Meta analyses of omics data

Workshop on Omics Integration

Ashfaq Ali • 20-Apr-2021

OUTPUT

0.1 Installation and setup

Before we begin reading this tutorial and running the code, let us set up our R environment for this session. We will be using “renv” package in R to install all the necessary packages needed for the sessions.

> *Note: Due to the use of the development versions of R packages that are not available through conda repositories we are not able to generate a conda environment for this session and an alternative solution needs to be used.*

R

```
setwd("~/Documents/GitHub/workshop_omics_integration/session_meta/")
getwd()
```

```
## [1] "/Users/cob-aaf/Documents/GitHub/workshop_omics_integration/sessio
n_meta"
```

R

```
## make sure that the renv.lock file for the session exists and do follow
        ing.
install.packages("renv")

renv::restore(lockfile = "./Meta_tutorial/tuturial_meta.lock", prompt = T
        RUE)
```

> If XML lib fails in ubuntu and you get the error "Error installing package 'XML':" go to terminal and install the libraries. "sudo apt-get update" followed by "sudo apt-get install libxml2-dev" Now, try the renv::init() again. > If you are trying to install within you conda environmen, you may need to install some dependencies > `install.packages(c("curl","remotes"))` and `remotes::install_github("joshuaulrich/TTR")` > In bash of your active conda environment run `conda install -c conda-forge r-gert` There maybe issues with installtion even after these depending on you architecture, in that case you can locate the results for each of the sections in "results.tar.gz" file which you can expand by `tar -xvzf results.tar.gz` and explore the results.

1 Introduction

Meta analyses are commonly used in clinical studies to assess the effect of a treatment or a genetic locus on a phenotype. With the advent of GWAS (Genome wide association studies), various studies report summary statistics on associated effects of genetic loci on phenotype. Meta analyses can be useful in determining whether the effect size is consistent across the body of data

The goal of a synthesis is to understand the results of any study in the context of all the other studies. First, we need to know whether or not the effect size is consistent across the body of data. If it is consistent, then we want to estimate the effect size as accurately as possible and to report that it is robust across the kinds of studies included in the synthesis. On the other hand, if it varies substantially from study to study, we want to quantify the extent of the variance and consider the implications. Meta-analysis is able to address these issues whereas the narrative review is not. We start with an example to show how meta-analysis and narrative review would approach the same question, and then use this example to highlight the key differences between the two.

2 Meta analyses Methods

1. **p-value**

a. **Fisher** (https://link.springer.com/chapter/10.1007%2F978-1-4612-4380-9_6): Sum of minus log-transformed *P*-values where larger Fisher score reflects stronger aggregated differential expression evidence.
b. **Stouffer** (https://cutt.ly/rc53t31): Sum of inverse normal transformed *P*-values where larger Stouffer score to reflect stronger aggregated statistical evidence.
c. **adaptively weighted Fisher(AW)** (https://doi.org/10.1214/10-AOAS393)**, original publication (https://ui.adsabs.harvard.edu/abs/2011arXiv1108.3180L/abstract)** : assigns different weights to each individual study and it searches through all possible weights to find the best adaptive weight with the smallest derived *p*-value. One significant advantage of this method is its ability to indicate which studies contribute to the evidence aggregation and elucidates heterogeneity in the meta-analysis.
d. **minimum p-value (minP)** (https://psycnet.apa.org/record/1951-06623-001)**:** The minP method takes the minimum *p*-value among the *K* studies as the test statistic
e. **maximum p-value (maxP)** (https://psycnet.apa.org/record/1951-06623-001)**:** The maxP method takes maximum *p*-value as the test statistic
f. **rth ordered p-value (rOP)** (https://pubmed.ncbi.nlm.nih.gov/25383132/)**:** The rOP method takes the *r*-th order statistic among sorted *p*-values of *K*combined studies

> Note: The assumption of uniformly distributed *P*-values under the null hypothesis or can be done non-parametrically by permutation-based analysis

2. **Effect Size based**

a. **fixed effects model (FEM)** (https://pubmed.ncbi.nlm.nih.gov/12855442/):FEM combines the effect size across *K* studies by assuming a simple linear model with an underlying true effect size plus a random error in each study
b. **random effects model (REM)** (https://pubmed.ncbi.nlm.nih.gov/12855442/): REM extends FEM by allowing random effects for the inter-study heterogeneity in the model.

3. **Rank based**

a. **rank product (rankProd)** (https://pubmed.ncbi.nlm.nih.gov/16982708/)RankProd and RankSum are based on the common biological belief that if a gene is repeatedly at the top of the lists ordered by up- or down-regulation fold change in replicate experiments, the gene is more likely a DE gene.
b. **naive sum of ranks and naive product of ranks** (https://pubmed.ncbi.nlm.nih.gov/16982708/)**:** These two methods apply a naïve product or sum of the DE evidence ranks across studies.

2.1 Statistical considerations

In addition to statistical methods mentioned above, a number of factors need to be considered for the choice of method. Different test statistics maybe used depending on the type of outcome variable (e.g. t-statistic or moderated t-statistic for binary outcome, F-statistic for multi-class outcome, regression or correlation coefficient for continuous outcome and log-rank statistic for survival outcome).

Here we will be using MetaDE (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3463115/) package that has implemented above mentioned methods.

A summary of the methods and their implementations for respective outcome variables are listed in the table below as presented in the original publication.

Outcome Variable		binary	multi-class	continuous	suvival
Test statistics		paired t-statistics unpaired t-statistics moderate t-statistics	F-statistics	Pearson correlation Spearman correlation	log-rank statistics
Combine p-values	Fisher (_OC)	✔	✔	✔	✔
	Stouffer (_OC)	✔	✔	✔	✔
	AW (_OC)	✔	✔	✔	✔
	minP (_OC)	✔	✔	✔	✔
	maxP (_OC)	✔	✔	✔	✔
	roP (_OC)	✔	✔	✔	✔
	SR	✔	✔	✔	✔
	PR	✔	✔	✔	✔
	minMCC		✔		
Combine effect sizes	FEM	✔	×	×	×
	REM	✔	×	×	×
combine ranks	randProd	✔	×	×	×

✔: the method can be applied on the corresponding type of outcome.

×: The method cannot be applied on the corresponding type of outcome.

_OC: The corresponding one-sided correction method can be implemented in MetaDE.

For a detailed review of the applied methods for meta analyses, related benchmarks and

2.2 Required packages

R

```
library(magrittr)
library(plyr)
library(preproc)
library(MetaQC)
library(MetaDE)
library(MetaPath)
library(MetaDCN)
```

3 Summary of the session

I this session we will be working with the “prostate8.rda” data set that is located in the data directory of the “session_meta” directory of the github repository for the course.

Some of the R packages packages we are using here are not available through conda repositories and therefore need to be install from the github repositories for the respective packages

The session comprises of following main steps

1. Explore the data and the packages
2. Filter the data based on QC
3. Perform differential expression analyses and meta analyses
4. Perfrom pathway analyses for meta analyses results
5. Perfrom pathway meta analyses
6. Network meta analyses (optional)
7. Exercise: compare different meta analyses methods including AW-Fisher, REM, FEM ## Perform QC

4 Data for the session

We will using prostate dataset

Load the protaste data set

R

```
load(file = "./data/prostate8.rda")
```

The prostate data is comprised of 8 microarray studies from different microarray platforms. Here it is provided in a list format where data matrices and corresponding labels are provided.

Let us take a quick look at the data.

R

```
names(prostate8)
```

```
## [1] "data"      "dataLabel"
```

We can have a look at the structure of each data sets and evaluate what sort of filtering steps we need to take.

R

```
library(magrittr)
lapply(prostate8$data, dim) %>% as.data.frame(row.names = c("genes", "sam
        ples"))
```

```
##         Welsh   Yu Lapointe Varambally Singh Wallace Nanni Tomlins
## genes    8798 8799    13579      19738  8799   12689 12688    9703
## samples    34  146      103         13   102      89    30      57
```

As we can see, the studies have different number of genes measured and contain different number of samples. In a meta analyses, studies can have different samples sizes but genes should match between different studies. To be able to merge data sets, all data matrices should have the same annotations

Let us look at the row names which are already set to gene names.

R

```
lapply(prostate8$data, row.names) %>% lapply(.,head, 5) %>% as.data.frame
        ()
```

```
##   Welsh    Yu Lapointe Varambally Singh Wallace  Nanni Tomlins
## 1  ACTB MAPK3    ITGB2       DDR1 MAPK3    DDR1   DDR1     ZFX
## 2 GLRA1  TIE1     HMBS       PAX8  TIE1    RFC2   RFC2    PIGS
## 3 KCNB2 CXCR5    GATA6       THRA CXCR5    PAX8  HSPA6    ZPBP
## 4 MGAT5 DUSP1    ICAM5       CCL5 DUSP1   ESRRA   PAX8  NOTCH3
## 5  BMP3 MMP10     RIN2      ESRRA MMP10    GAS6 PTPN21  IGSF11
```

It is clear that the gene names are set to gene symbols for all data sets and we can merge theses data set.

The second element of the lists corresponds to data labels where each value corresponds to disease status of the individual where the sample came from.

R

```
prostate8$dataLabel %>% lapply(. ,table) %>% as.data.frame() %>% t
```

```
##                 [,1] [,2]
## Welsh.Var1      "0"  "1"
## Welsh.Freq      " 9" "25"
## Yu.Var1         "0"  "1"
## Yu.Freq         "81" "65"
## Lapointe.Var1   "0"  "1"
## Lapointe.Freq   "41" "62"
## Varambally.Var1 "0"  "1"
## Varambally.Freq "6"  "7"
## Singh.Var1      "0"  "1"
## Singh.Freq      "50" "52"
## Wallace.Var1    "0"  "1"
## Wallace.Freq    "20" "69"
## Nanni.Var1      "0"  "1"
## Nanni.Freq      " 7" "23"
## Tomlins.Var1    "0"  "1"
## Tomlins.Freq    "27" "30"
```

Data labels indicate “0” as control and “1” as disease groups. We can set these values to character format for convenience.

If everything looks good, we are good to go to the next step.

> Take a moment to explore the data in your own ways and fimilarise yourself wit it.

---

5 Data preprocessing

5.1 Gene matching

Usually different microarray platforms use their own probe IDs or experiments from different omics platforms can have annotations for transcripts of proteins. To perform meta-analysis, one needs to match probe/transcript IDs from different platforms to the unique official gene ID, such as ENTREZ ID or gene symbol.

Options for situations for microarray data include

- take the average value of expression values across multiple probe IDs to represent the corresponded gene symbol
- select the probe ID with the largest interquartile range (IQR) of expression
- Or some version of summary at the gene level depending on the omics platform.

We do not cover the pre-processing steps for different omics technologies at the moment but the workshop participants are encouraged to apply the domain specific knowledge when setting up a meta analyses study.

R

```
MetaQC::metaOverlap(prostate8$data) %>%
                    lapply(dim) %>% as.data.frame(row.names = c("genes",
        "samples"))
```

```
##         Welsh   Yu Lapointe Varambally Singh Wallace Nanni Tomlins
## genes    4241 4241     4241        4241  4241    4241  4241    4241
## samples    34  146      103          13   102      89    30      57
```

As you can see here, only 4241 genes are present in all studies and if we are to filter out this way, we lose a lot of data.

> Biologically, it is likely that most genes are either un-expressed or un-informative. In gene expression analysis to find DE genes, these genes contribute to the false discoveries, so it is desirable to filter out these genes prior to analysis. After genes are matched across studies, the unique gene symbols are available across all studies. Two sequential steps of gene filtering can be performed. In the first step, we filter out genes with very low gene expression that are identified with small average expression values across majority of studies. And then we can remove genes that are not variable in your data sets using variance estimates as they are not useful in comparisons.

You can take a look at preproc () package for some of the functions available for filtering the data and the intuition behind the methods. Here `Annotate(), Impute(), Filter() and Merge()` maybe useful for pre-processing steps of the data analyses.

R

```
data2 <- prostate8$data
data2 <- preproc::Merge(data2)
data.type = rep("microarray", length(data2)) # a character vector for eac
        h study type

data2_filt <- preproc::Filter(data2, del.perc = c(0.1, 0.1),data.type = d
        ata.type ) # Here we specify the percentage of genes to be filter
        e, and specify the type of data for each study

summary_preproc <- rbind(as.data.frame(lapply(prostate8$data, dim))[1,],
                         as.data.frame(lapply(data2, dim))[1,],
                         as.data.frame(lapply(data2_filt, dim))
                          )

rownames(summary_preproc) <- c("orginal", "merged", "mergeFiltered", "sam
        ples")

summary_preproc
```

```
##                Welsh   Yu Lapointe Varambally Singh Wallace Nanni Tomli
ns
## orginal         8798 8799    13579      19738  8799   12689 12688    97
03
## merged          4241 4241     4241       4241  4241    4241  4241    42
41
## mergeFiltered   3434 3434     3434       3434  3434    3434  3434    34
34
## samples           34  146      103         13   102      89    30
57
```

Note that the multiple gene expression data sets may not be very well aligned by genes, and the number of genes in each study maybe different. When we combine a large number of studies, the number of common genes may be very small, so we need to allow for genes appearing in most studies and missing in few studies etc.

5.2 Load pathway database

Let us perform quality control of the data for this meta analyses. We are using the “MetaQC” (https://academic.oup.com/nar/article/40/2/e15/2408973) that identifies ways to objectively perform quality control for the microarray studies.

R

```
load(file = "./data/pathways.rda")

DList=prostate8$data
colLabel=prostate8$dataLabel
#GList=pathway[[1]]
#GList=pathwayDatabase

filterGenes=TRUE
cutRatioByMean=0.3 #
cutRatioByVar=0.3
#tic() not run
QCresult=MetaQC(DList, colLabel, GList=c(Hallmark.genesets, KEGG.geneset
        s, Immunologic.genesets),filterGenes,cutRatioByMean,cutRatioByVa
        r, ) ## This will take some time depending on the number of studi
        es and the type of data
#toc()
```

> If you are unable to run the above steps, please load the “QC.rda” from the results directory of the session.

OUTPUT

R

```
QCresult$scoreTable
```

```
##                  IQC       EQC          AQCg     AQCp          CQCg
CQCp
## Welsh      4.6297212 5.0993305 28.269084964 0.000000 1.671669e+02 1.43
4671e+02
## Yu         9.4947631 9.4876742 21.725636675 0.000000 1.594930e+02 1.38
3082e+02
## Lapointe   3.5445936 3.9678407 24.326120224 0.000000 9.159887e+01 2.04
5790e+01
## Varambally 4.2482870 3.7655707  4.272765763 0.000000 1.858428e+01 3.79
5110e+01
## Singh      0.8946880 2.0302734 14.740646685 6.945533 4.447641e+01 5.83
9619e+01
## Wallace    8.1666052 8.9729647  0.003464049 0.000000 4.453816e-05 3.18
3405e-04
## Nanni      0.8134973 0.6480432  0.000000000 0.000000 3.324084e-01 2.56
2951e-04
## Tomlins    0.9366496 0.4514895  0.984994108 0.000000 7.892800e+00 1.76
7292e+01
```

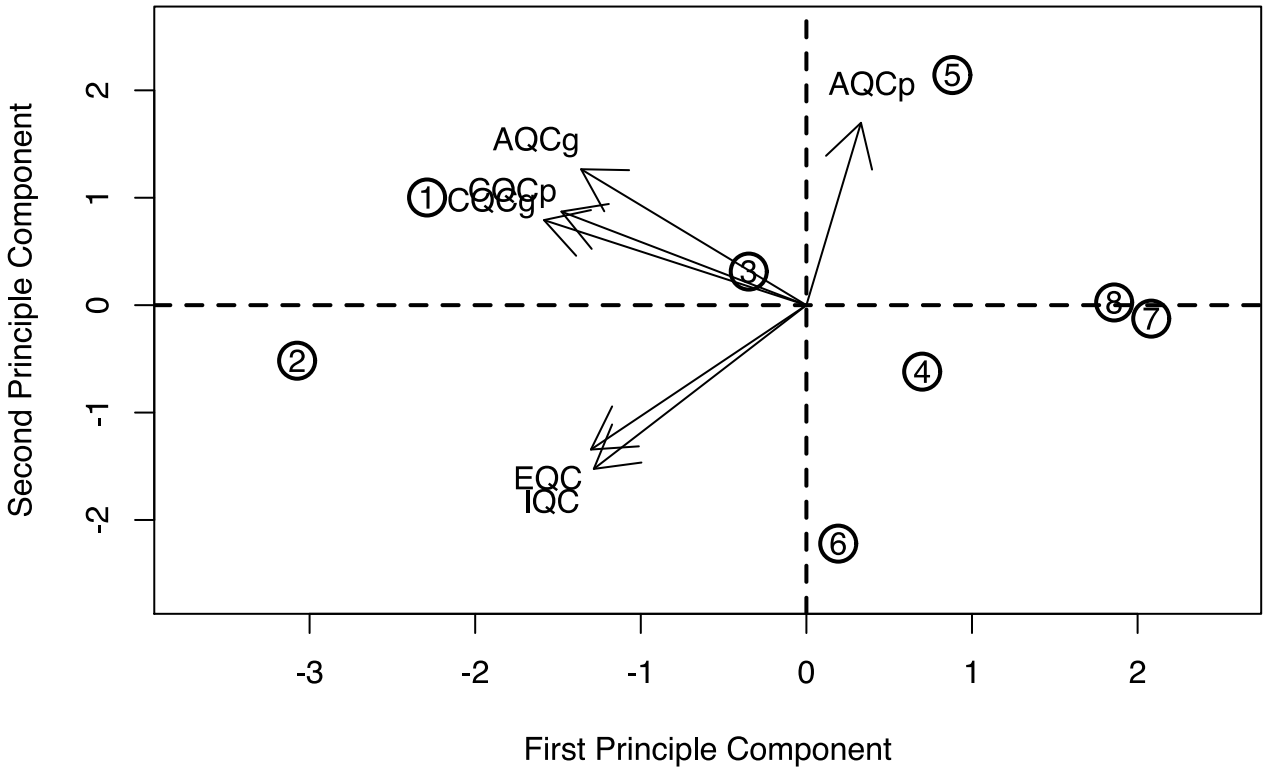
IQC, EQC, AQCg, AQCp, CQCg, CQCp

> Internal quality control index:small IQC indicated that the study had heterogeneous coexpression structure with other studies and was considered a candidate problematic study that should be excluded from meta-analysis
>
> the external quality control (EQC): small *EQC* indicated that the study had low association with pathway in terms of gene pairwise correlation structure and maybe considered a candidate problematic study.
>
> accuracy quality control (*AQC*) and a consistency quality control (*CQC*).
>
> Large *AQCg* measure for a given study indicate that DE genes produced by study were reproducible compared to DE genes detected by meta-analysis excluding study
>
> Having a large CQCg measure for a given study indicated that DE evidence produced by study was consistent with DE evidence generated by meta-analysis excluding study. Please read the original MetQC publication (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3898528/#B14) for a better understanding of the measures.

We can now look at the two dimensional biplot based on PCA analyses and get an idea of any studies that may not be of great quality.

R

```
MetaQC::plotMetaQC(QCresult$scoreTable)
```



Although the `plotMetaQC()` function is a useful tool for visualization, it does not necessarily allow automatic inclusion/exclusion criteria.

Note that our visualization and summarization tools were not meant for an automated recommendation for inclusion/exclusion decision. In the examples we explored, there were roughly three categories in the QC results: definite exclusion cases with poor quality, definite inclusion cases with good quality and borderline cases.

5.2.1 Update based on the QC

Here Nanni and Tomlins are the two studies that maybe of low quality and should perhaps be excluded from the analyses.

R

```
filterGenes=TRUE
cutRatioByMean=0.3 #
cutRatioByVar=0.3
to_remove <- c("Nanni", "Tomlins")
prostate6 <- list(data = within(prostate8$data, rm(Nanni, Tomlins)), data
        Label=within(prostate8$dataLabel, rm(Nanni, Tomlins)))
prostate_fil <- list(data = MetaQC::metaOverlap(prostate6$data), dataLabe
        l=prostate6$data)
prostate6$data <-MetaQC::metaFilterData(prostate_fil$data,cutRatioByVar =
        cutRatioByVar, cutRatioByMean = cutRatioByMean)
lapply(prostate6, names)

as.data.frame(lapply(prostate6$data, dim))
```

```
## $data
## [1] "Welsh"      "Yu"         "Lapointe"   "Varambally" "Singh"
## [6] "Wallace"
##
## $dataLabel
## [1] "Welsh"      "Yu"         "Lapointe"   "Varambally" "Singh"
## [6] "Wallace"
##
##   Welsh   Yu Lapointe Varambally Singh Wallace
## 1  3399 3399     3399       3399  3399    3399
## 2    34  146      103         13   102      89
```

R

```
DList=prostate6$data
colLabel=prostate6$dataLabel
#GList=pathway[[1]]
#GList=pathwayDatabase
filterGenes=TRUE
cutRatioByMean=0.3 #
cutRatioByVar=0.3
QCresult2=MetaQC(DList, colLabel, GList=c(Hallmark.genesets, KEGG.geneset
        s, Immunologic.genesets), filterGenes, cutRatioByMean, cutRatioBy
        Var)
```

> AGAIN, if you are unable to run the code, please load

R

```
load("results/QC_step2.rda")
```

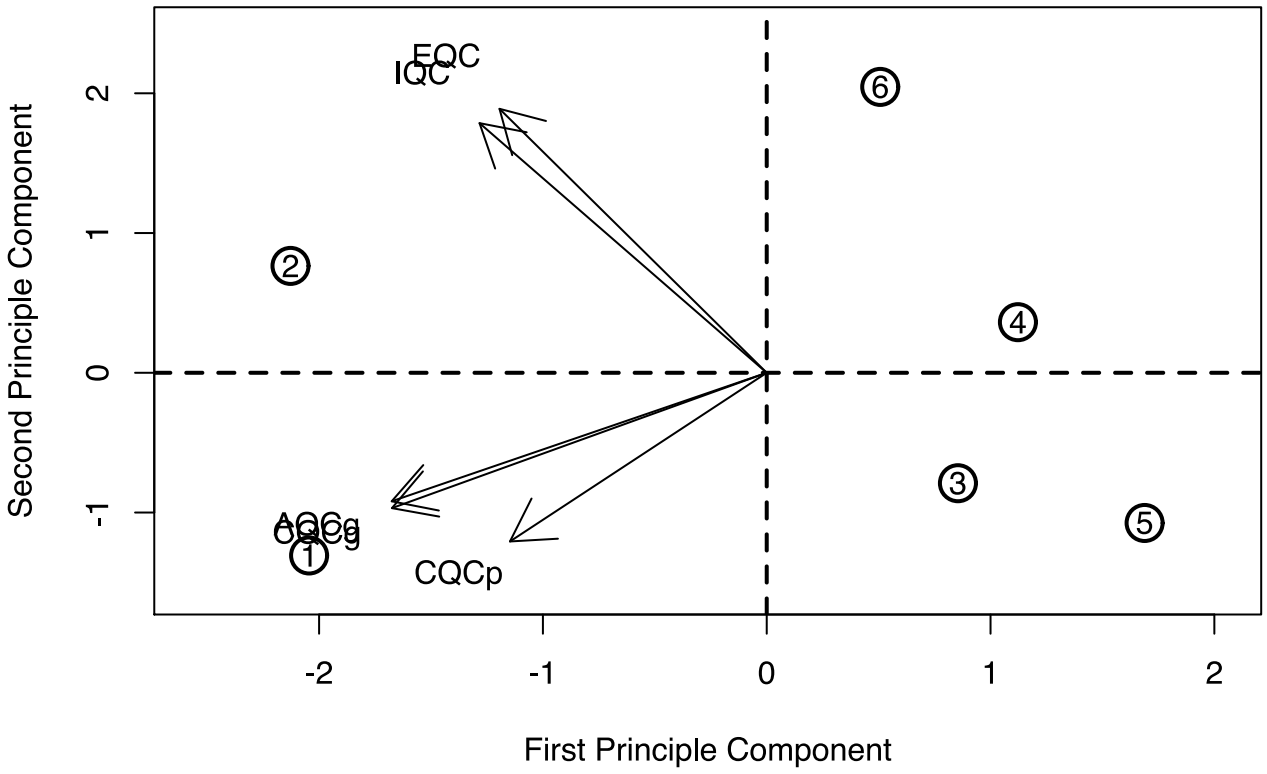
R

```
QCresult2$scoreTable
```

```
##                  IQC       EQC         AQCg AQCp          CQCg          C
QCp
## Welsh      4.3956311 3.9486143 38.42833030    0 1.913178e+02 4.028780e
-02
## Yu         7.0164906 6.8265364 42.25733812    0 1.964917e+02 1.920686e
-14
## Lapointe   1.5414898 1.7423903 16.70625993    0 9.884958e+01 0.000000e
+00
## Varambally 3.5800781 2.9649972  7.18472473    0 2.391464e+01 0.000000e
+00
## Singh      0.4912204 0.3180309 12.36863835    0 4.555982e+01 0.000000e
+00
## Wallace    6.4044351 6.7765190  0.02200338    0 2.785632e-03 0.000000e
+00
```

R

```
plotMetaQC(scoreTable = QCresult2$scoreTable)
```



6 Meta Differential Expression analyses

We have listed the methods for differential expression meta analyses in the introduction section that are implemented in the MetaDE package. Here we will try the best performing methods including AW Fischer, REM and rOP. A detailed comparison of relative performances of each of the statistical methods is descibed Lun-Ching etal (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3898528/#B14).

R

```
data <- prostate6$data # Extract the expression matrices
clin.data <- prostate6$dataLabel # extract lables for each sample

K <- length(data)

clin.data <- lapply(clin.data, function(x) {data.frame(x)} )

for (k in 1:length(clin.data)){
 colnames(clin.data[[k]]) <- "label"
 clin.data[[k]] <- (ifelse(clin.data[[k]]==0, "Control", "Cancer"))

}

#clin.data <- lapply(clin.data, function(x) {data.frame(x)} )
select.group <- c("Control", "Cancer")
ref.level <- "Control"
data.type <- "continuous"
ind.method <- rep('limma',length(data))
resp.type <- "twoclass"
paired <- rep(FALSE,length(data))


meta.method <- "Fisher"


meta.res <- MetaDE(data=data,clin.data = clin.data,
                   data.type=data.type,resp.type = resp.type,
                   response='label',
                   ind.method=ind.method, meta.method=meta.method,
                   select.group = select.group, ref.level=ref.level,
                   paired=paired,tail='abs',parametric=TRUE)
meta.res.summary <- MetaDE::summary.meta(meta.res, resp.type = resp.type,
        meta.method = meta.method)
head(meta.res.summary)
# save(meta.res, meta.res.summary, file = "./results/resMetaDE.rda")
```

```
## Please make sure the following is correct:
## *You input 6 studies
## *You selected limma limma limma limma limma limma for your 6 studies r
espectively
## * Fisher was chosen to combine the 6 studies,respectively
## dataset 1 is done
## dataset 2 is done
## dataset 3 is done
## dataset 4 is done
## dataset 5 is done
## dataset 6 is done
## Parametric method was used instead of permutation
##        ind.stat.Welsh ind.stat.Yu ind.stat.Lapointe ind.stat.Varamball
y
## GPR12     -0.01447604  0.37943628        0.28513849           0.108604
8
## RPS19      0.20673665  0.23162454        0.77668332           0.188085
4
## GALNT2    -0.11916266  0.11168920        0.36585563           0.387075
2
## MSI1       0.29758865  0.02367612        0.37455130           0.664799
6
## FCGRT     -0.36240927 -0.10877216       -0.53210661          -0.311400
9
## CD163      0.15648617  0.08638936        0.08651916           1.287503
4
##        ind.stat.Singh ind.stat.Wallace ind.p.Welsh    ind.p.Yu ind.p.L
apointe
## GPR12    -0.106957489       -0.3301548 0.978695463 0.006504890   2.762
384e-01
## RPS19     1.026749249       -0.4766147 0.016116661 0.002484948   3.768
341e-10
## GALNT2    0.005348824        0.2825572 0.428375201 0.437773064   8.001
044e-02
## MSI1     -0.180755089       -0.2662394 0.669863110 0.860595368   2.907
234e-03
## FCGRT     0.022247198        0.1003269 0.000154019 0.155941050   4.359
881e-06
## CD163    -0.211345323       -0.2461299 0.464836081 0.624141614   6.057
858e-01
##        ind.p.Varambally  ind.p.Singh ind.p.Wallace      stat         p
val
## GPR12        0.81888709 0.0354446198  5.153568e-02  25.69660 1.184580e
-02
```

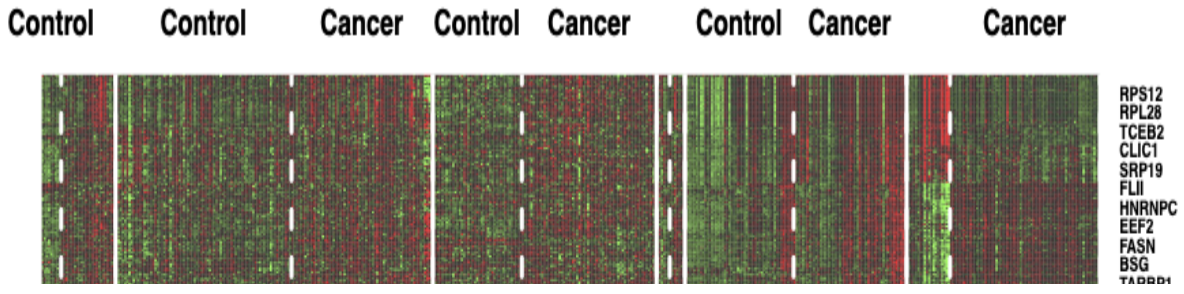
```
## RPS19         0.14991966 0.0005284280  6.684494e-06 106.36720 3.121382e
-17
## GALNT2        0.05198725 0.9191733351  2.091679e-02  22.21530 3.517649e
-02
## MSI1          0.26369912 0.0002454798  3.765074e-01  34.02685 6.679961e
-04
## FCGRT         0.13151895 0.8052767733  4.739980e-01  51.94301 6.345524e
-07
## CD163         0.04866036 0.0040981743  2.567059e-01  23.23721 2.577828e
-02
##                   FDR
## GPR12  1.563039e-02
## RPS19  5.358373e-16
## GALNT2 4.258009e-02
## MSI1   1.116283e-03
## FCGRT  1.846613e-06
## CD163  3.195602e-02
```

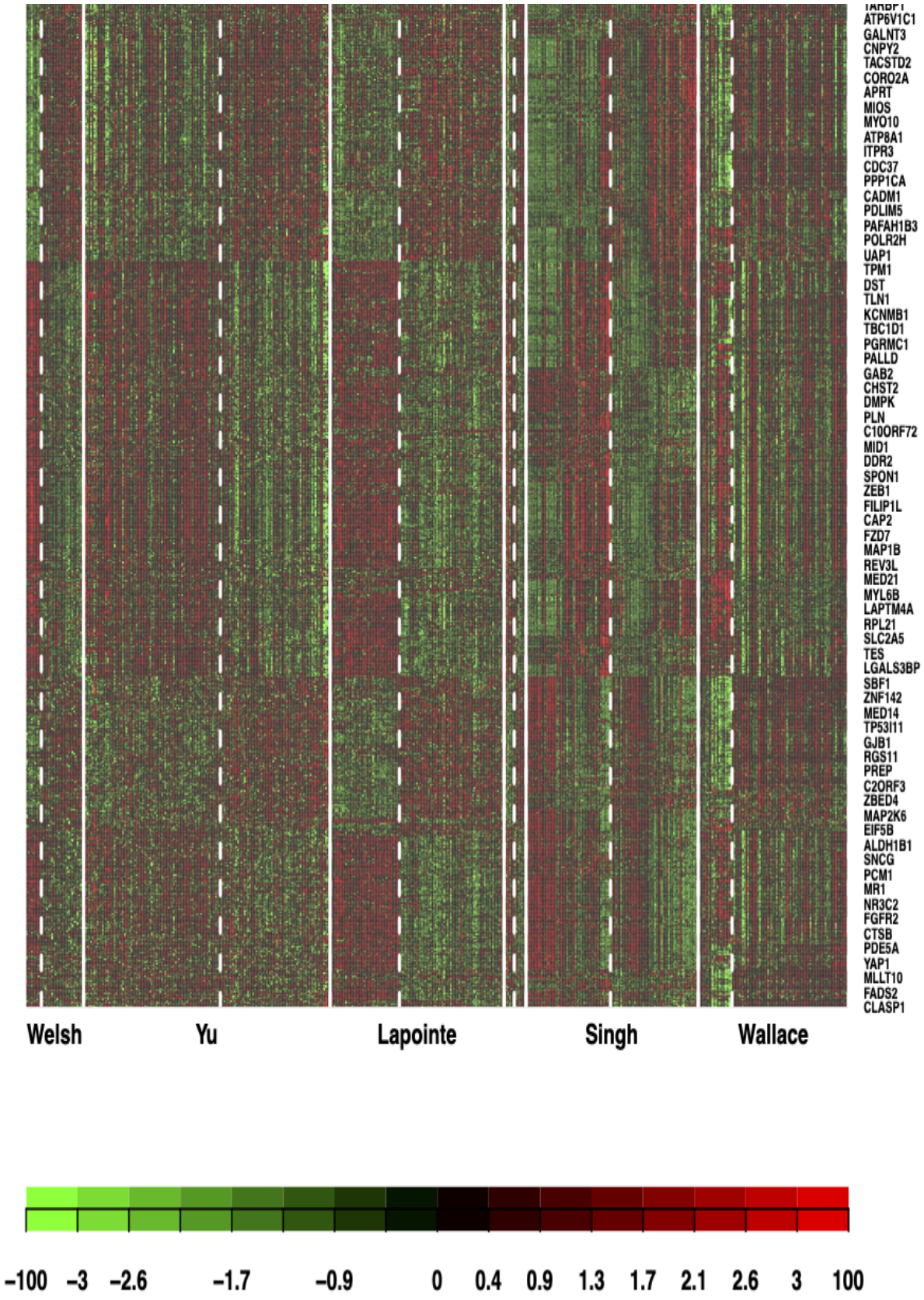
> If you were unable to run the MetaDE analyses, you can load the results of the above commands to review the output by usingh `load("./results/resMetaDE.rda")`

6.1 Plot meta analyses results in a heatmap.

R

```
fdr.cut <- 1e-7
pdf('./results/heatmap.DE.pdf')
heatmap.sig.genes(meta.res, meta.method=meta.method,
                    fdr.cut=fdr.cut,color="GR")
dev.off()
```




ATP6V1C1
GALNT3
CNPY2
TACSTD2
CORO2A
APRT
MIOS
MYO10
ATP8A1
ITPR3
CDC37
PPP1CA
CADM1
PDLIM5
PAFAH1B3
POLR2H
UAP1
TPM1
DST
TLN1
KCNMB1
TBC1D1
PGRMC1
PALLD
GAB2
CHST2
DMPK
PLN
C10ORF72
MID1
DDR2
SPON1
ZEB1
FILIP1L
CAP2
FZD7
MAP1B
REV3L
MED21
MYL6B
LAPTM4A
RPL21
SLC2A5
TES
LGALS3BP
SBF1
ZNF142
MED14
TP53I11
GJB1
RGS11
PREP
C2ORF3
ZBED4
MAP2K6
EIF5B
ALDH1B1
SNCG
PCM1
MR1
NR3C2
FGFR2
CTSB
PDE5A
YAP1
MLLT10
FADS2
CLASP1
Welsh
Yu
Lapointe
Singh
Wallace
−100 −3 −2.6 −1.7 −0.9 0 0.4 0.9 1.3 1.7 2.1 2.6 3 100

heatmap

6.2 Pathway analyses based on meta analyses

At this stage, one can do pathway analyses directly on the genes that appear to be differentially regulated based on the meta analyses performed above.

R

```
meta.p  <- meta.res$meta.analysis$pval
ks.result <- PathAnalysis(meta.p = meta.p, enrichment = "KS")
fisher.result <- PathAnalysis(meta.p = meta.p, enrichment = "Fisher's exa
        ct")



## One can customize pathway datbase to update the results like following
load(file =  "./data/pathwayDatabase.rda")

## Let us look into the database
pathway_names <- pathwayDatabase %>% names() %>% stringr::str_split(patte
        rn = "_", simplify = TRUE) %>% {.[,1]} %>% unique()
print(pathway_names)

path.res <- MetaDE::PathAnalysis(meta.p ,pathway = pathwayDatabase, p.cut
        = 0.05,
                    enrichment = "Fisher's exact", DEgene.number = 400, s
        ize.min = 10, size.max = 500)
```

> The pathway analyses performed here is based on the p-values obtained after the joint meta analyses of all the studies in your data. Some time individual study level pathway analyses is reported and we are interested in combining pathway level summary statistics. We will explore that in our next session where we will perform the analyses using **“MetaPath”** package.

6.3 Exercise

- Perform DE meta analyses using any two methods among “AW-Fisher”, “FEM,”REM and “minMCC” and compare the number of differentially expressed genes detected at FDR <0.05. Take a look at the /Library/Frameworks/R.framework/Versions/4.0/Resources/library/MetaDE/help/MetaDE function to change the parameters.
- Perform pathway analyses based one of the methods you tried and see whether any differences at pathway level were detected.

R

```
meta.method <- "AW"
meta.res <- MetaDE(data=data,clin.data = clin.data,
                   data.type=data.type,resp.type = resp.type,
                   response='label',covariate = NULL,
                   ind.method=ind.method, meta.method=meta.method,
                   select.group = select.group, ref.level=ref.level,
                   paired=paired, rth=NULL,
                   REM.type=NULL,tail='abs',parametric=TRUE)
```

```
## Please make sure the following is correct:
## *You input 6 studies
## *You selected limma limma limma limma limma limma for your 6 studies r
espectively
## * AW was chosen to combine the 6 studies,respectively
## dataset 1 is done
## dataset 2 is done
## dataset 3 is done
## dataset 4 is done
## dataset 5 is done
## dataset 6 is done
## Parametric method was used instead of permutation
```

R

```
meta.method <- "FEM"
meta.res <- MetaDE(data=data,clin.data = clin.data,
                   data.type=data.type,resp.type = resp.type,
                   response='label',
                   ind.method=ind.method, meta.method=meta.method,
                   select.group = select.group, ref.level=ref.level,
                   paired=paired, tail='abs')
```

```
## Please make sure the following is correct:
## *You input 6 studies
## * FEM was chosen to combine the 6 studies,respectively
```

R

```
meta.method <- "REM"
REM.type <- "HO"
meta.res <- MetaDE(data=data,clin.data = clin.data,
                   data.type=data.type,resp.type = resp.type,
                   response='label',
                   ind.method=ind.method, meta.method=meta.method,
                   select.group = select.group, ref.level=ref.level,
                   paired=paired,
                   REM.type=REM.type,tail='abs')
```

```
## Please make sure the following is correct:
## *You input 6 studies
## * REM was chosen to combine the 6 studies,respectively
```

R

```
meta.method <- 'minMCC'
meta.res <- MetaDE(data=data,clin.data = clin.data,
                   data.type=data.type,resp.type = resp.type,
                   response='label',
                   ind.method=ind.method, meta.method=meta.method,
                   select.group = select.group, ref.level=ref.level,
                   paired=paired,tail='abs',parametric=FALSE,nperm=100)
```

```
## Please make sure the following is correct:
## *You input 6 studies
## * minMCC was chosen to combine the 6 studies,respectively
```

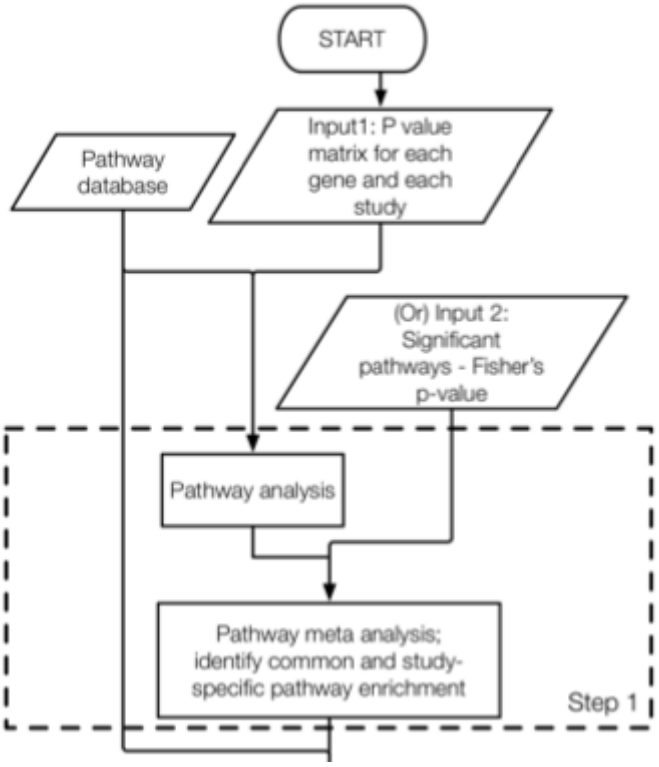
7 Pathway analyses

7.1 Introduction

Pathway analysis (a.k.a. gene set analysis) is a statistical tool to infer correlation of differential expression evidence in the data with pathway knowledge from established databases. The idea behind pathway analysis is to determine if there is enrichment in the detected DE genes based on an *a priori* defined biological category. Such a category might come from one or multiple databases such as Gene Ontology (GO; www.geneontology.org (http://www.geneontology.org/)), the Kyoto Encyclopedia of Genes and Genomes (KEGG; http://www.genome.jp/kegg/ (http://www.genome.jp/kegg/)), Biocarta Pathways (http://www.biocarta.com/ (http://www.biocarta.com/)) and the comprehensive Molecular Signatures Database (MSigDB; http://www.broadinstitute.org/gsea/msigdb/ (http://www.broadinstitute.org/gsea/msigdb/)). For the majority of recent microarray meta-analysis applications, pathway analysis has been a standard follow-up to identify pathways associated with detected DE genes e.g. (https://pubmed.ncbi.nlm.nih.gov/17974971/) and many others]. The result provides more insightful biological interpretation and it has been reported that pathway analysis results are usually more consistent and reproducible across studies than DE gene detection. Shen and Tseng (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2865865/) developed a systematic framework of Meta-Analysis for Pathway Enrichment (MAPE) by combining information at gene level, at pathway level and a hybrid of the two. We will use this package for systematic analyses of microarray data from different studies.

7.2 Step1: MAPE_P analysis

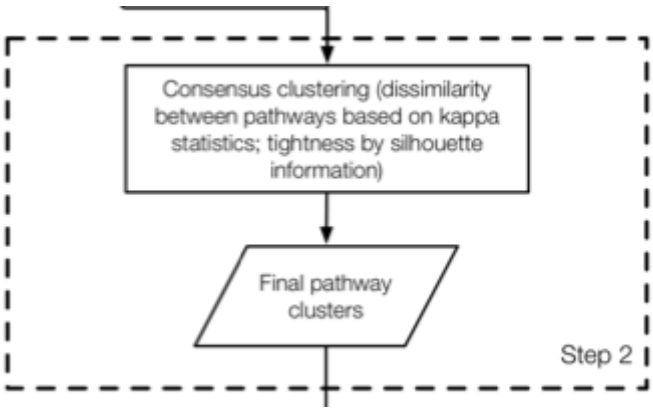
This is the first major function in the MetaPath2.0 package which combines the Meta-analysis for Pathway Enrichment (MAPE) methods introduced by Shen and Tseng (2010) and the Comparative Pathway Integrator (CPI) method introduced by Fang and Tseng (2016) (https://www.biorxiv.org/content/10.1101/444604v1.full.pdf). The default function is CPI which performs MAPE_P (integrating multiple studies at pathway level) with Adaptively Weighted Fisher's method as Meta-analysis statistics.



R

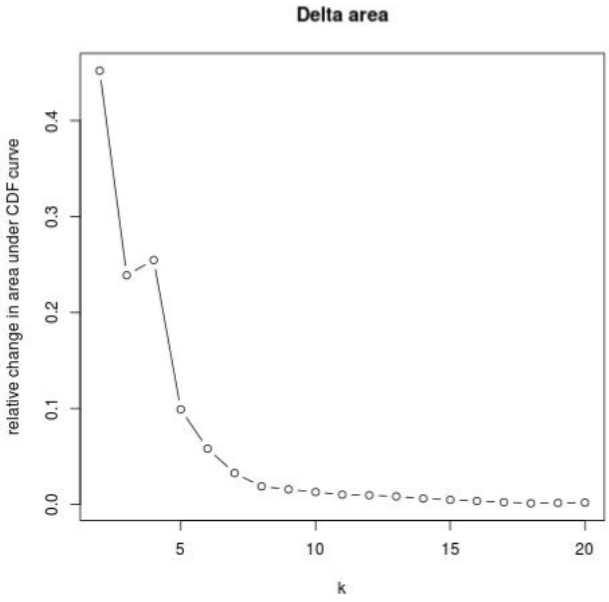
```
#load("./data/pathways.rda")

CPI_result = MAPE2.0(arraydata = data , clinical.data = clin.data,
                     label = "label",pmtx = NULL,pathway = c(Biocarta.gene
        sets, GOBP.genesets,
        GOCC.genesets, GOMF.genesets, KEGG.genesets, Reactome.genesets),
        data.type ="discrete",
        resp.type = "twoclass",method = "CPI", ind.method = rep("limma",l
        ength(data)),
        paired =rep(FALSE,length(data)),select.group=select.group, ref.le
        vel=ref.level ,
        tail="abs", enrichment = "Fisher's exact", DEgene.number = 400,st
        at = "AW Fisher")


save(CPI_result, file = "./results/MetaPEResults/MAPE_p_pathways.rds")
```

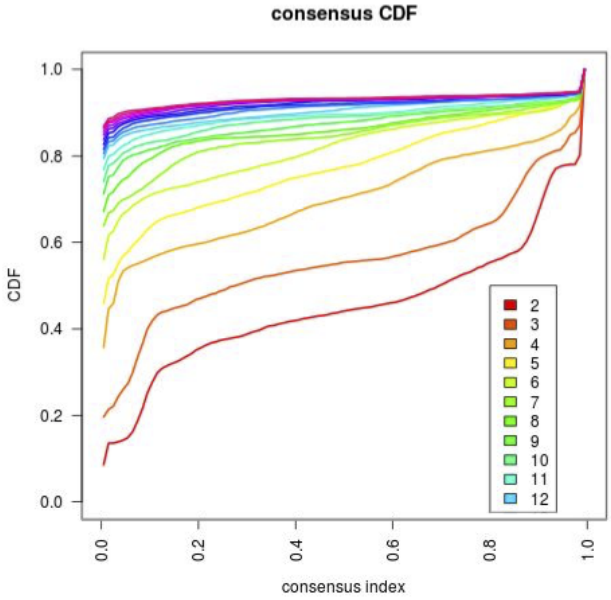
7.3 Step 2



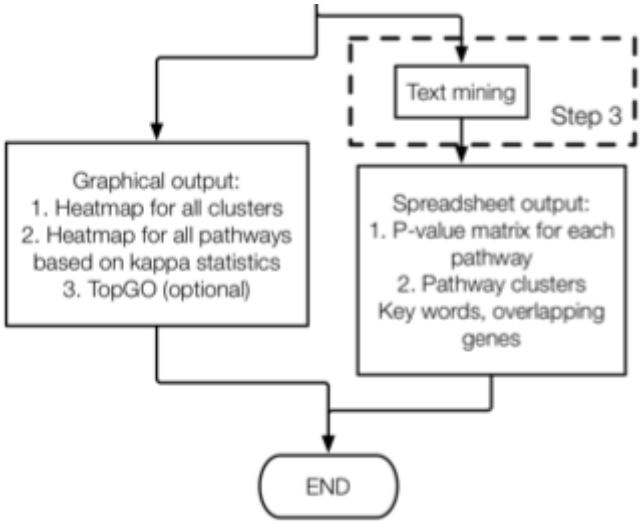
R

```
set.seed(15213)
CPI.kappa_result = MAPE.Kappa(summary = CPI_result$summary,pathway = CPI_
        result$pathway,
                              max_k = 10, q_cutoff = 0.05,software = CPI_
        result$method, output_dir = "./results/MetaPEResults/")
```


Delta area
relative change in area under CDF curve
0.0
0.1
0.2
0.3
0.4
5
10
15
20
k


consensus CDF
CDF
0.0
0.2
0.4
0.6
0.8
1.0
consensus index
2
3
4
5
6
7
8
9
10
11
12

7.4 Step 3

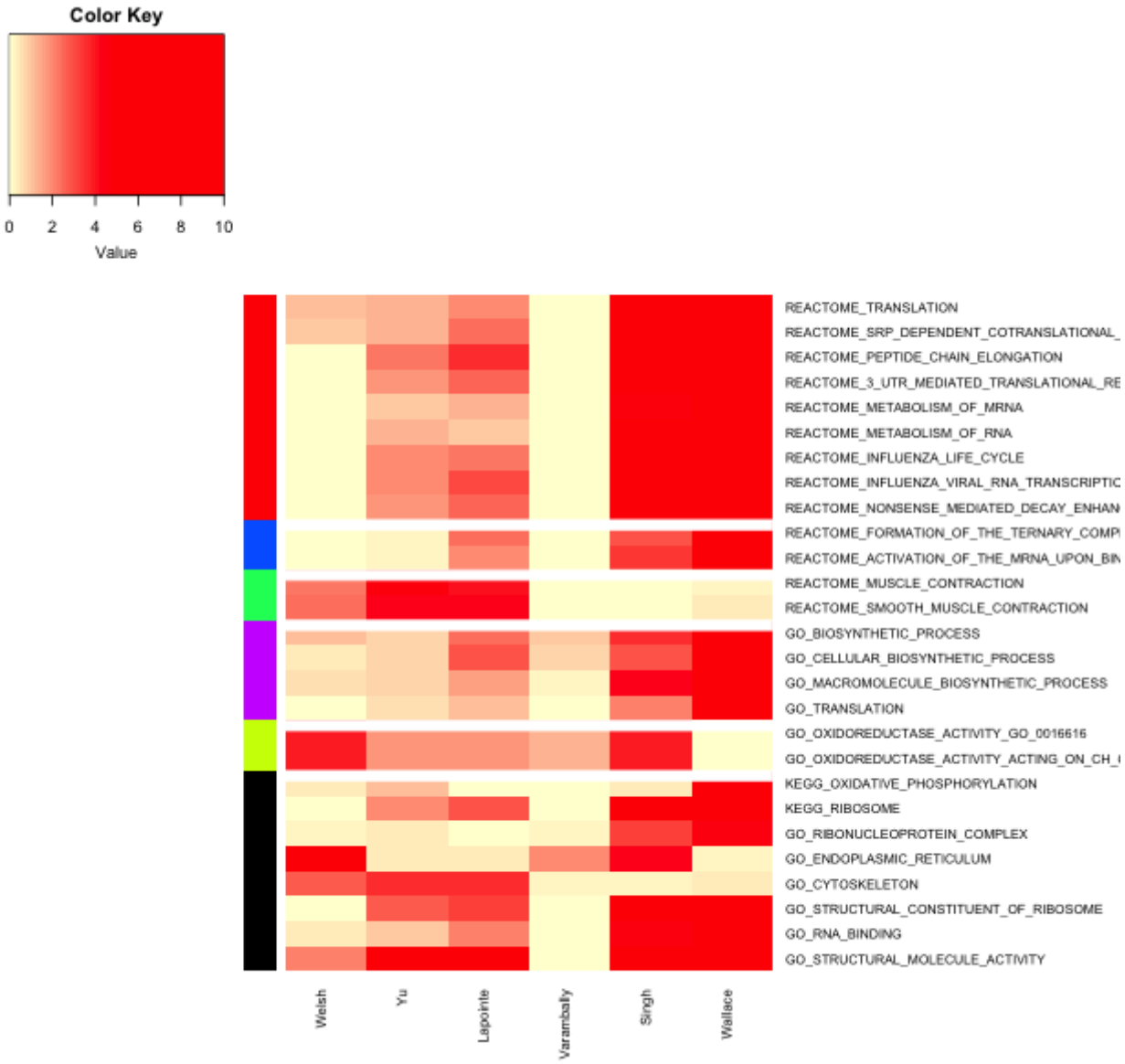

Text mining
Step 3
Graphical output:
1. Heatmap for all clusters
2. Heatmap for all pathways based on kappa statistics
3. TopGO (optional)
Spreadsheet output:
1. P-value matrix for each pathway
2. Pathway clusters
Key words, overlapping genes
END

R

```
data(hashtb)
#data("./data/hashtb.rda")
#xx <- data(hashtb)
set.seed(1)
MAPE.Clustering.results <- MAPE.Clustering(summary=CPI_result$summary,Num
        _Clusters = 6,
                           kappa.result = CPI.kappa_result$kappa,sil
        _cut=0.01,
                           Num_of_gene_lists=CPI_result$Num_of_gene_
        lists,genelist =CPI_result$genelist,
                           pathway=CPI_result$pathway, enrichment=CP
        I_result$enrichment,
                           method=CPI.kappa_result$method,software=C
        PI_result$method,                                  n.text.permute =
        1000, output_dir = "./results/MetaPEResults/"  )
```

You can take a look at the “Clustering_Summary.csv” to gen an idea of reproducibility of pathways and the amount of evidence provided by each study.

Of the heat

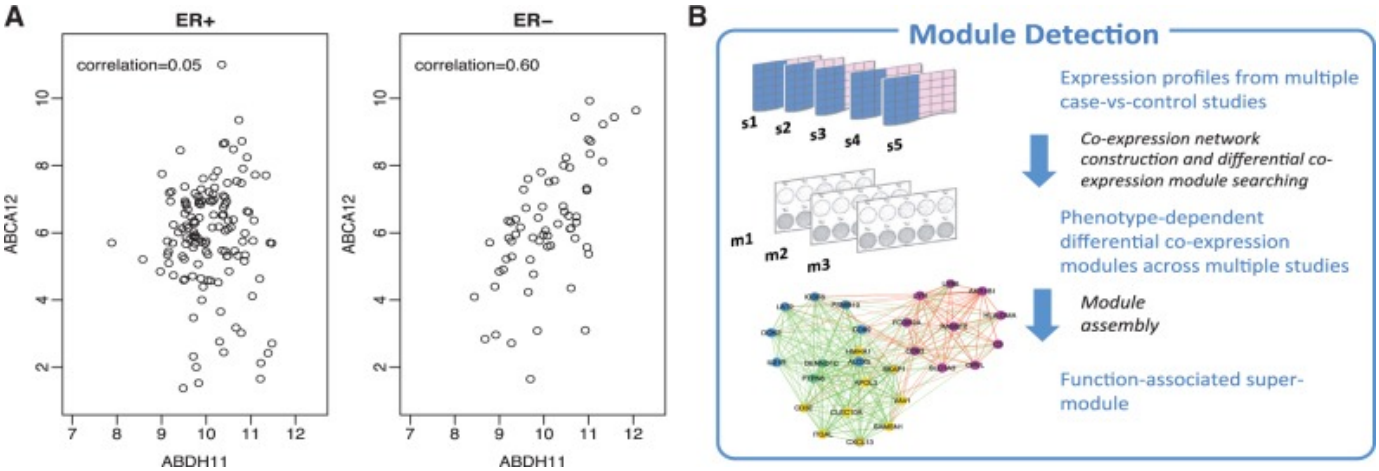


8 Network Meta analyses using MetaDCN

Co-expression analysis and network analysis of microarray data are used to investigate potential transcriptional co-regulation and gene interactions. Network analyses typically work with the gene–gene co-expression matrix, which represents the correlation between each pair

of genes in the study. A crucial assumption is that the magnitude of the co-expression between any pair of genes is associated with a greater likelihood that the two genes interact. Thus, networks of interactions between genes are inferred from the co-expression matrix.

Here we will use MetaDCN (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6041767/) to extract network modules in disease vs control conditions.



MetaDCN

8.1 Correction and adjacency matrices

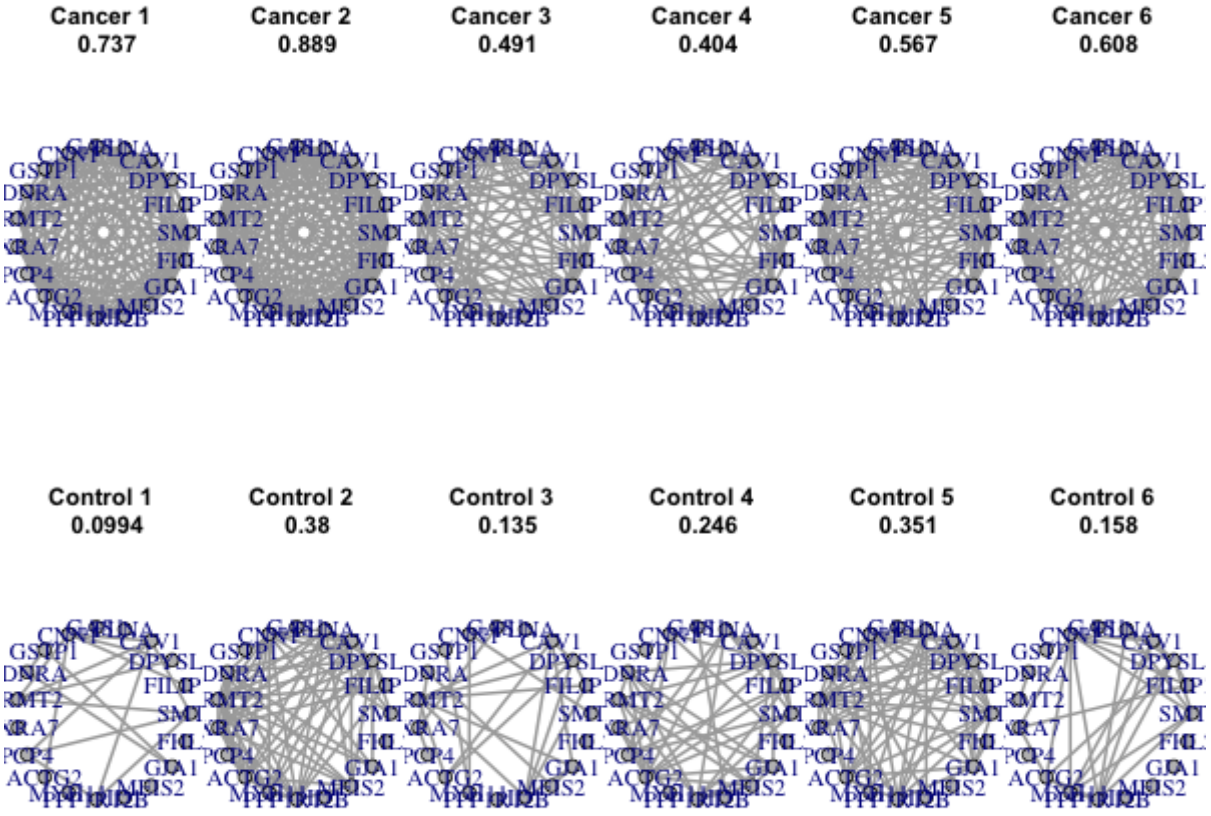
R

```
#data("./data/pathwayDatabase.rda")
#data(example)
# Generate correction and adjacency matrices for data and permutation
# GeneNet returns a list of information which will be used for SearchBM a
        nd MetaDCN function, and several RData files stored in folder pat
        h.
# AdjacencyMatrices.RData is a list of adjacency matrices for case and co
        ntrol in each study in the order of case studies and control stud
        ies.
#
# CorrelationMatrices.RData is a list of correlation matrices for case an
        d control in each study.
#
# AdjacencyMatricesPermutationP.RData is a list of correlation matrices f
        or case and control in each study in permutation P.
GeneNetRes.2  <- GeneNet(data , clin.data, caseName="Cancer", controlName
        ="Control", meanFilter=0.8, SDFilter=0.8, edgeCutoff=0.1, permuta
        tionTimes=4, CPUNumbers=4,  pathwayDatabase=c(Biocarta.genesets,G
        OBP.genesets,GOCC.genesets,GOMF.genesets,
                    KEGG.genesets,Reactome.genesets), silent=FALSE, folde
        r = "./results/MetaDCNResults/")
```

8.2 Correlations/Co-expressions

```
R
```

```
# This function will search for basic modules differentially co-expressed
        between case and control
# SearchBM will return a list and several Rdata, csv and png files saved
         in the folder path specified in GeneNet inputs.List of basic mod
        ule information:


# w1
# w1 weight with the most basic modules detected
#
# BMInCase
# data matrix listing the information of basic modules higher correlated
         in case
#
# BMInControl
# data matrix listing the information of basic modules higher correlated
         in control
#
# permutation_energy_direction_p.Rdata is a list of energies of basic mod
        ules from permutation p.
#
# basic_modules_summary_direction_weight_w.csv is a summary of basic modu
        les detected using weight w in forward/backward search.
#
# threshold_direction.csv is a table listing number of basic modules dete
        cted under different FDRs in forward/backward search.
#
# figure_basic_module_c_repeat_r_direction_weight_w.png is a plot of basi
        c module from component c repeat r using weight w in forward/back
        ward search.

SearchBMRes <- SearchBM(GeneNetRes.2, MCSteps=500, jaccardCutoff=0.7, rep
        eatTimes=5, outputFigure=TRUE, silent=FALSE )
```



8.3 Find and assemble basic modules

R

```
#This function will assemble basic modules detected from SearchBM into su
        permodules.
# w1:w1 used
#
# BMInCaseSig: Summary of basic modules higher correlated in Case control
        ing FDR
#
# BMInControlSig
# Summary of basic modules higher correlated in Control controling FDR
#
# Supermodule: Summary of supermodules
#A number chosen from (100, 200, ..., 700) to specify the weight1 used in
        objective function (optional). If not specified, w1 from SearchBM
        function will be used (recommended).


MetaDNCRes <- MetaDCN(GeneNetRes.2, SearchBMRes, FDRCutoff=0.05, w1=NULL,
        silent=FALSE)
```

You get the modules, their pathway annotations and cytoscape files in the results directory.

For the final assembled modules have a look at the “module_assembly_summary_weight_500.csv” and find out ho many “supe modules you are able to identity on q value cut off of 0.05”

9 Session info

```
OUTPUT
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/lib
Rblas.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/lib
Rlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
##  [1] grid      parallel  stats4    stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
##  [1] MetaDCN_0.0-1               igraph_1.2.6
##  [3] snowfall_1.84-6.1           snow_0.4-3
##  [5] MetaPath_2.0                shape_1.4.5
##  [7] Rgraphviz_2.34.0            ggplot2_3.3.3
##  [9] gplots_3.1.1                cluster_2.1.1
## [11] irr_0.84.1                  lpSolve_5.6.15
## [13] ConsensusClusterPlus_1.54.0 impute_1.64.0
## [15] genefilter_1.72.1           GSEABase_1.52.1
## [17] graph_1.68.0                annotate_1.68.0
## [19] XML_3.99-0.6                MetaDE_2.2.3
## [21] MetaQC_1.0                  multtest_2.46.0
## [23] GSA_1.03.1                  preproc_1.2-4
## [25] DMwR_0.4.1                  lattice_0.20-41
## [27] AnnotationDbi_1.52.0        IRanges_2.24.1
## [29] S4Vectors_0.28.1            Biobase_2.50.0
## [31] BiocGenerics_0.36.0         plyr_1.8.6
## [33] magrittr_2.0.1              captioner_2.2.3
## [35] bookdown_0.21               knitr_1.31
##
## loaded via a namespace (and not attached):
##  [1] colorspace_2.0-0            ellipsis_0.3.1
##  [3] class_7.3-18                XVector_0.30.0
##  [5] GenomicRanges_1.42.0        fs_1.5.0
##  [7] bit64_4.0.5                 fansi_0.4.2
##  [9] splines_4.0.3               cachem_1.0.4
## [11] geneplotter_1.68.0          jsonlite_1.7.2
```

```
## [13] shiny_1.6.0                  compiler_4.0.3
## [15] httr_1.4.2                   Matrix_1.3-2
## [17] fastmap_1.1.0                limma_3.46.0
## [19] later_1.1.0.1                htmltools_0.5.1.1
## [21] tools_4.0.3                  gtable_0.3.0
## [23] glue_1.4.2                   GenomeInfoDbData_1.2.4
## [25] Rcpp_1.0.6                   jquerylib_0.1.3
## [27] vctrs_0.3.7                  xfun_0.22
## [29] stringr_1.4.0                openxlsx_4.2.3
## [31] mime_0.10                    lifecycle_1.0.0
## [33] gtools_3.8.2                 edgeR_3.32.1
## [35] zlibbioc_1.36.0              MASS_7.3-53.1
## [37] zoo_1.8-9                    scales_1.1.1
## [39] promises_1.2.0.1             MatrixGenerics_1.2.1
## [41] SummarizedExperiment_1.20.0 RColorBrewer_1.1-2
## [43] yaml_2.2.1                   quantmod_0.4.18
## [45] curl_4.3                     memoise_2.0.0
## [47] sass_0.3.1                   rpart_4.1-15
## [49] stringi_1.5.3                RSQLite_2.2.5
## [51] highr_0.8                    TTR_0.24.2
## [53] caTools_1.18.2               zip_2.1.1
## [55] BiocParallel_1.24.1          GenomeInfoDb_1.26.7
## [57] rlang_0.4.10                 pkgconfig_2.0.3
## [59] matrixStats_0.58.0           bitops_1.0-6
## [61] evaluate_0.14                ROCR_1.0-11
## [63] bit_4.0.4                    samr_3.0
## [65] DESeq2_1.30.1                R6_2.5.0
## [67] combinat_0.0-8               DelayedArray_0.16.3
## [69] DBI_1.1.1                    withr_2.4.1
## [71] pillar_1.5.1                 xts_0.12.1
## [73] survival_3.2-10              abind_1.4-5
## [75] RCurl_1.98-1.3               tibble_3.1.0
## [77] crayon_1.4.1                 shinyFiles_0.9.0
## [79] KernSmooth_2.23-18           utf8_1.2.1
## [81] rmarkdown_2.7                locfit_1.5-9.4
## [83] blob_1.2.1                   digest_0.6.27
## [85] xtable_1.8-4                 httpuv_1.5.5
## [87] munsell_0.5.0                bslib_0.2.4
```

End of document