Figure 4 pAML reproducibility similarity analysis

Qun Li

22 Aug 2025 at 10:12:37

Contents

1 Env **1**

2 Load libraries **1**

3 define functions **8**

4 load data **10**

5 mutation Similarity DXREL for lineage switching (downsample) **12**

6 calSimilarity downsampling **15**

7 permutation test **46**

8 mutations shared by different lineage **66**

9 SNV tree cell type **71**

10 mutratio **78**

11 sessionInfo **81**

1 Env

```
rm(list = ls())
setwd("/Users/liqun/Desktop/scMutrace-tutorial/ReproducibilityAnalysis/Figure4/")
```

2 Load libraries

```
library(FSA)
## ## FSA v0.10.0. See citation('FSA') if used in publication.
## ## Run fishR() for related website and fishR('IFAR') for related book.
library(ineq)
library(grid)
library(vegan)
## Loading required package: permute
library(ArchR)
##
##                                                    / |
##                                                  /    \
##             .                                  /      |.
##             \\\                              /        |.
##               \\\                          /           `|.
##                 \\\                      /              |.
##                   \                    /                |\
##                   \\#####\           /                  ||
##                 ==###########>      /                   ||
##                  \\##==......\    /                     ||
##             ______ =       =|__ /__                     ||      \\\
##         ,--' ,----`-,__ ___/'  --,-`-===================##========>
##        \               '        ##_______ _____ ,--,__,=##,__   ///
##         ,    __==    ___,-,__,--'#'  ==='      `-'    | ##,-/
##         -,____,---'       \\####\\________________,--\\_##,/
##            ___      .______        ______  __    __  .______
##           /   \     |   _  \      /      ||  |  |  | |   _  \
##          /  ^  \    |  |_)  |    |  ,----'|  |__|  | |  |_)  |
##         /  /_\  \   |      /     |  |     |   __   | |      /
##        /  _____  \  |  |\  \___ |  `----.|  |  |  | |  |\  \___.
##       /__/     \__\ | _| `._____| \______||__|  |__| | _| `._____|
##
## ArchR : Version 1.0.3
## For more information see our website : www.ArchRProject.com
## If you encounter a bug please report : https://github.com/GreenleafLab/ArchR/issues
## Loading Required Packages...
##  Loading Package : devtools v2.4.5
##  Loading Package : grid v4.5.1
##  Loading Package : gridExtra v2.3
##  Loading Package : gtools v3.9.5
##  Loading Package : gtable v0.3.6
##  Loading Package : ggplot2 v4.0.0
##  Loading Package : magrittr v2.0.4
##  Loading Package : plyr v1.8.9
##  Loading Package : stringr v1.5.2
##  Loading Package : data.table v1.17.8
##  Loading Package : matrixStats v1.5.0
##  Loading Package : sparseMatrixStats v1.20.0
##  Loading Package : S4Vectors v0.46.0
##  Loading Package : GenomicRanges v1.60.0
##  Loading Package : BiocGenerics v0.54.0
##  Loading Package : Matrix v1.7.4
##  Loading Package : Rcpp v1.1.0
##  Loading Package : RcppArmadillo v15.0.2.2
```

```
##  Loading Package : SummarizedExperiment v1.38.1
##  Loading Package : rhdf5 v2.52.1
library(dplyr)
##
## Attaching package: 'dplyr'
## The following object is masked from 'package:Biobase':
##
##     combine
## The following objects are masked from 'package:GenomicRanges':
##
##     intersect, setdiff, union
## The following object is masked from 'package:GenomeInfoDb':
##
##     intersect
## The following objects are masked from 'package:IRanges':
##
##     collapse, desc, intersect, setdiff, slice, union
## The following objects are masked from 'package:S4Vectors':
##
##     first, intersect, rename, setdiff, setequal, union
## The following objects are masked from 'package:BiocGenerics':
##
##     combine, intersect, setdiff, setequal, union
## The following object is masked from 'package:generics':
##
##     explain
## The following object is masked from 'package:matrixStats':
##
##     count
## The following objects are masked from 'package:data.table':
##
##     between, first, last
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
## The following object is masked from 'package:gridExtra':
##
##     combine
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
library(GO.db)
## Loading required package: AnnotationDbi
##
## Attaching package: 'AnnotationDbi'
## The following object is masked from 'package:dplyr':
##
##     select
##
```

```r
library(tidyr)
##
## Attaching package: 'tidyr'
## The following objects are masked from 'package:Matrix':
##
##     expand, pack, unpack
## The following object is masked from 'package:S4Vectors':
##
##     expand
## The following object is masked from 'package:magrittr':
##
##     extract
library(ggpubr)
## Registered S3 methods overwritten by 'car':
##   method       from
##   hist.boot    FSA
##   confint.boot FSA
##
## Attaching package: 'ggpubr'
## The following object is masked from 'package:plyr':
##
##     mutate
library(Seurat)
## Loading required package: SeuratObject
## Loading required package: sp
##
## Attaching package: 'sp'
## The following object is masked from 'package:IRanges':
##
##     %over%
## 'SeuratObject' was built under R 4.5.0 but the current version is
## 4.5.1; it is recomended that you reinstall 'SeuratObject' as the ABI
## for R may have changed
##
## Attaching package: 'SeuratObject'
## The following object is masked from 'package:SummarizedExperiment':
##
##     Assays
## The following object is masked from 'package:GenomicRanges':
##
##     intersect
## The following object is masked from 'package:GenomeInfoDb':
##
##     intersect
## The following object is masked from 'package:IRanges':
##
##     intersect
## The following object is masked from 'package:S4Vectors':
##
##     intersect
## The following object is masked from 'package:BiocGenerics':
##
##     intersect
```

```
## The following objects are masked from 'package:base':
##
##     intersect, t
##
## Attaching package: 'Seurat'
## The following object is masked from 'package:SummarizedExperiment':
##
##     Assays
library(ggpmisc)
## Loading required package: ggpp
## Registered S3 methods overwritten by 'ggpp':
##   method                 from
##   heightDetails.titleGrob ggplot2
##   widthDetails.titleGrob  ggplot2
##
## Attaching package: 'ggpp'
## The following objects are masked from 'package:ggpubr':
##
##     as_npc, as_npcx, as_npcy
## The following object is masked from 'package:ggplot2':
##
##     annotate
library(stringr)
library(ggplot2)
library(circlize)
## ========================================
## circlize version 0.4.16
## CRAN page: https://cran.r-project.org/package=circlize
## Github page: https://github.com/jokergoo/circlize
## Documentation: https://jokergoo.github.io/circlize_book/book/
##
## If you use it in published research, please cite:
## Gu, Z. circlize implements and enhances circular visualization
##   in R. Bioinformatics 2014.
##
## This message can be suppressed by:
##   suppressPackageStartupMessages(library(circlize))
## ========================================
library(pheatmap)
library(corrplot)
## corrplot 0.95 loaded
library(bedtoolsr)
## Warning in fun(libname, pkgname): bedtoolsr was built with bedtools version 2.30.0 but you have vers
## options(bedtools.path = "[bedtools path]")
library(tidyverse)
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v lubridate 1.9.4     v tibble    3.3.0
## v purrr     1.1.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::%within%()     masks IRanges::%within%()
## x ggpp::annotate()          masks ggplot2::annotate()
## x dplyr::arrange()          masks plyr::arrange()
```

```
## x dplyr::between()          masks data.table::between()
## x dplyr::collapse()         masks IRanges::collapse()
## x dplyr::combine()          masks Biobase::combine(), BiocGenerics::combine(), gridExtra::combine()
## x purrr::compact()          masks plyr::compact()
## x dplyr::count()            masks matrixStats::count(), plyr::count()
## x dplyr::desc()             masks IRanges::desc(), plyr::desc()
## x tidyr::expand()           masks Matrix::expand(), S4Vectors::expand()
## x tidyr::extract()          masks magrittr::extract()
## x dplyr::failwith()         masks plyr::failwith()
## x dplyr::filter()           masks stats::filter()
## x dplyr::first()            masks S4Vectors::first(), data.table::first()
## x lubridate::hour()         masks data.table::hour()
## x dplyr::id()               masks plyr::id()
## x lubridate::isoweek()      masks data.table::isoweek()
## x dplyr::lag()              masks stats::lag()
## x dplyr::last()             masks data.table::last()
## x lubridate::mday()         masks data.table::mday()
## x lubridate::minute()       masks data.table::minute()
## x lubridate::month()        masks data.table::month()
## x ggpubr::mutate()          masks dplyr::mutate(), plyr::mutate()
## x tidyr::pack()             masks Matrix::pack()
## x BiocGenerics::Position()  masks ggplot2::Position(), base::Position()
## x lubridate::quarter()      masks data.table::quarter()
## x purrr::reduce()           masks GenomicRanges::reduce(), IRanges::reduce()
## x dplyr::rename()           masks S4Vectors::rename(), plyr::rename()
## x lubridate::second()       masks S4Vectors::second(), data.table::second()
## x lubridate::second<-()     masks S4Vectors::second<-()
## x AnnotationDbi::select()   masks dplyr::select()
## x purrr::set_names()        masks magrittr::set_names()
## x dplyr::slice()            masks IRanges::slice()
## x GenomicRanges::subtract() masks magrittr::subtract()
## x dplyr::summarise()        masks plyr::summarise()
## x dplyr::summarize()        masks plyr::summarize()
## x purrr::transpose()        masks data.table::transpose()
## x tidyr::unpack()           masks Matrix::unpack()
## x lubridate::wday()         masks data.table::wday()
## x lubridate::week()         masks data.table::week()
## x lubridate::yday()         masks data.table::yday()
## x lubridate::year()         masks data.table::year()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
library(patchwork)
library(gridExtra)
library(data.table)
library(UpSetR)
library(ComplexUpset)
##
## Attaching package: 'ComplexUpset'
##
## The following object is masked from 'package:UpSetR':
##
##     upset
library(ggsankey)
library(RColorBrewer)
```

```r
library(org.Hs.eg.db)
##
library(GenomicRanges)
library(ComplexHeatmap)
## ========================================
## ComplexHeatmap version 2.24.1
## Bioconductor page: http://bioconductor.org/packages/ComplexHeatmap/
## Github page: https://github.com/jokergoo/ComplexHeatmap
## Documentation: http://jokergoo.github.io/ComplexHeatmap-reference
##
## If you use it in published research, please cite either one:
## - Gu, Z. Complex Heatmap Visualization. iMeta 2022.
## - Gu, Z. Complex heatmaps reveal patterns and correlations in multidimensional
##     genomic data. Bioinformatics 2016.
##
##
## The new InteractiveComplexHeatmap package can directly export static
## complex heatmaps into an interactive Shiny app with zero effort. Have a try!
##
## This message can be suppressed by:
##   suppressPackageStartupMessages(library(ComplexHeatmap))
## ========================================
## ! pheatmap() has been masked by ComplexHeatmap::pheatmap(). Most of the arguments
##    in the original pheatmap() are identically supported in the new function. You
##    can still use the original function by explicitly calling pheatmap::pheatmap().
##
##
## Attaching package: 'ComplexHeatmap'
##
## The following object is masked from 'package:pheatmap':
##
##     pheatmap
library(clusterProfiler)
## clusterProfiler v4.16.0 Learn more at https://yulab-smu.top/contribution-knowledge-mining/
##
## Please cite:
##
## G Yu. Thirteen years of clusterProfiler. The Innovation. 2024,
## 5(6):100722
##
## Attaching package: 'clusterProfiler'
##
## The following object is masked from 'package:purrr':
##
##     simplify
##
## The following object is masked from 'package:AnnotationDbi':
##
##     select
##
## The following object is masked from 'package:IRanges':
##
##     slice
```

```
##
## The following object is masked from 'package:S4Vectors':
##
##     rename
##
## The following objects are masked from 'package:plyr':
##
##     arrange, mutate, rename, summarise
##
## The following object is masked from 'package:stats':
##
##     filter
library(networkD3)
##
## Attaching package: 'networkD3'
##
## The following object is masked from 'package:Seurat':
##
##     JS
##
## The following object is masked from 'package:SeuratObject':
##
##     JS
library(htmlwidgets)
##
## Attaching package: 'htmlwidgets'
##
## The following object is masked from 'package:networkD3':
##
##     JS
##
## The following object is masked from 'package:Seurat':
##
##     JS
##
## The following object is masked from 'package:SeuratObject':
##
##     JS
```

3 define functions

```
cosine_similarity <- function(a, b, na.rm = TRUE) {
  a <- as.numeric(a)
  b <- as.numeric(b)

  if (na.rm) {
    keep <- !(is.na(a) | is.na(b))
    a <- a[keep]
    b <- b[keep]
  }
```

```
  denom <- sqrt(sum(a^2)) * sqrt(sum(b^2))
  if (denom == 0) return(as.numeric(NA))

  sim <- sum(a * b) / denom

  sim <- max(min(sim, 1), -1)

  return(as.numeric(sim))
}

retriveCellsByLocation <- function(CSM, locList){
  CSM$CellTag <- rownames(CSM)
  CSM_tmp <- CSM[, c("CellTag",locList)]
  data_MutPerCell_CSM_tmp <- data.frame(
    CellTag = rownames(CSM_tmp),
    MutNum = rowSums(CSM_tmp == "Mut"),
    WTNum = rowSums(CSM_tmp == "WT")
  )
  cells_Total <- data_MutPerCell_CSM_tmp[rowSums(data_MutPerCell_CSM_tmp[,c("MutNum", "WTNum")]) > 0, ]$
  cells_WT <-    data_MutPerCell_CSM_tmp[data_MutPerCell_CSM_tmp[,"WTNum"] > 0,]$CellTag
  cells_Mut <-   data_MutPerCell_CSM_tmp[data_MutPerCell_CSM_tmp[,"MutNum"] > 0,]$CellTag
  cells <- list(Total = cells_Total, Mut = cells_Mut, WT = cells_WT)
  return(cells)
}

retriveLocsByCells <- function(CSM, cellList){
  CSM$CellTag <- rownames(CSM)
  CSM_tmp <- CSM[CSM$CellTag %in% cellList,]
  data_CellPerMut_CSM_tmp <- data.frame(
    Location = colnames(CSM_tmp),
    CellMutNum = colSums(CSM_tmp == "Mut"),
    CellWTNum = colSums(CSM_tmp == "WT")
  )
  pos_Total <- data_CellPerMut_CSM_tmp[rowSums(data_CellPerMut_CSM_tmp[, c("CellMutNum", "CellWTNum")])
  pos_Mut <- data_CellPerMut_CSM_tmp[data_CellPerMut_CSM_tmp[, "CellMutNum"] > 0 ,]$Location
  pos_WT <- data_CellPerMut_CSM_tmp[data_CellPerMut_CSM_tmp[, "CellWTNum"] > 0 ,]$Location
  pos <- list(Total = pos_Total, Mut = pos_Mut, WT = pos_WT)
  return(pos)
}

get_mut_cols <- function(df, loc) {
  row <- df[df$Location == loc, ]
  mut_cols <- names(row)[row == "Mut"]
  mut_cols[mut_cols != "Location"]
}
```

AML_census.rds and *AML_tiergenes.txt* were downloaded from the COSMIC database and curated for downstream analyses

AML_SSM.rds and *AML_ATA_CSM_list.rds*, *AML_RNA_CSM_list.rds* were generated using scMutrace as described in the Methods section and `0_pipelineFunctions.Rmd`(pipelines for generating single-cell somatic mutation data list).

AML_scRNA_meta.rds and *AML_scATA_meta.rds* contain meta data for cell type annotation similarity analysis and were generate from `0_pipelineFunctions.Rmd`(pipelines for porcessing scRNA-seq data and

pipelines for processing scATAC-seq data).

sampleInfo.txt contains sample information including cell numbers and patient clinical data for scRNA-seq and scATAC-seq data and was downloaded from original paper.

EGA_scATA.hg38_multianno.txt and *EGA_scRNA.hg38_multianno.txt* contain mutation annotation information generated using ANNOVAR.

Mutational_Matrix_SBS.txt and *COSMIC_SBS96_Activities.txt* were generated using SigProfiler suite as described in the Methods section.

4 load data

```
# load AML census data
AML_census <- readRDS("../Data/HumanpAML/AML_census.rds")
AML_genes_tiers <- read.table("../Data/HumanpAML/AML_tiergenes.txt", header = T, sep = "\t")

# load SSM data
AML_SSM <- readRDS("../Data/HumanpAML/AML_SSM.rds")
AML_SSM_scRNA <- AML_SSM$scRNA
AML_SSM_scATA <- AML_SSM$scATA

# CSM
AML_ATA_CSM_list <- readRDS("../Data/HumanpAML/AML_ATA_CSM_list.rds")
AML_RNA_CSM_list <- readRDS("../Data/HumanpAML/AML_RNA_CSM_list.rds")

# meta data for celltype annotation similarity
AML_scRNA_meta_list <- readRDS("../Data/HumanpAML/AML_scRNA_meta.rds")
# metaQun
AML_scATA_meta_list <- readRDS("../Data/HumanpAML/AML_scATA_meta.rds")

AML_scRNA_meta <- do.call(rbind, AML_scRNA_meta_list)
rownames(AML_scRNA_meta) <- sub("\\.", "#", rownames(AML_scRNA_meta))

AML_scATA_meta <- do.call(rbind, AML_scATA_meta_list)
AML_scATA_meta$SampleID <- sub("_.*", "", AML_scATA_meta$Sample)

rownames(AML_scATA_meta) <- sub("^AML[0-9]+\\.", "", rownames(AML_scATA_meta))

cellTypes <- unique(union(AML_scRNA_meta$Classified_Celltype, AML_scATA_meta$predictedGroup_Un))

# extract mutations and get bed format with census tag
AML_SSM_scRNA_bed <- separate(AML_SSM_scRNA, Location, into = c("chr", "pos", "ref", "alt"), sep = "_",
AML_SSM_scRNA_bed$end <- AML_SSM_scRNA_bed$pos
AML_SSM_scRNA_bed <- AML_SSM_scRNA_bed[,c("chr", "pos", "end", "ref", "alt", "Location")]
AML_SSM_scRNA_bed$Tag <- ifelse(AML_SSM_scRNA_bed$Location %in% AML_census$mutation$Location, "Census",
AML_scRNA_NotCensus_locs <- AML_SSM_scRNA_bed %>% dplyr::filter(Tag == "NotCensus") %>% pull(Location)

AML_SSM_scATA_bed <- separate(AML_SSM_scATA, Location, into = c("chr", "pos", "ref", "alt"), sep = "_",
AML_SSM_scATA_bed$end <- AML_SSM_scATA_bed$pos
AML_SSM_scATA_bed <- AML_SSM_scATA_bed[,c("chr", "pos", "end", "ref", "alt", "Location")]
AML_SSM_scATA_bed$Tag <- ifelse(AML_SSM_scATA_bed$Location %in% AML_census$mutation$Location, "Census",
AML_scATA_NotCensus_locs <- AML_SSM_scATA_bed %>% dplyr::filter(Tag == "NotCensus") %>% pull(Location)
```

```r
# load paired sample info data
AML_sampleInfo <- read.table("../Data/HumanpAML/sampleInfo.txt", header = T, sep = "\t")
AML_sampleInfo[is.na(AML_sampleInfo)] <- 0

# AML ann
AML_scATA_ann <- read.table("../Data/HumanpAML/EGA_scATA.hg38_multianno.txt", header = T, sep = "\t")
AML_scRNA_ann <- read.table("../Data/HumanpAML/EGA_scRNA.hg38_multianno.txt", header = T, sep = "\t")

col_freq_databases <- c("gnomad40_exome_AF", "gnomad40_exome_AF_raw", "gnomad40_exome_AF_XX",
                        "gnomad40_exome_AF_XY", "gnomad40_exome_AF_grpmax", "gnomad40_exome_faf95",
                        "gnomad40_exome_faf99", "gnomad40_exome_fafmax_faf95_max", "gnomad40_exome_fafma
                        "gnomad40_exome_AF_afr", "gnomad40_exome_AF_amr", "gnomad40_exome_AF_asj",
                        "gnomad40_exome_AF_eas", "gnomad40_exome_AF_fin", "gnomad40_exome_AF_mid",
                        "gnomad40_exome_AF_nfe", "gnomad40_exome_AF_remaining", "gnomad40_exome_AF_sas"
                        "gnomad40_genome_AF", "gnomad40_genome_AF_raw", "gnomad40_genome_AF_XX",
                        "gnomad40_genome_AF_XY", "gnomad40_genome_AF_grpmax", "gnomad40_genome_faf95",
                        "gnomad40_genome_faf99", "gnomad40_genome_fafmax_faf95_max",
                        "gnomad40_genome_fafmax_faf99_max", "gnomad40_genome_AF_afr",
                        "gnomad40_genome_AF_ami", "gnomad40_genome_AF_amr",
                        "gnomad40_genome_AF_asj", "gnomad40_genome_AF_eas",
                        "gnomad40_genome_AF_fin", "gnomad40_genome_AF_mid",
                        "gnomad40_genome_AF_nfe", "gnomad40_genome_AF_remaining", "gnomad40_genome_AF_sa

col_freq_databases %in% colnames(AML_scATA_ann)
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [31] TRUE TRUE TRUE TRUE TRUE TRUE TRUE
col_freq_databases %in% colnames(AML_scRNA_ann)
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [31] TRUE TRUE TRUE TRUE TRUE TRUE TRUE

AML_scATA_ann <- AML_scATA_ann %>%
  mutate(across(all_of(col_freq_databases),
                ~ as.numeric(replace(., . == ".", 0))))

AML_scRNA_ann <- AML_scRNA_ann %>%
  mutate(across(all_of(col_freq_databases),
                ~ as.numeric(replace(., . == ".", 0))))

# Annotate bed census and nonCensus
# scATA
AML_SSM_scATA_bed <- AML_SSM_scATA_bed %>%
  left_join(
    AML_scATA_ann %>%
      dplyr::select(Location, all_of(col_freq_databases)),
    by = "Location"
  ) %>%
  mutate(
    freq = if_else(
      if_any(all_of(col_freq_databases), ~ . > 0.0001),
      "high", "rare"
    )
```

```
  ) %>%
  dplyr::select(names(AML_SSM_scATA_bed), freq)

# scRNA
AML_SSM_scRNA_bed <- AML_SSM_scRNA_bed %>%
  left_join(
    AML_scRNA_ann %>%
      dplyr::select(Location, all_of(col_freq_databases)),
    by = "Location"
  ) %>%
  mutate(
    freq = if_else(
      if_any(all_of(col_freq_databases), ~ . > 0.0001),
      "high", "rare"
    )
  ) %>%
  dplyr::select(names(AML_SSM_scRNA_bed), freq)

# load mutation features data
# generate using sigProfiler suite
AML_mutation_features <- read.table("../Data/HumanpAML/Mutational_Matrix_SBS.txt", header = T, sep = "\t
AML_mutation_features_contribution <- read.table("../Data/HumanpAML/COSMIC_SBS96_Activities.txt", header
```

5 mutation Similarity DXREL for lineage switching (downsample)

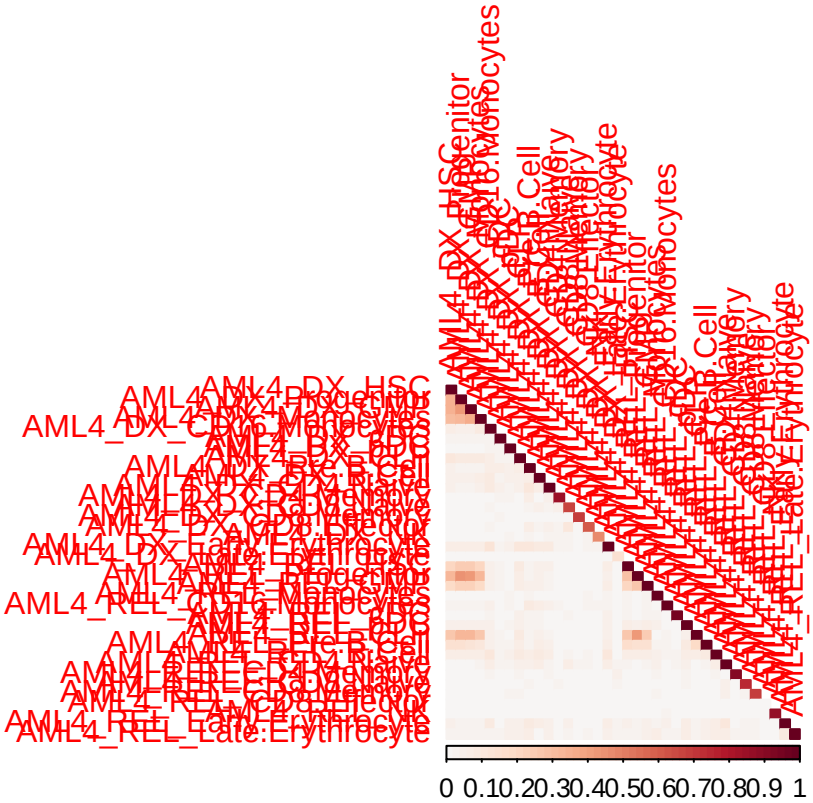
```
AML_scRNA_CSM_list <- AML_RNA_CSM_list
AML_scATA_CSM_list <- AML_ATA_CSM_list

# scRNA
# AML_CSM_DX_list
AML_scRNA_CSM_DX_list <- lapply(AML_scRNA_CSM_list, function(df) {
  df[grep("DX", rownames(df)), , drop = FALSE]
})
AML_scRNA_CSM_DX_list <- Filter(function(x) nrow(x) > 0, AML_scRNA_CSM_DX_list)

# AML_CSM_REM_list
AML_scRNA_CSM_REM_list <- lapply(AML_scRNA_CSM_list, function(df) {
  df[grep("REM", rownames(df)), , drop = FALSE]
})
AML_scRNA_CSM_REM_list <- Filter(function(x) nrow(x) > 0, AML_scRNA_CSM_REM_list)

# AML_CSM_REL_list
AML_scRNA_CSM_REL_list <- lapply(AML_scRNA_CSM_list, function(df) {
  df[grep("REL", rownames(df)), , drop = FALSE]
})
AML_scRNA_CSM_REL_list <- Filter(function(x) nrow(x) > 0, AML_scRNA_CSM_REL_list)

# scATA
# AML_CSM_DX_list
AML_scATA_CSM_DX_list <- lapply(AML_scATA_CSM_list, function(df) {
  df[grep("DX", rownames(df)), , drop = FALSE]
```

```r
})
AML_scATA_CSM_DX_list <- Filter(function(x) nrow(x) > 0, AML_scATA_CSM_DX_list)

# AML_CSM_REM_list
AML_scATA_CSM_REM_list <- lapply(AML_scATA_CSM_list, function(df) {
  df[grep("REM", rownames(df)), , drop = FALSE]
})
AML_scATA_CSM_REM_list <- Filter(function(x) nrow(x) > 0, AML_scATA_CSM_REM_list)

# AML_CSM_REL_list
AML_scATA_CSM_REL_list <- lapply(AML_scATA_CSM_list, function(df) {
  df[grep("REL", rownames(df)), , drop = FALSE]
})
AML_scATA_CSM_REL_list <- Filter(function(x) nrow(x) > 0, AML_scATA_CSM_REL_list)

HSPC <- c("HSC", "Progenitor")
Myeloid <- c("GMP", "Monocytes", "CD16.Monocytes", "cDC", "pDC")
Lymphoid <- c("CLP", "Pre.B.Cell", "B.Cell", "CD4.Naive", "CD4.Memory", "CD8.Naive", "CD8.Memory", "CD8
Erythroid_Basophil <- c("Early.Erythrocyte", "Late.Erythrocyte")
Other <- c("Early.Basophil", "Plasma")

AML_scRNA_meta <- AML_scRNA_meta %>%
  dplyr::mutate(
    maincelltype = case_when(
      Classified_Celltype %in% HSPC ~ "HSPC",
      Classified_Celltype %in% Myeloid ~ "Myeloid",
      Classified_Celltype %in% Lymphoid ~ "Lymphoid",
      Classified_Celltype %in% Erythroid_Basophil ~ "Erythroid",
      Classified_Celltype %in% Other ~ "Other",
      TRUE ~ NA_character_
    )
  )

AML_scATA_meta <- AML_scATA_meta %>%
  dplyr::mutate(
    maincelltype = case_when(
      predictedGroup_Un %in% HSPC ~ "HSPC",
      predictedGroup_Un %in% Myeloid ~ "Myeloid",
      predictedGroup_Un %in% Lymphoid ~ "Lymphoid",
      predictedGroup_Un %in% Erythroid_Basophil ~ "Erythroid",
      predictedGroup_Un %in% Other ~ "Other",
      TRUE ~ NA_character_
    )
  )

AML_scATA_meta$Classified_Celltype <- AML_scATA_meta$predictedGroup_Un

maincelltypes <- c("HSPC", "Myeloid", "Lymphoid", "Erythroid", "Other")
subcelltypes <- c(HSPC, Myeloid, Lymphoid, Erythroid_Basophil, Other)

AML_scRNA_meta_DX <- AML_scRNA_meta %>%
  dplyr::filter(grepl("DX", Library_ID))
```

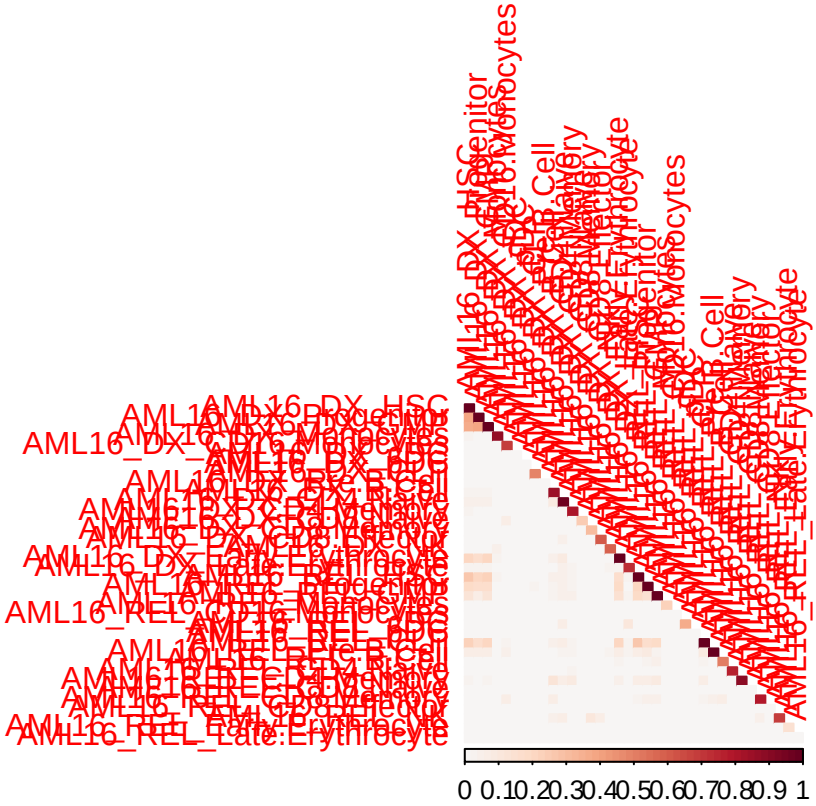
```
AML_scRNA_meta_REL <- AML_scRNA_meta %>%
  dplyr::filter(grepl("REL", Library_ID))

AML_scATA_meta_DX <- AML_scATA_meta %>%
  dplyr::filter(grepl("DX", Sample))

AML_scATA_meta_REL <- AML_scATA_meta %>%
  dplyr::filter(grepl("REL", Sample))

selected_samples_id <- c("AML4", "AML5", "AML6", "AML8",  "AML10", "AML12", "AML16", "AML18", "AML21")
selected_samples_id <- c("AML2", "AML4", "AML5", "AML6", "AML8", "AML10", "AML11", "AML12",
                         "AML13", "AML15", "AML16", "AML17", "AML18", "AML19", "AML21",
                         "AML22", "AML24", "AML25", "AML26", "AML27")

AML_scRNA_CSM_DX_select_list <- AML_scRNA_CSM_DX_list[names(AML_scRNA_CSM_DX_list) %in% selected_samples
names(AML_scRNA_CSM_DX_select_list) <- paste(selected_samples_id, "DX", sep = "_")
AML_scRNA_CSM_REL_select_list <- AML_scRNA_CSM_REL_list[names(AML_scRNA_CSM_REL_list) %in% selected_samp
names(AML_scRNA_CSM_REL_select_list) <- paste(selected_samples_id, "REL", sep = "_")
AML_scRNA_CSM_DX_REL_select_list <- c(AML_scRNA_CSM_DX_select_list, AML_scRNA_CSM_REL_select_list)

AML_scRNA_meta_DX$Classified_CelltypeQ <- paste(AML_scRNA_meta_DX$Lambo_et_al_ID, "DX", AML_scRNA_meta_
AML_scRNA_meta_REL$Classified_CelltypeQ <- paste(AML_scRNA_meta_REL$Lambo_et_al_ID, "REL", AML_scRNA_me
AML_scRNA_meta_DX$maincelltypeQ <- paste(AML_scRNA_meta_DX$Lambo_et_al_ID, "DX", AML_scRNA_meta_DX$main
AML_scRNA_meta_REL$maincelltypeQ <- paste(AML_scRNA_meta_REL$Lambo_et_al_ID, "REL", AML_scRNA_meta_REL$

AML_scATA_CSM_DX_select_list <- AML_scATA_CSM_DX_list[names(AML_scATA_CSM_DX_list) %in% selected_samples
names(AML_scATA_CSM_DX_select_list) <- paste(selected_samples_id, "DX", sep = "_")
AML_scATA_CSM_REL_select_list <- AML_scATA_CSM_REL_list[names(AML_scATA_CSM_REL_list) %in% selected_samp
names(AML_scATA_CSM_REL_select_list) <- paste(selected_samples_id, "REL", sep = "_")
AML_scATA_CSM_DX_REL_select_list <- c(AML_scATA_CSM_DX_select_list, AML_scATA_CSM_REL_select_list)
AML_scATA_meta_DX$Classified_CelltypeQ <- paste(AML_scATA_meta_DX$SampleID, "DX", AML_scATA_meta_DX$Cla
AML_scATA_meta_REL$Classified_CelltypeQ <- paste(AML_scATA_meta_REL$SampleID, "REL", AML_scATA_meta_REL

AML_scRNA_meta_DX_REL <- rbind(AML_scRNA_meta_DX, AML_scRNA_meta_REL)
AML_scATA_meta_DX_REL <- rbind(AML_scATA_meta_DX, AML_scATA_meta_REL)

# samples for lineage switching

AML_scRNA_meta_DX_REL <- AML_scRNA_meta_DX_REL[AML_scRNA_meta_DX_REL$Lambo_et_al_ID %in% selected_sample
AML_scRNA_meta_DX_REL$CellTag <- rownames(AML_scRNA_meta_DX_REL) %>%
                                  sub("-1$", "", .) %>%
                                  gsub("#", "_", .)
AML_scRNA_meta$CellTag <- rownames(AML_scRNA_meta) %>%
                                  sub("-1$", "", .) %>%
                                  gsub("#", "_", .)

AML_scRNA_meta_DX_REL$SampleID <- AML_scRNA_meta_DX_REL$Lambo_et_al_ID
AML_scATA_meta_DX_REL <- AML_scATA_meta_DX_REL[AML_scATA_meta_DX_REL$SampleID %in% selected_samples_id,
AML_scATA_meta_DX_REL$CellTag <- rownames(AML_scATA_meta_DX_REL) %>%
                                  sub("-1$", "", .) %>%
                                  gsub("#", "_", .)
AML_scATA_meta$CellTag <- rownames(AML_scATA_meta) %>%
                                  sub("-1$", "", .) %>%
```

```
                                    gsub("#", "_", .)

AML_scATA_meta_DX_REL$maincelltypeQ <- paste(AML_scATA_meta_DX_REL$Sample, AML_scATA_meta_DX_REL$maincel

cells_limits <- 0

celltypes_scRNA <- names(table(AML_scRNA_meta_DX_REL$Classified_CelltypeQ)[table(AML_scRNA_meta_DX_REL$C
celltypes_scATA <- names(table(AML_scATA_meta_DX_REL$Classified_CelltypeQ)[table(AML_scATA_meta_DX_REL$C

celltypes_scRNA_main <- names(table(AML_scRNA_meta_DX_REL$maincelltypeQ)[table(AML_scRNA_meta_DX_REL$mai
celltypes_scATA_main <- names(table(AML_scATA_meta_DX_REL$maincelltypeQ)[table(AML_scATA_meta_DX_REL$mai
```

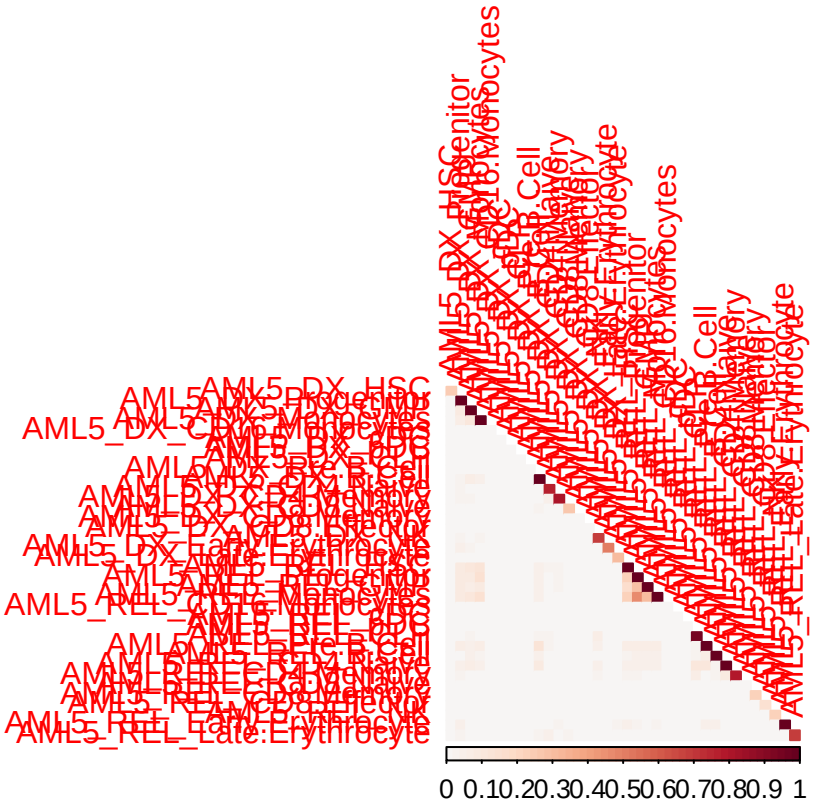
6 calSimilarity downsampling



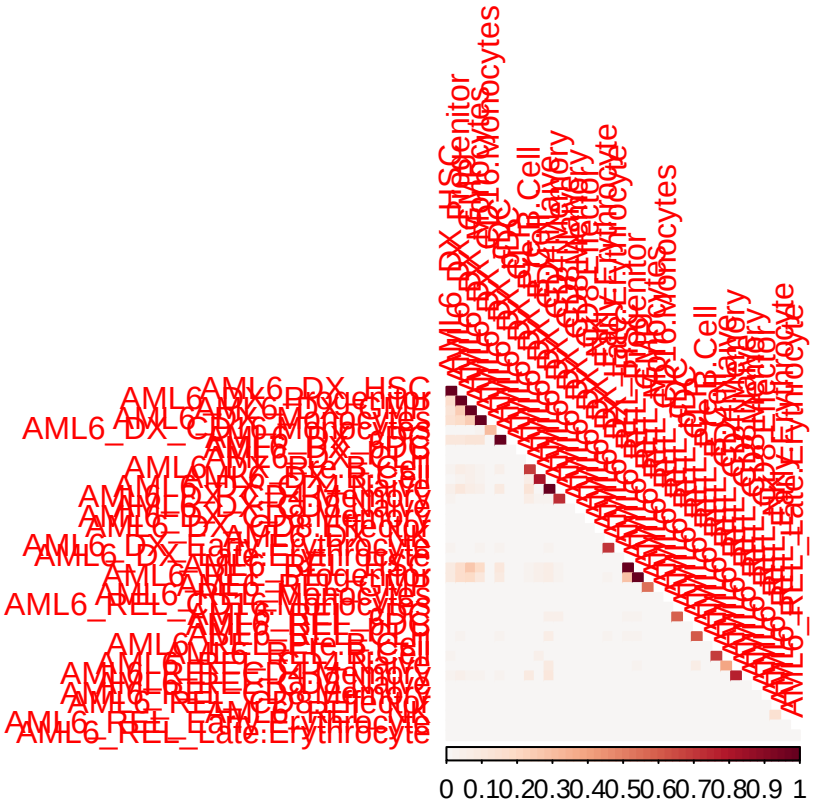

```
corrplot(mat_sorted_AML16_RNA, method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



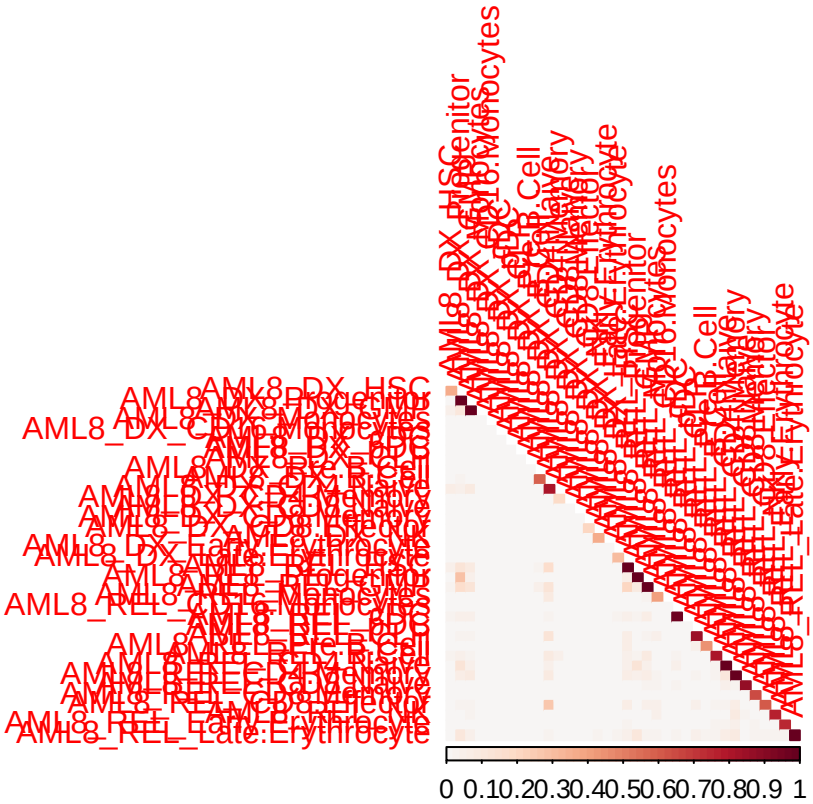

```
corrplot(mat_sorted_AML05_RNA, method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



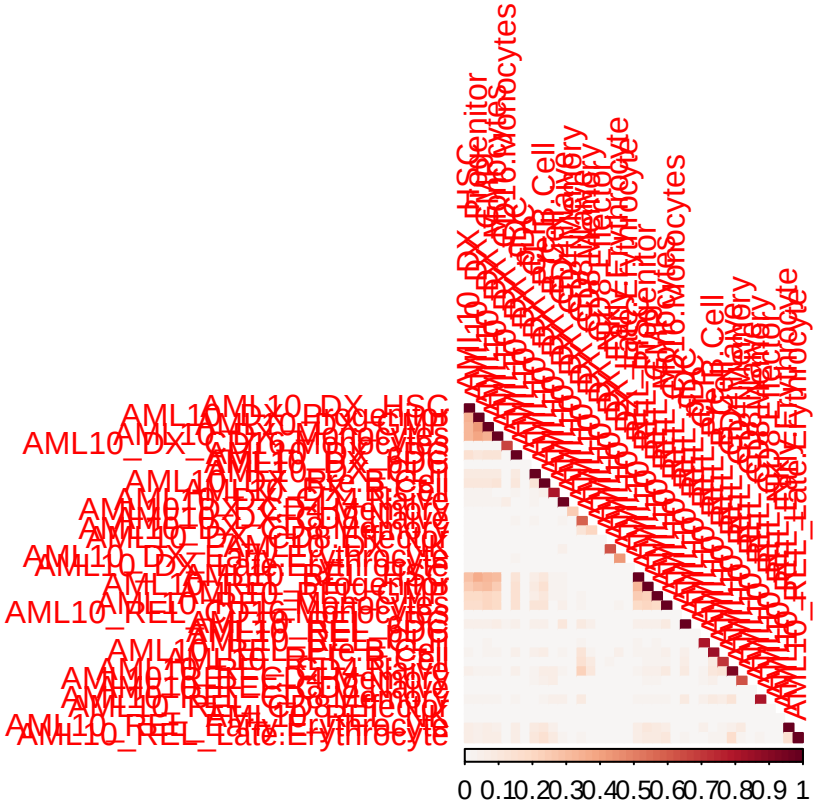

```
corrplot(mat_sorted_AML06_RNA, method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



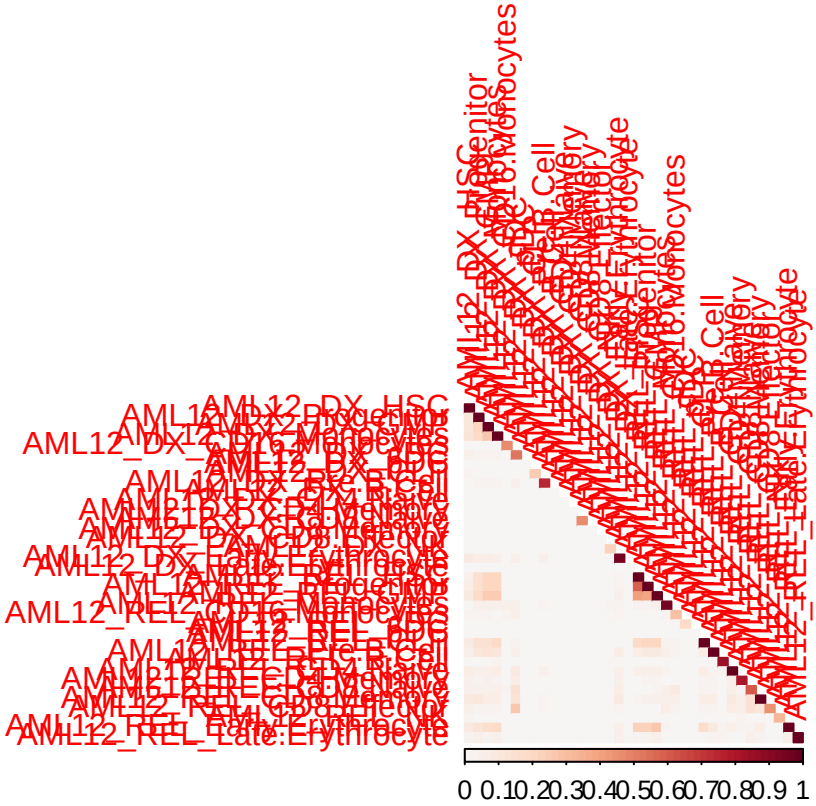

```
corrplot(mat_sorted_AML08_RNA, method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



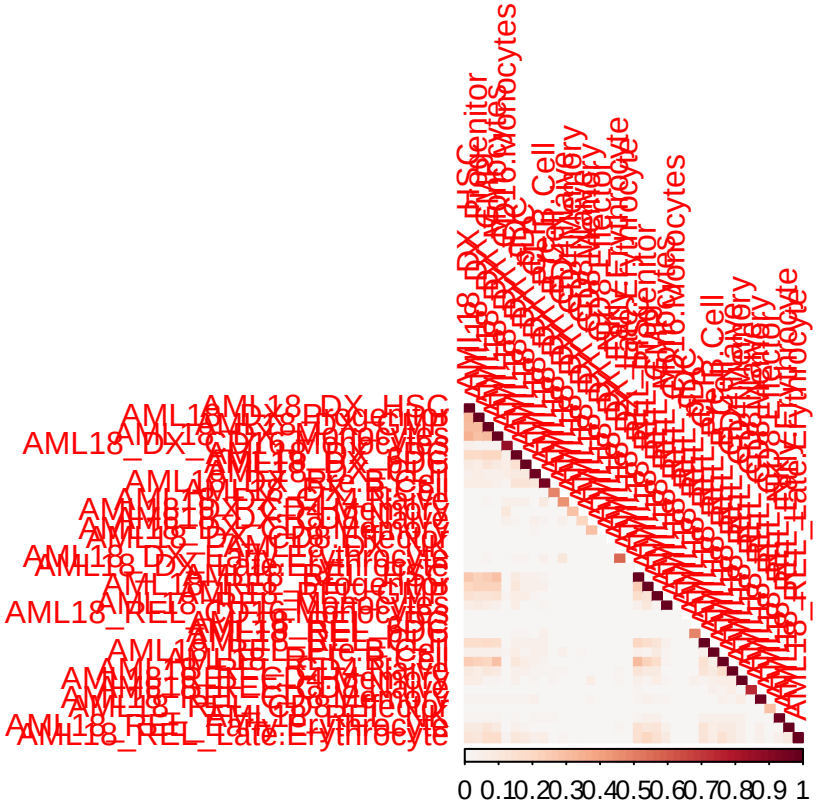

```
corrplot(mat_sorted_AML10_RNA, method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



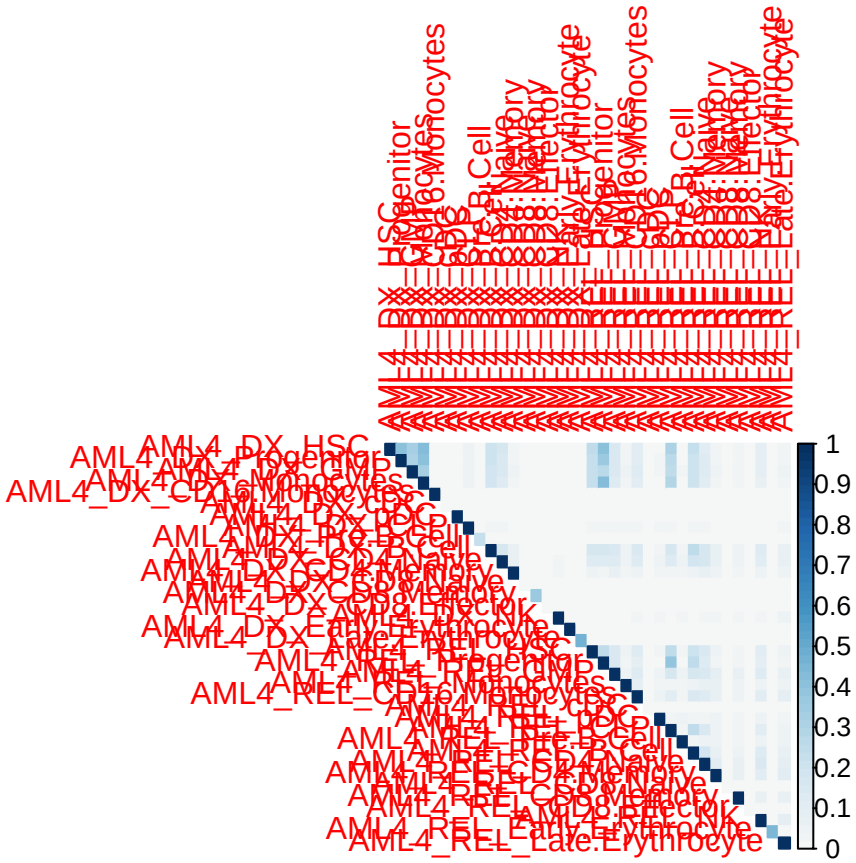

```
corrplot(mat_sorted_AML12_RNA, method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



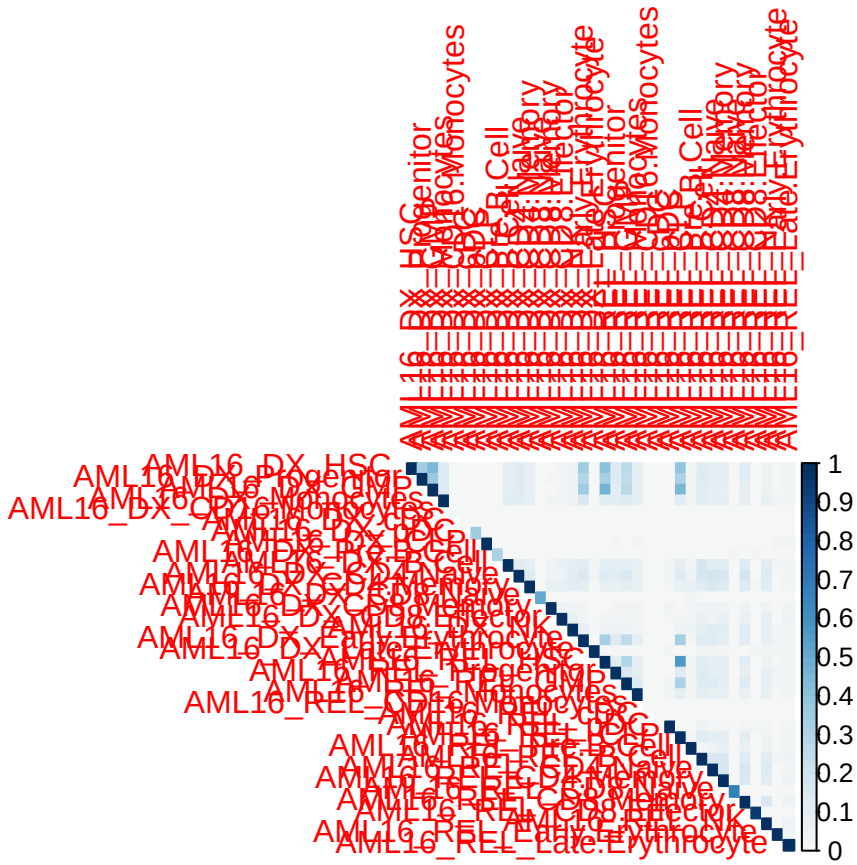

```
corrplot(mat_sorted_AML18_RNA, method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



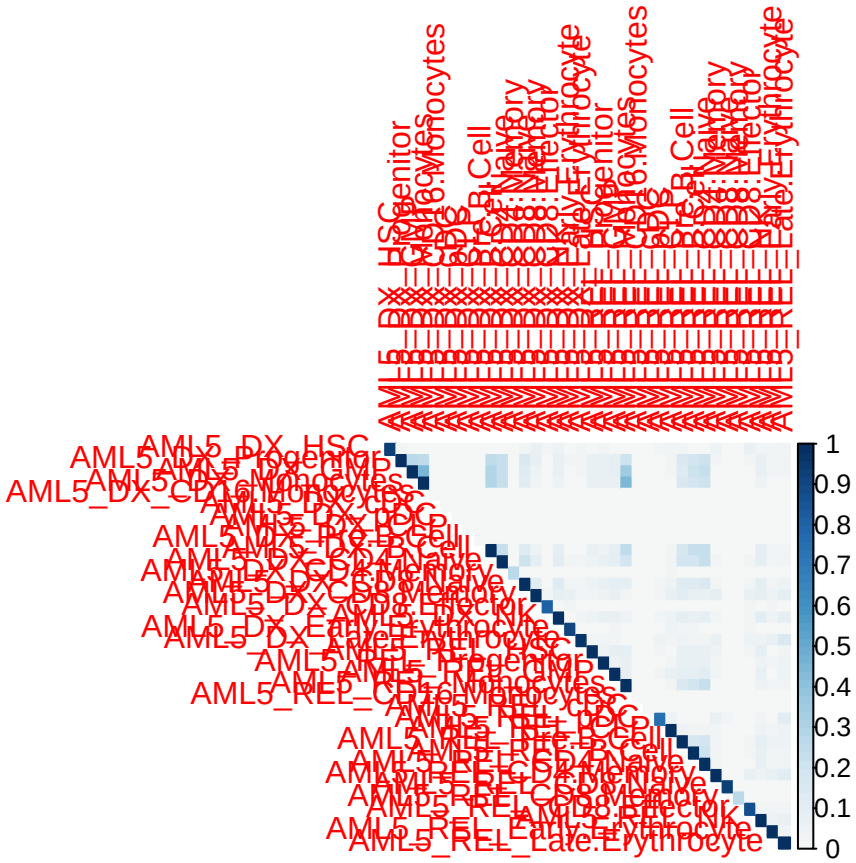

```
# ATA
corrplot(mat_sorted_AML04_ATA, method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



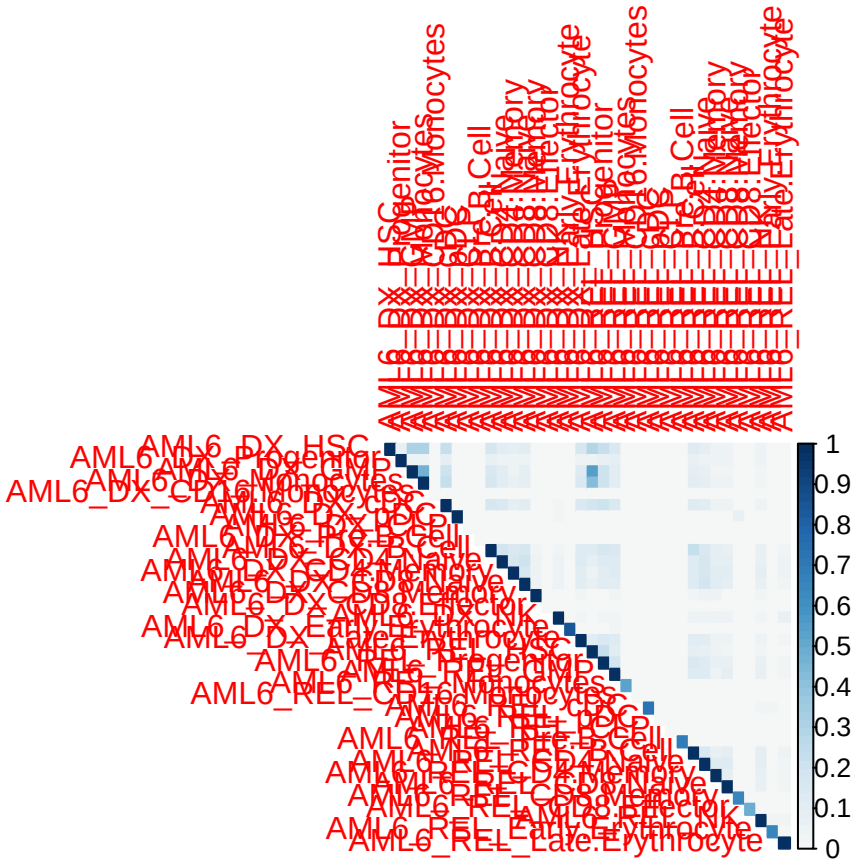

```
corrplot(mat_sorted_AML16_ATA, method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



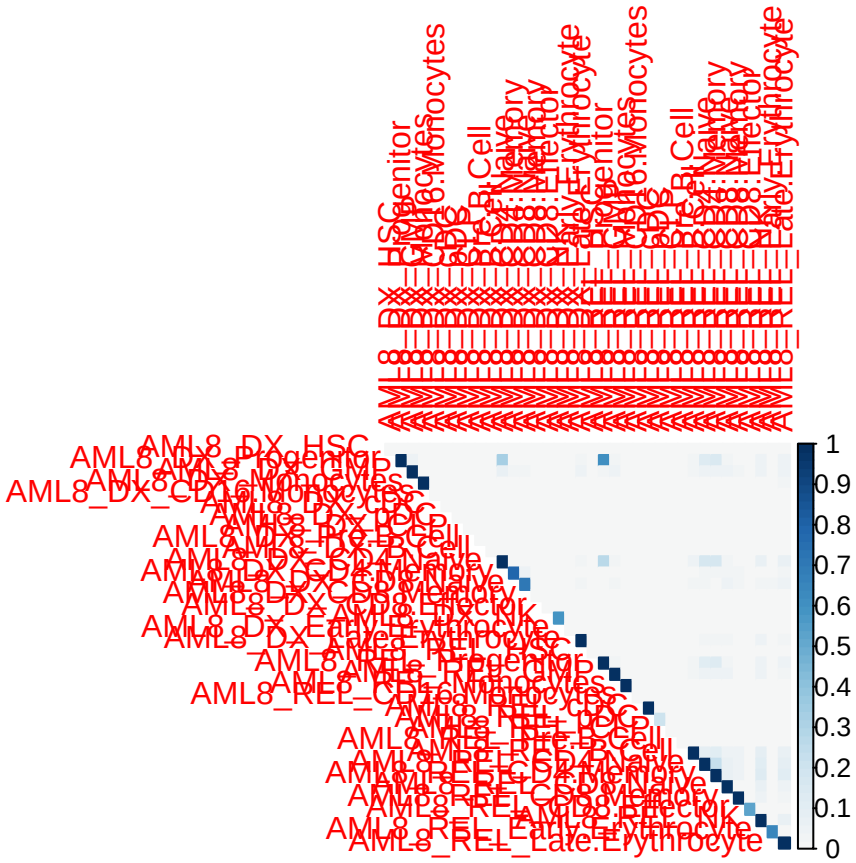

```
corrplot(mat_sorted_AML05_ATA, method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



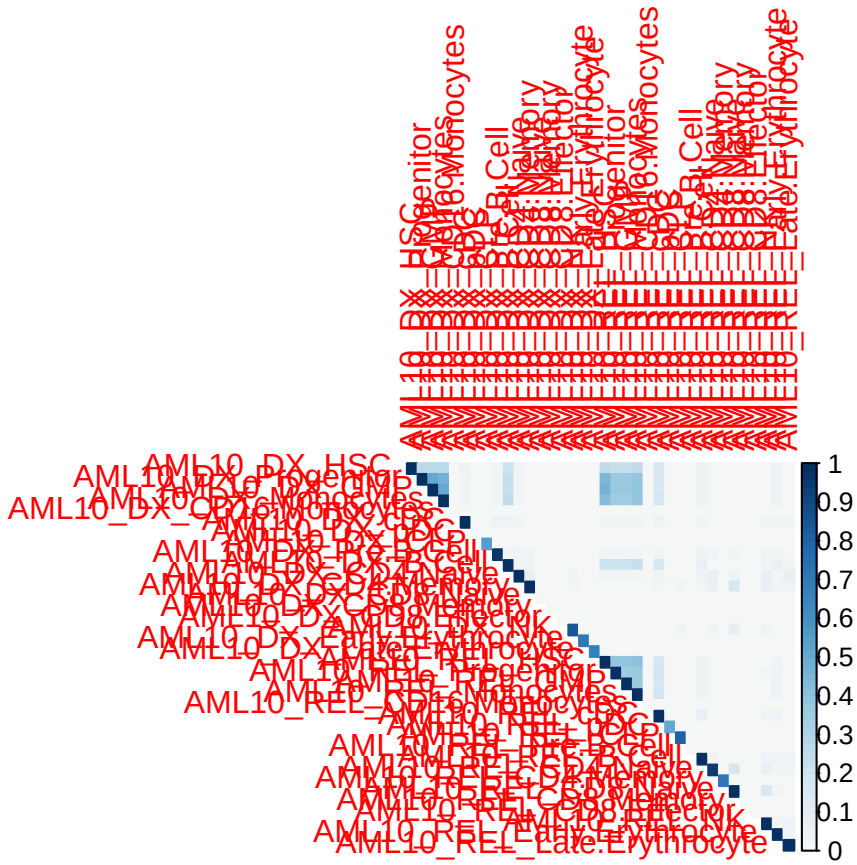

```
corrplot(mat_sorted_AML06_ATA, method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



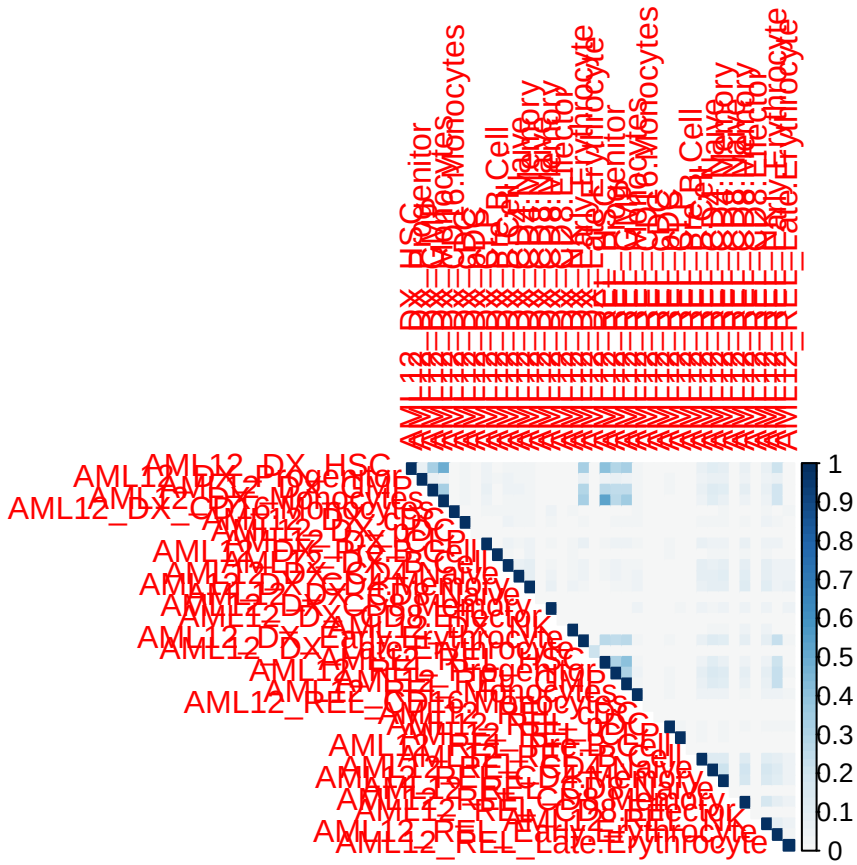

```
corrplot(mat_sorted_AML08_ATA, method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



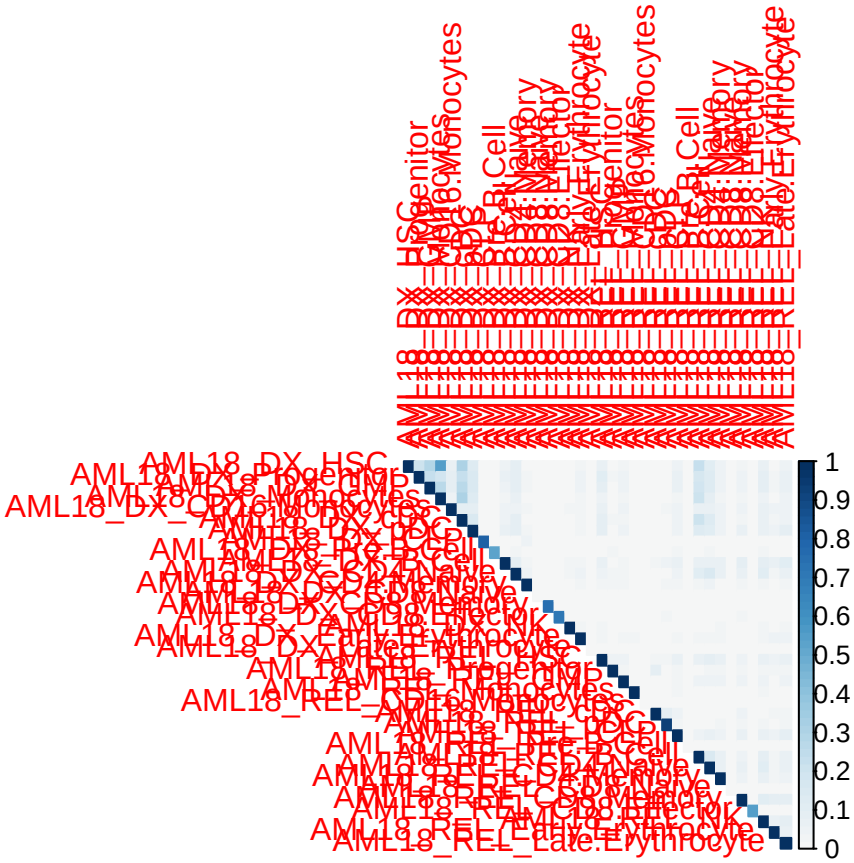

```
corrplot(mat_sorted_AML10_ATA, method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



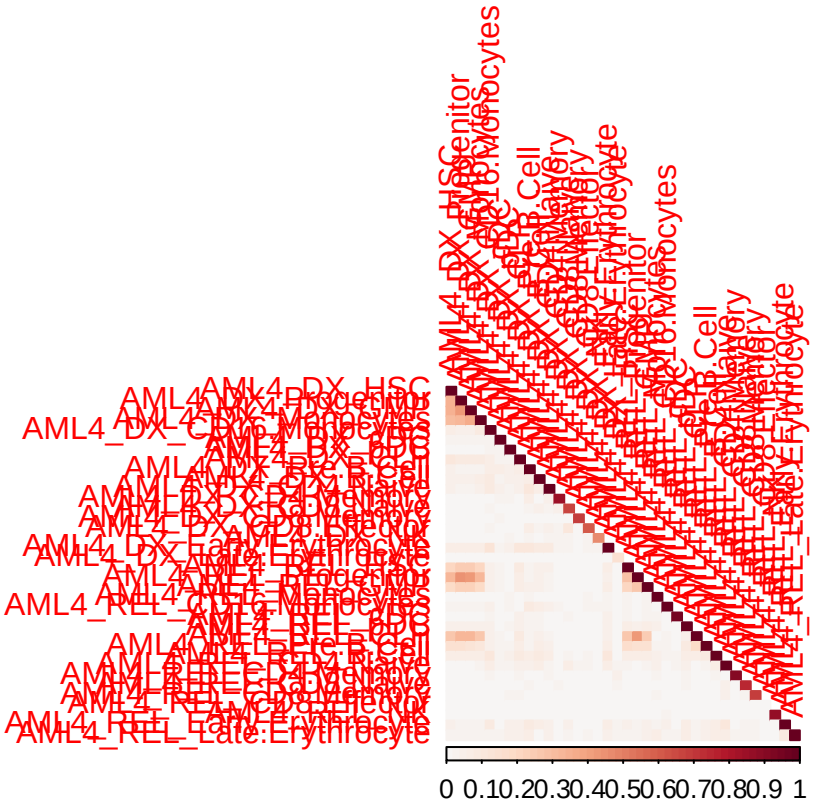

```
corrplot(mat_sorted_AML12_ATA, method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



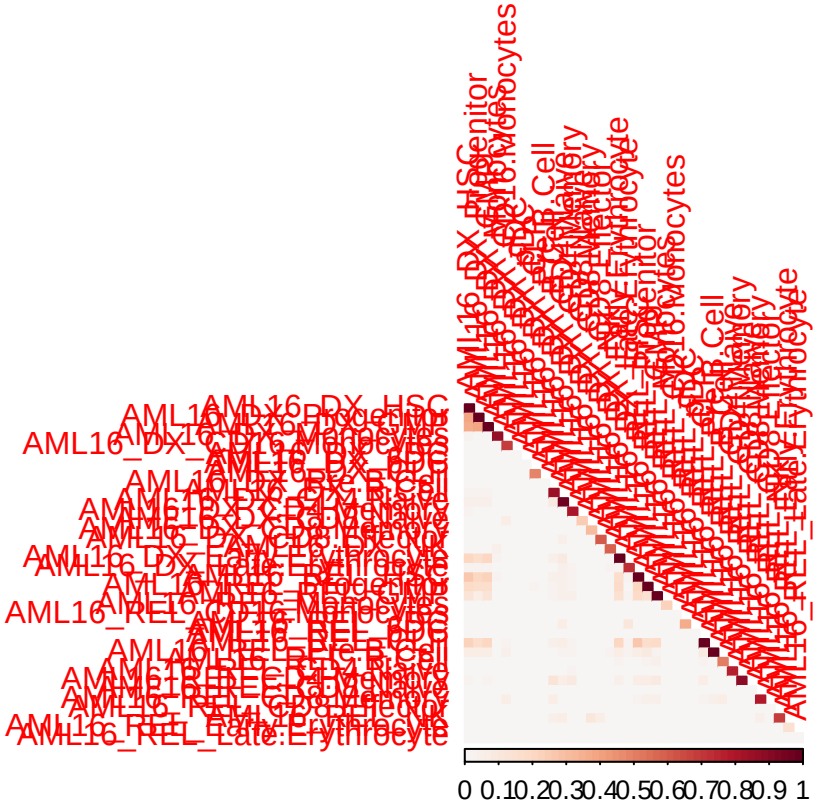

```
corrplot(mat_sorted_AML18_ATA, method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



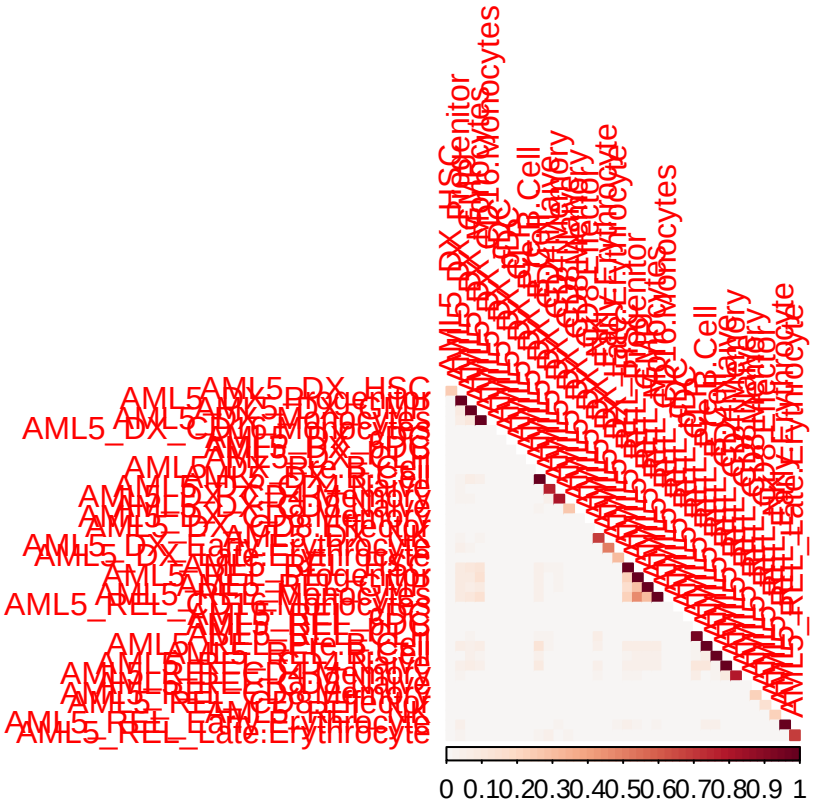

```
# RNA
corrplot(mat_sorted_AML04_RNA[celltype_comm_AML04, celltype_comm_AML04], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



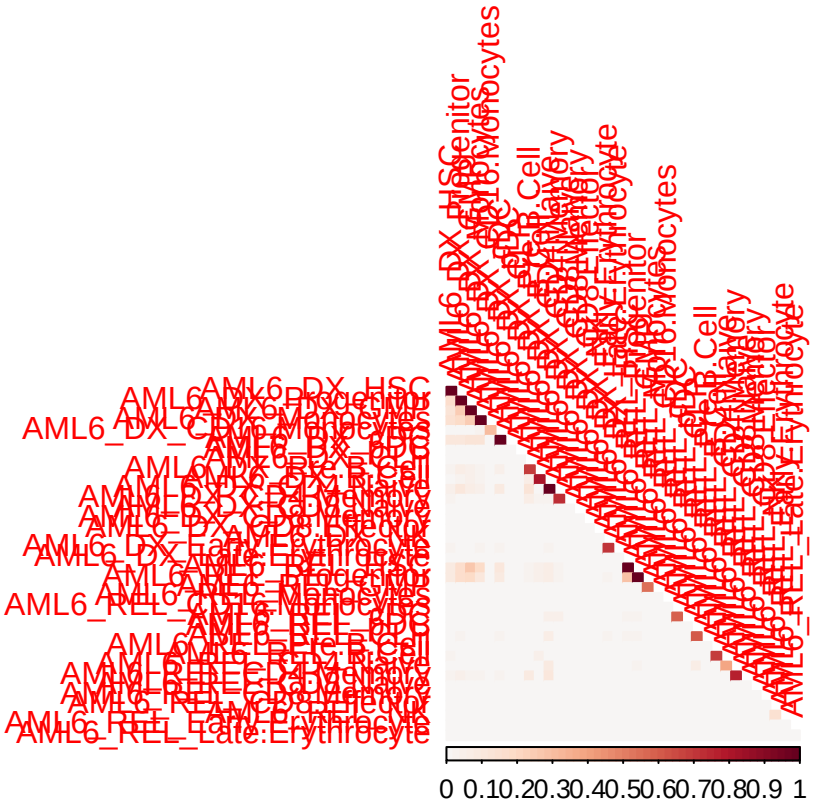

```
corrplot(mat_sorted_AML16_RNA[celltype_comm_AML16, celltype_comm_AML16], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



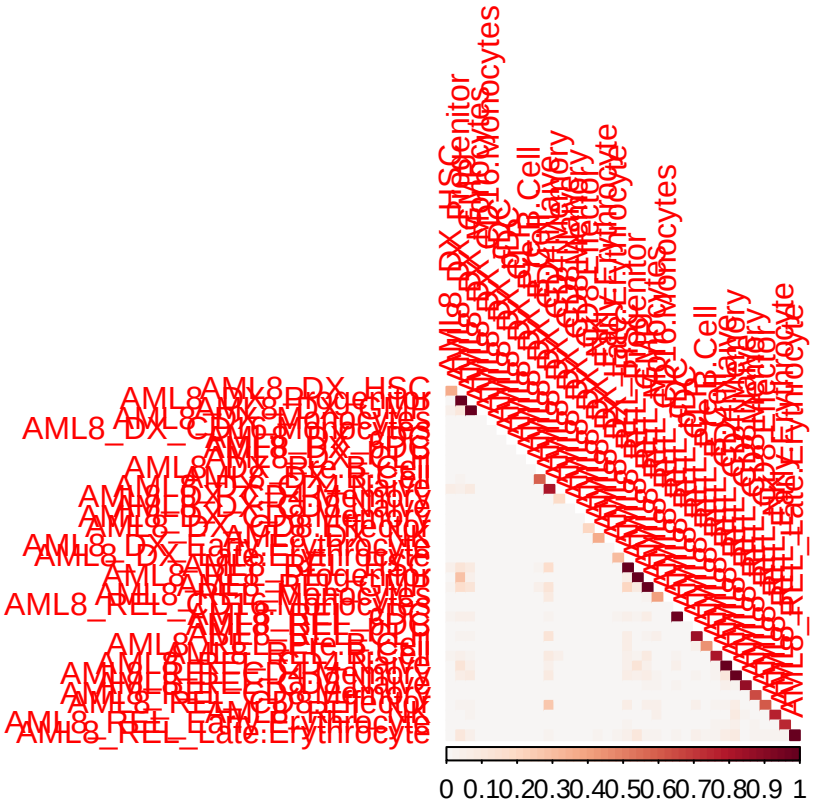

```
corrplot(mat_sorted_AML05_RNA[celltype_comm_AML05, celltype_comm_AML05], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



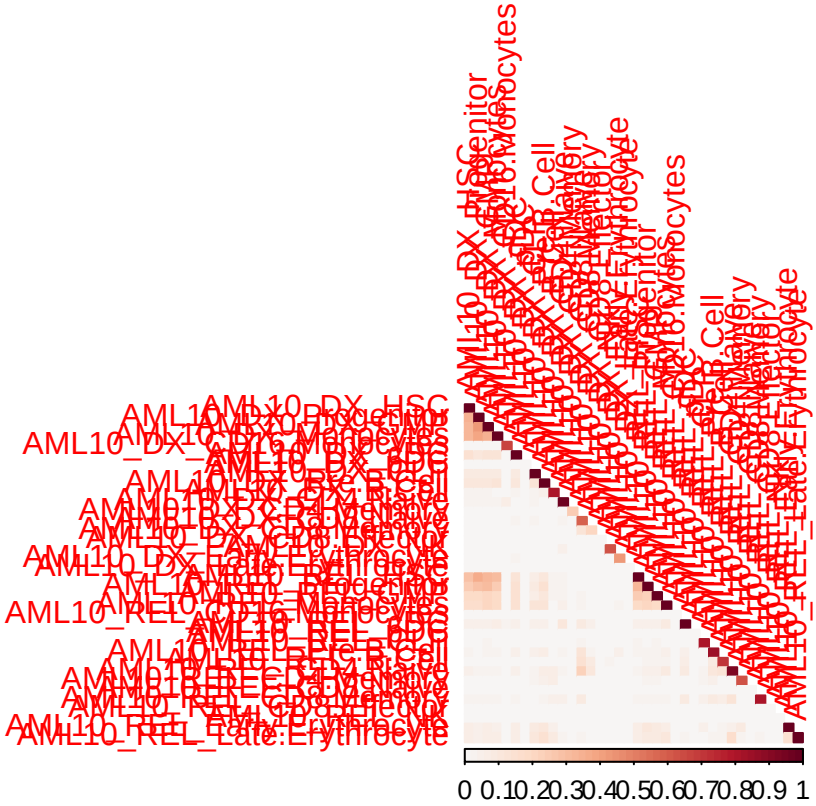

```
corrplot(mat_sorted_AML06_RNA[celltype_comm_AML06, celltype_comm_AML06], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



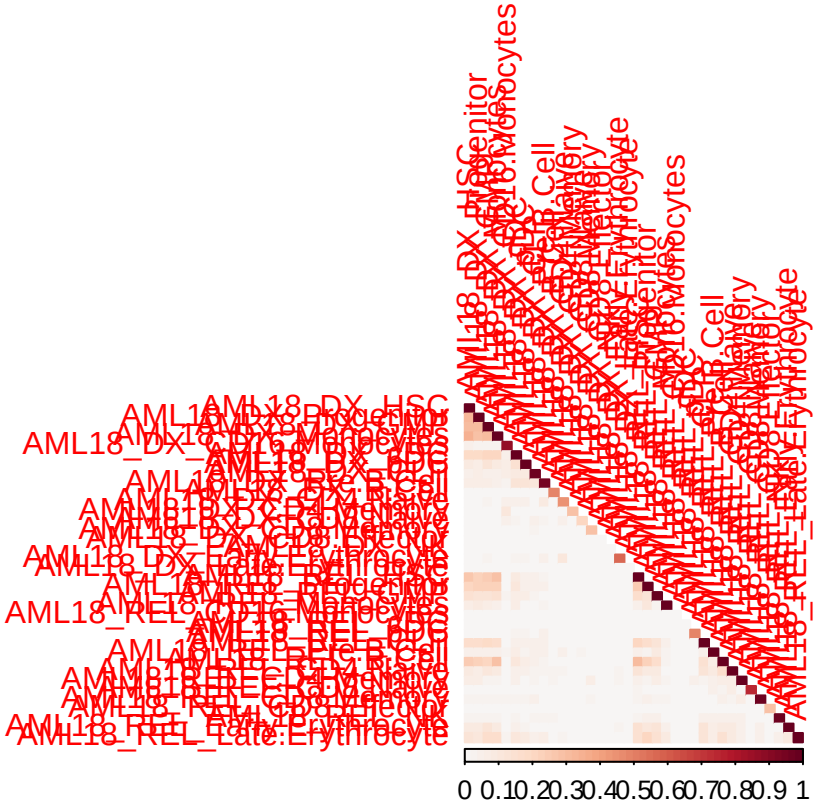

```
corrplot(mat_sorted_AML08_RNA[celltype_comm_AML08, celltype_comm_AML08], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



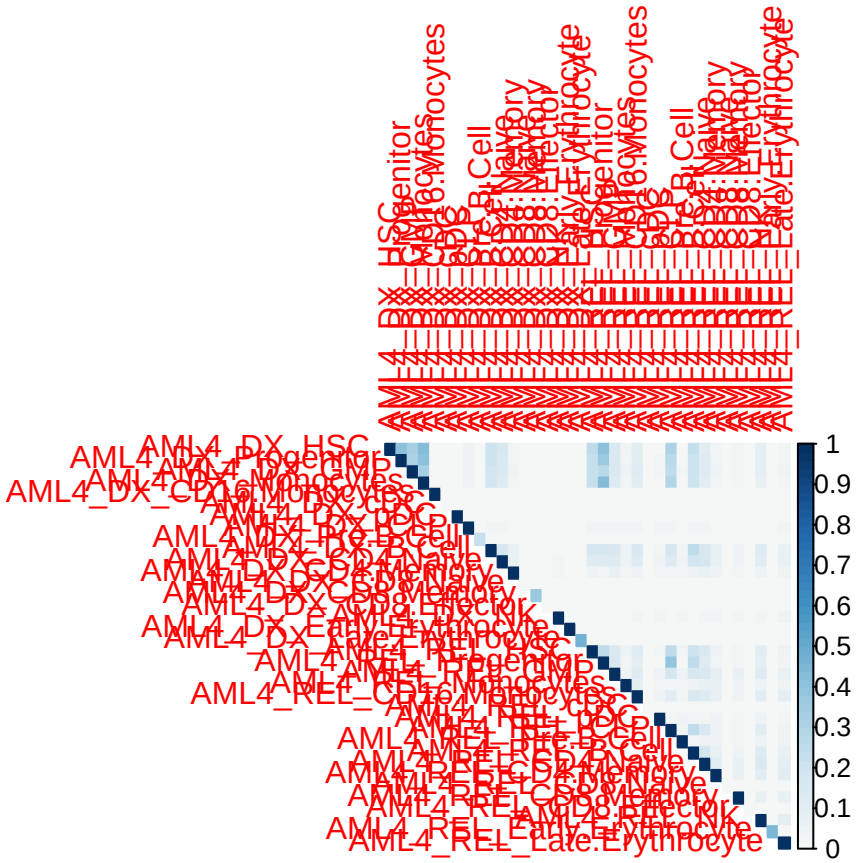

```
corrplot(mat_sorted_AML10_RNA[celltype_comm_AML10, celltype_comm_AML10], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



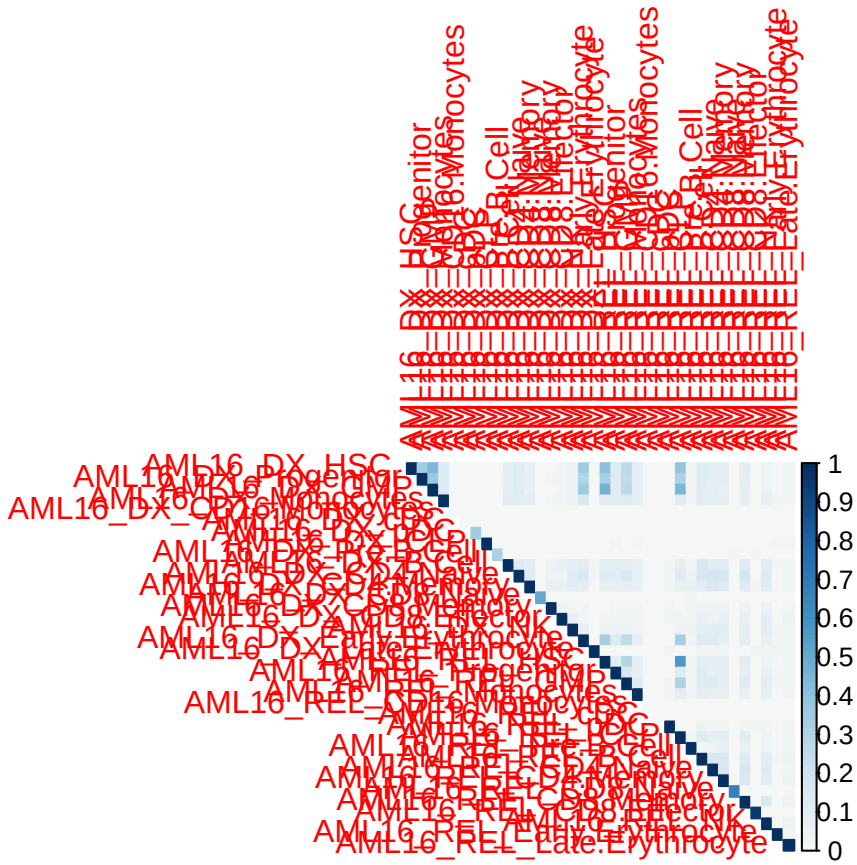

```
corrplot(mat_sorted_AML18_RNA[celltype_comm_AML18, celltype_comm_AML18], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
```



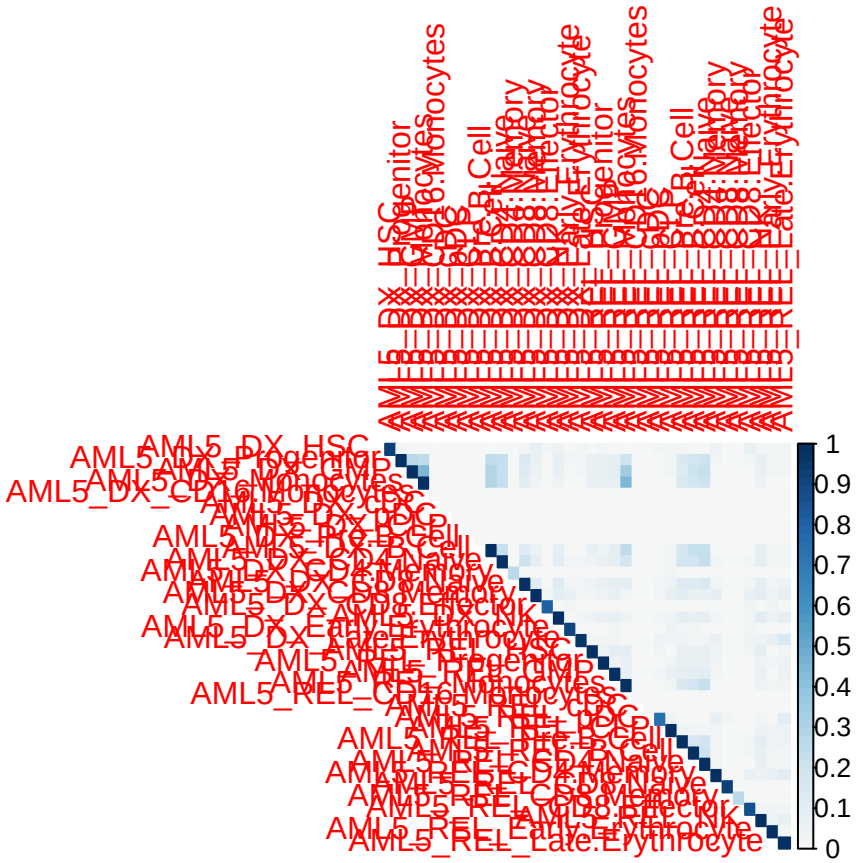

```
# ATA
corrplot(mat_sorted_AML04_ATA[celltype_comm_AML04, celltype_comm_AML04], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



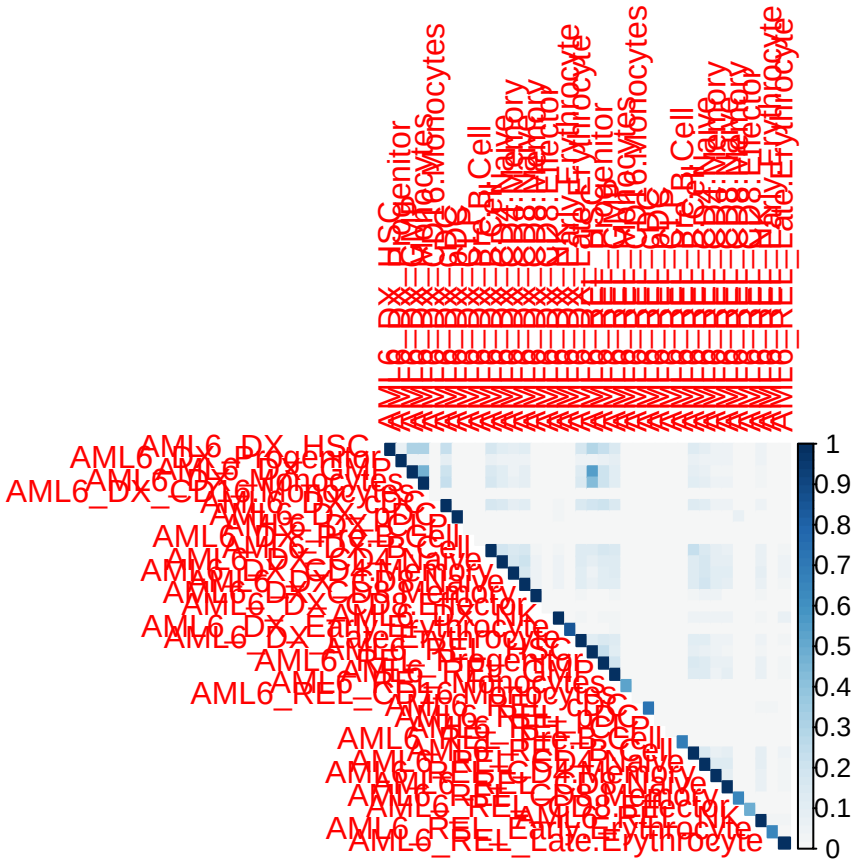

```
corrplot(mat_sorted_AML16_ATA[celltype_comm_AML16, celltype_comm_AML16], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



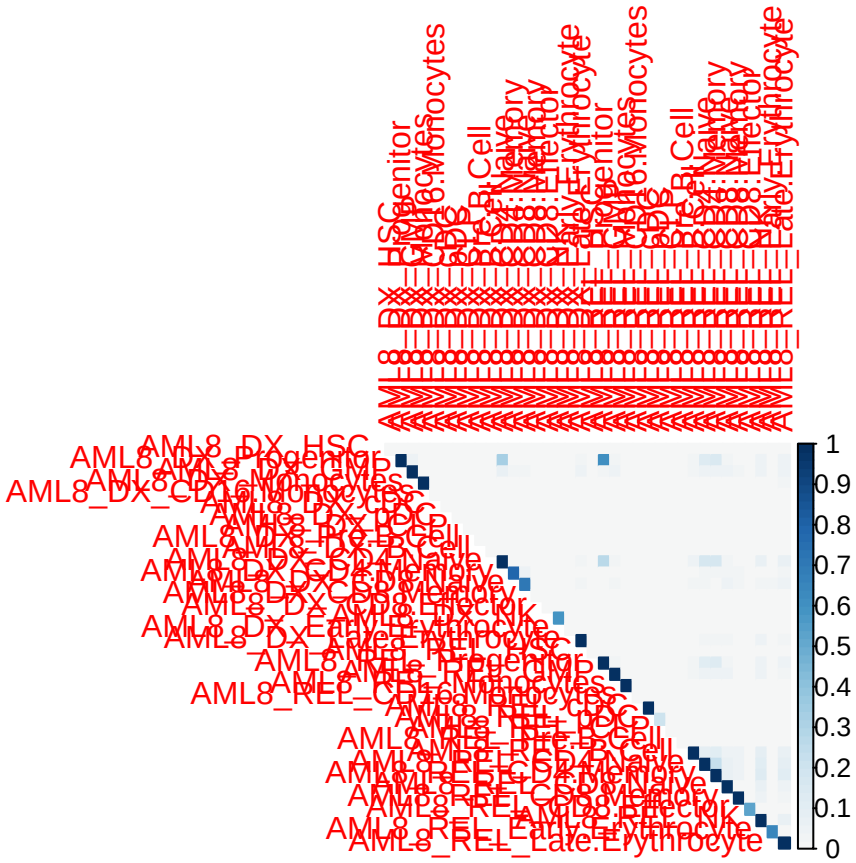

```
corrplot(mat_sorted_AML05_ATA[celltype_comm_AML05, celltype_comm_AML05], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



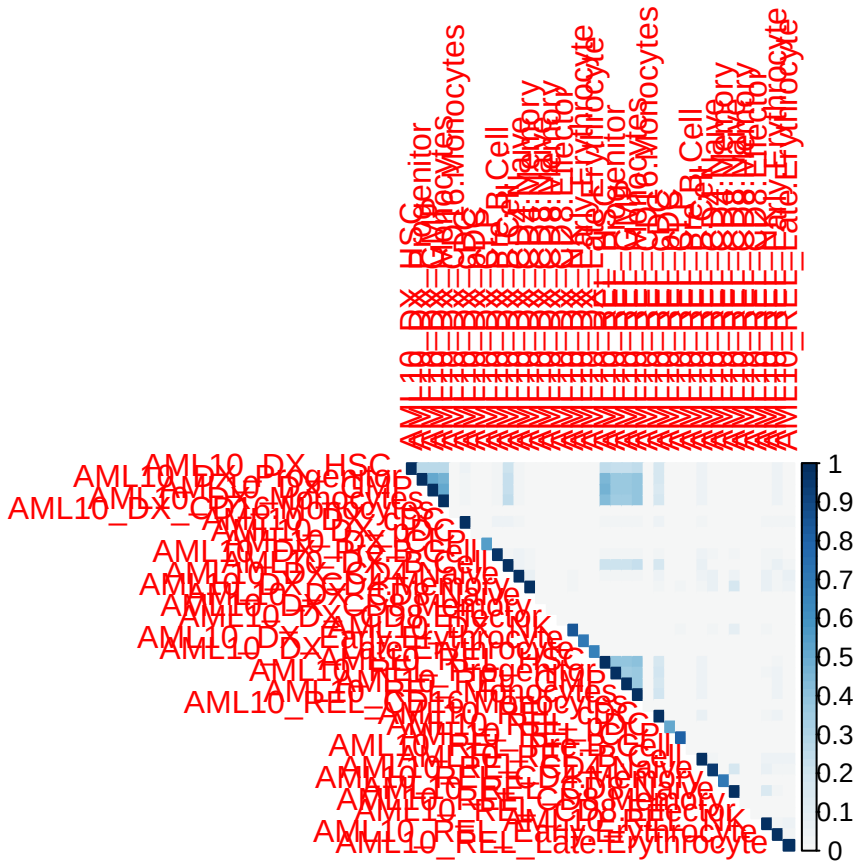

```
corrplot(mat_sorted_AML06_ATA[celltype_comm_AML06, celltype_comm_AML06], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```



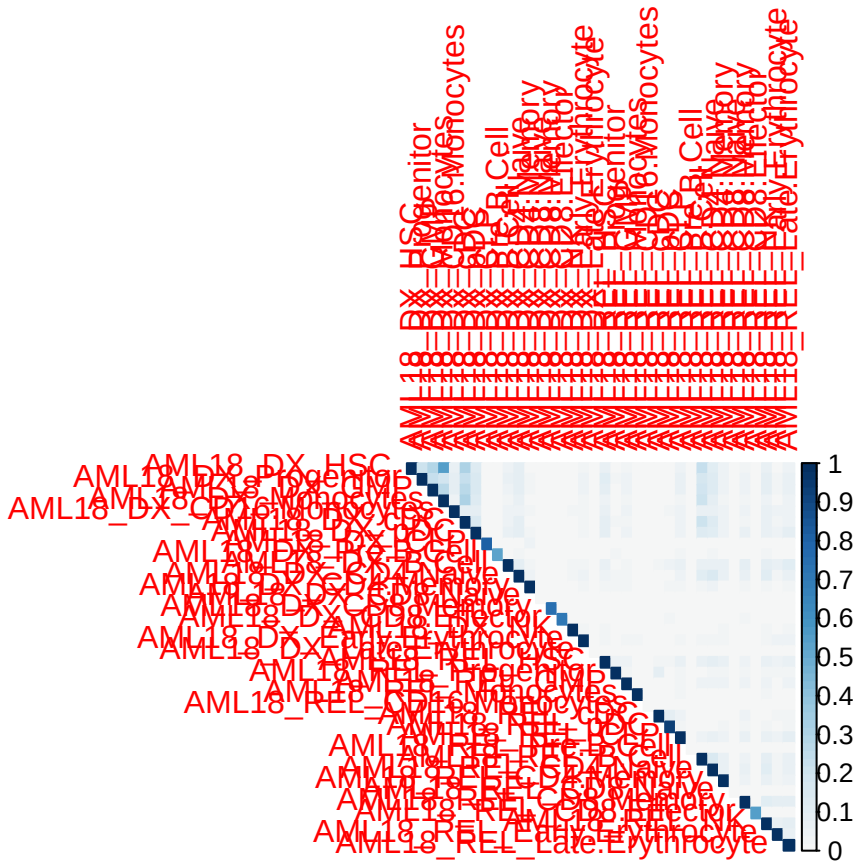

```
corrplot(mat_sorted_AML08_ATA[celltype_comm_AML08, celltype_comm_AML08], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```




```
corrplot(mat_sorted_AML10_ATA[celltype_comm_AML10, celltype_comm_AML10], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```




```
corrplot(mat_sorted_AML18_ATA[celltype_comm_AML18, celltype_comm_AML18], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
```




```
# save
pdf("../corrplot_AML04_RNA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML04_RNA[celltype_comm_AML04, celltype_comm_AML04], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
dev.off()
## pdf
##   2

pdf("../corrplot_AML16_RNA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML16_RNA[celltype_comm_AML16, celltype_comm_AML16], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
dev.off()
## pdf
##   2

pdf("../corrplot_AML05_RNA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML05_RNA[celltype_comm_AML05, celltype_comm_AML05], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
dev.off()
## pdf
##   2

pdf("../corrplot_AML06_RNA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML06_RNA[celltype_comm_AML06, celltype_comm_AML06], method = 'color',
```

```
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
dev.off()
## pdf
##   2

pdf("../corrplot_AML08_RNA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML08_RNA[celltype_comm_AML08, celltype_comm_AML08], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
dev.off()
## pdf
##   2

pdf("../corrplot_AML10_RNA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML10_RNA[celltype_comm_AML10, celltype_comm_AML10], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
dev.off()
## pdf
##   2

pdf("../corrplot_AML12_RNA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML12_RNA[celltype_comm_AML12, celltype_comm_AML12], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
dev.off()
## pdf
##   2

pdf("../corrplot_AML18_RNA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML18_RNA[celltype_comm_AML18, celltype_comm_AML18], method = 'color',
         type = 'lower', col = rev(colorRampPalette(brewer.pal(11, "RdBu"))(100)), col.lim = c(0, 1), o
dev.off()
## pdf
##   2

pdf("../corrplot_AML04_ATA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML04_ATA[celltype_comm_AML04, celltype_comm_AML04], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
dev.off()
## pdf
##   2

pdf("../corrplot_AML16_ATA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML16_ATA[celltype_comm_AML16, celltype_comm_AML16], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
dev.off()
## pdf
##   2

pdf("../corrplot_AML05_ATA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML05_ATA[celltype_comm_AML05, celltype_comm_AML05], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
dev.off()
## pdf
##   2
```

```r
pdf("../corrplot_AML06_ATA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML06_ATA[celltype_comm_AML06, celltype_comm_AML06], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
dev.off()
## pdf
##   2

pdf("../corrplot_AML08_ATA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML08_ATA[celltype_comm_AML08, celltype_comm_AML08], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
dev.off()
## pdf
##   2

pdf("../corrplot_AML10_ATA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML10_ATA[celltype_comm_AML10, celltype_comm_AML10], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
dev.off()
## pdf
##   2

pdf("../corrplot_AML12_ATA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML12_ATA[celltype_comm_AML12, celltype_comm_AML12], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
dev.off()
## pdf
##   2

pdf("../corrplot_AML18_ATA.pdf", width = 8, height = 8)
corrplot(mat_sorted_AML18_ATA[celltype_comm_AML18, celltype_comm_AML18], method = 'color',
         type = 'upper', col = colorRampPalette(brewer.pal(11, "RdBu"))(100), col.lim = c(0, 1), order =
dev.off()
## pdf
##   2
```

7 permutation test

```r
iteration_times <- 1:50
sample_size_f <- 0.50
sample_size <- 400

paired_perm_test_viz <- function(x, y, n_perm = 10000) {

  diff <- x - y
  obs <- mean(diff)

  # permutation null
  perm <- replicate(n_perm, {
    mean(diff * sample(c(-1, 1), length(diff), replace = TRUE))
  })
```

```r
  p <- mean(abs(perm) >= abs(obs))

  # CI from null distribution
  ci <- quantile(perm, c(0.025, 0.975))

  # return structured object for plotting
  list(
    obs = obs,
    p_value = p,
    ci = ci,
    perm_distribution = perm,
    diff_values = diff
  )
}

unpaired_perm_test_viz <- function(x, y, n_perm = 10000) {

  obs <- mean(x) - mean(y)
  combined <- c(x, y)
  n_x <- length(x)

  perm <- replicate(n_perm, {
    idx <- sample(length(combined))
    mean(combined[idx[1:n_x]]) - mean(combined[idx[-(1:n_x)]])
  })

  p <- mean(abs(perm) >= abs(obs))
  ci <- quantile(perm, c(0.025, 0.975))

  list(
    obs = obs,
    p_value = p,
    ci = ci,
    perm_distribution = perm
  )
}

plot_perm_test <- function(res, title = "Permutation Test") {

  df <- data.frame(perm = res$perm_distribution)

  ggplot2::ggplot(df, ggplot2::aes(x = perm)) +
    ggplot2::geom_histogram(bins = 50, fill = "grey80", color = "black") +
    ggplot2::geom_vline(xintercept = res$obs, color = "red", size = 1) +
    ggplot2::geom_vline(xintercept = res$ci, linetype = "dashed", color = "blue") +
    ggplot2::labs(
      title = title,
      subtitle = paste0("p = ", signif(res$p_value, 3)),
      x = "Permutation mean difference",
      y = "Frequency"
    ) +
    ggplot2::theme_classic()
}
```

```r
cal_sigValSimilarity <- function(mut_list, iteration_times, sampleid){
  # 1
  vals_RelLym_DxMye <- sapply(
    paste0("iter_", iteration_times),
    function(x) mut_list[[sampleid]][[x]][paste(sampleid, "REL_Lymphoid", sep = "_"), paste(sampleid, "
  )

  # 2
  vals_RelMye_DxMye <- sapply(
    paste0("iter_", iteration_times),
    function(x) mut_list[[sampleid]][[x]][paste(sampleid, "REL_Myeloid", sep = "_"), paste(sampleid, "DX
  )

  # 3
  vals_DxLym_RelMye <- sapply(
    paste0("iter_", iteration_times),
    function(x) mut_list[[sampleid]][[x]][paste(sampleid, "DX_Lymphoid", sep = "_"), paste(sampleid, "RE
  )

  # 4
  vals_DxLym_RelLym <- sapply(
    paste0("iter_", iteration_times),
    function(x) mut_list[[sampleid]][[x]][paste(sampleid, "DX_Lymphoid", sep = "_"), paste(sampleid, "RE
  )

  # 5
  vals_DxMye_DxLym <- sapply(
    paste0("iter_", iteration_times),
    function(x) mut_list[[sampleid]][[x]][paste(sampleid, "DX_Myeloid", sep = "_"), paste(sampleid, "DX_
  )

  # 6
  vals_DxMye_DxLym <- sapply(
    paste0("iter_", iteration_times),
    function(x) mut_list[[sampleid]][[x]][paste(sampleid, "REL_Myeloid", sep = "_"), paste(sampleid, "RE
  )

  # permutation test
  message(" 12")
  pair_12 <- unpaired_perm_test_viz(c(vals_RelLym_DxMye), c(vals_RelMye_DxMye))
  pair_12_p <- pair_12$p_value
  pair_12_c <- pair_12$obs - median(pair_12$perm_distribution)

  message(" 13")
  pair_13 <- unpaired_perm_test_viz(c(vals_RelLym_DxMye), c(vals_DxLym_RelMye))
  pair_13_p <- pair_13$p_value
  pair_13_c <- pair_13$obs - median(pair_13$perm_distribution)

  message(" 14")
  pair_14 <- unpaired_perm_test_viz(c(vals_RelLym_DxMye), c(vals_DxLym_RelLym))
  pair_14_p <- pair_14$p_value
```

```r
  pair_14_c <- pair_14$obs - median(pair_14$perm_distribution)

  message(" 15")
  pair_15 <- unpaired_perm_test_viz(c(vals_RelLym_DxMye), c(vals_DxMye_DxLym))
  pair_15_p <- pair_15$p_value
  pair_15_c <- pair_15$obs - median(pair_15$perm_distribution)

  message(" 16")
  pair_16 <- unpaired_perm_test_viz(c(vals_RelLym_DxMye), c(vals_DxMye_DxLym))
  pair_16_p <- pair_16$p_value
  pair_16_c <- pair_16$obs - median(pair_16$perm_distribution)

  result <- list(
    pair_12_p = pair_12_p,
    pair_12_c = pair_12_c,
    pair_13_p = pair_13_p,
    pair_13_c = pair_13_c,
    pair_14_p = pair_14_p,
    pair_14_c = pair_14_c,
    pair_15_p = pair_15_p,
    pair_15_c = pair_15_c,
    pair_16_p = pair_16_p,
    pair_16_c = pair_16_c
  )

  return(result)
}

selected_samples_id <- c("AML2", "AML4", "AML5", "AML6", "AML8", "AML10", "AML11", "AML12",
                         "AML13", "AML15", "AML16", "AML17", "AML18", "AML19", "AML21",
                         "AML22", "AML24", "AML25", "AML26", "AML27")
# RNA
data_scRNA_sigSimilarity <- as.data.frame(matrix(data = 0, nrow = length(selected_samples_id), ncol = 1
rownames(data_scRNA_sigSimilarity) <- selected_samples_id
colnames(data_scRNA_sigSimilarity) <- paste(c("pair_12_p", "pair_12_c", "pair_13_p", "pair_13_c",
                                              "pair_14_p", "pair_14_c", "pair_15_p", "pair_15_c",
                                              "pair_16_p", "pair_16_c"), "RNA", sep = "_")

# ATA
data_scATA_sigSimilarity <- as.data.frame(matrix(data = 0, nrow = length(selected_samples_id), ncol = 1
rownames(data_scATA_sigSimilarity) <- selected_samples_id
colnames(data_scATA_sigSimilarity) <- paste(c("pair_12_p", "pair_12_c", "pair_13_p", "pair_13_c",
                                              "pair_14_p", "pair_14_c", "pair_15_p", "pair_15_c",
                                              "pair_16_p", "pair_16_c"), "ATA", sep = "_")

for (sampleid in selected_samples_id){
  message(paste0(sampleid, "-scRNA"))
  r_scRNA <- cal_sigValSimilarity(mut_cellTypeAnn_similarity_list_RNA_main, iteration_times, sampleid)
  message(paste0(sampleid, "-scATA"))
  r_scATA <- cal_sigValSimilarity(mut_cellTypeAnn_similarity_list_ATA_main, iteration_times, sampleid)
  data_scRNA_sigSimilarity[sampleid, "pair_12_p_RNA"] <- r_scRNA$pair_12_p
  data_scRNA_sigSimilarity[sampleid, "pair_12_c_RNA"] <- r_scRNA$pair_12_c
  data_scRNA_sigSimilarity[sampleid, "pair_13_p_RNA"] <- r_scRNA$pair_13_p
```

```
  data_scRNA_sigSimilarity[sampleid, "pair_13_c_RNA"] <- r_scRNA$pair_13_c
  data_scRNA_sigSimilarity[sampleid, "pair_14_p_RNA"] <- r_scRNA$pair_14_p
  data_scRNA_sigSimilarity[sampleid, "pair_14_c_RNA"] <- r_scRNA$pair_14_c
  data_scRNA_sigSimilarity[sampleid, "pair_15_p_RNA"] <- r_scRNA$pair_15_p
  data_scRNA_sigSimilarity[sampleid, "pair_15_c_RNA"] <- r_scRNA$pair_15_c
  data_scRNA_sigSimilarity[sampleid, "pair_16_p_RNA"] <- r_scRNA$pair_16_p
  data_scRNA_sigSimilarity[sampleid, "pair_16_c_RNA"] <- r_scRNA$pair_16_c


  data_scATA_sigSimilarity[sampleid, "pair_12_p_ATA"] <- r_scATA$pair_12_p
  data_scATA_sigSimilarity[sampleid, "pair_12_c_ATA"] <- r_scATA$pair_12_c
  data_scATA_sigSimilarity[sampleid, "pair_13_p_ATA"] <- r_scATA$pair_13_p
  data_scATA_sigSimilarity[sampleid, "pair_13_c_ATA"] <- r_scATA$pair_13_c
  data_scATA_sigSimilarity[sampleid, "pair_14_p_ATA"] <- r_scATA$pair_14_p
  data_scATA_sigSimilarity[sampleid, "pair_14_c_ATA"] <- r_scATA$pair_14_c
  data_scATA_sigSimilarity[sampleid, "pair_15_p_ATA"] <- r_scATA$pair_15_p
  data_scATA_sigSimilarity[sampleid, "pair_15_c_ATA"] <- r_scATA$pair_15_c
  data_scATA_sigSimilarity[sampleid, "pair_16_p_ATA"] <- r_scATA$pair_16_p
  data_scATA_sigSimilarity[sampleid, "pair_16_c_ATA"] <- r_scATA$pair_16_c
}
## AML2-scRNA
##  12
##  13
##  14
##  15
##  16
## AML2-scATA
##  12
##  13
##  14
##  15
##  16
## AML4-scRNA
##  12
##  13
##  14
##  15
##  16
## AML4-scATA
##  12
##  13
##  14
##  15
##  16
## AML5-scRNA
##  12
##  13
##  14
##  15
##  16
## AML5-scATA
##  12
##  13
```

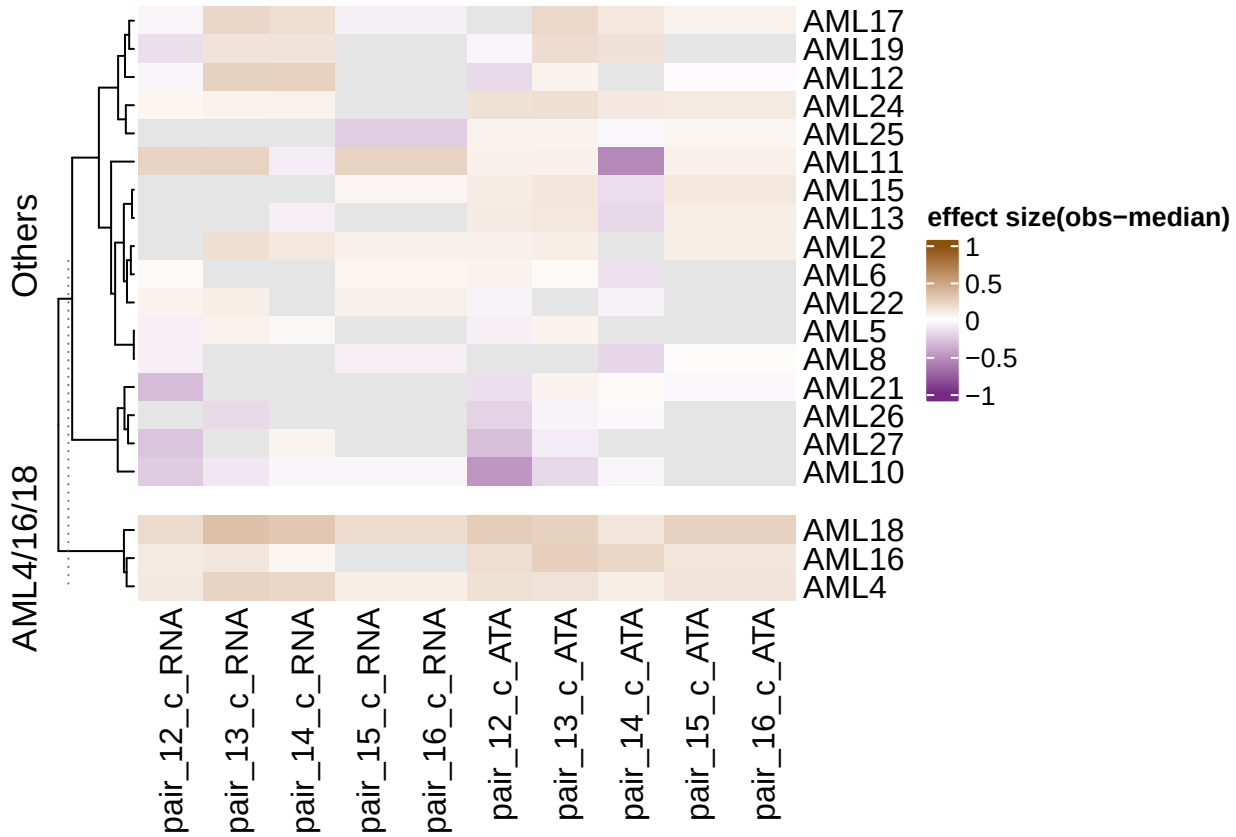
```
##  14
##  15
##  16
## AML6-scRNA
##  12
##  13
##  14
##  15
##  16
## AML6-scATA
##  12
##  13
##  14
##  15
##  16
## AML8-scRNA
##  12
##  13
##  14
##  15
##  16
## AML8-scATA
##  12
##  13
##  14
##  15
##  16
## AML10-scRNA
##  12
##  13
##  14
##  15
##  16
## AML10-scATA
##  12
##  13
##  14
##  15
##  16
## AML11-scRNA
##  12
##  13
##  14
##  15
##  16
## AML11-scATA
##  12
##  13
##  14
##  15
##  16
## AML12-scRNA
##  12
```

```
##  13
##  14
##  15
##  16
## AML12-scATA
##  12
##  13
##  14
##  15
##  16
## AML13-scRNA
##  12
##  13
##  14
##  15
##  16
## AML13-scATA
##  12
##  13
##  14
##  15
##  16
## AML15-scRNA
##  12
##  13
##  14
##  15
##  16
## AML15-scATA
##  12
##  13
##  14
##  15
##  16
## AML16-scRNA
##  12
##  13
##  14
##  15
##  16
## AML16-scATA
##  12
##  13
##  14
##  15
##  16
## AML17-scRNA
##  12
##  13
##  14
##  15
##  16
## AML17-scATA
```

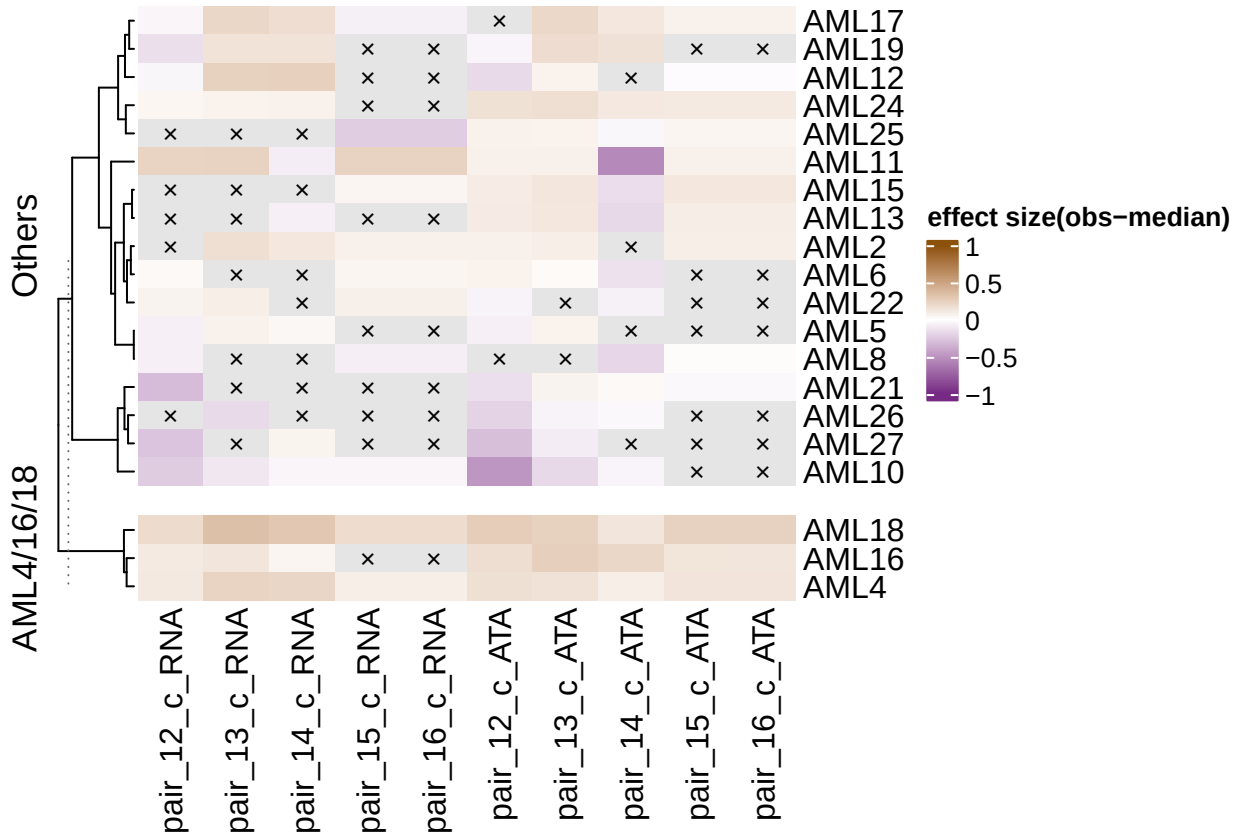
```
##  12
##  13
##  14
##  15
##  16
## AML18-scRNA
##  12
##  13
##  14
##  15
##  16
## AML18-scATA
##  12
##  13
##  14
##  15
##  16
## AML19-scRNA
##  12
##  13
##  14
##  15
##  16
## AML19-scATA
##  12
##  13
##  14
##  15
##  16
## AML21-scRNA
##  12
##  13
##  14
##  15
##  16
## AML21-scATA
##  12
##  13
##  14
##  15
##  16
## AML22-scRNA
##  12
##  13
##  14
##  15
##  16
## AML22-scATA
##  12
##  13
##  14
##  15
##  16
```

```
## AML24-scRNA
##  12
##  13
##  14
##  15
##  16
## AML24-scATA
##  12
##  13
##  14
##  15
##  16
## AML25-scRNA
##  12
##  13
##  14
##  15
##  16
## AML25-scATA
##  12
##  13
##  14
##  15
##  16
## AML26-scRNA
##  12
##  13
##  14
##  15
##  16
## AML26-scATA
##  12
##  13
##  14
##  15
##  16
## AML27-scRNA
##  12
##  13
##  14
##  15
##  16
## AML27-scATA
##  12
##  13
##  14
##  15
##  16

df_RNA <- data_scRNA_sigSimilarity
pairs_RNA <- unique(sub("_p_RNA", "", grep("_p_RNA$", colnames(df_RNA), value = TRUE)))
for (p in pairs_RNA) {
  p_col <- paste0(p, "_p_RNA")
```

```r
  c_col <- paste0(p, "_c_RNA")
  df_RNA[[c_col]][df_RNA[[p_col]] > 0.0001] <- NA
}

df_ATA <- data_scATA_sigSimilarity
pairs_ATA <- unique(sub("_p_ATA", "", grep("_p_ATA$", colnames(df_ATA), value = TRUE)))
for (p in pairs_ATA) {
  p_col <- paste0(p, "_p_ATA")
  c_col <- paste0(p, "_c_ATA")
  df_ATA[[c_col]][df_ATA[[p_col]] > 0.0001] <- NA
}


data_scRNA_scATA_sigSimilarity <- cbind(df_RNA[, c("pair_12_c_RNA", "pair_13_c_RNA", "pair_14_c_RNA", "p
                                        df_ATA[, c("pair_12_c_ATA", "pair_13_c_ATA", "pair_14_c_ATA", "p

#max_abs <- max(abs(data_scRNA_scATA_sigSimilarity), na.rm = TRUE)
#breaks <- seq(-max_abs, max_abs, length.out = 101)

#colors <- rev(colorRampPalette(brewer.pal(11, "BrBG"))(100))

col_fun <- colorRamp2(
  c(-1, 0, 1),
  c("#762a83", "white", "#8c510a")
)

mat <- data_scRNA_scATA_sigSimilarity

group <- ifelse(rownames(mat) %in% c("AML4", "AML16", "AML18"),
                "AML4/16/18", "Others")
group <- factor(group, levels = c("Others", "AML4/16/18"))

Heatmap(
  mat,
  name = "effect size(obs-median)",
  row_split = group,
  cluster_rows = TRUE,
  clustering_method_rows = "ward.D",
  cluster_columns = FALSE,
  col = col_fun,
  show_row_names = TRUE,
  row_gap = unit(4, "mm"),
  na_col = "grey90"
)
## Warning: The input is a data frame-like object, convert it to a matrix.
```



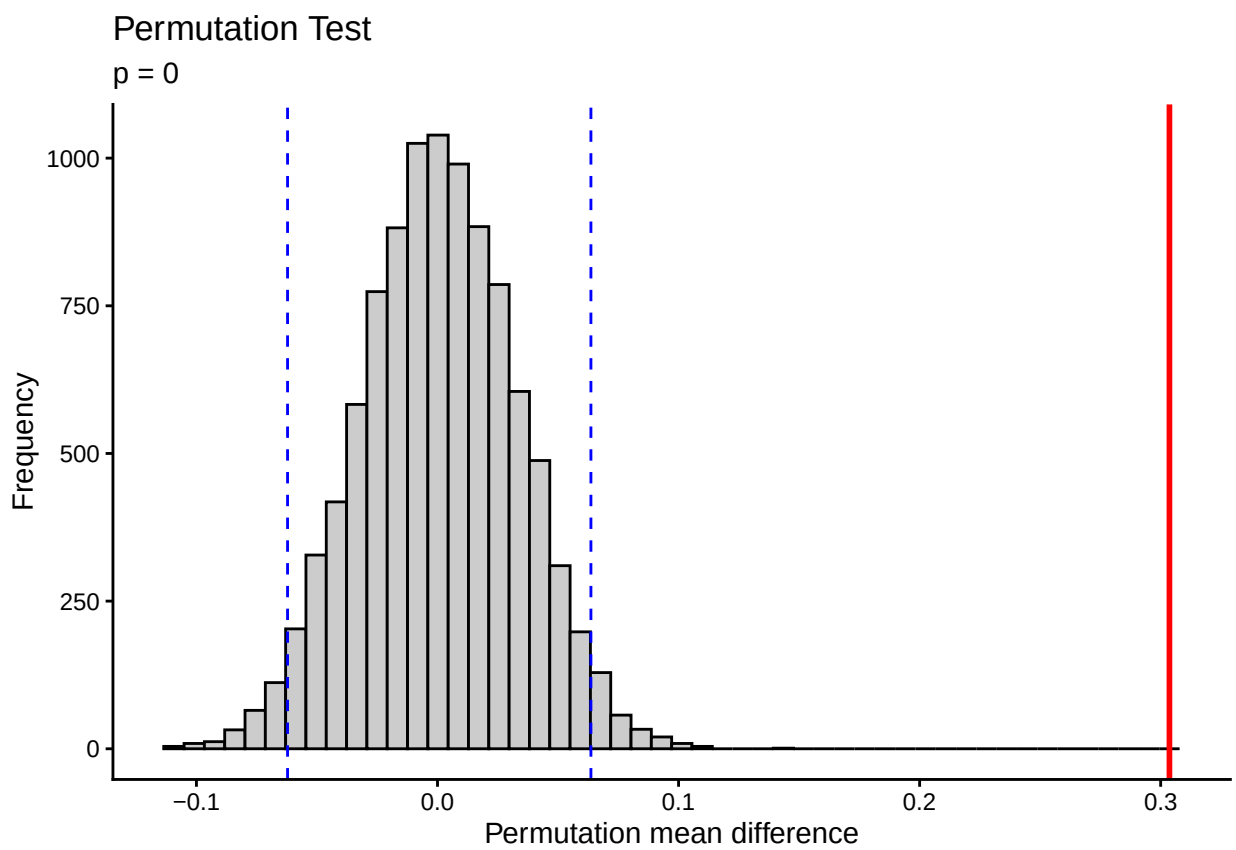

```
Heatmap(
  mat,
  name = "effect size(obs-median)",
  row_split = group,
  cluster_rows = TRUE,
  clustering_method_rows = "ward.D",
  cluster_columns = FALSE,
  col = col_fun,
  show_row_names = TRUE,
  row_gap = unit(4, "mm"),
  na_col = "grey90",
  cell_fun = function(j, i, x, y, w, h, fill) {
    if (is.na(mat[i, j])) {
      grid::grid.text(
        "×", x = x, y = y,
        gp = grid::gpar(fontsize = 10, col = "black")
      )
    }
  }
)
## Warning: The input is a data frame-like object, convert it to a matrix.
```



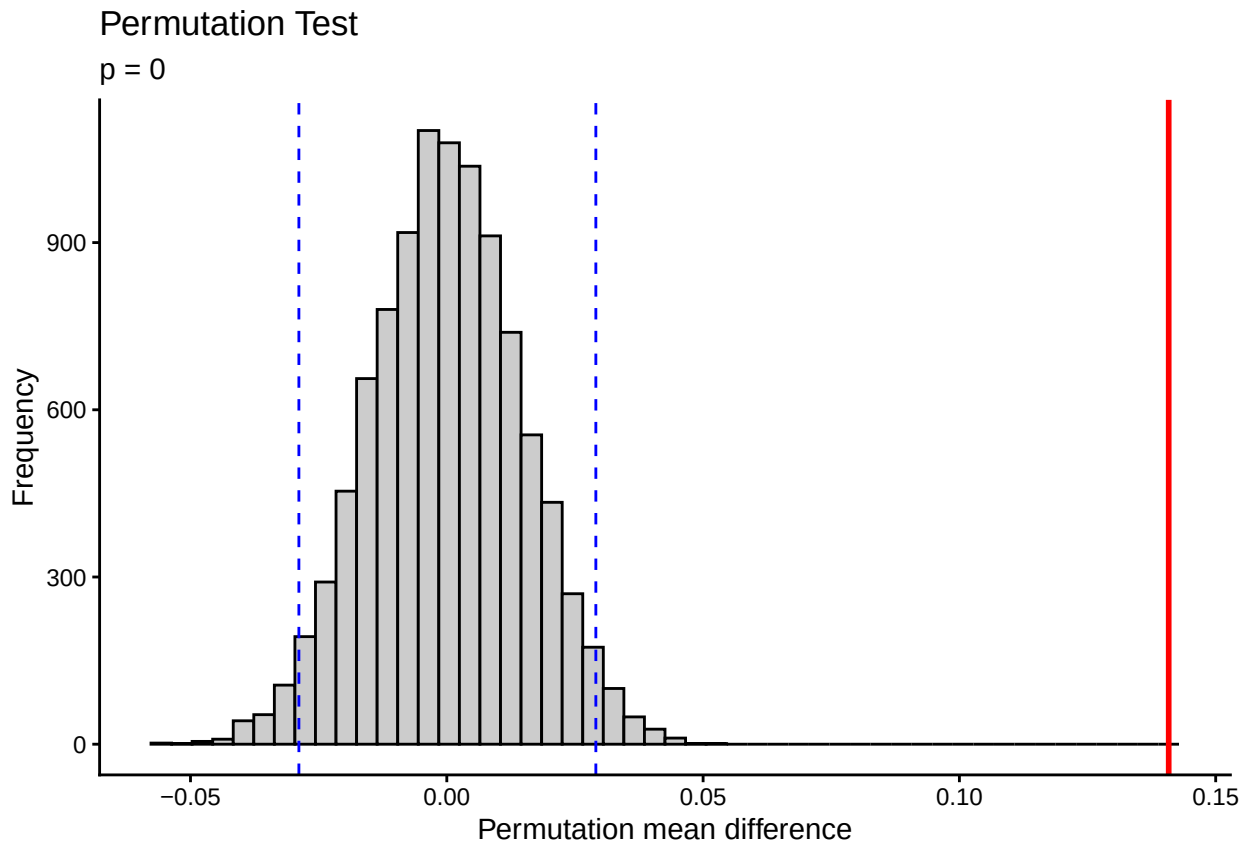

```
ht <- Heatmap(
  mat,
  name = "effect size(obs-median)",
  row_split = group,
  cluster_rows = TRUE,
  clustering_method_rows = "ward.D",
  cluster_columns = FALSE,
  col = col_fun,
  show_row_names = TRUE,
  row_gap = unit(4, "mm"),
  na_col = "grey90"
)
## Warning: The input is a data frame-like object, convert it to a matrix.

pdf("heatmap_effect_size.pdf", width = 5.8, height = 6)

draw(ht)

dev.off()
## pdf
##   2

sample_selected <- "AML18"
```

```r
# 1
vals_RelLym_DxMye_RNA <- sapply(
  paste0("iter_", iteration_times),
  function(x) mut_cellTypeAnn_similarity_list_RNA_main[[sample_selected]][[x]][paste(sample_selected, "
)

vals_RelLym_DxMye_ATA <- sapply(
  paste0("iter_", iteration_times),
  function(x) mut_cellTypeAnn_similarity_list_ATA_main[[sample_selected]][[x]][paste(sample_selected, "
)

# 2
vals_RelLym_DxLym_RNA <- sapply(
  paste0("iter_", iteration_times),
  function(x) mut_cellTypeAnn_similarity_list_RNA_main[[sample_selected]][[x]][paste(sample_selected, "
)

vals_RelLym_DxLym_ATA <- sapply(
  paste0("iter_", iteration_times),
  function(x) mut_cellTypeAnn_similarity_list_ATA_main[[sample_selected]][[x]][paste(sample_selected, "
)
# 3
vals_DxMye_DxLym_RNA<- sapply(
  paste0("iter_", iteration_times),
  function(x) mut_cellTypeAnn_similarity_list_RNA_main[[sample_selected]][[x]][paste(sample_selected, "
)

vals_DxMye_DxLym_ATA<- sapply(
  paste0("iter_", iteration_times),
  function(x) mut_cellTypeAnn_similarity_list_ATA_main[[sample_selected]][[x]][paste(sample_selected, "
)
# 4
vals_RelMye_RelLym_RNA <- sapply(
    paste0("iter_", iteration_times),
    function(x) mut_cellTypeAnn_similarity_list_RNA_main[[sample_selected]][[x]][paste(sample_selected,
  )
vals_RelMye_RelLym_ATA <- sapply(
    paste0("iter_", iteration_times),
    function(x) mut_cellTypeAnn_similarity_list_ATA_main[[sample_selected]][[x]][paste(sample_selected,
  )

set.seed(2025)
res_RNA12 <- unpaired_perm_test_viz(vals_RelLym_DxMye_RNA, vals_RelLym_DxLym_RNA)
res_RNA13 <- unpaired_perm_test_viz(vals_RelLym_DxMye_RNA, vals_DxMye_DxLym_RNA)
res_RNA14 <- unpaired_perm_test_viz(vals_RelLym_DxMye_RNA, vals_RelMye_RelLym_RNA)

res_ATA12 <- unpaired_perm_test_viz(vals_RelLym_DxMye_ATA, vals_RelLym_DxLym_ATA)
res_ATA13 <- unpaired_perm_test_viz(vals_RelLym_DxMye_ATA, vals_DxMye_DxLym_ATA)
res_ATA14 <- unpaired_perm_test_viz(vals_RelLym_DxMye_ATA, vals_RelMye_RelLym_ATA)


p_res_RNA12 <- plot_perm_test(res_RNA12)
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
```

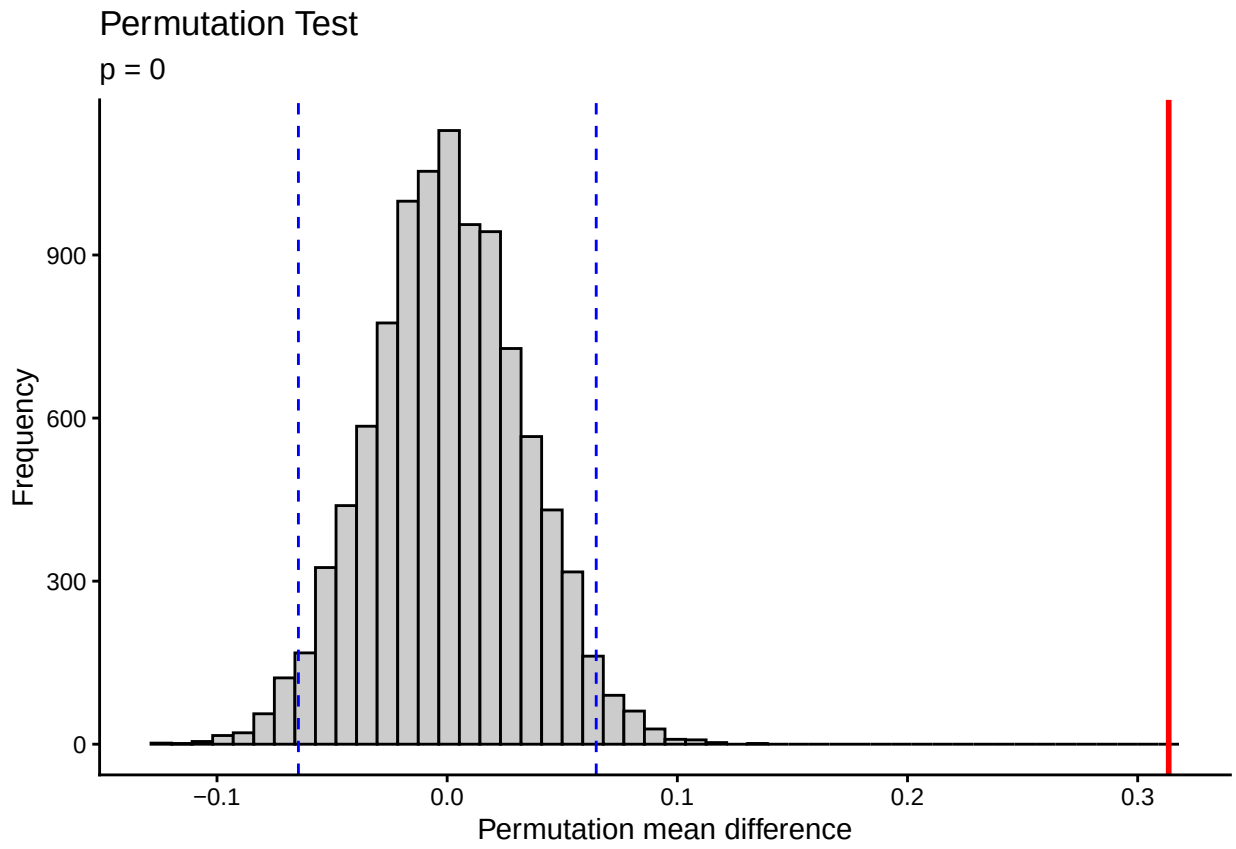
```
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
p_res_ATA12 <- plot_perm_test(res_ATA12)
p_res_RNA13 <- plot_perm_test(res_RNA13)
p_res_ATA13 <- plot_perm_test(res_ATA13)
p_res_RNA14 <- plot_perm_test(res_RNA14)
p_res_ATA14 <- plot_perm_test(res_ATA14)

p_res_RNA12
```



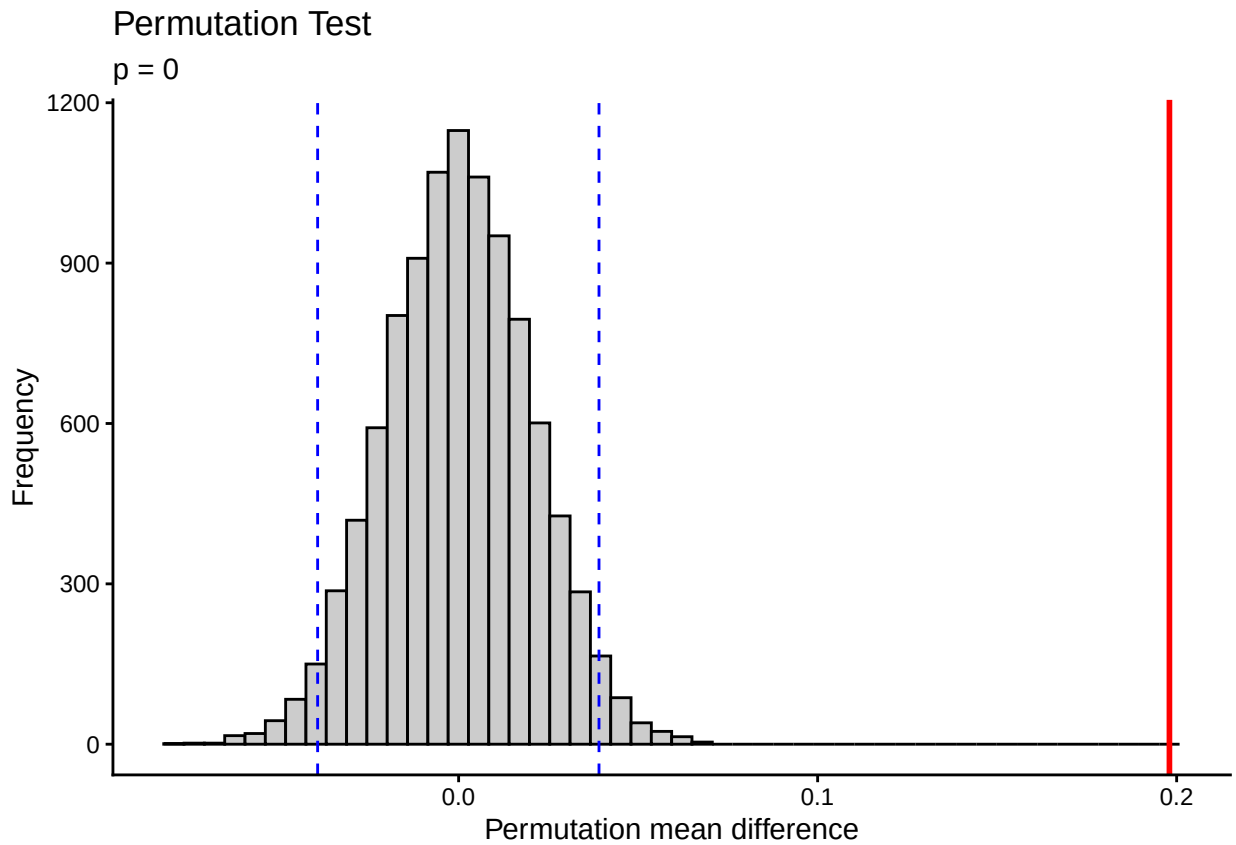

```
p_res_ATA12
```


Permutation Test
p = 0
Frequency
900
600
300
0
−0.05
0.00
0.05
0.10
0.15
Permutation mean difference

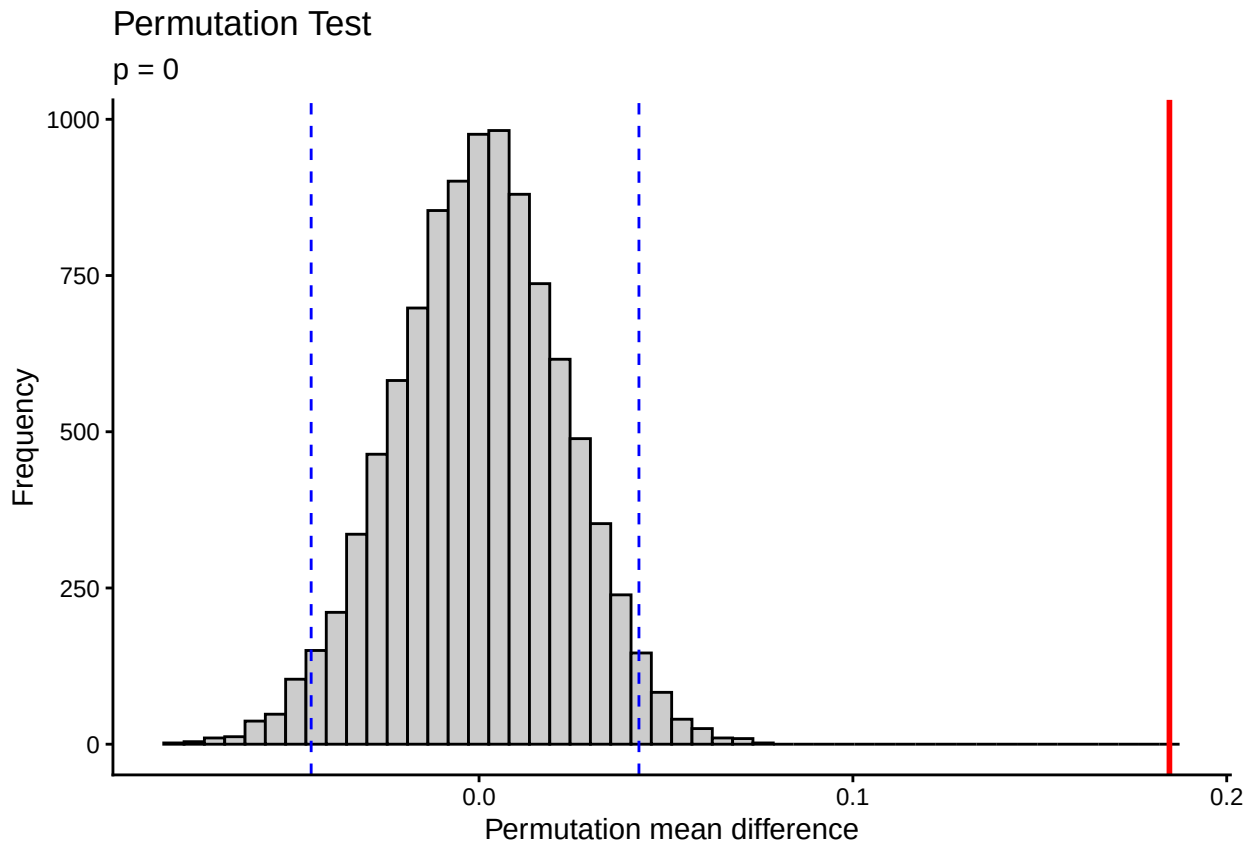
```
p_res_RNA13
```


Permutation Test
p = 0
Frequency
900
600
300
0
−0.1
0.0
0.1
0.2
0.3
Permutation mean difference

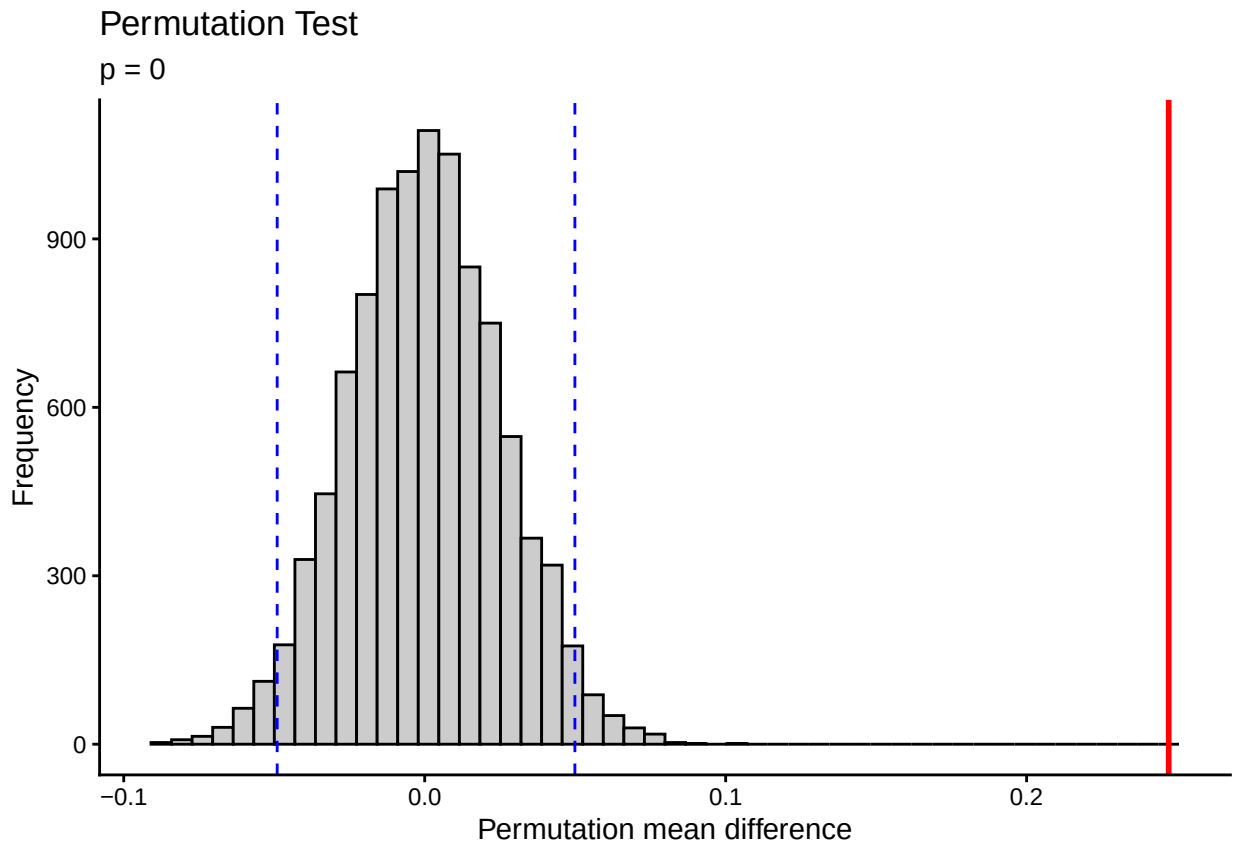
```
p_res_ATA13
```


Permutation Test
p = 0
1200
900
600
300
0
Frequency
0.0
0.1
0.2
Permutation mean difference

```
p_res_RNA14
```

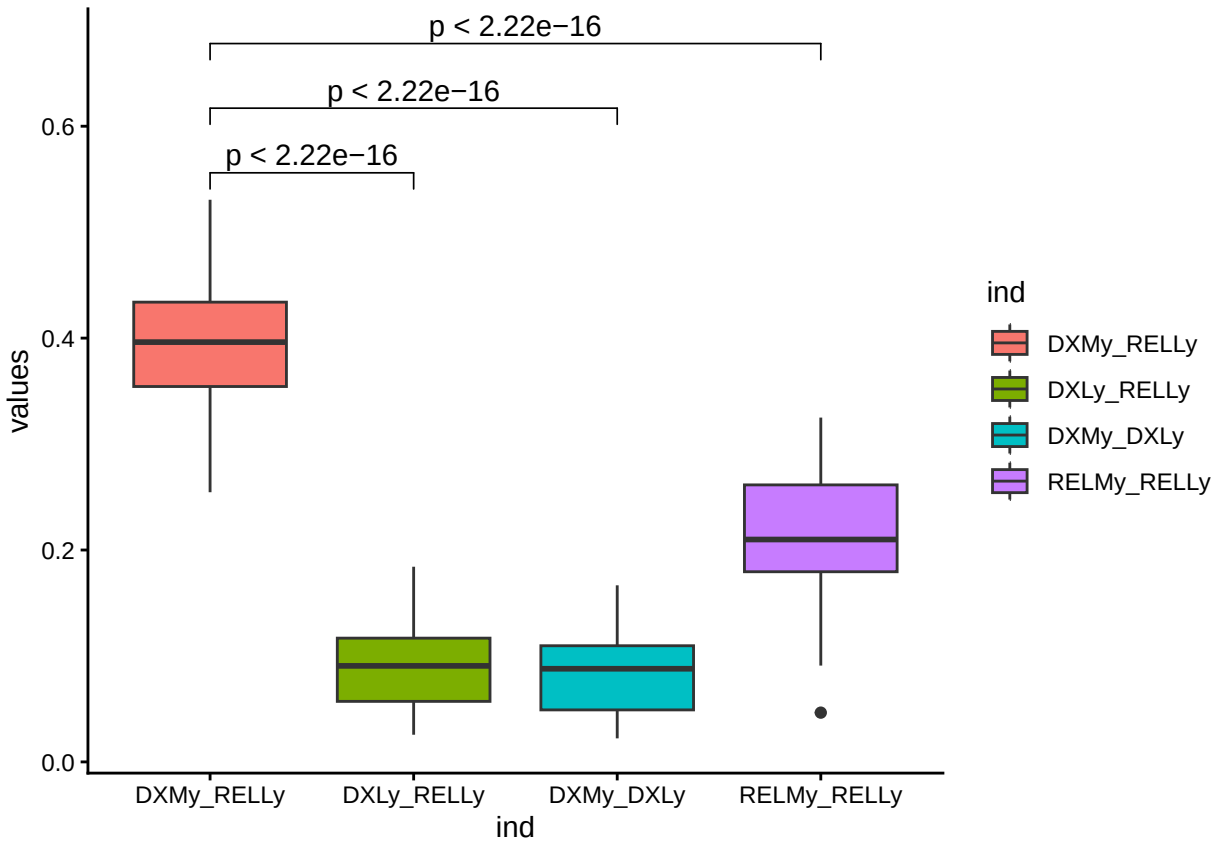

Permutation Test
p = 0
1000
750
500
250
0
Frequency
0.0
0.1
0.2
Permutation mean difference

```
p_res_ATA14
```

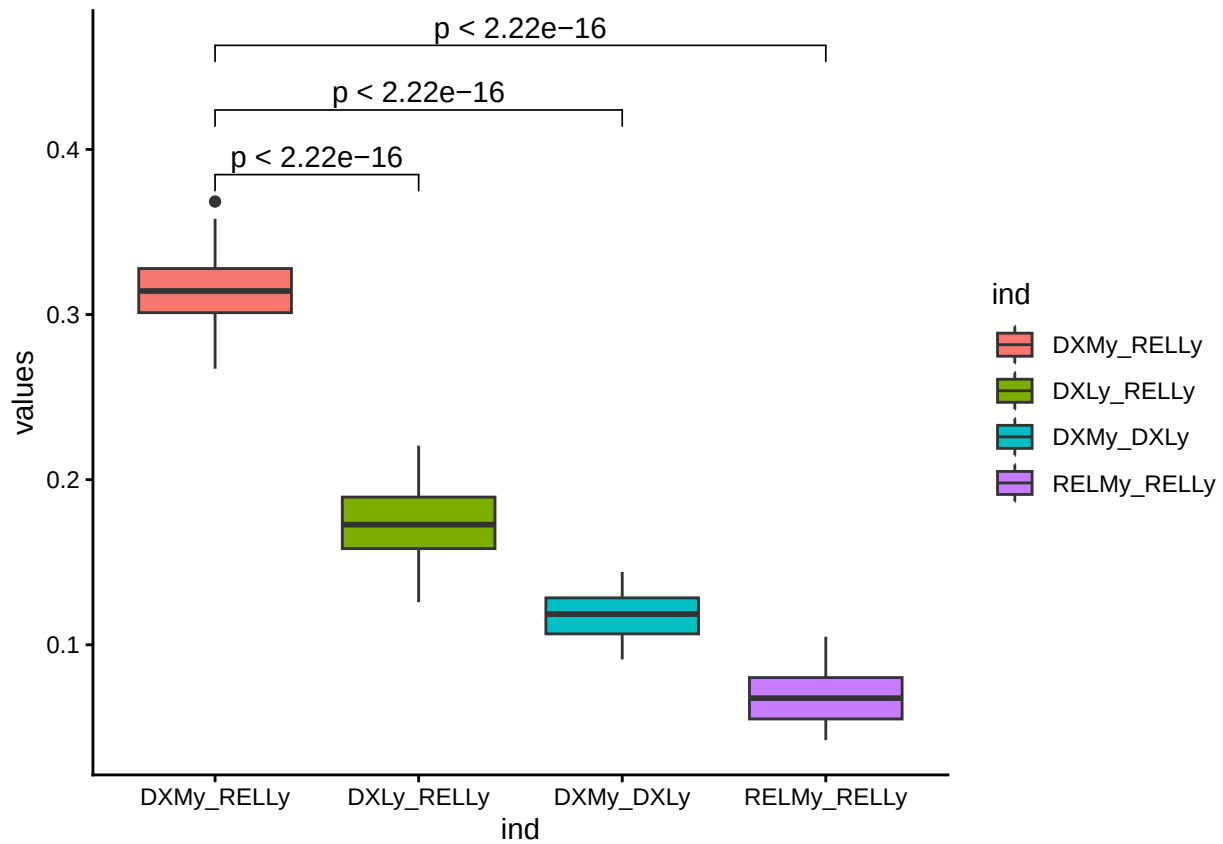



```
data_value_similarity_RNA <- list(DXMy_RELLy = vals_RelLym_DxMye_RNA,
                                  DXLy_RELLy = vals_RelLym_DxLym_RNA,
                                  DXMy_DXLy = vals_DxMye_DxLym_RNA,
                                  RELMy_RELLy = vals_RelMye_RelLym_RNA)

data_value_similarity_ATA <- list(DXMy_RELLy = vals_RelLym_DxMye_ATA,
                                  DXLy_RELLy = vals_RelLym_DxLym_ATA,
                                  DXMy_DXLy = vals_DxMye_DxLym_ATA,
                                  RELMy_RELLy = vals_RelMye_RelLym_ATA)

comparisons <- list(
  c("DXMy_RELLy", "DXLy_RELLy"),
  c("DXMy_RELLy", "DXMy_DXLy"),
  c("DXMy_RELLy", "RELMy_RELLy")
)

p_boxPlotdata_value_similarity_RNA <- ggplot(stack(data_value_similarity_RNA), aes(ind, values, fill = 
  geom_boxplot() +
  stat_compare_means(comparisons = comparisons, method = "wilcox.test") +
  theme_classic()

p_boxPlotdata_value_similarity_ATA <- ggplot(stack(data_value_similarity_ATA), aes(ind, values, fill = 
  geom_boxplot() +
  stat_compare_means(comparisons = comparisons, method = "wilcox.test") +
  theme_classic()
```

```
p_boxPlotdata_value_similarity_RNA
```




```
p_boxPlotdata_value_similarity_ATA
```




```
#ggsave("../p_boxPlotdata_value_similarity_RNA.pdf", p_boxPlotdata_value_similarity_RNA, width = 3.2, h
#ggsave("../p_boxPlotdata_value_similarity_ATA.pdf", p_boxPlotdata_value_similarity_ATA, width = 3.2, h
```

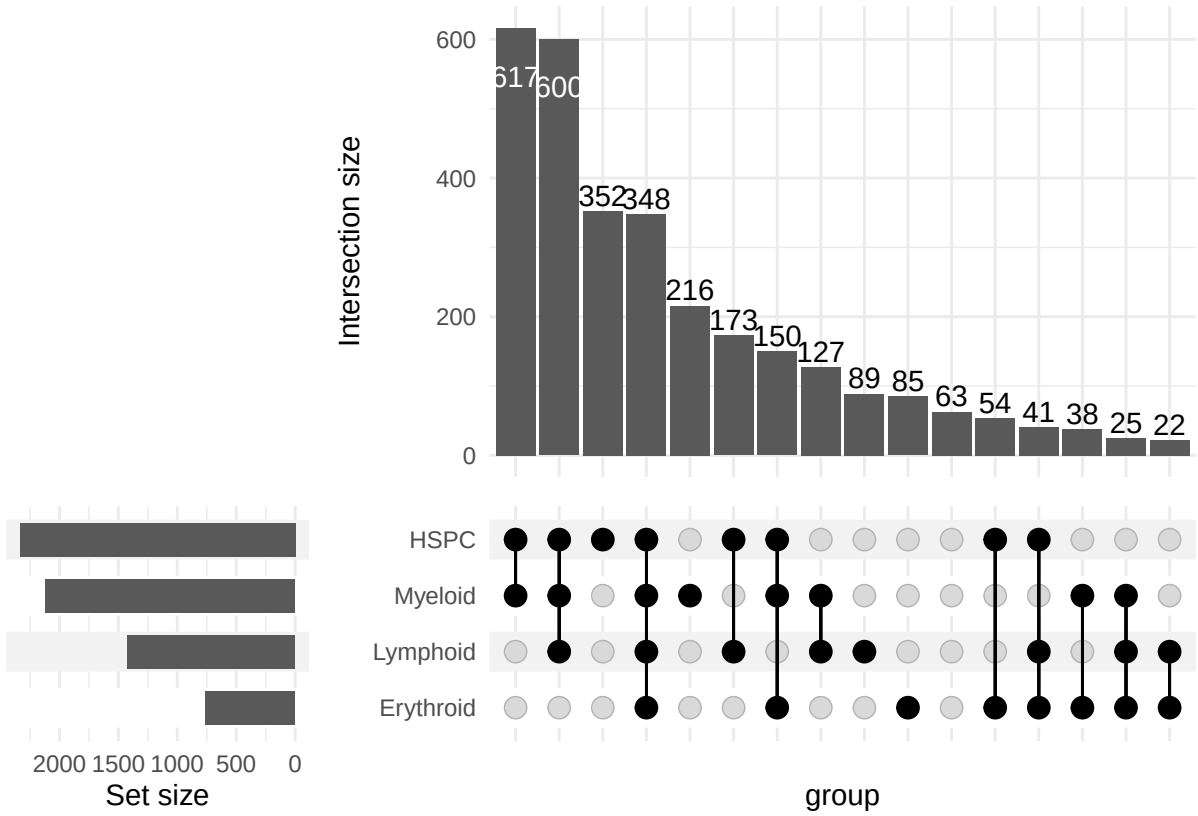
8 mutations shared by different lineage

```
cellxmut_to_01 <- function(cellxmut,
                           mapping = c(NoCall = 0L, WT = 0L, Mut = 1L),
                           default_value = NA_integer_,
                           return_matrix = TRUE) {
  if (!(is.matrix(cellxmut) || is.data.frame(cellxmut))) {
    stop("cellxmut must be a matrix or data.frame.")
  }

  rn <- rownames(cellxmut)
  cn <- colnames(cellxmut)
  if (is.null(rn) || any(rn == "")) stop("cellxmut must have non-empty rownames (cell names).")
  if (is.null(cn) || any(cn == "")) stop("cellxmut must have non-empty colnames (mutation sites).")

  # Work in character space, trim whitespace
  x <- as.matrix(cellxmut)
  x <- apply(x, c(1, 2), function(v) trimws(as.character(v)))
```

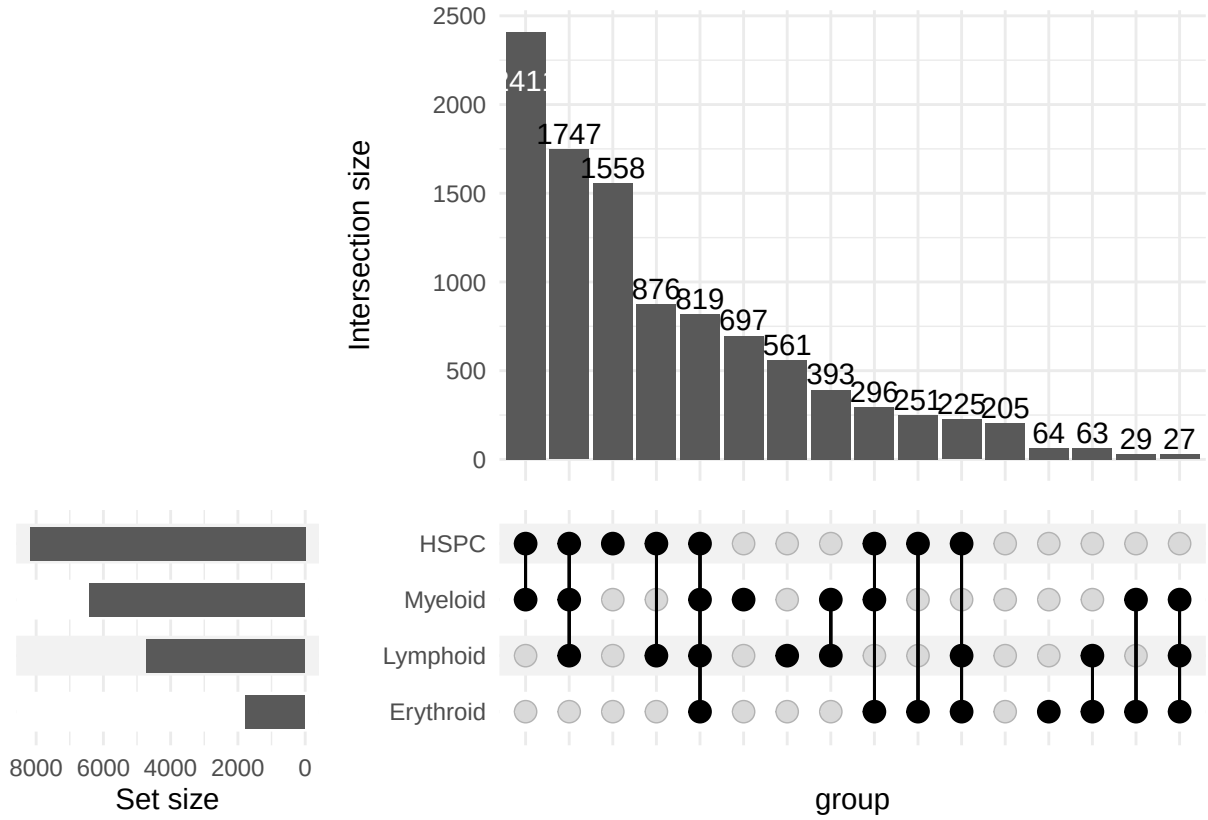
```r
  # If already numeric-ish (e.g., "0"/"1"), keep them
  suppressWarnings(num <- matrix(as.integer(x), nrow = nrow(x), ncol = ncol(x)))
  is_num <- !is.na(num) & x %in% c("0", "1")
  out <- matrix(NA_integer_, nrow = nrow(x), ncol = ncol(x), dimnames = list(rn, cn))
  out[is_num] <- num[is_num]

  # Map remaining via mapping
  map_vec <- as.integer(mapping); names(map_vec) <- names(mapping)
  idx <- is.na(out)
  if (any(idx)) {
    mapped <- map_vec[x[idx]]
    mapped[is.na(mapped)] <- default_value
    out[idx] <- as.integer(mapped)
  }

  if (return_matrix) return(out)
  as.data.frame(out, check.names = FALSE)
}

celltype_matrix <- function(mat, meta, method = "sum") {

  # 1. clean cell names
  cells_format_mat <- sub("_L[0-9]+_", "_", rownames(mat))

  # 2.  mapping
  meta_map <- setNames(meta$maincelltype, meta$CellTag)

  # 3.  cells
  keep_idx <- cells_format_mat %in% meta$CellTag

  mat <- mat[keep_idx, , drop = FALSE]
  cells_use <- cells_format_mat[keep_idx]

  celltype <- meta_map[cells_use]

  # 4. attach celltype
  df <- as.data.frame(mat)
  df$maincelltype <- celltype

  # 5. aggregation
  if (method == "sum") {
    agg <- aggregate(. ~ maincelltype, data = df, FUN = sum)
  } else if (method == "mean") {
    agg <- aggregate(. ~ maincelltype, data = df, FUN = mean)
  } else if (method == "binary") {
    agg <- aggregate(. ~ maincelltype, data = df, FUN = function(x) as.numeric(any(x > 0)))
  } else {
    stop("method must be 'sum', 'mean' or 'binary'")
  }

  rownames(agg) <- agg$maincelltype
  agg$maincelltype <- NULL
```

```
  return(as.matrix(agg))
}

# scRNA
celltypes <- c("HSPC", "Myeloid", "Lymphoid", "Erythroid")
samples_scRNA_csm <- intersect(names(AML_scRNA_CSM_list), unique(AML_scRNA_meta$Lambo_et_al_ID))

mutation_ct_scRNA <- vector("list", length(samples_scRNA_csm))
names(mutation_ct_scRNA) <- samples_scRNA_csm
for (sample in samples_scRNA_csm){
  message(sample)
  csm_01 <- cellxmut_to_01(AML_scRNA_CSM_list[[sample]])
  csm_01_ct <- celltype_matrix(mat = csm_01, meta = AML_scRNA_meta[AML_scRNA_meta$Lambo_et_al_ID == sam
  csm_01_ct <- csm_01_ct[celltypes,]
  csm_01_ct_bin <- ifelse(csm_01_ct >= 3, 1, 0)
  df <- as.data.frame(t(csm_01_ct_bin))
  df <- df %>% rownames_to_column("mutation")
  df[celltypes] <- lapply(df[celltypes], function(x) as.logical(as.integer(x)))
  mutation_ct_scRNA[[sample]] <- df
}
## AML1
## AML2
## AML3
## AML4
## AML5
## AML6
## AML7
## AML8
## AML9
## AML10
## AML11
## AML12
## AML13
## AML14
## AML15
## AML16
## AML17
## AML18
## AML19
## AML20
## AML21
## AML22
## AML23
## AML24
## AML25
## AML26
## AML27
## AML28

# scATA
celltypes <- c("HSPC", "Myeloid", "Lymphoid", "Erythroid")
samples_scATA_csm <- intersect(names(AML_scATA_CSM_list), unique(AML_scATA_meta$SampleID))
mutation_ct_scATA <- vector("list", length(samples_scATA_csm))
```

```r
names(mutation_ct_scATA) <- samples_scATA_csm
for (sample in samples_scATA_csm){
  message(sample)
  csm_01 <- cellxmut_to_01(AML_scATA_CSM_list[[sample]])
  csm_01_ct <- celltype_matrix(mat = csm_01, meta = AML_scATA_meta[AML_scATA_meta$SampleID == sample,])
  csm_01_ct <- csm_01_ct[celltypes,]
  csm_01_ct_bin <- ifelse(csm_01_ct >= 3, 1, 0)
  df <- as.data.frame(t(csm_01_ct_bin))
  df <- df %>% rownames_to_column("mutation")
  df[celltypes] <- lapply(df[celltypes], function(x) as.logical(as.integer(x)))
  mutation_ct_scATA[[sample]] <- df
}
## AML1
## AML2
## AML3
## AML4
## AML5
## AML6
## AML8
## AML10
## AML11
## AML12
## AML13
## AML15
## AML16
## AML17
## AML18
## AML19
## AML20
## AML21
## AML22
## AML24
## AML25
## AML26
## AML27

all_mutation_ct_scRNA <- do.call(rbind, mutation_ct_scRNA)
all_mutation_ct_scATA <- do.call(rbind, mutation_ct_scATA)


ComplexUpset::upset(
  all_mutation_ct_scRNA,
  intersect = celltypes,
  base_annotations = list(
    'Intersection size' = intersection_size()
  )
)
```




```
ComplexUpset::upset(
  all_mutation_ct_scATA,
  intersect = celltypes,
  base_annotations = list(
    'Intersection size' = intersection_size()
  )
)
```




```
p_all_mutation_ct_scRNA <- ComplexUpset::upset(
  all_mutation_ct_scRNA,
  intersect = celltypes,
  base_annotations = list(
    'Intersection size' = intersection_size()
  )
)

p_all_mutation_ct_scATA <- ComplexUpset::upset(
  all_mutation_ct_scATA,
  intersect = celltypes,
  base_annotations = list(
    'Intersection size' = intersection_size()
  )
)

#ggsave("../p_all_mutation_ct_scRNA.pdf", p_all_mutation_ct_scRNA, width = 4.7, height = 2.9)
#ggsave("../p_all_mutation_ct_scATA.pdf", p_all_mutation_ct_scATA, width = 4.7, height = 2.9)
```

9 SNV tree cell type

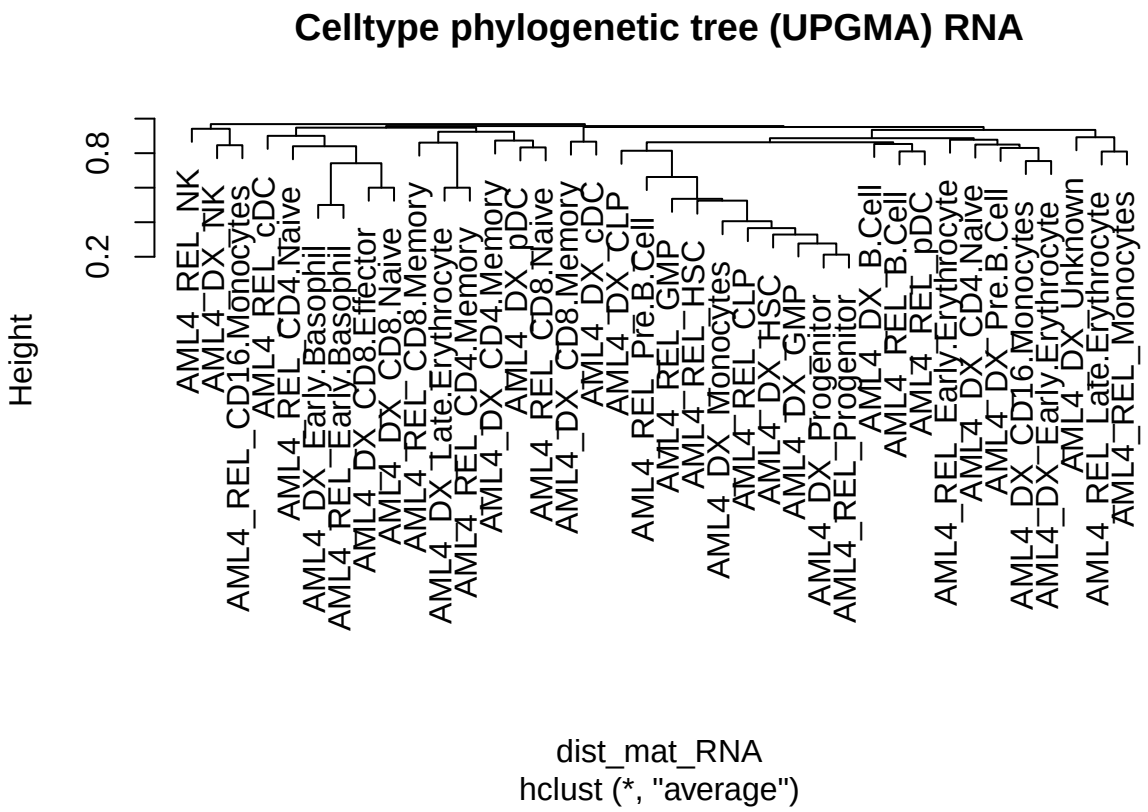
```
merge_cell_mut_matrix <- function(mat1, mat2, fill = 0) {
  all_cells <- union(rownames(mat1), rownames(mat2))
```

```r
  all_muts  <- union(colnames(mat1), colnames(mat2))

  merged_mat <- matrix(fill,
                       nrow = length(all_cells),
                       ncol = length(all_muts),
                       dimnames = list(all_cells, all_muts))

  merged_mat[rownames(mat1), colnames(mat1)] <- mat1

  merged_mat[rownames(mat2), colnames(mat2)] <-
    merged_mat[rownames(mat2), colnames(mat2)] + mat2

  return(merged_mat)
}

cellxmut_to_01 <- function(cellxmut,
                           mapping = c(NoCall = 0L, WT = 0L, Mut = 1L),
                           default_value = NA_integer_,
                           return_matrix = TRUE) {
  if (!(is.matrix(cellxmut) || is.data.frame(cellxmut))) {
    stop("cellxmut must be a matrix or data.frame.")
  }

  rn <- rownames(cellxmut)
  cn <- colnames(cellxmut)
  if (is.null(rn) || any(rn == "")) stop("cellxmut must have non-empty rownames (cell names).")
  if (is.null(cn) || any(cn == "")) stop("cellxmut must have non-empty colnames (mutation sites).")

  # Work in character space, trim whitespace
  x <- as.matrix(cellxmut)
  x <- apply(x, c(1, 2), function(v) trimws(as.character(v)))

  # If already numeric-ish (e.g., "0"/"1"), keep them
  suppressWarnings(num <- matrix(as.integer(x), nrow = nrow(x), ncol = ncol(x)))
  is_num <- !is.na(num) & x %in% c("0", "1")
  out <- matrix(NA_integer_, nrow = nrow(x), ncol = ncol(x), dimnames = list(rn, cn))
  out[is_num] <- num[is_num]

  # Map remaining via mapping
  map_vec <- as.integer(mapping); names(map_vec) <- names(mapping)
  idx <- is.na(out)
  if (any(idx)) {
    mapped <- map_vec[x[idx]]
    mapped[is.na(mapped)] <- default_value
    out[idx] <- as.integer(mapped)
  }

  if (return_matrix) return(out)
  as.data.frame(out, check.names = FALSE)
}

celltype_matrix <- function(mat, meta, method = "sum") {
  common_cells <- intersect(rownames(mat), meta$CellTag)
```

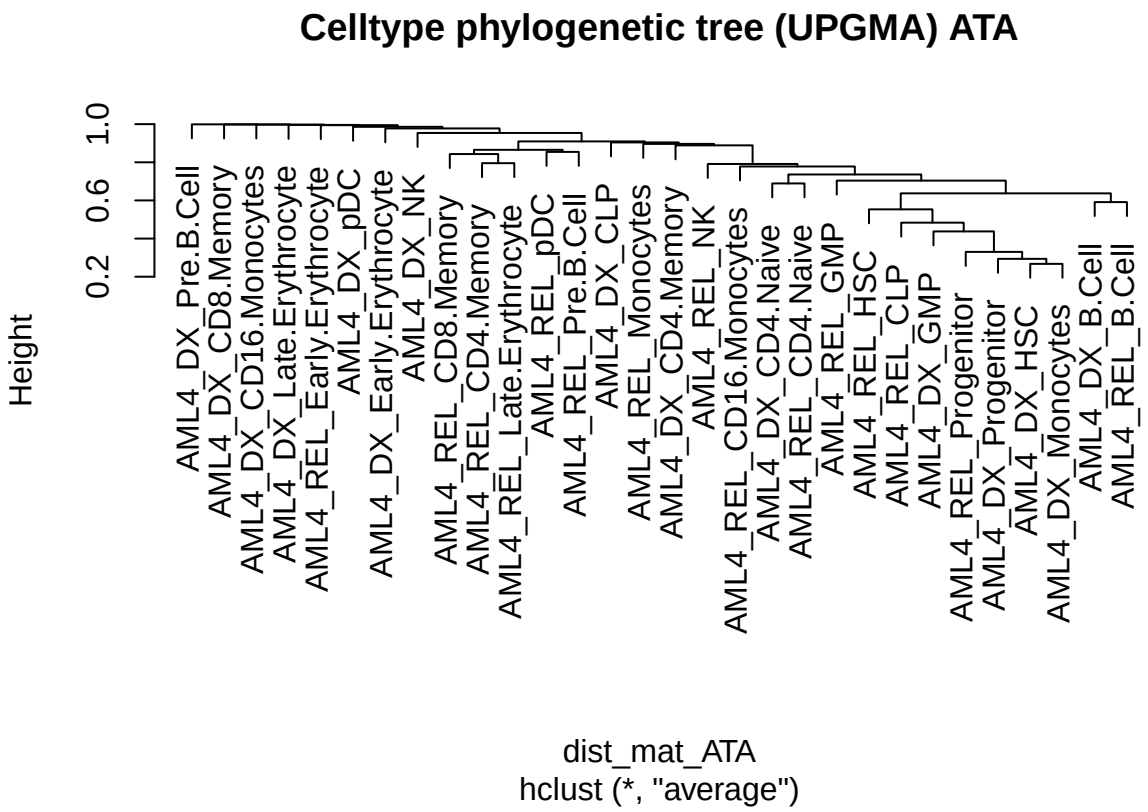
```r
  mat <- mat[common_cells, , drop = FALSE]
  meta_sub <- meta[match(common_cells, meta$CellTag), ]

  df <- as.data.frame(mat)
  df$Classified_CelltypeQ <- meta_sub$Classified_CelltypeQ

  if (method == "sum") {
    agg <- aggregate(. ~ Classified_CelltypeQ, data = df, FUN = sum)
  } else if (method == "mean") {
    agg <- aggregate(. ~ Classified_CelltypeQ, data = df, FUN = mean)
  } else if (method == "binary") {
    agg <- aggregate(. ~ Classified_CelltypeQ, data = df, FUN = function(x) as.numeric(sum(x) > 0))
  } else {
    stop("method must be 'sum', 'mean' or 'binary'")
  }

  rownames(agg) <- agg$Classified_CelltypeQ
  agg$Classified_CelltypeQ <- NULL

  return(as.matrix(agg))
}

celltype_matrix_state <- function(mat, meta) {

  common_cells <- intersect(rownames(mat), meta$CellTag)
  mat <- mat[common_cells, , drop = FALSE]
  ct <- meta$Classified_CelltypeQ[match(common_cells, meta$CellTag)]

  idx_list <- split(seq_along(ct), ct)

  res <- lapply(idx_list, function(idx) {

    sub_mat <- mat[idx, , drop = FALSE]

    apply(sub_mat, 2, function(x) {

      if (any(x == 1, na.rm = TRUE)) {
        return(1L)
      } else if (any(x == 0, na.rm = TRUE)) {
        return(0L)
      } else {
        return(3L)
      }

    })

  })

  res_mat <- do.call(rbind, res)

  return(res_mat)
}
```

```r
AML04_CSM_scRNA_DX <- cellxmut_to_01(cellxmut = AML_scRNA_CSM_DX_REL_select_list$AML4_DX, mapping = c(N
AML04_CSM_scRNA_REL <- cellxmut_to_01(cellxmut = AML_scRNA_CSM_DX_REL_select_list$AML4_REL, mapping = c

AML04_CSM_scATA_DX <- cellxmut_to_01(cellxmut = AML_scATA_CSM_DX_REL_select_list$AML4_DX, mapping = c(N
AML04_CSM_scATA_REL <- cellxmut_to_01(cellxmut = AML_scATA_CSM_DX_REL_select_list$AML4_REL, mapping = c

AML16_CSM_scATA_DX <- cellxmut_to_01(AML_scATA_CSM_DX_REL_select_list$AML16_DX, default_value = 0L)
AML16_CSM_scATA_REL <- cellxmut_to_01(AML_scATA_CSM_DX_REL_select_list$AML16_REL, default_value = 0L)

AML18_CSM_scATA_DX <- cellxmut_to_01(AML_scATA_CSM_DX_REL_select_list$AML18_DX, default_value = 0L)
AML18_CSM_scATA_REL <- cellxmut_to_01(AML_scATA_CSM_DX_REL_select_list$AML18_REL, default_value = 0L)

AML04_CSM_scRNA <- merge_cell_mut_matrix(AML04_CSM_scRNA_DX,
                                         AML04_CSM_scRNA_REL)

AML04_CSM_scATA <- merge_cell_mut_matrix(AML04_CSM_scATA_DX,
                                         AML04_CSM_scATA_REL)

AML16_CSM_scATA <- merge_cell_mut_matrix(AML16_CSM_scATA_DX,
                                         AML16_CSM_scATA_REL)

AML18_CSM_scATA <- merge_cell_mut_matrix(AML18_CSM_scATA_DX,
                                         AML18_CSM_scATA_REL)

AML04_CSM_scRNA_ct <- celltype_matrix(mat = AML04_CSM_scRNA, meta = AML_scRNA_meta_DX_REL)
AML04_CSM_scRNA_ct_state <- celltype_matrix_state(AML04_CSM_scRNA, AML_scRNA_meta_DX_REL)
AML04_CSM_scATA_ct <- celltype_matrix(mat = AML04_CSM_scATA, meta = AML_scATA_meta_DX_REL)
AML04_CSM_scATA_ct_state <- celltype_matrix_state(AML04_CSM_scATA, AML_scATA_meta_DX_REL)
AML16_CSM_scATA_ct <- celltype_matrix(mat = AML16_CSM_scATA, meta = AML_scATA_meta_DX_REL)
AML18_CSM_scATA_ct <- celltype_matrix(mat = AML18_CSM_scATA, meta = AML_scATA_meta_DX_REL)

#write.table(x = t(AML04_CSM_scRNA_ct_state), "/Users/liqun/Downloads/AML04_CSM_scRNA_ct_state.txt", se
#write.table(x = t(AML04_CSM_scATA_ct_state), "/Users/liqun/Downloads/AML04_CSM_scATA_ct_state.txt", se

binary_mat_RNA <- (AML04_CSM_scRNA_ct != 0) * 1
binary_mat_ATA <- (AML04_CSM_scATA_ct != 0) * 1
library(vegan)
library(ape)
##
## Attaching package: 'ape'
## The following object is masked from 'package:circlize':
##
##     degree
## The following object is masked from 'package:ggpubr':
##
##     rotate
## The following object is masked from 'package:dplyr':
##
##     where
dist_mat_RNA <- vegdist(binary_mat_RNA, method = "jaccard")
dist_mat_ATA <- vegdist(binary_mat_ATA, method = "jaccard")
hc_RNA <- hclust(dist_mat_RNA, method = "average")  # average = UPGMA
hc_ATA <- hclust(dist_mat_ATA, method = "average")
```

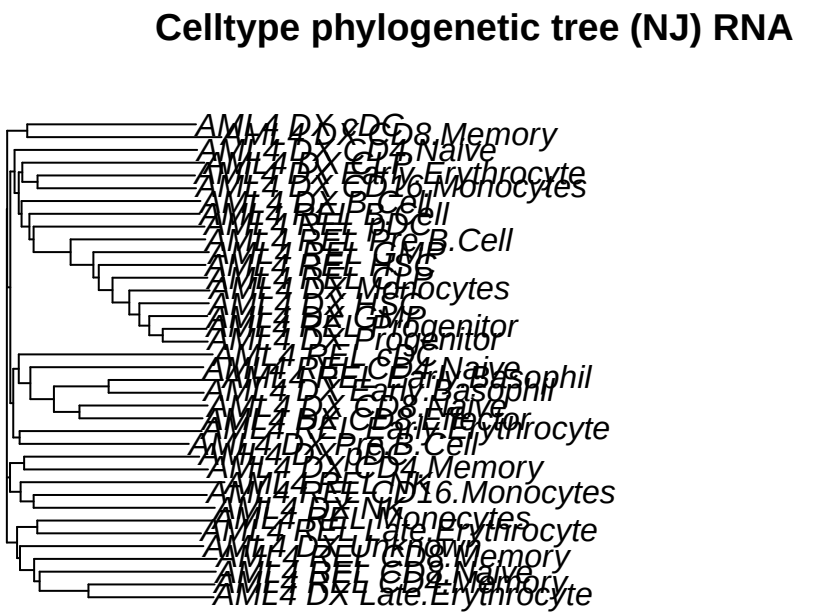
```r
plot(hc_RNA, main = "Celltype phylogenetic tree (UPGMA) RNA")
```



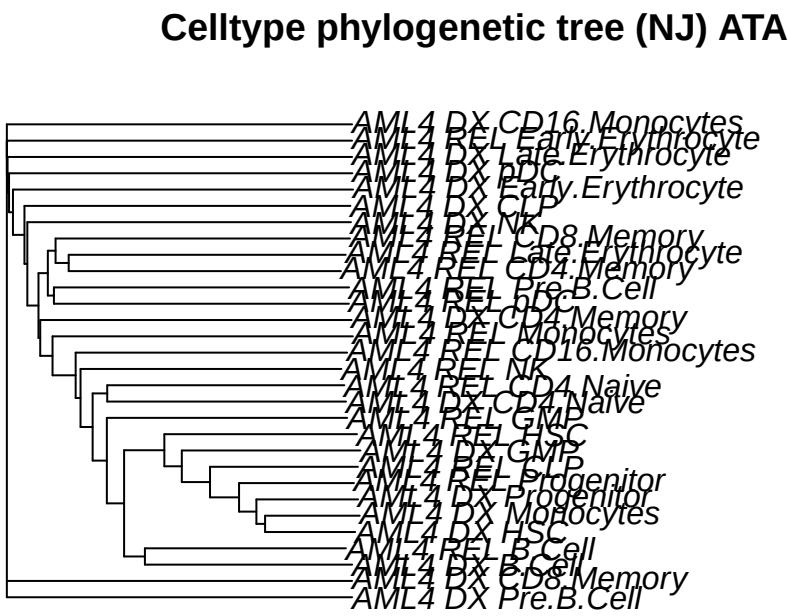

```r
plot(hc_ATA, main = "Celltype phylogenetic tree (UPGMA) ATA")
```




```
nj_tree_RNA <- nj(dist_mat_RNA)
nj_tree_ATA <- nj(dist_mat_ATA)
plot(nj_tree_RNA, main = "Celltype phylogenetic tree (NJ) RNA")
```




```
plot(nj_tree_ATA, main = "Celltype phylogenetic tree (NJ) ATA")
```



10 mutratio

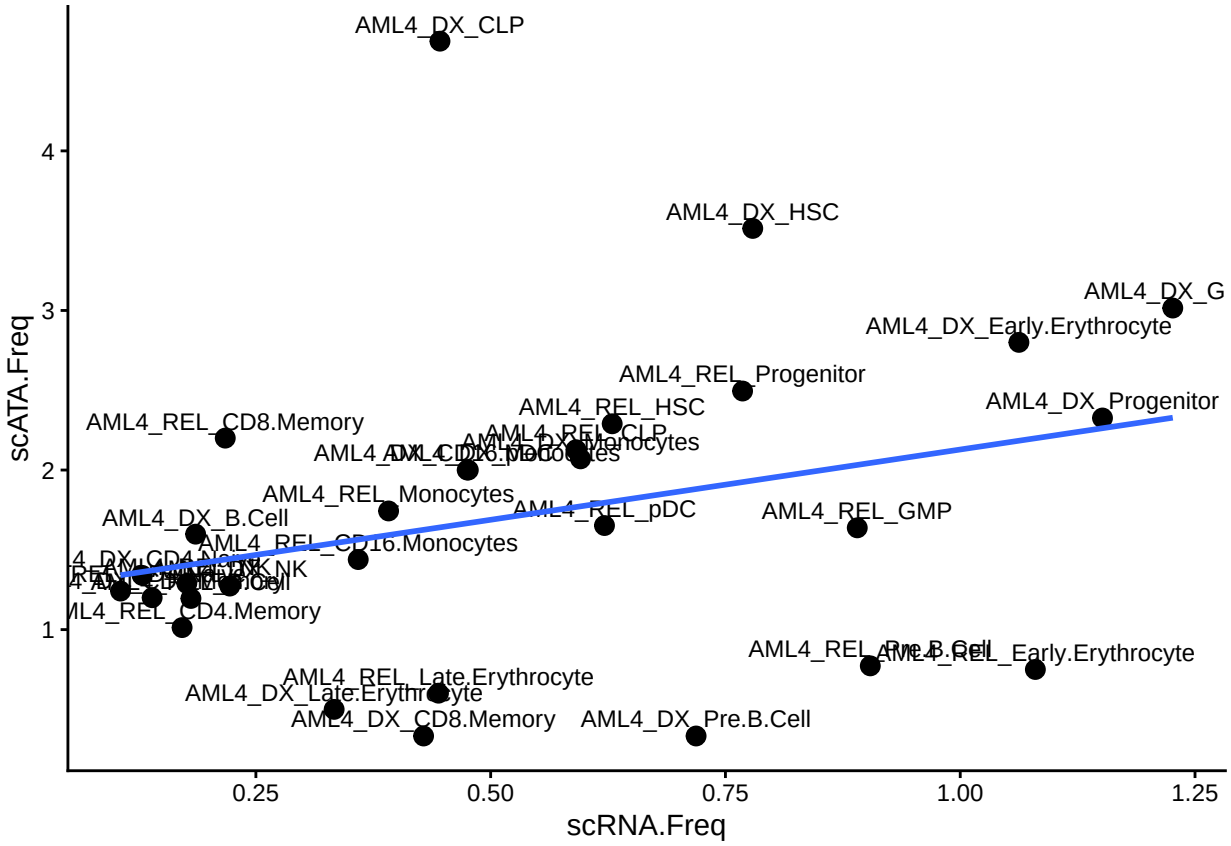
```
ct_counts_scRNA <- table(
  AML_scRNA_meta_DX_REL$Classified_CelltypeQ[
    AML_scRNA_meta_DX_REL$SampleID == "AML4"
  ]
)

ct_counts_scATA <- table(
  AML_scATA_meta_DX_REL$Classified_CelltypeQ[
    AML_scATA_meta_DX_REL$SampleID == "AML4"
  ]
)
# 2. mutation counts rowSums
mut_counts_scRNA <- rowSums(AML04_CSM_scRNA_ct)
mut_counts_scATA <- rowSums(AML04_CSM_scATA_ct)

# 3. map
common_ct_scRNA <- intersect(names(ct_counts_scRNA), names(mut_counts_scRNA))
ct_counts_scRNA <- ct_counts_scRNA[common_ct_scRNA]
mut_counts_scRNA <- mut_counts_scRNA[common_ct_scRNA]
common_ct_scATA <- intersect(names(ct_counts_scATA), names(mut_counts_scATA))
ct_counts_scATA <- ct_counts_scATA[common_ct_scATA]
```

```r
mut_counts_scATA <- mut_counts_scATA[common_ct_scATA]

# 4. ratio
ratio_scRNA <- mut_counts_scRNA / ct_counts_scRNA
ratio_scATA <- mut_counts_scATA / ct_counts_scATA

# 5. sort
ratio_sorted_scRNA <- sort(ratio_scRNA)
ratio_sorted_scATA <- sort(ratio_scATA)

ratio_sorted_scRNA
##
##         AML4_REL_CD4.Naive          AML4_DX_CD4.Naive
##                  0.1058824                  0.1288889
##         AML4_DX_CD4.Memory         AML4_REL_CD8.Naive
##                  0.1395349                  0.1481481
##        AML4_REL_CD4.Memory                AML4_REL_NK
##                  0.1714286                  0.1764706
##            AML4_REL_B.Cell             AML4_DX_B.Cell
##                  0.1808511                  0.1858407
##          AML4_DX_CD8.Naive        AML4_REL_CD8.Memory
##                  0.2000000                  0.2173913
##                 AML4_DX_NK       AML4_DX_CD8.Effector
##                  0.2222222                  0.3333333
##   AML4_DX_Late.Erythrocyte    AML4_REL_CD16.Monocytes
##                  0.3333333                  0.3589744
##         AML4_REL_Monocytes         AML4_DX_CD8.Memory
##                  0.3913043                  0.4285714
##  AML4_REL_Late.Erythrocyte                AML4_DX_CLP
##                  0.4444444                  0.4460432
##     AML4_DX_CD16.Monocytes                AML4_DX_pDC
##                  0.4750000                  0.4761905
##    AML4_REL_Early.Basophil                AML4_DX_cDC
##                  0.5000000                  0.5769231
##               AML4_REL_CLP          AML4_DX_Monocytes
##                  0.5908714                  0.5957132
##               AML4_REL_pDC               AML4_REL_HSC
##                  0.6212121                  0.6294227
##     AML4_DX_Early.Basophil         AML4_DX_Pre.B.Cell
##                  0.6666667                  0.7187500
##        AML4_REL_Progenitor                AML4_DX_HSC
##                  0.7682453                  0.7790698
##               AML4_REL_cDC               AML4_REL_GMP
##                  0.8000000                  0.8903743
##        AML4_REL_Pre.B.Cell  AML4_DX_Early.Erythrocyte
##                  0.9042553                  1.0625000
## AML4_REL_Early.Erythrocyte         AML4_DX_Progenitor
##                  1.0800000                  1.1514841
##                AML4_DX_GMP            AML4_DX_Unknown
##                  1.2263223                  1.8571429
ratio_sorted_scATA
##
##         AML4_DX_CD8.Memory         AML4_DX_Pre.B.Cell
```

```
##                  0.3333333                  0.3333333
##   AML4_DX_Late.Erythrocyte  AML4_REL_Late.Erythrocyte
##                  0.5000000                  0.6016260
## AML4_REL_Early.Erythrocyte        AML4_REL_Pre.B.Cell
##                  0.7500000                  0.7727273
##        AML4_REL_CD4.Memory            AML4_REL_B.Cell
##                  1.0126582                  1.1952191
##         AML4_DX_CD4.Memory         AML4_REL_CD4.Naive
##                  1.2000000                  1.2413793
##                 AML4_DX_NK                AML4_REL_NK
##                  1.2727273                  1.2893082
##          AML4_DX_CD4.Naive    AML4_REL_CD16.Monocytes
##                  1.3400000                  1.4388889
##             AML4_DX_B.Cell               AML4_REL_GMP
##                  1.5982143                  1.6376812
##               AML4_REL_pDC         AML4_REL_Monocytes
##                  1.6521739                  1.7435897
##     AML4_DX_CD16.Monocytes                AML4_DX_pDC
##                  2.0000000                  2.0000000
##          AML4_DX_Monocytes               AML4_REL_CLP
##                  2.0700863                  2.1274074
##        AML4_REL_CD8.Memory               AML4_REL_HSC
##                  2.2000000                  2.2897727
##         AML4_DX_Progenitor        AML4_REL_Progenitor
##                  2.3260573                  2.4950547
##  AML4_DX_Early.Erythrocyte                AML4_DX_GMP
##                  2.8000000                  3.0151515
##                AML4_DX_HSC                AML4_DX_CLP
##                  3.5140845                  4.6875000
```

```r
common_ct <- intersect(names(ratio_scRNA), names(ratio_scATA))

ratio_scRNA2 <- ratio_scRNA[common_ct]
ratio_scATA2 <- ratio_scATA[common_ct]
cor(ratio_scRNA2, ratio_scATA2, method = "spearman")
## [1] 0.4025367

df_cor <- data.frame(
  celltype = common_ct,
  scRNA = ratio_scRNA2,
  scATA = ratio_scATA2
)

ggplot(df_cor, aes(scRNA.Freq, scATA.Freq, label = celltype)) +
  geom_point(size = 3) +
  geom_text(size = 3, vjust = -0.5) +
  geom_smooth(method = "lm", se = FALSE) +
  theme_classic()
## `geom_smooth()` using formula = 'y ~ x'
## Warning: The following aesthetics were dropped during statistical transformation: label.
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
```

```
##   variable into a factor?
```



11 sessionInfo

```
sessionInfo()
## R version 4.5.1 (2025-06-13)
## Platform: aarch64-apple-darwin20
## Running under: macOS Tahoe 26.4.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRlapack.dylib;  LAPACK v
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: Europe/Stockholm
## tzcode source: internal
##
## attached base packages:
## [1] stats4    grid      stats     graphics  grDevices utils     datasets
## [8] methods   base
##
```

```
## other attached packages:
##  [1] ape_5.8-1                    htmlwidgets_1.6.4
##  [3] networkD3_0.4.1              clusterProfiler_4.16.0
##  [5] ComplexHeatmap_2.24.1        org.Hs.eg.db_3.21.0
##  [7] RColorBrewer_1.1-3           ggsankey_0.0.99999
##  [9] ComplexUpset_1.3.6           UpSetR_1.4.0
## [11] patchwork_1.3.2              lubridate_1.9.4
## [13] forcats_1.0.0                purrr_1.1.0
## [15] readr_2.1.5                  tibble_3.3.0
## [17] tidyverse_2.0.0              bedtoolsr_2.30.0-7
## [19] corrplot_0.95                pheatmap_1.0.13
## [21] circlize_0.4.16              ggpmisc_0.6.2
## [23] ggpp_0.5.9                   Seurat_5.3.0
## [25] SeuratObject_5.2.0           sp_2.2-0
## [27] ggpubr_0.6.1                 tidyr_1.3.1
## [29] GO.db_3.21.0                 AnnotationDbi_1.70.0
## [31] dplyr_1.1.4                  rhdf5_2.52.1
## [33] SummarizedExperiment_1.38.1 Biobase_2.68.0
## [35] RcppArmadillo_15.0.2-2       Rcpp_1.1.0
## [37] Matrix_1.7-4                 GenomicRanges_1.60.0
## [39] GenomeInfoDb_1.44.3          IRanges_2.42.0
## [41] S4Vectors_0.46.0             BiocGenerics_0.54.0
## [43] generics_0.1.4               sparseMatrixStats_1.20.0
## [45] MatrixGenerics_1.20.0        matrixStats_1.5.0
## [47] data.table_1.17.8            stringr_1.5.2
## [49] plyr_1.8.9                   magrittr_2.0.4
## [51] ggplot2_4.0.0                gtable_0.3.6
## [53] gtools_3.9.5                 gridExtra_2.3
## [55] devtools_2.4.5               usethis_3.2.1
## [57] ArchR_1.0.3                  vegan_2.7-1
## [59] permute_0.9-8                ineq_0.2-13
## [61] FSA_0.10.0
##
## loaded via a namespace (and not attached):
##   [1] fs_1.6.6                 spatstat.sparse_3.1-0   enrichplot_1.28.4
##   [4] doParallel_1.0.17        httr_1.4.7              profvis_0.4.0
##   [7] tools_4.5.1              sctransform_0.4.2       backports_1.5.0
##  [10] R6_2.6.1                 lazyeval_0.2.2          uwot_0.2.3
##  [13] mgcv_1.9-3               GetoptLong_1.0.5        rhdf5filters_1.20.0
##  [16] urlchecker_1.0.1         withr_3.0.2             progressr_0.16.0
##  [19] quantreg_6.1             cli_3.6.5               Cairo_1.6-5
##  [22] spatstat.explore_3.5-3   fastDummies_1.7.5       labeling_0.4.3
##  [25] S7_0.2.0                 spatstat.data_3.1-8     ggridges_0.5.7
##  [28] pbapply_1.7-4            yulab.utils_0.2.1       gson_0.1.0
##  [31] R.utils_2.13.0           DOSE_4.2.0              dichromat_2.0-0.1
##  [34] parallelly_1.45.1        sessioninfo_1.2.3       rstudioapi_0.17.1
##  [37] RSQLite_2.4.3            gridGraphics_0.5-1      shape_1.4.6.1
##  [40] ica_1.0-3                spatstat.random_3.4-2   car_3.1-3
##  [43] abind_1.4-8              R.methodsS3_1.8.2       lifecycle_1.0.4
##  [46] yaml_2.3.10              carData_3.0-5           qvalue_2.40.0
##  [49] SparseArray_1.8.1        Rtsne_0.17              blob_1.2.4
##  [52] promises_1.3.3           crayon_1.5.3            ggtangle_0.0.7
##  [55] miniUI_0.1.2             lattice_0.22-7          cowplot_1.2.0
```

```
##  [58] KEGGREST_1.48.1         pillar_1.11.1           knitr_1.50
##  [61] fgsea_1.34.2            rjson_0.2.23            future.apply_1.20.0
##  [64] codetools_0.2-20        fastmatch_1.1-6         glue_1.8.0
##  [67] ggfun_0.2.0             spatstat.univar_3.1-4   remotes_2.5.0
##  [70] treeio_1.32.0           vctrs_0.6.5             png_0.1-8
##  [73] spam_2.11-1             cachem_1.1.0            xfun_0.53
##  [76] S4Arrays_1.8.1          mime_0.13               survival_3.8-3
##  [79] iterators_1.0.14        ellipsis_0.3.2          fitdistrplus_1.2-4
##  [82] ROCR_1.0-11             nlme_3.1-168            ggtree_3.16.3
##  [85] bit64_4.6.0-1           RcppAnnoy_0.0.22        data.tree_1.2.0
##  [88] irlba_2.3.5.1           KernSmooth_2.23-26      colorspace_2.1-2
##  [91] DBI_1.2.3               tidyselect_1.2.1        bit_4.6.0
##  [94] compiler_4.5.1          SparseM_1.84-2          DelayedArray_0.34.1
##  [97] plotly_4.11.0           scales_1.4.0            lmtest_0.9-40
## [100] rappdirs_0.3.3          digest_0.6.37           goftest_1.2-3
## [103] spatstat.utils_3.2-0    rmarkdown_2.30          XVector_0.48.0
## [106] htmltools_0.5.8.1       pkgconfig_2.0.3         fastmap_1.2.0
## [109] rlang_1.1.6             GlobalOptions_0.1.2     UCSC.utils_1.4.0
## [112] shiny_1.11.1            farver_2.1.2            zoo_1.8-14
## [115] jsonlite_2.0.0          BiocParallel_1.42.2     R.oo_1.27.1
## [118] GOSemSim_2.34.0         ggplotify_0.1.3         polynom_1.4-1
## [121] Formula_1.2-5           GenomeInfoDbData_1.2.14 dotCall64_1.2
## [124] Rhdf5lib_1.30.0         reticulate_1.43.0       stringi_1.8.7
## [127] MASS_7.3-65             pkgbuild_1.4.8          parallel_4.5.1
## [130] listenv_0.9.1           ggrepel_0.9.6           deldir_2.0-4
## [133] Biostrings_2.76.0       splines_4.5.1           tensor_1.5.1
## [136] hms_1.1.3               igraph_2.1.4            spatstat.geom_3.6-0
## [139] ggsignif_0.6.4          RcppHNSW_0.6.0          reshape2_1.4.4
## [142] pkgload_1.4.0           evaluate_1.0.5          foreach_1.5.2
## [145] tzdb_0.5.0              httpuv_1.6.16           MatrixModels_0.5-4
## [148] RANN_2.6.2              polyclip_1.10-7         clue_0.3-66
## [151] future_1.67.0           scattermore_1.2         broom_1.0.10
## [154] xtable_1.8-4            tidytree_0.4.6          RSpectra_0.16-2
## [157] rstatix_0.7.2           later_1.4.4             viridisLite_0.4.2
## [160] aplot_0.2.9             memoise_2.0.1           cluster_2.1.8.1
## [163] timechange_0.3.0        globals_0.18.0
```