Scenic

Abinet Joan

2023-09-29 15:41:05 +0200

Contents

Loading packages **1**

Loading data **1**

SCENIC **2**
- Prepare cellinfo and expression matrix . . . 2
- Download binding motifs for mouse . . . 2
- Initialise settings . . . 2
- Gene filtering . . . 2
- Correlation . . . 2
- Genie3 . . . 3
- Build and score the GRN . . . 3

Plotting the results **3**
- Viewing markers based on specificity score . . . 3
- Plot on Heatmap the 10 markers the most specific of each cell type . . . 3
- plotting Mafb and c-Maf . . . 4

Loading packages

```
suppressMessages(library(dplyr))
suppressMessages(library(Seurat))
suppressMessages(library(patchwork))
suppressMessages(library(ggplot2))
suppressMessages(library(SCENIC))
suppressMessages(library(ComplexHeatmap))
suppressMessages(library(AUCell))
```

Loading data

```
myeloid_cells_clustered <-
readRDS("../3-Visualisation_Clustering/Myeloid_cells_Final.rds")
```

SCENIC

Prepare cellinfo and expression matrix

```r
exprMat <- myeloid_cells_clustered@assays$RNA@data
cellInfo <- data.frame(seuratCluster=Idents(myeloid_cells_clustered))
cellInfo$nGene <- colSums(exprMat>0)

dir.create ("int")
#saveRDS(exprMat, file = "int/exprMat.Rds")
#saveRDS(cellInfo, file = "int/cellInfo.Rds")
```

Download binding motifs for mouse

```r
dbFiles <-
c("https://resources.aertslab.org/cistarget/databases/old/mus_musculus/mm9/refseq_r45/mc9nr/gene_based/m
"https://resources.aertslab.org/cistarget/databases/old/mus_musculus/mm9/refseq_r45/mc9nr/gene_based/mm9

dir.create("cisTarget_databases");
setwd("cisTarget_databases")
for(featherURL in dbFiles)
{
  download.file(featherURL, destfile=basename(featherURL)) # saved in current dir
}
```

Initialise settings

```r
org <- "mgi"
dbDir <- "cisTarget_databases"
dbDir <- path.expand(dbDir)
myDatasetTitle <- "SCENIC Analysis"
data(defaultDbNames)
dbs <- defaultDbNames[[org]]
scenicOptions <- initializeScenic(org=org, dbDir=dbDir, dbs=dbs, nCores=25)
```

Gene filtering

1. Filter by the total number of reads per gene. Keeps only the genes with at least 6 UMI counts across all samples.
2. Filter by the number of cells in which the gene is detected.

```r
exprMat <- as.matrix(exprMat) # need a regulat matrix
genesKept <- geneFiltering(exprMat, scenicOptions=scenicOptions,
                           minCountsPerGene=3*.01*ncol(exprMat),
                           minSamples=ncol(exprMat)*.01)

exprMat_filtered <- exprMat[genesKept, ]
```

Correlation

```
runCorrelation(exprMat_filtered, scenicOptions)
```

Genie3

This step is time consuming

```
runGenie3(exprMat, scenicOptions)
```

Build and score the GRN

```
runGenie3(exprMat_filtered, scenicOptions)

scenicOptions@settings$verbose <- TRUE
scenicOptions@settings$nCores <- 10
scenicOptions@settings$seed <- 123

runSCENIC_1_coexNetwork2modules(scenicOptions)
runSCENIC_2_createRegulons(scenicOptions)
runSCENIC_3_scoreCells(scenicOptions, exprMat_filtered)
```

Plotting the results

Viewing markers based on specificity score

```
regulonAUC <- loadInt(scenicOptions, "aucell_regulonAUC") # require file
int/3.4_regulonAUC.Rds


rss <- calcRSS(AUC=getAUC(regulonAUC), cellAnnotation=cellInfo[colnames(regulonAUC),
"seuratCluster"])
#rssPlot <- plotRSS(rss["Mafb (44g)",])
rssPlot <- plotRSS(rss)
plotly::ggplotly(rssPlot$plot)
```

```
plotly::ggplotly(rssPlot$plot)
```

Plot on Heatmap the 10 markers the most specific of each cell type

```
rss <- rss[,c(8,11,2,3,7,1,12,9,4,6,5,10)]

top_tf <- c()
for (i in 1:length(colnames(rss))) {

  TFs <- sort(rss[,i], decreasing = T)
  top_10 <- head(TFs, 10)
  top_tf <- c(top_tf,top_10)
}

length(top_tf)
```

```
top_tf

# Creation of the Heatmap

regulonAUC.mat <- regulonAUC@assays@data@listData$AUC
Subset_regulonActivity <-regulonAUC.mat[names(top_tf),]

regulonActivity_byCellType_Scaled <- t(scale(t(Subset_regulonActivity), center = T,
scale=T))

colors <- c("#B2DF8A","#ABD61C","#1F78B4","#A6CEE3", "#E31A1C","#E3751C","#600078",
"#33A02C", "#FDBF6F", "#526317",  "#D4AAC6", "#784620")

colors <- c("#B2DF8A","#ABD61C","#1F78B4","#A6CEE3", "#E31A1C","#E3751C","#600078",
"#33A02C", "#FDBF6F", "#526317",  "#D4AAC6", "#784620")
names(colors) <- levels(myeloid_cells_clustered$CellType)

df <- as.data.frame(myeloid_cells_clustered$CellType)
colnames(df) <- "CellType"

color_df <- list(CellType =
                   c("CD206- IM" = "#B2DF8A",
                     "Ly6G Macs" = "#ABD61C",
                     "Ly6C- Mo" = "#1F78B4",
                     "Ly6C+ Mo" = "#A6CEE3",
                     "CD64+ cMo" = "#E31A1C",
                     "AM" = "#E3751C",
                     "Neutrophils" = "#600078",
                     "CD206+ IM" = "#33A02C",
                     "DCs" = "#FDBF6F",
                     "Dying cells" = "#526317",
                     "Cycling Macs" = "#D4AAC6",
                     "IAV-specific AM" = "#784620"))

png(file="/home/qiang/Documents/Travail_cell/Travail_V3/Cécilia_project_2/4Scenic/Heatmap.png",
width = 2000, height = 2500)

Heatmap(Subset_regulonActivity[], name="Regulon activity", show_column_names = FALSE,
        column_split = factor(myeloid_cells_clustered$CellType),
        cluster_column_slices = F,
        cluster_rows = F,
        bottom_annotation = HeatmapAnnotation(df = df, col = color_df))
dev.off()
```

plotting Mafb and c-Maf

```
TF <- c("Mafb (44g)", "Maf (220g)")

df <- as.data.frame(myeloid_cells_clustered$CellType)
colnames(df) <- "Clusters"

color_df <- list(Clusters =
                   c("CD206 IM" = "#B2DF8A",
                     "PR8 Mac1" = "#ABD61C",
                     "pMo" = "#1F78B4",
                     "cMo" = "#A6CEE3",
                     "CD64+ cMo" = "#E31A1C",
                     "AM" = "#E3751C",
                     "Neutro" = "#600078",
                     "CD206+ IM" = "#33A02C",
                     "DCs" = "#FDBF6F",
                     "PR8 Mac2" = "#526317",
                     "Cycling Mac" = "#D4AAC6",
                     "PR8 AM" = "#784620"))

#pdf(file="/home/qiang/Documents/Travail_cell/Travail_V3/Cécilia_project_2/4Scenic/Heatmap_cMafMafbstat
width = 2000, height = 1000)
heatmap_maf <- Heatmap(regulonAUC.mat[TF,], name="Regulon activity", show_column_names =
FALSE,
        column_split = factor(myeloid_cells_clustered$CellType),
        cluster_column_slices = F,
        cluster_rows = F,
        bottom_annotation = HeatmapAnnotation(df = df, col = color_df))
#dev.off()

#setequal(levels(df), names(color_df$test))
```

saving heatmap as pdf

```
tidyHeatmap::save_pdf(heatmap_maf,
"/home/qiang/Documents/Travail_cell/Travail_V3/Cécilia_project_2/4Scenic/Heatmap_cMafMafbstat2.pdf",
width = 30, height = 10, units = "cm")
```

```
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 22.04.3 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.20.so;  LAPACK version 3.10.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=fr_BE.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=fr_BE.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=fr_BE.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=fr_BE.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Brussels
## tzcode source: system (glibc)
##
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
## [1] AUCell_1.22.0         ComplexHeatmap_2.16.0 SCENIC_1.3.1
## [4] ggplot2_3.4.2         patchwork_1.1.2       SeuratObject_4.1.3
## [7] Seurat_4.3.0          dplyr_1.1.2
##
## loaded via a namespace (and not attached):
##   [1] RcppAnnoy_0.0.21            splines_4.3.1
##   [3] later_1.3.1                 bitops_1.0-7
##   [5] tibble_3.2.1                R.oo_1.25.0
##   [7] polyclip_1.10-4             graph_1.78.0
##   [9] XML_3.99-0.14               lifecycle_1.0.3
##  [11] doParallel_1.0.17           globals_0.16.2
##  [13] lattice_0.21-8              MASS_7.3-60
##  [15] magrittr_2.0.3              plotly_4.10.2
##  [17] rmarkdown_2.23              yaml_2.3.7
##  [19] httpuv_1.6.11               sctransform_0.3.5
##  [21] spam_2.9-1                  sp_2.0-0
##  [23] spatstat.sparse_3.0-2       reticulate_1.30
##  [25] cowplot_1.1.1               pbapply_1.7-2
##  [27] DBI_1.1.3                   RColorBrewer_1.1-3
##  [29] abind_1.4-5                 zlibbioc_1.46.0
##  [31] Rtsne_0.16                  GenomicRanges_1.52.0
##  [33] purrr_1.0.1                 R.utils_2.12.2
##  [35] BiocGenerics_0.46.0         RCurl_1.98-1.12
##  [37] circlize_0.4.15             GenomeInfoDbData_1.2.10
##  [39] IRanges_2.34.0              S4Vectors_0.38.1
##  [41] ggrepel_0.9.3               irlba_2.3.5.1
##  [43] listenv_0.9.0               spatstat.utils_3.0-3
##  [45] goftest_1.2-3               spatstat.random_3.1-5
##  [47] annotate_1.78.0             fitdistrplus_1.1-11
##  [49] parallelly_1.36.0           DelayedMatrixStats_1.22.1
##  [51] leiden_0.4.3                codetools_0.2-19
##  [53] DelayedArray_0.26.3         shape_1.4.6
##  [55] tidyselect_1.2.0            matrixStats_1.0.0
##  [57] stats4_4.3.1                spatstat.explore_3.2-1
##  [59] jsonlite_1.8.7              GetoptLong_1.0.5
##  [61] ellipsis_0.3.2              progressr_0.13.0
##  [63] iterators_1.0.14            ggridges_0.5.4
##  [65] survival_3.5-5              foreach_1.5.2
##  [67] tools_4.3.1                 ica_1.0-3
##  [69] Rcpp_1.0.11                 glue_1.6.2
##  [71] gridExtra_2.3               xfun_0.39
##  [73] MatrixGenerics_1.12.2       GenomeInfoDb_1.36.0
##  [75] withr_2.5.0                 fastmap_1.1.1
##  [77] fansi_1.0.4                 digest_0.6.33
##  [79] R6_2.5.1                    mime_0.12
##  [81] colorspace_2.1-0            scattermore_1.2
##  [83] tensor_1.5                  spatstat.data_3.0-1
##  [85] RSQLite_2.3.1               R.methodsS3_1.8.2
##  [87] utf8_1.2.3                  tidyr_1.3.0
##  [89] generics_0.1.3              data.table_1.14.8
##  [91] httr_1.4.6                  htmlwidgets_1.6.2
##  [93] S4Arrays_1.0.4              uwot_0.1.16
##  [95] pkgconfig_2.0.3             gtable_0.3.3
##  [97] blob_1.2.4                  lmtest_0.9-40
##  [99] XVector_0.40.0              htmltools_0.5.5
## [101] dotCall64_1.0-2             clue_0.3-64
## [103] GSEABase_1.62.0             scales_1.2.1
## [105] Biobase_2.60.0              png_0.1-8
## [107] knitr_1.43                  rstudioapi_0.14
## [109] rjson_0.2.21                reshape2_1.4.4
## [111] nlme_3.1-162                GlobalOptions_0.1.2
## [113] zoo_1.8-12                  cachem_1.0.8
## [115] stringr_1.5.0               KernSmooth_2.23-22
## [117] parallel_4.3.1              miniUI_0.1.1.1
## [119] AnnotationDbi_1.62.1        pillar_1.9.0
## [121] vctrs_0.6.3                 RANN_2.6.1
## [123] promises_1.2.0.1            xtable_1.8-4
## [125] cluster_2.1.4               evaluate_0.21
## [127] cli_3.6.1                   compiler_4.3.1
## [129] rlang_1.1.1                 crayon_1.5.2
## [131] future.apply_1.11.0         plyr_1.8.8
## [133] stringi_1.7.12              viridisLite_0.4.2
## [135] deldir_1.0-9                munsell_0.5.0
## [137] Biostrings_2.68.1           lazyeval_0.2.2
## [139] spatstat.geom_3.2-4         Matrix_1.6-0
## [141] sparseMatrixStats_1.12.0    bit64_4.0.5
## [143] future_1.33.0               KEGGREST_1.40.0
## [145] shiny_1.7.4.1               SummarizedExperiment_1.30.2
## [147] ROCR_1.0-11                 igraph_1.5.0.1
## [149] memoise_2.0.1               bit_4.0.5
```