Supplemetal material of Uncovering causal gene-tissue pairs and variants: A multivariable TWAS method controlling for infinitesimal effects

Yihe Yang, Noah Lorincz-Comi, Xiaofeng Zhu

Real data analysis

Data structure

Here we illustrate a real example of how to perform TGVIS and also TGFM in real data analysis of the manuscript. This tutorial starts with the structures of involved data.

LD reference panel

The first dataset is the LD reference panel. This dataset is derived from the 9,680 unrelated individuals we described in the paper, selected from approximately 500,000 imputed individuals in the UK Biobank (UKBB). We refined the data using the bim files from these individuals. The files we shared on Google Drive include all 9.32 million SNPs involved; however, in this tutorial, we will focus only on a subset in the PCSK9 locus. Below is a glimpse of the data structure:

```
library(data.table)
library(dplyr)
variant=readRDS("variant.rds")
variant
```

```
##                        SNP CHR       BP  A1    A2      Freq MarkerName
##     1: 1:54005191_CCACA_C   1 54005191   C CCACA 0.3992490 1:54005191
##     2:    1:54012271_AG_A   1 54012271  AG     A 0.6292130 1:54012271
##     3:    1:54012621_TG_T   1 54012621  TG     T 0.6295230 1:54012621
##     4:    1:54035956_TA_T   1 54035956   T    TA 0.1912090 1:54035956
##     5:   1:54070614_TAC_T   1 54070614 TAC     T 0.9896660 1:54070614
##    ---
## 10782:          rs9919296   1 54578136   C     T 0.0209929 1:54578136
## 10783:          rs9919314   1 54579134   C     T 0.0209763 1:54579134
## 10784:           rs993075   1 56994938   G     C 0.4923560 1:56994938
## 10785:          rs9970807   1 56965664   T     C 0.0921541 1:56965664
## 10786:           rs998154   1 55596384   C     T 0.1690460 1:55596384
```

In this reference panel, `SNP` is the identifier for the variants; `CHR` represents the chromosome; `BP` indicates the base pair position in the hg19 genome build; `A1` is the effect allele as specified in the BED file; `A2` is the other allele; `Freq` denotes the frequency of the effect allele; and `MarkerName` is another unique identifier for the variants in the format CHR:BP. It is important to note that some variants in the UKBB bed file do not have an rsID. For these variants, their `SNP` are in the format CHR:BP:A2:A1.

GWAS summary data

The second dataset is the GWAS summary data, which should include at least the following columns: `SNP`, `A1`, `A2`, `Zscore`, and `N`. In this dataset, `Zscore` represents the Z-score of the marginal effect size estimates from the outcome GWAS, while `N` denotes the sample size. Other statistics can be deduced from `Zscore` and `N`, e.g.,

$$\texttt{BETA} = \frac{\texttt{Zscore}}{\sqrt{\texttt{N}}}, \quad \texttt{SE} = \frac{1}{\sqrt{\texttt{N}}}.$$

Below is an example of the dataset's structure:

```
library(arrow)
LDL=read_parquet("LDL.parquet")
LDL[-5,]
```

```
##                    SNP CHR       BP  A1  A2      Zscore       N
##    1: 1:54388067_CAA_C   1 54388067   C CAA  -0.9536883  389563
##    2: 1:54389774_TAG_T   1 54389774 TAG   T  -0.7761577  487566
##    3:  1:54401303_CT_C   1 54401303   C  CT   1.0402465 1041795
##    4: 1:54406627_ATT_A   1 54406627   A ATT  -3.1568909 1065377
##    5:  1:54430681_GT_G   1 54430681   G  GT  -2.6594631 1058084
##   ---
## 7958:        rs9919142   1 54579153   G   A   1.0001670 1230988
## 7959:        rs9919295   1 54578100   C   T   0.9664437 1231010
## 7960:        rs9919296   1 54578136   C   T   0.9008325 1231009
## 7961:        rs9919314   1 54579134   C   T   0.9813689 1231009
## 7962:         rs998154   1 55596384   C   T -10.8367174 1231250
```

It should be noted that we did not use the original `SNP` identifiers from the GWAS. Instead, we merged the GWAS data with the variant file using the `MarkerName` (CHR:BP) identifiers, and then assigned `SNP` from the variant file to the corresponding entries in the GWAS data. In cases where the GWAS file is based on the hg38 genome build, we use LiftOver to convert it to hg19.

eQTL and sQTL summary data

The next dataset, which has a more complex structure, is the summary data for eQTL and sQTL. We preprocessed the data provided by GTEx and other studies to retain only the following columns: `SNP`, `CHR`, `BP`, `A1`, `A2`, `P`, `Zscore`, `N`, `Gene`, `GeneSymbol`, `Tissue`, `Variable`, and `xQTL`. Here,

- `P`: the P-value of the marginal effect for the xQTL;
- `Gene`: the Ensembl ID of the gene associated with the xQTL;
- `GeneSymbol`: the symbol for the corresponding gene,
- `Tissue`: the tissue in which the gene was sequenced;
- `Variable`: the combination of GeneSymbol (or Gene for sQTL) + Tissue
- `xQTL`: an indicator of sQTL or eQTL.

An example of the dataset structure is shown below:

```r
eQTLsQTL=read_parquet("eQTLsQTL.parquet")%>%
dplyr::select(SNP,CHR,BP,A1,A2,P,Zscore,N,Gene,GeneSymbol,Tissue,Variable,xQTL)
eQTLsQTL
```

```
##                  SNP CHR       BP A1 A2          P     Zscore   N
##        1: rs12090789   1 54387577  T  C 0.78223604 -0.2765702 124
##        2: rs12090789   1 54387577  T  C 0.58952588  0.5399070 124
##        3: rs12090789   1 54387577  T  C 0.54357336  0.6078765 124
##        4: rs12090789   1 54387577  T  C 0.35021166 -0.9351431 124
##        5: rs12090789   1 54387577  T  C 0.59624067 -0.5301882 124
##       ---
## 15168101:  rs6680237   1 56586870  G  C 0.34938904 -0.9375335 311
## 15168102:  rs6680237   1 56586870  G  C 0.59527600 -0.5316440 420
## 15168103:  rs6680237   1 56586870  G  C 0.86508400 -0.1700180 420
## 15168104:  rs6680237   1 56586870  G  C 0.60456600 -0.5182680 420
## 15168105:  rs6680237   1 56586870  G  C 0.00636787 -2.7430910 420
##                                                          Gene   GeneSymbol
##        1: chr1:52906611:52907868:clu_52882:ENSG00000121310.16 ECHDC2(sQTL)
##        2: chr1:52917615:52921553:clu_52883:ENSG00000121310.16 ECHDC2(sQTL)
##        3: chr1:52914095:52921553:clu_52883:ENSG00000121310.16 ECHDC2(sQTL)
##        4: chr1:52914095:52917544:clu_52883:ENSG00000121310.16 ECHDC2(sQTL)
##        5: chr1:52914095:52916465:clu_52883:ENSG00000121310.16 ECHDC2(sQTL)
##       ---
## 15168101:                                       ENSG00000173406         DAB1
## 15168102:                                       ENSG00000162407        PLPP3
## 15168103:                                       ENSG00000162409       PRKAA2
## 15168104:                                       ENSG00000187889         FYB2
## 15168105:                                       ENSG00000021852          C8B
##                         Tissue
##        1: Adipose_Subcutaneous
##        2: Adipose_Subcutaneous
##        3: Adipose_Subcutaneous
##        4: Adipose_Subcutaneous
##        5: Adipose_Subcutaneous
##       ---
## 15168101:          Kidney_Tube
## 15168102:               Islets
## 15168103:               Islets
## 15168104:               Islets
## 15168105:               Islets
##                                                                                Variable
##        1: chr1:52906611:52907868:clu_52882:ENSG00000121310.16+Adipose_Subcutaneous
##        2: chr1:52917615:52921553:clu_52883:ENSG00000121310.16+Adipose_Subcutaneous
##        3: chr1:52914095:52921553:clu_52883:ENSG00000121310.16+Adipose_Subcutaneous
##        4: chr1:52914095:52917544:clu_52883:ENSG00000121310.16+Adipose_Subcutaneous
##        5: chr1:52914095:52916465:clu_52883:ENSG00000121310.16+Adipose_Subcutaneous
##       ---
## 15168101:                                                          DAB1+Kidney_Tube
## 15168102:                                                              PLPP3+Islets
## 15168103:                                                             PRKAA2+Islets
## 15168104:                                                               FYB2+Islets
## 15168105:                                                                C8B+Islets
##           xQTL
```

```
##         1: sQTL
##         2: sQTL
##         3: sQTL
##         4: sQTL
##         5: sQTL
##        ---
## 15168101: eQTL
## 15168102: eQTL
## 15168103: eQTL
## 15168104: eQTL
## 15168105: eQTL
```

It should be noted that entries in the format chr1:52906611:52907868:clu_52882:ENSG00000121310.16 indicate a specific splicing event for the gene ENSG00000121310, as defined by LeafCutter.

In practice, combining multiple sQTL and eQTL datasets is a challenging task. In the section where we explain the key functions, we will detail our strategy for handling and integrating these datasets.

LD refernece panel with individual

We used the UKBB BED file to estimate the LD reference, with a sample size of 9,680. Below is a glimpse of the data structure:

```
UKBBGenotype=readRDS("UKBBGenotype.rds")
UKBBGenotype[1:10,1:5]
```

```
##    rs10047036 rs12728734 rs150256195 rs74510493 rs114570917
## 1           2          2           2          2           2
## 2           2          1           2          2           2
## 3           2          2           2          2           2
## 4           1          1           2          2           2
## 5           2          2           2          2           2
## 6           2          2           2          2           2
## 7           2          2           2          2           2
## 8           2          2           2          2           1
## 9           1          2           2          2           2
## 10          2          2           2          2           2
```

Step-by-step analysis

Allele harmonisation

In the first step, we adjust the direction of the Z-scores in the GWAS and xQTL summary data to ensure that the effect alleles in these datasets match the effect alleles in our reference panel. This step is crucial because the LD matrix is estimated from this reference panel, and accurate LD estimation is fundamental to all statistical methods based on GWAS summary data. We wrote a function, `allele_harmonise()` in the R package `TGVIS`, to perform this step:

```
library(TGVIS)
LDL=allele_harmonise(ref_panel=variant[,c("SNP","A1","A2")],gwas_data=LDL)
eQTLsQTL=allele_harmonise(ref_panel=variant[,c("SNP","A1","A2")],gwas_data=eQTLsQTL)
eQTLsQTL=eQTLsQTL[LDL,nomatch=0]
```

```
setnames(eQTLsQTL,"i.Zscore","Zscore.y")
setnames(eQTLsQTL,"Zscore","Zscore.x")
```

In `allele_harmonise()`, we automatically set `gwas_data` as a data.table with `key="SNP"`, allowing `eQTLsQTL=eQTLsQTL[LDL, nomatch=0]` to efficiently merge the two datasets. The reason for merging these datasets, as described in here, there is a vast and heterogeneous landscape of publicly available GWAS. These studies were conducted on different genotyping platforms, using different imputation schemes, and defined on different releases of the human genome., is that there is a vast and heterogeneous landscape of publicly available GWAS. These studies were conducted on different genotyping platforms, using different imputation schemes, and defined on different releases of the human genome. Therefore, we aim to find the common variants between the GWAS and xQTL summary data to perform the analysis.

Extracting the moderately correlated variants

The next step is to remove highly correlated variants using C+T. Although SuSiE can group highly correlated or statistically duplicated variants into a single group and assign them one effect, including many redundant variants can significantly increase the dimensionality of the model. Therefore, primarily to enhance computational efficiency, we recommend retaining only moderately correlated variants.

We use the smallest p-value across all tissue pairs corresponding to each variant as the input p-value for PLINK to extract a subset of moderately correlated variants. While we will not execute the following steps in this tutorial, we will provide the code for you. You can modify the file paths as needed for your own data.

```
A=eQTLsQTL%>%dplyr::select(SNP,P)
A=A[, .SD[which.min(P)], by=SNP]
A=A[which(A$P<5e-4),]
write.table(A,"Your_path.txt",quote=F, sep="\t", row.name=F)
setwd("Your_path_to_PLINK")
system("./plink --bfile Your_bed_file --clump Your_path.txt
            --clump-field P --clump-kb 1000 --clump-p1 1e-5
            --clump-p2 1e-5 --clump-r2 0.5 --out Your_path")
setwd("Your_analysis_path")
plinkfile=fread("Your_path.clumped")
plinkfile=plinkfile$SNP
```

The most important part in this step is:

- `-clump-kb 1000`: we consider the window size to be 1M,
- `-clump-p1 1e-5`: we use the threshold of 1E-5,
- `-clump-r2 0.5`: the correlation between two variants is in the range $(-\sqrt{0.5}, \sqrt{0.5})$.

Since direct causal variants might not be linked to any gene-tissue pairs, I perform clumping on the outcome GWAS to identify outcome-associated variants:

```
A1=LDL[which(LDL$SNP%in%unique(eQTLsQTL$SNP)),]
A1$P=pchisq(A1$Zscore^2,1,lower.tail=F);
A1=A1[,c("SNP","P")]
A1=A1[which(A1$P<min(5e-8,quantile(A1$P,0.1))),]
write.table(A1,"Your_path.txt",quote=F, sep="\t", row.name=F)
setwd("Your_path_to_PLINK")
```

```
system("./plink --bfile Your_bed_file --clump Your_path.txt
            --clump-field P --clump-kb 1000 --clump-p1 1e-5
            --clump-p2 1e-5 --clump-r2 0.5 --out Your_path")
setwd("Your_analysis_path")
plinkfile1=fread("Your_path.clumped")
plinkfile1=plinkfile1$SNP
```

Finally, we merge these two lists of variants and use C+T to remove any highly correlated variants (which are typically few), resulting in the final pool of variants for analysis:

```
A=data.frame(SNP=unique(c(plinkfile,plinkfile1)))
A$P=0.05
write.table(A,"Your_path.txt",quote=F, sep="\t", row.name=F)
setwd("Your_path_to_PLINK")
system("./plink --bfile Your_bed_file --clump Your_path.txt
            --clump-field P --clump-kb 1000 --clump-p1 1e-5
            --clump-p2 1e-5 --clump-r2 0.5 --out Your_path")
setwd("Your_analysis_path")
plinkfile=fread("Your_path.clumped")
rsid=plinkfile$SNP
```

We have recorded this pool of variants in the PCSK9 locus:

```
rsid=readRDS("SNP_lowLD.rds")
gwas_eQTLsQTL=eQTLsQTL[which(eQTLsQTL$SNP%in%rsid),]
```

Regularization of LD matrix

Our next step is to estimate a "good" LD matrix. We use the POET-shrinkage method (**?**), as described in this paper, to estimate such an LD matrix. The code is as follows:

```
R0=cor(UKBBGenotype)
R0[is.na(R0)]=0;diag(R0)=1
R0=poet_shrinkage(R0)
R0=(t(R0)+R0)/2
genosnp=colnames(UKBBGenotype)
rownames(R0)=colnames(R0)=genosnp
```

Construction of design matrix of gene-tissue pairs

Our next step is to extract the design matrix of gene-tissue pairs from the `eQTLsQTL` data.table. We provide the function `make_design_matrix()`, which converts the Z-scores in `eQTLsQTL` into an $M \times p$ design matrix, where M is the number of variants and p is the total number of gene-tissue pairs:

```
bX0=make_design_matrix(eQTLsQTL[,c("SNP","Variable","Zscore.x")])
bX0=bX0[genosnp,]
```

In data.table `eQTLsQTL`, `Zscore.x` represents the Z-score of the xQTL effect, while `Zscore.y` is the Z-score from the outcome GWAS. We match the rows of `bX0` with the LD matrix using the code `bX0=bX0[genosnp,]`.

Our next step is to impute missing values in the Z-scores as 0. Since GTEx only provides the marginal xQTL effect sizes for variants near the gene's TSS, this results in missing values. Before imputing, we remove variants and gene-tissue pairs with excessive missing values. In this analysis, we exclude variants with more than 95% missing values and genes with more than 90% missing values:

```
bX=remove_missing_row_column(bX0,rowfirst=F,rowthres=0.95,colthres=0.9)
genosnp=rownames(bX)
R0=R0[genosnp,genosnp]
bX=as.matrix(bX[genosnp,])
bX[is.na(bX)]=0
```

eQTL selection

Our next step is to use SuSiE for gene-tissue pair eQTL selection. The first step is to extract the average sample size for each gene-tissue pair from eQTLsQTL to use as the input sample size for SuSiE:

```
VariableName=unique(eQTLsQTL$Variable)
NeQTLsQTL=eQTLsQTL[,.(NeQTLsQTL=mean(N)),by=Variable][Variable%in%VariableName,NeQTLsQTL]
names(NeQTLsQTL)=VariableName
NeQTLsQTL=NeQTLsQTL[colnames(bX)]
```

We have encapsulated a for-loop function based on `susie_rss()` to perform eQTL selection for each gene-tissue pair:

```
fiteQTL=eQTLmapping_susie(bX=bX,LD=R0,Nvec=NeQTLsQTL,L=3,pip.thres=0.5,pip.min=0.25)
```

```
##   |                                                                      |
```

```
bXest=fiteQTL$Estimate
ind=which(colSums(abs(bXest))==0)
bXest=bXest[,-ind]
bX=bX[,-ind]
NeQTLsQTL=NeQTLsQTL[colnames(bX)]
```

Performing S-Predixcan and its modifier to remove noise gene-tissue pairs

In this example, our original design matrix includes $M = 381$ variants and $p = 4878$ gene-tissue pairs:

```
dim(bX0)
```

```
## [1]  381 4878
```

After quality control and eQTL selection, we retained $p = 664$ gene-tissue pairs.

```
dim(bX)
```

```
## [1] 379 664
```

However, in many cases, the original number of gene-tissue pairs could be tens of thousands, and even after eQTL selection, there could still be thousands of gene-tissue pairs. Therefore, we perform a univariable TWAS with S-PrediXcan and its modifier to pre-reduce the dimensionality, making TGVIS and TGFM more efficient.

Let's first organize the data and set up a data frame to store the results:

```
bY=eQTLsQTL[,c("SNP","Zscore.y")]
bY=bY[!duplicated(bY$SNP),]
rownames(bY)=bY$SNP
bY=bY[genosnp,]
UVTWAS=matrix(0,ncol(bXest),6)
colnames(UVTWAS)=c("Variable","Type","Est1","P1","Est2","P2")
UVTWAS=as.data.frame(UVTWAS)
UVTWAS[,1]=colnames(bXest)
UVTWAS[,2]=eQTLsQTL$xQTL[match(UVTWAS[,1],eQTLsQTL$Variable)]
UVTWAS[,c(4,6)]=1
```

Next, we execute S-PrediXcan and its modifier:

```
for(i in 1:ncol(bXest)){
errorindicate=0
tryCatch({
bxx=bX[,i]
indx=which(bxx!=0)
bx=bXest[indx,i]
by=bY$Zscore.y[indx]
bxx=bxx[indx]
if(sum(bx!=0)==1){
pleiotropy.rm=which(bx!=0)
}else{
pleiotropy.rm=NULL
}
fitModifier=modified_predixcan(by=by,bxest=bx,LD=R0[indx,indx],
            pleiotropy.rm=pleiotropy.rm,tauvec=seq(3,21,by=3))
fitSpredixcan=modified_predixcan(by=by,bxest=bx,LD=R0[indx,indx],
            pleiotropy.rm=pleiotropy.rm,tauvec=10000000)
UVTWAS[i,3]=fitSpredixcan$theta
UVTWAS[i,4]=pchisq(fitSpredixcan$theta^2/fitSpredixcan$covtheta,1,lower.tail=F)
UVTWAS[i,5]=fitModifier$theta
UVTWAS[i,6]=pchisq(fitModifier$theta^2/fitModifier$covtheta,1,lower.tail=F)
}, error=function(e){
errorindicate=1
})
if(errorindicate == 1) next
}
```

The structure of `UVTWAS` is:

```
head(UVTWAS)
```

```
##                  Variable Type        Est1        P1       Est2        P2
## 1 ACOT11+Adipose_Visceral eQTL -0.49957076 0.7895277 -1.0746827 0.2311542
```

```
## 2    ACOT11+Adrenal_Gland eQTL -2.39903575 0.7147184 -2.9340518 0.3592333
## 3     ACOT11+Artery_Aorta eQTL  0.17665197 0.9181448  0.2538559 0.7569147
## 4    ACOT11+Artery_Tibial eQTL -0.02026495 0.9829979  0.1955781 0.6705236
## 5       ACOT11+Cerebellum eQTL -0.93448220 0.8659363  0.5692827 0.8332270
## 6           ACOT11+Cortex eQTL -0.40462949 0.8721248 -0.2022807 0.8692957
```

As described in the manuscript, we only consider data with an P-value greater than 0.5. We systematically scan each locus of the GWAS trait.

```
Genelist=UVTWAS%>%mutate(pvthres=0.05) %>%
dplyr::filter(P1 < pvthres | P2 < pvthres) %>%
pull(Variable)%>%unique()
```

This leaves $p = 52$ gene-tissue pairs:

```
length(Genelist)
```

```
## [1] 52
```

We then organize the data:

```
bX=bX[,Genelist]
bXest=bXest[,Genelist]
bY=eQTLsQTL[,c("SNP","Zscore.y")]%>%as.data.frame(.)
bY=bY[!duplicated(bY$SNP),]
rownames(bY)=bY$SNP
bY=bY[genosnp,]
```

Performing TGFM and TGVIS

The main procedures for TGFM and TGVIS are relatively straightforward, as shown below:

```
fittgfm=tgfm(by=bY$Zscore.y,bX=bX,LD=R0, Nvec=c(mean(LDL$N),NeQTLsQTL[Genelist]),
         causal.sampling.time=100,eqtl.sampling.time=25,L.causal=10)
```

Before executing TGVIS, we do not recommend removing potential candidates for direct causal variants. Specifically, if a gene is solely an eQTL, we suggest not including these eQTLs as candidates for direct causal variants. We have defined a function, `findUniqueNonZeroRows()`, to identify the indices of these variants. The code for this function will be provided in GitHub.

Next, we execute TGVIS:

```
fittgvis=tgvis(by=bY$Zscore.y,bXest=bXest,LD=R0,Noutcome=mean(LDL$N),
                  L.causal.vec=c(1:10),varinf.upper.boundary=0.25,pip.min=0.1,
                  pleiotropy.rm=findUniqueNonZeroRows(bXest))
```

It should be pointed out that TGVIS requires the esimates of the joint xQTL effect `bXest` while TGFM requires the marginal xQTL effect estimate `bX`.

Finally, we organize the results, which is a bit more complex. The principle is to retain only those gene-tissue pairs and direct variants that are included in the 95% credible set. As for TGFM:

```
thetagamma=c(fittgfm$theta[which(fittgfm$theta!=0)],
         fittgfm$gamma[which(fittgfm$gamma!=0)])
if(length(thetagamma)>0){
se.mrjones=c(fittgfm$theta.se[which(fittgfm$theta!=0)],
         fittgfm$gamma.se[which(fittgfm$gamma!=0)])
pip.mrjones=c(fittgfm$theta.pip[which(fittgfm$theta!=0)],
          fittgfm$gamma.pip[which(fittgfm$gamma!=0)])
pratt.mrjones=c(fittgfm$theta.pratt[which(fittgfm$theta!=0)],
            fittgfm$gamma.pratt[which(fittgfm$gamma!=0)])
cs.mrjones=c(fittgfm$theta.cs[which(fittgfm$theta!=0)],
         fittgfm$gamma.cs[which(fittgfm$gamma!=0)])
cs.pip.mrjones=c(fittgfm$theta.cs.pip[which(fittgfm$theta!=0)],
             fittgfm$gamma.cs.pip[which(fittgfm$gamma!=0)])
TGFMResult=data.frame(Variable=names(thetagamma),estimate=thetagamma,
                  se=se.mrjones,pip=pip.mrjones,pratt=pratt.mrjones,
                  cs=cs.mrjones,cs.pip=cs.pip.mrjones)
TGFMResult$Type=c(rep("TissueGene",length(which(fittgfm$theta!=0))),
              rep("SNP",length(which(fittgfm$gamma!=0))))
TGFMResult$CHR=1
TGFMResult$BP=55.5e5
TGFMResult=TGFMResult%>%group_by(cs)%>%mutate(cs.pratt=sum(pratt))%>%ungroup()
TGFMResult=TGFMResult%>%group_by(cs)%>%mutate(cs.pip=sum(pip))%>%ungroup()
cs0=which(TGFMResult$cs==0)
if(length(cs0)>0){
TGFMResult$cs.pratt[cs0]=TGFMResult$pratt[cs0]
TGFMResult$cs.pip[cs0]=TGFMResult$pip[cs0]
}
}else{
TGFMResult=NULL
}
TGFMResult$xQTL=TGFMResult$Variable
if(sum(TGFMResult$Type=="TissueGene")>0){
TGFMResult$xQTL[which(TGFMResult$Type=="TissueGene")]=get_nonzero_rows(fittgfm$bXest,
               TGFMResult$Variable[which(TGFMResult$Type=="TissueGene")])$NonzeroRows
}
row.names(TGFMResult)=NULL
TGFMResult=TGFMResult%>%dplyr::select(Variable,cs,cs.pip,cs.pratt,xQTL,
                 CHR,BP,Type,estimate,se,pip,pratt)%>%arrange(.,cs,Type,Variable)
print(TGFMResult)
```

```
## # A tibble: 11 x 12
##    Variable    cs cs.pip cs.pratt xQTL    CHR     BP Type  estimate     se   pip
##    <chr>    <dbl>  <dbl>    <dbl> <chr> <dbl>  <dbl> <chr>    <dbl>  <dbl> <dbl>
##  1 rs11591~     1  1      0.524   rs11~     1 5.55e6 SNP     -76.6  1.02   1
##  2 PCSK9+W~     2  1      0.204   rs12~     1 5.55e6 Tiss~     6.46 0.258  1
##  3 rs11206~     3  1      0.0387  rs11~     1 5.55e6 SNP      18.9  0.352  1
##  4 rs11546~     4  1      0.0129  rs11~     1 5.55e6 SNP      -1.89 0.0411 0.121
##  5 chr1:54~     4  1      0.0129  rs11~     1 5.55e6 Tiss~   -10.5  0.228  0.879
##  6 rs24954~     5  1.00   0.0376  rs24~     1 5.55e6 SNP      15.3  0.666  1.00
##  7 rs15011~     6  1.00   0.0220  rs15~     1 5.55e6 SNP      13.4  0.748  1.00
##  8 Affx-52~     7  0.987  0.00873 Affx~     1 5.55e6 SNP     -10.8  0.781  0.987
##  9 rs39767~     8  1     -0.0259  rs39~     1 5.55e6 SNP      17.6  1.61   1
## 10 rs24793~     9  0.714  0.0179  rs24~     1 5.55e6 SNP       9.30 4.86   0.714
```

```
## 11 rs26472~    10  0.940 -0.00298 rs26~     1 5.55e6 SNP       9.75 2.37   0.940
## # i 1 more variable: pratt <dbl>
```

As for TGVIS:

```
thetagamma=c(fittgvis$theta[which(fittgvis$theta!=0)],
         fittgvis$gamma[which(fittgvis$gamma!=0)])
if(length(thetagamma)>0){
se.mrjones=c(fittgvis$theta.se[which(fittgvis$theta!=0)],
         fittgvis$gamma.se[which(fittgvis$gamma!=0)])
pip.mrjones=c(fittgvis$theta.pip[which(fittgvis$theta!=0)],
          fittgvis$gamma.pip[which(fittgvis$gamma!=0)])
pratt.mrjones=c(fittgvis$theta.pratt[which(fittgvis$theta!=0)],
            fittgvis$gamma.pratt[which(fittgvis$gamma!=0)])
cs.mrjones=c(fittgvis$theta.cs[which(fittgvis$theta!=0)],
         fittgvis$gamma.cs[which(fittgvis$gamma!=0)])
cs.pip.mrjones=c(fittgvis$theta.cs.pip[which(fittgvis$theta!=0)],
             fittgvis$gamma.cs.pip[which(fittgvis$gamma!=0)])
TGVIResult=data.frame(Variable=names(thetagamma),estimate=thetagamma,
                  se=se.mrjones,pip=pip.mrjones,pratt=pratt.mrjones,
                  cs=cs.mrjones,cs.pip=cs.pip.mrjones)
TGVIResult$Type=c(rep("TissueGene",length(which(fittgvis$theta!=0))),
              rep("SNP",length(which(fittgvis$gamma!=0))))
TGVIResult$CHR=1
TGVIResult$BP=55.5e6
TGVIResult=TGVIResult%>%group_by(Type, cs)%>%mutate(cs.pratt=sum(pratt))%>%ungroup()
}else{
TGVIResult=NULL
}
TGVIResult$xQTL=TGVIResult$Variable
if(sum(TGVIResult$Type=="TissueGene")>0){
TGVIResult$xQTL[which(TGVIResult$Type=="TissueGene")]=
get_nonzero_rows(bXest,TGVIResult$Variable[which(TGVIResult$Type=="TissueGene")])$NonzeroRows
}
row.names(TGVIResult)=NULL
TGVIResult=TGVIResult%>%dplyr::select(Variable,cs,cs.pip,cs.pratt,xQTL,
          CHR,BP,Type,estimate,se,pip,pratt)%>%arrange(.,cs,Type,Variable)
print(TGVIResult)
```

```
## # A tibble: 3 x 12
##   Variable      cs cs.pip cs.pratt xQTL    CHR     BP Type  estimate    se   pip
##   <chr>      <dbl>  <dbl>    <dbl> <chr> <dbl>  <dbl> <chr>    <dbl> <dbl> <dbl>
## 1 rs11591147     1      1   0.492  rs11~     1 5.55e7 SNP     -71.8  1.02      1
## 2 PCSK9+Who~     2      1   0.170  rs12~     1 5.55e7 Tiss~     5.61 0.149     1
## 3 rs11206517     3      1   0.0398 rs11~     1 5.55e7 SNP      19.4  0.974     1
## # i 1 more variable: pratt <dbl>
```

The Pratt indices of gene-tissue pairs, direct causal variants, and infinitesimal effect

We have defined a function, `R2_partition`, to calculate the Pratt indices for gene-tissue pairs, direct causal variants, and infinitesimal effects. For TGFM, the calculation is:

```
TGFMPratt=R2_partition(y=bY$Zscore.y,LD=R0,
                   eta.theta=c(R0%*%fittgfm$bXest%*%fittgfm$theta),
                   eta.gamma=c(R0%*%fittgfm$gamma))
TGFMPratt
```

```
##         r1       r2        r3
## 1 0.834437 0.211783 0.6226539
```

Here, $r1$ is the total Pratt index, $r2$ is the Pratt index of gene-tissue pairs $r3$ is the Pratt index of direct causal variants.

```
TGVIPratt=R2_partition(y=bY$Zscore.y,LD=R0,
                   eta.theta=c(R0%*%bXest%*%fittgvis$theta),
                   eta.gamma=c(R0%*%fittgvis$gamma),
                   eta.upsilon=c(R0%*%fittgvis$upsilon))
TGVIPratt
```

```
##          r1        r2       r3        r4
## 1 0.9466211 0.1674825 0.529892 0.2492466
```

Here, $r4$ is the Pratt index of infinitesimal effects.