Guided pseudotime reconstruction with TSCAN

Overview

The vignette demonstrates how to use TSCAN to infer cells' pseudotime ordering with the prior knowledge of the potential biological process. Standard single-cell analysis pipeline such as Seurat need to be run first to adequately preprocess the data. This is a semi-supervised and human-guided approach which has been widely used with the increasing complexity of single-cell genomic data.

This vignette uses the single-cell RNA-seq data from PBMCs processed by Seurat as an example, and constructs a pseudotime trajectory from naive CD4 T cells to memory CD4 T cells.

Installing packages

Use the following command to install the TSCAN package in R.

```
if (!require("devtools"))
  install.packages("devtools")
devtools::install_github("zji90/TSCAN")
```

This vignette uses Seurat object as an example. So if you haven't done so, install Seurat as well.

```
install.packages('Seurat')
```

Data download

The PBMC scRNA-seq has been processed by Seurat. The details of data processing is described in https://satijalab.org/seurat/articles/pbmc3k_tutorial.html. Download the Seurat object from https://www.dropbox.com/s/63gnlw45jf7cje8/pbmc3k_final.rds?dl=1. Then locate the working directory of R to the folder where the Seurat object was downloaded using the 'setwd' command.

Preparations

First load the essential packages of TSCAN and Seurat.

```
library(TSCAN)
library(Seurat)
```

```
## Attaching SeuratObject
```

Then read in the pbmc3k Seurat object.

```
s <- readRDS('pbmc3k_final.rds')
```

In this vignette, we are going to use the top 10 PCs and the cell clusters identified by Seurat to reconstruct the pseudotime. In real practice, TSCAN can accept any low-dimensional representations (e.g., PCA, UMAP, t-SNE, diffusion map) and cell clusters from any clustering method (e.g., Louvain clustering, k-means, hierarchical clustering), as long as the input follows the desired format.

Here we directly extract the top 10 PCs from the Seurat object. In real practice, the low-dimensional representation should be a numeric matrix where each row is a cell and each column is a reduced dimension. Row names of the matrix should be the cell names.

```
dr <- Embeddings(s, reduction = "pca")[,1:10]
str(dr)
```

```
##  num [1:2638, 1:10] -4.73 -0.517 -3.189 12.793 -3.129 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:2638] "AAACATACAACCAC" "AAACATTGAGCTAC" "AAACATTGATCAGC" "AAACCGTGCTTCCG" ...
##   ..$ : chr [1:10] "PC_1" "PC_2" "PC_3" "PC_4" ...
```

You can also use t-SNE or UMAP. For example, to extract UMAP coordinates from the Seurat object:

```
dr_umap <- Embeddings(s, reduction = "umap")
str(dr_umap)
```

```
##  num [1:2638, 1:2] 10.6 15.6 13 26.7 13.4 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:2638] "AAACATACAACCAC" "AAACATTGAGCTAC" "AAACATTGATCAGC" "AAACCGTGCTTCCG" ...
##   ..$ : chr [1:2] "UMAP_1" "UMAP_2"
```

Next, the cell clustering from Seurat can be extracted by:

```
clu <- Idents(s)
table(clu)
```

```
## clu
##  Naive CD4 T Memory CD4 T   CD14+ Mono            B        CD8 T FCGR3A+ Mono
##          697          483          480          344          271          162
##           NK           DC           Mk
##          155           32           14
```

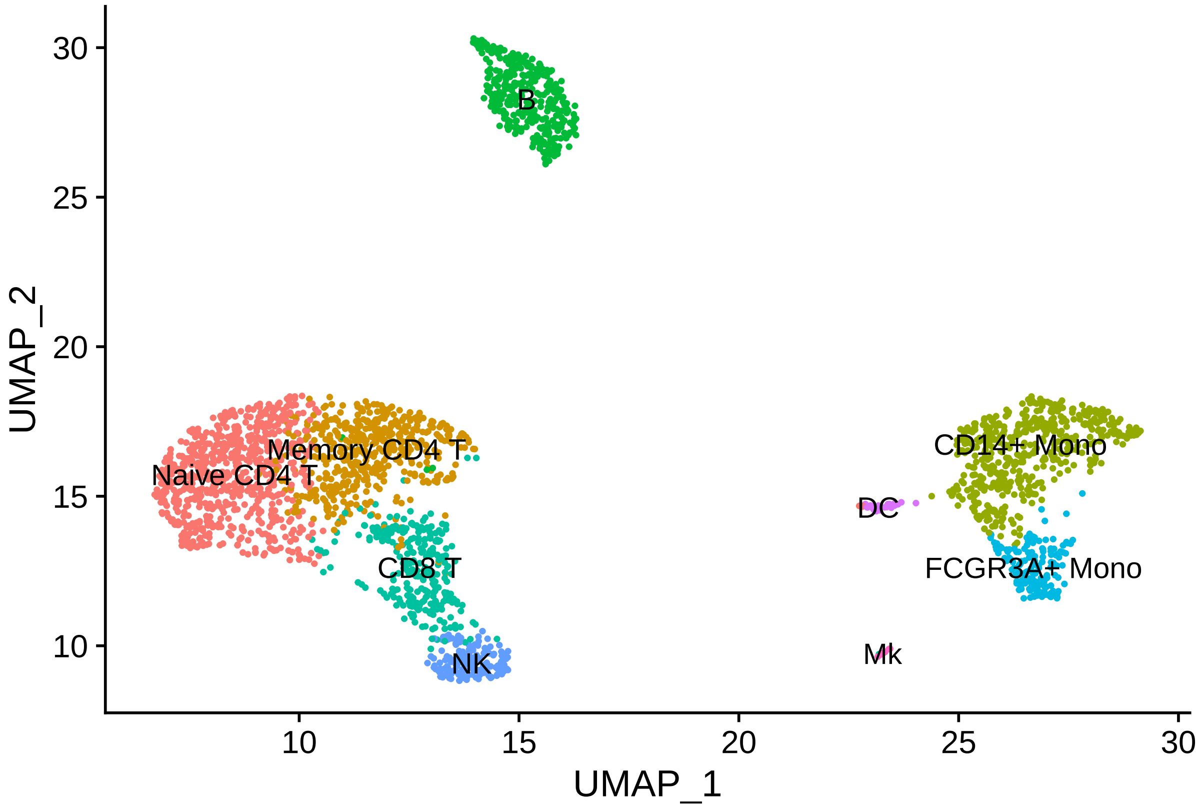
Finally, we also extract the normalized gene expression matrix for plotting and testing single gene dynamic patterns.

```
mat <- s@assays$RNA@data
```

Identify the desired trajectory

A common way to identify the trajectory is to visually inspect the PCA or UMAP plot and find neibouring clusters that are involved in a continuous biological process. In this example, we first inspect the UMAP plot:

```
DimPlot(s, reduction = "umap", label = TRUE, pt.size = 0.5) + NoLegend()
```



In the T cell compartment, we notice two clusters, naive CD4 T cells and memory CD4 T cells, that are close to each other on the UMAP. Biologically, we know naive T cells can differentiate into memory T cells in T cell activation process. So naive CD4 T cells to memory CD4 T cells is a possible cell trajectory.

Construct pseudotime with TSCAN

We use guided_tscan function to obtain the pseudotime trajectory from naive CD4 T cells to memory CD4 T cells. It requires three inputs: the low-dimensional representation (dr), the cell clustering (clu), and the desired order of cell clusters to form pseudotime (cluorder). For example, the following code contrusts a pseudotime ordering from naive CD4 T cells to memory CD4 T cells:

```
ord <- guided_tscan(dr=dr,clu=clu,cluorder=c('Naive CD4 T','Memory CD4 T'))
```

The output of the function is a character vector of cell ordering:

```
head(ord)
```

```
## [1] "CTAATAGAGCTATG" "TAGTCTTGGCTGTA" "ACTTGTACCTGTCC" "CACCCATGTTCTGT"
## [5] "CAATTCACTTGTGG" "CCGATAGACCTAAG"
```

```
str(ord)
```

```
##  chr [1:1180] "CTAATAGAGCTATG" "TAGTCTTGGCTGTA" "ACTTGTACCTGTCC" ...
```

Study gene expression dynamics along pseudotime

Once we obtained the pseudotime cell ordering, the next step is to study how gene expression changes dynamically along pseudotime. The key genes driving the pseudotime trajectory will give us deeper insights of the underlying biological process.

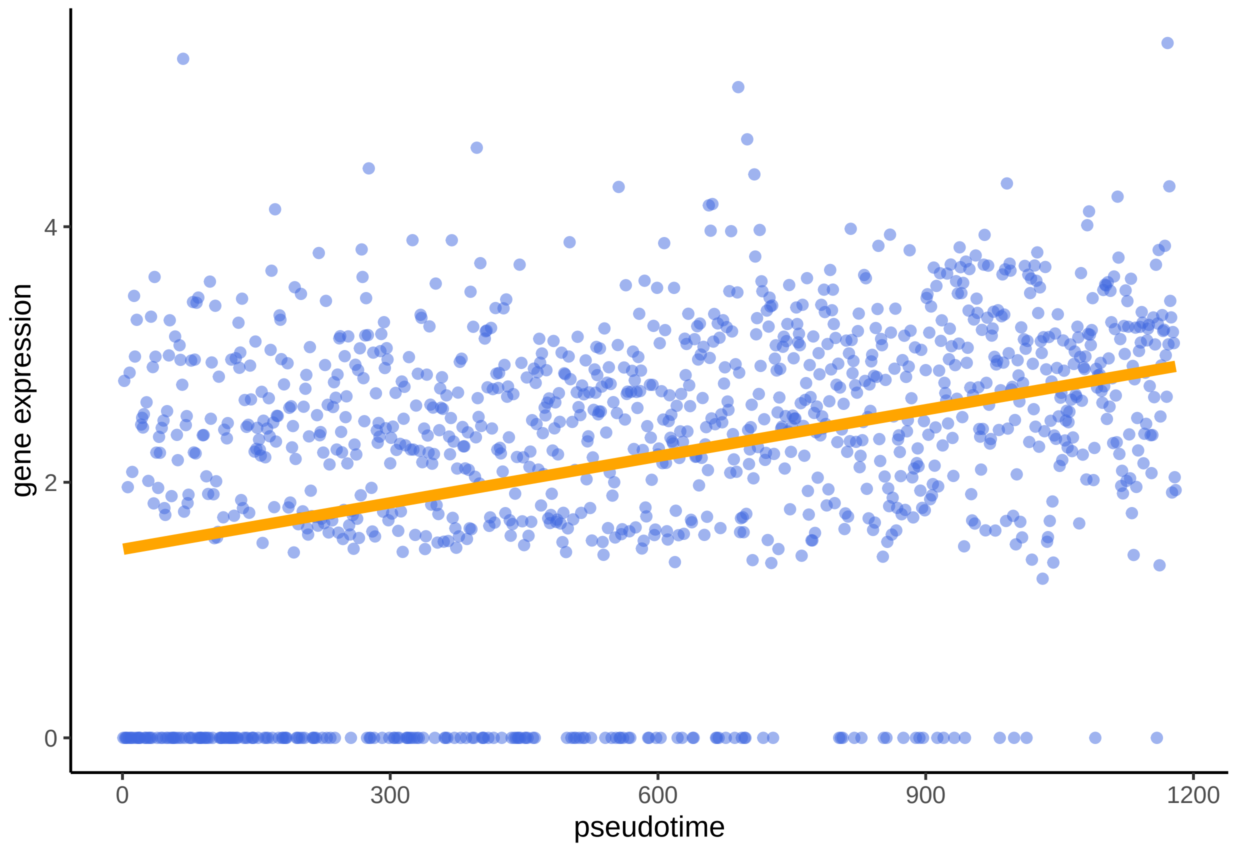
First, we need to identify the genes that have significant dynamic changing pattern along pseudotime using the difftest function based on generalized additive models (GAM). This function takes some time to run on all genes. For simplicity here we only run the tests on all genes whose names start with 'IL'.

```r
res <- difftest(mat[grep('^IL',rownames(s)),],ord)
head(res)
```

```
##                   pval          FDR
## IL32     9.220087e-46 4.610044e-44
## IL7R     5.846648e-19 1.461662e-17
## IL2RG    3.317638e-10 5.529396e-09
## IL10RA   1.401804e-03 1.752255e-02
## IL16     5.465837e-03 5.465837e-02
## ILF3-AS1 1.485497e-02 1.237914e-01
```

We noticed that IL32 is the top differential gene. Let's further explore its gene expression pattern along pseudotime.

```r
genedynamics(mat['IL32',],ord)
```



We can see that the expression of IL32 does have a significant increasing pattern, which agrees with previous literature that its expression increases after T cells are activated (See: https://www.ncbi.nlm.nih.gov/gene/9235).

Pseudotime analysis for multiple samples

This vignette only focuses on pseudotime analysis for one sample. If there are multiple samples in your dataset (e.g., multiple human patients or mouse subjects), the pseudotime needs to be constructed on the integrated space. See: https://satijalab.org/seurat/articles/integration_introduction.html

The procedure of pseudotime construction is the same as described in this vignette. However, the differential test and the visualization of single gene expression needs to be done by Lamian, a statistical framework for differential pseudotime analysis with multiple single-cell RNA-seq samples. Please refer to the preprint https://www.biorxiv.org/content/10.1101/2021.07.10.451910v1 and the software tool https://github.com/Winnie09/Lamian.

Session info

```
sessionInfo()
```

```
## R version 4.1.0 (2021-05-18)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] SeuratObject_4.0.2 Seurat_4.0.4       TSCAN_2.0.0
##
## loaded via a namespace (and not attached):
##   [1] Rtsne_0.15            colorspace_2.0-2      deldir_0.2-10
##   [4] ellipsis_0.3.2        ggridges_0.5.3        mclust_5.4.7
##   [7] spatstat.data_2.1-0   leiden_0.3.9          listenv_0.8.0
##  [10] farver_2.1.0          ggrepel_0.9.1         fansi_0.5.0
##  [13] codetools_0.2-18      splines_4.1.0         knitr_1.36
##  [16] polyclip_1.10-0       jsonlite_1.7.2        ica_1.0-2
##  [19] cluster_2.1.2         png_0.1-7             uwot_0.1.10
##  [22] shiny_1.7.0           sctransform_0.3.2     spatstat.sparse_2.0-0
##  [25] compiler_4.1.0        httr_1.4.2            Matrix_1.3-3
##  [28] fastmap_1.1.0         lazyeval_0.2.2        later_1.3.0
##  [31] htmltools_0.5.2       tools_4.1.0           igraph_1.2.6
##  [34] gtable_0.3.0          glue_1.4.2            RANN_2.6.1
##  [37] reshape2_1.4.4        dplyr_1.0.7           Rcpp_1.0.7
##  [40] scattermore_0.7       vctrs_0.3.8           nlme_3.1-152
##  [43] lmtest_0.9-38         xfun_0.26             stringr_1.4.0
##  [46] globals_0.14.0        mime_0.12             miniUI_0.1.1.1
##  [49] lifecycle_1.0.1       irlba_2.3.3           gtools_3.9.2
##  [52] goftest_1.2-2         future_1.22.1         MASS_7.3-54
##  [55] zoo_1.8-9             scales_1.1.1          spatstat.core_2.3-0
##  [58] promises_1.2.0.1      spatstat.utils_2.2-0  parallel_4.1.0
##  [61] RColorBrewer_1.1-2    yaml_2.2.1            reticulate_1.22
##  [64] pbapply_1.5-0         gridExtra_2.3         ggplot2_3.3.5
##  [67] rpart_4.1-15          fastICA_1.2-3         stringi_1.7.4
##  [70] highr_0.9             caTools_1.18.2        rlang_0.4.11
##  [73] pkgconfig_2.0.3       matrixStats_0.61.0    bitops_1.0-7
##  [76] evaluate_0.14         lattice_0.20-44       ROCR_1.0-11
##  [79] purrr_0.3.4           tensor_1.5            patchwork_1.1.1
##  [82] htmlwidgets_1.5.4    labeling_0.4.2        cowplot_1.1.1
##  [85] tidyselect_1.1.1     parallelly_1.28.1     RcppAnnoy_0.0.19
##  [88] plyr_1.8.6           magrittr_2.0.1        R6_2.5.1
##  [91] gplots_3.1.1         generics_0.1.0        combinat_0.0-8
##  [94] DBI_1.1.1            pillar_1.6.3          mgcv_1.8-35
##  [97] fitdistrplus_1.1-6   survival_3.2-11       abind_1.4-5
## [100] tibble_3.1.5         future.apply_1.8.1    crayon_1.4.1
## [103] KernSmooth_2.23-20   utf8_1.2.2            spatstat.geom_2.2-2
## [106] plotly_4.9.4.1       rmarkdown_2.11        grid_4.1.0
## [109] data.table_1.14.2    digest_0.6.28         xtable_1.8-4
## [112] tidyr_1.1.4          httpuv_1.6.3          munsell_0.5.0
## [115] viridisLite_0.4.0
```