Using vizAPA: a minimal tutorial

Xiaohui Wu, Xingyu Bi, Wenbin Ye

2023-09-07

Overview

This tutorial takes a `PACdataset` object storing a list of poly(A) sites as input and describes some simple but commonly used functions of vizAPA.

Demo PACdataset

In the package of vizAPA, there is a demo `PACdataset` object of mouse sperm cells, containing 974 pAs [poly(A) sites] from 413 genes. This PACdataset has been annotated, with both pAs' and cells' meta data.

```
library(vizAPA)

data(scPACds, package='vizAPA')

# summary of the PACdataset
movAPA::summary(scPACds)
```

```
## PAC# 974
## sample# 955
## summary of expression level of each PA
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1      72     957    3151    3636   96363
## summary of expressed sample# of each PA
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00   64.25  452.50  452.05  810.00  955.00
## gene# 413
##      nPAC
## 3UTR  974
```

```
# cell meta data
head(scPACds@colData)
```

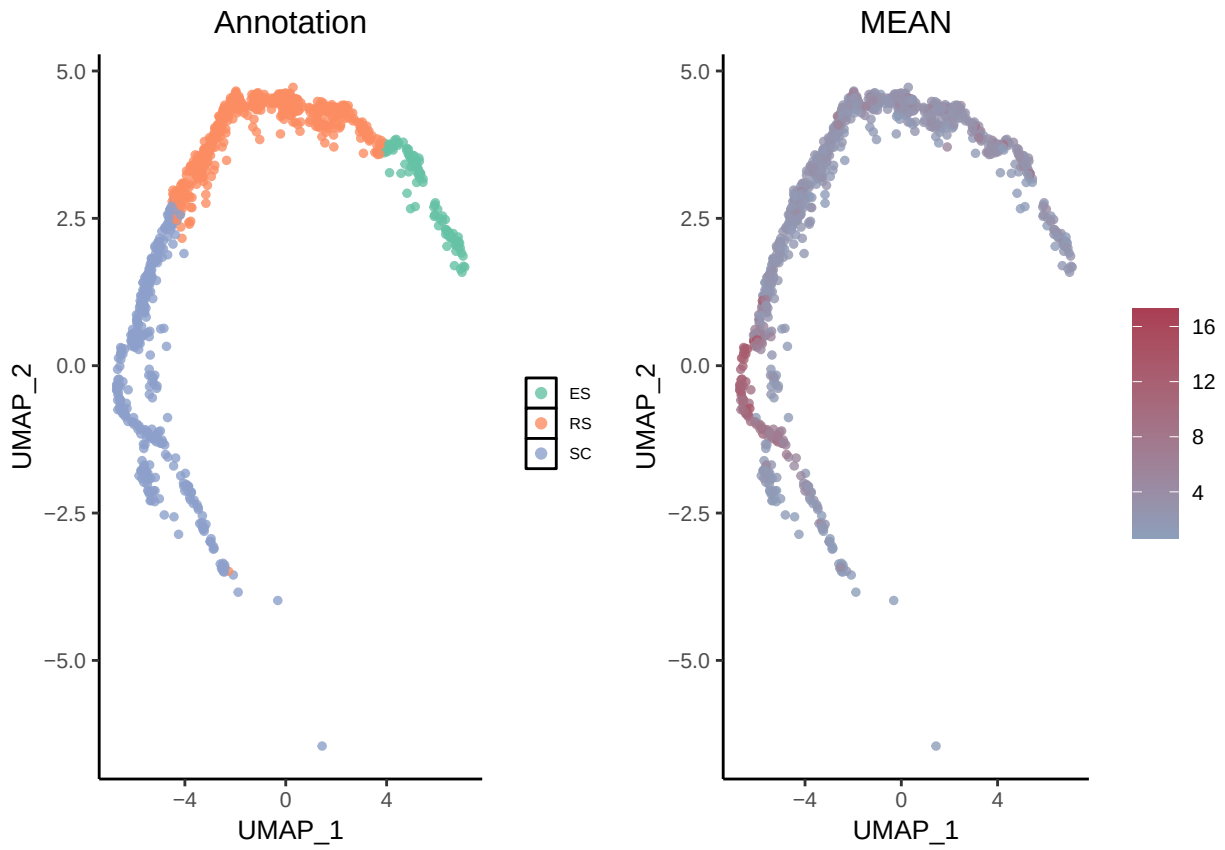
```
##                  orig.ident nCount_RNA nFeature_RNA RNA_snn_res.0.5
## AAACCTGAGCTTATCG       gene      23617         5061               9
## AAACCTGGTTGAGTTC       gene      19555         4802               9
## AAACCTGTCAACGAAA       gene      23467         5009               8
## AAACGGGCACAGGTTT       gene      28832         5484               8
## AAACGGGTCATTTGGG       gene      18931         4819               8
```

```
## AAACGGGTCCTCATTA       gene      15734         3855               8
##                   seurat_clusters celltype       UMAP_1      UMAP_2
## AAACCTGAGCTTATCG                9       RS  0.361751856 4.528803031
## AAACCTGGTTGAGTTC                9       RS -0.119255482 4.563224952
## AAACCTGTCAACGAAA                8       RS  3.023034156 4.074635188
## AAACGGGCACAGGTTT                8       RS  3.322863163  3.81046788
## AAACGGGTCATTTGGG                8       ES   4.73071772 3.419416826
## AAACGGGTCCTCATTA                8       ES  5.306060375 3.274766843
##                              barcode
## AAACCTGAGCTTATCG AAACCTGAGCTTATCG
## AAACCTGGTTGAGTTC AAACCTGGTTGAGTTC
## AAACCTGTCAACGAAA AAACCTGTCAACGAAA
## AAACGGGCACAGGTTT AAACGGGCACAGGTTT
## AAACGGGTCATTTGGG AAACGGGTCATTTGGG
## AAACGGGTCCTCATTA AAACGGGTCCTCATTA
```

Since the dataset already contains cell coordinates of UMAP, it is easy to view the 2D-embeddings of this dataset with vizAPA.

```
vizUMAP(scPACds, group='celltype', xcol='UMAP_1', ycol='UMAP_2')
```

```
## vizUMAP: group=celltype, x=UMAP_1, y=UMAP_2
```



`vizStats` to summarize APA usages across cell categories

`vizStats` draws different types of plots, including boxplot, violin plot, dot plot, and bubble plot, to show coordinates and expression (pA count or APA ratio) of given pAs or pAs in a gene across different conditions (e.g., cell types).

Here is an example to plot a boxplot to show the expression levels of pAs in a given gene across cell types.

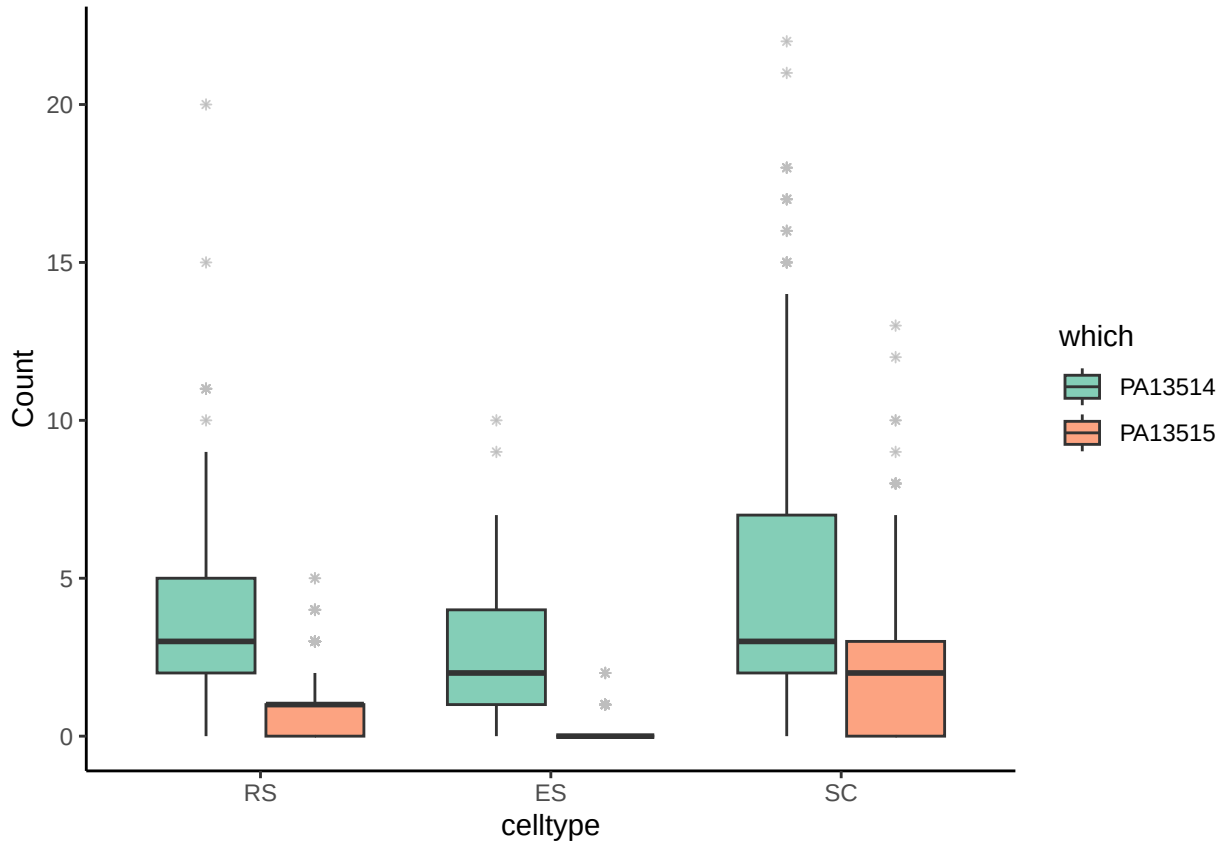
We chose an example gene Odf4 (252868, ENSMUSG00000032921) for demonstration.

```
## first, we check the gene column in the anno slot
head(scPACds@anno[, c('chr','strand','coord','gene','ftr')])
```

```
##           chr strand    coord      gene  ftr
## PA6062  chr17      +  6607918 100040531 3UTR
## PA15501 chr17      -  6648938 100040531 3UTR
## PA6073  chr17      + 13376977 100041352 3UTR
## PA6074  chr17      + 13377115 100041352 3UTR
## PA6072  chr17      + 13377893 100041352 3UTR
## PA15866 chr17      - 14964150 100041639 3UTR
```

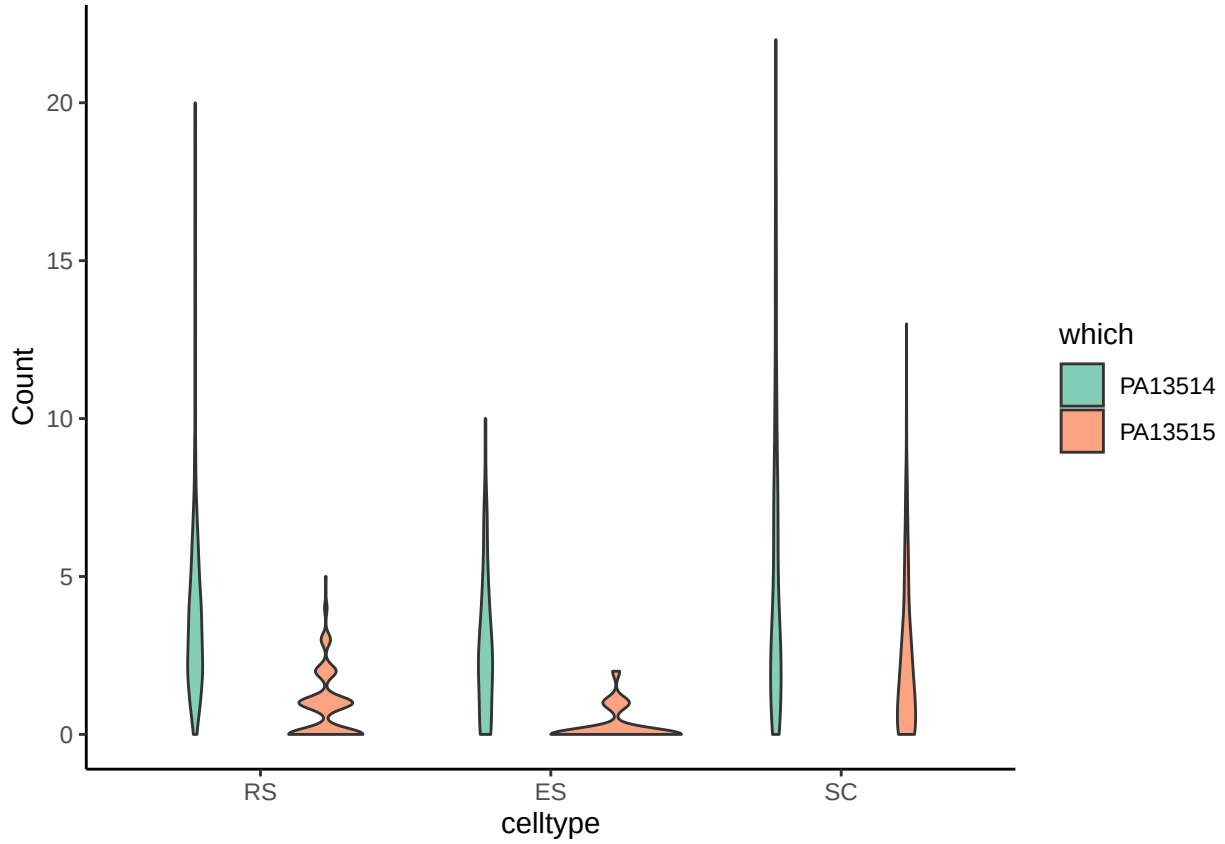
```
## the gene ID is entrez id, so we use 252868 instead of Odf4
gene='252868'

# plot a boxplot to compare the pA usage of this gene in different cell types
vizStats(scPACds, group='celltype', gene=gene, PAs=NULL, figType="box")
```

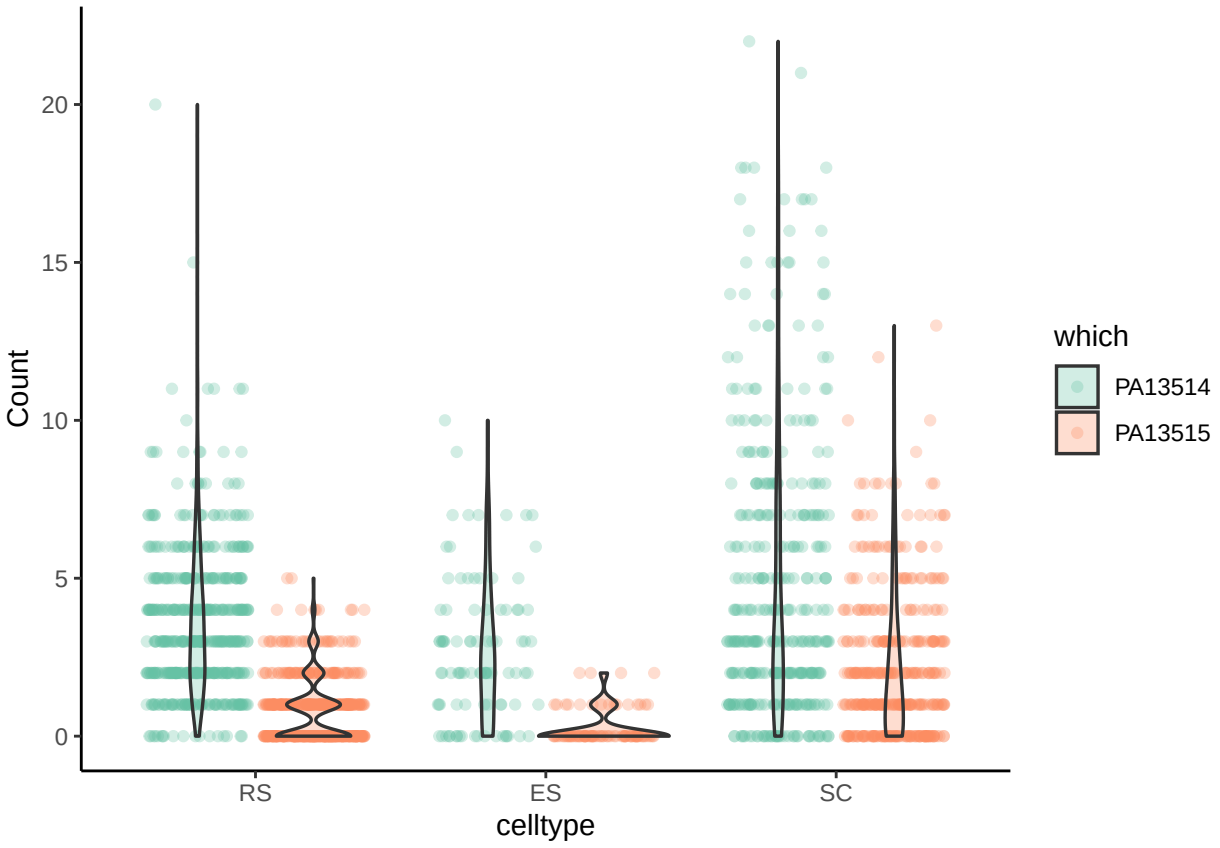


Plot other types of plots.

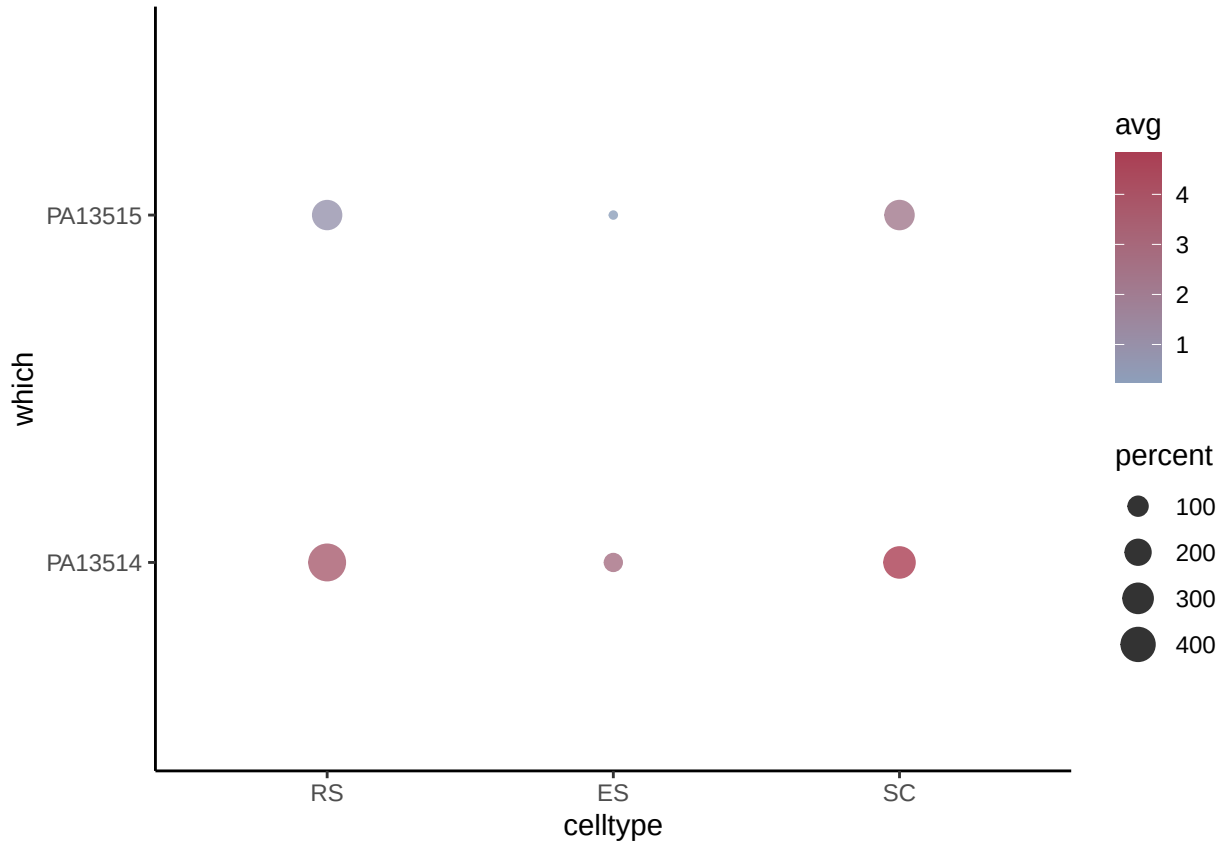
```
# violin plot
vizStats(scPACds, group='celltype', gene=gene, PAs=NULL, figType="violin")
```




```
# violin plot with dots
vizStats(scPACds, group='celltype', gene=gene, PAs=NULL, figType="dot")
```

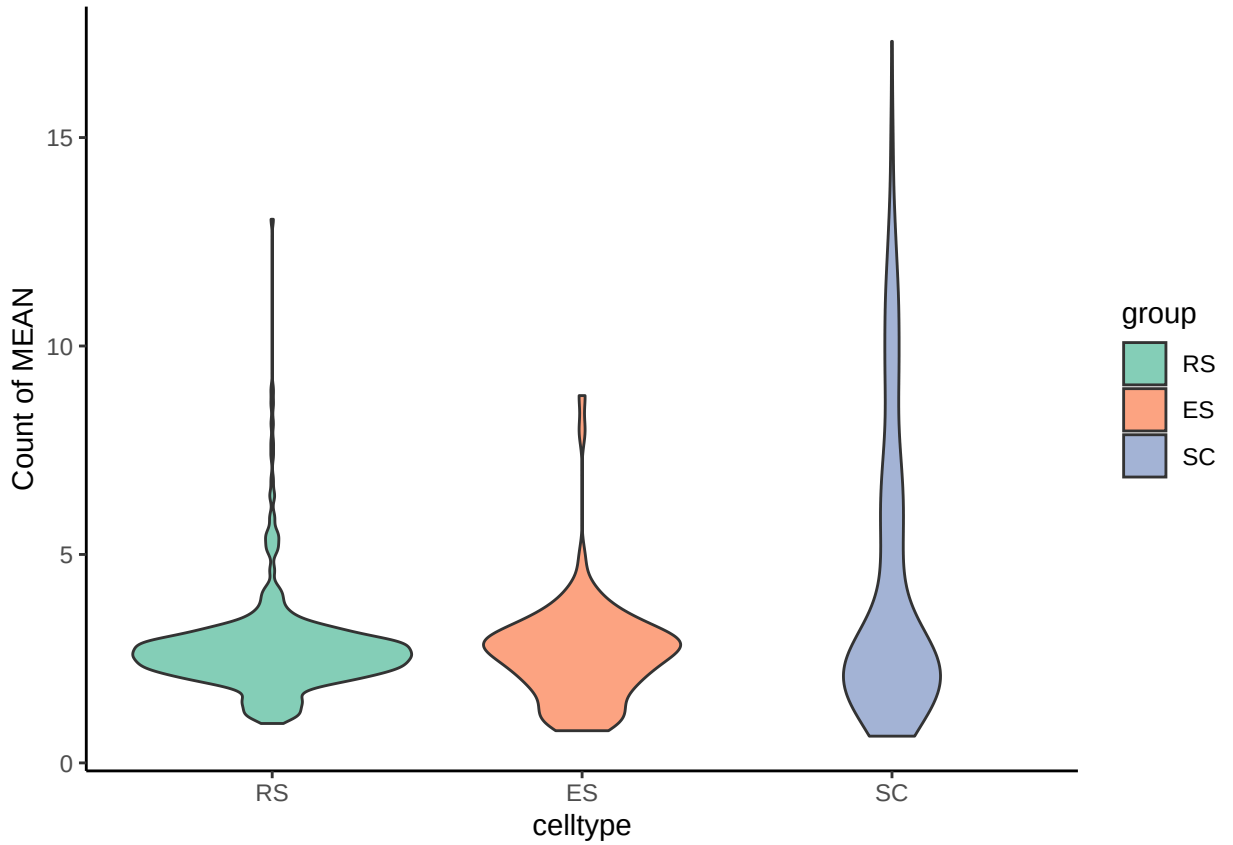



```
# bubble plot
vizStats(scPACds, group='celltype', gene=gene, PAs=NULL, figType="bubble")
```



If no pA or gene is provided, then it is to plot the mean of all pAs (if it is a pA matrix) or genes (if it is a gene or APA index matrix) in the PACds. Here the PACds is a pA-expression matrix, so `vizStats` plots the mean value of all pAs across cell types.

```
vizStats(scPACds, group='celltype', figType="violin")
```



vizUMAP to plot 2D-embeddings

vizUMAP plots a UMAP plot where each point is a cell and it's positioned based on the cell embedding determined by the reduction technique.

```
gene='252868'
# First, we check the coordinate labels of the 2D-embedding.
# For this data, the labels are UMAP_1 and UMAP_2.
colnames(scPACds@colData)
```

```
## [1] "orig.ident"      "nCount_RNA"      "nFeature_RNA"    "RNA_snn_res.0.5"
## [5] "seurat_clusters" "celltype"        "UMAP_1"          "UMAP_2"
## [9] "barcode"
```

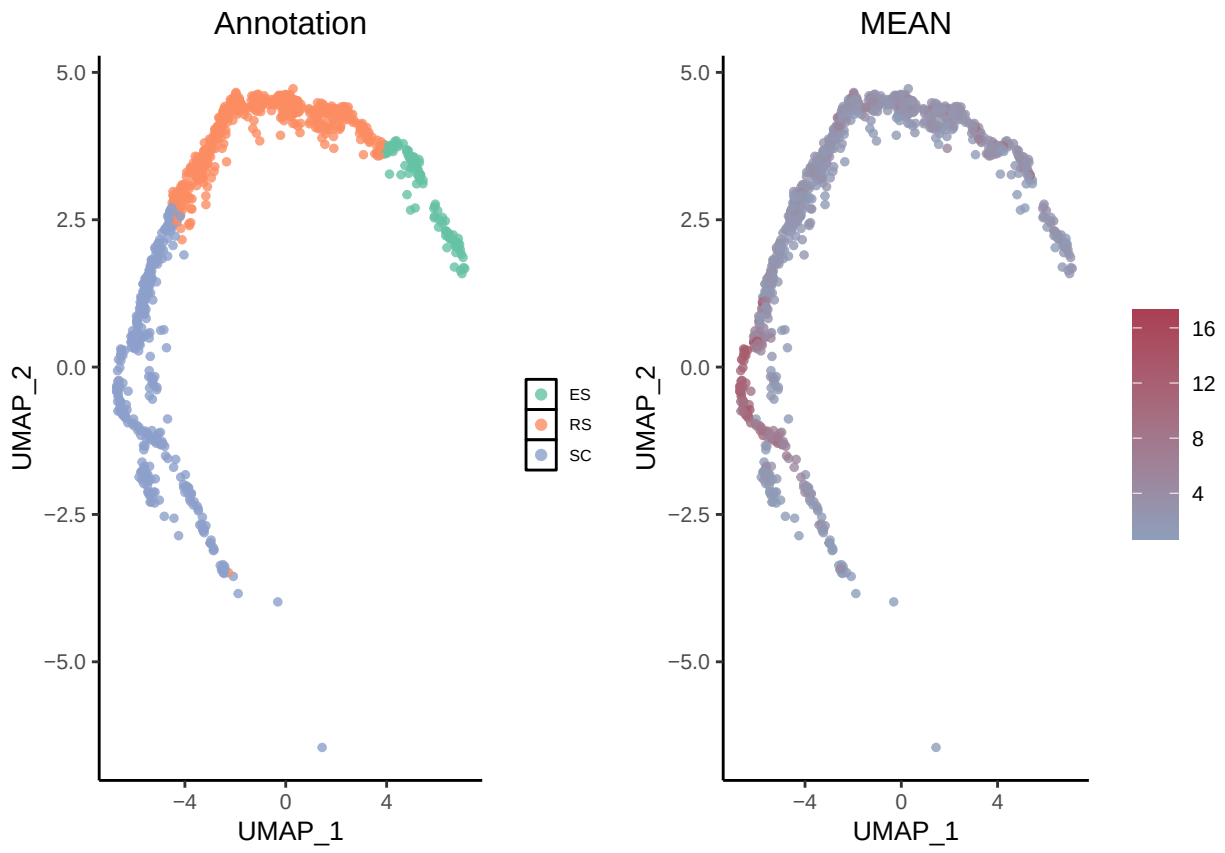
```
head(scPACds@colData)
```

```
##                  orig.ident nCount_RNA nFeature_RNA RNA_snn_res.0.5
## AAACCTGAGCTTATCG       gene      23617         5061               9
## AAACCTGGTTGAGTTC       gene      19555         4802               9
## AAACCTGTCAACGAAA       gene      23467         5009               8
## AAACGGGCACAGGTTT       gene      28832         5484               8
## AAACGGGTCATTTGGG       gene      18931         4819               8
## AAACGGGTCCTCATTA       gene      15734         3855               8
```

```
##                   seurat_clusters celltype       UMAP_1      UMAP_2
## AAACCTGAGCTTATCG                9       RS  0.361751856 4.528803031
## AAACCTGGTTGAGTTC                9       RS -0.119255482 4.563224952
## AAACCTGTCAACGAAA                8       RS  3.023034156 4.074635188
## AAACGGGCACAGGTTT                8       RS  3.322863163  3.81046788
## AAACGGGTCATTTGGG                8       ES   4.73071772 3.419416826
## AAACGGGTCCTCATTA                8       ES  5.306060375 3.274766843
##                             barcode
## AAACCTGAGCTTATCG AAACCTGAGCTTATCG
## AAACCTGGTTGAGTTC AAACCTGGTTGAGTTC
## AAACCTGTCAACGAAA AAACCTGTCAACGAAA
## AAACGGGCACAGGTTT AAACGGGCACAGGTTT
## AAACGGGTCATTTGGG AAACGGGTCATTTGGG
## AAACGGGTCCTCATTA AAACGGGTCCTCATTA
```

```
# Plot the UMAP plot showing cell clusters and another UMAP plot overlaying
# with the mean expression value of pAs in each cell.
vizUMAP(scPACds, group='celltype', xcol='UMAP_1', ycol='UMAP_2')
```

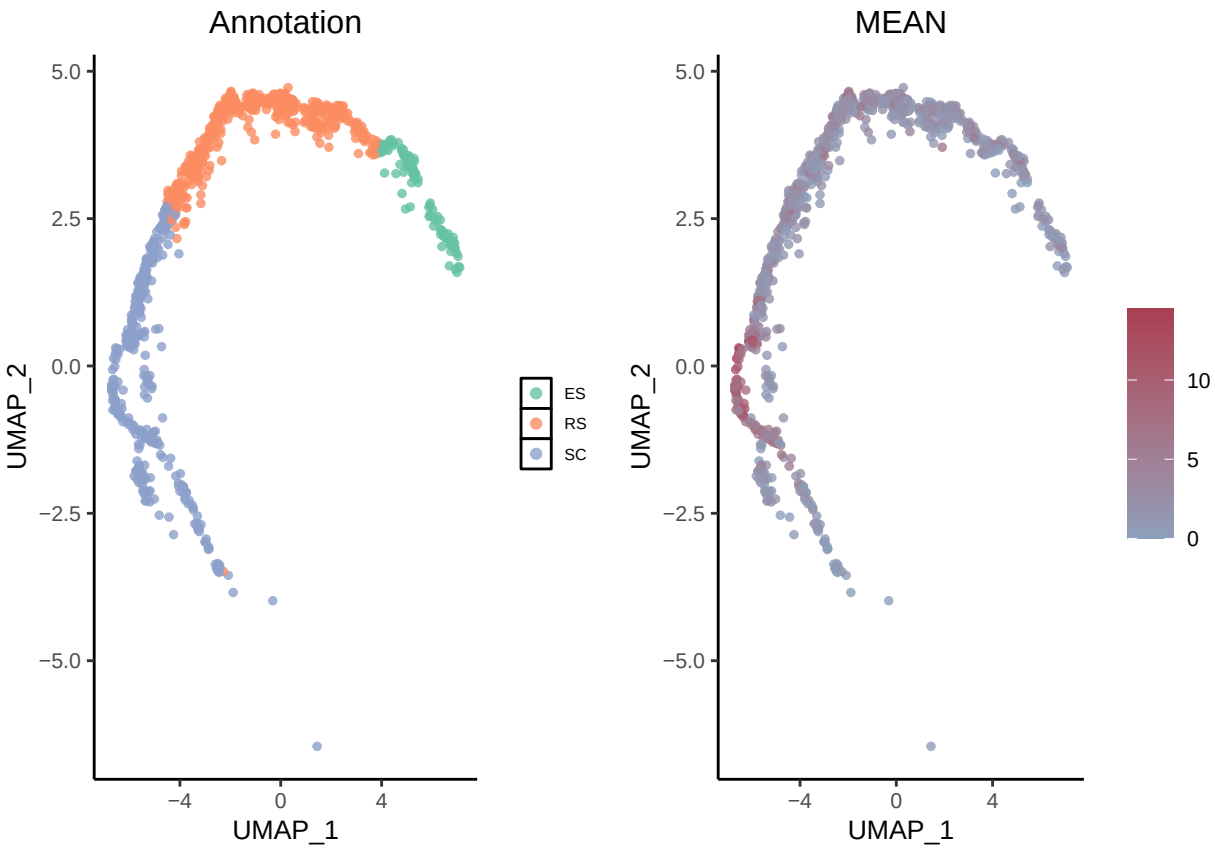
```
## vizUMAP: group=celltype, x=UMAP_1, y=UMAP_2
```



Providing a gene id or a list of genes in the gene column of the PACdataset, we can plot a UMAP overlaying with the mean expression value of the gene(s).

```
## Here we plot the Odf4 gene,
## overlaying the mean value of all pAs in this gene
vizUMAP(scPACds,
        group='celltype', xcol='UMAP_1',
        ycol='UMAP_2', genes=gene)
```

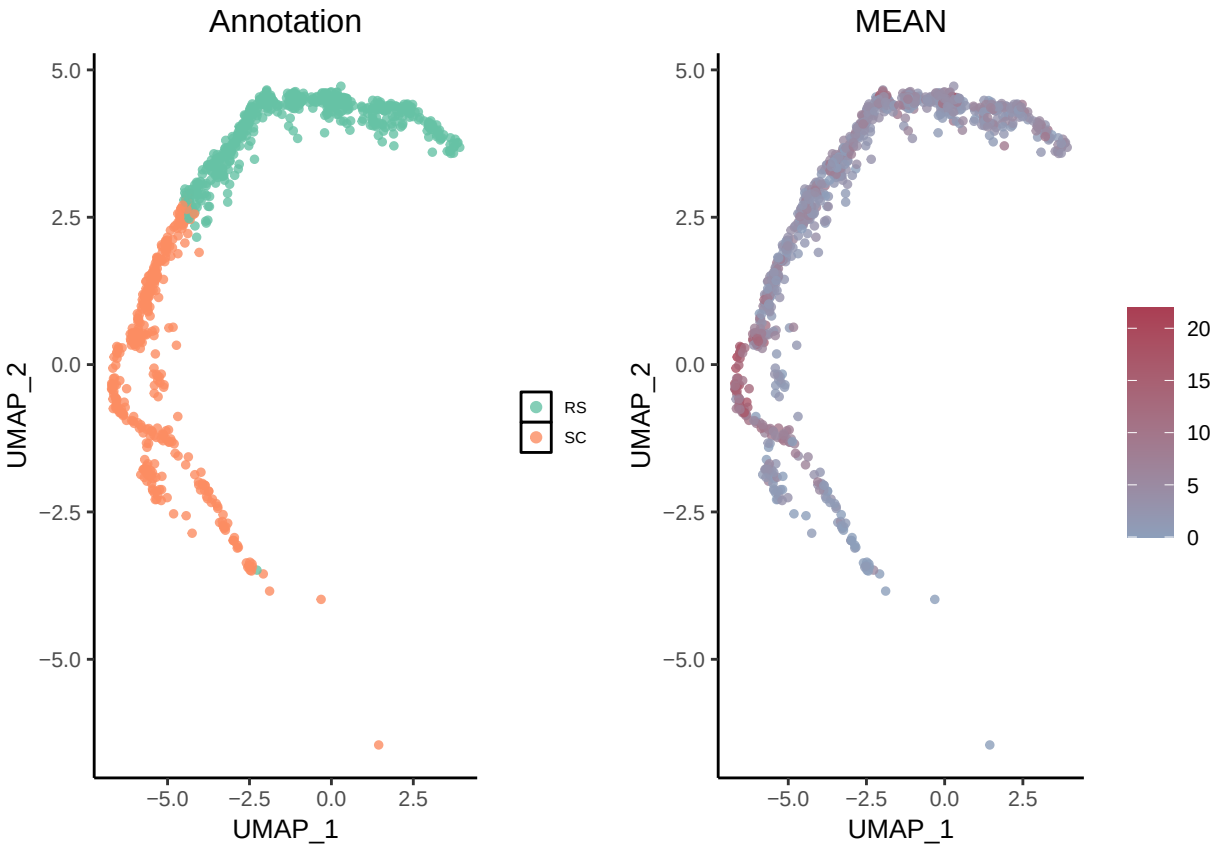
```
## vizUMAP: group=celltype, x=UMAP_1, y=UMAP_2
```



Similarly, we can provide ids of pAs corresponding to the rownames in the PACds instead of genes.

```
# here we only overlay one pA in Odf4 gene
# here only show two cell types by specifying selGroup
PAids='PA13514'
vizUMAP(scPACds, group='celltype', xcol='UMAP_1', ycol='UMAP_2', selGroups=c('SC','RS'), PAs=PAids)
```

```
## vizUMAP: group=celltype, x=UMAP_1, y=UMAP_2
```



`vizAPAmarkers` to visualize APA markers across cell categories

An APA marker is an APA gene with differential APA usage between two pAs in the 3'UTR of the gene. Here we calculate the relative usage of distal pA (RUD) to represent the APA usage of each gene. An larger RUD means the longer 3'UTR. Note: `getAPAindexPACds` only implements the RUD index in `movAPA`, users can use `movAPA::movAPAindex` for more types of APA index.

```
# First, calculate the RUD index for each gene.
# Only genes with 3'UTR APA can be used for RUD calculation.
iPACds=getAPAindexPACds(scPACds, choose2PA='PD')
head(iPACds[, 1:10])
```

```
## PAC# 6
## sample# 10
## AAACCTGAGCTTATCG AAACCTGGTTGAGTTC AAACCTGTCAACGAAA AAACGGGCACAGGTTT AAACGGGTCATTTGGG ...
## groups:
## @colData...[10 x 9]
##                  orig.ident nCount_RNA nFeature_RNA RNA_snn_res.0.5
## AAACCTGAGCTTATCG       gene      23617         5061               9
## AAACCTGGTTGAGTTC       gene      19555         4802               9
##                  seurat_clusters celltype       UMAP_1      UMAP_2
## AAACCTGAGCTTATCG               9       RS  0.361751856 4.528803031
## AAACCTGGTTGAGTTC               9       RS -0.119255482 4.563224952
##                          barcode
```

```
## AAACCTGAGCTTATCG AAACCTGAGCTTATCG
## AAACCTGGTTGAGTTC AAACCTGGTTGAGTTC
## @counts...[6 x 10]
## 2 x 10 sparse Matrix of class "dgCMatrix"
##
## 100040531 0.4871795 0.5510204 0.3888889 0.6206897 0.8 0.3333333 0.5434783
## 100041352 .         .         .         .         .   .         .
##
## 100040531 0.6027778 0.4788732 0.4814815
## 100041352 .         .         .
## @colData...[10 x 9]
##                   orig.ident nCount_RNA nFeature_RNA RNA_snn_res.0.5
## AAACCTGAGCTTATCG       gene      23617         5061               9
## AAACCTGGTTGAGTTC       gene      19555         4802               9
##                   seurat_clusters celltype       UMAP_1      UMAP_2
## AAACCTGAGCTTATCG               9       RS  0.361751856 4.528803031
## AAACCTGGTTGAGTTC               9       RS -0.119255482 4.563224952
##                            barcode
## AAACCTGAGCTTATCG AAACCTGAGCTTATCG
## AAACCTGGTTGAGTTC AAACCTGGTTGAGTTC
## @anno...[6 x 1]
##                gene
## 100040531 100040531
## 100041352 100041352
## @supp...[2]
## dataType row
```

Then we obtain APA markers by wilcox.test for each pair of cell types.

```
## obtain APA markers by wilcox.test for each pair of cell types
m=getAPAmarkers(iPACds,  group='celltype', everyPair = TRUE)
```

```
## PACds row = gene, PACds dataType = ratio
## It seems that PACds is APA ratio, will apply wilcox-test on the APA index to get DE APA events (each
```

```
## show marker numbers
table(m$cluster1, m$cluster2)
```

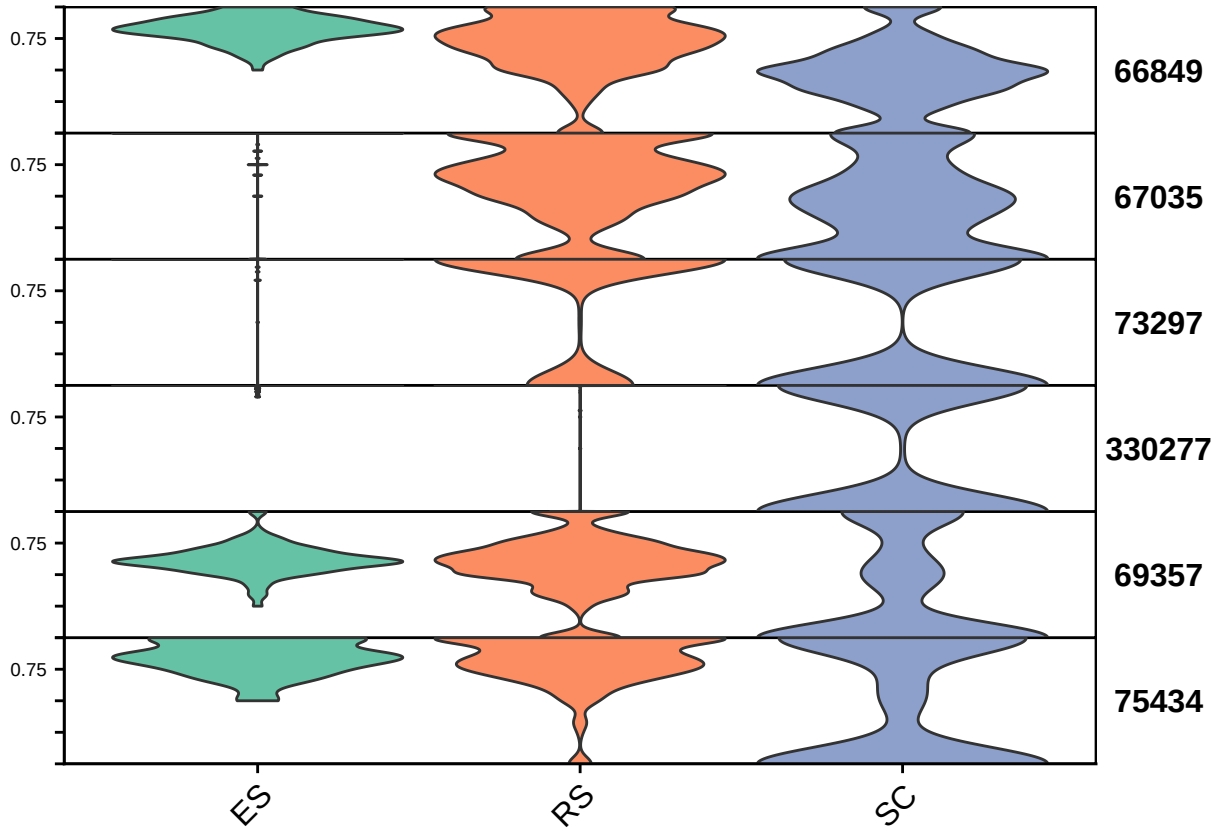
```
##
##      ES RS SC
##   ES  0  0  6
##   RS 33  0  3
##   SC  0  0  0
```

```
## show marker details
head(m)
```

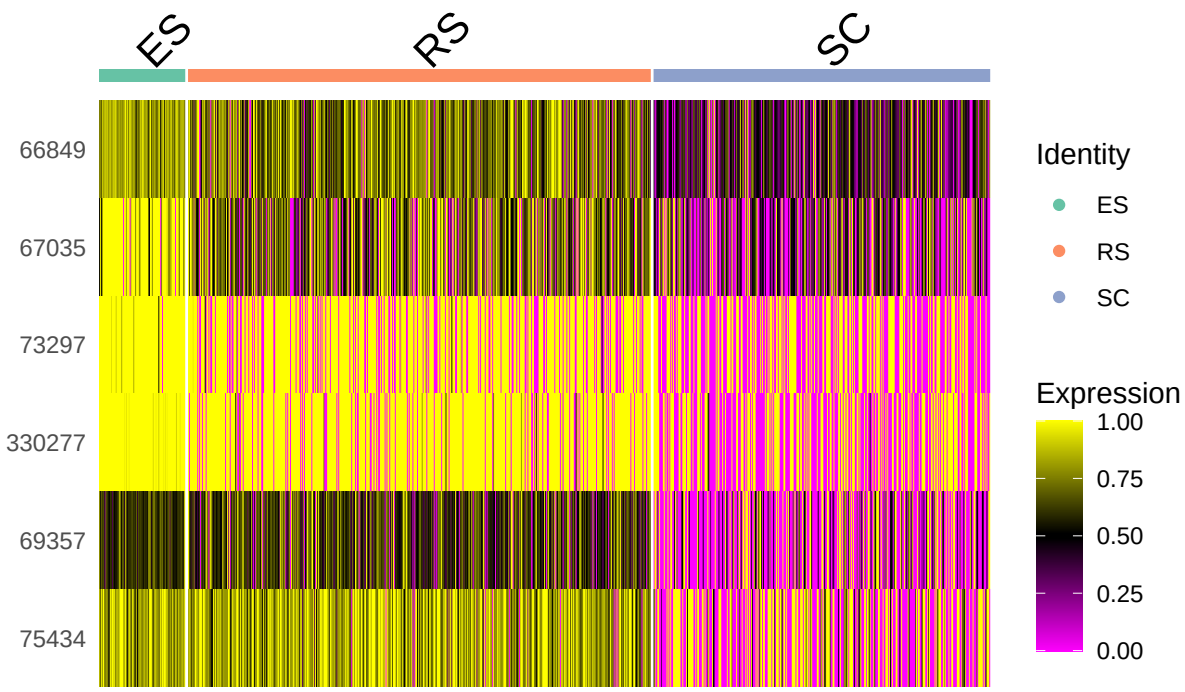
```
##                p_val avg_log2FC pct.1 pct.2    p_val_adj cluster1 cluster2
## 66849   7.500958e-33  0.3242110 1.000 0.873 3.097896e-30       ES       SC
## 67035   9.399927e-28  0.4355351 0.957 0.678 3.882170e-25       ES       SC
## 732971  7.115414e-19  0.4493426 0.989 0.449 2.938666e-16       ES       SC
## 3302771 7.989574e-17  0.4478787 1.000 0.466 3.299694e-14       ES       SC
```

```
## 69357   1.768628e-11  0.2642759 1.000 0.444 7.304434e-09       ES       SC
## 754341  2.128367e-05  0.3191785 1.000 0.518 8.790158e-03       ES       SC
##           rowid
## 66849     66849
## 67035     67035
## 732971    73297
## 3302771 330277
## 69357     69357
## 754341    75434
```

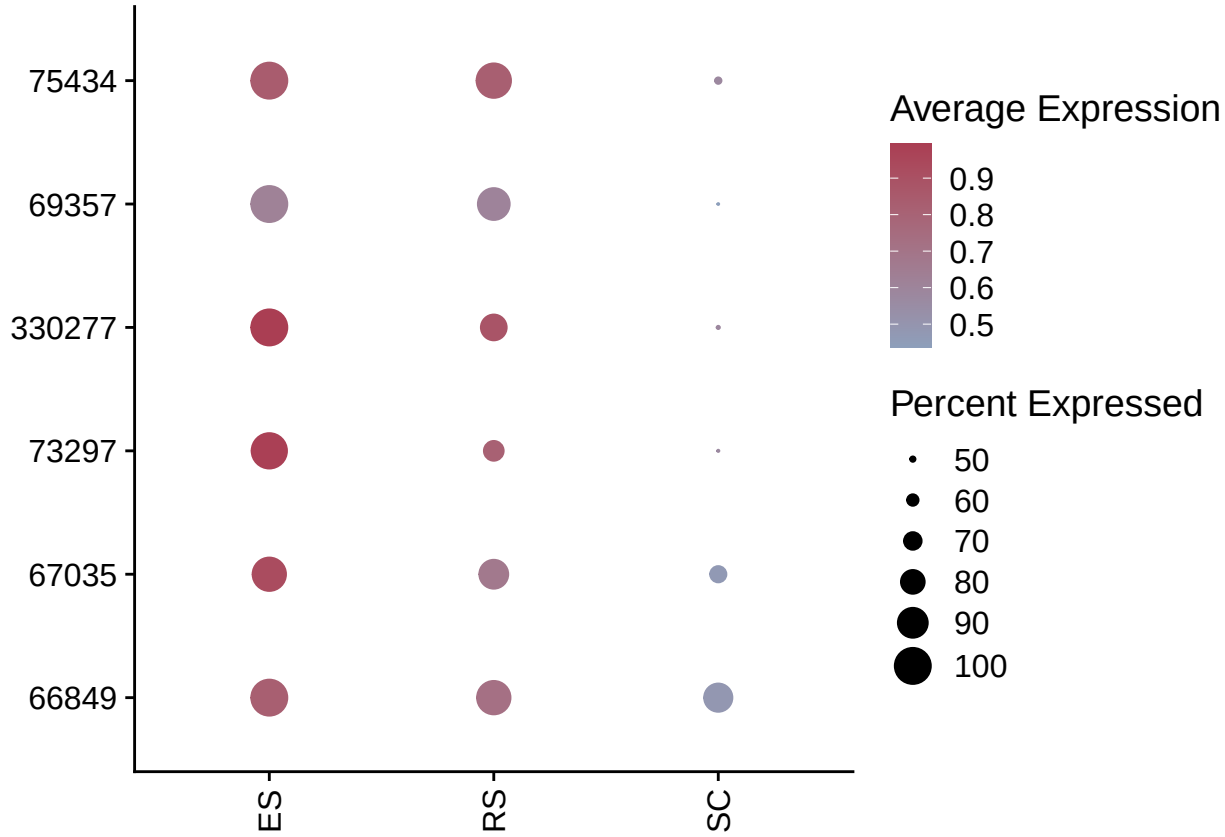
```
# Visualize the top 6 APA markers, showing all the three cell types
vizAPAMarkers(iPACds, group='celltype',
              markers=m$rowid[1:6],
              figType = 'violin')
```



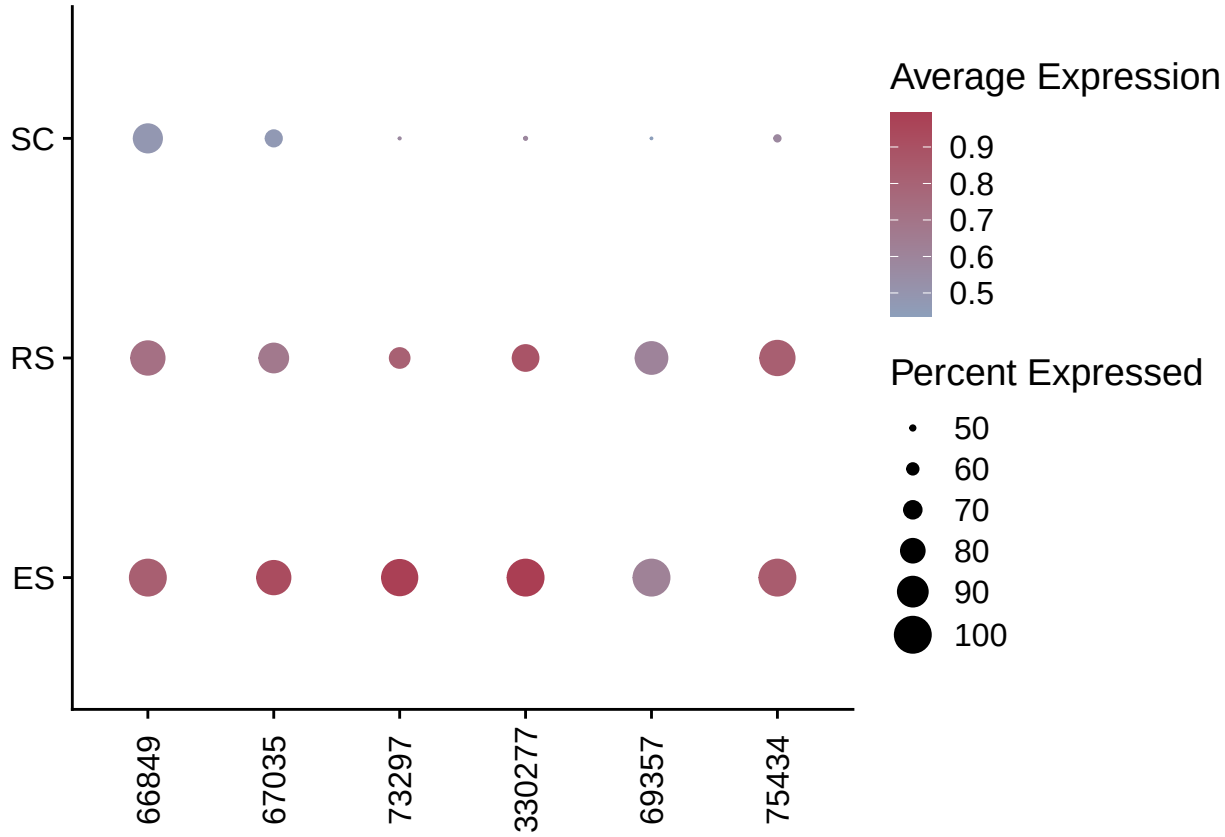

```
# Plot a heatmap for APA markers
vizAPAMarkers(iPACds, group='celltype',
              markers=m$rowid[1:6],
              figType = 'heatmap')
```




```
# Plot a bubble plot for markers
vizAPAMarkers(iPACds, group='celltype',
              markers=m$rowid[1:6],
              figType = 'bubble')
```

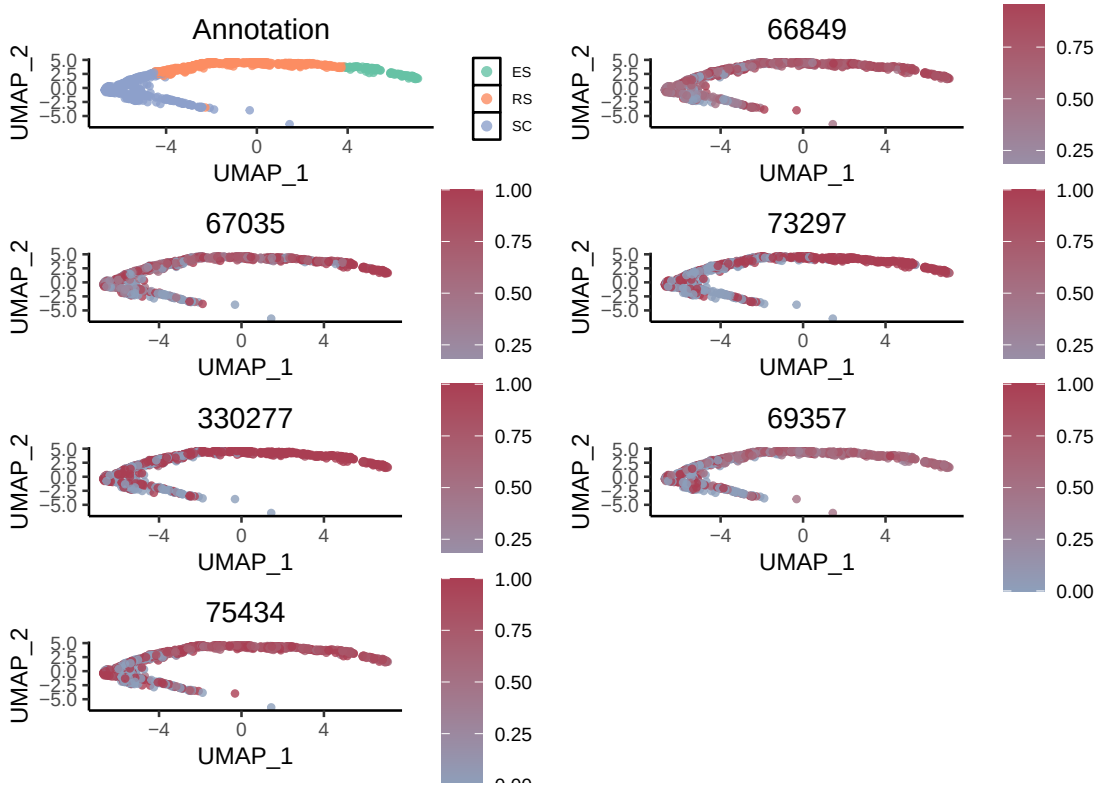



```
# Switch the x/y axis of the bubble plot
vizAPAMarkers(iPACds, group='celltype',
              markers=m$rowid[1:6],
              figType = 'bubble',
              statTheme=list(xgroup=FALSE))
```



Next, we can plot UMAP for these APA markers.

```
# Plot the UMAP plot
vizAPAMarkers(iPACds,
              group='celltype',
              markers=m$rowid[1:6],
              figType="umap",
              umap.x='UMAP_1', umap.y='UMAP_2')
```



The above examples detect markers between every pair of cell types. It is also possible to compare one cell type with all other cells.

```
# Detect markers between ES and all other cells.
m=getAPAmarkers(scPACds,  group='celltype', cluster1='ES')
```

```
## PACds row = PA, PACds dataType = count
```

```
## Warning in getAPAmarkers(scPACds, group = "celltype", cluster1 = "ES"): It seems that PACds is pA cou
## If you want APA markers, please use getAPAindexPACds() to transform your PACds to APA ratio.
```

```
head(m)
```

```
##                p_val avg_log2FC pct.1 pct.2    p_val_adj cluster1 cluster2
## PA2955  7.867291e-71   3.046038 1.000 0.311 7.662741e-68       ES   non-ES
## PA13906 4.219130e-60   1.330360 0.828 0.147 4.109432e-57       ES   non-ES
## PA9536  1.747775e-56   2.707196 0.989 0.454 1.702333e-53       ES   non-ES
## PA6248  8.215108e-56   2.500012 1.000 0.558 8.001515e-53       ES   non-ES
## PA4014  5.989791e-55   2.873583 1.000 0.588 5.834056e-52       ES   non-ES
## PA7661  9.854346e-55   3.625598 1.000 1.000 9.598133e-52       ES   non-ES
##           rowid
## PA2955   PA2955
## PA13906 PA13906
## PA9536   PA9536
## PA6248   PA6248
## PA4014   PA4014
## PA7661   PA7661
```

```
table(m$cluster1, m$cluster2)
```

```
##
##      non-ES
##   ES    124
```

vizTracks to plot gene model, pAs and BAM tracks

One unique feature of vizAPA is plotting IGV-like plot, including gene models, pA positions and BAM coverages.

Prepare BAM files

The BAM files and the corresponding index (.bai) files for the following analysis can be downloaded from the GitHub site of vizAPA: mouse.sperm.bam. For demonstration, these BAM files contain only five genes [252868(Odf4), 107566(Arl2bp), 67078 (Pgp), 100041639 (Dynlt2a2), 14202 (Fhl4)] extracted from the original BAM (accession number: GSM280334).

```
#Create the list of BAM files
bam.files=c("dedup_GSM2803334.ES.mini.sorted.bam",
            "dedup_GSM2803334.RS.mini.sorted.bam",
            "dedup_GSM2803334.SC.mini.sorted.bam")
bam.groups=c("ES","RS","SC")
bam.labels=c("ES" ,"RS","SC")
bam.path='./'

bams<-readBAMFileNames(bam.files=bam.files,
                       bam.path=bam.path,
                       bam.labels = bam.labels,
                       bam.groups = bam.groups)

bams
```

```
##                                fileName group label
## 1 ./dedup_GSM2803334.ES.mini.sorted.bam    ES    ES
## 2 ./dedup_GSM2803334.RS.mini.sorted.bam    RS    RS
## 3 ./dedup_GSM2803334.SC.mini.sorted.bam    SC    SC
```

Load genome annotation to an annoHub

In vizAPA, the genome annotation is used for the track plots to show gene models in a genomic region. The genome annotation could be retrieved from several sources, including gff3/gtf file, TxDb, EnsDb, BioMart, and OrganismDb. Users can provide one or more annotation sources.

- OrganismDb object: recommended, support gene symbols and other combination of columns as label.
- TxDb object: don't support gene symbol labeling.
- EnsDb object: supports gene symbol labeling, filtering etc.

Object type	example package	name contents
OrgDb	org.Hs.eg.db	gene based info. for Homo sapiens
TxDb	TxDb.Hsapiens.UCSC.hg19.knownGene	transcriptome ranges for Homo sapiens
OrganismDb	Homo.sapiens	composite information for Homo sapiens
BSgenome	BSgenome.Hsapiens.UCSC.hg19	genome sequence for Homo sapiens

We can make an `annoHub` object storing different annotation sources, which can be used by many functions in vizAPA. In the following, we used the TxDB annotation for demonstration.

```
annoSource=new("annoHub")
library(TxDb.Mmusculus.UCSC.mm10.knownGene, quietly = TRUE)
txdb=TxDb.Mmusculus.UCSC.mm10.knownGene
annoSource=addAnno(annoSource, txdb)
annoSource
```

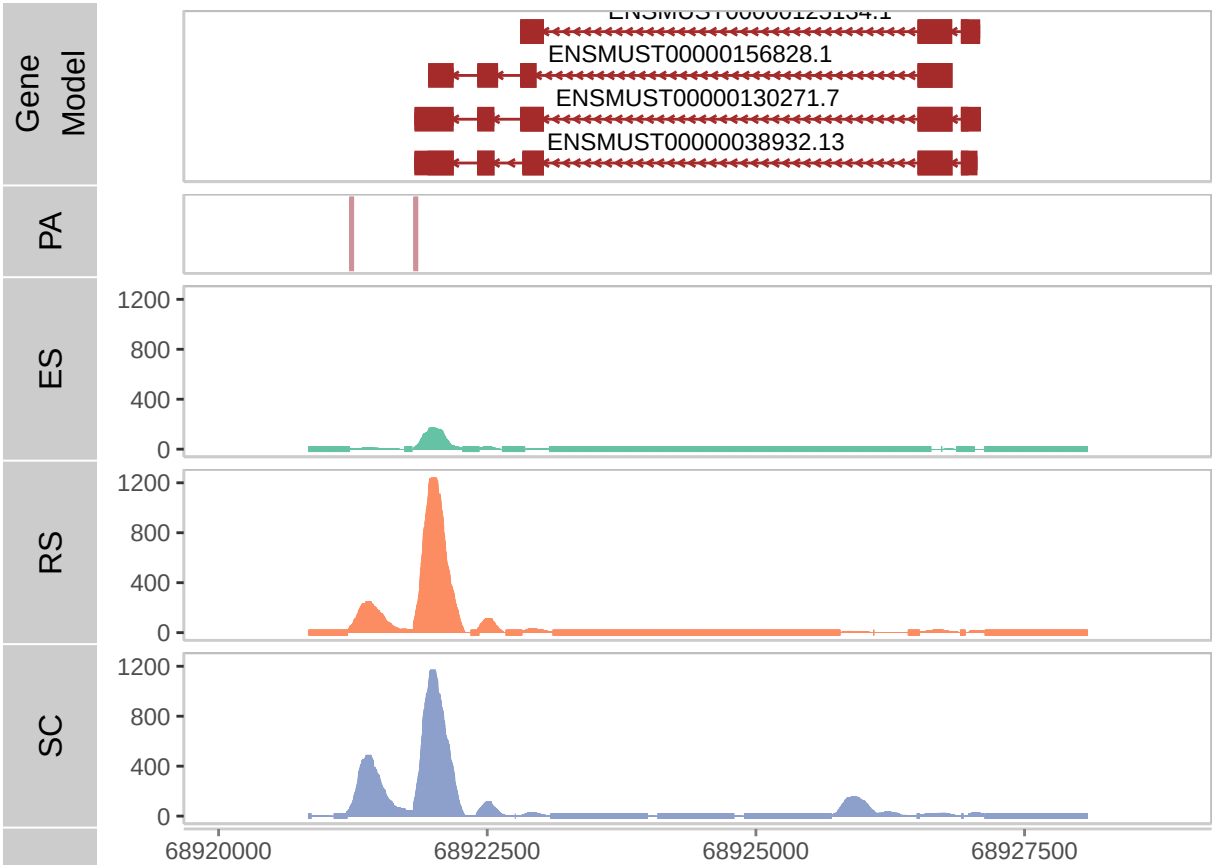
```
## @annos [annotation sources]:
## txdb=TxDb
## @defaultAnno:
##  txdb
```

Plot tracks for a specified gene

Having prepared the `PACdataset`, `annoHub`, and BAM files, we can easily plot an example gene (here is the Odf4 gene), with gene model, pA coordinates, and BAM coverages.

```
vizTracks(gene=gene,
          bams=bams,
          PACds.list=list(PA=scPACds),
          PA.show=c("pos"),
          annoSource=annoSource,
          PA.columns="coord", PA.width=10,
          space5=1000, space3=1000)
```

```
## Plot tracks for region: chr11:-:68920835:68928081
## Get gene model track from annoSource[ txdb ]...
## Get PACds track...
## chr11:-:68920835:68928081
## Get BAM tracks...
```



Session information

The session information records the versions of all the packages used in the generation of the present document.

```
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22621)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=Chinese (Simplified)_China.utf8
## [2] LC_CTYPE=Chinese (Simplified)_China.utf8
## [3] LC_MONETARY=Chinese (Simplified)_China.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=Chinese (Simplified)_China.utf8
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
```

```
## other attached packages:
##  [1] TxDb.Mmusculus.UCSC.mm10.knownGene_3.10.0
##  [2] GenomicFeatures_1.50.2
##  [3] AnnotationDbi_1.60.0
##  [4] Biobase_2.58.0
##  [5] GenomicRanges_1.50.1
##  [6] GenomeInfoDb_1.34.9
##  [7] IRanges_2.32.0
##  [8] S4Vectors_0.36.0
##  [9] BiocGenerics_0.44.0
## [10] vizAPA_0.1.0
##
## loaded via a namespace (and not attached):
##   [1] utf8_1.2.2                  spatstat.explore_3.0-5
##   [3] reticulate_1.30             tidyselect_1.2.0
##   [5] movAPA_2.0                  RSQLite_2.2.18
##   [7] htmlwidgets_1.5.4           grid_4.2.2
##   [9] BiocParallel_1.32.1         Rtsne_0.16
##  [11] munsell_0.5.0               codetools_0.2-18
##  [13] ica_1.0-3                   interp_1.1-3
##  [15] future_1.30.0               miniUI_0.1.1.1
##  [17] withr_2.5.0                 spatstat.random_3.0-1
##  [19] colorspace_2.0-3            progressr_0.12.0
##  [21] filelock_1.0.2              OrganismDbi_1.40.0
##  [23] highr_0.9                   knitr_1.41
##  [25] rstudioapi_0.14             Seurat_4.3.0
##  [27] ROCR_1.0-11                 tensor_1.5
##  [29] ggsignif_0.6.4              listenv_0.9.0
##  [31] MatrixGenerics_1.10.0       labeling_0.4.2
##  [33] GenomeInfoDbData_1.2.9      polyclip_1.10-4
##  [35] bit64_4.0.5                 farver_2.1.1
##  [37] parallelly_1.33.0           vctrs_0.5.1
##  [39] generics_0.1.3              xfun_0.35
##  [41] biovizBase_1.46.0           BiocFileCache_2.6.0
##  [43] R6_2.5.1                    AnnotationFilter_1.22.0
##  [45] spatstat.utils_3.0-1        bitops_1.0-7
##  [47] cachem_1.0.6                reshape_0.8.9
##  [49] DelayedArray_0.24.0         assertthat_0.2.1
##  [51] promises_1.2.0.1            BiocIO_1.8.0
##  [53] scales_1.2.1                nnet_7.3-18
##  [55] gtable_0.3.1                globals_0.16.2
##  [57] goftest_1.2-3               ggbio_1.46.0
##  [59] ensembldb_2.22.0            rlang_1.0.6
##  [61] splines_4.2.2               rtracklayer_1.58.0
##  [63] rstatix_0.7.1               lazyeval_0.2.2
##  [65] dichromat_2.0-0.1           spatstat.geom_3.0-3
##  [67] broom_1.0.2                 checkmate_2.1.0
##  [69] BiocManager_1.30.19         yaml_2.3.6
##  [71] reshape2_1.4.4              abind_1.4-5
##  [73] backports_1.4.1             httpuv_1.6.6
##  [75] Hmisc_5.0-0                 RBGL_1.74.0
##  [77] tools_4.2.2                 ggplot2_3.4.0
##  [79] ellipsis_0.3.2              RColorBrewer_1.1-3
##  [81] ggridges_0.5.4              Rcpp_1.0.9
```

```
##  [83] plyr_1.8.8                  base64enc_0.1-3
##  [85] progress_1.2.2              zlibbioc_1.44.0
##  [87] purrr_0.3.5                 RCurl_1.98-1.9
##  [89] prettyunits_1.1.1           ggpubr_0.5.0
##  [91] rpart_4.1.19                deldir_1.0-6
##  [93] pbapply_1.6-0               cowplot_1.1.1
##  [95] zoo_1.8-11                  SeuratObject_4.1.3
##  [97] SummarizedExperiment_1.28.0 ggrepel_0.9.2
##  [99] cluster_2.1.4               magrittr_2.0.3
## [101] scattermore_0.8             data.table_1.14.6
## [103] lmtest_0.9-40               RANN_2.6.1
## [105] ProtGenerics_1.30.0         fitdistrplus_1.1-8
## [107] matrixStats_0.63.0          patchwork_1.1.2
## [109] xtable_1.8-4                mime_0.12
## [111] hms_1.1.2                   evaluate_0.18
## [113] XML_3.99-0.12               jpeg_0.1-10
## [115] gridExtra_2.3               compiler_4.2.2
## [117] biomaRt_2.54.0              tibble_3.1.8
## [119] KernSmooth_2.23-20          crayon_1.5.2
## [121] htmltools_0.5.3             later_1.3.0
## [123] Formula_1.2-4               tidyr_1.2.1
## [125] DBI_1.1.3                   dbplyr_2.2.1
## [127] MASS_7.3-58.1               rappdirs_0.3.3
## [129] Matrix_1.5-3                car_3.1-1
## [131] cli_3.4.1                   parallel_4.2.2
## [133] igraph_1.3.5                pkgconfig_2.0.3
## [135] GenomicAlignments_1.34.0    foreign_0.8-83
## [137] sp_1.5-1                    spatstat.sparse_3.0-0
## [139] plotly_4.10.1               xml2_1.3.3
## [141] XVector_0.38.0              stringr_1.4.1
## [143] VariantAnnotation_1.44.0    digest_0.6.30
## [145] sctransform_0.3.5           RcppAnnoy_0.0.20
## [147] graph_1.76.0                spatstat.data_3.0-0
## [149] Biostrings_2.66.0           rmarkdown_2.18
## [151] leiden_0.4.3                htmlTable_2.4.1
## [153] uwot_0.1.14                 restfulr_0.0.15
## [155] curl_4.3.3                  shiny_1.7.3
## [157] Rsamtools_2.14.0            rjson_0.2.21
## [159] nlme_3.1-160                lifecycle_1.0.3
## [161] jsonlite_1.8.3              carData_3.0-5
## [163] limma_3.54.0                viridisLite_0.4.1
## [165] BSgenome_1.66.2             fansi_1.0.3
## [167] pillar_1.8.1                lattice_0.20-45
## [169] GGally_2.1.2                KEGGREST_1.38.0
## [171] fastmap_1.1.0               httr_1.4.4
## [173] survival_3.4-0              glue_1.6.2
## [175] png_0.1-7                   bit_4.0.5
## [177] stringi_1.7.8               blob_1.2.3
## [179] latticeExtra_0.6-30         memoise_2.0.1
## [181] dplyr_1.0.10                irlba_2.3.5.1
## [183] future.apply_1.10.0
```