1 Guide to software requirements

To use this code, download the code files and ensure that you have the dependencies explained below. To run the code files, you have to type:

```
python3 run_MVMR.py "/home/user/file.csv" "/home/user/ld.csv"
```

Here, python3 should refer to at least Python 3.5 and depends on your specific installation of Python.

Before running the code files, check whether you have the following requirements and install them if necessary:

```
Python 3.5 or later
```

You can check this by typing 'python' (or a more specific command as explained above) in the command line. For further support, in particular how to install python please visit `https://www.python.org/`.

This Python version has the packages numpy (version 1.11.0 or later), scipy, pandas, sys and rpy2 (version 2.9.4 or later).

You can check this by starting this python version (check it especially carefully if you have multiple Python versions on your system) and typing

```
import numpy
import pandas
import rpy2
import sys
```

You would also need statistical programming language R 3.2.0 or later. You can check this by typing 'R' in the command line. To install R, please visit `https://www.r-project.org/`.

If you need to use GWAS summary data for your exposure data (gene expression data) or if you need to get the LD-matix of SNPs in your data, you can use the package TwoSampleMR (`https://mrcieu.github.io/TwoSampleMR/`). It uses the IEU GWAS database to obtain data automatically, and you can install and call them in R using the following commands

```
install.packages("devtools")
remotes::install_github("MRCIEU/TwoSampleMR")
remotes::install_github("MRCIEU/MRInstruments")
```

and

```
library(devtools)
library(TwoSampleMR)
library(MRInstruments)
```

2 Data and Preparation

We have used this method to estimate the causal effect of genes which are shared on a locus on outcome Coronary Artery Disease using summary statistics from genome wide association studies (GWAS). We used two different studies for GWAS summary data, firstly, ebi-a-GCST003116 with trait as coronary artery disease, from the year 2015 and secondly, `finn-b-I9_CHD` with trait as Major coronary heart disease event, from the year 2021.

For the summary data on eQTL analysis, we have used STARNET for association analysis from instruments (SNPs) to exposures (genes). But since this data is not publicly available, exposure data fom Gtex can be download from `https://gtexportal.org/home/datasets`. Here in the Single-Tissue cis-QTL data, you can download the full summary statistics of the cis-eQTLs mapped in European-American subjects. You can check the alignment of the effect allele from `https://www.gtexportal.org/home/faq#interpretEffectSize`.

To extract outcome data for the study of interest, we used the TwoSampleMR Package (package for performing Mendelian randomization using GWAS summary data, `https://mrcieu.github.io/TwoSampleMR/`).

Note that if you are using GTEx for exposure datasets, you have to make sure that this minimum information is provided for the extraction of the GWAS summary data from the MRBase package:

```
SNP - rs ID.
beta - The effect size (binary traits log(OR)).
se - The standard error of the effect size.
effect_allele - allele of SNP which has the effect marked in beta.
```

When you download data from Gtex, you would have data in the format such as:

```
variant_id         gene_id           maf  slope slope_se  pval_beta
chr1_64764_C_T_b38 ENSG00000227232.5 0.06 0.5    0.1        1.3e-05
```

To get the GWAS summary data for this exposure data, you would firstly need to replace the variant_id's to rs ID's from the gtex annotation file. The *slope* in the file is *beta.exposure* for the MR-Base package, *slope_se* is *se.exposure*, *pval_beta* is *pval.exposure*, *gene_id* is *exposure*, the *effect_allele.exposure* is the allele C in this example. You have to remember to match the rs ID's of the build b37 as MRBase package uses this build.

The corresponding GTex file for extraction of GWAS summary data from the MRBase package, should look like

```
SNP         exposure           maf   beta.exposure se.exposure
rs769952832 ENSG00000227232.5  0.06    0.5            0.1
```

```
pval.exposure     effect_allele.exposure   other_allele.exposure
 1.3e-05                          C                           T
```

To save the effort of going from GTEx exposure data to data usable in the MRBase package, we have GTEx exposure data aligned with corresponding rs ID's and structured in the MRBase format, available at `https://drive.google.com/drive/folders/14u2dN8k3OwnZZkSkAQFNOndboTFJFH-J?usp=share_link`.

Now if you have the exposure data in the correct format, you can get the GWAS summary data as follows:

The *available_outcomes* function returns a table of all the available studies in the database. Each study has a unique ID.

```
library(TwoSampleMR)
available_outcomes <- available_outcomes()
View(available_outcomes)
```

After this you can extract the outcome data for any study using this code that specifies these steps in R:

```
id_outcome <- ebi-a-GCST005195
outcome_dat <- extract_outcome_data(exposure_data$SNP, id_outcome)
write.csv(cad_out_dat, "Outcome.csv" ,row.names = FALSE)
write.csv(exposure_data, "Exposure.csv" ,row.names = FALSE)
```

Now that you have the outcome and exposure data, you can choose a chromosome and genetic variants within 1Mb distance of each other to run the causal analysis. Please note that this outcome data is not harmonized with the effect allele of the exposure data.

We will come to data harmonization shortly but before make sure that for the causal analysis,you either give data in the format mentioned in 3 and run the analysis as explained there or, you can use some of our functions to arrive there.

Firstly, to make sure that you have data from the same locus and of SNPs within 1 Mb of each other, you can either choose a lead SNP and run the function *Choose_SNPs.py*. This function takes as input *exposure data* which you used to extract the GWAS summary data from the MR-Base package and *GWAS summary data* as first two arguments and *chromosome* and *position* as the next two arguments. Once you run this, you will get SNPs on the chromosome *(integer given as argument for chromosome number)* 1Mb around the position of the lead SNP you gave as argument for position, saved in exposure and outcome data *.csv* files. These files will be saved in the same directory as the original files with the suffix of the chromosome and position appended to them.

As an example, if you run

```
python3 Choose_SNPs.py "/home/user/Exposure.csv"
"/home/user/Outcome.csv" 3 137997742
```

Here the lead SNP has position 137997742 on Chromosome 3 and you wish to have SNPs around this lead SNP within 1Mb of distance. You will then get the same *Exposure_3_137997742.csv* and *Outcome_3_137997742.csv* files with SNPs which are significant ($p-value \leq e-08$) and 1 Mb around 137997742 on chromosome3.

If there is no specific locus where you wish to perform the analysis but rather the entire exposure and outcome data, we have a code *Seperate_chr.py* which can output the exposure and outcome data segregated per chromosome and position. You can give as input the *exposure data file* which you used to extract the GWAS summary data from the MR-Base package and *GWAS summary data file* itself as the first two arguments. To this you will give a third argument which is an integer value (usually 2) and this function will take the exposure and output files, segregate both files per chromosome and additionally per position.

For the following code

```
python3 Seperate_chr.py "/home/user/Exposure.csv"
"/home/user/Outcome.csv" 2
```

You will have multiple files saved in your initial directory, for each chromosome and for different positions of the varuants on the chromosome. As an example, with argument 2, you will have all SNPs on a particular chromosome on positions 13____, in one file and on the same chromosome, position 91____ in a different file (into batches of SNPs sharing only the first two digits in their base pair position). The input and outcome would both have *comma* as a separator for the *.csv* files. Using this function you can approximately segregate data and then manually check for exceptions. To use this function, make sure your output and exposure data are in the format you need for the MRBase package.

Lastly to harmonize the exposure and outocme data files as well as having them in format ready for causal analysis. you can give the outputs of either of the last two functions and use the file *Data_preparation.py* to get the datasets in the format needed for running the causal analysis. The input would be exposure and outcome files you get after you run *Seperate_chr.py*, with *comma* as a separator and the output would be a file with the format required for the causal analysis saved in the same input directory with suffix *_prepared.csv*.

3 Scripts for causal analysis

There are two major files for running the causal analysis. Firstly, run_MVMR.py, is the file for getting the causal genes for cases the user can supply the LD-matrix. This file can be downloaded and run on the terminal as

```
python3 run_MVMR.py "/home/user/file.csv" "/home/user/ld.csv"
```

After python3, the first argument should be the *file.csv* file containing the SNPs to exposure effects and SNPs to outcome effects. The first column is always the SNP ID's (the ID is irrelevant if you are providing the LD-matrix but should still be filled with default values), the last column is always the SNPs to outcome effect. Every other column in between is treated as an exposure variable. The separator to be used is comma. As an example:

```
SNPs,gene1,gene2,outcome
rs11191416,0.5,0.37,0.079
rs7098825,0.34,0.0,0.078
rs17115100,0.4,0.54,0.05
```

The second argument is ld.csv file for the LD matrix. Please make sure the ordering of the SNPs is same as in the exposure *file.csv* file. This has the format:

```
1.0,0.9,0.8
0.9,1.0,0.7
0.8,0.7,1.0
```

The delimiter is comma. The results are saved as .csv file in the same directory as the one given for the exposure *file.csv* file. The code automatically prunes for SNPs in perfect-LD and keeps only the first occurring SNPs. If you would like to keep the most significant SNPs amongst perfect LD SNPs then please order the SNPs in decreasing order of significance in both .csv files.

Secondly, MVMR_withoutLD.py is the file which has the same purpose as run_MVMR.py except here you have to run the file on the terminal as

```
python3 MVMR_withoutLD.py "/home/user/file.csv"
```

i.e. without the file with the LD-matrix. The LD-matrix can be generated using the TwoSampleMR function ld_matrix. Please make sure all SNPs belong to the LD panel. To get the LD-matrix, run the following commands in R

```
snps <- list('rs7776079','rs36049381','rs9349379')
matrix <- ld_matrix(snps, with_alleles = FALSE, pop = "EUR")
```

In *snps*, add the SNPs you wish to get the LD-matrix for.

The following code will save the LD-matrix in the format required for our analysis

```
write.table(matrix,file= "ld.csv",sep=" ",quote=F,col.names=F,row.names=F)
```

3.1 Error messages

You can get the following errors while using the code:

Error Message : You require at least as many instruments as exposures to run this analysis. In this case you cannot run the analysis. The code prunes for SNPs in perfect LD so it may happen that you have more SNPs in the dataset you provided but they end up getting removed in the pruning and you get this error.

R[write to console]: The following variants are not present in the LD reference panel rs28789513. In this case you should remove the mentioned SNPs (rs28789513) from the dataset. You can only get this error if you do not provide an LD-matrix and it is generated using the R-package TwoSampleMR.

4 Comparison to other methods

You can compare your results to other methods in the MVMR community like TWMR `https://www.nature.com/articles/s41467-019-10936-0`, MVMR `https://onlinelibrary.wiley.com/doi/10.1002/gepi.21758`, and functions in the MR-Base package `https://mrcieu.github.io/TwoSampleMR/`.

For our simulations, we used TWMR as from their github page (code found under `https://github.com/eleporcu/TWMR`), MRBase with the exposure and outcome data and analysing after harmonizing the datasets as metioned in `https://mrcieu.github.io/TwoSampleMR/articles/perform_mr.html#multivariable-mr`. As for MVMR, we supplied data as mentioned in their vignette and then ran the analysis `https://cran.r-project.org/web/packages/MendelianRandomization/vignettes/Vignette_MR.pdf`.