Using haploR, an R package for querying HaploReg and RegulomeDB

Ilya Y. Zhbannikov

Sat Apr 15 20:26:45 2017

Overview

HaploReg http://archive.broadinstitute.org/mammals/haploreg/haploreg.php and RegulomeDB http://www.regulomedb.org are web-based tools that extracts biological information such as eQTL, LD, motifs, etc. from large genomic projects such as ENCODE, the 1000 Genomes Project, Roadmap Epigenomics Project and others. This is sometimes called "post-GWAS" analysis.

The R-package `haploR` was developed to query those tools (HaploReg and RegulomeDB) directly from `R` in order to facilitate high-throughput genomic data analysis. Below we provide several examples that show how to work with this package.

Note: you must have a stable Internet connection to use this package.

Contact: ilya.zhbannikov@duke.edu for questions of usage the `haploR` or any other issues.

Motivation and general strategy

This package was inspired by the fact that many web-based annotation databases do not have Application Programing Interface (API) and, therefore, do not allow users to query them remotedly. In our research we used Haploreg and Regulome annotation databases and had a hard time with downloading results from Haploreg web site since it does not allow to do this. Regulome was a little bit friendly, but still not offering API therefore we could not include it to our automated data processing pipeline.

We developed a custom analysis pipeline whcih prepares data, performs genetic association analysis and presents results in user-friendly form. Results include a list of genetic variants (SNPs), their corresponding p-values, phenotypes (traits) tested and other meta-information such as LD, alternative allele, minor allele frequency, motifs changed, etc. Of course, we could go thought the SNPs with genome-wide significant p-values (1e-8) and submit each SNP to Haploreg and Regulome manually, one-by-one, but of course it would take time and will not be fully automatic (which ruins one of the pipeline's paradigms). This is especially difficult if the web site does not have a download results option.

Therefore, we developed *haploR*, a user-friendly R package to connet to Haploreg and Regulome remotedly. This package siginificantly saved our time in developing reporting system for our internal genomic analysis pipeline.

Installation of *haploR* package

In order to install the *haploR* package, the user must first install R https://www.r-project.org. After that, *haploR* can be installed either:

- From CRAN (stable version):

```
install.packages("haploR", dependencies = TRUE)
```

- Or from the package web page (developing version):

```
devtools::install_github("izhbannikov/haplor", buildVignette=TRUE)
```

The package depends on the following packages:

- *httr*, version 1.2.1 or later.
- *XML*, version version 3.98-1.6 or later.
- *tibble*, version 1.3.0 or later.
- *RUnit*, version 0.4.31 or later.

Examples

Querying HaploReg

One or several genetic variants

```
library(haploR)
x <- queryHaploreg(query=c("rs10048158","rs4791078"))
x
```

```
## # A tibble: 33 x 33
##      chr pos_hg38    r2  `D'` is_query_snp       rsID   ref   alt   AFR
##    <dbl>    <dbl> <dbl> <dbl>        <dbl>      <chr> <chr> <chr> <dbl>
## 1     17 66213160  0.82  0.93            0  rs4790914     C     G  0.84
## 2     17 66213422  0.82  0.93            0  rs4791079     T     G  0.85
## 3     17 66213896  0.82  0.93            0  rs4791078     A     C  0.84
## 4     17 66214285  0.83  0.93            0  rs1971682     G     C  0.86
## 5     17 66216124  0.83  0.93            0  rs4366742     T     C  0.93
## 6     17 66219453  0.83  0.93            0  rs2215415     G     A  0.91
## 7     17 66220526  0.83  0.93            0  rs3744317     G     A  0.93
## 8     17 66227121  0.83  0.94            0  rs8178827     C     T  0.90
## 9     17 66230111  0.83  0.93            0 rs71160546    GA     G  0.87
## 10    17 66231972  0.82  0.99            0 rs11079645     G     T  0.88
## # ... with 23 more rows, and 24 more variables: AMR <dbl>, ASN <dbl>,
## #   EUR <dbl>, GERP_cons <dbl>, SiPhy_cons <dbl>, Chromatin_States <chr>,
## #   Chromatin_States_Imputed <chr>, Chromatin_Marks <chr>, DNAse <chr>,
## #   Proteins <chr>, eQTL <chr>, gwas <chr>, grasp <chr>, Motifs <chr>,
## #   GENCODE_id <chr>, GENCODE_name <chr>, GENCODE_direction <dbl>,
## #   GENCODE_distance <dbl>, RefSeq_id <chr>, RefSeq_name <chr>,
## #   RefSeq_direction <dbl>, RefSeq_distance <dbl>,
## #   dbSNP_functional_annotation <chr>, query_snp_rsid <chr>
```

Here `query` is a vector with names of genetic variants. `results` are the table similar to the output of HaploReg.

We then can create a subset from the results, for example, to choose only SNPs with r2 > 0.9:

```
subset.high.LD <- x[x$r2 > 0.9, c("rsID", "r2", "chr", "pos_hg38", "is_query_snp", "ref", "alt")]
subset.high.LD
```

```
## # A tibble: 13 x 7
##          rsID    r2   chr pos_hg38 is_query_snp   ref   alt
##         <chr> <dbl> <dbl>    <dbl>        <dbl> <chr> <chr>
## 1  rs10048158  1.00    17 66240200            1     T     C
## 2   rs9895261  1.00    17 66244318            0     A     G
## 3  rs12603947  0.99    17 66248387            0     T     C
## 4   rs7342920  0.99    17 66248527            0     T     G
```

```
## 5   rs4790914  1.00    17 66213160           0     C     G
## 6   rs4791079  1.00    17 66213422           0     T     G
## 7   rs4791078  1.00    17 66213896           1     A     C
## 8   rs1971682  0.98    17 66214285           0     G     C
## 9   rs4366742  0.99    17 66216124           0     T     C
## 10  rs2215415  0.99    17 66219453           0     G     A
## 11  rs3744317  0.99    17 66220526           0     G     A
## 12  rs8178827  0.96    17 66227121           0     C     T
## 13 rs71160546  0.94    17 66230111           0    GA     G
```

We can then save the *subset.high.LD* into an Excel workbook:

```
require(openxlsx)
write.xlsx(x=subset.high.LD, file="subset.high.LD.xlsx")
```

These steps are summarized in a picture below.

Uploading file with variants

If you have a file with your SNPs you would like to analyze, you can supply it on an input as follows:

```
library(haploR)
x <- queryHaploreg(file=system.file("extdata/snps.txt", package = "haploR"))
x
```

```
## # A tibble: 33 x 33
##      chr pos_hg38    r2  `D'` is_query_snp       rsID   ref   alt   AFR
##    <dbl>    <dbl> <dbl> <dbl>        <dbl>      <chr> <chr> <chr> <dbl>
## 1     17 66213160  0.82  0.93            0  rs4790914     C     G  0.84
## 2     17 66213422  0.82  0.93            0  rs4791079     T     G  0.85
## 3     17 66213896  0.82  0.93            0  rs4791078     A     C  0.84
## 4     17 66214285  0.83  0.93            0  rs1971682     G     C  0.86
## 5     17 66216124  0.83  0.93            0  rs4366742     T     C  0.93
## 6     17 66219453  0.83  0.93            0  rs2215415     G     A  0.91
## 7     17 66220526  0.83  0.93            0  rs3744317     G     A  0.93
## 8     17 66227121  0.83  0.94            0  rs8178827     C     T  0.90
## 9     17 66230111  0.83  0.93            0 rs71160546    GA     G  0.87
## 10    17 66231972  0.82  0.99            0 rs11079645     G     T  0.88
## # ... with 23 more rows, and 24 more variables: AMR <dbl>, ASN <dbl>,
## #   EUR <dbl>, GERP_cons <dbl>, SiPhy_cons <dbl>, Chromatin_States <chr>,
## #   Chromatin_States_Imputed <chr>, Chromatin_Marks <chr>, DNAse <chr>,
## #   Proteins <chr>, eQTL <chr>, gwas <chr>, grasp <chr>, Motifs <chr>,
## #   GENCODE_id <chr>, GENCODE_name <chr>, GENCODE_direction <dbl>,
## #   GENCODE_distance <dbl>, RefSeq_id <chr>, RefSeq_name <chr>,
## #   RefSeq_direction <dbl>, RefSeq_distance <dbl>,
## #   dbSNP_functional_annotation <chr>, query_snp_rsid <chr>
```

File "snps.txt" is a text file which contains one rs-ID per line.

Using existing studies

Sometimes you would like to explore results from already performed study. In this case you should first the explore existing studies from HaploReg web site and then use one of them as an input parameter. See example below:

```
library(haploR)
# Getting a list of existing studies:
```

```
studies <- getHaploRegStudyList()
# Let us look at the first element:
studies[[1]]
```

```
## $name
## [1] "<ce><b2>2-Glycoprotein I (<ce><b2>2-GPI) plasma levels (Athanasiadis G, 2013, 9 SNPs)"
##
## $id
## [1] "1756"
```

```
# Let us look at the second element:
studies[[2]]
```

```
## $name
## [1] "5-HTT brain serotonin transporter levels (Liu X, 2011, 1 SNP)"
##
## $id
## [1] "2362"
```

```
# Query Hploreg to explore results from
# this study:
x <- queryHaploreg(study=studies[[1]])
x
```

```
## # A tibble: 117 x 33
##      chr pos_hg38    r2  `D'` is_query_snp       rsID   ref   alt   AFR
##    <dbl>    <dbl> <dbl> <dbl>        <dbl>      <chr> <chr> <chr> <dbl>
## 1     11 34524785  0.97  1.00            0   rs836138     C     A  0.34
## 2     11 34524788  0.87  0.97            0 rs11032744     C     T  0.04
## 3     11 34526877  1.00  1.00            0   rs836137     A     G  0.37
## 4     11 34527359  1.00  1.00            0   rs836135     G     A  0.36
## 5     11 34527815  1.00  1.00            0   rs704727     T     A  0.16
## 6     11 34530979  0.96  0.99            0   rs836133     C     T  0.16
## 7     11 34531545  0.90  1.00            0 rs77003093     C     T  0.01
## 8     11 34533644  1.00  1.00            1   rs836132     G     A  0.16
## 9     11 34534390  1.00  1.00            0   rs836131     C     T  0.16
## 10    11 34535548  1.00  1.00            0   rs836130     G     T  0.36
## # ... with 107 more rows, and 24 more variables: AMR <dbl>, ASN <dbl>,
## #   EUR <dbl>, GERP_cons <dbl>, SiPhy_cons <dbl>, Chromatin_States <chr>,
## #   Chromatin_States_Imputed <chr>, Chromatin_Marks <chr>, DNAse <chr>,
## #   Proteins <chr>, eQTL <chr>, gwas <chr>, grasp <chr>, Motifs <chr>,
## #   GENCODE_id <chr>, GENCODE_name <chr>, GENCODE_direction <dbl>,
## #   GENCODE_distance <dbl>, RefSeq_id <chr>, RefSeq_name <chr>,
## #   RefSeq_direction <dbl>, RefSeq_distance <dbl>,
## #   dbSNP_functional_annotation <chr>, query_snp_rsid <chr>
```

Querying RegulomeDB

To query RegulomeDB use this function:

```
queryRegulome(query = NULL,
              format = "full",
              url = "http://www.regulomedb.org/results",
              timeout = 10,
              check_bad_snps = TRUE,
              verbose = FALSE)
```

This function queries RegulomeDB www.regulomedb.org web-based tool and returns results in a data frame.

Arguments

- query: Query (a vector of rsIDs).
- format: An output format. Only 'full' is currently supported. See http://www.regulomedb.org/results.
- url: Regulome url address. Default: http://www.regulomedb.org/results
- timeout: A 'timeout' parameter for 'curl'. Default: 10.
- check_bad_snps: Checks if all query SNPs are annotated (i.e. presented in the Regulome Database). Default: 'TRUE'
- verbose: Verbosing output. Default: FALSE.

Output

A list of two: (1) a data frame (table) wrapped to a *tibble* object and (2) a list of bad SNP IDs. Bad SNP ID are those IDs that were not found in 1000 Genomes Phase 1 data

Example

```
library(haploR)
x <- queryRegulome(c("rs4791078","rs10048158"))
x$res.table
```

```
## # A tibble: 2 x 5
##   `#chromosome` coordinate       rsid
##           <chr>      <dbl>      <chr>
## 1         chr17   64236317 rs10048158
## 2         chr17   64210013  rs4791078
## # ... with 2 more variables: hits <chr>, score <dbl>
```

Session information

```
sessionInfo()
```

```
## R Under development (unstable) (2017-03-04 r72303)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: macOS Sierra 10.12.4
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib
##
## locale:
## [1] C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] openxlsx_4.0.17 haploR_1.4.4
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.10    XML_3.98-1.6    digest_0.6.12   rprojroot_1.2
```

```
##  [5] mime_0.5        R6_2.2.0        backports_1.0.5 magrittr_1.5
##  [9] evaluate_0.10   httr_1.2.1      stringi_1.1.5   curl_2.4
## [13] RUnit_0.4.31    rmarkdown_1.4   tools_3.4.0     stringr_1.2.0
## [17] yaml_2.1.14     compiler_3.4.0  htmltools_0.3.5 knitr_1.15.1
## [21] tibble_1.3.0
```