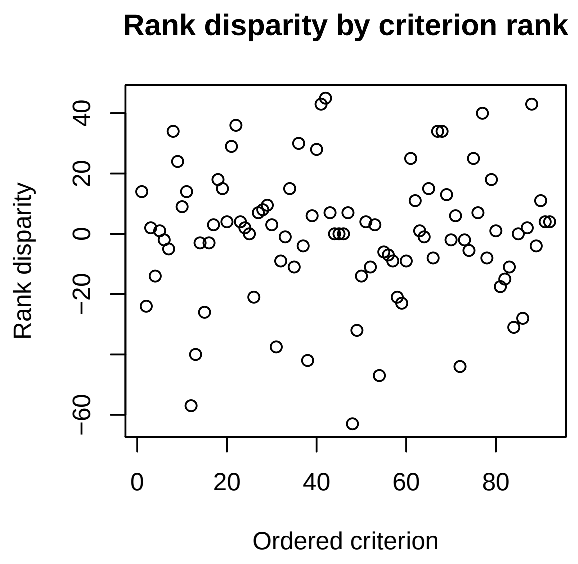
Though no method exists for verifying that results of a random search process on empirical data, one quick way to check the suitability of a solution is to demonstrate convergent results across starting conditions. A quick way to check genetic algorithm performance for a given dataset is to plot the best criterion value across generations and chains.

```
check.mod <- gemm(murder.rate ~ pasture + gini + gnp, data = culture, check.convergence = TRUE)
plot(check.mod)
```









Predict

The `predict` function for `gemm` serves two roles. The first is to generate model predictions based on the best chain of a given model. `predict` will also generate the counts of concordances, disconcordances, outcome ties and predictor ties for a given model.

```
yhat <- predict(mod, tie.struct = TRUE)
head(yhat)

##       [,1]
## 1   6.4641
## 2   7.7362
## 3 11.0760
## 4  8.8805
## 5  2.1789
## 6  0.5268
```

```
attr(yhat, "tie.struct")
```

```
##   correct incorrect cue.tie crit.tie
## 1    3118      1066       0        2
```

A note on information criteria in `gemmR`

The information criteria calculated by `gemmR` are based on ordinal statistics and cannot be directly compared with likelihood-based criteria. `gemmR` includes a number of methods so that traditional information criteria can be easily extracted for comparison with other models.

```
logLik(mod)
```

```
## 'log Lik.' -330.7 (df=1)
```

```
AIC(mod)
```

```
## [1] 663.4
```

```
BIC(mod)
```

```
## [1] 665.9
```

References

[1] Anna Chrabaszcz and Nan Jiang. The role of the native language in the use of the english nongeneric definite article by l2 learners: A cross-linguistic comparison. *Second Language Research*, 30(3):351–379, 2014.

[2] Michael R Dougherty and Rick P Thomas. Robust decision making in a nonlinear world. *Psychological review*, 119(2):321, 2012.

[3] Michael R Dougherty, Rick P Thomas, Ryan P Brown, Jeffrey S Chrabaszcz, and Joe W Tidwell. An introduction to the general monotone model with application to two problematic datasets. Sociological Methods.

[4] Joe W Tidwell, Michael R Dougherty, Jeffrey R Chrabaszcz, Rick P Thomas, and Jorge L Mendoza. What counts as evidence for working memory training? problems with correlated gains and dichotomization. *Psychonomic bulletin & review*, 21(3):620–628, 2014.