twinSNP

Lucia de Hoyos (luciadeh@gmail.com)

March 2020

TwinSNP is an R package used to find genetic variants (SNPs) with similar genetic characteristics of an input SNP. The package is based on the SNPsnap database.

To go through this package, firstly an introduction to the SNPsnap database will be done, analysing the data type and the aim of SNPsnap. Then, in the following sections, the different functions of the package will be explained.

1 Introduction to SNPsnap Database

The SNPsnap webserver (https://data.broadinstitute.org/mpg/snpsnap/) enables SNP-based enrichment analysis by providing matched sets of SNPs that can be used to calibrate background expectations. Specifically, SNPsnap efficiently identifies sets of randomly drawn SNPs that are matched to a set of query SNPs based on: minor allele frequency, number of SNPs in linkage disequilibrium (LD buddies), distance to nearest gene and gene density.

1.1 Data

SNPsnap uses 1000 Genomes Project Phase 3 variants from the three different ancestral cohorts.

Population	Number of SNPs
EUR	9,535,060
EAS	8,433,735
WAFR	16,191,783

To download the data you need to go to the Download tab and then, choose the population to be used, the distance type and the distance cut-off. An example of dataset downloaded using the following options:

- Population: European (EUR) - 1000G Phase 3
- Distance type: LD
- Distance cut-off: $r^2 = 0.5$.

The name of the downloaded file is: ld0.5_collection.tab

1.2 Genetic Properties

The genetic properties used to match the SNPs and find those with similar characteristics are 4, as mentioned in section 1.

1. **Minor allele frequency (MAF):** SNPs are partitioned into minor allele frequency bins of 1-2, 2-3, ..., 49-50% strata.
2. **Number of SNPs in linkage disequilibrium (LD buddies):** the number of "buddy" (or "proxy") SNPs in LD at various thresholds. SNPsnap currently offers LD buddy counts for thresholds using $r^2 > 0.1, 0.2, ..., 0.9$.
3. **Distance to nearest gene:** the distance to the nearest 5' start site using GENCODE gene coordinates. If the SNP is within a gene, the distance to that gene's start site is used.
4. **Gene density:** the number of genes in loci around the SNP, using LD ($r^2 > 0.1, 0.2, ..., 0.9$) and physical distance (100, 200, ..., 1000 kb) to define loci.

1.3 Algorithm

The algorithm that SNPsnap follows is the seen in Figure 1. It enclose 4 main steps.

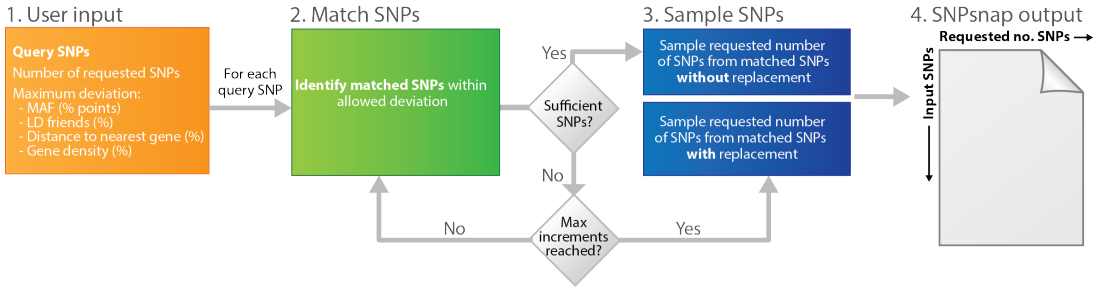

Figure 1: SNPsnap Algorithm

The algorithm used in this project is slightly different. In step 2, instead of creating different intervals to match the SNPs, only the biggest interval (the one set by the user) is used.

2 Introduction to twinSNP

Once the database SNPsnap has been explained briefly, we will focus on the twinSNP package.

2.1 Aim

The main aim of twinSNP is to find SNPs of similar characteristics to a input SNP, being able to do this in a vector with several SNPs and obtaining the matches for each of them. This can be useful in order to look at the SNPs from a more functional point of view, rather than a computational one.

2.2 Functions

The functions that are included in this package are:

- findSNPs
- findSNPs2
- listMatches
- listMatches2
- orderSNPs
- permSNPs
- SNPmatch
- SNPmatch_list

2.2.1 findSNPs and findSNPs2

Both functions do the same: find the SNPs that match with the input SNP. The difference is that findSNPs orders the output SNPs and findSNPs2 does not. This may be valuable in terms of computation time.

The input of both is the same:

- x : A data frame with two columns named input_snp and matched_snps
- y : Character with the input snp in the format chr:pos

Then, the functions will output a vector with all the matches of y in x, ordered (findSNPs) or not (findSNPs2) by chromosome and then by position.

2.2.2 listMatches and listMatches2

After running the function SNPmatch, you get a data frame with the input snp and the matches. In order to convert that to a list where each object is an input SNP with a vector where all its correspondent matches are, you need to run listMatches or listMatches2 over the output of SNPmatch.

The input is then matched_snps or whatever you call the output from SNPmatch, with the columns "input_snp" and "matched_snp". The output is a list with all the SNPs and a vector of its matches ordered (listMatches) or not (listMatches2) by chromosome and position.

2.2.3 orderSNPs

This function inputs a vector of SNPs and outputs the same SNPs but ordered by chromosome and position.

2.2.4 permSNPs

This function randomly chooses SNPs with similar characteristics to your vector of input SNPs. To do so, the input of the function is:

- SNPlist : A list where each object is a input SNP (specified in the name of the object) with a vector of its matches. The output of listMatches, listMatches2 or SNPmatch_list.
- numSNPs : Numeric value with the information of the original number of SNPs.

In case the number of SNPs in SNPlist is less than numSNPs, the function tells you how many SNPs are not found and fills those with random matches of the available SNPs.

2.2.5 SNPmatch and SNPmatch_list

The functions SNPmatch and SNPmatch_list do similar things, the difference is the output. In SNPmatch, the output is the input snps and their matches, whereas in SNPmatch_list it also runs SNPmatch but adding the function of listMatches. Thus, the second function is more useful in terms of computation time. Both functions can take a long time to run (around 20 mins).

The input for both functions is:

- database : Data frame with information of SNPs, for instance, the database downloaded from SNPsnap (ld0.5_collection.tab). It should have column names of snpID, snp_maf, dist_nearest_gene_snpsnap, gene_count, friends_ld05.
- inputSNPs : Character vector with the input SNPs in format chr:pos. It should have the column name "input_snp"
- MAF : Numeric value with interval limits of minor allele frequency in decimals not percentage.
- GD : Numeric value with interval limits of gene density in decimals not percentage.

- DNG : Numeric value with interval limits of distance to nearest gene in decimals not percentage.
- LDB : Numeric value with interval limits of LD buddies in decimals not percentage.

The common output from both functions is the creation of different text files:

- input_snps_excluded.txt : The input SNPs that are not in the database, thus, are excluded as no matches can be found.
- input_snps_annotated.txt : Those SNPs that are in the database.
- input_snps_annotated_unmatched.txt : After running the matching, if no matches are found for certain SNPs even though they appear in the databse, these SNPs will be recorded here.
- matched_snps_annotation.txt : All the matches and their input SNP will be written here, this data frame is also the final output of SNPmatch.

Apart from this text files that are automatically written in the working directory, SNPmatch outputs the data frame used to write matched_snps_annotation.txt whereas SNPmatch_list outputs a list where each object is an input SNP and inside it, a vector with its matches.

3 Example of Code Implementation

```
library(data.table)
setwd("C:/")
## LOAD DATABASE
snpSnap <- readRDS(file = "snpSnap_database_filtered.rds")
colnames(snpSnap) <- c("snpID", "snp_maf", "gene_count", "dist_nearest_gene_snpsnap", "friends_ld05
    ")

## LOAD TARGET VARIANTS TO BE MATCHED
# Read the variants file, this
b <- read.table("variableIDs.txt", stringsAsFactors = F, header = F)
b <- data.frame(b, stringsAsFactors= F); colnames(b) <- "input_snp"

matcx <- SNPmatch_list(database = snpSnap, inputSNPs = b, MAF = 0.05, GD = 0.5, DNG = 0.5, LDB =
    0.5)
# Around 15 minutes.

save(matcx, file = "SNPinfo.RData") # Very important to save it.

## which(lengths(matcx) ==1)
# make sure there are no matches empty
#  this will also pop up if the file "input_snps_annotated_unmatched.txt"
#  is created. Otherwise, there are no SNPs without a match.

## DO PERMUTATIONS
p = permSNPs(SNPlist = matcx, numSNPs = nrow(b)) # Around 6 sec.
```