snplist

Creating Gene Sets

October 8, 2021

Outline

Introduction

Get Gene and SNP Information

Load Packages

Set the Gene Info Table

Set the SNP Info Table

Make the SNP Sets

Save the Gene Sets

Introduction

This document provides an example of how to use the `snplist` package to create SNP Sets. The SNP Sets created here use genes as the loci of interest, i.e. they are sets of SNPs relevant to specific genes, so we refer to them as Gene Sets. Note that a similar approach could be used to generate SNP Sets for other genomic loci, including bands and pathways.

To make the Gene Sets, we first create a table of gene names, chromosomes, and start and end positions. This information can be pulled from Ensembl [2], based on a list of specified gene names. Next, a similar SNP Table is made which contains rsIDs, chromosomes, and positions. Finally, the Gene Sets are composed from the two tables, based on SNP locations. We also demonstrate exporting the Gene Sets as a PLINK [4] set.

Get Gene and SNP Information

In practice, comprehensive lists of genes and SNPs can be downloaded from the HUGO Gene Nomenclature Committee [3], and dbSNP [5], respectively. For the purposes of this demonstration, we will code our own small collections of genes and SNPs.

First, make a vector of genes on which to base our SNP Sets:

```
genes <- c("BRCA1","BRCA2")
```

The SNP data can come from either a `data.frame` object or a file. Either way, it must have one SNP per line, with columns 'chr' (chromosome),'pos' (position), and 'rsid' (rsID):

```
snpInfo <- data.frame(chr=c(17,17,13,13),
                      pos=c(41211653,41213996,32890026,32890572),
                      rsid=c("rs8176273","rs8176265","rs9562605","rs1799943"),
                      stringsAsFactors=FALSE)
```

Load Packages

Load `snplist` (after installing its dependent packages):

```
library(snplist)

## Loading required package:  RSQLite
## Warning:  multiple methods tables found for 'union'
## Warning:  multiple methods tables found for 'intersect'
## Warning:  multiple methods tables found for 'setdiff'
## Warning:  multiple methods tables found for 'setequal'
## Warning:  multiple methods tables found for 'union'
## Warning:  multiple methods tables found for 'intersect'
## Warning:  multiple methods tables found for 'setdiff'
## Warning:  multiple methods tables found for 'intersect'
## Warning:  multiple methods tables found for 'union'
## Warning:  multiple methods tables found for 'intersect'
## Warning:  multiple methods tables found for 'setdiff'
## Warning:  multiple methods tables found for 'setequal'
```

Get the Gene Information

To create the Gene Table, we need the chromosome and start and end positions of each gene.

For the selected genes, we can get this data from Ensembl using `biomaRt` [1]:

```
geneInfo <- getBioMartData(genes, biomart="ENSEMBL_MART_ENSEMBL",
                                     host="grch37.ensembl.org",
                                     path="/biomart/martservice",
                                   dataset="hsapiens_gene_ensembl")
```

```
geneInfo
```

```
##    gene chr    start      end
## 1 BRCA1  17 41196312 41277500
## 2 BRCA2  13 32889611 32973805
```

Set the Gene Info Table

Now we can create the table of gene info:

```
setGeneTable(geneInfo)

## Create Table : gene
##    gene chr    start      end
## 1 BRCA1  17 41196312 41277500
## 2 BRCA2  13 32889611 32973805
## .....
##   COUNT(*)
## 1        2
```

Set the SNP Info Table

Use the SNP info to create the SNP Table:

```
setSNPTable(snpInfo)

## Create Table : allchrpos
##   chr      pos      rsid
## 1  17 41211653 rs8176273
## 2  17 41213996 rs8176265
## 3  13 32890026 rs9562605
## 4  13 32890572 rs1799943
## .....
##   COUNT(*)
## 1        4
```

Make the SNP Sets

Now we are ready to make the Gene Sets. The SNP Set for each gene is the collection of SNPs located either between the start and end locations of the gene, or within a specified neighborhood around the gene (here 50,000bp).

```
geneset <- makeGeneSet(margin=50000)

## create table anno
##   COUNT(*)
## 1        4
##        rsid
## 1 rs8176273
## 2 rs8176265
## 3 rs9562605
## 4 rs1799943
## .....
##
## Create view annoChrPos
##   chr      pos      rsid
## 1  17 41211653 rs8176273
## 2  17 41213996 rs8176265
## 3  13 32890026 rs9562605
## 4  13 32890572 rs1799943
## .....
##
## Create table annoToGene
##   COUNT(*)
## 1        4
##        rsid  gene
## 1 rs8176273 BRCA1
## 2 rs8176265 BRCA1
## 3 rs9562605 BRCA2
## 4 rs1799943 BRCA2
## .....
##
## Result :
## Gene sets : 2
## rsIDs : 4
```

Make Chip-Secific SNP Sets

The `makeGeneSet()` function has an optional argument to specify a subset of SNP rsIDs we wish to analyze. For example, if we had created a comprehensive SNP Table above, we could use this argument to limit the SNP Sets to include only those SNPs captured on a particular genotyping chip. The argument can be a vector, a `data.frame` with an 'rsid' column, or a reference to a file with one rsID per line. We can also change the `annoTable` name to keep the Gene Sets separate in the SQLite database.

```
chipSNPs <- c("rs0000000","rs8176273","rs9562605")
geneset2 <- makeGeneSet(annoInfo=chipSNPs,
                        annoTable='myChip')


## create table myChip
##   COUNT(*)
## 1        3
##       rsid
## 1 rs0000000
## 2 rs8176273
## 3 rs9562605
## .....
##
## Create view myChipChrPos
##   chr      pos      rsid
## 1  17 41211653 rs8176273
## 2  13 32890026 rs9562605
## .....
##
## Create table myChipToGene
##   COUNT(*)
## 1        2
##       rsid  gene
## 1 rs8176273 BRCA1
## 2 rs9562605 BRCA2
## .....
##
## Result :
## Gene sets : 2
## rsIDs : 2
```

Save the Gene Sets

By default, the gene and SNP tables and the Gene Sets are saved in an SQLite database (its default name is `snplistdb.sqlite`). We can also export the Gene Sets to a PLINK set for later use:

```
exportPLINKSet(geneset2, "mySNPset.set")

## [1] TRUE
```

References

[1] Durinck S, Spellman PT, Birney E, and Huber W. Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. *Nat. Protoc.*, 4:1184–1191, 2009.

[2] Flicek P, Ahmed I, Amode MR, et al. Ensembl 2013. *Nucleic Acids Res.*, 41(Database issue):D48–D55, 2013.

[3] Gray KA, Daugherty LC, Gordon SM, et al. Genenames.org: the hgnc resources in 2013. *Nucleic Acids Res.*, 41(D1):D545–D552, 2013.

[4] Purcell S, Neale B, Todd-Brown K, et al. PLINK: a toolset for whole-genome association and population-based linkage analysis. *Am. J. Hum. Gen.*, 81, 2007.

[5] Sherry ST, Ward MH, Kholodov M, et al. dbSNP: the NCBI database of genetic variation. *Nucleic Acids Res.*, 29(1):308-11, 2001.

Session Information

- R version 4.4.2 (2024-10-31), `x86_64-pc-linux-gnu`
- Running under: `Ubuntu 24.04.1 LTS`
- Matrix products: default
- BLAS: `/usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3`
- LAPACK: `/usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3` ; LAPACK version3.12.0
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: RSQLite 2.3.7, knitr 1.49, snplist 0.18.2
- Loaded via a namespace (and not attached): AnnotationDbi 1.69.0, Biobase 2.67.0, BiocFileCache 2.15.0, BiocGenerics 0.53.2, Biostrings 2.75.1, DBI 1.2.3, GenomeInfoDb 1.43.0, GenomeInfoDbData 1.2.13, IRanges 2.41.0, KEGGREST 1.47.0, R.methodsS3 1.8.2, R.oo 1.27.0, R.utils 2.12.3, R6 2.5.1, Rcpp 1.0.13-1, S4Vectors 0.45.1, UCSC.utils 1.3.0, XVector 0.47.0, biomaRt 2.63.0, bit 4.5.0, bit64 4.5.2, blob 1.2.4, buildtools 1.0.0, cachem 1.1.0, cli 3.6.3, compiler 4.4.2, crayon 1.5.3, curl 6.0.0, dbplyr 2.5.0, digest 0.6.37, dplyr 1.1.4, evaluate 1.0.1, fansi 1.0.6, fastmap 1.2.0, filelock 1.0.3, generics 0.1.3, glue 1.8.0, highr 0.11