SKiM simulation

Zijian Ni

## Simulation settings

### Marginal settings

* is total number of PMC papers.
* is disease. is the number of occurrence of in all papers. In the following examples we let .
* are symptoms. are drugs. ’s and ’s are their number of occurrence and are stored in separate lists.
* We filter out ’s and ’s with zero occuurence. There are 6 ’s and 65 ’s with zero occurrence. After filtering, there are 9266 ’s and 9600 ’s left.

### Some Notations

* Between A and B’s: =P(B occurs | A occurs), =P(B occurs | A not occurs)
* Between B’s and C’s: =P(C occurs | B occurs), =P(C occurs | B not occurs)
* Odds ratio:

### Define **associations** (pre-assign true significancy)

* is associated with randomly selected 20 of ’s. A is not associated with the rest ’s.
* Each is associated with randomly selected 10 of ’s. i.e. Among all edges between ’s and ’s, there are significant edges.
* Associated: (for simplicity, consider one-sided case)
* Not associated:
* is associated with if and are associated with at least one common . i.e. we have at most (due to overlapping) ’s associated with A among all 9665 ’s.

### Model assumptions for simulating contingency table

For simplicity, take as example. All the rest and follow the same model with different parameters.

Recall that =P(B occurs | A occurs), =P(B occurs | A not occurs). We simulate the number of co-occurrence of and , , from . We simulate the number of papers where occurs but does not occur, , from .

Note that does not necessarily equal to , but will be pretty close because the values of and are calculated based on (details in below).

### Parameter values setting

* Whether associated or not, is estimated using the marginal probability. i.e. Between and ’s: =P( occurs)=. Between and : =P( occurs)=.
* For non-associated pairs: , .
* For associated pairs: . is calculated using and .
* For associated pairs, in the simulated contingency table, a pseudo-count of min(5, sum of first row, sum of first column) is added to the two slots of co-occurrence and co-unoccurrence and subtracted from the rest two slots . The reason is: for tables with zero co-occurrence, FET tends to overestimate p-value (close to 1) and is underpowered to distinguish them even if they have large OR. For example, if and , there is a strong association with OR=~10. However, there can easily be tables with zero co-occurrence when is not large enough. In our case, . Expected co-occurrence is .
* A pseudo-count could hugely increase . For example, if , then will be 1. This is OK since we only apply to associated pairs. We are just making them more associated so that FET could test them out.

### Our goals

* Simulate contingency tables with known association and perform FET to get p-values.
* Evaluate the power and FDR under p-value cutoff 1e-5 for
  1. all associations
  2. all association for top significant ’s with highest prediction score
  3. all associations as defined previously.

## Implementation in R

### Load data and marginal settings

set.seed(2020) # for reproducibility  
setwd("~/Google Drive/Hallu/codes/ckgroup/SKIM/")   
source("simulation\_functions.R")  
  
N <- 29613663 # total number of papers in database  
n\_A <- 6507 # number of papers containing A  
meta\_B <- readxl::read\_xlsx("SKiM\_Files/To\_Zijian/Phenotypes\_and\_symptoms\_count.xlsx",   
 sheet = 1)  
meta\_C <- readxl::read\_xlsx("SKiM\_Files/To\_Zijian/Drugs\_count.xlsx",sheet = 1)  
n\_B <- meta\_B$Phenotype\_and\_symptom\_count # number of papers containing each B  
n\_C <- meta\_C$Drug\_count # number of papers containing each C  
  
# Filter out zero occurrence  
meta\_B <- meta\_B[n\_B>0,]  
n\_B <- n\_B[n\_B>0]  
meta\_C <- meta\_C[n\_C>0,]  
n\_C <- n\_C[n\_C>0]  
  
p\_B <- n\_B/N  
p\_C <- n\_C/N

Check the metadata, counts and probability of ’s:

head(meta\_B)

## # A tibble: 6 x 3  
## Phenotype\_and\_symptom Phenotype\_and\_symptom\_~ db\_article\_count  
## <chr> <dbl> <dbl>  
## 1 C0239337\_T190:abnormal\_limbs 1441 29613663  
## 2 M00004289\_T033:abnormality\_of\_the\_po~ 100 29613663  
## 3 C0596875\_T047:lysinemia 2 29613663  
## 4 C1843112\_T033:broad\_nail 2 29613663  
## 5 D00000686\_T777:normal\_cry 3 29613663  
## 6 C1859698\_T033:large\_joint\_contractur~ 39 29613663

head(n\_B)

## [1] 1441 100 2 2 3 39

head(p\_B)

## [1] 4.865997e-05 3.376820e-06 6.753639e-08 6.753639e-08 1.013046e-07  
## [6] 1.316960e-06

### Define **associations**

#######################################  
# Set significant B's  
n\_signif\_B <- 20 # number of significant B's  
  
# indices of significant B's to A  
which\_signif\_BtoA <- sample.int(nrow(meta\_B), n\_signif\_B)   
  
  
#######################################  
# Set significant C's for each B  
n\_signif\_C <- 10 # number of significant C's for each B  
# indices of significant C's to each B, as columns  
which\_signif\_CtoB <- replicate(nrow(meta\_B),sample.int(nrow(meta\_C), n\_signif\_C))  
  
  
#######################################  
# Set significant C's to A  
which\_signif\_CtoA <- unique(as.vector(which\_signif\_CtoB[,which\_signif\_BtoA]))

Overview of assigned associated terms:

str(which\_signif\_BtoA)

## int [1:20] 7767 8920 4417 8465 170 7878 945 4992 2602 3062 ...

str(which\_signif\_CtoB)

## int [1:10, 1:9266] 1524 7468 8827 8182 4690 3142 5163 2509 1404 4952 ...

str(which\_signif\_CtoA)

## int [1:198] 8280 8779 8358 1978 946 4271 7793 2368 4291 9159 ...

### Parameter setting for ’s

#######################################  
# Set p\_1 and p\_2 for A and each B  
p\_1\_BtoA <- p\_2\_BtoA <- p\_B  
OR\_BtoA <- 2+rexp(n\_signif\_B)  
p\_1\_BtoA[which\_signif\_BtoA] <- get\_p1(p\_2\_BtoA[which\_signif\_BtoA],OR\_BtoA)

Check and for associated ’s:

p\_1\_BtoA[which\_signif\_BtoA]

## [1] 3.349202e-06 2.540042e-07 1.466906e-06 3.438339e-06 8.673815e-04  
## [6] 4.466043e-06 1.366215e-04 1.635897e-06 3.494111e-05 1.037920e-05  
## [11] 6.905228e-04 1.797883e-06 1.238637e-05 1.169223e-03 9.842889e-06  
## [16] 6.070601e-05 1.631564e-05 1.527495e-05 3.804508e-06 2.169919e-05

p\_2\_BtoA[which\_signif\_BtoA]

## [1] 9.455095e-07 1.013046e-07 6.753639e-07 1.215655e-06 3.692215e-04  
## [6] 1.587105e-06 2.232078e-05 4.389866e-07 1.266307e-05 3.376820e-06  
## [11] 2.922975e-04 6.078275e-07 6.179580e-06 3.921501e-04 1.553337e-06  
## [16] 2.775746e-05 7.395235e-06 7.125090e-06 1.891019e-06 6.280885e-06

### Simulate contingency table and perform FET for all ’s

#######################################  
# Simulate contingency tables, calculate sort ratios and perform FET between A and each B  
pval\_BtoA <- sort\_ratio\_BtoA <- numeric(nrow(meta\_B))  
  
for(B\_idx in seq\_along(pval\_BtoA)){  
 temp\_table <- simulate\_table(N, n\_A, N-n\_A, p\_1\_BtoA[B\_idx], p\_2\_BtoA[B\_idx])  
 # Add pseudo-count for true associations  
 if(B\_idx%in%which\_signif\_BtoA){  
 pseudo\_mat <- min(5,sum(temp\_table[,1]),sum(temp\_table[1,]))\*  
 matrix(c(1,-1,-1,1),2,2)  
 temp\_table <- temp\_table+pseudo\_mat  
 }  
 sort\_ratio\_BtoA[B\_idx] <- temp\_table[1,1]/sum(temp\_table[,1])  
 pval\_BtoA[B\_idx] <- fisher.test(temp\_table)$p.value  
}  
  
# Calculate prediction score  
score\_BtoA <- -log10(pval\_BtoA)+log10(sort\_ratio\_BtoA)

Example table with association:

simulate\_table(N, n\_A, N-n\_A,   
 p\_1\_BtoA[which\_signif\_BtoA[1]],   
 p\_2\_BtoA[which\_signif\_BtoA[1]])+  
 5\*matrix(c(1,-1,-1,1),2,2)

## have\_y no\_y  
## have\_x 5 6502  
## no\_x 19 29607137

Example table without association:

simulate\_table(N, n\_A, N-n\_A, p\_1\_BtoA[1], p\_2\_BtoA[1])

## have\_y no\_y  
## have\_x 0 6507  
## no\_x 1397 29605759

Check p-value of associated ones and randomly selected 10:

pval\_BtoA[which\_signif\_BtoA]

## [1] 1.024798e-13 1.060390e-11 2.707238e-14 5.005108e-14 4.485487e-05  
## [6] 8.688404e-13 1.144461e-08 2.229382e-15 2.013313e-08 2.914793e-11  
## [11] 8.105453e-04 3.157713e-15 4.478054e-10 3.290097e-04 2.320763e-12  
## [16] 1.435043e-06 1.197714e-11 2.017229e-09 5.514549e-13 9.174042e-10

sample(pval\_BtoA,10)

## [1] 0.1109063 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## [8] 1.0000000 1.0000000 0.7860476

Number of p-values less or equal to 1e-5:

sum(pval\_BtoA<=1e-5)

## [1] 17

### Select top ’s with highest prediction score among significant ’s

Since we have less than significant ’s, we just use these significant ones instead of 50.

######################################  
# Keep significant Bs with p-value <=1e-5, then  
# choose top 50 B's with largest prediction score  
# or largest p-values  
  
  
#cand\_B <- which(rank(pval\_BtoA)<=50)  
cand\_B <- which(rank(-score\_BtoA)<=50)  
  
# Since there are only 20 pval left, just use these 20 instead of 50.  
cand\_B <- cand\_B[pval\_BtoA[cand\_B]<=1e-5]

Indices of candidate ’s to keep:

cand\_B

## [1] 78 724 945 2602 3062 3662 4417 4483 4684 4992 5104 6888 7687 7767 7878  
## [16] 8465 8920

For stress test, we can also use top ’s with smallest p-values regardless of their significancy.

cand\_B\_2 <- which(rank(pval\_BtoA)<=50)

### Simulate contingency table and perform FET for top ’s and all ’s

For significant ’s:

res\_CtoB <- test\_CtoB(cand\_B,p\_C,which\_signif\_CtoB, verbose=F)

A matrix of p-values:

str(res\_CtoB$PVAL\_CtoB)

## num [1:9600, 1:17] 1 1 1 1 1 1 1 1 1 1 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : NULL  
## ..$ : chr [1:17] "78" "724" "945" "2602" ...

Number of significant ’s for :

sum(res\_CtoB$PVAL\_CtoB[,1]<=1e-5)

## [1] 9

For top 50 ’s:

## num [1:9600, 1:50] 1 1 1 1 1 1 1 1 1 1 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : NULL  
## ..$ : chr [1:50] "78" "170" "303" "724" ...

### Evaluate power and FDR by comparing predicted association (p-value<=1e-5) with true association

#### Evaluate for predicted significancy between all ’s

get\_power\_FDR(pval\_BtoA<=1e-5, seq\_len(nrow(meta\_B))%in%which\_signif\_BtoA)

## $power  
## [1] 0.85  
##   
## $FDR  
## [1] 0

#### Evaluate for predicted significancy between top ’s and all ’s

For significant ’s:

# True significancy  
true\_signif\_CtoB <- apply(which\_signif\_CtoB[,cand\_B],  
 2, function(x) seq\_len(nrow(meta\_C))%in%x)  
  
get\_power\_FDR(as.vector(res\_CtoB$PVAL\_CtoB<=1e-5), as.vector(true\_signif\_CtoB))

## $power  
## [1] 0.9823529  
##   
## $FDR  
## [1] 0

For top 50 ’s:

# True significancy  
true\_signif\_CtoB\_2 <- apply(which\_signif\_CtoB[,cand\_B\_2],  
 2, function(x) seq\_len(nrow(meta\_C))%in%x)  
  
get\_power\_FDR(as.vector(res2\_CtoB$PVAL\_CtoB<=1e-5), as.vector(true\_signif\_CtoB\_2))

## $power  
## [1] 0.93  
##   
## $FDR  
## [1] 0.002145923

#### Evaluate for predicted significancy between and ’s

For significant ’s:

SKiM\_signif\_CtoA <- apply(res\_CtoB$PVAL\_CtoB,1,function(x) any(x<=1e-5))  
  
get\_power\_FDR(SKiM\_signif\_CtoA, seq\_len(nrow(meta\_C))%in%which\_signif\_CtoA)

## $power  
## [1] 0.8333333  
##   
## $FDR  
## [1] 0

For top 50 ’s (same result since only significant ’s will be used to link between and ’s):

SKiM\_signif\_CtoA\_2 <- apply(res\_CtoB$PVAL\_CtoB,1,function(x) any(x<=1e-5))  
  
get\_power\_FDR(SKiM\_signif\_CtoA\_2, seq\_len(nrow(meta\_C))%in%which\_signif\_CtoA)

## $power  
## [1] 0.8333333  
##   
## $FDR  
## [1] 0

### Why FDR is zero

* The reason of zero FDR is under such biased contigency tables, the p-value distribution of FET under null hypothesis (no association) is discrete and highly skewed towards 1. It’s hard to observe small p-values for un-associated pairs. See discussions in [this paper](https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/fdr20for20contingency20tables.pdf). When the p-value cutoff is conservative enough (1e-5 in SKiM), we are not making any false positives.

## More evaulations with random replications

We now test on a wide range of common and rare diseases. Candidate values of are 500, 5,000, 50,000, 100,000, 200,000. For each , we repeat 10 times of the simulation and report the average power and FDR. Codes for this part are stored in a separate file. We just show the final results here:

sim\_out <- read.csv("sim\_out\_10012020.csv")  
sim\_out

## X n\_A power\_AtoB power\_BtoC power\_AtoC FDR\_AtoB FDR\_BtoC FDR\_AtoC  
## 1 1 n\_A=500 0.96 0.9395182 0.9026694 0 0.002295158 0.002323967  
## 2 2 n\_A=5000 0.90 0.9562515 0.8619289 0 0.001618715 0.001651652  
## 3 3 n\_A=50000 0.83 0.9504965 0.7887817 0 0.000000000 0.000000000  
## 4 4 n\_A=1e+05 0.76 0.9403226 0.7157341 0 0.002638067 0.002646379  
## 5 5 n\_A=2e+05 0.77 0.9272452 0.7169642 0 0.003433889 0.003457114

## Session Information

sessionInfo()

## R version 4.0.2 (2020-06-22)  
## Platform: x86\_64-w64-mingw32/x64 (64-bit)  
## Running under: Windows 10 x64 (build 18363)  
##   
## Matrix products: default  
##   
## locale:  
## [1] LC\_COLLATE=Chinese (Simplified)\_China.936   
## [2] LC\_CTYPE=Chinese (Simplified)\_China.936   
## [3] LC\_MONETARY=Chinese (Simplified)\_China.936  
## [4] LC\_NUMERIC=C   
## [5] LC\_TIME=Chinese (Simplified)\_China.936   
##   
## attached base packages:  
## [1] stats graphics grDevices utils datasets methods base   
##   
## other attached packages:  
## [1] qvalue\_2.21.0  
##   
## loaded via a namespace (and not attached):  
## [1] Rcpp\_1.0.5 pillar\_1.4.6 compiler\_4.0.2 cellranger\_1.1.0  
## [5] plyr\_1.8.6 tools\_4.0.2 digest\_0.6.25 evaluate\_0.14   
## [9] lifecycle\_0.2.0 tibble\_3.0.3 gtable\_0.3.0 pkgconfig\_2.0.3   
## [13] rlang\_0.4.7 cli\_2.0.2 yaml\_2.2.1 xfun\_0.18   
## [17] stringr\_1.4.0 dplyr\_1.0.2 knitr\_1.30 generics\_0.0.2   
## [21] vctrs\_0.3.4 grid\_4.0.2 tidyselect\_1.1.0 glue\_1.4.2   
## [25] R6\_2.4.1 fansi\_0.4.1 readxl\_1.3.1 rmarkdown\_2.4   
## [29] ggplot2\_3.3.2 purrr\_0.3.4 reshape2\_1.4.4 magrittr\_1.5   
## [33] scales\_1.1.1 ellipsis\_0.3.1 htmltools\_0.5.0 splines\_4.0.2   
## [37] assertthat\_0.2.1 colorspace\_1.4-1 utf8\_1.1.4 stringi\_1.5.3   
## [41] munsell\_0.5.0 crayon\_1.3.4