gatars version 0.2.0

Contents

“2017-09-12 14:17:17 PDT”

1 Introduction

For comments or questions, please contact Gail Gong at gailgongster@gmail.com

gatars is an acronym for "Genetic Association Tests for Arbitrarily Related Subjects"

gatars tests the association between a specified set of M genetic markers, called target markers, and a binary or quantitative trait on subjects with any genealogical relationship. We condition on trait phenotypes and treat genotypes as random. Test statistics include three "basic" statistics–the squared burden statistic Q_B, the sequence kernel association test (SKAT) statistic Q_S, and a new trait-based kernel statistic Q_T–as well as an ensemble of four statistics Q_{BS}, Q_{BT}, Q_{ST}, Q_{BST} that optimize linear combinations of the three basic statistics. All seven statistics are summarized in Table 1 of the manuscript.

The test statistics use observations on a number N of people, or subjects, who are sampled based on a binary or quantitative trait y_n. Also the calculations require a user-specified trait expectation $(ey)_n$ (for example the predicted value of y_n from a logistic or linear regression on nongenetic covariates and on principle components of ancestry), and a vector of genotypes (either the dosage or carriage of the minor allele) $g_n = (g_{n1}, \cdots, g_{nM})$ at M target markers. Define the genotype matrix of target markers, or more briefly, the "genotype matrix" G to be the $N \times M$ matrix whose (n, m)-th component is the genotype of the n-th subject at the m-th target marker)

The test statistics take into account covariances of the elements in G. For pairwise covariances among the M markers (summarized by Γ in the manuscript) our calculations use the empirical covariance matrix of G. To account for pairwise covariances among the subjects, we require the user to provide a Ψ matrix consisting of interpersonal genotype correlation coefficients (described in the manuscript by the same notation).

Obtaining the p-values of the optimized statistics further requires resampling from sets of markers located throughout the human genome. We require M sampling sets, one for each target marker. The sampling set for the m-th target marker requires markers whose minor allele frequencies match closely with that of its target marker. Also the sampling sets should be independent of the target markers as well as markers known to be associated with the binary trait. The gatars package can create these sampling sets provided the user supplies the following two items: (1) A data set containing genotypes of the N subjects at a large number of markers or snps (in addition to the target markers) throughout the human genome. (In the analysis of the prostate data in the manuscript, we obtained 126702 snps which we feel is a large enough number.) (2) A data set containing the locations of markers associated with the binary trait.The creation of the sampling sets also uses hotspots from "A Fine-Scale Map of Recombination Rates and Hotspots Across the Human Genome", Myers, Simon;Bottolo, Leonardo;Freeman, Colin;McVean, Gil;Donnelly, Peter Science; Oct 14, 2005; 310, 5746; ProQuest Research Library, pg. 321. gatars has translated the base-pair positions of the hotspots from this reference to Build hg38/GRCh38

TOP

2 Installing gatars

If you have not yet installed R, download the latest version (at least 3.2.4) for your operating system at:

http://www.r-project.org

Run the R application. To install the gatars package, enter the following lines of code to the R prompt:

```
install.packages("devtools")
library(devtools)
install_github("gailg/gatars")
if("gatars" %in% rownames(installed.packages())){
  print("gatars installed successfully--you are good to go!")
} else {
  print("something went wrong--ask for help")
}
```

If your installation was successful, you should see the message

```
[1] "gatars installed successfully--you are good to go!"
```

TOP

3 Data required

The function gatars requires quite large data, and typically, you would organize your data in Plink.(http://pngu.mgh.harvard.edu/~purcell/plink/). We assume that you can bring your data into R and create the following data sets.

3.1 bim

A data.frame containing (at least) the three columns chromosome, snp, bp and L rows corresponding to a very large number of L markers. These markers include the target markers and those used to build the sampling sets. The l-th row of bim summarizes the l-th marker and corresponds to the l-th column of genotype(described next). For each row, chromosome is an integer between 1 and 22 (other integers may be included, but only chromosomes 1 through 22 will be used), snp is a character naming the marker, and bp is the position (bp) of the marker. (Because the gatars data set hotspot is in Build hg38/GRCh38, bp must also be expressed in Build hg38/GRCh38.)

3.2 genotype

A matrix with N rows and L columns, whose (n, l)-th element records either the number (0, 1 or 2) of minor alleles of snp l found in the n-th subject or an indicator (0 or 1) for the n-th person carrying at least one minor allele at snp l. The l-th column of genotype corresponds to the l-th row of `bim`. The object `genotype` could be gotten by reading in the `.bed` file from plink after massaging genotype information into either dosage or carriage. (Distinguish `genotype` here the matrix containing target markers AND sampling set markers from the "genotype matrix" dentoed by G in the manuscript, the matrix containing just the target markers.)

3.3 `target_markers`

A vector that is a subset of the column `bim$snp`. This vector names the target markers.

3.4 `fam`

A `data.frame` containing N rows and at least the two columns y and ey where $y_n = 1(n\text{-th person is affected})$ or a quantitative phenotype, and ey_n is a trait prediction, the predicted value of y_n based on nongenetic covariates and possibly principal components of ancestry. After we saw that principal conponents of ancestry were insignificant, we performed a logistic regression of y on (1) age, (2) membership in family or case/control data, and (3) their interaction, and used ey to be the fitted values of the logistic regression.

3.5 `Psi` $= \Psi$

A matrix with N rows and N columns. Ψ_{n_1,n_2} is the correlation for the genotype at one marker between the n_1-th and n_2-th subjects. Ψ can be estimated from known family pedigree structures and/or from the subjects' genotypes at markers independent of those in the target set. The diagonal elements are all unity and two people known to be non-identical-twin full sibs and who have parents known to be completely unrelated have correlation 1/2.

3.6 `exclusion_region`

A `data.frame` with one or several rows of the three columns `chromosome`, `begin`, and `end`. Each row of this `data.frame` reflects one contiguous genomic region known to be associated with the binary trait and therefore a region used by `gatars` when creating the sampling sets. The column `chromosome` is an integer between `1` and `22` naming which chromosome the region lies, and `begin` and `end` describe its beginning and ending positions (bp). If the region consists of a single marker, then `begin` and `end` are both equal to the position of this marker. `begin` and `end` must be expressed in Build hg38/GRCh38 for the same reason the column `bp` in `bim` must be expressed in Build hg38/GRCh38.

3.7 `hotspot`

The `gatars` package provides this data set for your convenience. This file contains (at least) the columns `chromosome` and `center`; `chromosome` describes the number of the chromosome (`1:22`), and `center` describes the location of hotspots.

TOP

4 An example data set

The package `gatars` provides an example data set which I will use to illustrate how to use the package. You can access this example data with the following commands.

```
library(gatars)
bim = alternative_example$bim
exclusion_region = alternative_example$exclusion_region
fam = alternative_example$fam
genotype = alternative_example$genotype
target_markers = alternative_example$target_markers[3:5]
Psi = alternative_example$Psi
```

4.1 `bim` has 24509 rows and includes the columns `chromosome`, `snp`, and `bp`

```
str(bim)
```

```
## 'data.frame':    24509 obs. of  3 variables:
##  $ chromosome: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ snp       : chr  "exm53" "exm94" "exm222" "exm269" ...
##  $ bp        : int  865665 874501 880160 881918 891393 899789 902088 908944 909253 90930
```

4.2 `genotype` has 200 rows and 24509 columns

I chose a relatively small number of columns to keep the example small.

```
str(genotype)
```

```
##  int [1:200, 1:24509] 0 0 0 0 0 0 0 1 0 0 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:200] "1" "2" "3" "4" ...
##   ..$ : chr [1:24509] "exm53" "exm94" "exm222" "exm269" ...
```

The column names of `genotype` must match `bim$snp`:

```
all(colnames(genotype) == bim$snp)
```

```
## [1] TRUE
```

4.3 target_markers is a character vector containing 3 names

```
str(target_markers)
```

```
##  chr [1:3] "exm1061853" "exm1061861" "exm1061863"
```

The elements in target_markers must be included in bim$snp

```
all(target_markers %in% bim$snp)
```

```
## [1] TRUE
```

4.4 fam has 200 rows and includes the columns y and ey

```
str(fam)
```

```
## 'data.frame':    200 obs. of  3 variables:
##  $ id : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ y  : int  0 0 1 0 1 1 1 1 1 0 ...
##  $ e_y: num  0.838 0.898 0.82 0.675 0.944 ...
```

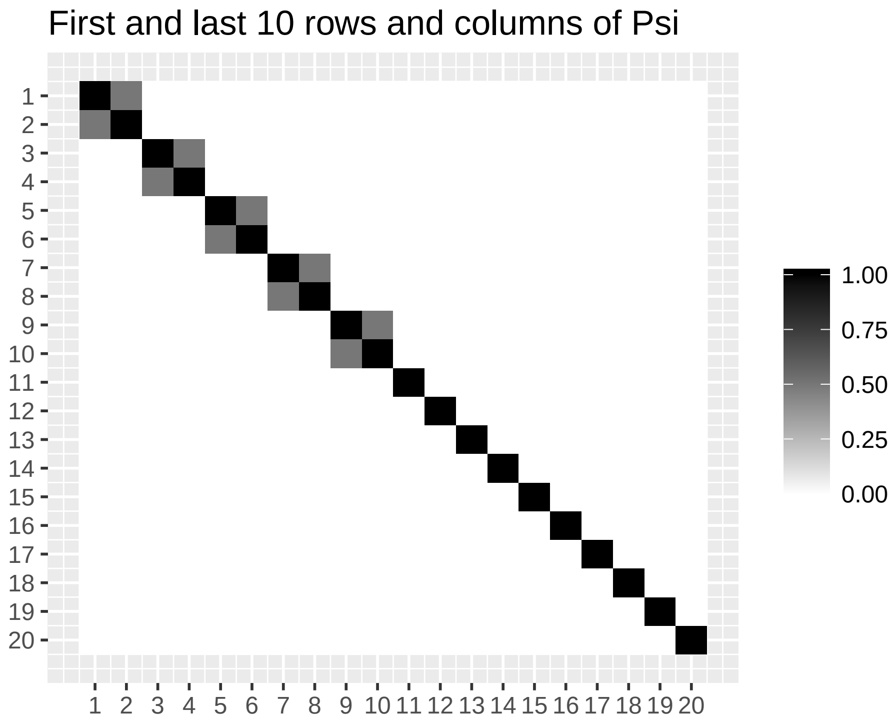
4.5 Psi is a 200 × 200 square matrix

```
str(Psi)
```

```
##  num [1:200, 1:200] 1 0.5 0 0 0 0 0 0 0 0 ...
```

The following figure reflects the fact that the first 100 people are made up of 50 sib pairs, and the last 100 people‘ are independent.

```
NNN = nrow(fam)
first_ten = 1:10
last_ten = NNN - (9:0)
matrix_image_fn(Psi[c(first_ten, last_ten), c(first_ten, last_ten)],
                main = "First and last 10 rows and columns of Psi")
```



4.6 `exclusion_region` is a `data.frame` containing the columns `chromosome`, `begin`, and `end`

```r
str(exclusion_region)
```

```
## 'data.frame':    115 obs. of  3 variables:
##  $ chromosome: num  1 1 1 1 1 2 2 2 2 2 ...
##  $ begin     : int  10496040 150685811 154861707 204549714 205788696 9977740 10570604 20
##  $ end       : int  10496040 150685811 154861707 204549714 205788696 9977740 10570604 20
```

4.7 `hotspot` is available as soon as you enter `gatars`:

```r
str(hotspot)
```

```
## 'data.frame':    25657 obs. of  4 variables:
##  $ chromosome: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ center    : num  949794 1725398 2021339 2102281 2282281 ...
##  $ start     : int  635391 1722898 1950897 2099781 2273781 2387781 2462781 2835085 28810
##  $ end       : int  1264196 1727898 2091781 2104781 2290781 2437781 2493781 2841085 2902
```

5 Check that your genotype matrix has full rank

`genotype_target_markers` is the genotype matrix.

```
library(Matrix)
genotype_target_markers = genotype[, target_markers]
list(target_markers = target_markers,
     rank = as.numeric(rankMatrix( genotype_target_markers)))
```

```
## $target_markers
## [1] "exm1061853" "exm1061861" "exm1061863"
##
## $rank
## [1] 3
```

6 gatars-sampling-set

Once you have your data in R, and you have checked that your genotype matrix has full rank, use the function `gatars_sampling_set` to create your sampling sets.

6.1 A description of how `gatars_sampling_set` builds the sampling sets

Each column in `genotype` corresponds to a row in `bim`, and both correspond to a marker or snp. After removing the target markers, we want to form, from the remaining snps, M sampling sets, one for each of the target markers. Recall that the two conditions we require of the sampling sets are ***the matching requirement***: the snps in the sampling set for a target marker has minor allele frequencies that match closely with that of the target marker and ***the independence requirement***: Any snp from any sampling set is statistically independent of the target markers and any marker known to be associated with the binary trait.

We assume that hotspots from Myers et.al. cut the genome into independent segments, and so snps residing within two consecutive hotspots are independent of snps residing within another two consecutive hotspots. Defining a segment to be a set of snps that all lie within two consecutive hotspots, we obtain a (large) set of independent segments. To satisfy the ***the independence requirement*** we need only remove any segments that contain any target markers or any markers defined by the `exclusion_region` data set.

Of the markers residing in the remaining segments, we calculate their empirical minor allele frequencies and say that a marker's frequency matches the minor allele frequency p_m of the m-th target marker if it is within $p_m \times \left[1- \texttt{epsilon\_on\_log\_scale} , 1+ \texttt{epsilon\_on\_log\_scale} \right]$. If the number of markers satisfying the matching requirement exceeds `1000`, `gatars` randomly chooses `1000`.

6.2 The arguments of `gatars_sampling_set`

The following is the function header of `gatars_sampling_set` showing which arguments are needed and in which order.

```
gatars_sampling_set(
  bim,
  genotype,
  target_markers,
  exclusion_region,
  hotspot,
  epsilon_on_log_scale = 0.02
)
```

6.2.1 bim, genotype, target_markers, exclusion_region, and hotspot

These objects have already been discussed in An example data set.

6.2.2 epsilon_on_log_scale

A positive small real number used to parametrize the matching.

When creating the m-th sampling set for target marker with minor allele frequency π_m, only those markers whose minor allele frequencies falling within the interval $\pi_m \times \left[1 - \texttt{epsilon\_on\_log\_scale}, 1 + \texttt{epsilon\_on\_log\_scale}\right]$ can be included in the sampling set.

6.3 Example calls to gatars_sampling_set

```r
set.seed(2)
epsilon_on_log_scale = 0.02
exclusion_region = NULL
sampling_set = gatars_sampling_set(
  bim,
  epsilon_on_log_scale,
  exclusion_region,
  genotype,
  hotspot,
  target_markers
)
print(sampling_set)
```

```
## $sampling_set_report
##      min p_target    max set_size
## 1 0.0750   0.0750 0.0750      588
## 2 0.0250   0.0250 0.0250      658
## 3 0.0675   0.0675 0.0675      663
##
## $minimum_sampling_set_size
## [1] 588
```

```r
previous_sampling_set = sampling_set
set.seed(2)
```

```
exclusion_region = alternative_example$exclusion_region
head(exclusion_region)
```

```
##    chromosome     begin       end
## 1           1  10496040  10496040
## 2           1 150685811 150685811
## 3           1 154861707 154861707
## 4           1 204549714 204549714
## 5           1 205788696 205788696
## 11          2   9977740   9977740
```

```
sampling_set = gatars_sampling_set(
  bim,
  epsilon_on_log_scale,
  exclusion_region,
  genotype,
  hotspot,
  target_markers
)
print(previous_sampling_set)
```

```
## $sampling_set_report
##      min p_target    max set_size
## 1 0.0750   0.0750 0.0750      588
## 2 0.0250   0.0250 0.0250      658
## 3 0.0675   0.0675 0.0675      663
##
## $minimum_sampling_set_size
## [1] 588
```

```
print(sampling_set)
```

```
## $sampling_set_report
##      min p_target    max set_size
## 1 0.0750   0.0750 0.0750      480
## 2 0.0250   0.0250 0.0250      543
## 3 0.0675   0.0675 0.0675      536
##
## $minimum_sampling_set_size
## [1] 480
```

TOP

7 gatars

7.1 A description of gatars

gatars calculates the p-values of the following seven statistics: the squared burden Q_B, the SKAT Q_S, the case based Q_T, and optimal linear combinations Q_{BS}, Q_{BT}, Q_{ST}, Q_{BST}. The p-value of any linear combination of the basic statistics Q_B, Q_S, and Q_T can be calculated using the davies function from the CompQuadForm package, but this theory no longer applies when a statistic is an optimal one in a universe of linear combinations. We obtain the null distribution of these optimal statistics by resorting to an innovative kind of simulation, which we call *Genome Resampling*.

Recall our notation: G is the $N \times M$ genotype matrix of target markers, y is the N-vector indicator for disease or a quantative measurement, and ey is an N-vector of trait predictions. Any linear combination of the basic statistics can be written as a quadratic form $Q(\alpha) = Z^T A(\alpha) Z$ where Z is a linear function of the two vectors WGy and WGp, W is a diagonal matrix whose nonzero elements weight the target markers, $A(\alpha)$ is a square matrix that does not depend on G, and α are the coeffients of the linear combination of basic statistics. Z is a multivariate normal vector when N is large, with mean μ and covariance matrix Σ which depend on the first two moments of G. For a fixed value of α, the davies function can calculate the p-value for $Q(\alpha)$. Each optimal statistic Q_{BS}, Q_{BT}, Q_{ST}, Q_{BST} is a monotonoic (decreasing) function, $-\log_{10}$, of the smallest p-value of $Q(\alpha)$ over a set of values of α.

To run a simulation to get the null distribution of an optimal statistic Q_{optimal}, we need to obtain simulated observations of a large number S of replications of the genotype matrix $\left\{\tilde{G}^{(1)}, \cdots, \tilde{G}^{(S)}\right\}$. On the s-th simulated genotype matrix $\tilde{G}_s$, we calculate $Q(\alpha)$ over the set of values of α and obtain the optimal one $Q^{(s)}_{\text{optimal}}$. The p-value of Q_{optimal} is the proportion of $\left\{Q^{(1)}_{\text{optimal}}, \cdots, Q^{(S)}_{\text{optimal}}\right\}$ that exceed Q_{optimal}.

The question becomes how to generate each $\tilde{G}^{(s)}$ so that it has the same distribution as the observed genotype matrix G conditional on its following the null model? We propose *Genome Resampling*: Form $\tilde{G}^{(s)}$ by sampling one column from each sampling set created according to Section gatars-sampling-set.

7.2 The arguments of gatars

The following is the function header of gatars showing which arguments are needed and in which order.

```
gatars(fam, Psi, sampling_set, N_sim_reps, weights)
```

7.2.1 fam and Psi

These objects have already been discussed in An example data set.

7.2.2 sampling_set

This object is the result of `gatars_sampling_set` and you may follow the example from gatars-sampling-set

7.2.3 ‘N_sim_reps

An positive integer which specifies the number of replications in the something.

7.2.4 weights

A vector of length `MMM = 3`. The entries in the vector are non-negative real numbers. The size of the `m`-th entry reflects the importance of the `m`-th target marker. If `weights` is not specified, the function assume you would like equal weights among the `MMM` target markers.

TOP

7.3 Example calls to gatars

These take some time to run. If you have larger numbers N of people or larger numbers M of target markers, run times will increase.

```
start = Sys.time()
set.seed(1)
gatars(fam, Psi, sampling_set, N_sim_reps = 1000, weights = c(5, 3, 2))
elapsed_time = Sys.time() - start
paste("1000 sim reps required",
      round(elapsed_time, 1), attributes(elapsed_time)$units)
```

```
start = Sys.time()
set.seed(1)
gatars(fam, Psi, sampling_set, N_sim_reps = 2000)
elapsed_time = Sys.time() - start
paste("2000 sim reps required",
      round(elapsed_time, 1), attributes(elapsed_time)$units)
```

8 All the code in one place

```
library(gatars)

# Preparing the data
bim = alternative_example$bim
exclusion_region = alternative_example$exclusion_region
fam = alternative_example$fam
```

```
genotype = alternative_example$genotype
target_markers = alternative_example$target_markers[3:5]
Psi = alternative_example$Psi
NNN = nrow(fam)
first_ten = 1:10
last_ten = NNN - (9:0)
matrix_image_fn(Psi[c(first_ten, last_ten), c(first_ten, last_ten)],
                main = "First and last 10 rows and columns of Psi")

# Checking the rank of the genotype_target_markers matrix
library(Matrix)
genotype_target_markers = genotype[, target_markers]
list(target_markers = target_markers,
     rank = as.numeric(rankMatrix( genotype_target_markers)))

# Creating the sampling_set
set.seed(2)
sampling_set = gatars_sampling_set(
  bim,
  genotype,
  target_markers,
  exclusion_region = NULL,
  hotspot,
  epsilon_on_log_scale = 0.02)
print(sampling_set)

str(exclusion_region)
previous_sampling_set = sampling_set
set.seed(2)
sampling_set = gatars_sampling_set(
  bim,
  genotype,
  target_markers,
  exclusion_region = exclusion_region,
  hotspot,
  epsilon_on_log_scale = 0.02)
print(previous_sampling_set)
print(sampling_set)

# Calling gatars with N_sim_reps = 1000 and 2000
start = Sys.time()
gatars(fam, Psi, sampling_set, N_sim_reps = 1000)
elapsed_time = Sys.time() - start
paste("1000 sim reps required",
      round(elapsed_time, 1), attributes(elapsed_time)$units)
start = Sys.time()
```

```r
gatars(fam, Psi, sampling_set, N_sim_reps = 2000)
elapsed_time = Sys.time() - start
paste("2000 sim reps required",
      round(elapsed_time, 1), attributes(elapsed_time)$units)
```