prostate cancer archr tutorial

Xiaosai Yao

16 November 2022

Package

epiregulon 1.0.18

Contents

1 Introduction 2

2 Installation 2

3 Data preparation 2

3.1 Load ArchR project 2

3.2 Retrieve matrices from ArchR project 5

4 Quick start 7

4.1 Retrieve bulk TF ChIP-seq binding sites 7

4.2 Link ATAC-seq peaks to target genes 7

4.3 Add TF motif binding to peaks 8

4.4 Generate regulons 8

4.5 Calculate TF activity 10

4.6 Perform differential activity 11

4.7 Visualize the results 11

4.8 Geneset enrichment 13

4.9 Network analysis 14

5 Session Info 15

1 Introduction

This tutorial walks through an example of TF activity inference in unpaired scATAC-seq/scRNAseq of parental LNCaP cells treated with DMSO, Enzalutamide and Enza resistant cells. The dataset was taken from Taavitsainen et al GSE168667 and GSE168668.

2 Installation

Epiregulon is currently available on R/dev

```
library(epiregulon)
library(ArchR, quietly = TRUE)
```

Alternatively, you could install from gitlab

```
devtools::install_github(repo='xiaosaiyao/epiregulon')

library(epiregulon)
```

3 Data preparation

Please refer to the full ArchR manual for instructions

Before running Epiregulon, the following analyses need to be completed: 1. Obtain a peak matrix on scATACseq by using addGroupCoverages > addReproduciblePeakSet > addPeakMatrix. See chapter 10 from ArchR manual 2. RNA-seq integration. a. For unpaired scATAC-seq, use addGeneIntegrationMatrix. See chapter 8 from ArchR manual b. For multiome data, use addGeneExpressionMatrix. See multiome tutorial 3. Perform dimensionality reduction from with either single modalities or joint scRNAseq and scATACseq using addCombinedDims

3.1 Load ArchR project

Copy this ArchR project into your own directory

```
archR_project_path <- "/gstore/project/lineage/prostate/GSE168667/OUTPUT/multiome/"
proj <- loadArchRProject(path = archR_project_path, showLogo = F)
```

We verify that "GeneExpressionMatrix" and "PeakMatrix" are present for this tutorial.

```
getAvailableMatrices(proj)
#> [1] "GeneIntegrationMatrix" "GeneScoreMatrix"       "MotifMatrix"
#> [4] "PeakMatrix"            "TileMatrix"
```

We will use the joint reducedDims - "LSI_Combined" and joint embeddings - "UMAP_Combined"

```
head(getReducedDims(proj, reducedDims = "iLSI_Combined"))
#>                                     LSI1       LSI2       LSI3        LSI4
#> SRR13927735#TTATGTCTCCAGGTAT-1 -2.713935 -0.3677949 -0.4484238 -0.30645138
#> SRR13927735#TATTGCTCATCAGAAA-1 -2.642781 -0.2767556 -0.9142714 -0.19675812
#> SRR13927735#TTCGATTGTAGGGTTG-1 -2.322865 -0.1543080 -1.4106049 -0.08891276
#> SRR13927735#CATTCATTCGGATGTT-1 -2.572976 -0.1917188 -1.0464294 -0.12660121
#> SRR13927735#ACGTTAGGTCAACTGT-1 -2.478552 -0.1776639 -1.1037295 -0.22976613
#> SRR13927735#AAATGCCCAGCAATGG-1 -2.595352 -0.3803464 -0.7770309 -0.52431765
#>                                        LSI5        LSI6        LSI7       LSI8
#> SRR13927735#TTATGTCTCCAGGTAT-1 -0.046845365 -0.14806535  0.36102164 0.46297594
#> SRR13927735#TATTGCTCATCAGAAA-1  0.075746940 -0.06852359  0.14803384 0.27287412
#> SRR13927735#TTCGATTGTAGGGTTG-1  0.019873276  0.47366272 -0.15422837 0.09187684
#> SRR13927735#CATTCATTCGGATGTT-1  0.009947438  0.03001987  0.16610446 0.12911657
#> SRR13927735#ACGTTAGGTCAACTGT-1 -0.150097539  0.37821625 -0.05693471 0.09996632
#> SRR13927735#AAATGCCCAGCAATGG-1 -0.243074591  0.12202430  0.38184389 0.20992437
#>                                       LSI9      LSI10       LSI11      LSI12
#> SRR13927735#TTATGTCTCCAGGTAT-1  0.01682013 0.29611278  0.07657745 0.03883701
#> SRR13927735#TATTGCTCATCAGAAA-1  0.07502874 0.40427598  0.05240523 0.12557032
#> SRR13927735#TTCGATTGTAGGGTTG-1 -0.07306421 0.35186116 -0.08342128 0.16843772
#> SRR13927735#CATTCATTCGGATGTT-1  0.02916821 0.20807871  0.22959596 0.10711768
#> SRR13927735#ACGTTAGGTCAACTGT-1 -0.03435258 0.40627666 -0.26857174 0.04646805
#> SRR13927735#AAATGCCCAGCAATGG-1  0.25086864 0.04358147  0.19340922 0.11899600
#>                                    LSI13      LSI14     LSI15     LSI16
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.2018137 0.20558422 0.1023789 0.2195854
#> SRR13927735#TATTGCTCATCAGAAA-1 0.2429284 0.12608854 0.1715782 0.1731747
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.3813882 0.15746515 0.2877341 0.1183476
#> SRR13927735#CATTCATTCGGATGTT-1 0.4454115 0.09955226 0.1430440 0.1260116
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.3875503 0.08277628 0.3739350 0.1719490
#> SRR13927735#AAATGCCCAGCAATGG-1 0.2244726 0.20214931 0.1344259 0.1370816
#>                                     LSI17       LSI18       LSI19     LSI20
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.19996628  0.10875713  0.24293288 0.1112905
#> SRR13927735#TATTGCTCATCAGAAA-1 0.13061765  0.11369220  0.17366568 0.1901134
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.05806227 -0.02644245 -0.02346502 0.2238800
#> SRR13927735#CATTCATTCGGATGTT-1 0.28125398 -0.14020962  0.21664823 0.2432303
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.26310264  0.02246457  0.13454621 0.1528337
#> SRR13927735#AAATGCCCAGCAATGG-1 0.09861403  0.17826822  0.23243879 0.1685854
#>                                    LSI21       LSI22      LSI23     LSI24
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.1810559  0.06416606 0.09047377 0.1716186
#> SRR13927735#TATTGCTCATCAGAAA-1 0.1737929  0.15944153 0.13163950 0.1482455
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.3598573 -0.11920835 0.34153417 0.1523253
#> SRR13927735#CATTCATTCGGATGTT-1 0.2018985  0.06210571 0.06702196 0.3012980
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.2882480  0.10880791 0.27567952 0.2035930
#> SRR13927735#AAATGCCCAGCAATGG-1 0.1869793  0.14358246 0.21654445 0.1749158
#>                                    LSI25        LSI26     LSI27      LSI28
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.1403622  0.175519412 0.1355804 0.12816113
#> SRR13927735#TATTGCTCATCAGAAA-1 0.1801719  0.184515106 0.1700654 0.13060154
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.3049924  0.151665898 0.2127343 0.15963237
#> SRR13927735#CATTCATTCGGATGTT-1 0.2559221 -0.004725876 0.1544931 0.08703268
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.1753307  0.211852608 0.2488278 0.13749527
#> SRR13927735#AAATGCCCAGCAATGG-1 0.1892084  0.195339053 0.2198201 0.15207755
#>                                    LSI29     LSI30      LSI1       LSI2
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.1609296 0.1389944 -2.209204 -0.4032802
#> SRR13927735#TATTGCTCATCAGAAA-1 0.2074106 0.1374118 -2.209137 -0.3309160
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.1912926 0.2498970 -2.209636 -0.5382591
#> SRR13927735#CATTCATTCGGATGTT-1 0.2797500 0.2088381 -2.202239 -0.3678359
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.2089094 0.1208387 -2.202195 -0.5524992
#> SRR13927735#AAATGCCCAGCAATGG-1 0.1855635 0.1594068 -2.212632 -0.4745915
#>                                     LSI3      LSI4      LSI5       LSI6
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.1854419 0.1286875 0.5048250 0.07524760
#> SRR13927735#TATTGCTCATCAGAAA-1 0.1323689 0.1268916 0.6162274 0.07814090
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.3150100 0.1307959 0.4933786 0.08463096
#> SRR13927735#CATTCATTCGGATGTT-1 0.1696307 0.1317155 0.5964244 0.07006297
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.4028122 0.1425672 0.1999717 0.12659465
#> SRR13927735#AAATGCCCAGCAATGG-1 0.3222372 0.1335688 0.2225833 0.12202483
#>                                       LSI7      LSI8        LSI9      LSI10
#> SRR13927735#TTATGTCTCCAGGTAT-1 -0.35785473 0.4817790 -0.05664032 0.03204436
#> SRR13927735#TATTGCTCATCAGAAA-1 -0.33266771 0.4738935 -0.05501021 0.05411872
#> SRR13927735#TTCGATTGTAGGGTTG-1  0.06296525 0.2133872  0.09193898 0.08350961
#> SRR13927735#CATTCATTCGGATGTT-1 -0.23863574 0.4141768 -0.02025247 0.06031304
#> SRR13927735#ACGTTAGGTCAACTGT-1 -0.15761402 0.3717331  0.04228017 0.07367151
#> SRR13927735#AAATGCCCAGCAATGG-1 -0.32839556 0.4796359 -0.02417076 0.05700212
#>                                     LSI11     LSI12     LSI13      LSI14
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.08595580 0.1109641 0.2561409 -0.3478001
#> SRR13927735#TATTGCTCATCAGAAA-1 0.09405159 0.1003529 0.2892364 -0.4078682
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.08640174 0.1015529 0.3230336 -0.5319555
#> SRR13927735#CATTCATTCGGATGTT-1 0.09162578 0.1095239 0.3242656 -0.4718060
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.10927919 0.1199679 0.3302774 -0.5167105
#> SRR13927735#AAATGCCCAGCAATGG-1 0.10083782 0.1319900 0.2948185 -0.4510144
#>                                    LSI15      LSI16      LSI17      LSI18
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.1419596 0.05441494 0.05018115 0.09520132
#> SRR13927735#TATTGCTCATCAGAAA-1 0.1151644 0.07613093 0.06766369 0.09468311
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.1244211 0.05123131 0.04831037 0.11283007
#> SRR13927735#CATTCATTCGGATGTT-1 0.1179657 0.06305818 0.07729603 0.08787124
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.1436639 0.06566479 0.08390887 0.12688689
#> SRR13927735#AAATGCCCAGCAATGG-1 0.1567464 0.07526364 0.07965473 0.11700643
#>                                     LSI19     LSI20      LSI21      LSI22
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.08324288 0.3892976 0.08381601 0.11539127
#> SRR13927735#TATTGCTCATCAGAAA-1 0.08070564 0.2933891 0.10302506 0.06988481
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.11299285 0.2832497 0.08539803 0.07264449
#> SRR13927735#CATTCATTCGGATGTT-1 0.07341707 0.3523934 0.06983856 0.04895783
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.11725370 0.3079927 0.10094968 0.11564471
#> SRR13927735#AAATGCCCAGCAATGG-1 0.12183265 0.2742798 0.11723113 0.13268959
#>                                     LSI23        LSI24       LSI25       LSI26
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.09745417 -0.002162669 -0.04688207 0.008186978
#> SRR13927735#TATTGCTCATCAGAAA-1 0.08886289  0.017777285  0.03437342 0.072836804
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.07610108  0.044337563  0.04797779 0.085051746
#> SRR13927735#CATTCATTCGGATGTT-1 0.07687693  0.040649380  0.06753790 0.106114526
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.11329027  0.007616145  0.00301091 0.064081736
#> SRR13927735#AAATGCCCAGCAATGG-1 0.11257099  0.002192787 -0.02552127 0.029155362
#>                                     LSI27      LSI28        LSI29         LSI30
#> SRR13927735#TTATGTCTCCAGGTAT-1 0.08678964 0.12048334  0.114421880  0.1218968657
#> SRR13927735#TATTGCTCATCAGAAA-1 0.11431432 0.10840706  0.032296813  0.0008020251
#> SRR13927735#TTCGATTGTAGGGTTG-1 0.15082590 0.08956411 -0.030534956 -0.0611557294
#> SRR13927735#CATTCATTCGGATGTT-1 0.14092826 0.10458593 -0.025202526 -0.0692581235
#> SRR13927735#ACGTTAGGTCAACTGT-1 0.15699428 0.11138111  0.003277357 -0.0117534629
#> SRR13927735#AAATGCCCAGCAATGG-1 0.12282663 0.12647146  0.098017281  0.0856883114
head(getEmbedding(proj, embedding = "UMAP_Combined"))
#>                                iLSI_Combined#UMAP_Dimension_1
#> SRR13927735#TTATGTCTCCAGGTAT-1                      -9.622903
#> SRR13927735#TATTGCTCATCAGAAA-1                      -9.360211
#> SRR13927735#TTCGATTGTAGGGTTG-1                      -8.617347
#> SRR13927735#CATTCATTCGGATGTT-1                      -9.285448
#> SRR13927735#ACGTTAGGTCAACTGT-1                      -8.809260
#> SRR13927735#AAATGCCCAGCAATGG-1                      -9.261216
#>                                iLSI_Combined#UMAP_Dimension_2
#> SRR13927735#TTATGTCTCCAGGTAT-1                     -0.2908237
#> SRR13927735#TATTGCTCATCAGAAA-1                     -0.2892935
#> SRR13927735#TTCGATTGTAGGGTTG-1                     -0.2154103
#> SRR13927735#CATTCATTCGGATGTT-1                     -0.3267481
#> SRR13927735#ACGTTAGGTCAACTGT-1                     -0.2168703
#> SRR13927735#AAATGCCCAGCAATGG-1                      0.3200356
```

3.2 Retrieve matrices from ArchR project

Retrieve gene expression and peak matrix from the ArchR project

```
GeneExpressionMatrix <- getMatrixFromProject(
    ArchRProj = proj,
    useMatrix = "GeneIntegrationMatrix",
    useSeqnames = NULL,
    verbose = TRUE,
    binarize = FALSE,
    threads = 1,
    logFile = "x"
)
#> 2022-11-16 23:14:49 : Organizing colData, 1.367 mins elapsed.
#> 2022-11-16 23:14:49 : Organizing rowData, 1.369 mins elapsed.
#> 2022-11-16 23:14:49 : Organizing rowRanges, 1.369 mins elapsed.
#> 2022-11-16 23:14:49 : Organizing Assays (1 of 1), 1.369 mins elapsed.
#> 2022-11-16 23:14:55 : Constructing SummarizedExperiment, 1.467 mins elapsed.
#> 2022-11-16 23:14:58 : Finished Matrix Creation, 1.514 mins elapsed.

PeakMatrix <- getMatrixFromProject(
    ArchRProj = proj,
    useMatrix = "PeakMatrix",
    useSeqnames = NULL,
    verbose = TRUE,
    binarize = FALSE,
    threads = 1,
    logFile = "x"
)
```

```
#> 2022-11-16 23:15:57 : Organizing colData, 0.981 mins elapsed.
#> 2022-11-16 23:15:57 : Organizing rowData, 0.983 mins elapsed.
#> 2022-11-16 23:15:57 : Organizing rowRanges, 0.983 mins elapsed.
#> 2022-11-16 23:15:57 : Organizing Assays (1 of 1), 0.984 mins elapsed.
#> 2022-11-16 23:16:00 : Constructing SummarizedExperiment, 1.04 mins elapsed.
#> 2022-11-16 23:16:21 : Finished Matrix Creation, 1.383 mins elapsed.
```

Convert gene expression matrix to SingleCellExperiment object

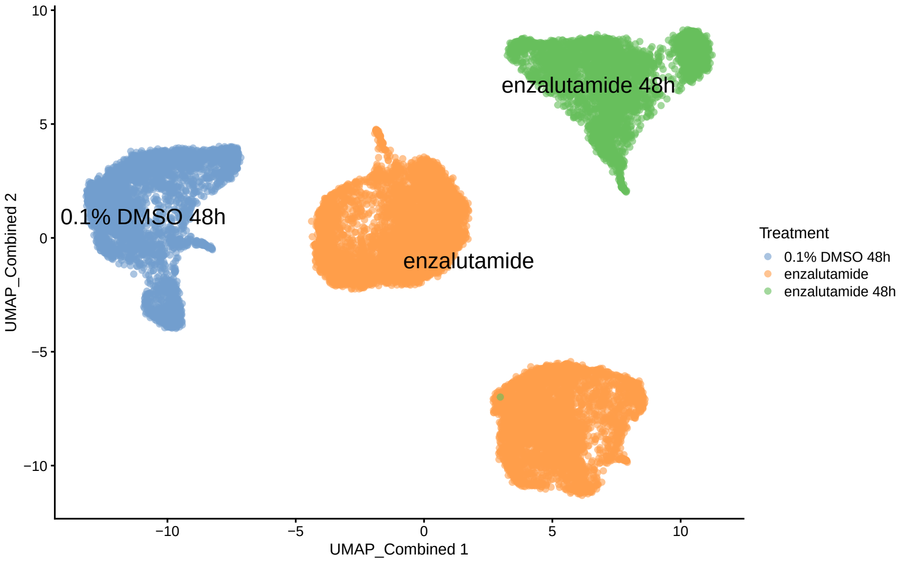
```
GeneExpressionMatrix <- as(GeneExpressionMatrix, "SingleCellExperiment")
assayNames(GeneExpressionMatrix)  <-  "logcounts"
assayNames(PeakMatrix)  <-  "counts"
```

Transfer cell and gene information and embeddings from ArchR project to singleCellExperiment

```
reducedDim(GeneExpressionMatrix, "UMAP_Combined") <- getEmbedding(ArchRProj = proj,
                                                                  embedding = "UMAP_Combined",
                                                                  returnDF = TRUE)[colnames(GeneExpressionMat
colData(GeneExpressionMatrix) <- getCellColData(proj)[colnames(GeneExpressionMatrix),]
rownames(GeneExpressionMatrix) <- rowData(GeneExpressionMatrix)$name
```

Visualize singleCellExperiment by UMAP

```
scater::plotReducedDim(GeneExpressionMatrix,
                       dimred = "UMAP_Combined",
                       text_by = "Treatment",
                       colour_by = "Treatment")
```



4 Quick start

4.1 Retrieve bulk TF ChIP-seq binding sites

First, we retrieve the information of TF binding sites collected from Cistrome and ENCODE ChIP-seq, which are hosted on Genomitory. Currently, human genomes HG19 and HG38 and mouse mm10 are available.

```
grl <- getTFMotifInfo(genome = "hg38")
#> redirecting from 'GMTY162:hg38_motif_bed_granges@REVISION-4' to 'GMTY162:hg38_motif_bed_granges@24c22e4f42
head(grl)
#> GRangesList object of length 6:
#> $`5-hmC`
#> GRanges object with 24048 ranges and 0 metadata columns:
#>           seqnames            ranges strand
#>              <Rle>         <IRanges>  <Rle>
#>       [1]     chr1       10000-10685      *
#>       [2]     chr1       13362-13694      *
#>       [3]     chr1       29631-29989      *
#>       [4]     chr1       40454-40754      *
#>       [5]     chr1     135395-135871      *
#>       ...      ...               ...    ...
#>   [24044]     chrY 56864377-56864627      *
#>   [24045]     chrY 56876124-56876182      *
#>   [24046]     chrM           84-2450      *
#>   [24047]     chrM       13613-14955      *
#>   [24048]     chrM       15134-16490      *
#>   -------
#>   seqinfo: 25 sequences from an unspecified genome; no seqlengths
#>
#> ...
#> <5 more elements>
```

4.2 Link ATAC-seq peaks to target genes

Next, we compute peak to gene correlations using the `addPeak2GeneLinks` function from the ArchR package. The user would need to supply a path to an ArchR project already containing peak and gene matrices, as well as Latent semantic indexing (LSI) dimensionality reduction.

```
# path to ArchR project
p2g <- calculateP2G(ArchR_path = archR_project_path,
                    useDim = "iLSI_Combined",
                    useMatrix = "GeneIntegrationMatrix")
#> Setting ArchRLogging = FALSE
#> Using ArchR to compute peak to gene links...
#> 2022-11-16 23:17:05 : Getting Available Matrices, 0 mins elapsed.
#> 2022-11-16 23:17:09 : Filtered Low Prediction Score Cells (0 of 15522, 0), 0.028 mins elapsed.
#> 2022-11-16 23:17:10 : Computing KNN, 0.047 mins elapsed.
#> 2022-11-16 23:17:10 : Identifying Non-Overlapping KNN pairs, 0.055 mins elapsed.
#> 2022-11-16 23:17:13 : Identified 498 Groupings!, 0.096 mins elapsed.
#> 2022-11-16 23:17:13 : Getting Group RNA Matrix, 0.097 mins elapsed.
#> 2022-11-16 23:20:51 : Getting Group ATAC Matrix, 3.726 mins elapsed.
#> 2022-11-16 23:24:23 : Normalizing Group Matrices, 7.272 mins elapsed.
#> 2022-11-16 23:24:32 : Finding Peak Gene Pairings, 7.409 mins elapsed.
#> 2022-11-16 23:24:32 : Computing Correlations, 7.419 mins elapsed.
#> 2022-11-16 23:24:42 : Completed Peak2Gene Correlations!, 7.576 mins elapsed.
head(p2g)
#>   idxATAC  chr  start    end idxRNA target Correlation distance          FDR
#> 1      15 chr1 912762 913262      7  NOC2L   0.5467220    46297 2.268176e-38
#> 2      15 chr1 912762 913262      8 KLHL17   0.5165395    47575 1.150334e-33
#> 3      25 chr1 920261 920761      7  NOC2L   0.6494254    38798 1.161988e-58
#> 4      25 chr1 920261 920761      8 KLHL17   0.6377107    40076 5.991862e-56
#> 5      32 chr1 927728 928228      7  NOC2L   0.6102405    31331 5.010307e-50
#> 6      32 chr1 927728 928228      8 KLHL17   0.5500926    32609 6.302048e-39
```

4.3 Add TF motif binding to peaks

The next step is to add the TF motif binding information by overlapping the regions of the peak matrix with the bulk chip-seq database loaded in 2. The user can supply either an archR project path and this function will retrieve the peak matrix, or a peakMatrix in the form of a Granges object or RangedSummarizedExperiment.

```
overlap <- addTFMotifInfo(archR_project_path = archR_project_path, grl = grl, p2g = p2g)
#> Successfully loaded ArchRProject!
#> Computing overlap...
#> Success!
```

4.4 Generate regulons

A long format dataframe, representing the inferred regulons, is then generated. The dataframe consists of three columns:

- tf (transcription factor)
- target gene
- peak to gene correlation between tf and target gene

```
regulon <- getRegulon(p2g = p2g, overlap = overlap, aggregate = FALSE)
head(regulon)
#>   idxATAC idxTF       tf  chr  start    end idxRNA target      corr distance
#> 1      15    10     AGO1 chr1 912762 913262      8 KLHL17 0.5165395    47575
#> 2      15    10     AGO1 chr1 912762 913262      7  NOC2L 0.5467220    46297
#> 3      15    22 AML1-ETO chr1 912762 913262      8 KLHL17 0.5165395    47575
#> 4      15    22 AML1-ETO chr1 912762 913262      7  NOC2L 0.5467220    46297
#> 5      15    32   ARID4A chr1 912762 913262      8 KLHL17 0.5165395    47575
#> 6      15    32   ARID4A chr1 912762 913262      7  NOC2L 0.5467220    46297
#>            FDR
#> 1 1.150334e-33
#> 2 2.268176e-38
#> 3 1.150334e-33
#> 4 2.268176e-38
#> 5 1.150334e-33
#> 6 2.268176e-38
```

```
pruned.regulon <- pruneRegulon(expMatrix = GeneExpressionMatrix,
                               peakMatrix = PeakMatrix,
                               peak_assay = "counts",
                               regulon = regulon[regulon$tf %in% c("AR","FOXA1"),],
                               clusters = GeneExpressionMatrix$Sample,
                               prune_value = "pval",
                               regulon_cutoff = 0.05,
                               BPPARAM = BiocParallel::MulticoreParam(progressbar = TRUE))
#> pruning network with binom tests using a regulon cutoff of pval<0.05
#> binarizing matrices
#> pruning regulons
#> 
  |                                                                            
  |                                                                      |   0%
  |                                                                            
  |===================================                                   |  50%
  |                                                                            
  |======================================================================| 100%
```

Epiregulon computes weights using either correlation, linear regression, mutual information, log fold change or wilcoxon rank sum test. In this case, we chose logFC and set `tf_re.merge = TRUE` because the drug enzalutamide acts to alter chromatin accessibility of AR but less so AR expression.

```
regulon.w <- addWeights(regulon = pruned.regulon,
                        expMatrix = GeneExpressionMatrix,
                        exp_assay = "logcounts",
                        peakMatrix = PeakMatrix,
                        peak_assay = "counts",
                        clusters = GeneExpressionMatrix$Sample,
                        method = "logFC",
                        tf_re.merge = TRUE,
                        BPPARAM = BiocParallel::SerialParam(progressbar = TRUE))
#> adding weights using logFC
#> binarizing matrices...
#> computing weights...
```

```
head(regulon.w)
#>      idxATAC idxTF tf  chr   start     end idxRNA target      corr distance
#> 357       73    25 AR chr1 1079335 1079835     26 UBE2J2 0.5334895   194300
#> 358       73    25 AR chr1 1079335 1079835     23   SDF4 0.5431564   152446
#> 1429      74    25 AR chr1 1080041 1080541     26 UBE2J2 0.5412880   193594
#> 1430      74    25 AR chr1 1080041 1080541     23   SDF4 0.5987732   151740
#> 2147      76    25 AR chr1 1109628 1110128     26 UBE2J2 0.6634661   164007
#> 2148      76    25 AR chr1 1109628 1110128     23   SDF4 0.6799768   122153
#>               FDR     pval_all pval_SRR13927735 pval_SRR13927736
#> 357  3.000898e-36 3.627978e-13        1.0000000       1.00000000
#> 358  8.644608e-38 3.879655e-09        0.7781537       0.69204769
#> 1429 1.730885e-37 5.694933e-17        1.0000000       1.00000000
#> 1430 1.000312e-47 3.238105e-10        0.4203787       0.02189775
#> 2147 4.520046e-62 9.114956e-18        1.0000000       1.00000000
#> 2148 2.485338e-66 5.873488e-14        1.0000000       1.00000000
#>      pval_SRR13927737 pval_SRR13927738 stats_all stats_SRR13927735
#> 357       0.005993675      0.418440077  7.268762         0.0000000
#> 358       0.518345846      0.451586331  5.889244        -0.2817258
#> 1429      0.242611928      0.006095821  8.371376         0.0000000
#> 1430      0.137275483      0.369133167  6.286927        -0.8057645
#> 2147      1.000000000      0.032141910  8.584604         0.0000000
#> 2148      0.576153708      0.876455453  7.510872         0.0000000
#>      stats_SRR13927736 stats_SRR13927737 stats_SRR13927738     padj_all
#> 357          0.0000000         2.7481272         0.8091305 9.711373e-09
#> 358          0.3960777        -0.6458973        -0.7527730 1.007857e-04
#> 1429         0.0000000         1.1684832         2.7425821 1.544181e-12
#> 1430         2.2921368         1.4860138        -0.8980983 8.492579e-06
#> 2147         0.0000000         0.0000000         2.1426413 2.478083e-13
#> 2148         0.0000000         0.5590117         0.1554641 1.577443e-09
#>      padj_SRR13927735 padj_SRR13927736 padj_SRR13927737 padj_SRR13927738
#> 357                 1                1                1                1
#> 358                 1                1                1                1
#> 1429                1                1                1                1
#> 1430                1                1                1                1
#> 2147                1                1                1                1
#> 2148                1                1                1                1
#>          weight
#> 357  0.06266724
#> 358  0.05482771
#> 1429 0.08591384
#> 1430 0.07151088
#> 2147 0.19433566
#> 2148 0.19032049
```

4.5 Calculate TF activity

Finally, the activities for a specific TF in each cell are computed by averaging the weighted expressions of target genes linked to the TF.

$$y = \frac{1}{n}\sum_{i=1}^{n} x_i * weight_i$$

where y is the activity of a TF for a cell n is the total number of targets for a TF x_i is the log count expression of target i where i in $\{1,2,\ldots,n\}$ $weight_i$ is the weight of TF and target i

```
score.combine <- calculateActivity(expMatrix = GeneExpressionMatrix,
                                   regulon = regulon.w,
                                   normalize = TRUE,
                                             mode = "weight",
                                             method = "weightedMean")
#> calculating TF activity from regulon using weightedmean
```

4.6 Perform differential activity

```
markers <- findDifferentialActivity(activity_matrix = score.combine,
                                    groups = GeneExpressionMatrix$Sample,
                                    pval.type = "some",
                                    direction = "up",
                                    test.type = "t")
```

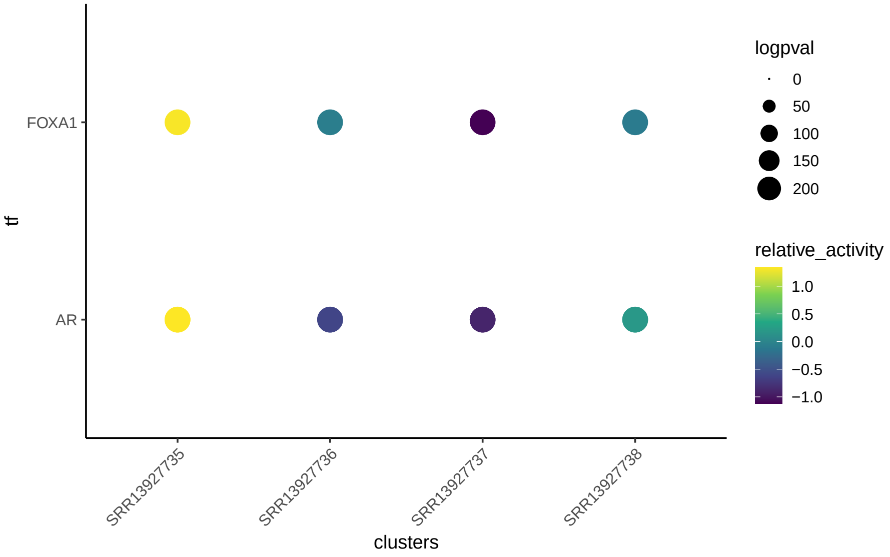
Take the top TFs

```
markers.sig <- getSigGenes(markers, topgenes = 5 )
#> Using a logFC cutoff of 0 for class SRR13927735
#> Using a logFC cutoff of 0 for class SRR13927736
#> Using a logFC cutoff of 0 for class SRR13927737
#> Using a logFC cutoff of 0 for class SRR13927738
```

4.7 Visualize the results

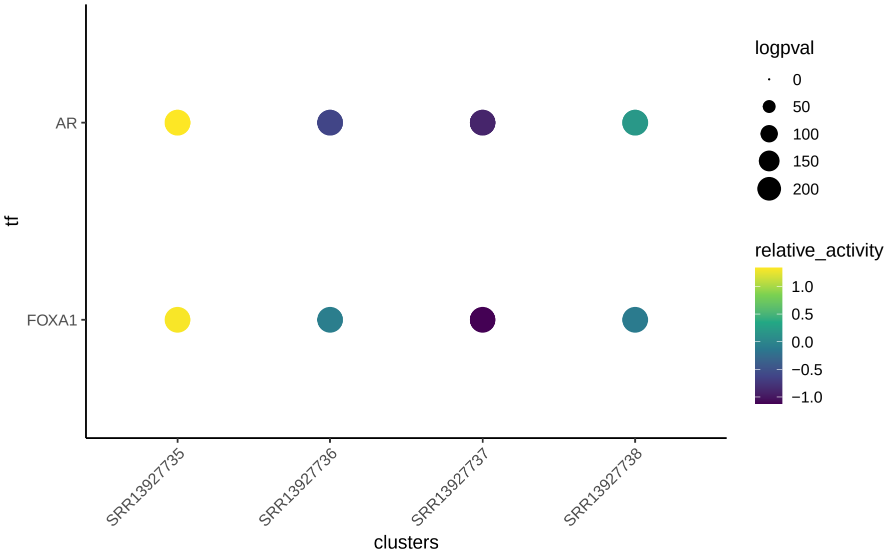
First visualize the known differential TFs by bubble plot

```
plotBubble(activity_matrix = score.combine,
           tf = c("AR","FOXA1"),
           clusters = GeneExpressionMatrix$Sample)
```



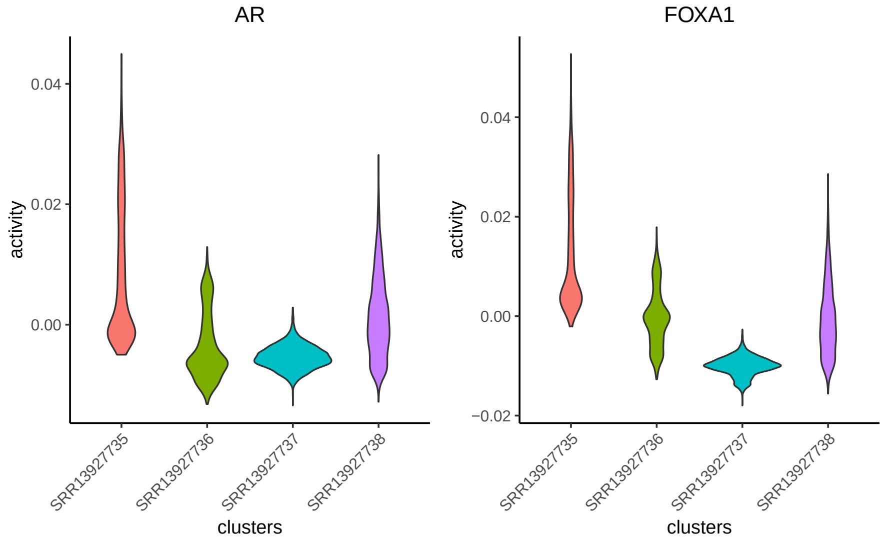
Then visualize the most differential TFs by clusters

```
plotBubble(activity_matrix = score.combine,
           tf = markers.sig$tf,
           clusters = GeneExpressionMatrix$Sample)
```



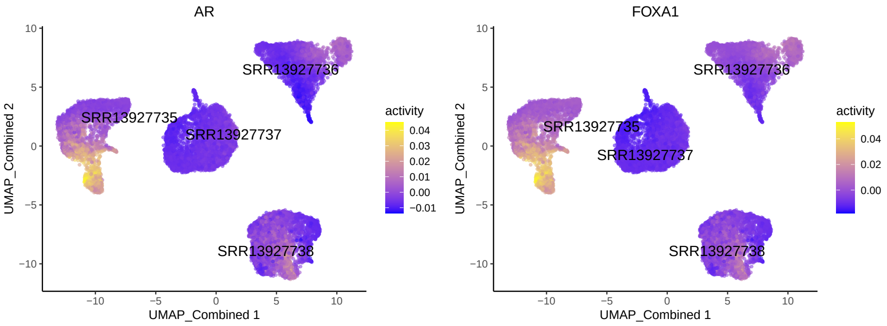
Visualize the known differential TFs by violin plot.

```
plotActivityViolin(activity_matrix = score.combine,
                   tf = c( "AR", "FOXA1"),
                   clusters = GeneExpressionMatrix$Sample)
```



Visualize the known differential TFs by UMAP

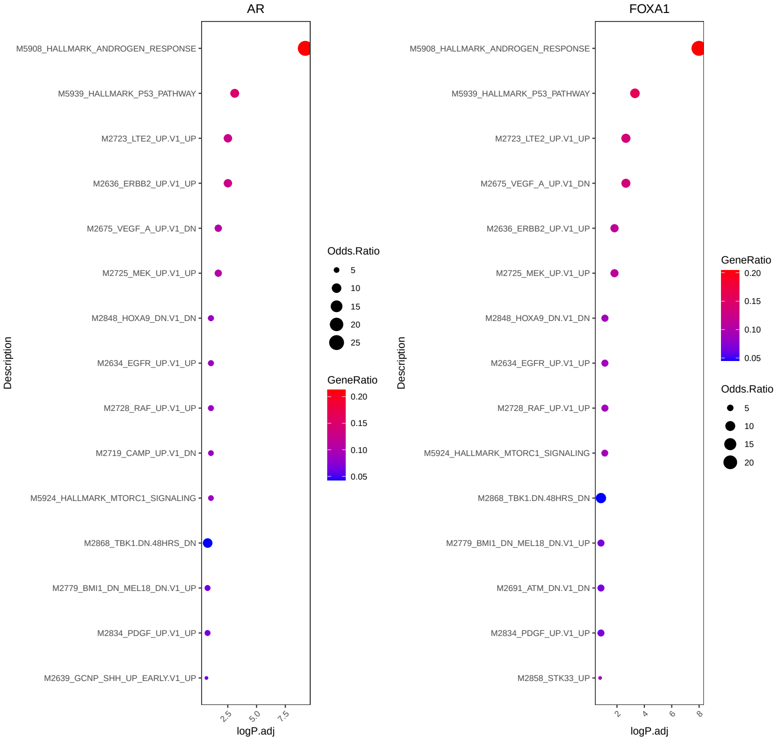
```
plotActivityDim(sce = GeneExpressionMatrix,
                activity_matrix = score.combine,
                tf = c( "AR", "FOXA1"),
                dimtype = "UMAP_Combined",
                label = "Sample",
                point_size = 1,
                ncol = 2,
                nrow = 2)
```



4.8 Geneset enrichment

Sometimes we are interested to know what pathways are enriched in the regulon of a particular TF. We can perform geneset enrichment using the enricher function from clusterProfiler.

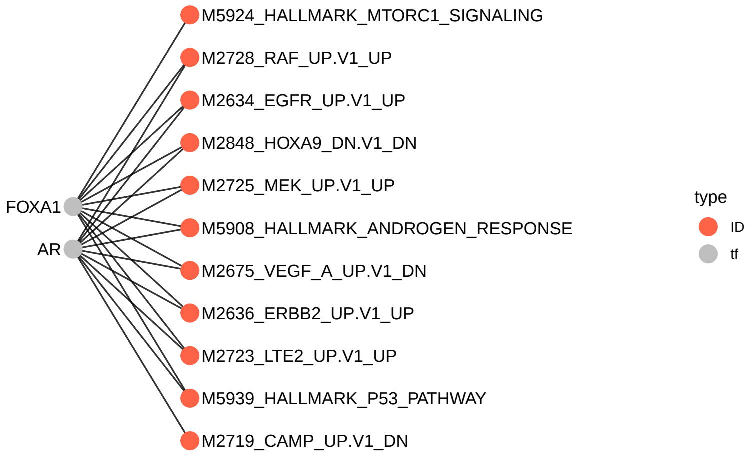
```
#retrieve genesets
H <- EnrichmentBrowser::getGenesets(org = "hsa", db = "msigdb", cat = "H", gene.id.type = "SYMBOL" )
#> Using cached version from 2022-11-11 22:43:20
C6 <- EnrichmentBrowser::getGenesets(org = "hsa", db = "msigdb", cat = "C6", gene.id.type = "SYMBOL" )
#> Using cached version from 2022-11-11 22:43:25

#combine genesets and convert genesets to be compatible with enricher
gs <- c(H,C6)
gs.list <- do.call(rbind,lapply(names(gs), function(x) {data.frame(gs=x, genes=gs[[x]])}))

enrichresults <- regulonEnrich(TF = c("AR","FOXA1"),
                               regulon = regulon.w,
                               corr = "weight",
                               corr_cutoff = 0.5,
                               genesets = gs.list)

#plot results
enrichPlot(results = enrichresults)
```



4.9 Network analysis

We can visualize the genesets as a network

```
plotGseaNetwork(tf = names(enrichresults),
                enrichresults = enrichresults,
                p.adj_cutoff = 0.1,
                ntop_pathways = 10)
```



differential networks

5 Session Info

```
sessionInfo()
#> R version 4.2.0 (2022-04-22)
#> Platform: x86_64-pc-linux-gnu (64-bit)
#> Running under: Ubuntu 18.04.6 LTS
#>
#> Matrix products: default
#> BLAS:   /usr/local/lib/R/lib/libRblas.so
#> LAPACK: /usr/local/lib/R/lib/libRlapack.so
#>
#> Random number generation:
#>  RNG:     L'Ecuyer-CMRG
#>  Normal:  Inversion
#>  Sample:  Rejection
#>
#> locale:
#>  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
#>  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
#>  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
#>  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
#>  [9] LC_ADDRESS=C               LC_TELEPHONE=C
#> [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
#>
#> attached base packages:
#>  [1] parallel  grid      stats4    stats     graphics  grDevices utils
#>  [8] datasets  methods   base
#>
```

```
#> other attached packages:
#>  [1] nabor_0.5.0                 rhdf5_2.42.0
#>  [3] Rcpp_1.0.9                  Matrix_1.5-3
#>  [5] data.table_1.14.6           stringr_1.4.0
#>  [7] plyr_1.8.8                  magrittr_2.0.3
#>  [9] gtable_0.3.1                gtools_3.9.3
#> [11] gridExtra_2.3               ArchR_1.0.2
#> [13] msigdbr_7.5.1               epiregulon_1.0.18
#> [15] scater_1.26.1               ggplot2_3.4.0
#> [17] scuttle_1.8.0               scRNAseq_2.12.0
#> [19] SingleCellExperiment_1.20.0 SummarizedExperiment_1.29.1
#> [21] Biobase_2.58.0              GenomicRanges_1.50.1
#> [23] GenomeInfoDb_1.34.3         IRanges_2.32.0
#> [25] S4Vectors_0.36.0            BiocGenerics_0.44.0
#> [27] MatrixGenerics_1.10.0       matrixStats_0.62.0
#> [29] dorothea_1.10.0             BiocStyle_2.26.0
#> [31] rmarkdown_2.18
#>
#> loaded via a namespace (and not attached):
#>   [1] rappdirs_0.3.3                rtracklayer_1.58.0
#>   [3] tidyr_1.2.1                   bit64_4.0.5
#>   [5] knitr_1.40                    irlba_2.3.5.1
#>   [7] DelayedArray_0.24.0           KEGGREST_1.38.0
#>   [9] RCurl_1.98-1.9                AnnotationFilter_1.22.0
#>  [11] generics_0.1.3                GenomicFeatures_1.50.2
#>  [13] ScaledMatrix_1.6.0            cowplot_1.1.1
#>  [15] RSQLite_2.2.18                shadowtext_0.1.2
#>  [17] artificer.mae_1.3.4           base64url_1.4
#>  [19] bit_4.0.5                     enrichplot_1.18.0
#>  [21] gp.cache_1.7.1                xml2_1.3.3
#>  [23] httpuv_1.6.6                  genomitory_2.1.6
#>  [25] assertthat_0.2.1              viridis_0.6.2
#>  [27] xfun_0.31                     hms_1.1.2
#>  [29] babelgene_22.9                evaluate_0.18
#>  [31] promises_1.2.0.1              fansi_1.0.3
#>  [33] restfulr_0.0.15               progress_1.2.2
#>  [35] dbplyr_2.2.1                  Rgraphviz_2.42.0
#>  [37] igraph_1.3.5                  DBI_1.1.3
#>  [39] purrr_0.3.5                   ellipsis_0.3.2
#>  [41] dplyr_1.0.10                  backports_1.4.1
#>  [43] bookdown_0.30                 annotate_1.76.0
#>  [45] biomaRt_2.54.0                sparseMatrixStats_1.10.0
#>  [47] artificer.matrix_1.3.7        vctrs_0.5.1
#>  [49] Cairo_1.6-0                   ensembldb_2.22.0
#>  [51] dsdb.plus_1.3.2               cachem_1.0.6
#>  [53] withr_2.5.0                   ggforce_0.4.1
#>  [55] HDO.db_0.99.0                 checkmate_2.1.0
#>  [57] metacommons_1.9.0             GenomicAlignments_1.34.0
#>  [59] treeio_1.22.0                 MultiAssayExperiment_1.24.0
#>  [61] prettyunits_1.1.1             scran_1.26.0
#>  [63] cluster_2.1.3                 DOSE_3.23.3
#>  [65] BiocBaseUtils_1.1.0           ExperimentHub_2.6.0
#>  [67] ape_5.6-2                     lazyeval_0.2.2
#>  [69] crayon_1.5.1                  edgeR_3.40.0
#>  [71] pkgconfig_2.0.3               labeling_0.4.2
#>  [73] tweenr_2.0.2                  nlme_3.1-160
#>  [75] vipor_0.4.5                   ProtGenerics_1.30.0
#>  [77] rlang_1.0.6                   lifecycle_1.0.3
#>  [79] artificer.schemas_0.99.2      downloader_0.4
#>  [81] filelock_1.0.2                artificer.base_1.3.19
#>  [83] BiocFileCache_2.6.0           rsvd_1.0.5
#>  [85] AnnotationHub_3.6.0           polyclip_1.10-4
#>  [87] GSVA_1.46.0                   graph_1.76.0
#>  [89] aplot_0.1.8                   Rhdf5lib_1.20.0
#>  [91] beeswarm_0.4.0                png_0.1-7
#>  [93] viridisLite_0.4.1             rjson_0.2.21
#>  [95] artificer.ranges_1.3.4        bitops_1.0-7
#>  [97] artificer.se_1.3.4            getPass_0.2-2
#>  [99] gson_0.0.9                    rhdf5filters_1.10.0
#> [101] EnrichmentBrowser_2.28.0      Biostrings_2.66.0
#> [103] blob_1.2.3                    DelayedMatrixStats_1.20.0
#> [105] qvalue_2.30.0                 gridGraphics_0.5-1
#> [107] beachmat_2.14.0               scales_1.2.1
#> [109] memoise_2.0.1                 GSEABase_1.60.0
#> [111] zlibbioc_1.44.0               compiler_4.2.0
#> [113] scatterpie_0.1.8              dqrng_0.3.0
#> [115] tinytex_0.42                  BiocIO_1.8.0
#> [117] RColorBrewer_1.1-3            KEGGgraph_1.58.0
#> [119] Rsamtools_2.14.0              cli_3.4.1
#> [121] XVector_0.38.0                patchwork_1.1.2
#> [123] ArtifactDB_1.9.5              MASS_7.3-58.1
#> [125] tidyselect_1.2.0              stringi_1.7.6
#> [127] yaml_2.3.5                    GOSemSim_2.24.0
#> [129] BiocSingular_1.14.0           locfit_1.5-9.6
#> [131] ggrepel_0.9.2                 bcellViper_1.34.0
#> [133] fastmatch_1.1-3               tools_4.2.0
#> [135] bluster_1.8.0                 metapod_1.6.0
#> [137] farver_2.1.1                  ggraph_2.1.0
#> [139] digest_0.6.29                 BiocManager_1.30.19
#> [141] FNN_1.1.3.1                   shiny_1.7.3
#> [143] BiocVersion_3.16.0            later_1.3.0
#> [145] gp.auth_1.7.0                 httr_1.4.3
#> [147] AnnotationDbi_1.60.0          colorspace_2.0-3
#> [149] XML_3.99-0.12                 splines_4.2.0
#> [151] uwot_0.1.14                   yulab.utils_0.0.5
#> [153] statmod_1.4.37                tidytree_0.4.1
#> [155] graphlayouts_0.8.3            ggplotify_0.1.0
#> [157] xtable_1.8-4                  jsonlite_1.8.3
#> [159] ggtree_3.6.2                  tidygraph_1.2.2
#> [161] ggfun_0.0.8                   ShadowArray_1.7.1
#> [163] R6_2.5.1                      pillar_1.8.1
#> [165] htmltools_0.5.3               mime_0.12
#> [167] glue_1.6.2                    fastmap_1.1.0
#> [169] clusterProfiler_4.6.0         BiocParallel_1.32.1
#> [171] BiocNeighbors_1.16.0          interactiveDisplayBase_1.36.0
#> [173] codetools_0.2-18              gp.version_1.5.0
#> [175] fgsea_1.24.0                  utf8_1.2.2
#> [177] lattice_0.20-45               tibble_3.1.8
#> [179] genomitory.schemas_0.99.0     curl_4.3.2
#> [181] ggbeeswarm_0.6.0              artificer.sce_1.3.4
#> [183] GO.db_3.16.0                  limma_3.54.0
#> [185] dsassembly_1.7.6              dsdb.schemas_0.99.1
#> [187] munsell_0.5.0                 GenomeInfoDbData_1.2.9
#> [189] HDF5Array_1.26.0              reshape2_1.4.4
```