The RepeatABEL package - a tutorial

Lars Ronnegard

May 7, 2015

Introduction

This vignette gives an introduction to the RepeatABEL package. The package performs GWAS for data where there are repeated observations on individuals. Various random effects, including polygenic effects, can be fitted, as well as spatial effects. In a first step a model without the fixed SNP effect is fitted using the `hglm` package for estimating variance components (see *Ronnegard, Shen & Alam (2010) hglm: A package for fitting hierarchical generalized linear models. The R Journal 2:20-28*). These estimates are subsequently used in the GWAS where each marker is fitted one at a time. Thus, this is similar to using the `polygenic_hglm` function and subsequently the `mmscore` function in GenABEL. The package consists of three main functions `rGLS`, `preFitModel` and `simulate_PhenData`.

Theory for the `rGLS` function

The `rGLS` function is the main function of the package and by default fits a linear mixed model including random polygenic effects $\boldsymbol{g}$ and permanent environmental effects $\boldsymbol{p}$ in

$$\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{g} + \boldsymbol{Zp} + \boldsymbol{e} \qquad (1)$$

where $\boldsymbol{y}$ is the studed trait, $\boldsymbol{\beta}$ are fixed effects, $\boldsymbol{e}$ are residuals with $\boldsymbol{e} \sim N(0, \sigma_e^2)$ having residual variance σ_e^2. Furthermore, $\boldsymbol{X}$ and $\boldsymbol{Z}$ are model matrices. The random effects are assumed multivariate normal such that $\boldsymbol{p} \sim N(0, \boldsymbol{I}\sigma_p^2)$ and $\boldsymbol{g} \sim N(0, \boldsymbol{G}\sigma_g^2)$ where $\boldsymbol{G}$ is the genomic relationship matrix. Thus the estimated (co)variance matrix for this model is

$$\hat{\boldsymbol{V}} = \boldsymbol{G}\hat{\sigma}_g^2 + \boldsymbol{ZZ}^T\hat{\sigma}_p^2 + \boldsymbol{I}\hat{\sigma}_e^2. \qquad (2)$$

Subsequently, a linear model is fitted (using generalized least squares, GLS) for each marker where the covariate x_{SNP} is coded as 0,1,2:

$$\boldsymbol{y} = \boldsymbol{X\beta} + x_{SNP}\beta + \boldsymbol{\varepsilon} \qquad (3)$$

with

$$\boldsymbol{\varepsilon} \sim N(0, \sigma_\epsilon^2 \hat{\boldsymbol{V}}). \qquad (4)$$

A Wald test is used to compute the P value for SNP effect β.

The computations are made fast by applying a eigen-decomposition of $\hat{V}$ and using the built-in `qr` function in R to fit the linear models (3).

Using the `rGLS` function

The following example illustrates the use of the function. There are two data objects to be included in the input: a GenABEL object including the genotypic information and a data frame including the phenotypic information. The name of the ID variable in the phenotype data frame should be "id" (otherwise specify `id.name` equal to the ID variable name).

In this example there are 360 observations from 100 individuals, and there are 5792 SNP to be tested.

```
> library(RepeatABEL)
> data(gen.data) #GenABEL object including IDs and marker genotypes
> data(Phen.Data) #Phenotype data with repeated observations
```

The data frame `Phen.Data` includes the trait value y, two covariates (age and sex) and the ID of the individuals. We wish to include age and sex as fixed effects so the function input is

```
> GWAS1 <- rGLS(y ~ age + sex, genabel.data = gen.data,
    phenotype.data = Phen.Data)
[1] "GRM ready"
[1] "Variance component estimation ready"
[1] "Rotation matrix ready"
[1] "Rotate LMM started"
[1] "Rotate LMM ready"
```

The computations in this function consists of four parts: construction of the GRM, variance component estimation using the `hglm` function, rotating the GLS using eigen-decomposition transforming it to an ordinary least squares (OLS) problem, and finally fitting an OLS for each SNP. How far the computations have come is shown in the output.

The class of the output object GWAS1 is `gwaa.scan`, so we can apply the generic functions `summary()` and `plot()` defined by the GenABEL package.

```
> summary(GWAS1)
Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2       effB
rs120315           1  3725352      +  T  C  2.1696316
rs9670687          1  4779800      -  C  A -2.0412127
rs891586           2  8024318      +  A  G -1.6669267
rs1352451          2  8022651      -  A  C  1.6169429
rs1252282          2  8167984      -  A  T -0.9982089
rs9922492          1  3729983      +  C  T -1.8586037
rs7633966          1  3721952      +  C  T  1.8586037
rs4378234          1  4800348      -  G  T -1.9012073
rs7499832          3 10242953      -  G  C -1.0897248
rs6561272          1  3715538      -  T  G -1.9073305
          chi2.1df         P1df Pc1df
rs120315        NA 3.501204e-07    NA
rs9670687       NA 1.366120e-04    NA
rs891586        NA 1.975997e-04    NA
rs1352451       NA 3.959643e-04    NA
rs1252282       NA 4.109058e-04    NA
rs9922492       NA 4.832322e-04    NA
rs7633966       NA 4.832322e-04    NA
rs4378234       NA 5.588239e-04    NA
```

```
rs7499832        NA 6.167705e-04    NA
rs6561272        NA 8.656541e-04    NA
```

Using the preFitModel function

The function `preFitModel` is used for variance component estimation and increases the flexibility of modeling in the `rGLS` function.

Fitting the same model as above

To start with we have a look at how the model in the previous section can be fitted in two steps using the `preFitModel` function and thereafter the `rGLS` function. Note that the results are exactly the same as in the previous section.

```
> #The same results can be computed using the preFitModel as follows
> fixed=y ~ age + sex
> Mod1 <- preFitModel(fixed, random=~1|id, genabel.data = gen.data,
    phenotype.data = Phen.Data, corStruc=list( id=list("GRM","Ind") ))
[1] "GRM ready"
> GWAS1b <- rGLS(fixed, genabel.data = gen.data,
    phenotype.data = Phen.Data, V = Mod1$V)
[1] "Rotation matrix ready"
[1] "Rotate LMM started"
[1] "Rotate LMM ready"
> summary(GWAS1b)
Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2       effB
rs120315           1  3725352      +  T  C  2.1696316
rs9670687          1  4779800      -  C  A -2.0412127
rs891586           2  8024318      +  A  G -1.6669267
rs1352451          2  8022651      -  A  C  1.6169429
rs1252282          2  8167984      -  A  T -0.9982089
rs7633966          1  3721952      +  C  T  1.8586037
rs9922492          1  3729983      +  C  T -1.8586037
rs4378234          1  4800348      -  G  T -1.9012073
rs7499832          3 10242953      -  G  C -1.0897248
rs6561272          1  3715538      -  T  G -1.9073305
          chi2.1df         P1df Pc1df
rs120315        NA 3.501204e-07    NA
rs9670687       NA 1.366120e-04    NA
rs891586        NA 1.975997e-04    NA
rs1352451       NA 3.959643e-04    NA
rs1252282       NA 4.109058e-04    NA
rs7633966       NA 4.832322e-04    NA
rs9922492       NA 4.832322e-04    NA
rs4378234       NA 5.588239e-04    NA
rs7499832       NA 6.167705e-04    NA
rs6561272       NA 8.656541e-04    NA
```

The only information transferred from the `preFitModel` function to `rGLS` is the estimated (co)variance matrix `Mod1$V`. The `corStruc` option specifies the correlation structure to be applied on each random effect. In the example we wish to fit polygenic effects and permanent environmental effects. The former requires a correlatioon structure given by the GRM whereas the permanent environmental effects are iid. Consequently, `corStruc=list( id=list("GRM","Ind") ))` is specfied.

A model having several different random effects

In this (fake) example there are 60 observations on each nest and there are 6 different nests. We wish to include these as random effect too, and to start with they are modelled as indendent random effects.

```
> # In this example there are 6 nests and 60 observations per nest
> Phen.Data$nest <- rep(1:6, each=60)
> # A model including polygenic effects, permanent environmental effects,
> # and nest effect as random
> Mod2 <- preFitModel(fixed, random=~1|id + 1|nest,
    genabel.data = gen.data, phenotype.data = Phen.Data,
    corStruc=list( id=list("GRM","Ind") , nest=list("Ind")) )
> GWAS2 <- rGLS(fixed, genabel.data = gen.data,
    phenotype.data = Phen.Data, V = Mod2$V)
```

Spatial modelling

This example shows how random effects having a spatial correlation structure can be included to account for populaton structure. The spatial model used is a Conditional AutoRegressive (CAR) model and the input spatial information is given by a neighborhood matrix. (A neighborhood matrix has a non-zero value for an element (i,j) where the subject (nest in our example) i and j come from neighboring locations. The diagonal elements are zero.) Here the neighborhood matrix (`D`) is a 6×6 matrix

```
> D= matrix(0,6,6)
> D[1,2] = D[2,1] = 1
> D[5,6] = D[6,5] = 1
> D[2,4] = D[4,2] = 1
> D[3,5] = D[5,3] = 1
> D[1,6] = D[6,1] = 1
> D[3,4] = D[4,3] = 1
```

where for instance nests 1 and 2 are defined as neighbors. The model including polygenic effects, permanent environmental effects and a spatial correlation between nests is then fitted as

```
> Mod3 <- preFitModel(y ~ age + sex, random=~1|id + 1|nest,
    genabel.data = gen.data, phenotype.data = Phen.Data,
    corStruc=list( id=list("GRM","Ind") ,
    nest=list("CAR")), Neighbor.Matrix=D )
> GWAS2b <- rGLS(fixed, genabel.data = gen.data,
    phenotype.data = Phen.Data, V = Mod3$V)
```

Using the simulate_PhenData function

The third function included in the RepeatABEL package is a simulation function where one can simulate phenotypic data having repeated observations. The input genotype information is given by a GenABEL object.

Suppose for instance we want to simulate 4 observations from each individual and the three variance components (polygenic, permanent env., residual) are 1.

```
> VC.poly <- VC.perm <- VC.res <- 1
> n.obs <- rep(4, nids(gen.data))
```

In this example, the GenABEL object `gen.data` is used as input and an additive genetic effect of 2.0 is simulated at the location of SNP number 1000. The phenotype data is then simulated as

```
> Phen.Sim <- simulate_PhenData(y ~ 1, genabel.data=gen.data,
    n.obs=n.obs, SNP.eff=2, SNP.nr=1000,
    VC=c(VC.poly,VC.perm,VC.res))
```

The simulated data can then be fitted as

```
> GWAS.sim1 <- rGLS(y ~ 1, genabel.data = gen.data,
    phenotype.data = Phen.Sim)
```

The data set `gen.data` includes information on sex, so we can also add a fixed sex effect to our simulations. Here a sex effect of 1.0 is simulated

```
> Phen.Sim <- simulate_PhenData(y ~ sex,
    genabel.data=gen.data, n.obs=n.obs, SNP.eff=2,
    SNP.nr=1000, VC=c(VC.poly,VC.perm,VC.res), beta=c(0,1))
```

where `beta` is a vector specifying the simulated values of the fixed effects and since we fit an intercept and a sex effect the length of `beta` is two. The data can then be fitted as

```
> GWAS.sim1 <- rGLS(y ~ sex, genabel.data = gen.data,
    phenotype.data = Phen.Sim)
> plot(GWAS.sim1, main="Simulation results")
```

The produced Manhattan plot is given below