snpGeneSets: an *R* Package for Genome-wide Study Annotation

Hao Mei and Lianna Li

Version 1.10, 09-30-2015

Contents

1. Introduction

Genome-wide studies (GWS) of SNP association and differential gene expression have generated abundant results, and the next-generation sequencing technology has further boosted the increasing. Effective interpretation of these results and understanding of the genetic effects often require massive annotation and post-analysis over genome, which is however a computationally challenging task. To address this challenge, the *snpGeneSets* package is developed to simplify post-annotation and analysis of GWS results. The package integrates local copies of parsed NCBI dbSNP [1] and Entrez Gene [2] databases based on two recent genome builds of GRCh37/hg19 and GRCh38/hg38 and MSigDB gene sets V4.0 [3], and provides three types of main annotations: 1) genomic mapping annotation for SNPs and genes, and function annotation for gene sets; 2) bidirectional mapping relation between SNPs and genes, and between genes and gene sets; and 3) gene effect measures from SNP associations and enrichment analysis-based annotations for identifying function pathways from genes. The auxiliary functions are also provided to facilitate the annotation and analysis for genome-wide study. The package structures and components are summarized at the Figure 1.

Note: The examples below are from the old version of V1.10. The updated manual based on new version is not ready yet. If there is any mistake, please load the help document under R by help(package="snpGeneSets") and refer to the description of related functions for usage.

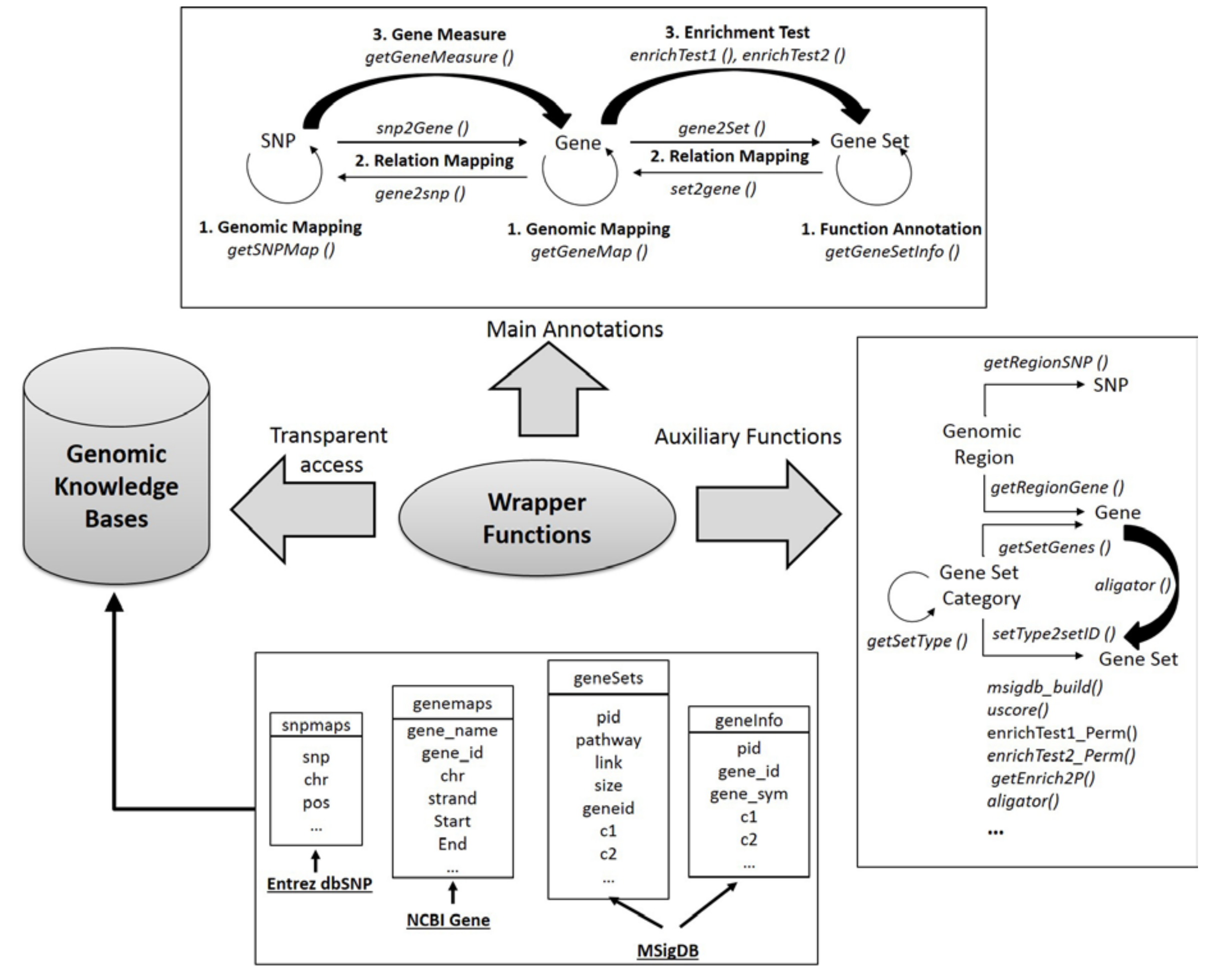


Figure 1. Systematic structures and annotation components of the *snpGeneSets* package

2. Installation

Before the installation of new version, an old version of *snpGeneSets* can be removed by system command:

```
R CMD REMOVE snpGeneSets
```

Or by R command

```
>remove.packages("snpGeneSets")
```

The package source file of *snpGeneSets_1.10.tar.gz* and windows binary file of *snpGeneSets_1.10.zip* can be downloaded from https://www.umc.edu/biostats_software/ .

Installation from the source file of *snpGeneSets_1.10.tar.gz* can be completed through the system command:

```
R CMD INSTALL snpGeneSets_1.10.tar.gz
```

Installation from the binary file of *snpGeneSets_1.10.zip* for Windows can be completed through the GUI interface: "Packages"→" Install package(s) from local zip files..." .

Notes: The package is integrated with parsed NCBI dbSNP 138 (GRCh37/hg19) and 142 (GRCh38/hg38) [1], Entrez gene 105 (GRCh37/hg19) and 106 (GRCh38/hg38) [2]. The installation will automatically download and install the integrated databases based on GRCh37 and GRCh38, which requires high-speed internet access. The SNP annotation data based on GRCh37/hg19 includes common variants with unique position from NCBI dbSNP and those low-frequency variants from *1000* Genome project. The SNP annotation data based on GRCh38/hg38 includes common variants with unique position from NCBI dbSNP, but does not have low-frequency variants from 1000 Genome project.

3. Installation of MSigDB gene sets

Due to the license issue of MSigDB gene sets, the data is not directly provided by the *snpGeneSets* package. Instead, the user needs to visit the MSigDB download web at http://www.broadinstitute.org/gsea/downloads.jsp and registers with the email.

To install the MSigDB 4.0, the zipped file of *msigdb_v4.0_files_to_download_locally.zip* ("MSigDB version 4.0 - zipped msigdb.xml, gmt and chip files") has to be downloaded and extracted locally. All required *gmt* files can be founded at the extracted directory of “msigdb_v4.0_GMTs”. The installation can be completed by function *msigdb_build*:

```
> library(snpGeneSets)
> msigdb_build(gmt_dir="~/tmp/msigdb_v4.0_files_to_download_locally/msigdb_v4.0_GMTs")
```

The argument of *gmt_dir* shows where all the extracted *gmt* files can be found. The function will parse all *gmt* files and build the integrated database.

4. Identification of SNP and gene map positions from an updated reference genome

Many GWAS of SNP associations were based on an old reference genome build, e.g. NCBI36. The *snpGeneSets* can quickly convert old map positions for a large number of GWAS SNPs to updated positions based on a recent genome build, GRCh37 or GRCh38, simultaneously by function of *getSNPMap()*. Map positions of GRCh37 and GRCh38 for genes can be identified by function of *getGeneMap()*.

The *snpGeneSets* comes with two GWS data, T2D-GWAS and T2D-GWES. The T2D-GWAS contains GWAS SNP association for T2D in Finnish population from dbGaP (Analysis ID: pha002839) [5], and T2D-GWES presents differential expression p-values at pancreases of *10* control and *10* T2D human subjects [6], which we obtained by analysis of GEO expression data (GDS3782) using the linear models with empirical Bayes adjusting method [7].

4.1 *Example: Identification of T2D-GWAS SNP map positions*

```
> library(snpGeneSets)

> data("T2DGWAS")

> class(T2DGWAS)
[1] "data.frame"

> dim(T2DGWAS)
[1] 306368      2

> head(T2DGWAS)
         snp          p
1  rs4649592 0.95773144
2 rs41332249 0.98747972
3  rs1079109 0.42112743
4  rs3934834 0.38536813
5  rs3737728 0.64311534
6  rs6687776 0.08061468
```

The T2DGWAS results can be loaded into R by the function *data()*. There are total *306,368* SNPs with GWAS association p-values available. Identifiers of these SNPs and their map positions are obtained based on old genome build. Genomic map positions of these SNPs based on a recent map build can be obtained simultaneously by function of *getSNPMap()* and reference genome build can be specified by parameter *GRCh=37* (in default) or *GRCh=38*.

```
> snpMapAnn<- getSNPMap(T2DGWAS$snp)
> snpMapAnn38<- getSNPMap(T2DGWAS$snp, GRCh=38)
```

Depending on the computer performance, the map annotation may take up to 1 minute for completing the process.

```
> names(snpMapAnn)
[1] "rsid_map" "other"
```

The returned result variables of *snpMapAnn* and *snpMapAnn38* are a list and it contains two components, a data frame of *'rsid_map'* and a character vector of *'other'*. The *'rsid_map'* contains all SNP identifiers that can be found for their genomic positions. The *'other'* contains the SNP identifiers that cannot be found for map positions.

```
> class(snpMapAnn$rsid_map)
[1] "data.frame"

> dim(snpMapAnn$rsid_map)
[1] 306252      3

> dim(snpMapAnn38$rsid_map)
[1] 306045      3

> head(snpMapAnn$rsid_map)
  chr       pos        snp
1   4  21618674 rs10000010
2   4  95733906 rs10000023
3   4 103374154 rs10000030
4   2 237752054  rs1000007
5   4  21895517 rs10000092
6   4 157574035 rs10000121

> head(snpMapAnn38$rsid_map)
  chr       pos        snp
1   4  21617051 rs10000010
2   4  94812755 rs10000023
3   4 102452997 rs10000030
4   2 236843411  rs1000007
5   4  21893894 rs10000092
6   4 156652883 rs10000121

> class(snpMapAnn$other)
[1] "character"

> length(snpMapAnn$other)
[1] 116
```

```
> length(snpMapAnn38$other)
[1] 323

> head(snpMapAnn$other)
[1] "rs4649592"  "rs41332249" "rs1079109"  "rs7549320"  "rs7412106"
[6] "rs12619064"

> head(snpMapAnn38$other)
[1] "rs4649592"  "rs41332249" "rs1079109"  "rs41511844" "rs17559902"
[6] "rs4297265"
```

The mapping annotation based on GRCh37 showed *306,252* SNPs have been identified for genomic map positions and *116* SNPs cannot be identified, which may be due to alteration or obsolete of these rs ids. For reference genome GRCh38, total *306,045* SNPs have been identified, but *323 SNPs* are not.

4.2 *Example: Identification of gene map positions for T2D-GWES*

```
> data("T2DGWES")

> class(T2DExpression)
[1] "data.frame"

> dim(T2DExpression)
[1] 20185     3

> head(T2DExpression)
      symbol gene_id            p
9199   MDFIC   29969 6.399265e-07
3613  PPP2CB    5516 1.209549e-06
3503   FXYD3    5349 1.955109e-06
2292  IGFBP3    3486 2.853953e-06
6984  UNC13B   10497 2.876275e-06
11673  RRAGD   58528 5.047322e-06
```

The T2D-GWES data can be loaded by the *data("T2DGWES")* command, and the results of *T2DExpression* variable are stored as a data frame that contains differential expression p-values for *20,185* genes. The *T2DExpression* contains gene symbol ('symbol'), its Entrez gene ID ('gene_id') and the differential expression p-value ('p'). Map positions of T2D-GWES genes can be identified by *getGeneMap()* function with reference genome specified at parameter of *GRCh* that is 37 in default.

```
> geneMapAnn<-getGeneMap(T2DExpression$gene_id)

> names(geneMapAnn)
[1] "gene_map" "other"

> class(geneMapAnn$gene_map)
[1] "data.frame"

> dim(geneMapAnn$gene_map)
[1] 19299     6
```

```
> head(geneMapAnn$gene_map)
  chr     start       end strand gene_name gene_id
1  19  58858172  58864865      -      A1BG       1
2  12   9220304   9268558      -       A2M       2
3   8  18027971  18081198      +      NAT1       9
4   8  18248755  18258723      +      NAT2      10
5  14  95078639  95090395      +  SERPINA3      12
6   3 151531769 151546276      +     AADAC      13

> class(geneMapAnn$other)
[1] "numeric"

> length(geneMapAnn$other)
[1] 920

> head(geneMapAnn$other)
[1] 100130051 100129513      5558    727770 100132999 100134017

> geneMapAnn38<-getGeneMap(T2DExpression$gene_id, GRCh=38)

> dim(geneMapAnn38$gene_map)
[1] 19283     6

> head(geneMapAnn38$gene_map)
  chr     start       end strand gene_name gene_id
1  19  58346806  58353499      -      A1BG       1
2  12   9067708   9115962      -       A2M       2
3   8  18170462  18223689      +      NAT1       9
4   8  18391245  18401213      +      NAT2      10
5  14  94612377  94624053      +  SERPINA3      12
6   3 151813974 151828488      +     AADAC      13

> length(geneMapAnn38$other)
[1] 927

> head(geneMapAnn38$other)
[1] 100130051 100129513    727770 100132999 100134017    730184
```

The returned annotation variables of *geneMapAnn* and *geneMapAnn38* are a list with two components: *"gene_map"* and *"other"*. The *"gene_map"* is a data frame with *19,299* genes from GRCh37 and *19,283* genes from GRCh38, and the map position of a gene is defined by chromosome (‘chr’), transcription start position (‘start’) and transcription termination position (‘end’). The *"other"* component is a numeric vector and it contains *920* Entrez gene IDs for GRCh37 and *927* genes for GRCh38 that are not identified for their map positions.

The *getGeneMap()* function has a second argument of logical variable, *isGeneID,* that determines if the searched genes are character vector of gene symbol or numerical vector of Entrez Gene ID.

5. Two-way mapping between SNP, gene and pathway

5.1 Mapping between SNP and Gene

Fast mapping of GWAS SNPs to genes is important for interpreting and understanding GWAS results. *snp2Gene* identifies genes spanning the target SNPs, based on user-defined gene boundary and SNP positions.

```
> T2DGWAS[T2DGWAS$p==min(T2DGWAS$p),]
            snp            p
70765   rs886374        2.37573e-06
```

The top SNP hit of the T2D-GWAS is the *rs886374* with association p-value of *2.4E-06.* We can apply the *snp2Gene()* function to obtain the genes that cover this SNP based on either GRCh37 (in default) or GRCh38.

```
> rs886374_map<-getSNPMap("rs886374")$rsid_map

> rs886374_map
        chr     pos             snp
1       4       7738369         rs886374

> rs886374_gene<-snp2Gene(rs886374_map)

> rs886374_gene
$map
        snp             gene_id
1       rs886374        57537

$other
character(0)

> getGeneMap(57537)$gene_map
        chr     start           end             strand  gene_name       gene_id
1       4       7194374         7744564         +       SORCS2          57537

> rs886374_map38<-getSNPMap("rs886374", GRCh=38)$rsid_map

> rs886374_map38
         chr    pos             snp

1       4       7736642         rs886374

> rs886374_gene38<-snp2Gene(rs886374_map38, GRCh=38)

> rs886374_gene38
$map
      snp gene_id
1 rs886374   57537

$other
character(0)
```

```
> getGeneMap(57537,GRCh=38)$gene_map
        chr     start       end         strand  gene_name   gene_id
1       4       7192647     7742837     +       SORCS2      57537
```

The *snp2Gene()* function requires a data frame of SNP map including *'chr'*, *'pos'* and *'snp'* as the input to perform the SNP-Gene mapping. We first obtained the data frame of *rs886374_map* (GRCh37) and *rs886374_map38* (GRCh38) by *getSNPMap()* function. The *rs886374_gene* and *rs886374_gene38* returned by *snp2Gene()* function showed that SNP *rs886374* mapped to Entrez gene ID *57537*. The *getGeneMap()* function showed that the gene ID of *57537* is *SORCS2*, which is at Chromosome 4 from *7,194,374* to *7,744,564* bp for GRCh37 and from *7,192,647* to *7,742,837* bp for GRCh38.

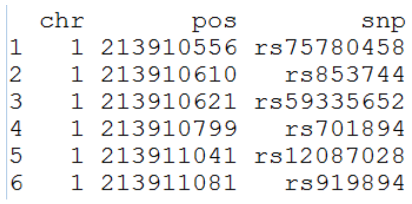
The *snp2Gene()* function can be applied to map all T2D-GWAS SNPs to genes simultaneously. The mapping may take >1 hour depending on the number of GWAS SNPs. To speed the process, GWAS SNPs can be splitted to map *100,000* SNPs every time.

```
> snpGeneMapAnn<-snp2Gene(snpMapAnn$rsid_map)

> names(snpGeneMapAnn)
[1] "map"   "other"

> class(snpGeneMapAnn$map)
[1] "data.frame"

> dim(snpGeneMapAnn$map)
[1] 172041      2

> head(snpGeneMapAnn$map)
          snp gene_id
1  rs10000010   80333
2  rs10000023     658
5  rs10000092   80333
10 rs10000169   57619
11  rs1000022  171425
14 rs10000300   54502

> length(unique((snpGeneMapAnn$map$gene_id)))
[1] 24339

> class(snpGeneMapAnn$other)
[1] "character"

> length(snpGeneMapAnn$other)
[1] 146506

> head(snpGeneMapAnn$other)
[1] "rs10000030" "rs1000007"  "rs10000121" "rs1000014"  "rs10000141"
[6] "rs1000016"

> snpGeneMapAnn38<-snp2Gene(snpMapAnn$rsid_map, GRCh=38)
> head(snpGeneMapAnn38$map)
```

The *snp2Gene()* function returned the SNP-gene mapping annotation results of *snpGeneMapAnn* (GRCh37) and *snpGeneMapAnn38* (GRCh38) for *306,252* GWAS SNPs. The *snpGeneMapAnn* and *snpGeneMapAnn38* are a list with two components: a data frame of *"map"* and a character vector of *"other"*. The *snpGeneMapAnn$map* showed that *172,041* SNPs were successfully mapped to *24,339* genes and *snpGeneMapAnn$other* indicated that *146,506* SNPs are out of gene boundary. The gene boundary is defined by two arguments, *'up'* for the upstream region and *'down'* for the downstream region with default value of *2,000* bp for both. Depending on the computer performance, the SNP-gene mapping for all T2D-GWAS SNPs may take up to *30* minutes. The mapping results can be directly found at *'snpGeneMap'* variable from *"T2DGWAS"* data, which is the same as *snpGeneMapAnn$map*.

In contrast to the *snp2Gene()* function, the *getRegionSNP()* function performs the reverse mapping and it shows annotated common SNPs spanned by the target gene or genomic region. The *getRegionSNP()* function takes a data frame including *'chr'*, *'start'* and *'end'* as the input.

```
> chr=c("14","1","18","16","16")

> start=c(78786077, 213910494, 57850422, 53813450, 53820527)

> end=start+1000

> regionDF=data.frame(chr=chr, start=start, end=end, stringsAsFactors=FALSE)

> regionSNPs<-getRegionSNP(regionDF)

> class(regionSNPs)
[1] "data.frame"

> dim(regionSNPs)
[1] 73  3

> head(regionSNPs)

  chr       pos         snp
1   1 213910494   rs1704198
2   1 213910566  rs10864067
3   1 213910585 rs141152028
4   1 213910675  rs79688837
5   1 213910826 rs182273155
6   1 213910983 rs186584814

>regionSNPs38<-getRegionSNP(regionDF, GRCh=38)

> dim(regionSNPs38)
[1] 24  3

> head(regionSNPs38)
```



```
  chr       pos        snp
1   1 213910556 rs75780458
2   1 213910610   rs853744
3   1 213910621 rs59335652
4   1 213910799   rs701894
5   1 213911041 rs12087028
6   1 213911081   rs919894
```

For the example above, the *getRegionSNP()* function returned the results to a data frame variable of *regionSNPs* for mapping annotations of *73* SNPs (GRCh37) and *regionSNPs38* for mapping annotations of *24* SNPs (GRCh38)

5.2 Mapping between Gene and Pathway

For a significant gene from GWAS or GWES, identification of its implicated pathways may shed light on novel gene function for disease genetics, and the mapping of gene to pathway is implemented by the *gene2Set()* function.

```
> T2DExpression[T2DExpression$p==min(T2DExpression$p),]
        symbol        gene_id        p
9199  MDFIC           29969          6.399265e-07
```

The top gene of the T2D-GWES is MDFIC (Entrez gene ID: gene_id=*29969*) with p-value of *6.4E-07*, which acts as a transcriptional activator or repressor. The *gene2Set()* function can be applied to identify the MSigDB gene sets that include the MDFIC gene.

```
> gid29969_C2<-gene2Set(29969, setType=2)

> length(gid29969_C2)
[1] 45

> head(gid29969_C2)
[1] 4074 4926 4928 4973 5029 5074

> gid29969_C5<-gene2Set(29969, setType=14)

> length(gid29969_C5)
[1] 49

> head(gid29969_C5)
[1] 239 280 301 305 307 343
```

Application of *gene2Set()* function shows that *MDFIC* gene is the component gene of *45* MSigDB gene sets at the category of “C2: curated gene sets” and the component gene of *49* MSigDB gene sets at the category of “C5: GO gene sets”. The category of gene sets can be specified by the argument of *‘setType’*, which takes the value of category ID from *0* to *19*. The *20* gene-set categories and their description can be shown by *getSetType()* function. The Table 1 below summarizes all categories and their IDs, and *setType=2* and *setType=14* correspond to category of “C2: curated gene sets” and “C5: GO gene sets” respectively.

Table 1. Summary of 20 MSigDB gene-set categories

ID	symbol	name
0	c0	C0: all gene sets
1	c1	C1: positional gene sets
2	c2	C2: curated gene sets
3	c2_cgp	C2_CGP: chemical and genetic perturbations
4	c2_cp	C2_CP: Canonical pathways
5	c2_biocarta	C2_CP:BIOCARTA: BioCarta gene sets
6	c2_kegg	C2_CP:KEGG: KEGG gene sets
7	c2_reactome	C2_CP:REACTOME: Reactome gene sets
8	c3	C3: motif gene sets
9	c3_mir	C3_MIR: microRNA targets
10	c3_tft	C3_TFT: transcription factor targets
11	c4	C4: computational gene sets
12	c4_cgn	C4_CGN: cancer gene neighborhoods
13	c4_cm	C4_CM: cancer modules
14	c5	C5: GO gene sets
15	c5_bp	C5_BP: GO biological process
16	c5_cc	C5_CC: GO cellular component
17	c5_mf	C5_MF: GO molecular function
18	c6	C6: oncogenic signatures
19	c7	C7: immunologic signatures

In contrast to *gene2Set()* function, the *getGeneSetInfo()* function identifies all member genes of a pathway and provides the mapping of pathway to genes. The *gid29969_C2* showed that the ***MDFIC*** gene is a member gene of gene-set *ID=5029*. Description of the pathway can be shown by the *getGeneSetInfo()*.

The *getGeneSetInfo()* function below returns the results to *pid5029Ann* which contains 5 components: the *'setID'* of gene set identifier, the *'set_name'* of the gene set name, the *'set_link'* of the MSigDB web link describing the gene set, the *'set_type'* of the gene-set category including the gene set and the *'set_geneid'* of Entrez gene IDs belonging to the gene set.

```
> pid5029Ann<-getGeneSetInfo(5029)

> names(pid5029Ann)
[1] "setID"     "set_name"   "set_link"   "set_type"   "set_geneid"

> pid5029Ann
```

```
$setID
[1] 5029

$set_name
[1] "AKL_HTLV1_INFECTION_DN"

$set_link
[1] "http://www.broadinstitute.org/gsea/msigdb/cards/AKL_HTLV1_INFECTION_DN"

$set_type
         c1          c2      c2_cgp       c2_cp c2_biocarta     c2_kegg
      FALSE        TRUE        TRUE       FALSE       FALSE       FALSE
c2_reactome          c3      c3_mir      c3_tft          c4      c4_cgn
      FALSE       FALSE       FALSE       FALSE       FALSE       FALSE
      c4_cm          c5       c5_bp       c5_cc       c5_mf          c6
      FALSE       FALSE       FALSE       FALSE       FALSE       FALSE
         c7
      FALSE

$set_geneid
 [1]  22934  84525   5774   1848   7273  54504  25957  10632   5367    917
[11]   5771 163486   9218    892  55076  57515  51646 158471  10656   3001
[21]   5175   1846   5743   9659   2015  10129    942   3002  23097   3725
[31]   2534  10578  29103   4212  27230  29909  10314  51761  51465    301
[41]  23376  55125  84864   4128  55540  10049  51389    999  23266   3688
[51]   1288   3490  10142   5168   8477   4208  10447   5476   3359  26123
[61]   7259   5569  29969  51150  28683  10225  26228   5980
```

6. Gene measures by SNP associations and U-score calculation for gene effects

A gene typically contains associations of multiple SNPs from a GWAS, and the *getGeneMeasure()* function provides four measures (*minP*, *2ndP*, *simP* and *fishP*) of the gene effect by summarizing SNP association *p*-values. *U*-score of a gene measure represents percentage of genome-wide genes with effects stronger than the given gene and it can be calculated by *uscore()* function.

For *K* SNPs mapped to a gene with GWAS p-values ($p_1, p_2, \dots p_k$), the ordered p-value is defined as $p_{(1)} \leq p_{(2)} \leq \dots \leq p_{(k)}$, where $p_{(1)} = \min\{p_1, p_2, \dots p_k\}$ and $p_{(k)} = \max\{p_1, p_2, \dots p_k\}$. Four gene measures are calculated respectively as $minP = p_{(1)}$, $2ndP = p_{(2)}$, $simP = \min_i\{Kp_{(i)}/i\}$ and $fishP = Pr\left(X \geq x = -2\sum_{i=1}^{K} log(p_i)\right) = \Psi(x)$, where Ψ is the chi-square distribution function with *df=2K*. Uniform score (*U*-score) is calculated as $U_i = (\sum_j I(M_j < M_i) + 0.5 \cdot \sum_j I(M_j = M_i))/L$, where M_i is gene measure of the *i*-th gene and *L* is the total number of genes.

The *getGeneMeasure()* takes an arguments of *'snpGeneP'*. The *'snpGeneP'* is a data frame containing column of *'snp'* for rs id, column of *'gene_id'* for Entrez gene IDs spanning the *'snp'*, and column of *'p'* for SNP association p-value.

```
> snpGeneMap <- snpGeneMapAnn$map #snpGeneMap can be found from data T2DGWAS

> snpGeneP<-merge(snpGeneMap, T2DGWAS, all=FALSE)

> head(snpGeneP)
```

```
          snp gene_id         p
1 rs10000010   80333 0.2489708
2 rs10000023     658 0.2059405
3 rs10000092   80333 0.7070708
4 rs10000169   57619 0.5055075
5  rs1000022  171425 0.8532224
6 rs10000300   54502 0.5191723
```

> T2DGWASGene0<-getGeneMeasure(snpGeneP)

> head(T2DGWASGene0)

```
  gene_id       minp       sndp       simp       fishp
1       1 0.14992377 0.61819639 0.29984753 0.092682331
2       2 0.63210108 0.65196227 0.79051801 0.585242462
3       3 0.33866379 0.33866379 0.33866379 0.141139178
4       9 0.28229107 0.43147721 0.80126634 0.265557328
5      10 0.04538995 0.05860277 0.08790415 0.003165650
6      12 0.10190141 0.13136668 0.19705002 0.008034187
```

> minp_uscore<-uscore(T2DGWASGene$minp)

> head(minp_uscore)

[1] 0.3713382 0.8397428 0.6154937 0.5545215 0.1592300 0.2841735

> T2DGWASGene <- T2DGWASGene0

> for (ms in c("minp", "sndp", "simp", "fishp")) T2DGWASGene[[ms]]<-uscore(T2DGWASGene[[ms]])

> head(T2DGWASGene)

```
  gene_id      minp      sndp       simp     fishp
1       1 0.3713382 0.7145733 0.28760426 0.4086035
2       2 0.8397428 0.7440528 0.74729857 0.8902790
3       3 0.6154937 0.4576400 0.32357533 0.4921936
4       9 0.5545215 0.5503307 0.75900818 0.6498418
5      10 0.1592300 0.1039279 0.08642508 0.1212252
6      12 0.2841735 0.2105469 0.19181150 0.1590246
```

The *'snpGeneMap'*, *'snpGeneP'* and *'T2DGWASGene'* can be manually created as above. These variables are also pre-generated and automatically loaded with 'T2DGWAS' data. The *T2DGWASGene0* contains measures of *minP, 2ndP, simP* and *fishP* for every T2DGWAS gene. The *minp_uscore* is the uniform score for *minp* measure and U-score can also be similarly generated for other three measures. The *T2DGWASGene* contains U-scores for every gene measure.

We examined *9* genes that were previously reported to have associations with T2D, and their measures (*'gmeasure'*) and U-scores (*'gscore'*) were shown in the Figure 1.

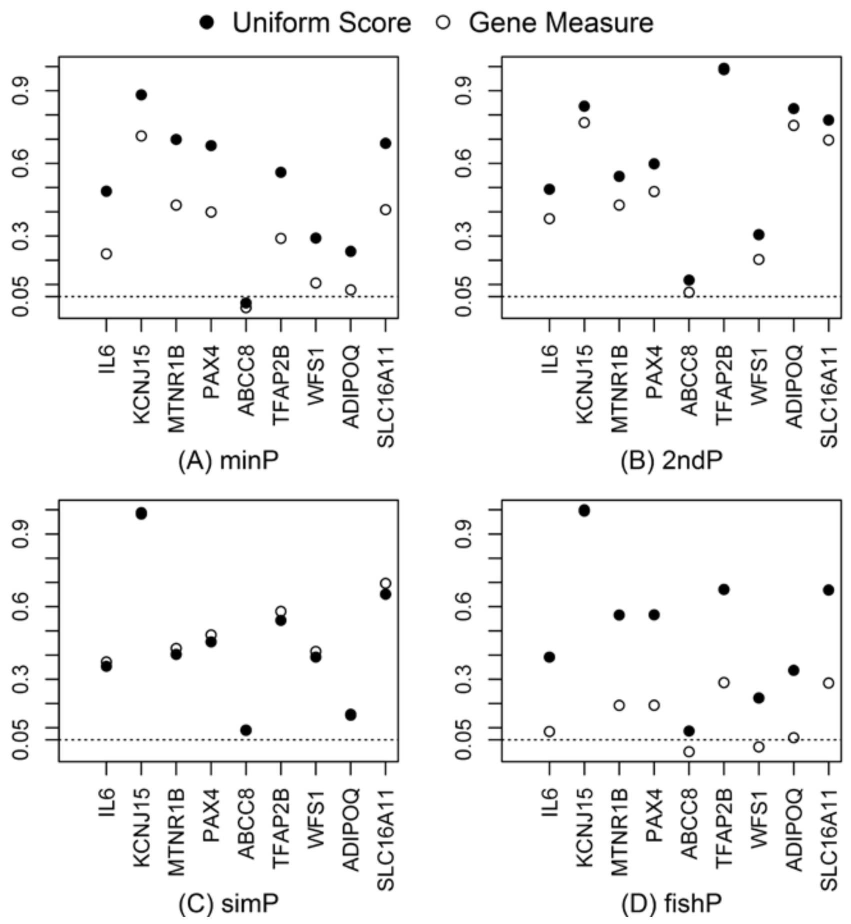


Figure 1. Gene measures and uniform scores of 9 T2D-GWAS genes

```
> genes<-c("IL6", "KCNJ15", "MTNR1B", "PAX4", "ABCC8", "TFAP2B", "WFS1", "ADIPOQ", "SLC16A11")

> genes<-getGeneMap(genes, FALSE)$gene_map[,c("gene_name","gene_id")]

> gmeasure<-merge(genes, T2DGWASGene0,all=FALSE)

> gmeasure
  gene_id gene_name        minp       sndp       simp        fishp
1    3569       IL6 0.226553618 0.37160095 0.37160095 0.0841875404
2    3772    KCNJ15 0.713388908 0.76853023 0.98839315 0.9947997402
3    4544    MTNR1B 0.427681378 0.42768138 0.42768138 0.1924510584
4    5078      PAX4 0.398935200 0.48357556 0.48357556 0.1929153130
5    6833     ABCC8 0.004314517 0.06752818 0.09060486 0.0008465779
6    7021    TFAP2B 0.290199840 0.98718080 0.58039968 0.2864797108
7    7466      WFS1 0.105901871 0.20330600 0.41484572 0.0209583156
8    9370    ADIPOQ 0.077799163 0.75677862 0.15559833 0.0588767430
9  162515  SLC16A11 0.408816921 0.69633535 0.69633535 0.2846736745

> gscore<-merge(genes, T2DGWASGene,all=FALSE)

> gscore
```

```
  gene_id gene_name       minp      sndp       simp      fishp
1    3569       IL6 0.48488023 0.4930564 0.35365052 0.39165537
2    3772    KCNJ15 0.88304778 0.8361272 0.98206582 0.99969185
3    4544    MTNR1B 0.69865237 0.5464275 0.40295411 0.56549160
4    5078      PAX4 0.67338428 0.5981141 0.45464070 0.56635441
5    6833     ABCC8 0.02368626 0.1180205 0.08909569 0.08630182
6    7021    TFAP2B 0.56331402 0.9924607 0.54346933 0.67104236
7    7466      WFS1 0.29148691 0.3056617 0.39186080 0.22287276
8    9370    ADIPOQ 0.23651341 0.8262459 0.15142364 0.33705165
9  162515  SLC16A11 0.68275196 0.7788734 0.65144418 0.66870044
```

The calculation showed that only ABCC8 has all *4* gene measures and U-scores around or smaller then 0.05. The results presented that a stronger gene measure (i.e. smaller p-values) tends to have a smaller *U*-score. However, different gene measures for the same gene have varied *U*-scores, showing inconsistent measures of gene effects over genome. The calculation of *U*-score will unify these gene measures for comparability with the same interpretability. For example, the *minP*, *2ndP*, *simP* and *fishP* presented summary SNP association p-values of *0.004*, *0.068*, *0.091* and *0.0008* for ABCC8 gene respectively, and the corresponding *U*-scores indicated that *2.4%*, *11.8%*, *8.9%* and *8.6%* GWAS genes have stronger gene effects than ABCC8.

For T2D-GWES, differential expression p-value is used to directly measure gene effect and calculate *U*-scores of the selected 9 genes. The p-value of ABCC8 is *3.4E-04,* showing only 0.4% of genes over genome with stronger measured effect than the ABCC8.

```
>data(T2DGWES)

> escore<-uscore(T2DExpression$p)

> T2DExpression$us<-escore

> T2DExpression[T2DExpression$symbol %in% genes$gene_name,]
```

```
        symbol gene_id            p          us
4560     ABCC8    6833 0.0003363277 0.004235819
2490    KCNJ15    3772 0.0268452946 0.091825613
3293      PAX4    5078 0.1437856302 0.270027248
16017 SLC16A11  162515 0.1471156303 0.273792420
2934    MTNR1B    4544 0.1978929899 0.331904880
4994      WFS1    7466 0.2134392425 0.346569235
2330       IL6    3569 0.3036799699 0.440351746
4685    TFAP2B    7021 0.4280112100 0.552068368
6062    ADIPOQ    9370 0.8169270613 0.865320783
```

7. Pathway Enrichment Analysis I of candidate genes

The type I analysis is a generalized pathway enrichment analysis that aims to identify gene sets enriched for a candidate list of genes. The list can be previously identified susceptibility genes or top genes from a GWAS or GWES. The analysis can be performed by function *enrichTest1().*

7.1 ***Example: Enrichment analysis I of T2D-GWAS***

For *T2DGWAS* data, the top *5%* genes were selected as candidate genes by measures of *minP*, *2ndP*, *simP* and *fishP* respectively, and they were tested for pathway enrichment at *186* KEGG gene sets.

```
> topMinpGenes<-
T2DGWASGene[order(T2DGWASGene$minp),][1:trunc(nrow(T2DGWASGene)*0.05),"gene_id"]
> topsndpGenes<-
T2DGWASGene[order(T2DGWASGene$sndp),][1:trunc(nrow(T2DGWASGene)*0.05),"gene_id"]
> topsimpGenes<-
T2DGWASGene[order(T2DGWASGene$simp),][1:trunc(nrow(T2DGWASGene)*0.05),"gene_id"]
> topfishpGenes<-
T2DGWASGene[order(T2DGWASGene$fishp),][1:trunc(nrow(T2DGWASGene)*0.05),"gene_id"]
```

The *enrichTest1()* function takes an argument of *'genes'* for the candidate genes tested for pathway enrichment, and the argument *'setType'* takes the category ID that defines which category of gene sets will be tested for enrichment. For example, *setType=6* defines the KEGG gene-sets for enrichment test. Description of the category ID can be found at Table 1.

```
> minpGeneSets_KEGG<-enrichTest1(topMinpGenes,setType=6)
> names(minpGeneSets_KEGG)
[1] "enrich_test" "useGenes"    "nGenes"     "nTopGenes"   "setTypeInfo"
> head(minpGeneSets_KEGG$enrich_test)
   pid size genesSize       effect         sd        pval
1 2718   62         4  0.009646184 0.02892126 0.252203325
2 2719   32         0 -0.054869945 0.04025669 0.836564565
3 2720   27         2  0.019204129 0.04382593 0.182359889
4 2721   28         0 -0.054869945 0.04303621 0.794908305
5 2722   34         3  0.033365349 0.03905473 0.113359667
6 2723   26         5  0.137437747 0.04466079 0.002345904

> length(minpGeneSets_KEGG$useGenes)
[1] 289

> minpGeneSets_KEGG$nGenes
[1] 5267

> minpGeneSets_KEGG$nTopGenes
[1] 289

> minpGeneSets_KEGG$setTypeInfo
$id
[1] 6

$symbol
[1] "c2_kegg"

$name
[1] "C2_CP:KEGG: KEGG gene sets"

$description
[1] "Gene sets derived from the KEGG pathway database, http://www.genome.jp/kegg/pathway.html"
```

For the candidate genes selected by *minP* measure, the *enrichTest1()* function returned the results to the *minpGeneSets_KEGG* variable, which consists of a data frame of *"enrich_test"*, an integer vector of *"useGenes"*, a number of *"nGenes"*, a number of *"nTopGenes"* and a list of *"setTypeInfo"*. The *"enrich_test"* shows the enrichment test results for every gene set in the specified category defined by *setType*. The *"useGenes"* lists the effective candidate genes used for enrichment test. The *"nGenes"* is the total number of genes in the specified category and the *"nTopGenes"* is the number of effective candidate genes for enrichment test. The analysis above indicated that the KEGG category contains *5,267* genes, of which *289* genes are candidates, and the test aims to identify which gene set in the KEGG category is significantly enriched for the *289* candidate genes. The *"setTypeInfo"* presents description of the specified category, *KEGG*.

```
> minpGeneSets_KEGG$enrich_test[order(minpGeneSets_KEGG$enrich_test$pval),][1:10,]
     pid size genesSize     effect         sd         pval
184 2901   76        17 0.16881427 0.02612199 8.314768e-08
183 2900   85        14 0.10983594 0.02470038 4.694155e-05
185 2902   92        14 0.09730397 0.02374210 1.206243e-04
116 2833   75        11 0.09179672 0.02629556 6.869875e-04
117 2834  134        16 0.06453304 0.01967255 9.446004e-04
138 2855   70        10 0.08798720 0.02721849 1.331522e-03
86  2803  267        26 0.04250833 0.01393662 1.335274e-03
6   2723   26         5 0.13743775 0.04466079 2.345904e-03
147 2864   47         7 0.09406623 0.03321728 3.600516e-03
136 2853   70         9 0.07370148 0.02721849 4.478571e-03
```

```
> sndpGeneSets_KEGG<-enrichTest1(topsndpGenes,setType=6)
> sndpGeneSets_KEGG$enrich_test[order(sndpGeneSets_KEGG$enrich_test$pval),][1:10,]
     pid size genesSize     effect         sd         pval
184 2901   76        19 0.19797798 0.02547334 8.286438e-10
86  2803  267        33 0.07157348 0.01359055 7.224809e-07
185 2902   92        17 0.13276058 0.02315255 8.001061e-07
183 2900   85        15 0.12444856 0.02408703 5.658256e-06
113 2830  201        23 0.06240584 0.01566371 9.876542e-05
117 2834  134        17 0.07484365 0.01918405 1.711858e-04
166 2883   52         9 0.12105490 0.03079577 2.786636e-04
176 2893   54         9 0.11464464 0.03022010 3.843800e-04
167 2884   65        10 0.10182413 0.02754457 4.479005e-04
35  2752   21         5 0.18607321 0.04845997 5.245832e-04
```

```
> simpGeneSets_KEGG<-enrichTest1(topsimpGenes,setType=6)
> simpGeneSets_KEGG$enrich_test[order(simpGeneSets_KEGG$enrich_test$pval),][1:10,]
     pid size genesSize     effect         sd        pval
82  2799   44         6 0.09573330 0.02976404 0.001760317
124 2841   71         8 0.07204572 0.02343090 0.002118911
41  2758   25         4 0.11936966 0.03948646 0.002884391
149 2866   25         4 0.11936966 0.03948646 0.002884391
180 2897   38         5 0.09094861 0.03202775 0.003892430
22  2739   29         4 0.09730069 0.03666226 0.005649881
169 2886   29         4 0.09730069 0.03666226 0.005649881
16  2733   31         4 0.08840192 0.03545989 0.007568922
148 2865   44         5 0.07300602 0.02976404 0.008132063
134 2851   48         5 0.06353633 0.02849690 0.012362053
```

```
> fishpGeneSets_KEGG<-enrichTest1(topfishpGenes,setType=6)
> fishpGeneSets_KEGG$enrich_test[order(fishpGeneSets_KEGG$enrich_test$pval),][1:10,]
```

```
     pid size genesSize     effect         sd         pval
184 2901   76        21 0.21764862 0.02695643 1.547159e-10
183 2900   85        19 0.16486224 0.02548941 5.432837e-08
185 2902   92        17 0.12611544 0.02450052 4.564353e-06
136 2853   70        14 0.14133283 0.02808796 9.070141e-06
86  2803  267        33 0.06492833 0.01438181 1.115548e-05
113 2830  201        27 0.07566119 0.01657567 1.306251e-05
114 2831   84        15 0.11990426 0.02564068 2.224456e-05
176 2893   54        11 0.14503653 0.03197955 4.944214e-05
167 2884   65        12 0.12594821 0.02914825 7.779820e-05
117 2834  134        19 0.08312387 0.02030097 8.851877e-05
```

> getGeneSetInfo(2901)

> getGeneSetInfo(2799)

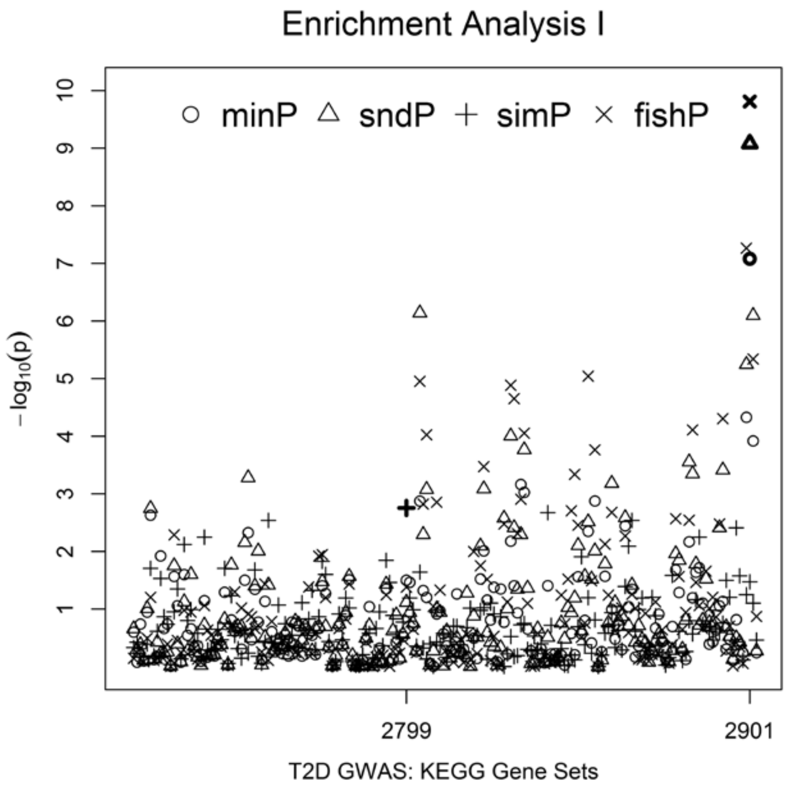


Figure 2. Empirical p-values of KEGG gene sets by enrichment analysis I

The top *10* gene sets for each gene measure were shown above. The *–log₁₀(empirical p-values)* for every gene set was plotted at Figure 2. The four gene measures selected different candidate genes for enrichment test, which caused the pathway test results varied over the measures. The most enriched gene set was pathway of 'arrhythmogenic right ventricular cardiomyopathy' (*PID=2901*) for *minP*, *sndP* and *fishP*, and it was pathway of 'nucleotide excision repair' (*PID=2799*) for *simP*. The pathway of *'2901'*, containing 76 genes, involves *17* candidate genes from *minP* (*effect=16.9%, p_e=8.31E-08*), *19* candidate genes from *2ndP* (*effect=19.8%, p_e=8.29E-10*) and *21* candidate genes from *fishP* (*effect=21.8%, p_e=1.55e-10*); and the pathway of *'2799'*, containing *44* genes, involves *6* candidate genes from *simP* (*effect=9.6%, p_e=1.76E-03*). All component genes for a particular gene set can be identified by the function *getGeneSetInfo()* function, e.g. *getGeneSetInfo(2901)* where 2901 is the pathway ID.

Different pathways may share common genes and these pathways will be dependent, potentially leading to an inflated type I error. To adjust for this issue and multiple testing, the *enrichTest1_Perm()* function applies a permutation-based test to obtain the adjusted p-value (*p_perm*) for pathway enrichment.

The most enriched gene set by every gene measure is prepared for permutation test and the R codes are as below:

```
> KEGG_rst<-
rbind(minpGeneSets_KEGG$enrich_test[minpGeneSets_KEGG$enrich_test$pval==min(minpGeneSets_KE
GG$enrich_test$pval),],
sndpGeneSets_KEGG$enrich_test[sndpGeneSets_KEGG$enrich_test$pval==min(sndpGeneSets_KEGG$en
rich_test$pval),],
simpGeneSets_KEGG$enrich_test[simpGeneSets_KEGG$enrich_test$pval==min(simpGeneSets_KEGG$en
rich_test$pval),],
fishpGeneSets_KEGG$enrich_test[fishpGeneSets_KEGG$enrich_test$pval==min(fishpGeneSets_KEGG$en
rich_test$pval),]
simpGeneSets_KEGG$enrich_test[simpGeneSets_KEGG$enrich_test$pval==min(simpGeneSets_KEGG$en
rich_test$pval),],
fishpGeneSets_KEGG$enrich_test[fishpGeneSets_KEGG$enrich_test$pval==min(fishpGeneSets_KEGG$en
rich_test$pval),] )
> KEGG_rst<-cbind(measure=c("minp","2ndp","simp","fishp"),
             topGenes=c(minpGeneSets_KEGG$nTopGenes,sndpGeneSets_KEGG$nTopGenes,
             simpGeneSets_KEGG$nTopGenes,fishpGeneSets_KEGG$nTopGenes), KEGG_rst)
> colnames(KEGG_rst)<-c("measure", "topGenes","pid","size","setTopGenes","effect","sd","p")
```

Results of the most enriched pathway for every gene measure were saved to *KEGG_rst* variable and the results were shown below:

```
> KEGG_rst
     measure topGenes  pid size setTopGenes    effect         sd            p
184     minp      289 2901   76          17 0.1688143 0.02612199 8.314768e-08
1841    2ndp      274 2901   76          19 0.1979780 0.02547334 8.286438e-10
82      simp      214 2799   44           6 0.0957333 0.02976404 1.760317e-03
1842   fishp      309 2901   76          21 0.2176486 0.02695643 1.547159e-10
```

The *enrichTest1_Perm()* function was applied to get permutation distribution table for calculating permutation p-value of the most enriched pathway. The argument of *geneSize* defines the number of effective candidate genes for enrichment test ***I*** of the target gene set. The argument of *setType* defines the category of gene sets for permutation adjusting. The argument of *times* specifies the number of permutations for generating distribution table and the argument of *seed* assigns a random seed for permutation.

```
> minp_dist=enrichTest1_Perm(geneSize=KEGG_rst[1,"topGenes"], setType=6,times=1000, seed=1)
> sndp_dist=enrichTest1_Perm(geneSize=KEGG_rst[2,"topGenes"], setType=6,times=1000, seed=1)
> simp_dist=enrichTest1_Perm(geneSize=KEGG_rst[3,"topGenes"], setType=6,times=1000, seed=1)
> fishp_dist=enrichTest1_Perm(geneSize=KEGG_rst[4,"topGenes"], setType=6,times=1000, seed=1)
```

The minimum p-value of the KEGG category (*setType=6*) was extract to construct the distribution table and calculate permutation p-value (*p_perm*)

```
> minp_min=apply(minp_dist,2,min)
> sndp_min=apply(sndp_dist,2,min)
> simp_min=apply(simp_dist,2,min)
> fishp_min=apply(fishp_dist,2,min)
> KEGG_rst$p_perm<-c(sum(minp_min<=KEGG_rst[1,"p"]),sum(sndp_min<=KEGG_rst[2,"p"]),
sum(simp_min<=KEGG_rst[3,"p"]),sum(fishp_min<=KEGG_rst[4,"p"]))/1000
> KEGG_rst
```

The results were summarized at Table 2. The gene set of *'2901'* has *p_perm<1e-03* for enrichment of candidate genes from *minP*, *2ndP* and *fishP*, and the gene set of '*2799*' has *p_perm=0.463* for enrichment of candidate genes from *simP*

Table 2. The most enriched KEGG pathway of T2D-GWAS by enrichment analysis I

Measure	Genes	PID	size	setGenes	effect(%)	sd(%)	p_e	*p_perm*
minP	289	2901	76	17	16.9	2.6	8.31E-08	<1e-03
2ndP	274	2901	76	19	19.8	2.5	8.29E-10	<1e-03
simP	214	2799	44	6	9.6	3.0	1.76E-03	0.463
fishP	309	2901	76	21	21.8	2.7	1.55E-10	<1e-03

'Genes': the number of candidate that is taken for enrichment analysis; 'PID': the pathway ID used by *snpGeneSets*. 'size': the number of member genes of a pathway; 'setGenes': the number of candidate genes contained by the pathway.

7.2 ***Example: Enrichment analysis I of T2D-GWES***

For T2D-GWES, the top *5%* genes with the smallest p-values of differential expression were selected as candidate genes and the pathway enrichment test were performed for KEGG gene sets by *enrichTest1()* function.

```
> topExpGenes<-
T2DExpression[order(T2DExpression$p),][1:trunc(nrow(T2DExpression)*0.05),"gene_id"]

> length(topExpGenes)
[1] 1009
```

There are *1,009* candidate genes selected for the enrichment test *I* of KEGG gene sets. However, only *262* genes belongs to the KEGG gene sets and are effectively used for pathway analysis. The 10 most enriched gene sets were saved to *exp_rst* variable.

```
> expGeneSets_KEGG<-enrichTest1(topExpGenes,setType=6)

> expGeneSets_KEGG$nTopGenes
[1] 262

> exp_rst<-expGeneSets_KEGG$enrich_test[order(expGeneSets_KEGG$enrich_test$pval),][1:10,]

> exp_rst
```

Table 3. Ten most enriched KEGG pathways of T2D-GWES by enrichment analysis I

PID	size	setGenes	effect(%)	sd(%)	p_e	*p_perm*
2872	53	7	8.2	3.0	4.25E-03	0.764
2869	23	4	12.4	4.5	4.71E-03	0.788
2803	267	22	3.3	1.3	6.54E-03	0.874
2866	25	4	11.0	4.3	6.86E-03	0.906
2825	47	6	7.8	3.2	7.92E-03	0.932
2719	32	4	7.5	3.8	1.96E-02	0.995
2751	44	5	6.4	3.3	2.06E-02	0.997
2787	22	3	8.7	4.6	2.15E-02	0.997
2864	47	5	5.7	3.2	2.77E-02	0.999
2874	35	4	6.5	3.7	2.80E-02	1

'PID': the pathway ID used by *snpGeneSets*. 'size': the number of member genes of a pathway; 'setGenes': the number of candidate genes contained by the pathway.

The permutation test was applied to obtain permutation adjusted p-value (*p_perm*) by *enrichTest1_Perm()* function.

```
> exp_dist<-enrichTest1_Perm(geneSize =expGeneSets_KEGG$nTopGenes, setType=6,times=1000,
seed=1)

> exp_min=apply(exp_dist,2,min)

> exp_rst$p_perm<-unlist(lapply(exp_rst$pval, function(x) sum(exp_min<=x)/1000))
> colnames(exp_rst)=c("pid","size","setTopGenes","effect","sd","p","p_perm")

> exp_rst

> getGeneSetInfo(2872)
```

The enrichment test *I* and its permutation adjustment were summarized at Table 3. The most enriched gene set is the pathway of ‘Amyotrophic lateral sclerosis’ (*PID=2872*) that contains *53* member genes. The pathway presented enrichment effect of *8.2%* with empirical p_e*=4.25E-03*, but the test based on *1,000* permutations showed that the adjusted p-value was *0.764*.

8. Pathway Enrichment Analysis *II* of GWS genes

The type II analysis is a specialized pathway enrichment analysis that aims to identify enriched gene sets based on genome-wide association and expression study results. The analysis can be performed by *enrichTest2()* function, which test for pathway enrichment by the *USGSA* method. The test depends the threshold of *U*-score that defines genome-wide significant genes. The default value of threshold is *0.05* for *enrichTest2()*, which assumes that *5%* of genome-wide genes are involved in pathway of studied phenotype.

8.1 ***Example: Enrichment analysis II of T2D-GWAS***

Measures of *minP*, *2ndP, simP* and *fishP* or their *U*-scores can all be applied for pathway enrichment test. The required parameter of *geneDF* for *enrichTest2()* function is a data frame which contains at least a column of *‘gene_id’* for Entrez gene IDs and a column of *‘score’* for a gene measure or *U*-score. The argument of *‘setType’* defines the pathway category for enrichment test. For the T2D-GWAS, the example below used *U*-score of *minp* measure for the analysis and *‘setType=6’* limited enrichment analysis to pathways of the KEGG category.

```
> e2_minp<-enrichTest2(geneDF =
data.frame(gene_id=T2DGWASGene$gene_id,score=T2DGWASGene$minp), setType=6)

> names(e2_minp)
[1] "enrich_test" "useGenes"    "nGenes"     "nSigGenes"   "setTypeInfo"

> head(e2_minp$enrich_test)
   pid size genes sigGenes      effect         sd        pval
1 2718   62    50        4  0.01146787 0.03573107 0.256079675
2 2719   32    24        0 -0.06853213 0.05157336 0.818895315
3 2720   27    18        2  0.04257898 0.05955179 0.121221901
4 2721   28    22        0 -0.06853213 0.05386662 0.791100789
5 2722   34    24        3  0.05646787 0.05157336 0.077531279
6 2723   26    25        5  0.13146787 0.05053137 0.005729825

> length(e2_minp$useGenes)
[1] 4217
```

```
> e2_minp$nGenes
[1] 4217

> e2_minp$nSigGenes
[1] 289

> e2_minp$setTypeInfo
$id
[1] 6

$symbol
[1] "c2_kegg"

$name
[1] "C2_CP:KEGG: KEGG gene sets"

$description
[1] "Gene sets derived from the KEGG pathway database, http://www.genome.jp/kegg/pathway.html"
```

```
>e2_minp$enrich_test[order(e2_minp$enrich_test$pval),][1:10,]

     pid size genes sigGenes     effect         sd         pval
184 2901   76    69       17 0.17784468 0.03041631 4.626583e-07
183 2900   85    76       14 0.11567839 0.02898173 1.496373e-04
185 2902   92    81       14 0.10430737 0.02807298 3.136454e-04
116 2833   75    70       11 0.08861073 0.03019827 2.482412e-03
117 2834  134   120       16 0.06480120 0.02306431 2.997884e-03
138 2855   70    65       10 0.08531402 0.03133822 4.127198e-03
86  2803  267   237       26 0.04117251 0.01641183 5.435991e-03
6   2723   26    25        5 0.13146787 0.05053137 5.729825e-03
35  2752   21    18        4 0.15369009 0.05955179 5.956215e-03
9   2726   17    12        3 0.18146787 0.07293575 6.886650e-03
```

For *U*-score of *minP*, the *enrichTest2()* function returned the results to the *e2_minp* variable, which consists of a data frame of *"enrich_test"*, an integer vector of *"useGenes"*, a number of *"nGenes"*, a number of *"nSigGenes"* and a list of *"setTypeInfo"*. The *"enrich_test"* shows the enrichment test results for every gene set in the specified category defined by *setType*. The *"useGenes"* lists GWS genes used for enrichment test. The *"nGenes"* is the total number of GWS genes in the specified category (i.e. the length of *"useGenes"*) and the *"nSigGenes"* is the number of GWS significant genes for enrichment test. The *"setTypeInfo"* presents description of the specified category.

The examples below similarly used *U*-scores of *2ndP, simP* and *fishP* for pathway tests.

(Notes: Either a gene measure or its U-score can be used for type II pathway test. Since a gene measure will automatically be converted to its U-score by enrichTest2 function, they will present the same results.)

```
> e2_sndp<-enrichTest2(geneDF =
data.frame(gene_id=T2DGWASGene$gene_id,score=T2DGWASGene$sndp), setType=6)

> e2_sndp$enrich_test[order(e2_sndp$enrich_test$pval),][1:10,]
```

```
     pid size genes sigGenes     effect         sd         pval
184 2901   76    69       19 0.21038722 0.02967294 5.799842e-09
185 2902   92    81       17 0.14490144 0.02738688 2.677123e-06
86  2803  267   237       33 0.07426541 0.01601072 6.431854e-06
183 2900   85    76       15 0.13239332 0.02827342 2.025952e-05
113 2830  201   180       23 0.06280268 0.01837168 4.965014e-04
117 2834  134   120       17 0.07669157 0.02250062 6.261319e-04
35  2752   21    18        5 0.21280268 0.05809635 6.794323e-04
166 2883   52    47        9 0.12651426 0.03595308 6.859949e-04
176 2893   54    50        9 0.11502490 0.03485781 1.144063e-03
167 2884   65    59       10 0.10451642 0.03208921 1.208255e-03
```

> e2_simp<-enrichTest2(geneDF = data.frame(gene_id=T2DGWASGene$gene_id,score=T2DGWASGene$simp), setType=6)

> e2_simp$enrich_test[order(e2_simp$enrich_test$pval),][1:10,]

```
     pid size genes sigGenes     effect         sd        pval
124 2841   71    50        8 0.10925302 0.03103924 0.000765214
82  2799   44    35        6 0.12068159 0.03709899 0.001563236
149 2866   25    18        4 0.17147525 0.05173207 0.001598354
180 2897   38    28        5 0.12782445 0.04147793 0.002341210
16  2733   31    20        4 0.14925302 0.04907735 0.002661440
41  2758   25    21        4 0.13972921 0.04789459 0.003351116
148 2865   44    35        5 0.09211017 0.03709899 0.007499138
22  2739   29    26        4 0.10309918 0.04304368 0.008809557
134 2851   48    37        5 0.08438816 0.03608238 0.009872187
76  2793   36    27        4 0.09740117 0.04223906 0.010375498
```

> e2_fishp<-enrichTest2(geneDF = data.frame(gene_id=T2DGWASGene$gene_id,score=T2DGWASGene$fishp), setType=6)

> e2_fishp$enrich_test[order(e2_fishp$enrich_test$pval),][1:10,]

```
     pid size genes sigGenes     effect         sd         pval
184 2901   76    69       21 0.23107299 0.03137100 1.276659e-09
183 2900   85    76       19 0.17672516 0.02989139 2.700813e-07
185 2902   92    81       17 0.13660170 0.02895412 1.470682e-05
136 2853   70    61       14 0.15623336 0.03336476 2.143464e-05
114 2831   84    75       15 0.12672516 0.03009001 7.509137e-05
86  2803  267   237       33 0.06596567 0.01692695 8.415314e-05
113 2830  201   180       27 0.07672516 0.01942302 8.888730e-05
88  2805  178   149       23 0.08108758 0.02134813 1.658807e-04
176 2893   54    50       11 0.14672516 0.03685258 1.858624e-04
167 2884   65    59       12 0.13011499 0.03392555 2.533609e-04
```

The top 10 gene sets for each gene measure were shown above. The *–log_{10}(empirical p-values)* for every gene set was plotted at Figure 3. Consistent with the type *I* analysis, the most enriched gene set is the pathway of *PID=2901* for *minP*, *2ndP* and *fishP* measures (Figure 3). However, the most enriched pathway for *simP* is the 'RIG-I-like receptor signaling pathway' (*PID=2841*) in contrast to the pathway of *PID=2799* by enrichment analysis *I*.

```
> KEGG_rst<-rbind(
      e2_minp$enrich_test[e2_minp$enrich_test$pval==min(e2_minp$enrich_test$pval),],
      e2_sndp$enrich_test[e2_sndp$enrich_test$pval==min(e2_sndp$enrich_test$pval),],
      e2_simp$enrich_test[e2_simp$enrich_test$pval==min(e2_simp$enrich_test$pval),],
      e2_fishp$enrich_test[e2_fishp$enrich_test$pval==min(e2_fishp$enrich_test$pval),])
```

```
> KEGG_rst<-cbind(measure=c("minp","2ndp","simp","fishp"),
                  topGenes=c(e2_minp$nSigGenes,e2_sndp$nSigGenes,
                  e2_simp$nSigGenes,e2_fishp$nSigGenes), KEGG_rst)

> colnames(KEGG_rst)<-c("measure",
                  "sigGenes","pid","size","effectGenes","setSigGenes","effect","sd","p")

> KEGG_rst
```

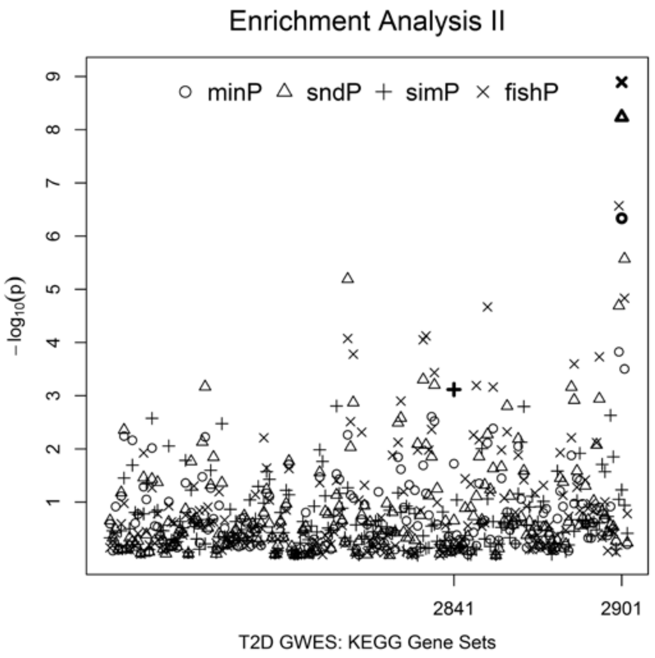


Figure 3. Empirical p-values of KEGG gene sets by enrichment analysis II

For enrichment analysis *II*, the pathway of *'2901'*, containing *69* GWAS genes, involves *17* significant genes from *minP* (*effect=17.8%,* p_e*=4.63E-07*), *19* significant genes from *2ndP* (*effect=21.0%,* p_e *=5.80E-09*) and *21* significant genes from *fishP* (*effect=23.1%,* p_e *=1.28E-09*); and the pathway of *'2841'*, containing *50* GWAS genes, involves *8* significant genes from *simP* (*effect=10.9%,* p_e *=7.65E-04*) (Table 4).

To adjust for pathway dependence and multiple testing, the *enrichTest2_Perm()* function calculates the adjusted p-value (*p_perm*) by *1,000* permutations. The argument of *geneDF* is the data frame for enrichment test II by *enrichTest2()* function. The argument of *setType* defines the category of gene sets for permutation adjusting. The argument of *times* specifies the number of permutations for generating distribution table and the argument of *seed* assigns a random seed for permutation. The permutation adjusted p-value (*p_perm*) was shown in Table 4. The *p_perm* is <1E-3, <1E-3, 0.306 and <1E-3 for the most enriched pathways based on gene measures of *minP*, *2ndP, simP* and *fishP* respectively.

Table 4. The most enriched KEGG pathway of T2D-GWAS by enrichment analysis II

Measure	Genes	PID	size	setGenes	effect(%)	sd(%)	p_e	*p_perm*	*p_table*
minp	289	2901	69	17	17.8	3.0	4.63E-07	<1E-3	0.0003
2ndp	274	2901	69	19	21.0	3.0	5.80E-09	<1E-3	<1E-4
simp	214	2841	50	8	10.9	3.1	7.65E-04	0.306	0.2617
fishp	309	2901	69	21	23.1	3.1	1.28E-09	<1E-3	<1E-4

'Genes': the number of GWAS significant genes that is taken for enrichment analysis; 'PID': the pathway ID used by *snpGeneSets*. 'size': the number of GWAS genes of a pathway; 'setGenes': the number of GWAS significant genes contained by the pathway.

```
>  minp_dist =
enrichTest2_Perm(data.frame(gene_id=T2DGWASGene$gene_id,score=T2DGWASGene$minp),
setType=6,times=1000, seed=1)

>  sndp_dist =
enrichTest2_Perm(data.frame(gene_id=T2DGWASGene$gene_id,score=T2DGWASGene$sndp),
setType=6,times=1000, seed=1)

>  simp_dist =
enrichTest2_Perm(data.frame(gene_id=T2DGWASGene$gene_id,score=T2DGWASGene$simp),
setType=6,times=1000, seed=1)

>  fishp_dist =
enrichTest2_Perm(data.frame(gene_id=T2DGWASGene$gene_id,score=T2DGWASGene$fishp),
setType=6,times=1000, seed=1)

> minp_min=apply(minp_dist,2,min)

> sndp_min=apply(sndp_dist,2,min)

> simp_min=apply(simp_dist,2,min)

> fishp_min=apply(fishp_dist,2,min)

> KEGG_rst$p_perm<-c(sum(minp_min<=KEGG_rst[1,"p"]),sum(sndp_min<=KEGG_rst[2,"p"]),
                     sum(simp_min<=KEGG_rst[3,"p"]),sum(fishp_min<=KEGG_rst[4,"p"]))
```

To enable direct calculation of permutation p-value, a pre-generated distribution table based on *10,000* permutations is made [4] and *getEnrich2P()* function is provided to obtain the permutation p-value (*p_table*) directly. The ***p_table*** is ***3.00E-04***, ***<1E-4***, ***0.2617*** and ***<1E-4*** for the most enriched pathways based on gene measures of *minP*, *2ndP, simP* and *fishP* respectively (Table 4). The codes are shown below:

```
> KEGG_rst$p_table<-getEnrich2P(setP=KEGG_rst$p, setType=6)$perm$p
> KEGG_rst
```

8.2 *Example: Enrichment analysis II of T2D-GWES*

For type II enrichment analysis of GWES, differential expression p-value is typically used as measure of gene effect. As pathway analysis of GWAS, both gene measure and its calculated *U*-score can be applied to test pathway enrichment by *enrichTest2* function. For the example of T2D-GWES data, the default value of *U*-score threshold=*0.05* were used for enrichment test of KEGG pathways (i.e. *setType=6*).

```
> expGeneSets_KEGG<-
enrichTest2(data.frame(gene_id=T2DExpression$gene_id,score=uscore(T2DExpression$p)),
setType=6)
```

The *10* most enriched pathways for significant GWES genes were identified as below and results were saved to *exp_rst* variable.

```
> exp_rst<-expGeneSets_KEGG$enrich_test[order(expGeneSets_KEGG$enrich_test$pval),][1:10,]
```

The permutation test was applied to obtain permutation p-value (*p_perm*) by *enrichTest2_Perm()* function.

```
> exp_dist =
enrichTest2_Perm(data.frame(gene_id=T2DExpression$gene_id,p=uscore(T2DExpression$p)),
                              setType=6,times=1000, seed=1)
> exp_min=apply(exp_dist,2,min)
> exp_rst$p_perm<-unlist(lapply(exp_rst$pval, function(x) sum(exp_min<=x)/1000))
```

To enable direct calculation of permutation p-value, a pre-generated distribution table based on *10,000* permutations [4] is made and *getEnrich2P()* function is provided to obtain the permutation p-value (*p_table*) directly.

```
> exp_rst$p_table<-getEnrich2P(setP=exp_rst$pval, setType=6)$perm$p
> exp_rst
```

The results of enrichment analysis II for T2D-GWES were shown at Table 5, and *9* of them were also shown as the top *10* pathways by enrichment analysis I. The most enriched pathway is the same for both type *I* and *II* analysis, which is the pathway of '*2872'* with effect=*7.9%* and empirical p-value=*7.28E-03*. However, the *1,000* permutations got the *p_perm=0.889* and pre-generated distribution table showed the *p_table=0.9118*.

Table 5. Ten most enriched KEGG pathways of T2D-GWES by enrichment analysis II

PID	**size**	**setGenes**	**effect(%)**	**sd(%)**	***p_e***	***p_perm***	***p_table***
2872	52	7	7.90	3.18	7.28E-03	0.889	0.9118
2866	23	4	11.83	4.78	7.51E-03	0.912	0.9194
2869	23	4	11.83	4.78	7.51E-03	0.912	0.9194
2825	46	6	7.49	3.38	1.25E-02	0.988	0.9808
2803	259	22	2.94	1.42	1.61E-02	0.996	0.9918
2719	29	4	8.24	4.25	2.02E-02	0.998	0.9967
2751	40	5	6.94	3.62	2.17E-02	0.999	0.9977
2787	20	3	9.44	5.12	2.23E-02	0.999	0.998
2746	44	5	5.81	3.45	3.32E-02	1	0.9999
2874	34	4	6.21	3.93	3.78E-02	1	1

PID': the pathway ID used by *snpGeneSets*. 'size': the number of GWAS genes of a pathway; 'setGenes': the number of GWAS significant genes contained by the pathway.

9. Pathway Enrichment Analysis of GWAS by ALIGATOR

The ALIGATOR (Association LIst Go AnnoTatOR)[8] method is also implemented in the *snpGeneSets* package by the function *alligator().* The method tests pathway enrichment for GWAS significant gene that is defined through p-value threshold *pcut* of SNP association. The default value of *pcut* is 0.05 for *alligator()*, and any gene with a SNP p-value < *pcut* is defined as significant. The method applies permutation to obtain empirical unadjusted p-value and the number of permutation is defined through parameter *Nsample* that takes default value of *5000*. The adjusted p-value is obtained through bootstrap sampling and the number of bootstrapping is set through parameter Btimes that takes default value of *1000*.

The example below shows the analysis of pathway enrichment for T2DGWAS by ALIGATOR method. The first parameter *snpGeneP* is a data frame containing at least columns of '*snp*' (SNP rsid) , *'gene_id'* (Entrez gene ID) and '*p'* (SNP association p-value) . The data of *T2DGWAS* comes with the *snpGeneP* data frame and *pcut* of *0.001* is applied to test pathway enrichment.

```
> data(T2DGWAS)
>head(snpGeneP)
> path0=aligator(snpGeneP, pcut=0.001)
>  path0[order(path0$p),][1:10,]
```

```
        pid      p adj_p
4243 7717 0.0004 0.609
2133 5607 0.0010 0.827
3358 6832 0.0012 0.858
4270 7744 0.0014 0.886
28   2745 0.0022 0.953
4526 8000 0.0030 0.980
4106 7580 0.0032 0.983
4630 8104 0.0044 0.995
4164 7638 0.0046 0.997
16   2733 0.0054 0.999
```

It was shown that the first pathway with *pid=7717* has empirical unadjusted p-value of 4E-04, but the permutation adjusted p-value is 0.609.

References:

1. Sherry ST, Ward MH, Kholodov M, Baker J, Phan L, Smigielski EM, Sirotkin K: **dbSNP: the NCBI database of genetic variation**. *Nucleic Acids Res* 2001, **29**(1):308-311.
2. Maglott D, Ostell J, Pruitt KD, Tatusova T: **Entrez Gene: gene-centered information at NCBI**. *Nucleic Acids Res* 2011, **39**(Database issue):D52-57.
3. Liberzon A: **A description of the Molecular Signatures Database (MSigDB) Web site**. *Methods in molecular biology* 2014, **1150**:153-160.
4. Mei H, Li L, Liu S, Jiang F, Griswold M, Mosley T: **The uniform-score gene set analysis for identifying common pathways associated with different diabetes traits**. *BMC genomics* 2015, **16**(1):336.
5. Scott LJ, Mohlke KL, Bonnycastle LL, Willer CJ, Li Y, Duren WL, Erdos MR, Stringham HM, Chines PS, Jackson AU *et al*: **A genome-wide association study of type 2 diabetes in Finns detects multiple susceptibility variants**. *Science* 2007, **316**(5829):1341-1345.
6. Marselli L, Thorne J, Dahiya S, Sgroi DC, Sharma A, Bonner-Weir S, Marchetti P, Weir GC: **Gene expression profiles of Beta-cell enriched tissue obtained by laser capture microdissection from subjects with type 2 diabetes**. *PLoS One* 2010, **5**(7):e11499.
7. Smyth GK: **Linear models and empirical Bayes methods for assessing differential expression in microarray experiments.** . *Statistical Applications in Genetics and Molecular Biology 3, No 1, Article 3* 2004.
8. Holmans P, Green EK, Pahwa JS, Ferreira MA, Purcell SM, Sklar P, Owen MJ, O'Donovan MC, Craddock N: **Gene ontology analysis of GWA study data sets provides insights into the biology of bipolar disorder**. *American journal of human genetics* 2009, **85**(1):13-24.