GIGSEA: Genotype Imputed Gene Set Enrichment Analysis

Shijia Zhu
Department of Genetics and Genomic Sciences and Icahn Institute for Genomics and Multiscale Biology

August 28, 2017

Contents

Abstract

We presented the Genotype Imputed Gene Set Enrichment Analysis (GIGSEA), a novel method that uses GWAS-and-eQTL-imputed differential gene expression to interrogate gene set enrichment for the trait-associated SNPs. By incorporating eQTL from large gene expression studies, e.g. GTEx, GIGSEA appropriately addresses such challenges for SNP enrichment as gene size, gene boundary, andmultiple-marker regulation. The weighted linear regression model, taking as weights both imputation accuracy and model completeness, was used to test the enrichment, properly adjusting the bias due to redundancy in different gene sets. The permutation test, furthermore, is used to evaluate the significance of enrichment, whose efficiency can be largely elevated by expressing the computational intensive part in terms of large matrix operation. We have shown the appropriate type I error rates for GIGSEA (<5%), and the preliminary results also demonstrate its good performance to uncover the real biological signal.

1. Import packages

In GIGSEA, the gene sets are saved as matrices. Such matrices are largely sparse, so, in order to save space, we used the functions provided by the R package "Matrix" to build the sparse matrices and pre-saved into the GIGSEA package. In addition, GIGSEA uses the local fdr, implemented by the R package "locfdr", to adjust for multiple hypothesis testing.

```
library(GIGSEA)
```

```
## Loading required package: Matrix
```

```
## Loading required package: MASS
```

```
## Loading required package: locfdr
```

2. Quick start

GIGSEA first uses MetaXcan (also called "s-predixcan") to impute trait-associated differential gene expression from both GWAS summary and eQTL database with LD structure adjusted, and next, builds a weighted regression model to perfrom gene set enrichment analysis. In user's convenience, we combine these procedures together into one function `runGIGSEA()`, which writes the enrichment test results at the local directory. Users only need to provide their GWAS summary data, and specify the paths to the MetaXcan.py file, the eQTL database (e.g. GTex and DGN) and the reference popultation (e.g. 1000 Genome).

```
# runGIGSEA( MetaXcan="software/MetaXcan.py" ,
# model_db_path="eQTL/DGN-WB_0.5.db",
# covariance="reference/covariance.DGN-WB_0.5.txt.gz",
# gwas_folder="data/GWAS_summary", gwas_file_pattern="heart.sumstats",
# output_dir="result/GIGSEA", permutation_num=1000)
```

Alternatively, users can also submit their GWAS summary data to the online version of MetaXcan (s-predixcan) https://cloud.hakyimlab.org/s-predixcan to impute the trait-associated differential gene expression, and next, run the following steps provided by GIGSEA to perform the gene set enrichment analysis for the trait-associated SNPs.

***In case that users had neither installed MetaXcan nor run the online MetaXcan, an example of MetaXcan-imputed differential gene expression is given as follows. Users can use it to test the codes of GIGSEA step by step.

3. One example of MetaXcan output

MetaXcan integrates GWAS summary result with eQTL information to map trait-associated genes. See https://github.com/hakyimlab/MetaXcan. It provides a novel way to aggregate the multiple markers within each gene, address the long-range regulation, and adjust bias from gene boundaries and gene size. We use MetaXcan to impute the complex-trait-associated differential gene expression from eQTL summary and GWAS summary datasets. MetaXcan imputes ~10,000 genes with high quality prediction in most tissues. The training dataset for the expression prediction or 1000 Genomes was used as reference population to address the LD structure (covariance) of markers. Users can also specify their own genotype data to address the LD structure. The eQTL summary data was pre-calculated from large gene expression studies, such as the Genotype-Tissue Expression Project (GTEx; a comprehensive set of tissues from of ~20,000 samples) (Lonsdale, et al., 2013) and Depression Genes and Networks (DGN; 922 whole-blood samples) (Battle, et al., 2014). So, users only need to provide the GWAS summary data to estimate the genetically regulated gene expression.

We take as an example the cardiovascular disease (CVD) GWAS, CARDIoGRAMplusC4D (60,801 cases, 123,504 controls and 9.4M SNPs) (Nikpay, et al., 2015). The summary data is downloaded from <www.cardiogramplusc4d.org/data-downloads/>. We run the MetaXcan on it based on DGN eQTL database and 1000 Genomes as covariance.

```
data(heart.metaXcan)
head(heart.metaXcan)
```

```
##              gene gene_name    zscore effect_size       pvalue        var_g
## 1 ENSG00000134222     PSRC1 -8.857086 -0.13668948 8.213281e-19 0.1981241807
## 2 ENSG00000107798      LIPA  6.946756  0.08913688 3.737808e-12 0.2702763074
## 3 ENSG00000135148    TRAFD1  6.104691  2.80669383 1.030000e-09 0.0002579440
## 4 ENSG00000111249      CUX2 -6.074906 -2.76985557 1.240602e-09 0.0002799391
## 5 ENSG00000186063      AIDA  5.997636  0.37241430 2.002105e-09 0.0129998359
## 6 ENSG00000115486      GGCX  5.806570  0.06182678 6.376555e-09 0.4009731992
##   pred_perf_r2 pred_perf_pval pred_perf_qval n_snps_used n_snps_in_cov
## 1  0.251784300   5.893333e-60   4.019735e-59          11            11
## 2  0.608481823  1.560951e-189  6.547543e-188           8             8
```

```
## 3  0.003758508   6.277686e-02   8.269168e-02           1             1
## 4  0.005794522   2.079893e-02   2.853868e-02           2             2
## 5  0.020511033   1.269367e-05   2.143166e-05          11            11
## 6  0.459503100  4.710932e-125  8.726820e-124          24            24
##   n_snps_in_model
## 1              11
## 2               9
## 3               1
## 4               2
## 5              11
## 6              25
```

Each row is a gene's association result:

- `gene`: a gene's id
- `gene_name`: a gene's name
- `zscore`: MetaXcan's association result for the gene
- `effect_size`: MetaXcan's association effect size for the gene
- `pvalue`: P-value of the aforementioned statistic
- `pred_perf_r2`: R2 of transcriptome prediction model's correlation to gene's measured transcriptome
- `pred_perf_pval`: pval of transcriptome prediction model's correlation to gene's measured transcriptome
- `pred_perf_qval`: qval of transcriptome prediction model's correlation to gene's measured transcriptome
- `n_snps_used`: number of snps from GWAS that got used in MetaXcan analysis
- `n_snps_in_cov`: number of snps in the covariance matrix
- `n_snps_in_model`: number of snps in the prediction model
- `var_g`: variance of the gene expression

We use the linear regression model to build a threshold-free gene set enrichment test, checking whether genes are significantly differentially expressed in a given gene set, as compared to the background. In the regression model, we regress the imputed Z-score of differential gene expression on the gene sets. However, the gene expression cannot be perfectly predicted using genotype, which is indicated by the prediction R^2, and moreover, not all SNPs in the prediction model are in user's dataset. In order to take into account such two factors, we use as weights the multiplication of prediction R^2 and fraction of imputation-used SNPs, building a weighted linear regression model.

```r
gene = heart.metaXcan$gene_name
# extract the imputed Z-score of differential gene expression, which follows
# the normal distribution
fc <- heart.metaXcan$zscore
# use the prediction R^2 and fraction of imputation-used SNPs as weights
usedFrac <- heart.metaXcan$n_snps_used / heart.metaXcan$n_snps_in_model
r2 <- heart.metaXcan$pred_perf_r2
weights <- usedFrac*r2
# build a new data frame for the following weighted linear regression-based
# enrichment analysis
data <- data.frame(gene,fc,weights)
head(data)
```

```
##     gene        fc     weights
## 1  PSRC1 -8.857086 0.251784300
## 2   LIPA  6.946756 0.540872732
## 3 TRAFD1  6.104691 0.003758508
## 4   CUX2 -6.074906 0.005794522
## 5   AIDA  5.997636 0.020511033
## 6   GGCX  5.806570 0.441122976
```

4. Load data of gene sets

GIGSEA is built on the weighted linear regression model, so it permits both discrete-valued and continuous-valued gene sets. We already incorporated several gene sets into the GIGSEA package, including:

1) **discrete-valued gene sets**:

- `MSigDB.KEGG.Pathway`: Gene sets derived from the KEGG pathway database. It comprises 186 pathways (column) and 5267 genes (row). See c2.cp.kegg.v6.0.symbols.gmt.txt at http://software.broadinstitute.org/gsea/msigdb/collections.jsp#C2
- `MSigDB.TF`: Gene sets that share upstream cis-regulatory motifs which can function as potential transcription factor binding sites. It comprises 615 TFs (column) and 12774 genes (row). See c3.tft.v6.0.symbols.gmt.txt at http://software.broadinstitute.org/gsea/msigdb/collections.jsp#C3
- `MSigDB.miRNA`: Gene sets that contain genes sharing putative target sites (seed matches) of human mature miRNA in their 3'-UTRs. It comprises 221 miRNAs (column) and 7444 genes (row). See c3.mir.v6.0.symbols.gmt.txt at http://software.broadinstitute.org/gsea/msigdb/collections.jsp#C3
- `GO`: Gene sets that contain genes annotated by the same Gene Ontology (GO) term. For each GO term, we not only incorporate its own gene sets, but also incorporate the gene sets belonging to its offsprings. See the "database GO.db" in R.

2) **continuous-valued gene sets**:

- `Fantom5.TF`: The human transcript promoter locations were obtained from Fantom5. Based on the promoter locations, the tool MotEvo was used to predict the human transcriptional factor (TF) target sites. The dataset contains 500 Positional Weight Matrices (PWM) and 21964 genes. For each PWM, there is a list of associated human TFs, ordered by percent identity of TFs known to bind sites of the PWM. The list of associations was checked manually. The entire set of PWMs and mapping to associated TFs is available from the SwissRegulon website http://www.swissregulon.unibas.ch.
- `TargetScan.miRNA`: Gene sets of predicted human miRNA target sites were downloaded from TargetScan. TargetScan groups miRNAs that have identical subsequences at positions 2 through 8 of the miRNA, i.e. the 2-7 seed region plus the 8th nucleotide, and provides predictions for each such seed motif. TargetScan covers 87 human miRNA seed motifs in total. It provides a score for each seed motif and each RefSeq transcript, called preferential conservation scoring (aggregate Pct), which shows consistently high performance in various benchmark tests. To obtain a site count associated with each gene, we average the TargetScan Pct scores of all RefSeq transcripts associated with each gene. It comprises 87 miRNA seed motifs and 9861 genes. See http://www.targetscan.org.
- `LINCS.CMap.drug`: Large perturbational datasets of gene expression signature from small-molecule compounds in multiple cell types from LINCS/CMap database. We downloaded the data of LINCS phase 2 level 5 from GEO (GSE70138). The data is saved in the GCTx format (binary format based on HDF5 that enables fast i/o than text), and we parsed it using the R package cmapR. The LINCS level 5 data is a numeric matrix, comprising 118050 drugs/doses and 12328 genes. Each entry is a replicate-collapsed z-score of differential gene expression due to a drug perturbation, which is calculated by aggregating across individual replicates. See https://clue.io. In order to generate a single signature for each drug perturbation, we further average the differential gene expression across different drug doses, resulting in a condensed matrix of 1826 drugs (column) and 12328 genes (row).

4.1 Discrete-valued gene sets:

We first show an example of discrete-valued gene set: MSigDB.KEGG.Pathway, where the row represents the gene, and the column represents the pathway. Each entry takes discrete values of 0 or 1, where 1 represents the gene (row) belongs to the pathway (column), and otherwise, not.

```
# load data
data(MSigDB.KEGG.Pathway)
# MSigDB.KEGG.Pathway is a list comprising two components: net and annot
class(MSigDB.KEGG.Pathway)
```

```
## [1] "list"
```

```r
names(MSigDB.KEGG.Pathway)
```

```
## [1] "net"   "annot"
```

```r
dim(MSigDB.KEGG.Pathway$net)
```

```
## [1] 5267  186
```

```r
# the column is the pathway and the row is the gene
head(colnames(MSigDB.KEGG.Pathway$net))
```

```
## [1] "KEGG_GLYCOLYSIS_GLUCONEOGENESIS"
## [2] "KEGG_CITRATE_CYCLE_TCA_CYCLE"
## [3] "KEGG_PENTOSE_PHOSPHATE_PATHWAY"
## [4] "KEGG_PENTOSE_AND_GLUCURONATE_INTERCONVERSIONS"
## [5] "KEGG_FRUCTOSE_AND_MANNOSE_METABOLISM"
## [6] "KEGG_GALACTOSE_METABOLISM"
```

```r
head(rownames(MSigDB.KEGG.Pathway$net))
```

```
## [1] "A2M"    "A4GALT" "AACS"   "AADAT"  "AANAT"  "AARS"
```

```r
# the annotation of the pathway
head(MSigDB.KEGG.Pathway$annot)
```

```
##                                          term
## 1               KEGG_GLYCOLYSIS_GLUCONEOGENESIS
## 2                  KEGG_CITRATE_CYCLE_TCA_CYCLE
## 3                KEGG_PENTOSE_PHOSPHATE_PATHWAY
## 4 KEGG_PENTOSE_AND_GLUCURONATE_INTERCONVERSIONS
## 5          KEGG_FRUCTOSE_AND_MANNOSE_METABOLISM
## 6                     KEGG_GALACTOSE_METABOLISM
##                                                                                                link
## 1               http://www.broadinstitute.org/gsea/msigdb/cards/KEGG_GLYCOLYSIS_GLUCONEOGENESIS
## 2                  http://www.broadinstitute.org/gsea/msigdb/cards/KEGG_CITRATE_CYCLE_TCA_CYCLE
## 3                http://www.broadinstitute.org/gsea/msigdb/cards/KEGG_PENTOSE_PHOSPHATE_PATHWAY
## 4 http://www.broadinstitute.org/gsea/msigdb/cards/KEGG_PENTOSE_AND_GLUCURONATE_INTERCONVERSIONS
## 5          http://www.broadinstitute.org/gsea/msigdb/cards/KEGG_FRUCTOSE_AND_MANNOSE_METABOLISM
## 6                     http://www.broadinstitute.org/gsea/msigdb/cards/KEGG_GALACTOSE_METABOLISM
##   totalGenes
## 1         62
## 2         32
## 3         27
## 4         28
## 5         34
## 6         26
```

```r
# the net takes discrete values of 0 or 1
head(MSigDB.KEGG.Pathway$net[,1:30])
```

```
## 6 x 30 sparse Matrix of class "dgCMatrix"
```

```
##    [[ suppressing 30 column names 'KEGG_GLYCOLYSIS_GLUCONEOGENESIS', 'KEGG_CITRATE_CYCLE_TCA_CYCLE',
```

```
##
## A2M    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
## A4GALT . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
## AACS   . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
## AADAT  . . . . . . . . . . . . . . . . . . . . . . 1 . . . . 1 . . . . .
```

```
## AANAT  . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . .
## AARS   . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

4.2 Continuous-valued gene sets:

Followed is an example of continuous-valued gene set: TargetScan.miRNA, where the row represents the gene, and the column represents the miRNA. Each entry takes continuous values of pct, representing the binding affinity of miRNA on the gene 3' UTR.

```
# load data
data(TargetScan.miRNA)
# TargetScan.miRNA is a list comprising two components: net and annot
class(TargetScan.miRNA)
```

```
## [1] "list"
```

```
names(TargetScan.miRNA)
```

```
## [1] "net"   "annot"
```

```
dim(TargetScan.miRNA$net)
```

```
## [1] 9861   87
```

```
# the column is the miRNA and the row is the gene
head(colnames(TargetScan.miRNA$net))
```

```
## [1] "hsa-let-7a hsa-let-7b hsa-let-7c hsa-let-7d hsa-let-7e hsa-let-7f hsa-miR-98 hsa-let-7g hsa-let
## [2] "hsa-miR-1 hsa-miR-206 hsa-miR-613"
## [3] "hsa-miR-101"
## [4] "hsa-miR-103a hsa-miR-107"
## [5] "hsa-miR-10a hsa-miR-10b"
## [6] "hsa-miR-122"
```

```
head(rownames(TargetScan.miRNA$net))
```

```
## [1] "A1CF"  "A2LD1" "AAGAB" "AAK1"  "AAMP"  "AARS"
```

```
# the annotation of the miRNA
head(TargetScan.miRNA$annot)
```

```
##
## 49 hsa-let-7a hsa-let-7b hsa-let-7c hsa-let-7d hsa-let-7e hsa-let-7f hsa-miR-98 hsa-let-7g hsa-let-7
## 57                                                                                          hsa-mi
## 14
## 53
## 18
## 59
##     family totalGenes
## 49 GAGGUAG       1046
## 57 GGAAUGU        769
## 14 ACAGUAC        788
## 53 GCAGCAU        638
## 18 ACCCUGU        216
## 59 GGAGUGU        112
```

```
# the net takes continuous values
head(TargetScan.miRNA$net[,1:20])
```

```
## 6 x 20 sparse Matrix of class "dgCMatrix"
```

```
##    [[ suppressing 20 column names 'hsa-let-7a hsa-let-7b hsa-let-7c hsa-let-7d hsa-let-7e hsa-let-7f
##
## A1CF  0.000 . . . .     . .     . . . 0.331 .     .     . . . .     . . .
## A2LD1 .     . . . .     . .     . . . .     .     .     . . . .     . . .
## AAGAB .     . . . .     . .     . . . .     .     .     . . . .     . . .
## AAK1  0.922 . . . 0.581 0 0.836 . . . .     0.951 0.879 . . . 0.376 . . .
## AAMP  .     . . . .     . .     . . . .     .     .     . . . .     . . .
## AARS  .     . . . .     . .     . . . .     .     .     . . . .     . . .
```

4.3 User self-defined gene set

In adition to the above predefined gene sets, users can also specify their own gene set. Here, we take as an example the CREEDS <amp.pharm.mssm.edu/CREEDS/>. It is a manually curated database of gene signatures of single drug perturbations. For each drug perturbation, it lists both up-regulated and down-regulated gene sets. In the following example, we transform the gmt format file into a sparse matrix, where for each drug perturbation, the up-regulated genes take the value of 1, the down-regulated genes take the value of -1, and the others take the value of 0.

```
# downlaod the gmt file
gmt <- readLines( paste0('http://amp.pharm.mssm.edu/CREEDS/download/',
                         'single_drug_perturbations-v1.0.gmt') )
# obtain the index of up-regulated and down-regulated gene sets
index_up <- grep('-up',gmt)
index_down <- grep('-dn',gmt)
# transform the gmt file into gene sets. The gene set is a data frame,
# comprising three vectors: term (here is drug), geneset (a gene symbol list
# separated by comma), and value (1 and -1 separated by comma)
gff_up <- gmt2geneSet( gmt[index_up] , termCol=c(1,2) , singleValue = 1 )
gff_down <- gmt2geneSet( gmt[index_down] , termCol=c(1,2) , singleValue = -1 )

# as following, combine the up and down-regulated gene sets together,
# and use value of 1 and -1 to indicate their direction:
# extract the drug names
term_up <- sapply( gff_up$term , function(x) gsub('-up','',x) )
term_down <- sapply( gff_down$term , function(x) gsub('-dn','',x) )
all(term_up==term_down)
```

```
## [1] TRUE
```

```
# combine up and down-regulated gene names for each drug perturbation
geneset <- sapply( 1:nrow(gff_up) , function(i)
  paste(gff_up$geneset[i],gff_down$geneset[i],sep=',') )
# use 1 and -1 to indicate direction of up-regulated and down-regulated genes
value <- sapply( 1:nrow(gff_up) , function(i)
  paste(gff_up$value[i],gff_down$value[i],sep=',') )
# transform the gene set into matrix, where the row represents the gene,
# the column represents the drug perturbation, and each entry takes values of
# 1 and -1
net1 <- geneSet2Net( term=term_up , geneset=geneset , value=value )
# transform the gene set into sparse matrix, where the row represents the gene,
# the column represents the drug perturbation, and each entry takes values of
# 1 and -1
net2 <- geneSet2sparseMatrix( term=term_up , geneset=geneset , value=value )
tail(net1[,1:30])
```

```
##       Fluorouracil,drug:3639 Resveratrol,drug:3499 Citalopram,drug:3292
## cra                        0                     0                    0
## l7Rn6                      0                     0                    0
## mrpl11                     0                     0                    0
## mrpl24                     0                     0                    0
## mrpl9                      0                     0                    0
## ste2                       0                     0                    0
##       Fluorouracil,drug:3638 Ethanol,drug:3475 Resveratrol,drug:3498
## cra                        0                0                     0
## l7Rn6                      0                0                     0
## mrpl11                     0                0                     0
## mrpl24                     0                0                     0
## mrpl9                      0                0                     0
## ste2                       0                0                     0
##       Fluorouracil,drug:3637 Resveratrol,drug:3497 Sodium arsenite,drug:3357
## cra                        0                     0                        0
## l7Rn6                      0                     0                        0
## mrpl11                     0                     0                        0
## mrpl24                     0                     0                        0
## mrpl9                      0                     0                        0
## ste2                       0                     0                        0
##       Vitamin c,drug:3510 Vitamin c,drug:3511 Perfluorooctanoic acid,drug:3516
## cra                     0                   0                                0
## l7Rn6                   0                   0                                0
## mrpl11                  0                   0                                0
## mrpl24                  0                   0                                0
## mrpl9                   0                   0                                0
## ste2                    0                   0                                0
##       Perfluorooctanoic acid,drug:3517 Formaldehyde,drug:3550
## cra                                  0                     0
## l7Rn6                                0                     0
## mrpl11                               0                     0
## mrpl24                               0                     0
## mrpl9                                0                     0
## ste2                                 0                     0
##       Tretinoin,drug:3634 Vitamin e,drug:3515 Gefitinib,drug:3474
## cra                     0                   0                  0
## l7Rn6                   0                   0                  0
## mrpl11                  0                   0                  0
## mrpl24                  0                   0                  0
## mrpl9                   0                   0                  0
## ste2                    0                   0                  0
##       Methylprednisolone,drug:3668 Methylprednisolone,drug:3669
## cra                              0                            0
## l7Rn6                            0                            0
## mrpl11                           0                            0
## mrpl24                           0                            0
## mrpl9                            0                            0
## ste2                             0                            0
##       Levetiracetam,drug:2671 Rosiglitazone,drug:2672 Alitretinoin,drug:2673
## cra                         0                       0                     0
## l7Rn6                       0                       0                    -1
## mrpl11                      0                       0                     0
## mrpl24                      0                       0                     0
```

```
## mrpl9                        0                       0                       0
## ste2                         0                       0                       0
##        Tretinoin,drug:3233 Tretinoin,drug:3232 Tretinoin,drug:3231
## cra                      0                   0                   0
## l7Rn6                    0                   0                   0
## mrpl11                   0                   0                   0
## mrpl24                   0                   0                   0
## mrpl9                    0                   0                   0
## ste2                     0                   0                   0
##        Pristane,drug:3230 Tretinoin,drug:3237 Hypochlorous acid,drug:3198
## cra                      0                   0                          0
## l7Rn6                    0                   0                          0
## mrpl11                   0                   0                          0
## mrpl24                   0                   0                          0
## mrpl9                    0                   0                          0
## ste2                     0                   0                          0
##        Methoxychlor,drug:3235 4-vinylcyclohexene diepoxide,drug:3234
## cra                         0                                     0
## l7Rn6                       0                                     0
## mrpl11                      0                                     0
## mrpl24                      0                                     0
## mrpl9                       0                                     0
## ste2                        0                                     0
```

```
tail(net2[,1:30])
```

```
## 6 x 30 sparse Matrix of class "dgCMatrix"
##    [[ suppressing 30 column names 'Fluorouracil,drug:3639', 'Resveratrol,drug:3499', 'Citalopram,drug
##
## cra    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
## l7Rn6  . . . . . . . . . . . . . . . . . . . . . . . . -1 . . . . . . .
## mrpl11 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
## mrpl24 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
## mrpl9  . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
## ste2   . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

```
# the size of sparse matrix is much smaller than the matrix
format( object.size(net1), units = "auto")
```

```
## [1] "197.6 Mb"
```

```
format( object.size(net2), units = "auto")
```

```
## [1] "7.9 Mb"
```

5. Gene set enrichment analysis

5.1 Gene set enrichment analysis using weighted simple linear regression

After obtaining the imputed differential gene expression and the weights, we build the weighted linear regression model to investigate the gene set enrichment. Permutation test was used to adjust the p values of the regression coefficients. We repeatedly shuffle the differential gene expression to obtain a global null distribution of no associated gene sets and calculate the empirical p value for each gene set. For the Single Gene Set Erichment Analysis (SGSEA), especially with many gene sets tested, a large number of weighted simple linear regression model would be interrogated. To improve the efficiency, we use weighted Pearson correlation to rank the significance, which uses the same hypothesis statistic with the weighted linear regression

model. Furthermore, we expressed it in terms of large matrix inner product, substantially improving the time efficiency.

```
# take MSigDB.KEGG.Pathway as an example
net <- MSigDB.KEGG.Pathway$net
# intersect the permuted genes with the gene sets of interest
data2 <- orderedIntersect( x = data , by.x = data$gene , by.y = rownames(net) )
net2 <- orderedIntersect( x = net , by.x = rownames(net) , by.y = data$gene  )
all( rownames(net2) == as.character(data2$gene) )
```

```
## [1] TRUE
```

```
# the SGSEA.res1 uses the weighted simple linear regression model, while
# SGSEA.res2 used the weighted Pearson correlation. The latter one takes
# substantially less time.
system.time( SGSEA.res1 <- permutationSimpleLm( fc=data2$fc , net=net2 ,
                              weights=data2$weights , num=100 ) )
```

```
## 0%.....10%.....20%.....30%.....40%.....50%.....60%.....70%.....80%.....90%.....100%.
```

```
##    user  system elapsed
##  70.617   3.897  75.301
```

```
system.time( SGSEA.res2 <- permutationSimpleLmMatrix( fc=data2$fc ,
                              net=net2 , weights=data2$weights , num=100 ) )
```

```
## 0%.....10%.....20%.....30%.....40%.....50%.....60%.....70%.....80%.....90%.....100%.
```

```
##    user  system elapsed
##   0.230   0.008   0.239
```

```
head(SGSEA.res2)
```

```
##                                                term usedGenes observedCorr
## 9                           KEGG_STEROID_BIOSYNTHESIS        13   0.10579471
## 117                               KEGG_TIGHT_JUNCTION        81   0.09104426
## 105           KEGG_VASCULAR_SMOOTH_MUSCLE_CONTRACTION        70   0.08248315
## 67  KEGG_METABOLISM_OF_XENOBIOTICS_BY_CYTOCHROME_P450        27  -0.07912328
## 29                       KEGG_GLUTATHIONE_METABOLISM        36  -0.06984190
## 68               KEGG_DRUG_METABOLISM_CYTOCHROME_P450        23  -0.06442495
##     empiricalPval BayesFactor
## 9    0.0002741935  3813.47468
## 117  0.0009731183  1113.28648
## 105  0.0023172043   374.55825
## 67   0.0053817204    69.85393
## 29   0.0100591398     0.00000
## 68   0.0144677419     0.00000
```

5.2 Gene set enrichment analysis using weighted multiple regression model

A gene may function in multiple ways and thus appear multiple times in functional gene sets. In spite of reflecting the crosstalk between gene sets, such overlap may make the results of gene set analysis more difficult to interpret. To address the redundancy existing among gene sets, we build a weighted multiple linear regression model, taking into account all gene sets in one model. The redundancy of one gene set can be adjusted by considering all other gene sets as covariates. To improve the time efficiency, we also use matrix operation to solve the weighted multiple regression model.

```
# MGSEA.res1 uses the weighted multiple linear regression model
system.time( MGSEA.res1 <- permutationMultipleLm( fc=data2$fc , net=net2 ,
```

```
                              weights=data2$weights , num=1000 ) )
```

```
## 0%.....10%.....20%.....30%.....40%.....50%.....60%.....70%.....80%.....90%.....100%.
```

```
##    user  system elapsed
## 149.434   8.731 159.473
```

```
# MGSEA.res2 used the matrix solution
system.time( MGSEA.res2 <- permutationMultipleLmMatrix( fc=data2$fc ,
                              net=net2 , weights=data2$weights , num=1000 ) )
```

```
## 0%.....10%.....20%.....30%.....40%.....50%.....60%.....70%.....80%.....90%.....100%.
```

```
##    user  system elapsed
##   4.324   0.124   4.452
```

```
head(MGSEA.res2)
```

```
##                                       term usedGenes observedTstats
## 9                  KEGG_STEROID_BIOSYNTHESIS        13       6.478134
## 105 KEGG_VASCULAR_SMOOTH_MUSCLE_CONTRACTION        70       5.198424
## 8                KEGG_FATTY_ACID_METABOLISM        28       3.778185
## 60                 KEGG_RETINOL_METABOLISM        19      -3.554576
## 117                   KEGG_TIGHT_JUNCTION        81       3.405676
## 72                         KEGG_RIBOSOME        63       3.252726
##     empiricalPval BayesFactor
## 9    0.0001833333   814.54143
## 105  0.0013768817    52.26426
## 8    0.0114897849     0.00000
## 60   0.0162747312     0.00000
## 117  0.0189198925     0.00000
## 72   0.0231456989     0.00000
```

5.3 One-step weightedGSEA

In user's convenience, we combine the above procedures together into one function `weightedGSEA()`. Based on the imputed differential gene expression, `weightedGSEA()` checks multiple classes of gene sets simultaneously and writes out the enrichment analysis results. Users need to provide the same data with above, and specify the columns of gene names (geneCol), imputed differential gene expression (fcCol), and weights (weightCol). Additionally, users specify the gene sets of interest (geneSet), the times of permutation (permutationNum) and the directory for saving the results (outputDir). By default, we only do SGSEA. Users can specify `MGSEAthres` to perform MGMEA for those with less than `MGSEAthres` gene sets.

```
# import packages and prepare data as above
library(GIGSEA)

# prepare the dataset
data(heart.metaXcan)
gene = heart.metaXcan$gene_name
fc <- heart.metaXcan$zscore
usedFrac <- heart.metaXcan$n_snps_used / heart.metaXcan$n_snps_in_cov
r2 <- heart.metaXcan$pred_perf_r2
weights <- usedFrac*r2
data <- data.frame(gene,fc,weights)

# run one-step GIGSEA
#weightedGSEA(data, geneCol='gene', fcCol='fc', weightCol= 'weights',
```

```
#geneSet=c("MSigDB.KEGG.Pathway","Fantom5.TF","TargetScan.miRNA","GO"),
#permutationNum=10000, outputDir="./GIGSEA" )

#dir("./GIGSEA")
```