

# SMGR: Setup and Introduction

*SMGR (Joint statistical method for integrative analysis of single-cell multi-omics data)*

Vignette built on April 20, 2022.

## SMGR workflow

This tutorial goes through the steps in the SMGR workflow:

- Identify consistent signals of genes and peaks based on scRNA-seq & scATAC-seq data.
- Reveal the latent representation of co-expressed genes and peaks, i.e. co-regulatory program
- Identify co-expressed TF and target genes, based on single-cell multi-omics analysis

The setup and running steps are shown as follows in this tutorial:

## Requirements

- Input: expression matrix The input to SMGR is a list of scRNA-seq expression matrix and scATAC-seq matrix:

*For scRNA-seq matrix:* Each column corresponds to a sample (cell) and each row corresponds to a gene. Expression units: The preferred expression values are gene-summarized raw counts. Other measurements (such as FPKM) are also accepted.

*For scATAC-seq matrix:* Each column corresponds to a sample (cell) and each row corresponds to a peak.

## Installation

The R implementation of SMGR is based on R packages as below.

Therefore, you will need to install these packages, and some extra dependencies, to run SMGR:

```
if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")
BiocManager::version()
```

```
## [1] '3.12'
```

```
# If your bioconductor version is previous to 4.0, see the section bellow
# Required
packages = c('glmnet', 'MASS', 'purrr', 'mpath', 'zic', 'pscl', 'parallel')
# BiocManager::install(packages)
```

Now you are ready to install SMGR:

```
# devtools::install_github("QSong-github/SMGR")
library('SMGR')
```

*Note: please invoke these packages to run the following codes:*

```
pkg = c('glmnet', 'MASS', 'purrr', 'mpath', 'zic', 'pscl', 'parallel')
sapply(pkg, require, character.only = TRUE)
```

```
##   glmnet    MASS    purrr    mpath    zic    pscl parallel
##   TRUE     TRUE     TRUE     TRUE     TRUE    TRUE     TRUE
```

## Run SMGR with example data

Example data is deposited in the data folder.

```
# @param rna.cts simulated scRNA-seq data
rna.cts <- readRDS('simulation_scRNA-seq.RDS')

# @param atac.cts simulated scATAC-seq data
atac.cts <- readRDS(file='simulation_scATAC-seq.RDS')

input_data <- list(rna.cts, atac.cts)
```

### SMGR process

Input data is a list of scRNA-seq and scATAC-seq data

```
input_data <- list(as.matrix(rna.cts), as.matrix(atac.cts))

result1 <- smgr_main(sm.data = input_data, K=nrow(input_data[[1]]))
```

```
## ## =====
## ## |                                     running SMGR                                     |
## ## |-----|
## ## |          scRNA-seq data and scATAC-seq data identified ....          |
## ## |          Set initial values ....          |
## ## |          iter= 1.....          |
## ## |          iter= 2.....          |
## ## |          iter= 3.....          |
## ## |          iter= 4.....          |
## ## |          iter= 5.....          |
## ## |          iter= 6.....          |
## ## |          iter= 7.....          |
## ## |          iter= 8.....          |
## ## |          iter= 9.....          |
## ## |          iter=10.....          |
## ## |          iter=11.....          |
## ## |          iter=12.....          |
## ## |          iter=13.....          |
## ## |          iter=14.....          |
## ## |          iter=15.....          |
## ## |          iter=16.....          |
## ## |          iter=17.....          |
## ## |          iter=18.....          |
## ## |          iter=19.....          |
## ## |          iter=20.....          |
```

if output is set to 'all', you will obtain the coefficients and BIC values

```
# member1: is the ground truth
member1 <- c(rep(1,600),rep(2,600),rep(3,600))

# programs: co-expressed gene & peaks from latent representation
programs <- result1$clusters
```

report ARI score

```
library(clues)
print (adjustedRand(programs, member1))
```

```
##          Rand          HA          MA          FM  Jaccard
## 0.9985195 0.9966671 0.9966708 0.9977769 0.9955636
```

latent.var : latent representation of genes & peaks

```
latent.var <- result1$latent_f
```

evaluation metrics

eval : evaluation metrics

```
library(clValid); library(clusterCrit)

eval <- vector('list',length=2)
for ( i in 1:2 ){
  data <- log2(input_data[[i]]+1)
  ind1 <- Evaluate(data,programs)
  ind2 <- intCriteria(data,as.integer(programs),
                     c("Calinski_Harabasz",'Davies_Bouldin'))
  inds <- as.data.frame(rbind(do.call(rbind,ind2),ind1))
  if (i==1){ inds$data <- 'scRNA-seq' } else { inds$data <- 'scATAC-seq' }
  colnames(inds) <- c('metrics','data')
  eval[[i]] <- inds
}
```

Evaluation index for scRNA-seq data

```
print (eval[[1]])
```

```
##               metrics      data
## calinski_harabasz 129.95102608 scRNA-seq
## davies_bouldin    4.48655027  scRNA-seq
## Dunn              0.59328525  scRNA-seq
## Sihouette         0.05246821  scRNA-seq
```

```
eval[[1]][1,1] = log10(eval[[1]][1,1])
eval[[1]]$type = rownames(eval[[1]])
library(ggplot2)
```

```
## Warning: The packages 'ellipsis' (>= 0.3.2) and 'vctrs' (>= 0.3.8) are required
## as of rlang 1.0.0.
```

```
## Warning: replacing previous import 'ellipsis::check_dots_unnamed' by
## 'rlang::check_dots_unnamed' when loading 'tibble'
```

```
## Warning: replacing previous import 'ellipsis::check_dots_used' by
## 'rlang::check_dots_used' when loading 'tibble'
```

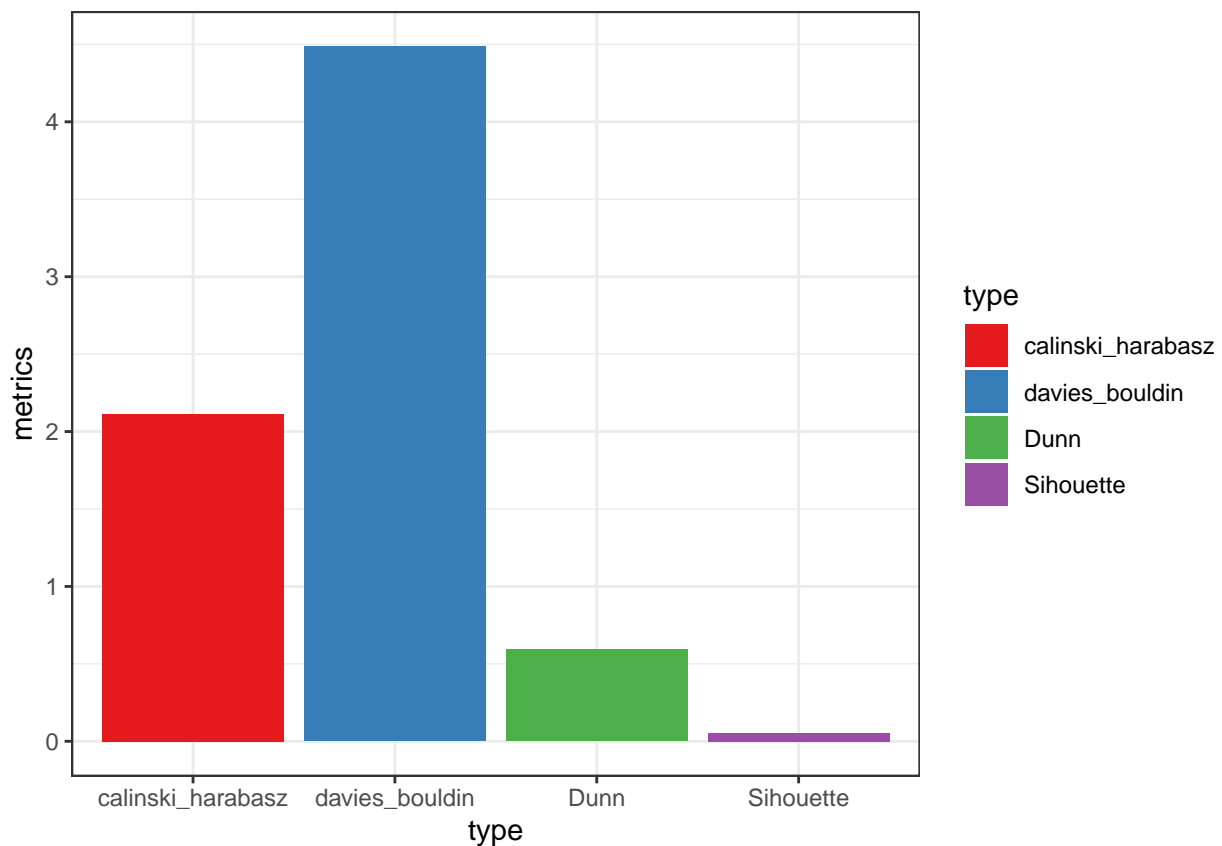
```
## Warning: replacing previous import 'ellipsis::check_dots_empty' by
## 'rlang::check_dots_empty' when loading 'tibble'
```

```
## Warning: replacing previous import 'ellipsis::check_dots_unnamed' by
## 'rlang::check_dots_unnamed' when loading 'pillar'
```

```
## Warning: replacing previous import 'ellipsis::check_dots_used' by
## 'rlang::check_dots_used' when loading 'pillar'
```

```
## Warning: replacing previous import 'ellipsis::check_dots_empty' by
## 'rlang::check_dots_empty' when loading 'pillar'
```

```
p <- ggplot(eval[[1]], aes(x=type, y=metrics, fill=type)) +
  geom_bar(stat='identity')+
  scale_fill_brewer(palette='Set1')+ theme_bw()
print (p)
```

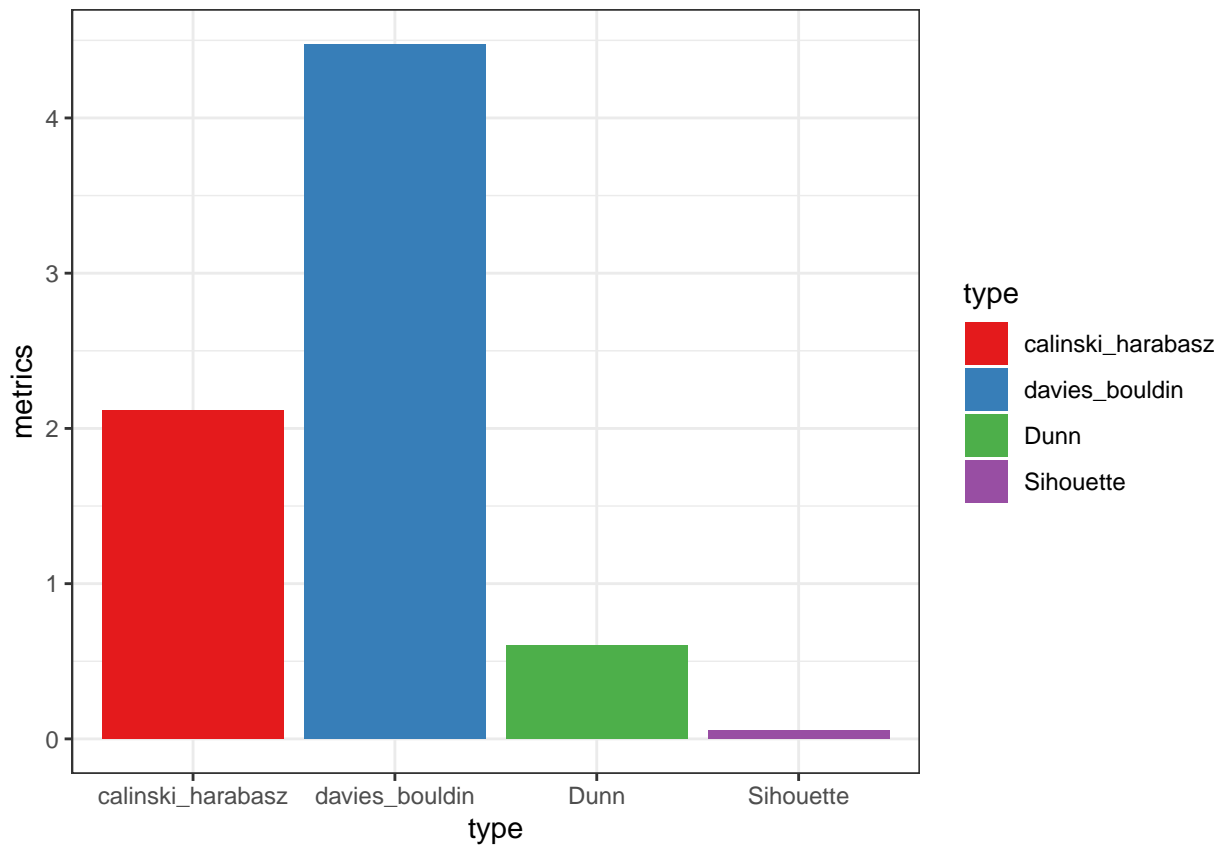


*Evaluation index for scATAC-seq data*

```
print (eval[[2]])
```

```
##               metrics      data
## calinski_harabasz 129.9482711 scATAC-seq
## davies_bouldin    4.4759378 scATAC-seq
## Dunn              0.5991397 scATAC-seq
## Sihouette         0.0526948 scATAC-seq
```

```
eval[[2]][1,1] = log10(eval[[2]][1,1])
eval[[2]]$type = rownames(eval[[2]])
p <- ggplot(eval[[2]], aes(x=type, y=metrics, fill=type)) +
  geom_bar(stat='identity') +
  scale_fill_brewer(palette='Set1') + theme_bw()
print (p)
```



### Visualization of results data

*for scRNA-seq data*

```
library(ComplexHeatmap)
d1 <- log2(input_data[[1]]+1)
```

*Suppose these programs are identified in different cell types*

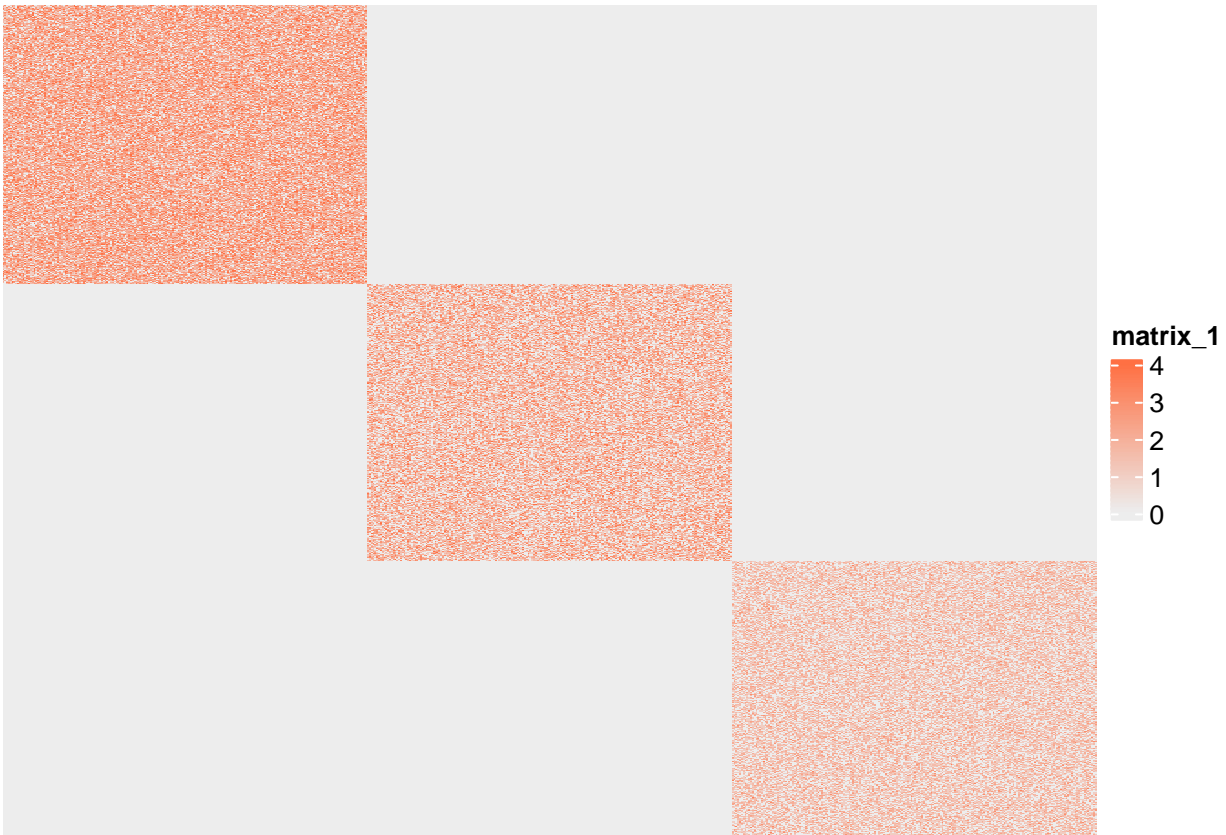
```
m1 <- d1[programs==1,]; m2 <- d1[programs==2,]; m3 <- d1[programs==3,]
program.d <- rbind(cbind(m1,matrix(0,nrow=nrow(m1),ncol=400)),
                  cbind(matrix(0,nrow=nrow(m2),ncol=200),m2,matrix(0,nrow=nrow(m2),ncol=200)),cbind(ma
```

*color setting*

```
library(circlize)
c1 <- c("#EDEDFF","dodgerblue"); c1 <- c("#EDEDFF","#FF7043FF")
myCol1 <- colorRampPalette(c1)(100); myBreaks1 <- seq(0, 4, length.out = 100)
```

## Heatmap

```
print (Heatmap(program.d,
               cluster_columns = FALSE,cluster_rows =F,
               show_column_names = FALSE,show_row_names=FALSE,
               col = colorRamp2(myBreaks1, myCol1)))
```



## Run SMGR with your own data

For your own real data: here we use the simulated toy data to demonstrate the process

```
#devtools::install_github("bowang-lab/simATAC")
library(simATAC);
count <- getCountFromh5("GSE99172.snap") # GSE99172.snap is accessible from simATAC package: https://d
obj1 <- simATACEstimate(t(count))
obj1 <- setParameters(obj1, noise.mean = -0.3, noise.sd = 0.1)
obj1 <- simATACSimulate(obj1, nCells = 200)
```

the .gtf file can be downloaded from <https://grch37.ensembl.org/info/data/ftp/index.html>

map the link between peak regions and genes:

```
# atac.cts: toy scATAC data
atac.cts = linkActivityMatrix(peak.matrix = counts(obj1)[1:6000,1:200],
                             annotation.file = 'Homo_sapiens.GRCh37.82.gtf',
                             seq.levels = c(1:22,'X','Y'),
                             upstream = 2000)$activity_matrix[1:600,1:200]

# rna.cts: toy scRNA-seq data
library(splatter); library(scater)
sim.groups <- splatSimulate(group.prob = 1, method = "groups",
                           verbose = FALSE, nGenes=600, batchCells=200)
rna.cts <- counts(sim.groups)
rownames(rna.cts) <- rownames(atac.cts);
colnames(rna.cts) <- paste0('cell_',1:ncol(rna.cts))
```

input\_data: with a list of scRNA-seq and scATAC-seq toy data

```
input_data <- list(rna.cts, atac.cts)
```

from here, you can start with the smgr function and use the evaluation index to check if you achieve good performance of idnetifying co-regulation program

```
library(SMGR)
result1 <- smgr_main(sm.data = input_data, K=nrow(input_data[[1]]))
```

```
## ## =====
## ## | running SMGR |
## ## -----
## ## | scRNA-seq data and scATAC-seq data identified .... |
## ## | Set initial values .... |
## ## | iter= 1.... |
## ## | iter= 2.... |
## ## | iter= 3.... |
## ## | iter= 4.... |
## ## | iter= 5.... |
## ## | iter= 6.... |
## ## | iter= 7.... |
## ## | iter= 8.... |
## ## | iter= 9.... |
## ## | iter=10.... |
## ## | iter=11.... |
## ## | iter=12.... |
```

|       |              |  |
|-------|--------------|--|
| ## ## | iter=13..... |  |
| ## ## | iter=14..... |  |
| ## ## | iter=15..... |  |
| ## ## | iter=16..... |  |
| ## ## | iter=17..... |  |
| ## ## | iter=18..... |  |
| ## ## | iter=19..... |  |
| ## ## | iter=20..... |  |