Package Ravages (RAre Variant Analysis and GEnetic Simulation)

Herve Perdry and Ozvan Bocher

2018-07-19

```
library("mlogit")
library("knitr")
require("Ravages")
```

Introduction

Ravages can be used for burden tests, a type of rare variant association tests and genetics data simulation. Ravages relies on the package Gaston developped by Herve Perdry and Claire Dandine-Roulland. Most functions are written in C++ thanks to the packages Rcpp, RcppParallel and RcppEigen. Functions of this package use bed.matrix to manipulate genetic data as in the package Gaston (see documentation of this package for more details). In this vignette, we show how to simulate genetics data and we illustrate association tests using these simulated data. To learn more about all options of the functions, the reader is advised to look at the manual pages.

Defining genomic regions

For rare variant association tests, the unit of analysis is not a single variant but a genomic region, typically a gene. The difficulty in this type of study is therefore to define the genomic region. In this package, two methods are proposed to group variants into genomic regions. The first one, called by **region.by.pos()** groups the variants based on the distance between them and on the maximum number of groups we want to make. Indeed, the distance between each adjacent variant pair is calculated and the maximum distance between two variants within a genomic region increases until we have less groups than the allowed maximum number of groups. The second one, called by **region.by.gene()** uses positions of known genes to group variants into genomic regions. If the option *include.all=FALSE* is used, only variants within known genes will be assigned to a genomic region, the other ones being left out. If the option *include.all=TRUE*, each variant will be assigned to the nearest gene. The files **hg37** and **hg38** can be used to define gene positions in this function.

Simulation of genetic data

Genetic data can be simulated using the package Ravages. The procedure is similar to the one from Suzanne Leal et al used in the programm SeqPower. Using functions from the package Ravages, it is possible to compute MAF in the controls group and in each group of cases based on MAF in the general population and GRR values.

Calculation of frequencies in each group of individuals

The GRR associated to the heterozygous genotype in the group c corresponds to the ratio between the penetrance of the heterozygous in group c and the penetrance of the homozygous for the reference allele in group c as follow:

$$GRR_{Aa} = \frac{P(Y = c|Aa)}{P(Y = c|AA)}$$

. The proportion of each genotype in each group of cases c can be calculated using Bayes theorem:

$$P(Aa|Y=c) = \frac{P(Y=c|Aa) * P(Aa)}{\sum_{Geno=AA,Aa,aa} P(Y=c|Geno) * P(Geno)} = \frac{GRR_{Aa} * P(Aa)}{P(AA) + GRR_{Aa} * P(Aa) + GRR_{aa} * P(aa)}$$

These frequencies are then calculated in the controls group using the rule of total probability:

$$P(Geno|Y=1) = P(Geno) - \sum_{c=2}^{c=C} P(Geno|Y=c) * P(Y=c)$$

This calculation of genotype probabilities in each group of individuals can be performed using the function **group.mafs.GRR()**. The user needs to give *P(Y=c)* which corresponds to

$$baseline_c$$

, the prevalence of each group of cases ; and the *GRR* values. The GRR values need to ben in a matrix given by the user with one row per cases group and one column per variant. If there is no supposed link between the genetic relative risk associated to the heterozygous genotype and the genetic relative risk associated to the homozygous for the alternate allele (general model of the disease, $model =$ "$general$"), the user needs to specify two GRR matrices: one for

$$GRR_{Aa}$$

and one for

$$GRR_{AA}$$

. If *model="recessive", "multiplicative" or "dominant"*, only one GRR matrix is needed. To help the user with the construction of this GRR matrix, we implemented the function **compute.GRR.matrix()**. To use this function, the user needs to specify how the GRR needs to be calculated (argument *GRR*). The user can choose to give the same GRR to all the variants (*GRR="constant"*), by specifying this GRR value to the argument *GRR.value*. It is also possible to compute the GRR by using the formula from the publication presenting the method SKAT (*GRR="SKAT"*). Finally, the user can choose to calculate the GRR in another way depening on MAF of the variants in the general population (*GRR="variable"*); in this case, the function to compute the GRR depending on the MAF needs to be given to the argument *GRR.formula*. In the two last situations, a file containing at least a column maf and a column gene should be given to the function with the MAF in the general population (argument *file.pop.maf*). The user can use for example the files *Kryukov* containing MAF simulated under a demographic model of Kryukov or *GnomADgenes* containing MAF from the population NFE in GnomAD. A particular gene from this file can be selected using the argument *select.gene*. The user needs finally to specifiy by which the GRR are multiplicated between each group of cases compared to the first group of cases (number of values: number of cases group - 1) This function will return a GRR matrix in the appropriate format for the function **group.mafs.GRR()**. Examples of these two functions are show below:

```
GRR.del <- compute.GRR.matrix(GRR = "SKAT", file.pop.maf = Kryukov, n.case.groups = 2,
                              GRR.multiplicative.factor=2, select.gene = "R1")
#GRR calculated using the formula from the SKAT paper with two groups of cases,
#the second group having GRR values twice as high as the first one.
GRR.del[,1:5]
```

```
##           [,1]      [,2]     [,3]      [,4]      [,5]
## [1,]  5.037728  6.222656 12.34472  9.042877  8.133663
## [2,] 10.075455 12.445313 24.68944 18.085755 16.267327
```

```
#Calculation of frequency in the two groups of cases and the controls group
#under a multiplicative model of the disease
#MAF of the first unit from the file Kryukov are considered as MAF in the general population
#All variants are deleterious and the prevalence in each group of cases is 0.001
MAF.groups <- group.mafs.GRR(file.pop.maf = Kryukov, GRR = GRR.del, baseline = c(0.001, 0.001), model =
MAF.groups[,1:5]
```

```
##                  [,1]         [,2]         [,3]         [,4]         [,5]
## controls 9.375276e-05 2.785699e-05 5.403418e-07 3.246158e-06 5.972867e-06
## cases_1  4.784006e-04 1.762618e-04 6.912999e-06 3.011198e-05 4.969452e-05
## cases_2  9.563437e-04 3.524614e-04 1.382590e-05 6.022214e-05 9.938410e-05
```

Simulation of genotypes

Once the frequencies are obtained in each group of individuals, it is possible to simulate genotypes. This can be done using the function **random.bed.matrix.GRR()**. This function relies on the function **group.mafs.GRR()** explained previously. The arguments *file.pop.maf*, *select.gene*, *baseline* and *model* are the same as in the function **group.mafs.GRR()**. The proportion of deleterious and protective variants simulated in the genomic region should be specified to *prop.del* and *prop.pro* respectively. The argument *GRR.matrix* should contain a matrix GRR values as if all the variants were deleterious. If *model="general"*, two GRR matrices need to be given (one for the heterozygous genotype and the other for the homozygous genotype for the alternate allele) as a list to the argument *GRR.matrix*. If the user wants to simulate protective variants in addition to deleterious variants, the same type of argument as *GRR.matrix* should be given to *GRR.matrix.pro* but with GRR values as if all variants were protective. If the argument *GRR.matrix.pro* is empty and *prop.pro*>0, the GRR values for protective variants will be calculated as 1/GRR of the deleterious variants. The size of the different groups of individuals should be a vector specified to *size*, and the user should choose whether the causal variants will be the same between the different groups of cases with the argument *same.variant*. Finally, the number of simulations is specified with the argument *replicates*. This function will return a bed matrix with the group of each individual in the field *@ped$pheno*, the first one being considered by default as the controls group, and the replicate number corresponding to the genomic region in the field *@snps$genomic.region* The example below shows how to simulate a group of 1,000 controls and two groups of 500 cases with only deleterious variants having a GRR calculated in the previous example. 50% of deleterious variants are present and they are different between the two groups of cases. 5 genomic regions are simulated.

```
x <- random.bed.matrix.GRR(file.pop.maf = Kryukov, size = c(1000, 500, 500), baseline = c(0.001, 0.001)
                           GRR.matrix = GRR.del, prop.del = 0.5, prop.pro = 0, same.variant = FALSE,
                           genetic.model = "multiplicative", select.gene = "R1", replicates = 5)
x
```

```
## A bed.matrix with 2000 individuals and 1915 markers.
## snps stats are set
##   There are  1633  monomorphic SNPs
## ped stats are set
```

```
table(x@ped$pheno)
```

```
##
##    0    1    2
## 1000  500  500
```

```
table(x@snps$genomic.region)
```

```
##
##  R1  R2  R3  R4  R5
## 383 383 383 383 383
```

Rare variant definition

To perform rare variant analysis, it is important to define what is a rare variant in order to leave out common ones. We therefore computed the function **filter.rare.variants()** which enables to keep only variants of interest based on a given MAF threshold. This function uses and returns a bed.matrix with three filters

available. If the filter *"whole"* is used, all the variants with a MAF lower than the threshold in the entire sample will be kept. If the filter *"controls"* is chosen, all the variants with a MAF lower than the threshold in the control groups will be kept. Finally, if the filter *"any"* is used, all the variants with a MAF lower than the threshold in any of the groups will be kept. Monomorphic variants are also filtered out using this function. It is also possible to specify the minimum number of variants needed to keep a genomic region using the parameter *min.nb.snps*.

```
#Filter of rare variants based on the MAF in any of the groups,
#only the genomic regions with at least 5 variants are kept
x.filter <- filter.rare.variants(x, filter = "controls", maf.threshold = 0.01,
                                 min.nb.snps = 5)
x.filter
```

```
## A bed.matrix with 2000 individuals and 277 markers.
## snps stats are set
## ped stats are set
```

```
table(x.filter@snps$genomic.region)
```

```
##
## R1 R2 R3 R4 R5
## 59 52 56 55 55
```

Rare variant association tests

We have implemented the generalisation of two burden tests, a type of rare variant association tests: CAST and WSS . For these extensions, a non-ordinal multinomial regression is used in the function **score.reg.mlogit()**. If only two groups are compared, a classical logistic regression is performed. This function uses a bed.matrix and depends on the package mlogit. The independant variable in this regression is the genetic effect of the gene represented by a genetic score. The scores CAST or WSS explained later can be directly calculated in this function, but the user can specify another genetic score in the function. All the examples will use the bed.matrix x which was simulated using the simulation procedure explained previously using a multiplicative model of the disease and Kryukov's MAF as MAF in the general population. x contains a group of 1000 controls and two groups of 500 cases. There are 50% of causal variants (all deleterious) which are different between the two groups of cases. GRR are calculated using the formula from the paper presenting the SKAT method and GRR in the second group of cases are twice the GRR in the first one. 5 replicates corresponding to 5 genomic regions have been simulated under this scenario. Finally, only the variants having a maf lower than 1% in any of the three groups will be kept for the analysis.

```
my.pars <- list(OR.del = c(2, 4), prob.del = 0.2, prob.pro = 0.05)
#Simulation of genotypes with 5 replicates for 400 controls and two groups of 200 cases
#with the the the same variants being deleterious or protective but different OR values
x <- random.bed.matrix(pop.maf = Kryukov$maf[Kryukov$gene=="R1"], size = c(400, 200, 200),
                       baseline = c(0.001, 0.001), replicates = 5, OR.pars = my.pars,
                       OR.function = OR.matrix.same.variant)
#Keep only variants with MAF<1% in one of the three groups
x <- filter.rare.variants(x, filter = "any", maf.threshold = 0.01)
x
```

```
## A bed.matrix with 800 individuals and 71 markers.
## snps stats are set
## ped stats are set
```

```
table(x@snps$genomic.region)
```

```
##
```

```
## R1 R2 R3 R4 R5
## 15 14 13 15 14
```

Genetic scores

CAST

The CAST score is a binary score which has a value of one if the individual carries at least one variant in the considered genomic region, 0 otherwise. A MAF threshold for the definition of a rare variant is therefore needed. This score can be computed using the function **CAST.0()**.

WSS

The WSS (Weighted Sum Statistic) score is a continuous score depending on the MAF of each variant. It can be computed using the function **WSS.0()** as follow:

$$WSS_j = \sum_{i=1}^{R} I_{ij} * w_i$$

with

$$w_i = \frac{1}{\sqrt{(t_i * q_i * 1 - qi)}}$$

and

$$q_i = \frac{n_i + 1}{2 * t_i + 1}$$

n_i is the total number of minor alleles genotyped for SNP i, t_i is the total number of alleles genotyped for SNP i and I_{ij} is the number of minor alleles of SNP i for the invidual j. In the original method, each SNP is weighted according to its frequency in the controls group, in our version of WSS, the weights depend on allele frequency calculated on the entire sample.

Regressions

We have extended the two tests CAST and WSS using non-ordinal multinomial regression models. Let consider C groups of individuals including a group of controls (

$$c = 1$$

) and $C - 1$ groups of cases with different sub-phenotypes of the disease. We can compute $C - 1$ probability ratios:

$$ln\frac{P(Y_j = c)}{P(Y_j = 1} = \beta_{0,c} + \beta_{G,c}X_G + \beta_{k1,c}K_1 + ... + \beta_{kl,c}K_l$$

Wehre Y_j corresponds to the phneotyphe of the individual {j} and K_l is a vector for the lth covariate with the corresponding coefficient β_{kl}. The genetic effect is represented by X_G corresponding to the genetic score CAST or WSS with $\beta_{G,c}$ the log-odds ratio associated to this burden score. The p-value associated to the genetic effect is computed using a likelihood ratio test comparing this model to the same model without the genetic effect (null hypothesis). If only two groups are compared using this package, a classical logistic regression is performed. The p-value of theses tests can be obtained using the function **score.reg.mlogit()** which can also return the odds ratio associated to the genetic score with its confidence interval (argument *get.OR.value=TRUE*) at a given alpha threshold (argument *alpha*) in each group of cases. This regression can also be performed on another genetic score than CAST or WSS, which has to be specified in the argument *other.score*.

Power calculation

The power of the burden tests extensions can be directly calculated on simulations using the function **power.burden()**.