Application Tutorial: OmicKriging

Keston Aquino-Michaels, Heather E. Wheeler, Vassily V. Trubetskoy and Hae Kyung Im

June 17, 2014

Method citation: Wheeler HE, Aquino-Michaels K, Gamazon ER, Trubetskoy VV, Dolan ME, Huang RS, Cox NJ, Im HK. Poly-Omic Prediction of Complex Traits: OmicKriging. Genet Epidemiol. 2014 May 2. doi: 10.1002/gepi.21808.

---

1 Running OmicKriging with Example Data

To install from CRAN:

```
> install.packages("OmicKriging")
```

Start by loading OmicKriging functions into R:

```
> library(OmicKriging)
```

Define paths to the genetic relatedness (GCTA binary format), gene expression, and phenotype data files (paths may differ based on where the files are located). The path.package() function returns the package installation directory. These files will later be passed to upcoming functions:

```
> library(OmicKriging)
> binaryFile <- system.file(package = "OmicKriging",
+                 "doc/vignette_data/ig_genotypes.grm.bin")
> binaryFileBase <- substr(binaryFile,1, nchar(binaryFile) - 4)
> expressionFile <- system.file(package = "OmicKriging",
+                     "doc/vignette_data/ig_gene_subset.txt.gz")
> phenotypeFile <- system.file(package = "OmicKriging",
+                   "doc/vignette_data/ig_pheno.txt")
```

Load the phenotype data into R:

```
> pheno <- read.table(phenotypeFile, header = T)
```

Load a pre-computed GCTA GRM into R:

```
> grmMat <- read_GRMBin(binaryFileBase)
```

By default, grmFilePrefix is set to NULL. However, if specified, this function will save the computed GRM to disk in GCTA binary format. Additionally by default both snpList and sampleList are set to NULL. However you may restrict the GRM calculation by specifying a vector of sample IDs or a vector of SNP IDs.

Load and calculate a gene expression relatedness matrix (GXM) with the following function:

```
> gxmMat <- make_GXM(expFile = expressionFile)
```

Similarly, by default, gxmFilePrefix is set to NULL, however if specified, this function will save the computed GXM to disk in GCTA binary format.

Additional convenience functions are included to perform principal components analysis (PCA):

```
> pcMatXM <- make_PCs_irlba(gxmMat, n.top = 10)
> pcMatGM <- make_PCs_irlba(grmMat, n.top = 10)
> pcMat <- cbind(pcMatGM, pcMatXM[match(rownames(pcMatGM), rownames(pcMatXM)),])
```

The following convenience function allows the user to perform n-fold cross-validation. Specify the number of cores you wish to use (default = "all"), the number of cross-validation folds desired (default = 10), covariates (by default covar.mat = NULL), the phenotype object, pheno.id (by default = 1 (the first phenotype in the file)), the h2 vector and a list of the correlation matrices to be included.

Note: The sum of the h2 vector must be between 0 and 1. In this example, we will give each matrix equal weight.

```
> result <- krigr_cross_validation(pheno.df = pheno,
+         cor.list = list(grmMat, gxmMat),
+         h2.vec = c(0.5, 0.5),
+         covar.mat = pcMat,
+         ncore = 2,
+         nfold = "LOOCV")

Detected 99 samples...
Set leave-one-out cross-validation...
With 2 logical core(s)...
Running OmicKriging...

Call:
lm(formula = Ytest ~ Ypred, data = res)

Residuals:
     Min       1Q   Median       3Q      Max
-2.54750 -0.66269  0.09617  0.74782  2.25937
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.001745   0.097916  -0.018   0.9858
Ypred        0.087819   0.044362   1.980   0.0506 .
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

Residual standard error: 0.9742 on 97 degrees of freedom
Multiple R-squared:  0.03883,        Adjusted R-squared:  0.02892
F-statistic: 3.919 on 1 and 97 DF,  p-value: 0.05058

Finished OmicKriging in 1.866 seconds
```

This function will return a data.frame with column Ypred corresponding to the predicted values and column Ytest corresponding to the measured phenotypes.

Congratulations!
You have just completed the OmicKriging tutorial!

---

2 Cleanup

This is a cleanup step for the vignette on Windows; typically not needed for users.

```
> allCon <- showConnections()
> socketCon <- as.integer(rownames(allCon)[allCon[, "class"] == "sockconn"])
> sapply(socketCon, function(ii) close.connection(getConnection(ii)) )
```