GWAS-based Mendelian Randomization Path Analysis

Yuan-De Tan and Dajiang Liu  tanyuande@gmail.com

October 17, 2016

Abstract

GMRP can perform analyses of Mendelian randomization (MR), correlation, path of causal variables onto disease of interest and SNP annotation analysis. MR includes SNP selection with given criteria and regression analysis of causal variables on the disease to generate beta values of causal variables on the disease. Using the beta vectors, GMRP performs correlation and path analyses to construct path diagrams of causal variables to the disease. GMRP consists of 8 R functions: chrp, fmerg, mktable, pathdiagram, pathdiagram2, path, snpposit, ucsccannot and 5 datasets: beta.data, cad.data, lpd.data, SNP358.data, SNP368annot.data. chrp is used to separate string vector hg19 into two numeric vectors: chromosome number and SNP chromosome position. Function fmerg is used to merge two GWAS result datasets into one dataset. Function mktable performs SNP selection and creates a standard beta table for function path to do MR and path analyses. Function pathdiagram is used to create a path diagram of causal variables onto the disease or onto outcome. Function pathdiagram2 can merge two-level pathdiagrams into one nested pathdiagram where inner pathdiagram is a pathdiagram of causal variables contributing to outcome and the outside pathdiagram is a path diagram of causal variables including outcome onto the disease. The five datasets provide examples for running these functions. lpd.data and cad.data provide an example to create a standard beta dataset for path function to do path analysis and SNP data for SNP annotation analysis by performing mktable and fmerg. beta.data are a standard beta dataset for path analysis. SNP358.data provide an example for snpposit to do SNP position annotation analysis and SNP368annot.data are for ucsccannot to do SNP function annotation analysis.

Contents

1 Introduction 1
2 Loading Data 2
3 Preparation of Standard Beta Table 4
4 Two-way Scatter Plots for Beta Values of Disease and Undefined Causal Variables 6
5 MR and Path Analysis 7
6 Create Path Diagram 8
7 Create Two-level Nested Path Diagram 8
8 SNP Annotation Analysis 10
9 Session Info 15

1 Introduction

As an example of human disease, coronary artery disease (CAD) is one of the causes leading to death and infirmity worldwide [1]. Low-density lipoprotein cholesterol (LDL) and triglycerides (TG) are viewed as risk factors causing CAD. In epidemiological studies, plasma concentrations of increasing TG and LDL and deceasing HDL have been observed to be associated with risk for CAD [2, 3]. However, from observational studies, one could not directly infer that these cholesterol concentrations in plasma are risk factors causing CAD [4, 3, 5, 6]. A big limitation of
observational studies is to difficulty distinguish between causal and spurious associations due to confounding [7]. An efficient approach to overcome this limitation is Mendelian Randomization (MR) analysis [5, 8] where genetic variants are used as instrumental variables. For this reason, many investigators tried to use genetic variants to assess causality and estimate the causal effects on the diseases.

MR analysis can perfectly exclude confounding factors associated with disease. However, when we expand one causal variable to many, MR analysis becomes challenged and complicated because the genetic variant would have additional effects on the other risk factors, which violate assumption of no pleiotropy. An unknown genetic variant in MR analysis possibly provides a false instrument for causal effect assessment of risk factors on the disease. The reason is that if this genetic variant is in LD with another gene that is not used but has effect on the disease of study [5, 9]. It then violates the third assumption. These two problems can be addressed by using multiple instrumental variables. For this reason, Do et al (2013) developed statistic approach to address this issue [4]. However, method of Do et al [4] cannot disentangle correlation effects among the multiple undefined risk factors on the disease of study. The beta values obtained from regression analyses are not direct causal effects because their effects are entangled with correlations among these undefined risk factors.

The best way to address the entanglement of multiple causal effects is path analysis that was developed by Wright [12, 11]. This is because path analysis can dissect beta values into direct and indirect effects of causal variables on the disease. However, path analysis has not broadly been applied to diseases because diseases are usually binary variable.

This vignette is intended to give a rapid introduction to the commands used in implementing MR analysis, regression analysis, and path analysis, including SNP annotation and chromosomal position analysis by means of the GMRP package.

We assume that user has the GWAS result data from GWAS analysis or GWAS meta analysis of SNPs associated with risk or confounding factors and a disease of study. If all studied causal variables of GWAS data are separately saved in different sheet files, then files are assumed to have the same sheet format and they are required to be merged by using function fmerg into one sheet file without disease GWAS data. After a standard beta table is created with mktable, user can use function path to perform RM and path analyses. Using the result of path analysis, user can draw path plot (pathdiagram) with functions pathdiagram and pathdiagram2. These will be introduced in detail in the following examples.

We begin by loading the GMRP package.

```r
> library(GMRP)
```

## 2 Loading Data

GMRP provides five data files: beta.data, cad.data, lpd.data, SNP358.data and SNP368annot.data where

1pd.data was a subset (1069 SNPs) of four GWAS result datasets for LDL, HDL, TG and TC. These GWAS result data sheets were downloaded from the website\(^1\) where there are 120165 SNPs scattered on 23 chromosomes and 40 variables. Four GWAS result datasets for LDL, HDL, TG and TC were merged into one data sheet by using fmerg where fl1 and fl2 are two GWAS result data sheets. ID1 and ID2 are key id in files fl1 and fl2, respectively, and required. A and B are postfix for fl1 and fl2. Default values are A="" and B="". method is method for merging. In the current version, there are four methods: method="No" or "no" or "NO" or "N" or "n" means that the data with unmatched SNPs in file1 and file 2 are not saved in the merged file; method="ALL" or "All" or "all" or "A" or "a" indicates that the data with all unmatched SNPs in file 1 and file 2 are saved in the unpaired way in the merged data file; if method="file1", then those with unmatched SNPs only from file1 are saved or if method="file2", fmerg will save the data with unmatched SNPs only from file2. Here is a simple example:

```r
> data1 <- matrix(NA, 20, 4)
> data2 <- matrix(NA, 30, 7)
> SNPID1 <- paste("rs", seq(1:20), sep="")
> SNPID2 <- paste("rs", seq(1:30), sep="")
> data1[,1:4] <- c(round(runif(20), 4), round(runif(20), 4), round(runif(20), 4), round(runif(20), 4))
> data2[,1:7] <- c(round(runif(30), 4), round(runif(30), 4), round(runif(30), 4), round(runif(30), 4), round(runif(30), 4), round(runif(30), 4), round(runif(30), 4))
```

\(^1\)http://csg.sph.umich.edu//abecasis/public/lipids2013/
```R
> data2[,1:4] <- c(round(runif(30), 4),round(runif(30), 4), round(runif(30), 4), round(runif(30), 4))
> data2[,5:7] <- c(round(seq(30)*runif(30), 4), round(seq(30)*runif(30), 4), seq(30))
> data1 <- cbind(SNPID1, as.data.frame(data1))
> data2 <- cbind(SNPID2, as.data.frame(data2))
> dim(data1)
> dim(data2)
> colnames(data1) <- c("SNP", "var1", "var2", "var3", "var4")
> colnames(data2) <- c("SNP", "var1", "var2", "var3", "var4", "V1", "V2", "V3")
> data1<-DataFrame(data1)
> data2<-DataFrame(data2)
> data12 <- fmerg(fl1=data1, fl2=data2, ID1="SNP", ID2="SNP", A=".dat1", B=".dat2", method="No")

User can take the following approach to merge all four lipid files into a data sheet: LDLHDL<-fmerg(fl1=LDLfile,fl2=HDLfile, ID1=" SNP" , ID2=" SNP" , A=".LDL", B=".HDL", method="No")
TGTC<-fmerg(fl1=TGfile,fl2=TCfile, ID1=" SNP" , ID2=" SNP" , A=".TG", B=".TC", method="No")
lpd<-fmerg(fl1=LDLHDL,fl2=TGTC,ID1=" SNP" ,ID2=" SNP" ,A="",B="",method="No")
cad.data was also a subset (1069 SNPs) of original GWAS meta-analyzed dataset that was downloaded from the website and contains 2420360 SNPs and 12 variables.

beta.data that was created by using function mktable and fmerg from lpd.data and cad.data is a standard beta table for MR and path analyses.

SNP358.data contains 358 SNPs selected by mktable for SNP position annotation analysis.

SNP368annot.data is the data obtained from function analysis with SNP Annotation Tool and provides example of performing function ucscanno to draw a 3D pie and output the results of proportions of SNPs coming from gene function various elements.

```
### Preparation of Standard Beta Table

The standard beta table for MR and path analyses must have the standard format. It has columns: `chrn`, `posit`, `rsid`, `a1.x`, `a2.x`, ···, `an.x`, `freq.x`, `freq.x1`, `freq.x2`, ···, `freq.xn`, `beta.x`, `beta.x1`, `beta.x2`, ···, `beta.xn`, `sd.x`, `sd.x1`, `sd.x2`, ···, `sd.xn`, `pv.j`, `N.x`, `N.x1`, `N.x2`, ···, `N.xn`, `pc.j`, `hg.d`, `SNP.d`, `freq.d`, `beta.d`, `N.d`, `freq.case`, `pd.j` where `x1`, `x2`, ···, `xn` are variables.

- `beta` is vector of beta values of SNPs on variable vector `X=(x1,x2,···,xn)`.
- `freq` is vector of frequency of allele 1 with respect to variable vector `X=(x1,x2,···,xn)`.
- `sd` is vector of standard deviations of variable `(x1,x2,···,xn)` specific to SNP. If `sd` does not specifically correspond to SNP, then `sd.x` has the same value for all SNPs.

- `d` denotes disease.

- `N` is sample size.

- `freq.case` is frequency of disease.

- `chrn` is vector of chromosome number.

- `posit` is position vector of SNPs on chromosomes. Some time, `chrn` and posit are combined into string `hg19` or `hg18`.

- `pv.j` is defined as `p-value`, `pc.j`, and `pd.j` as proportions of sample size for SNP `j` to the maximum sample size in causal variables and in disease, respectively.

We use function `mktable` to choose SNPs and make a standard beta table for MR and path analyses. For convenience, we first assign `lpd.data` to `lpd` and `cad.data` to `cad`:

The standard beta table will be created via 15 steps:

#### Step1: calculate pvj

```r
> pvalue.LDL <- lpd$P.value.LDL
> pvalue.HDL <- lpd$P.value.HDL
> pvalue.TG <- lpd$P.value.TG
> pvalue.TC <- lpd$P.value.TC
```
> pv <- cbind(pvalue.LDL, pvalue.HDL, pvalue.TG, pvalue.TC)
> pvj <- apply(pv, 1, min)

Step2: retrieve causal variables from data lpd and construct a matrix for beta:

> beta.LDL <- lpd$beta.LDL
> beta.HDL <- lpd$beta.HDL
> beta.TG <- lpd$beta.TG
> beta.TC <- lpd$beta.TC
> beta <- cbind(beta.LDL, beta.HDL, beta.TG, beta.TC)

Step3: construct a matrix for allele1:

> a1.LDL <- lpd$A1.LDL
> a1.HDL <- lpd$A1.HDL
> a1.TG <- lpd$A1.TG
> a1.TC <- lpd$A1.TC
> allele <- cbind(a1.LDL, a1.HDL, a1.TG, a1.TC)

Step4: give sample sizes of causal variables and calculate pcj

> N.LDL <- lpd$N.LDL
> N.HDL <- lpd$N.HDL
> N.TG <- lpd$N.TG
> N.TC <- lpd$N.TC
> ss <- cbind(N.LDL, N.HDL, N.TG, N.TC)
> sm <- apply(ss, 1, sum)
> pcj <- round(sm / max(sm), 6)

Step5: Construct matrix for frequency of allele1 in each causal variable in 1000G.EUR

> freq.LDL <- lpd$Freq.A1.1000G.EUR.LDL
> freq.HDL <- lpd$Freq.A1.1000G.EUR.HDL
> freq.TG <- lpd$Freq.A1.1000G.EUR.TG
> freq.TC <- lpd$Freq.A1.1000G.EUR.TC
> freq <- cbind(freq.LDL, freq.HDL, freq.TG, freq.TC)

Step6: construct matrix for sd of each causal variable (here sd is not specific to SNP j). The following sd values for LDL, HDL, TG and TC were means of standard deviations of these lipoprotein concentrations in plasma over 63 studies from Willer et al [10].

> sd.LDL <- rep(37.42, length(pvj))
> sd.HDL <- rep(14.87, length(pvj))
> sd.TG <- rep(92.73, length(pvj))
> sd.TC <- rep(42.74, length(pvj))
> sd <- cbind(sd.LDL, sd.HDL, sd.TG, sd.TC)

Step7: SNPID and position:

> hg19 <- lpd$SNP_hg19.HDL
> rsid <- lpd$rsid.HDL

Step8: separate chromosome number and SNP position using chrp:

> chr <- chrp(hg = hg19)

Step9: get new data:

> newdata <- cbind(freq, beta, sd, pvj, ss, pcj)
> newdata <- cbind(chr, rsid, allele, as.data.frame(newdata))
> dim(newdata)

Step10: retrieve data from cad and calculate pdj and frequency of coronary artery disease: freq.case in population:

> hg18.d <- cad$chr_pos_b36
> SNP.d <- cad$SNP #SNPID
> a1.d <- tolower(cad$reference_allele)
Step11: combine these cad variables into new data sheet using cbind

```r
> newcad <- cbind(freq.d, beta.d, N.case, N.ctr, freq.case)
> dim(newcad)
```

Step12: give name vector of causal variables:

```r
> varname <-c("CAD", "LDL", "HDL", "TG", "TC")
```

Step13: choose SNPs using parameters LG, Pv, Pc and Pd and create standard beta table using mktable(cdata, ddata, rt, varname, LG, Pv, Pc, Pd) where

cdata is beta data of SNPs regressed on causal variables. Here cdata=newdata.
ddata is beta data of SNPs regressed on the disease (here CAD). Here ddata=newcad.

LG: a numeric parameter. LG is used to choose SNPs with given interval threshold for linkage disequilibrium (LD). Default LG = 10.
Pv: a numeric parameter. Pv is used to choose SNPs with a given p-value cutoff. Default Pv = 5 × 10−8.
Pc: a numeric parameter. Pc is used to choose SNPs with a given cutoff for the proportion of sample size to maximum sample size in causal variable data. Default Pc=0.979.
Pd: a numeric parameter. Pd is used to choose SNP with a given cutoff for the proportion of sample size to the maximum sample size in disease data. Default Pd =0.979.

rt has two options: “beta” and “path”. If rt=”beta” or “Beta” or “B”, then mktable return a beta coefficient matrix of SNPs regressed on causal variables and disease , if rt=”path” or “Path” or “P” it returns a path coefficient matrix of SNPs directly contributing to causal variables and disease.

```r
> mybeta <- mktable(cdata=newdata, ddata=newcad, rt="beta", varname=varname, LG=1, Pv=0.00000005, Pc=0.979, Pd=0.979)
> dim(mybeta)
> beta <- mybeta[,4:8] # standard beta table for path analysis
> snp <- mybeta[,1:3] # snp data for annotation analysis
> beta<-DataFrame(beta)
> head(beta)
```

4 Two-way Scatter Plots for Beta Values of Disease and Undefined Causal Variables

To roughly display relationship of the undefined causal variables to disease of study, we use simple R plot function to create two-way plots of beta of multiple SNP regressed on the undefined causal variable versus the disease.

```r
> data(beta.data)
> beta.data<-DataFrame(beta.data)
> CAD <- beta.data$cad
> LDL <- beta.data$ldl
> HDL <- beta.data$hdl
> TG <- beta.data$tg
> TC <- beta.data$tc
> par(mfrow=c(2, 2), mar=c(5.1, 4.1, 4.1, 2.1), oma=c(0, 0, 0, 0))
> plot(LDL,CAD, pch=19, col="blue", xlab="beta of SNPs on LDL", ylab="beta of SNP on CAD", cex.lab=1.5,
```
MR and Path Analysis

After standard beta table was successfully created by mztb, user can use function path to perform RM analysis (regression analysis of causal variable beta values on the disease or outcome beta values), correlation among the undefined causal variables and disease and path analyses with model of

\[ y \sim x_1 + x_2 + \cdots + x_n \]

where \( y \) is disease or outcome variable, \( x_i \) is undefined causal variable \( i \). Path analysis is based on RM analysis(regression coefficients of the causal beta on the disease beta). path will produce three tables: beta coefficients, sd values and t-test results of causal variables on disease or outcome, correlation matrix and path matrix(direct and indirect path coefficients).
> data(beta.data)
> mybeta <- DataFrame(beta.data)
> mod <- CAD~LDL+HDL+TG+TC
> pathvalue <- path(betav=mybeta, model=mod, outcome="CAD")

### 6 Create Path Diagram

Once user finished performance of path, user will have correlation matrix and direct path coefficients of undefined causal variable onto the disease. User is required to open a csv file saving results of path analysis and make table in R **Console** or copy correlation matrix without disease correlation coefficients to **excel** and copy direct path coefficients to the last column. Here is an example of making correlation and path table:

> mypath <- matrix(NA,3,4)
> mypath[1,] <- c(1.000000, -0.066678, 0.420036, 0.764638)
> mypath[2,] <- c(-0.066678, 1.000000, -0.559718, 0.496831)
> mypath[3,] <- c(0.420036, -0.559718, 1.000000, 0.414346)
> colnames(mypath) <- c("LDL", "HDL", "TG", "path")
> mypath <- as.data.frame(mypath)
> mypath

<table>
<thead>
<tr>
<th></th>
<th>LDL</th>
<th>HDL</th>
<th>TG</th>
<th>path</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.000000</td>
<td>-0.066678</td>
<td>0.420036</td>
<td>0.764638</td>
</tr>
<tr>
<td>2</td>
<td>-0.066678</td>
<td>1.000000</td>
<td>-0.559718</td>
<td>0.496831</td>
</tr>
<tr>
<td>3</td>
<td>0.420036</td>
<td>-0.559718</td>
<td>1.000000</td>
<td>0.414346</td>
</tr>
</tbody>
</table>

The last column is direct path coefficients, we use "path" to name this column. With this table (for example, mypath), user can use function `pathdiagram(pathdata,disease,R2,range)` to create path diagram. Here

- **pathdata** is path result data consisting of causal correlation matrix and direct path coefficient vectors.
- **disease** is a string that specifies disease name. If the disease name is long or has multiple words, then we suggest an abbreviated name, for example, coronary artery disease are shorted as "CAD".
- **R2**, a numeric parameter, is $R$-square obtained from path analysis.
- **range** is range of specified columns for correlation matrix. For example, `range=c(2:4)` means the correlation coefficient begins with column 2 and end at column 4. For our current example, `range=c(1:3)`.

> library(diagram)
> pathdiagram(pathdata=mypath, disease="CAD", R2=0.988243, range=c(1:3))

### 7 Create Two-level Nested Path Diagram

Consider one of the causal variables is outcome of the other causal variables, but we also concern if all variables are risk factors for the disease of study. In this case we want to construct two-level nested path diagram using function `pathdiagram2(pathD,pathO,rangeD,rangeO,disease,R2D,R2O)` where

- **pathD** is a $R$ object that is disease path result data consisting of correlation matrix of undefined causal variables to be identified in Mendelian randomization analysis and path coefficient vector of these variables directly causing the disease of study.
- **pathO** is a $R$ object that is outcome path result data consisting of correlation matrix of undefined causal variables and path coefficient vector of these variables directly contributing to outcome. This outcome variable may be one of risk factors or causal variables in disease path data. These variables in **pathO** are the same with those in **pathD**.
- **rangeD** is numeric vector, specifies column range for correlation coefficient matrix in **pathD**, for example, `rangeD=c(2:4)` means the correlation coefficient begins with column 2 and end at column 4.
- **rangeO** is numeric vector, specifies column range for correlation coefficient matrix in **pathO**, see example in **rangeD**.
Figure 2: Path diagram demo. This path diagram shows the direct risk effects of causal variables LDL, HDL and TG on the disease CAD and their correlations.

disease is a string that specifies disease name. If the disease name is long or has multiple words, then we suggest an abbreviated name, for example, "coronary artery disease" can be shortened as "CAD".

Here is an example of pathD data:

```r
> pathD <- matrix(NA, 4, 5)
> pathD[1,] <- c(1, -0.070161, 0.399038, 0.907127, 1.210474)
> pathD[2,] <- c(-0.070161, 1, -0.552106, 0.212201, 0.147933)
> pathD[3,] <- c(0.399038, -0.552106, 1, 0.44100, 0.64229)
> pathD[4,] <- c(0.907127, 0.212201, 0.441007, 1, -1.035677)
> colnames(pathD) <- c("LDL", "HDL", "TG", "TC", "path")
> pathD <- as.data.frame(pathD)

> pathD
  LDL       HDL       TG       TC     path
  1 1.000000 -0.070161 0.399038 0.907127  1.210474
table 2 0.937061 1.000000 -0.552106 0.212201  0.147933
table 3 0.399038 -0.552106 1.000000 0.441000  0.64229
> pathD[4,] <- c(0.907127, 0.212201, 0.441007, 1, -1.035677)
> colnames(pathD) <- c("LDL", "HDL", "TG", "TC", "path")
> pathD <- as.data.frame(pathD)

Using pathD and mypath, we can perform function pathdiagram2 to create a two-level nested path diagram:

> pathdiagram2(pathD = pathD, pathO = mypath, rangeD = c(1:4), rangeO = c(1:3), disease = "CAD", R2D = 0.536535, R2O = 0.988243)
Figure 3: Demo of two-level nested path diagram. The outside path diagram shows the direct risk effects of undefined causal variables LDL, HDL, TG and TC on the disease CAD, the inner path diagram indicates the direct contributions of LDL, HDL and TG to TC and the correlation relationships among these variables.

Note that in the current version, GMRP can just create two-level nested path diagram, maybe in the later version, GMRP will create more complex path diagrams such as more than one inner path diagram and/or multiple-disease path diagram using structure equations.

8 SNP Annotation Analysis

SNPs chosen will be annotated in function and chromosome position. Position annotation analysis will give position information of these selected SNPs on chromosomes including chromosome distribution and averaged intervals between SNPs. We use snpposit to perform SNP position annotation. This package provides 358 SNPs selected by mktable.

> data(SNP358.data)
> SNP358 <- DataFrame(SNP358.data)
> head(SNP358)

Dataframe with 6 rows and 3 columns
  rsid chr posit
  <factor> <integer> <integer>
1 rs10056022 5    74969415
2 rs10059560 5   156405784
head displays data format required by snpposit. User can create similar table for SNP position annotation analysis. To create chromosome position histogram, we need graphics:

```r
> library(graphics)
```

With SNP data SNP358, we can perform SNP position annotation using function snpposit(SNPdata, SNP_hg19, LG, main, maxd) where

- **SNPdata** is R object that may be hg19 that is a string vector(chr####.#########) or two numeric vectors (chromosome number and SNP position).
- **SNP_hg19** is a string parameter. It may be "hg19" or "chr". If SNP_hg19="hg19", then SNPdata contains a string vector of hg19 or if SNP_hg19="chr", then SNPdata consists of at least two numeric columns: chr and posit. chr is chromosome number and posit is SNP physical position on chromosomes. Note that "chr" and "posit" are required column names in SNPdata if SNP_hg19="chr".
- **LG** is a numeric parameter that gives maximum permissible distance between positions. Its default is 10.
- **main** is a string which is title of graph. If no title is given, then main="". Its default is "A".
- **maxd** is a numeric parameter that is maximum distance for truncating chromosome columns. If there are not big differences among 23 chromosomes, then maxd can be set to be larger than 2000kbp. Its default is 2000kbp.

```r
> snpposit(SNPdata=SNP358, SNP_hg19="chr", LG=10, main="A", maxd=2000)
```

SNP function annotation analysis has two steps:

**Step 1:** copy SNP IDs selected to Batch Query box in SNP Annotation Tool. After setting parameters and running by clicking run button, SNP annotation result will be obtained after running for a while. Choose consequence sheet of UCSC and copy the results to excel sheet, "Predicted function" column name is changed to "functionunit" name and save it as csv format.

**Step 2:** input the csv file into R Console using R function read.csv. In GMRP package, we have provided data for SNP function annotation analysis.

```r
> data(SNP368annot.data)
> SNP368<-DataFrame(SNP368annot.data)
> SNP368[1:10,]
```

```r
DataFrame with 10 rows and 6 columns
SNP Allele Strand Symbol Gene function_unit
<factor> <factor> <integer> <factor> <factor> <factor>
1 rs10056022 Q|A 1 POC5 uc003keg.4 3downstream
2 rs10056022 Q|A 1 POC5 uc003keh.4 3downstream
3 rs10056022 Q|A 1 ANKDD1B uc010izt.4 3downstream
4 rs10056022 Q|A 1 POC5 uc010izu.3 3downstream
5 rs10061689 Q|A 1 COL4A3BP uc003kds.3 intronic
6 rs10061689 Q|A 1 COL4A3BP uc003kdt.3 intronic
7 rs10061689 Q|A 1 COL4A3BP uc003kdu.2 intronic
8 rs10061689 Q|A 1 COL4A3BP uc011csu.2 intronic
9 rs10075465 C|G 1 POC5 uc003keg.4 intronic
10 rs10075465 C|G 1 POC5 uc003keh.4 intronic
```

We perform function ucsannot to summarize proportions of SNPs coming from gene various elements such as code region, introns, etc, and then create 3D pie with pie3D of plotrix.

```r
> library(plotrix)
```

ucsanoot has four parameters to be inputted: SNPN, A, B and C, a method and UCSC annotated data:

UCSCannot is annotation data obtained by performing SNP tools.
Figure 4: Chromosomal histogram of 358 selected SNPs. Averaged lengths of SNP intervals on chromosome mean that the SNPs on a chromosome have their averaged lengths of intervals between them. All averaged lengths over 2000kb on chromosomes were truncated, the SNPs on these chromosomes have at least more than 2000 kbp averaged length of interval. Numbers above chr columns are numbers of SNP distributed on the chromosomes.

SNPn is numeric parameter for number of SNPs contained in UCSCannot.
A is numeric parameter for title size, default=2.5.
B is numeric parameter for label size, default=1.5.
C is numeric parameter for labelrad distance, default=0.1.
method is numeric parameter for choosing figure output methods. It has two options: method=1 has no legend but color and pie components are labeled with gene elements, method=2 has legend over pie. The default = 1.

> ucscannot(UCSCannot=SNP368, SNPn=368)

<table>
<thead>
<tr>
<th>genes</th>
<th>exons</th>
<th>introns</th>
<th>TUR3</th>
<th>UTR5</th>
<th>intergenes</th>
<th>upstream</th>
<th>downstream</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>166</td>
<td>0.01234568</td>
<td>0.8138651</td>
<td>0.01424501</td>
<td>0.001899335</td>
<td>0.004748338</td>
<td>0.05508072</td>
</tr>
</tbody>
</table>

> ucscannot(UCSCannot=SNP368, SNPn=368, A=3, B=2, C=1.3, method=2)

<table>
<thead>
<tr>
<th>genes</th>
<th>exons</th>
<th>introns</th>
<th>TUR3</th>
<th>UTR5</th>
<th>intergenes</th>
<th>upstream</th>
<th>downstream</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>166</td>
<td>0.01234568</td>
<td>0.8138651</td>
<td>0.01424501</td>
<td>0.001899335</td>
<td>0.004748338</td>
<td>0.05508072</td>
</tr>
</tbody>
</table>
Distribution of 368 SNPs within 166 genes

- **Intron**: 81.4%
- **Exon**: 1.2%
- **5' UTR**: 0.2%
- **3' UTR**: 1.4%
- **5' Upstream**: 5.5%
- **3' Downstream**: 9.8%
- **Intergenes**: 0.5%

Figure 5: Distribution of the selected SNPs in gene function elements.
Figure 6: Distribution of the selected SNPs in gene function elements.
9 Session Info

> sessionInfo()

R Under development (unstable) (2016-09-29 r71410)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 16.04.1 LTS

locale:
  [1] LC_CTYPE=en_US.UTF-8 LC_NUMERIC=C LC_TIME=en_US.UTF-8
  [4] LC_COLLATE=C LC_MONETARY=en_US.UTF-8 LC_MESSAGES=en_US.UTF-8
  [7] LC_PAPER=en_US.UTF-8 LC_NAME=C LC_ADDRESS=C
  [10] LC_TELEPHONE=C LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
  [1] parallel stats4 stats graphics grDevices utils datasets methods
  [9] base

other attached packages:
  [1] GMRP_1.3.0 GenomicRanges_1.27.0 GenomeInfoDb_1.11.0 IRanges_2.9.0
  [5] S4Vectors_0.13.0 BiocGenerics_0.21.0 plotrix_3.6-3 diagram_1.6.3
  [9] shape_1.4.2

loaded via a namespace (and not attached):
  [1] zlibbioc_1.21.0 BiocStyle_2.3.0 XVector_0.15.0 tools_3.4.0 data.table_1.9.6
  [6] chron_2.3-47

References


