

Kameneva Lab Data analysis task

Zafer Kosar

April 25, 2025

Abstract

Kameneva Lab single-cell data analysis task includes a series of single-cell RNAseq datasets that span from week 6 to week 12 of the development of human adrenal glands. These datasets were shown to provide sufficient information on developmental transitions in addition to the gene regulatory programs in adrenal gland development. Here, utilizing multiple single-cell analysis tools and approaches, I demonstrate that Schwann cell precursors in the adrenal gland can potentially give rise to Sympathoblasts. Moreover, Sympathoblasts can transition into chromaffin cells. I also highlight the gene expression dynamics — the possible drivers — of these transitions. Overall, the findings of the task align well with the original study.

Index Terms: Neuroblastoma Origin, Single-cell Transcriptomics, Trajectory Analysis

1 Methods

1.1 Quality-Control, Filtering, Integration

As I am much more proficient in the Python single-cell ecosystem than R's, I started the task with tools like Scanpy and scVI. However, even after a substantial amount of time of trial-and-error, it became clear that I needed to switch ecosystems.

After switching to the R ecosystem, except for memory issues that were resolved by downsampling each dataset to 25% of their original sample size, the integration worked seamlessly after quality-control and filtering steps in Seurat. Integration was validated via visual inspection of the UMAP (Figure 1).

For initial filtering, genes that appear in fewer than 3 cells and cells that have less than 200 genes were filtered out for each dataset. Additionally, cells with fewer than 500 and more than 6000 gene counts were filtered out to remove potentially empty or doublet cells, respectively. Finally, cells with 15% or more mitochondrial genes are filtered out.

Cell cycle scores, using cell cycle markers built into Seurat, were added to the data. Variations in cell cycles were then regressed out during scaling.

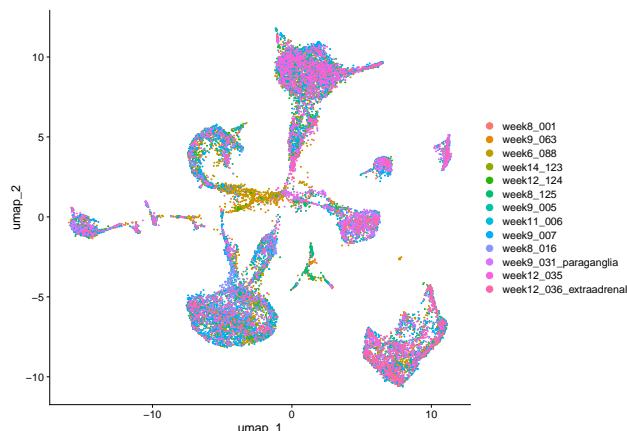


Figure 1. UMAP after integration of the datasets.

1.2 UMAP and Cluster Annotation

UMAP calculation had already been performed after integration using the standard Seurat workflow. Then, clustering has been performed. The cluster annotations have been conducted using markers from Kameneva et. al. 2021, Figure 1b, and visualized on a UMAP plot (Figure 2). Nevertheless, some markers, PAX2 (kidney), HNF4A and AHSG (Liver), AZU1 (HSCs) were absent in the data. Therefore, automatic annotation could not be performed. Instead, manual annotation of clusters through inspecting the marker gene expression of clusters (dot plot, feature plots) was performed. Upon annotation, 5 genes that are differentially expressed for each cluster were visualized using a dot plot (Figure 3).

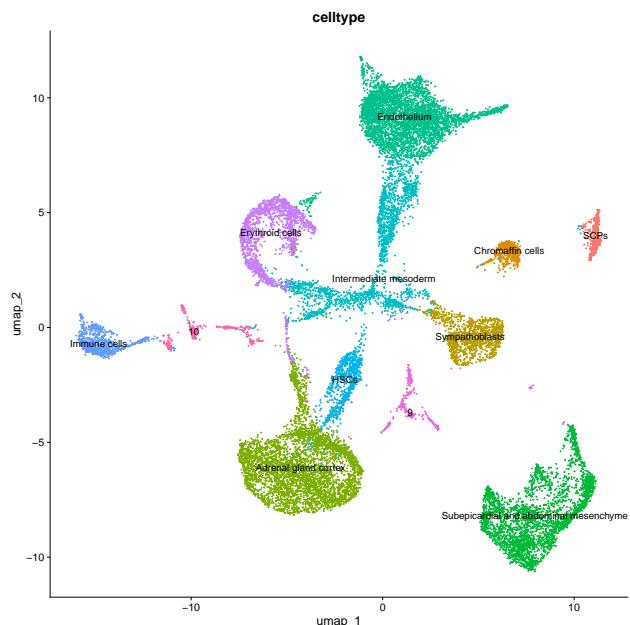


Figure 2. Annotated Clusters.

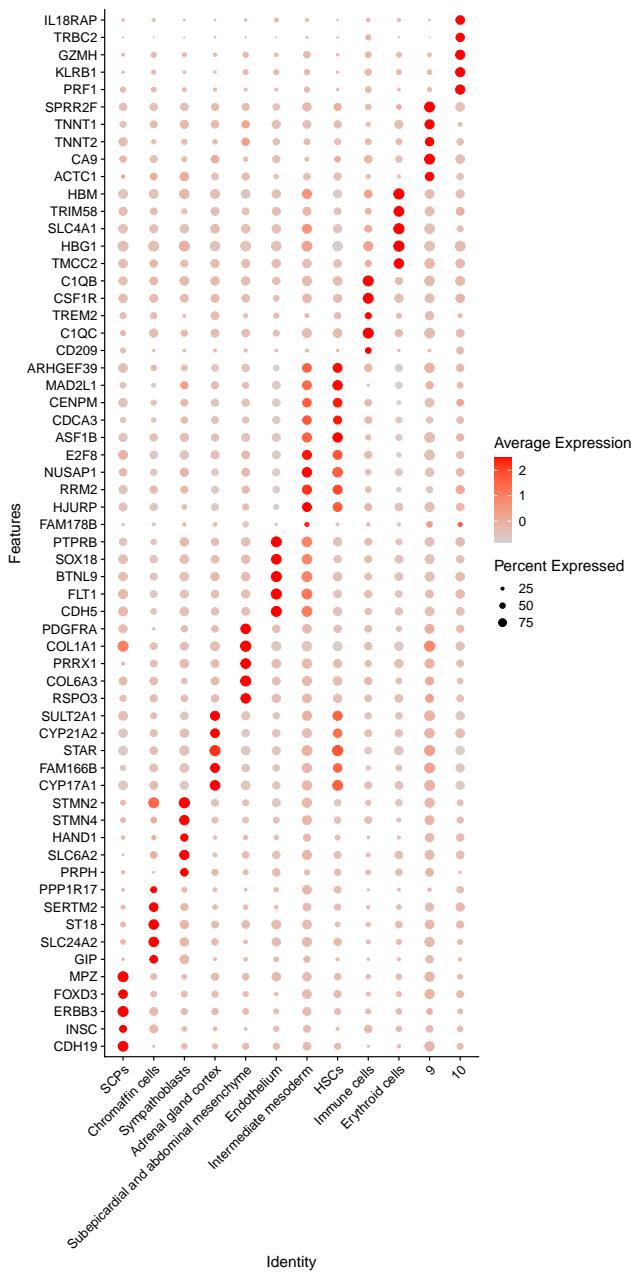


Figure 3. Differentially expressed genes of the clusters.

1.3 Subsetting Adrenal Medulla Clusters

The original data was subsetted to include only SCPs, Chromaffin cells, and Sympathoblasts. The subset was reclustered to improve the resolution of transitions for the upcoming analysis. Clusters were again annotated according to the markers provided in Kameneva et. al. 2021, Figure 2a.

1.4 Trajectory Analysis

For trajectory analysis, I first attempted to utilize monocle3 of the R single-cell data analysis. Monocle3 is an excellent tool for trajectory analysis due to its graph-based approach to trajectory learning and the capacity to handle complex branching. Moreover, monocle3 is well documented, and it has many example use cases due to its popularity. However, after assigning pseudotimes to cells, I was not able to extract the significant gene expression changes along the trajectory.

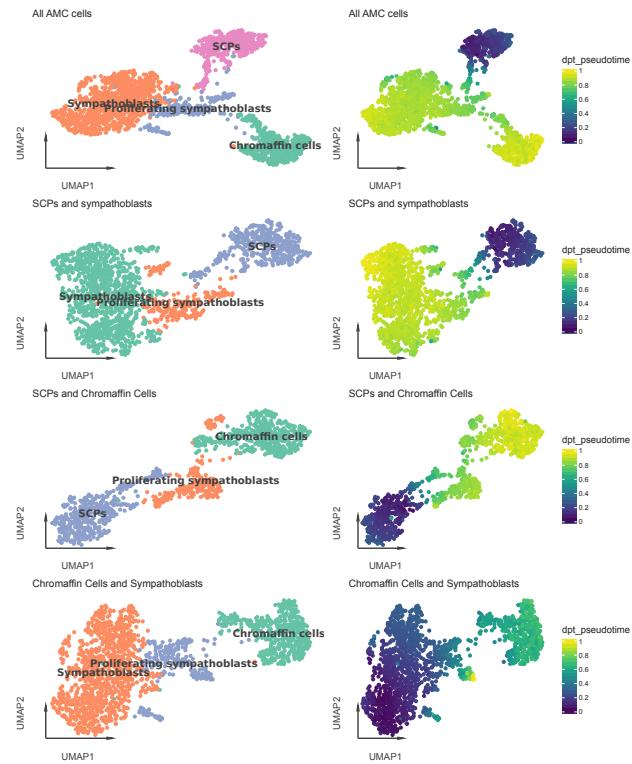
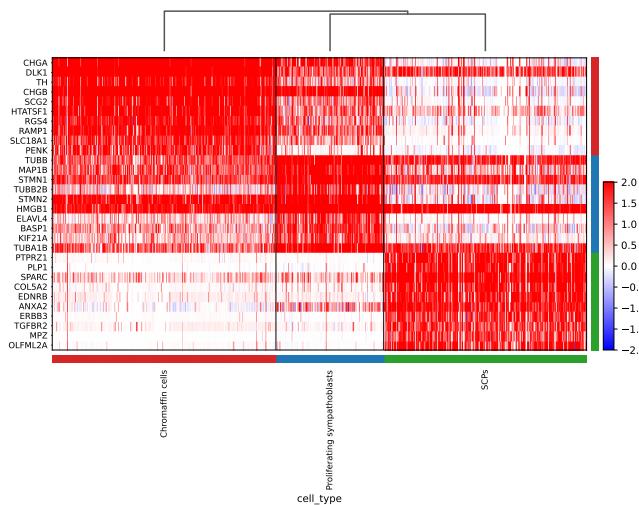


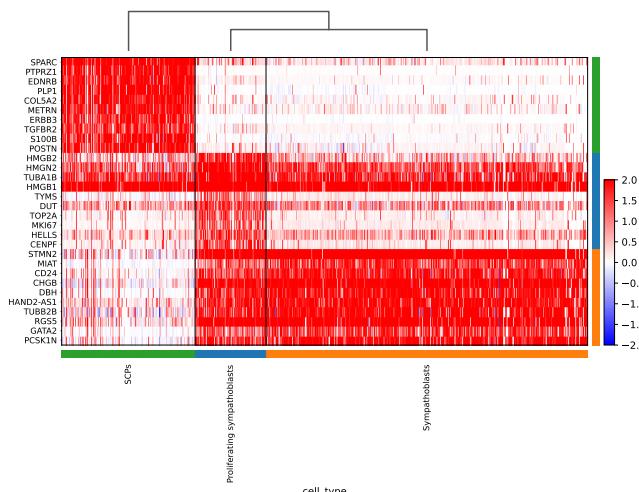
Figure 4. Annotated Clusters.

At that point, I decided to switch to the Python ecosystem, as there are great tools for trajectory analysis within that ecosystem, such as scVelo and paga. Indeed, scVelo requires spliced and unspliced RNA information for such analysis, which this data does not contain. PAGA (Partition-based Graph Abstraction), included in Scanpy, is a scalable tool that allows cluster-level trajectory inference, which is indeed needed here.

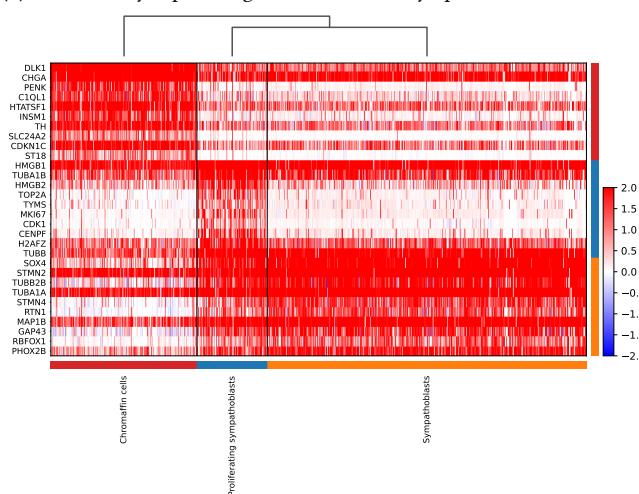
Utilizing PAGA, I was able to get pseudotime trajectories between SCPs and chromaffin cells, SCPs and sympathoblasts, and chromaffin cells and sympathoblasts. Here, I had the opportunity to utilize my own single-cell data visualization tool **Celestial** (Figure 4). Additionally, PAGA allowed me to obtain differentially expressed genes along the trajectories, which were then visualized



(a) Differentially expressed genes in SCPs and chromaffin cells.



(b) Differentially expressed genes in SCPs and Sympathoblasts.



(c) Differentially expressed genes in Chromaffin cells and Sympathoblasts.

on heatmaps (Figure 5a,5b,5c).

2 Discussion

To address task 7c, there are conclusions to draw both from the pseudotime plots of the cells and heatmaps of the differentially expressed genes along the transitions. Firstly, SCPs appear to be the progenitor of both Sympathoblasts and Chromaffin cells (Figure 4). Furthermore, Sympathoblasts might be the progenitors of the Chromaffin cells, establishing a cell lineage from SCPs to Chromaffin cells.

The significant changes in gene expression profiles of the cells along the transition trajectory from SCPs to other cell types, and the transition from Sympathoblasts to Chromaffin cells, reveal important cellular program shifts along such trajectories (Figure 5a,5b,5c). Remarkably, those genes are not only expressed differentially as a result of such transitions, but they are possibly the drivers of cell type transitions in Adrenal Medulla.

In all trajectory analyses, proliferating Sympathoblasts were also employed as they seem to have a role in transitions. This transitional role becomes apparent through the dendograms provided via the heatmaps of the gene expressions (Figure 5a,5b,5c).

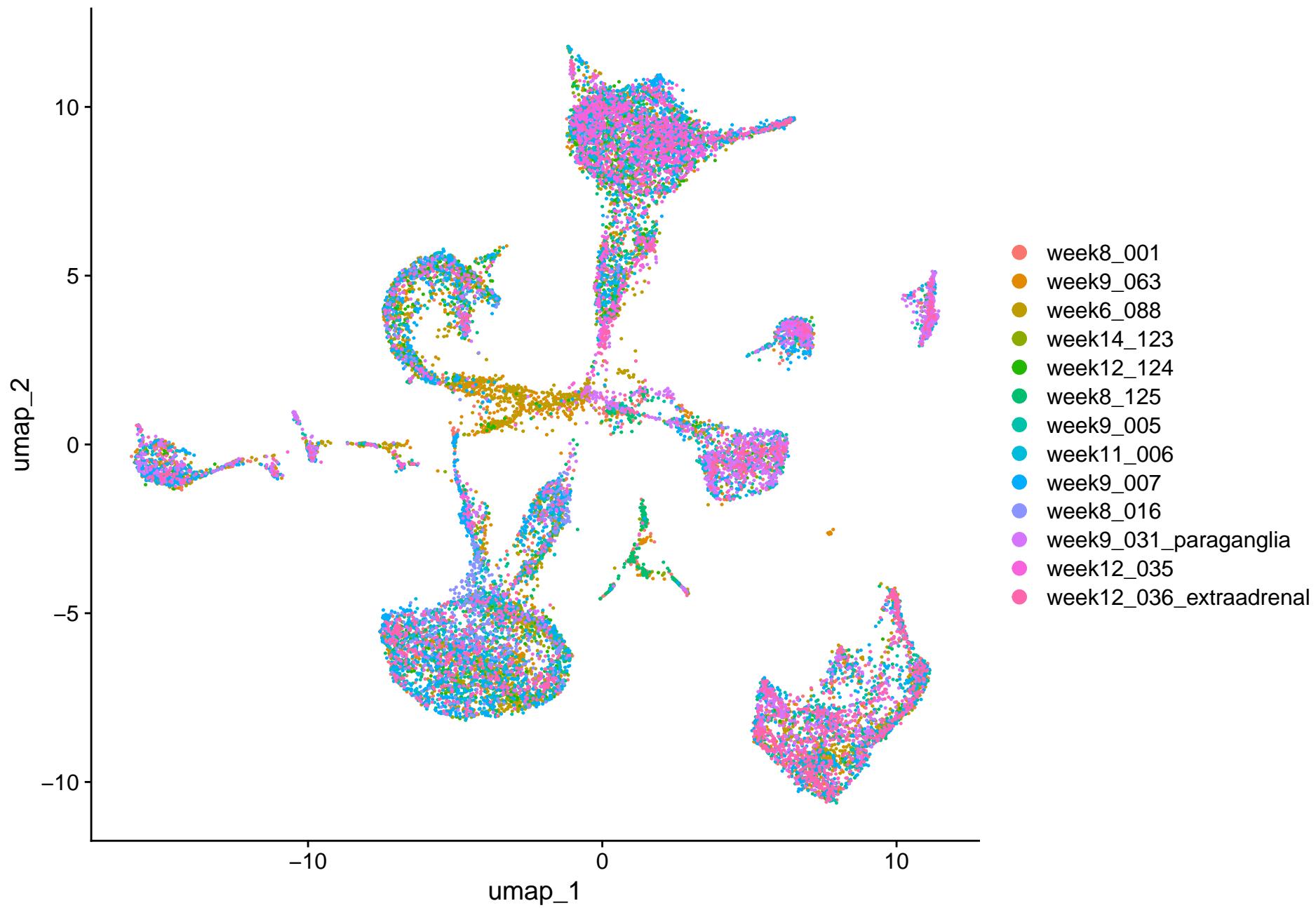
It should be noted that there are some minor and some notable differences from the original work (Kameneva et al., 2021). One significant difference is that, due to computational resource limitations, downscaling of the datasets has been performed. This downscaling is expected to influence all the downstream workflow. Nevertheless, there were sufficient numbers of cells to conduct the downstream analysis.

Surprisingly, the datasets did not include some markers that are present in the original work. Markers such as PAX2 (kidney), HNF4A and AHSG (Liver), AZU1 (HSCs) were not present in the data despite a very loose filtering of genes. This difference made manual annotation necessary for cluster annotations.

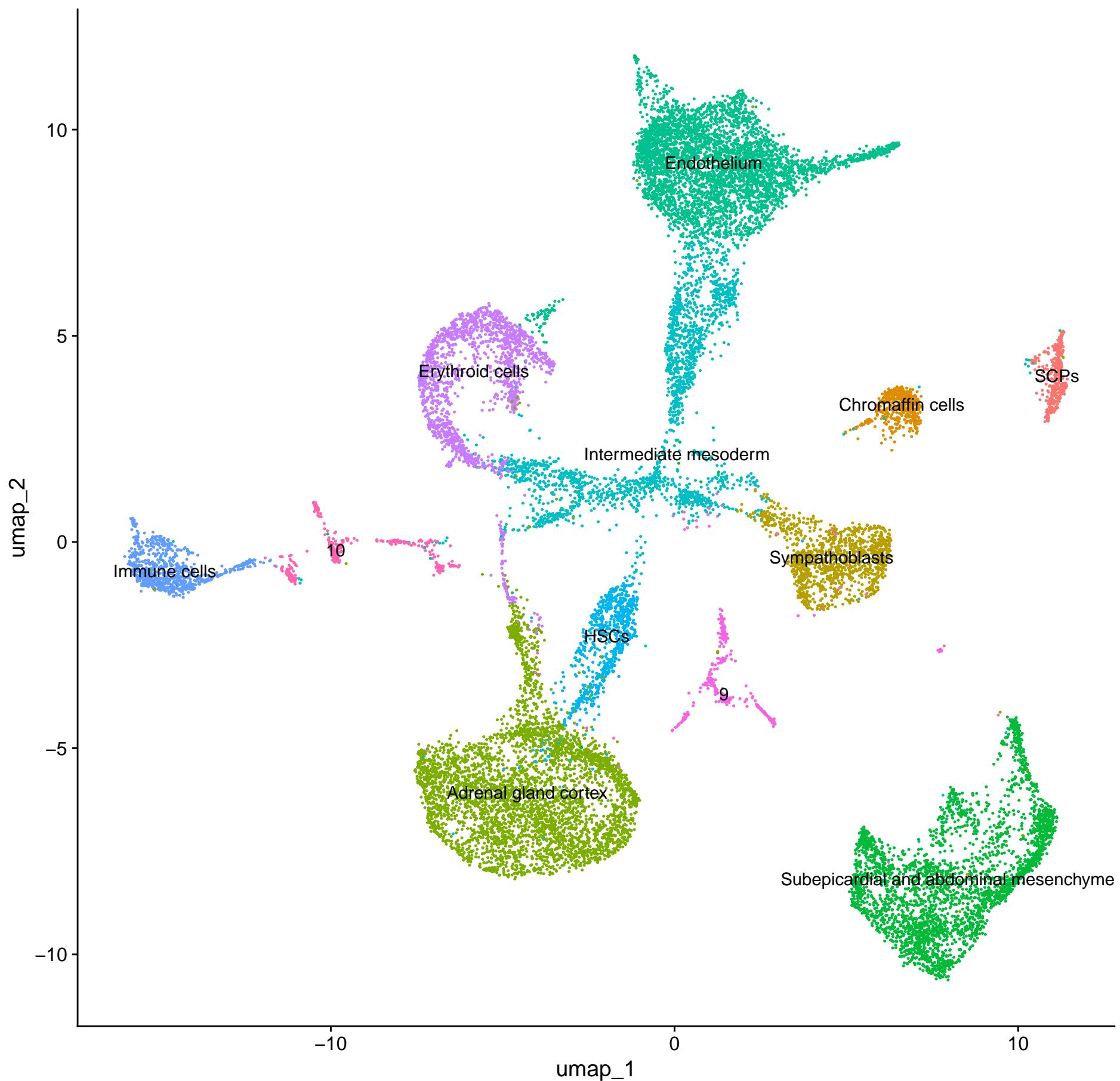
Overall, the workflow for the task could be completed, and the possible transitions from SCPs to Sympathoblasts and Chromaffin cells were shown, in addition to revealing important cellular programs that drive those transitions. Therefore, the results of this task are in agreement with the original work.

References

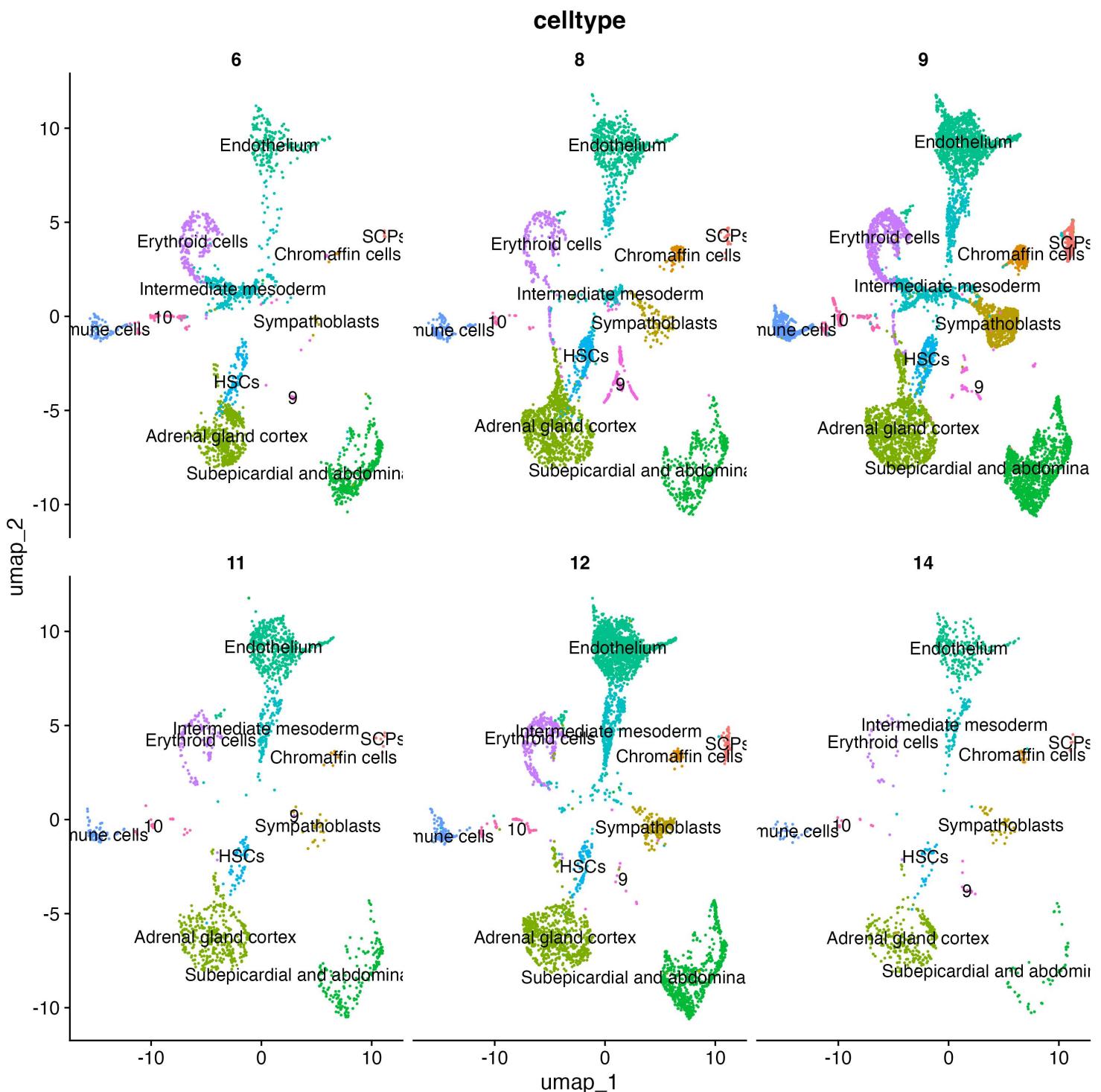
- Kameneva, Polina et al. (Apr. 2021). "Single-cell transcriptomics of human embryos identifies multiple sympathoblast lineages with potential implications for neuroblastoma origin". In: *Nature Genetics* 53.5, pp. 694–706. ISSN: 1546-1718. doi: 10.1038/s41588-021-00818-x. URL: <http://dx.doi.org/10.1038/s41588-021-00818-x>.

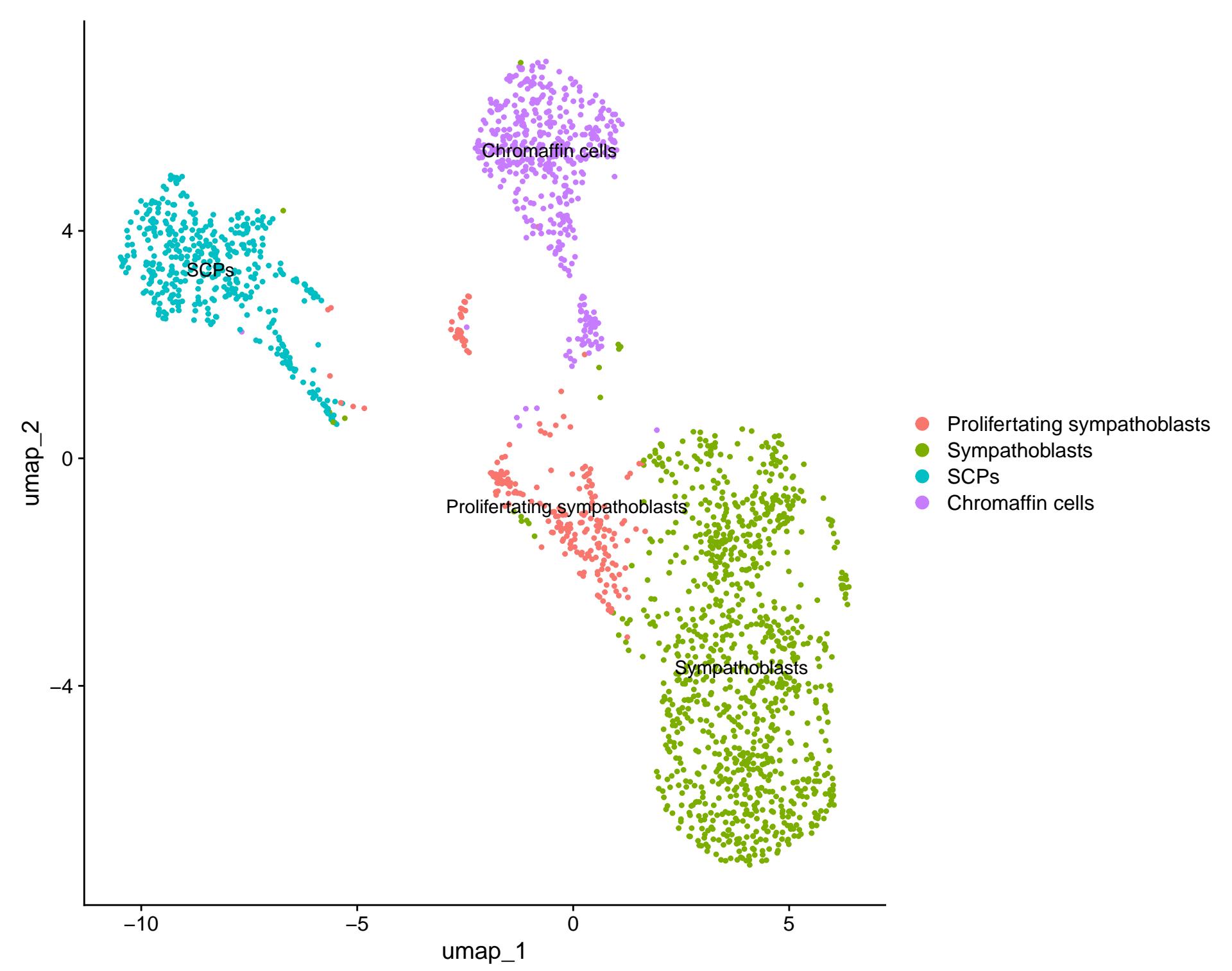


celltype

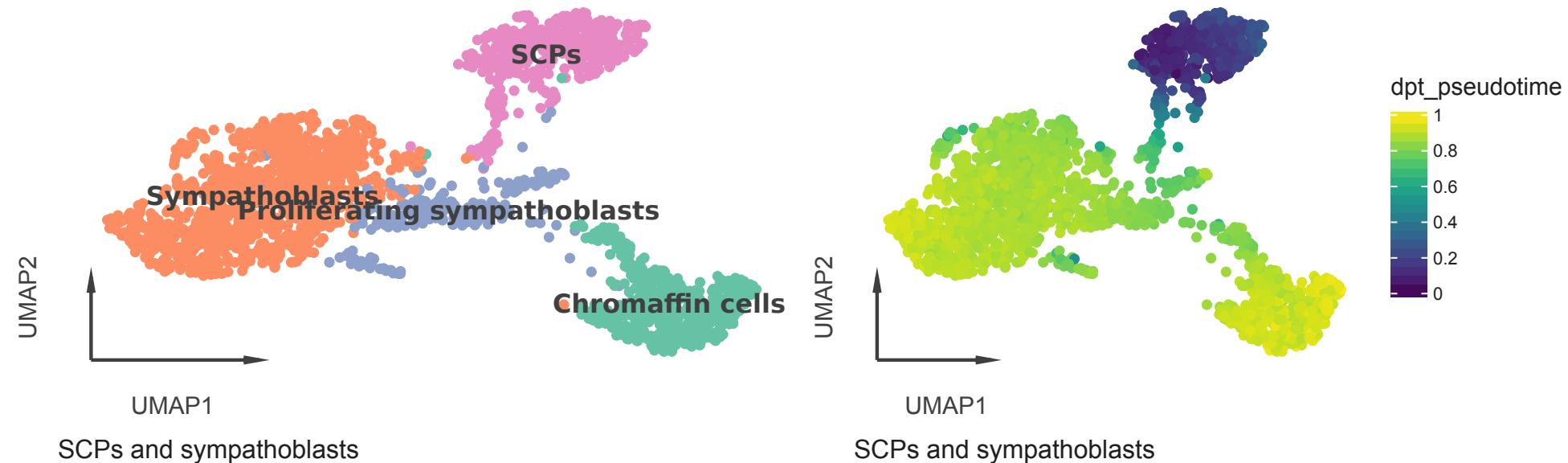




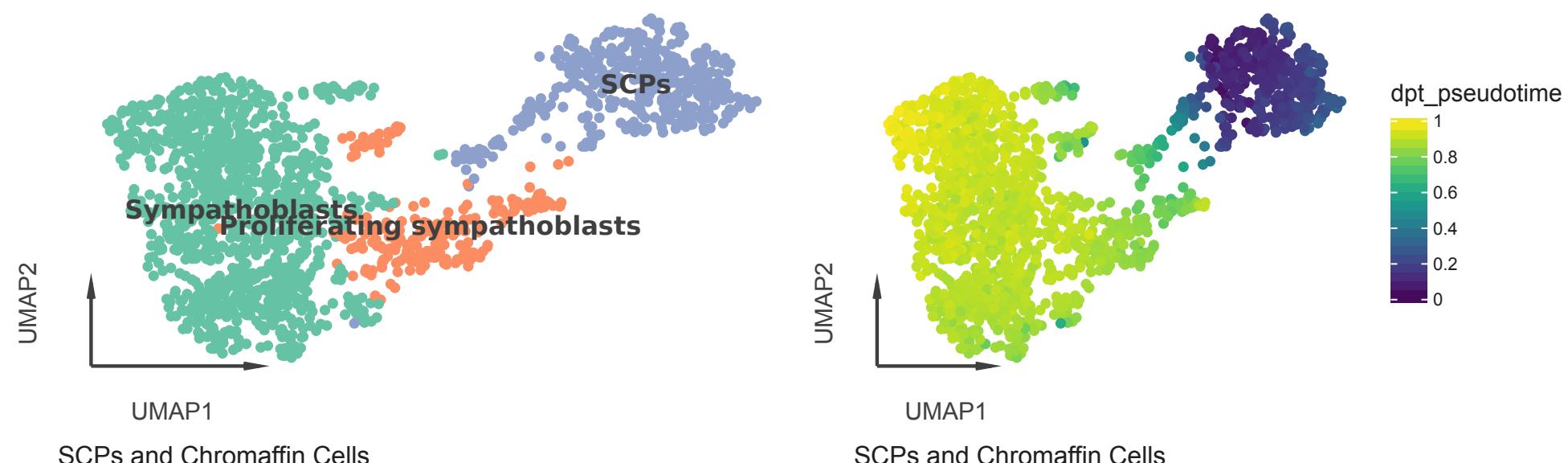




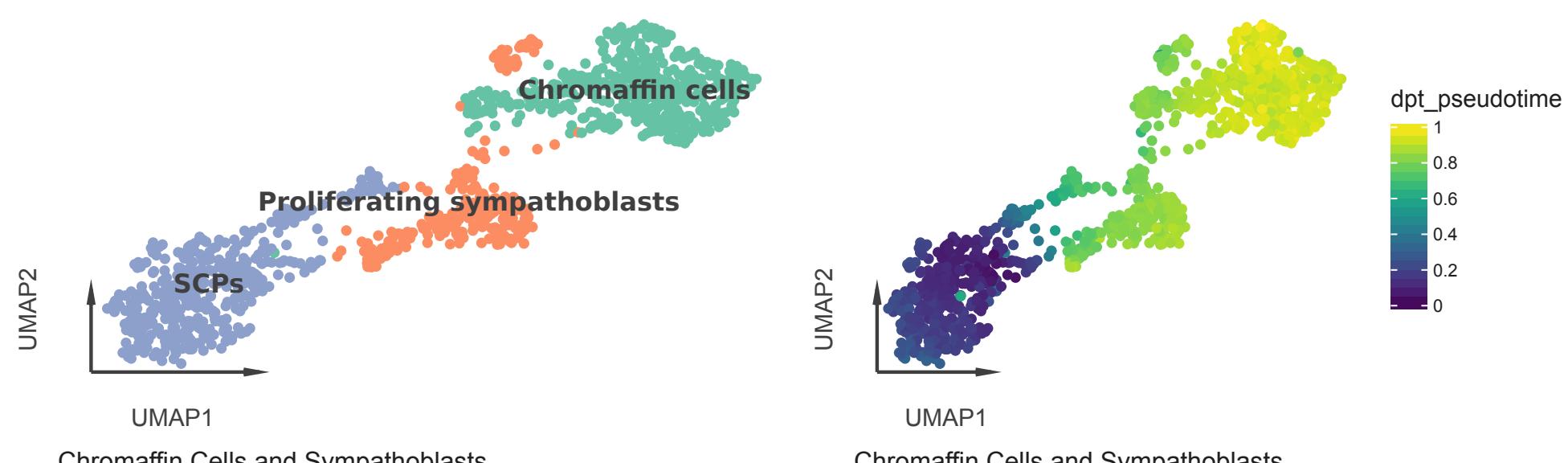
All AMC cells



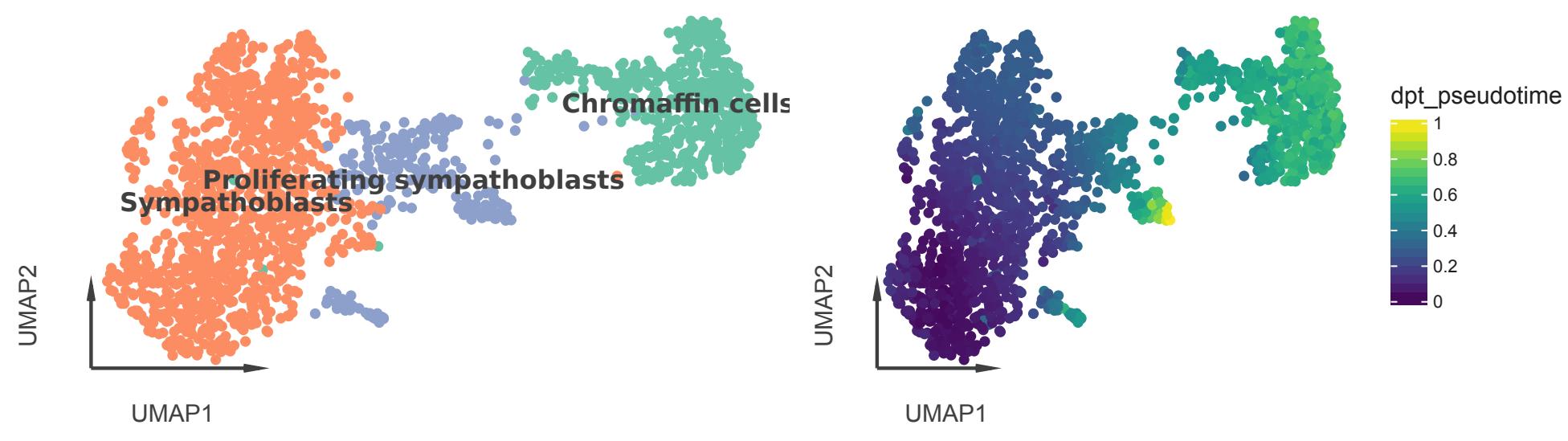
SCPs and sympathoblasts



SCPs and Chromaffin Cells

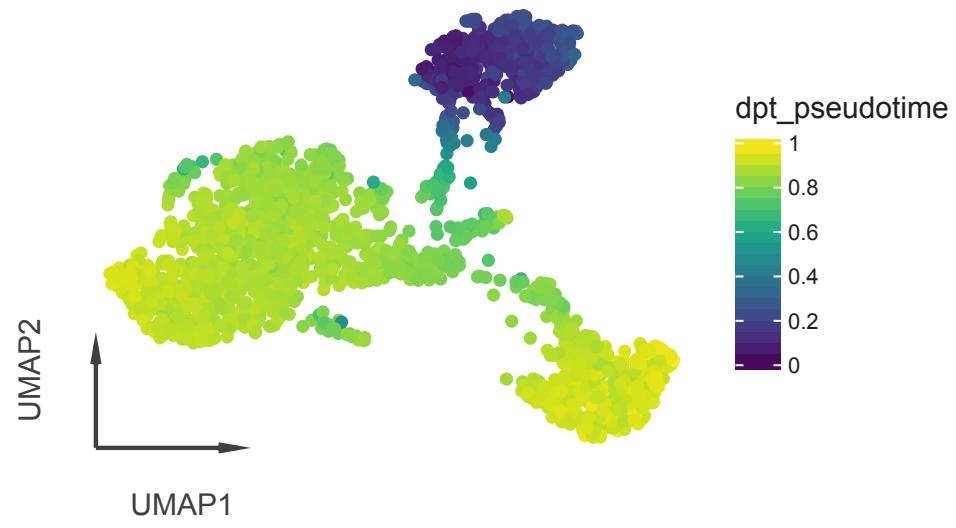


Chromaffin Cells and Sympathoblasts

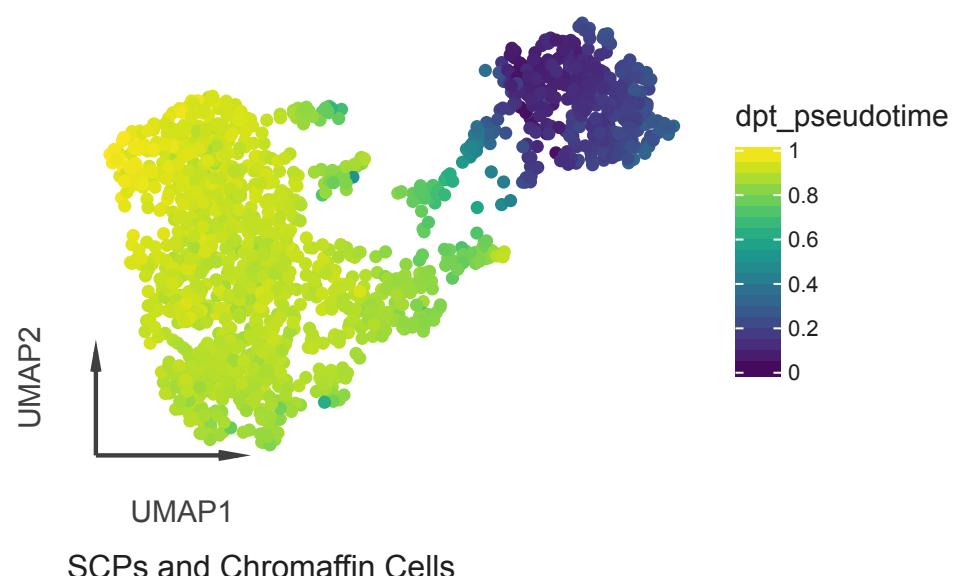


UMAP1

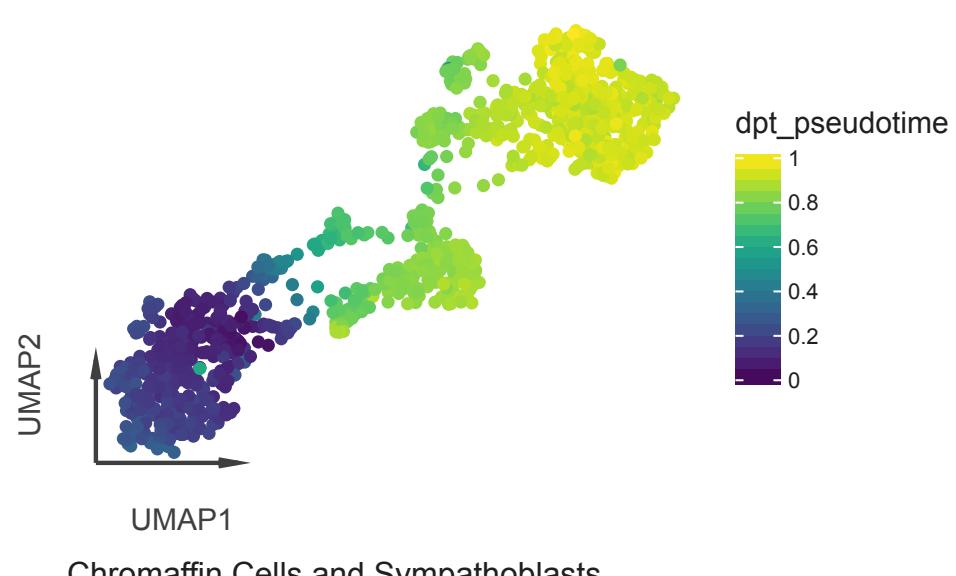
All AMC cells



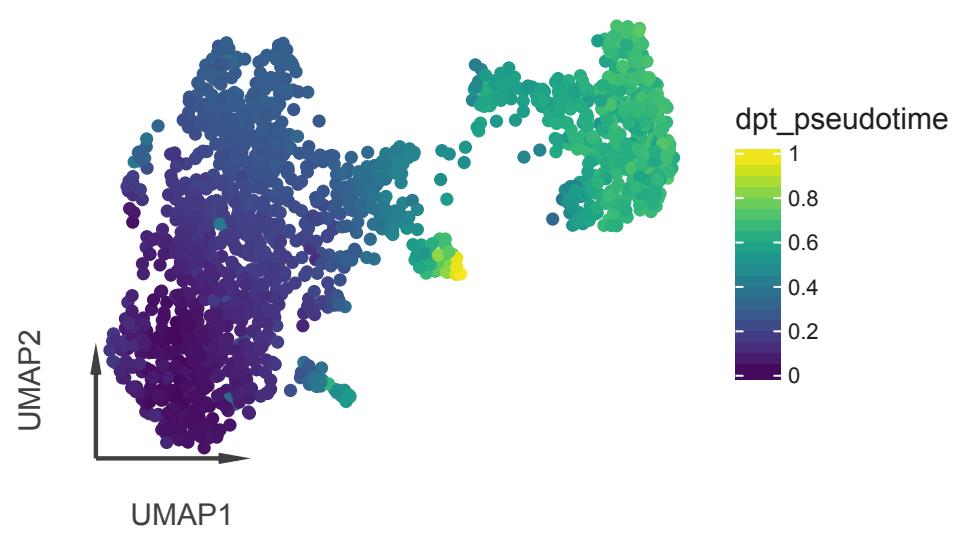
SCPs and sympathoblasts



SCPs and Chromaffin Cells

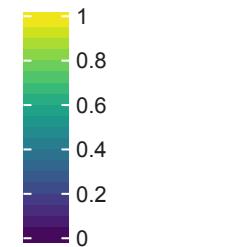


Chromaffin Cells and Sympathoblasts

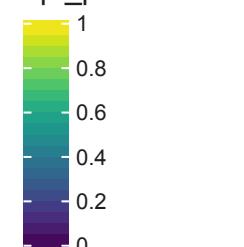


UMAP1

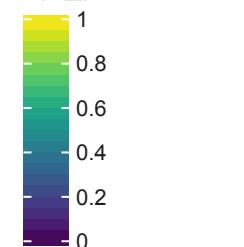
dpt_pseudotime



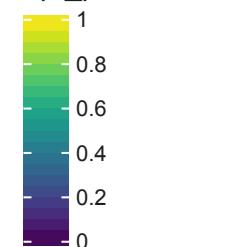
dpt_pseudotime

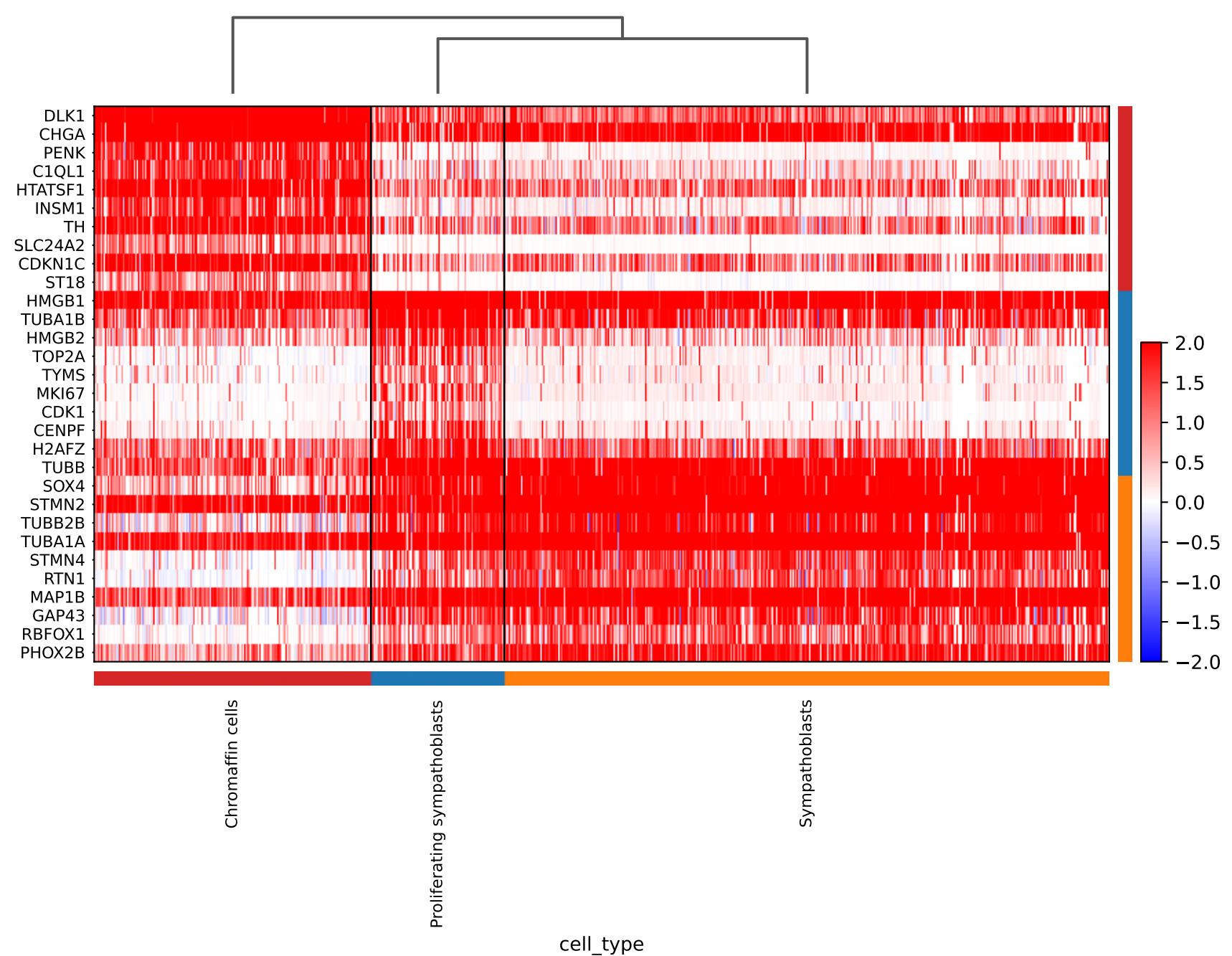


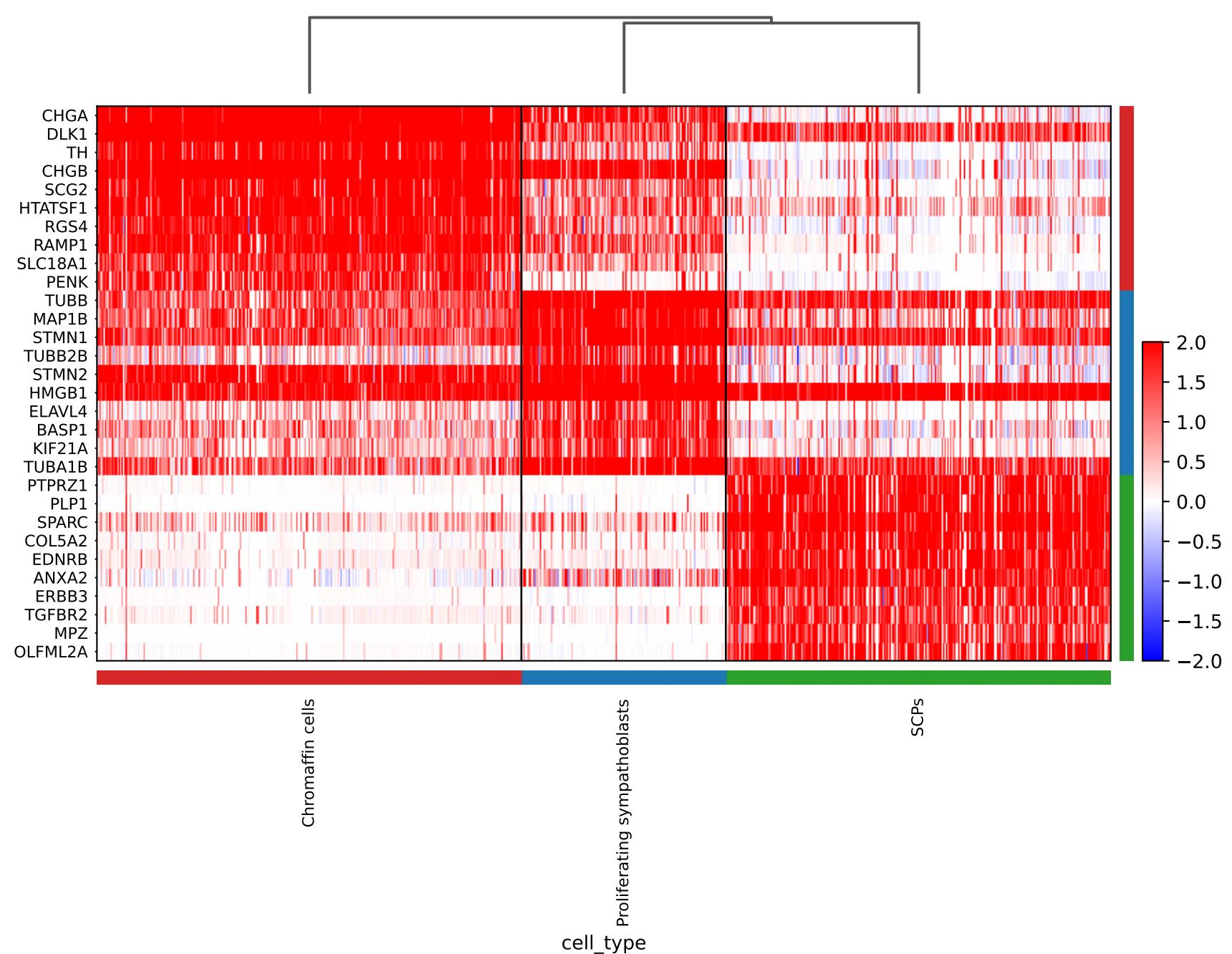
dpt_pseudotime

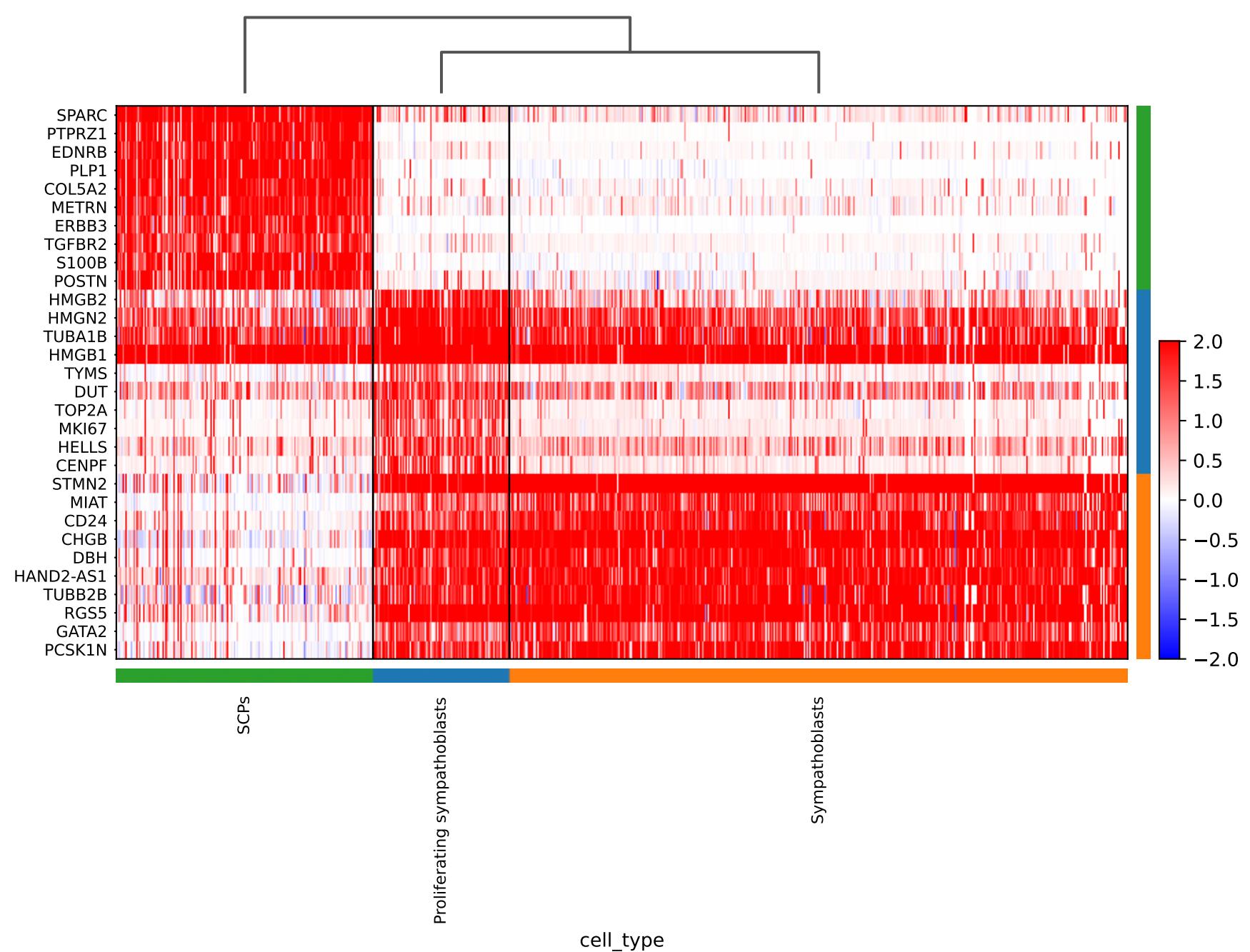


dpt_pseudotime









Integration, QC, Filtering, and Normalization

```
library(Seurat)
library(ggplot2)
library(dplyr)
library(patchwork)
setwd("~/dev/CCRItask")

# map GSM ids to info
files <- list(
  "GSM4446535" = "week8_001",
  "GSM4446536" = "week9_063",
  "GSM4446537" = "week6_088",
  "GSM4446538" = "week14_123",
  "GSM4446539" = "week12_124",
  "GSM4446540" = "week8_125",
  "GSM4446541" = "week9_005",
  "GSM4446542" = "week11_006",
  "GSM4446543" = "week9_007",
  "GSM4734601" = "week8_016",
  "GSM4734602" = "week9_031_paraganglia",
  "GSM4734603" = "week12_035",
  "GSM4734604" = "week12_036_extraadrenal"
)

# List all H5 files in the raw data directory
h5_files <- list.files("data/raw/GSE147821_RAW", pattern = ".h5$", full.names = TRUE)
h5_files

1. 'data/raw/GSE147821_RAW/GSM4446535_10X_19_001.raw_feature_bc_matrix.h5'
2. 'data/raw/GSE147821_RAW/GSM4446536_10X_19_063.raw_feature_bc_matrix.h5'
3. 'data/raw/GSE147821_RAW/GSM4446537_10X_19_088.raw_feature_bc_matrix.h5'
4. 'data/raw/GSE147821_RAW/GSM4446538_10X_19_123.raw_feature_bc_matrix.h5'
5. 'data/raw/GSE147821_RAW/GSM4446539_10X_19_124.raw_feature_bc_matrix.h5'
6. 'data/raw/GSE147821_RAW/GSM4446540_10X_19_125.raw_feature_bc_matrix.h5'
7. 'data/raw/GSE147821_RAW/GSM4446541_10X_20_005.raw_feature_bc_matrix.h5'
8. 'data/raw/GSE147821_RAW/GSM4446542_10X_20_006.raw_feature_bc_matrix.h5'
9. 'data/raw/GSE147821_RAW/GSM4446543_10X_20_007.raw_feature_bc_matrix.h5'
10. 'data/raw/GSE147821_RAW/GSM4734601_10X_20_016.raw_feature_bc_matrix.h5'
11. 'data/raw/GSE147821_RAW/GSM4734602_10X_20_031.raw_feature_bc_matrix.h5'
12. 'data/raw/GSE147821_RAW/GSM4734603_10X_20_035.raw_feature_bc_matrix.h5'
13. 'data/raw/GSE147821_RAW/GSM4734604_10X_20_036.raw_feature_bc_matrix.h5'
```

```

names(h5_files) <- sub("_10X_.*", "", basename(h5_files))
h5_files

```

GSM4446535 'data/raw/GSE147821_RAW/GSM4446535_10X_19_001.raw_feature_bc_matrix.h5'GSM4446536
 'data/raw/GSE147821_RAW/GSM4446536_10X_19_063.raw_feature_bc_matrix.h5'GSM4446537
 'data/raw/GSE147821_RAW/GSM4446537_10X_19_088.raw_feature_bc_matrix.h5'GSM4446538
 'data/raw/GSE147821_RAW/GSM4446538_10X_19_123.raw_feature_bc_matrix.h5'GSM4446539
 'data/raw/GSE147821_RAW/GSM4446539_10X_19_124.raw_feature_bc_matrix.h5'GSM4446540
 'data/raw/GSE147821_RAW/GSM4446540_10X_19_125.raw_feature_bc_matrix.h5'GSM4446541
 'data/raw/GSE147821_RAW/GSM4446541_10X_20_005.raw_feature_bc_matrix.h5'GSM4446542
 'data/raw/GSE147821_RAW/GSM4446542_10X_20_006.raw_feature_bc_matrix.h5'GSM4446543
 'data/raw/GSE147821_RAW/GSM4446543_10X_20_007.raw_feature_bc_matrix.h5'GSM4734601
 'data/raw/GSE147821_RAW/GSM4734601_10X_20_016.raw_feature_bc_matrix.h5'GSM4734602
 'data/raw/GSE147821_RAW/GSM4734602_10X_20_031.raw_feature_bc_matrix.h5'GSM4734603
 'data/raw/GSE147821_RAW/GSM4734603_10X_20_035.raw_feature_bc_matrix.h5'GSM4734604
 'data/raw/GSE147821_RAW/GSM4734604_10X_20_036.raw_feature_bc_matrix.h5'

Load each sample into a list of Seurat objects and apply filtering

```

# Load each sample into a list of Seurat objects
# 3. Process each sample
seurat_list <- lapply(names(files), function(gsm_id) {

  # Get sample info
  sample_name <- files[[gsm_id]]
  week <- as.numeric(sub("week(\\d+).*", "\\\\1", sample_name))
  sample_id <- sub(".*_(_\\d+).*", "\\\\1", sample_name)
  # Read data
  counts <- Read10X_h5(h5_files[[gsm_id]])
  # Create Seurat object
  seurat_obj <- CreateSeuratObject(
    counts = counts,
    project = sample_name,
    min.cells = 3,
    min.features = 200
  )
  seurat_obj <- subset(seurat_obj, downsample = ncol(seurat_obj)/4) # Keep 3rd of the cells
  # Add comprehensive metadata
  seurat_obj$orig.ident <- sample_name
  seurat_obj$sample <- sample_name
  seurat_obj$week <- week
  seurat_obj$sample_id <- sample_id
  seurat_obj$gsm_id <- gsm_id
  seurat_obj$condition <- ifelse(
    grepl("paraganglia|extraadrenal", sample_name),
    "special", "regular"
  )
  # Calculate mitochondrial percentage
  seurat_obj[["percent.mt"]] <- PercentageFeatureSet(
    seurat_obj,

```

```

        pattern = "^\MT-"
    )

    return(seurat_obj)
})

names(seurat_list) <- sapply(seurat_list, function(x) unique(x$sample))

```

QC and More Filtering

These genes are not to be filtered out

```

cell_type_markers <- list(
  "SCPs" = c("SOX10", "PLP1", "FOXD3"),
  "Chromaffin cells" = c("ELAVL3", "ELAVL4", "PHOX2B", "TH"),
  "Sympathoblasts" = c("STMN2"),
  "Adrenal gland cortex" = c("NR5A1"),
  "Melanocytes" = c("MITF"),
  "Kidney" = c("PAX2"),
  "Subepicardial and abdominal mesenchyme" = c("PRRX1"),
  "Endothelium" = c("PECAM1", "KDR"),
  "Intermediate mesoderm" = c("GATA4", "HAND2"),
  "Liver" = c("HNF4A", "AHSG"),
  "HSCs" = c("SPINK2", "AZU1"),
  "Immune cells" = c("FCGR1A", "CD163"),
  "Erythroid cells" = c("HBA2", "HBB")
)
markers_unique <- unique(unlist(cell_type_markers))

```

```

alist = list()
for (i in 1:length(seurat_list)) {
  setdiff(markers_unique, rownames(seurat_list[[i]]))
  alist <- union(alist, setdiff(markers_unique, rownames(seurat_list[[i]])))
}
to_remove <- unique(alist)

genes_to_conserve <- setdiff(markers_unique, to_remove)
genes_to_conserve

```

1. 'SOX10'
2. 'PLP1'
3. 'FOXD3'
4. 'ELAVL3'
5. 'ELAVL4'
6. 'PHOX2B'
7. 'TH'
8. 'STMN2'
9. 'NR5A1'
10. 'MITF'

11. 'PRRX1'
12. 'PECAM1'
13. 'KDR'
14. 'GATA4'
15. 'HAND2'
16. 'SPINK2'
17. 'FCGR1A'
18. 'CD163'
19. 'HBA2'
20. 'HBB'

```
# QC
seurat_list <- lapply(seurat_list, function(x) {
  x[["percent.mt"]] <- PercentageFeatureSet(x, pattern = "^\u00d7MT-")
  x <- subset(x, subset = nFeature_RNA > 500 & nFeature_RNA < 6000 & percent.mt < 15)
  x <- NormalizeData(x)
  x <- FindVariableFeatures(x, selection.method = "vst", nfeatures = 2000)

  # Add them to the variable features
  VariableFeatures(x) <- union(VariableFeatures(x), genes_to_conserve)
  return(x)
})
```

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

```

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

Normalizing layer: counts

Finding variable features for layer counts

```

Cell Cycle Information

```

# Add cell cycle correction
s_genes <- cc.genes$s.genes
g2m_genes <- cc.genes$g2m.genes
seurat_list <- lapply(seurat_list, function(x) {
  x <- CellCycleScoring(
    x,
    s.features = s_genes,
    g2m.features = g2m_genes,
    set.ident = FALSE
  )
  x$CC.Difference <- x$S.Score - x$G2M.Score # for regression
  return(x)
})

```

```

Warning message:
"The following features are not present in the object: MLF1IP, not searching for symbol synonyms"
Warning message:
"The following features are not present in the object: FAM64A, HN1, not searching for symbol synonyms"
Warning message:
"The following features are not present in the object: MLF1IP, not searching for symbol synonyms"
Warning message:
"The following features are not present in the object: FAM64A, HN1, not searching for symbol synonyms"
Warning message:
"The following features are not present in the object: MLF1IP, not searching for symbol synonyms"

```

```
features <- SelectIntegrationFeatures(object.list = seurat_list)
```

`length(features)`

2000

```
unique(unlist(cell_type_markers))
```

1. 'SOX10'
2. 'PLP1'
3. 'FOXD3'
4. 'ELAVL3'
5. 'ELAVL4'
6. 'PHOX2B'
7. 'TH'
8. 'STMN2'
9. 'NR5A1'
10. 'MITF'
11. 'PAX2'
12. 'PRRX1'
13. 'PECAM1'
14. 'KDR'
15. 'GATA4'
16. 'HAND2'
17. 'HNF4A'
18. 'AHSG'
19. 'SPINK2'
20. 'AZU1'
21. 'FCGR1A'
22. 'CD163'
23. 'HBA2'
24. 'HBB'

Perform Integration

```
# Select integration anchors
anchors <- FindIntegrationAnchors(
  object.list = seurat_list,
  dims = 1:30,
  anchor.features = union(features, genes_to_conserve),
  normalization.method = "LogNormalize"
)
rm(seurat_list) # to open memory space
```

```
anchors
```

An AnchorSet object containing 374062 anchors between 13 Seurat objects
This can be used as input to IntegrateData.

```
# Integrate data
integrated <- IntegrateData(
  anchorset = anchors,
  dims = 1:30,
```

```

    new.assay.name = "integrated"
}

# Switch to integrated assay for downstream analysis
DefaultAssay(integrated) <- "integrated"

```

Scale data

Scaling and regression for cell cycle

```
integrated <- ScaleData(integrated, vars.to.regress = "CC.Difference", verbose = FALSE)
```

Run PCA

```
integrated <- RunPCA(integrated)
```

PC_ 1

Positive: HLA-E, EGFL7, IFITM3, IGFBP4, KDR, TMSB4X, FLT1, ELK3, RAMP2, PLVAP
CD93, CYBA, CAVIN2, CALCRL, CDH5, ARHGAP29, TGFBR2, ANXA2, PLPP3, TFPI
ETS1, EMCN, PECAM1, PRCP, ADGRF5, TMEM88, CLDN5, ESAM, CD109, NEAT1

Negative: NRCAM, DLK1, CDH2, FDXR, STAR, CADM1, APOA1, PEG3, KCNK3, NR5A1
DHCR24, ALDH1A2, PEG10, NOV, APOE, ASB4, PEBP1, MC2R, PPIF, DPP10
RALYL, INHA, MCF2, CYP11A1, COL15A1, SNCG, CACNB2, MGARP, TBX3, SLC16A9

PC_ 2

Positive: COL1A1, COL3A1, COL1A2, PLAC9, COL5A1, COL12A1, GPC3, COL5A2, ISLR, CDH11
CXCL12, PCOLCE, FBN1, COL6A3, VIM, CALD1, OGN, FZD1, PCDH7, DCN
COL16A1, COL6A1, PDE5A, LRRC17, POSTN, PDGFRA, PRRX1, CD248, SULT1E1, TSHZ2

Negative: LAPTM5, ARHGAP18, HLA-B, MAN1A1, CD74, TYROBP, C1QC, MEF2C, SRGN, FYB1
LYVE1, CSF1R, DAB2, MRC1, C1QB, CD163, STAB1, C1QA, FCER1G, CD36
MS4A6A, HCST, PLD4, MS4A7, PTPRE, DOCK8, VSIG4, CD83, GYPC, CYBB

PC_ 3

Positive: NOSTRIN, HSPG2, TIMP3, SQLE, SPARC, GNG11, TSPAN13, MGST2, CALCRL, PLPP3
HPGD, KDR, FLT1, SH3BP5, PRCP, CAVIN2, TMEM47, CDH5, BTNL9, PLVAP
MMRN2, GJA1, TMEM88, CLDN5, F8, ROBO4, TM4SF18, CLEC14A, SNCG, PEG10

Negative: TYROBP, C1QA, C1QB, C1QC, FYB1, CSF1R, CYBB, CD163, PLD4, MS4A7
FCGR1A, AIF1, VSIG4, HCST, FGD2, RUNX1, MS4A4A, MPEG1, NCKAP1L, RGS1
ADAP2, FCGR3A, TFEC, PTPRC, CD68, FCER1G, FOLR2, CCR1, TYMP, CD53

PC_ 4

Positive: CTSC, PLAGL1, C7, COL1A2, APOE, NRK, COL14A1, DCN, VCAN, NR2F1
COL3A1, TPM2, FBN2, RARRES2, DAB2, CXCL12, COL1A1, GPC6, ZEB2, SPARC
AXL, CDH11, NPR3, TGFB1, HLA-DRB1, FSTL1, COL5A1, LRRC17, COL12A1, COL11A1

Negative: STMN2, DBH, EML5, HAND2-AS1, PCSK1N, RGS5, HAND2, CHGB, MIAT, PHOX2B
EEF1A2, ELAVL4, GATA2, PHOX2A, CHGA, ISL1, CHRNA3, ELAVL3, CD24, TFAP2B
TUBB2B, STMN4, LINC00682, GAL, GATA3, DPP6, VSTM2L, SCN3B, SCG2, SLC18A1

PC_ 5

Positive: HMGB1, STMN1, TUBB, HMGN2, H2AFZ, TMSB4X, JUN, KLF6, HSPA5, CALM2
TUBA1B, MEG3, FOS, DUT, CST3, NAMPT, TMPO, HELLS, HSPD1, CCND1

SMC4, VMP1, PEBP1, MEF2C, MEG8, IRF1, IER2, TOP2A, SLC8A1, CYBA
 Negative: SLC4A1, ALAS2, HBG1, HBA2, HBA1, AHSP, HBG2, HEMGN, HBM, GYPA
 GYPB, SPTA1, TENT5C, HMBS, SELENBP1, SLC25A37, EPB42, HBB, TRIM58, SNCA
 KLF1, BPGM, MYL4, ANK1, TMEM56, TMCC2, NFE2, BLVRB, HBQ1, RBM38

Run UMAP

```
integrated <- RunUMAP(integrated, dims = 1:50)
```

19:30:46 UMAP embedding parameters a = 0.9922 b = 1.112

19:30:46 Read 19803 rows and found 50 numeric columns

19:30:46 Using Annoy for neighbor search, n_neighbors = 30

19:30:46 Building Annoy index with metric = cosine, n_trees = 50

0% 10 20 30 40 50 60 70 80 90 100%

[-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|]

* * * * *

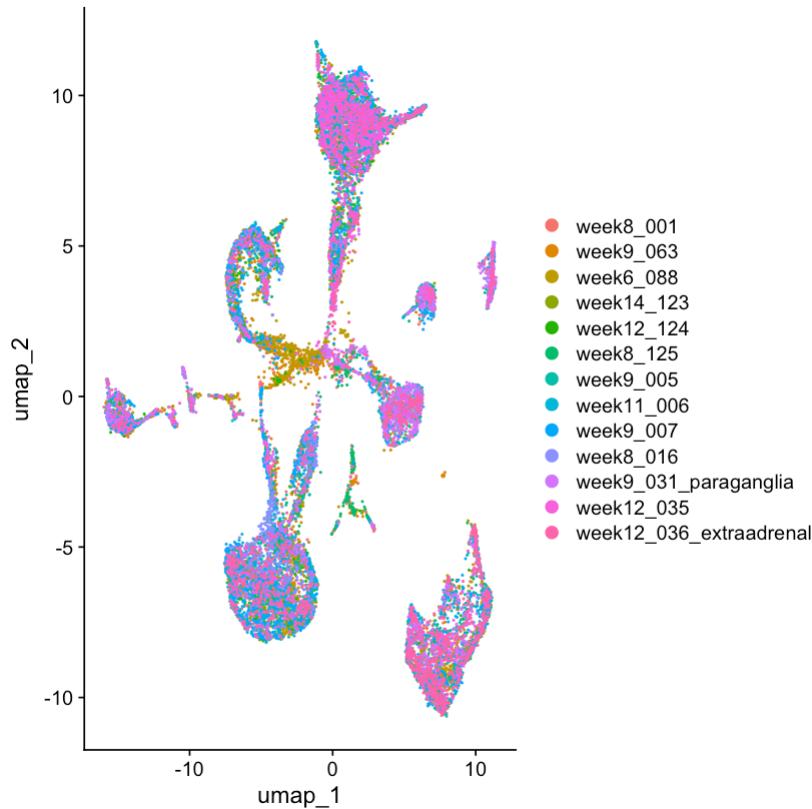
```
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
*  
|  
  
19:30:47 Writing NN index file to temp file /var/folders/wl/jrkngsm57b944tj7rtjg12000000gn/T//RtmpM  
19:30:47 Searching Annoy index using 1 thread, search_k = 3000  
19:30:49 Annoy recall = 100%  
19:30:50 Commencing smooth kNN distance calibration using 1 thread  
with target n_neighbors = 30  
19:30:50 Initializing from normalized Laplacian + noise (using RSpectra)  
19:30:51 Commencing optimization for 200 epochs, with 912616 positive edges  
19:30:51 Using rng type: pcg  
19:30:54 Optimization finished
```

```
integrated
```

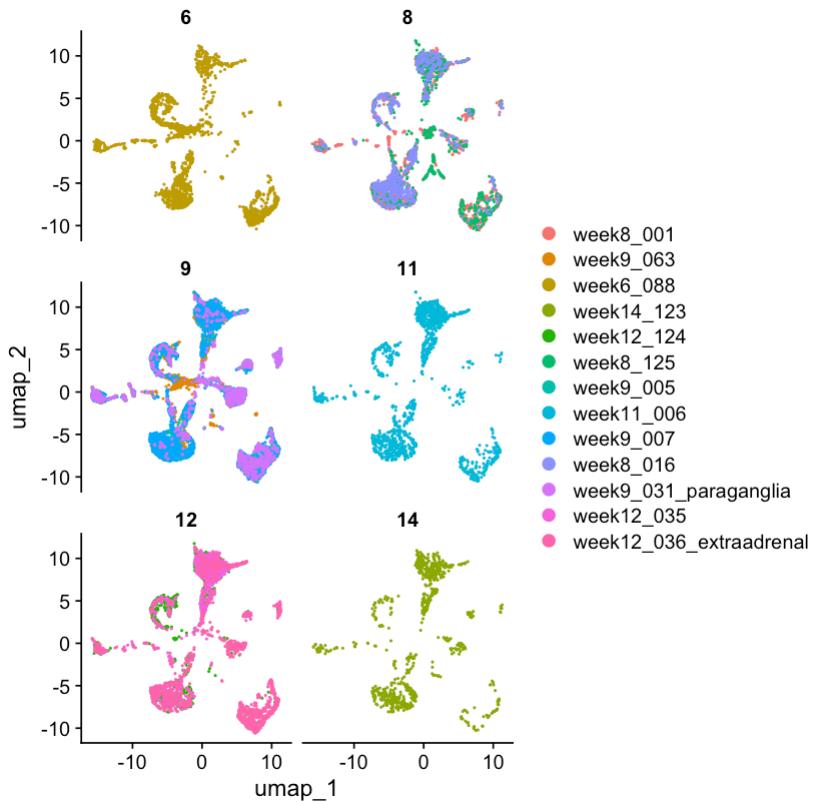
An object of class Seurat
28181 features across 19803 samples within 2 assays
Active assay: integrated (2000 features, 2000 variable features)
2 layers present: data, scale.data
1 other assay present: RNA
2 dimensional reductions calculated: pca, umap

Umap Plot

```
umapfig <- DimPlot(integrated, raster.dpi = c(600,400))
umapfig
```



```
DimPlot(integrated, raster.dpi = c(600,400),split.by = "week", ncol = 2)
```



```
saveRDS(integrated, "data/processed/integrated.rds")
```

```
library(ggplot2)
ggsave('plots/umap_after_integration.pdf', width = 10, height = 7)
```

Notes: Integration Successful

UMAP, clustering, and annotation

```
library(Seurat)
library(ggplot2)
library(dplyr)
library(patchwork)
library(ggplot2)
setwd("~/dev/CCRItask")
```

Loading required package: SeuratObject

Loading required package: sp

Attaching package: 'SeuratObject'

The following objects are masked from 'package:base':

intersect, t

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

filter, lag

The following objects are masked from 'package:base':

intersect, setdiff, setequal, union

Load the integrated data

```
integrated <- readRDS("data/processed/integrated.rds")
```

Perfrom Clustering

```
integrated <- FindNeighbors(integrated, dims = 1:30)
integrated <- FindClusters(integrated, resolution = 0.1)
```

Computing nearest neighbor graph

Computing SNN

Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck

Number of nodes: 19803

Number of edges: 805203

Running Louvain algorithm...

Maximum modularity in 10 random starts: 0.9772

Number of communities: 12

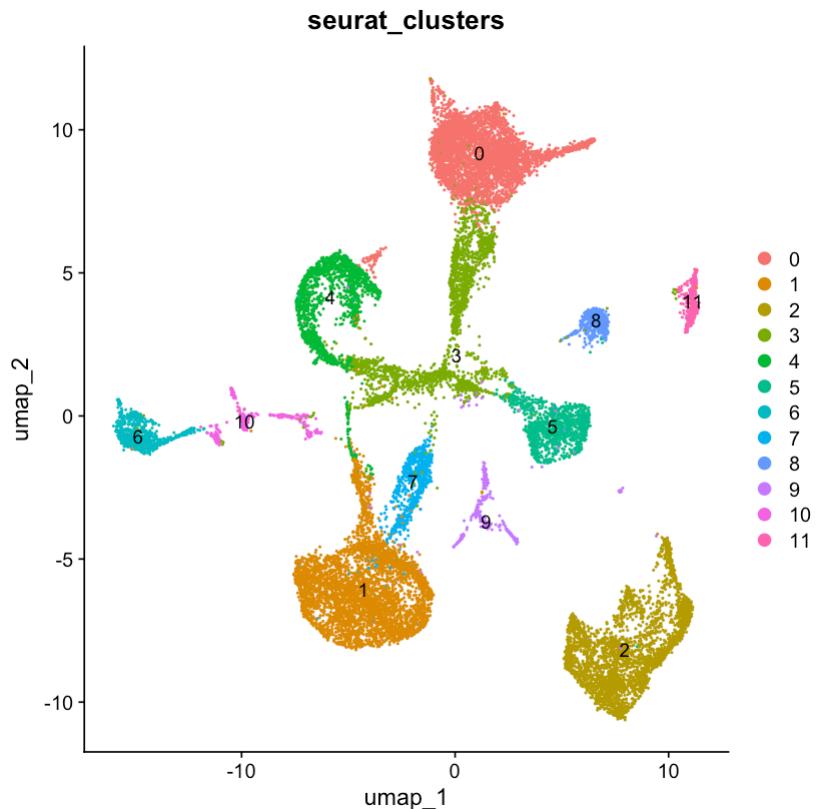
Elapsed time: 2 seconds

```
head(integrated[[]])
```

A data.frame: 6 x 15

| orig.idem | Count | TRNA | Sample | PN | Week | sample | gsm_id | condition | percentS | ScoreG2M.S | Phase | CC.Dif | Integrated | rast | incluster |
|----------------------------------|-------|------------|--------|-------|----------|------------------|--------|------------|----------|------------|-------|--------|------------|-------|-----------|
| <chr> | <dbl> | <int> | <chr> | <dbl> | <chr> | <chr> | <chr> | <chr> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> |
| AAACGGAAAGTGGCAAACTC- week8_8001 | 001 | GSM4446535 | g | r | 9.716130 | - | G1 | 0.14416732 | 1 | | | | | | |
| 1_1 | | | | | | 0.0101333543011 | | | | | | | | | |
| AAACGGCTGTAAATCCTGAA- | 001 | GSM4446535 | g | r | 3.954196 | - | G1 | 0.09915505 | 8 | | | | | | |
| 1_1 | | | | | | 0.0841684833235 | | | | | | | | | |
| AAAGAACGACGAGGCGGAT- | 001 | GSM4446535 | g | r | 1.818182 | - | G1 | 0.08046040 | 0 | | | | | | |
| 1_1 | | | | | | 0.09471020751724 | | | | | | | | | |
| AAAGGATAAGTCATTG- week8_8001 | 001 | GSM4446535 | g | r | 4.649305 | - | G1 | 0.22243079 | 0 | | | | | | |
| 1_1 | | | | | | 0.0464322688638 | | | | | | | | | |
| AAAGGATTCATGCCATG- week8_8001 | 001 | GSM4446535 | g | r | 4.886914 | - | G1 | 0.12607253 | 8 | | | | | | |
| 1_1 | | | | | | 0.1312632573361 | | | | | | | | | |
| AAAGGGCGATGAAATAAC- | 001 | GSM4446535 | g | r | 3.492117 | - | G1 | 0.12725061 | 10 | | | | | | |
| 1_1 | | | | | | 0.0414270686783 | | | | | | | | | |

```
DimPlot(integrated,reduction = "umap", group.by = "seurat_clusters", label = TRUE)
```



Cluster annotation

From the <https://www.nature.com/articles/s41588-021-00818-x>

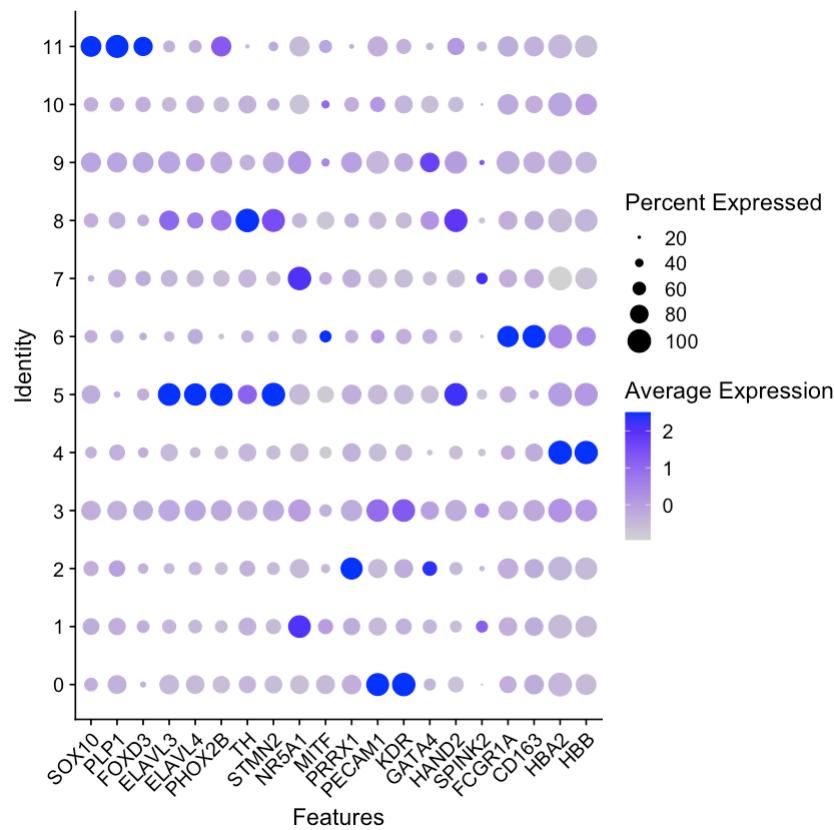
```
cell_type_markers <- list(
  "SCPs" = c("SOX10", "PLP1", "FOXD3"),
  "Chromaffin cells" = c("ELAVL3", "ELAVL4", "PHOX2B", "TH"),
  "Sympathoblasts" = c("STMN2"),
  "Adrenal gland cortex" = c("NR5A1"),
  "Melanocytes" = c("MITF"),
  #Kidney" = c("PAX2"),
  "Subepicardial and abdominal mesenchyme" = c("PRRX1"),
  "Endothelium" = c("PECAM1", "KDR"),
  "Intermediate mesoderm" = c("GATA4", "HAND2"),
  #Liver" = c("HNF4A", "AHSG"),
  "HSCs" = c("SPINK2"), # AZU1
  "Immune cells" = c("FCGR1A", "CD163"),
  "Erythroid cells" = c("HBA2", "HBB"))
)

markers_unique  = unique(unlist(cell_type_markers))

length(unique(unlist(cell_type_markers)))
```

Dotplot to map clusters to cell types

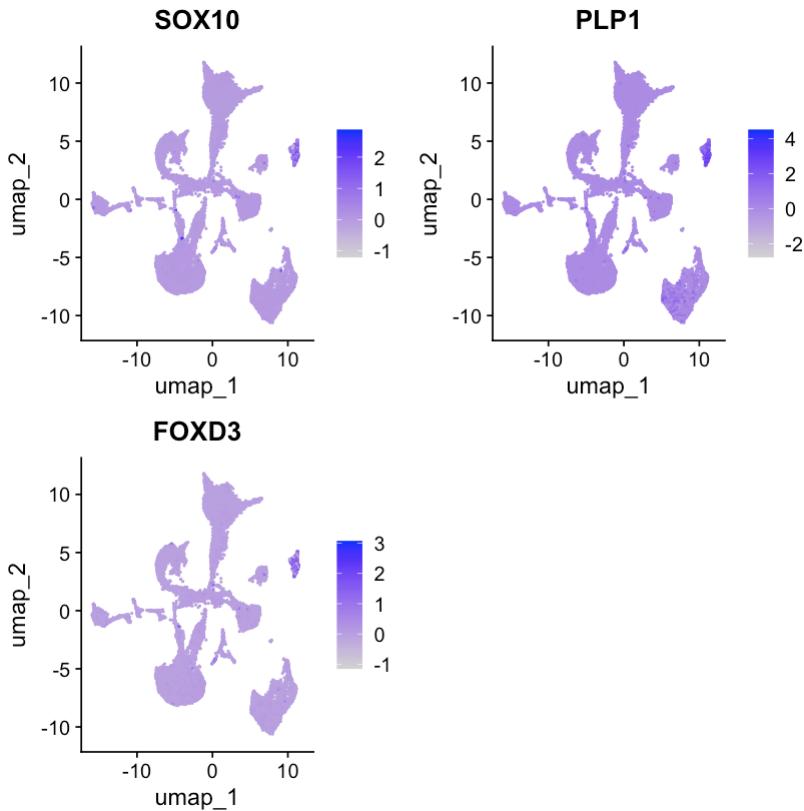
```
DotPlot(integrated, features = unique(unlist(cell_type_markers))) + RotatedAxis()
```



in depth analysis for manual annotation

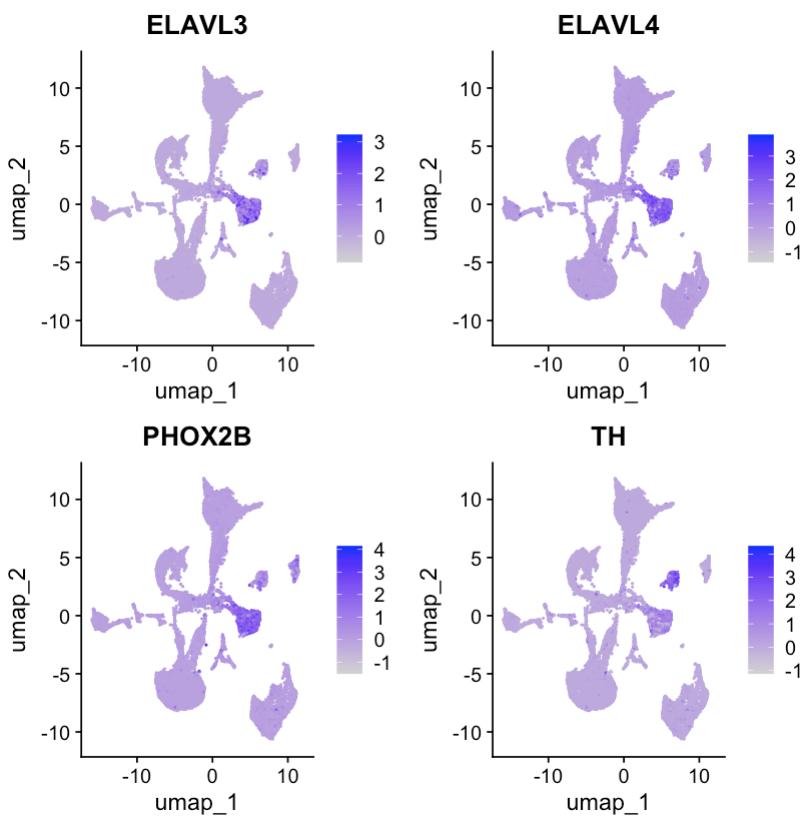
SCPs

```
FeaturePlot(integrated, c("SOX10", "PLP1", "FOXD3"), ncol=2, raster.dpi = c(800,100))
```

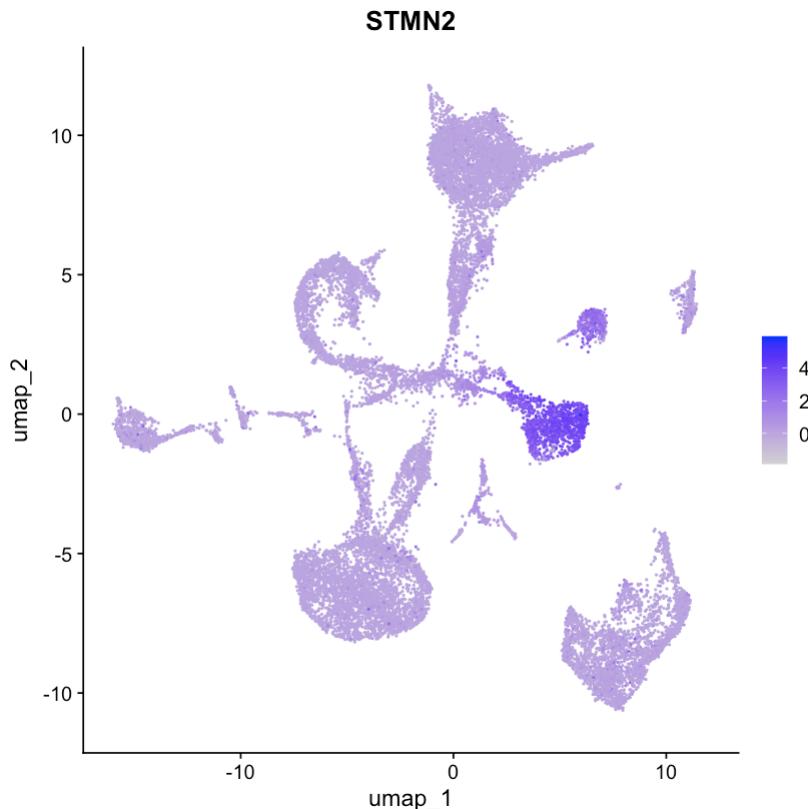


chromaffin cells

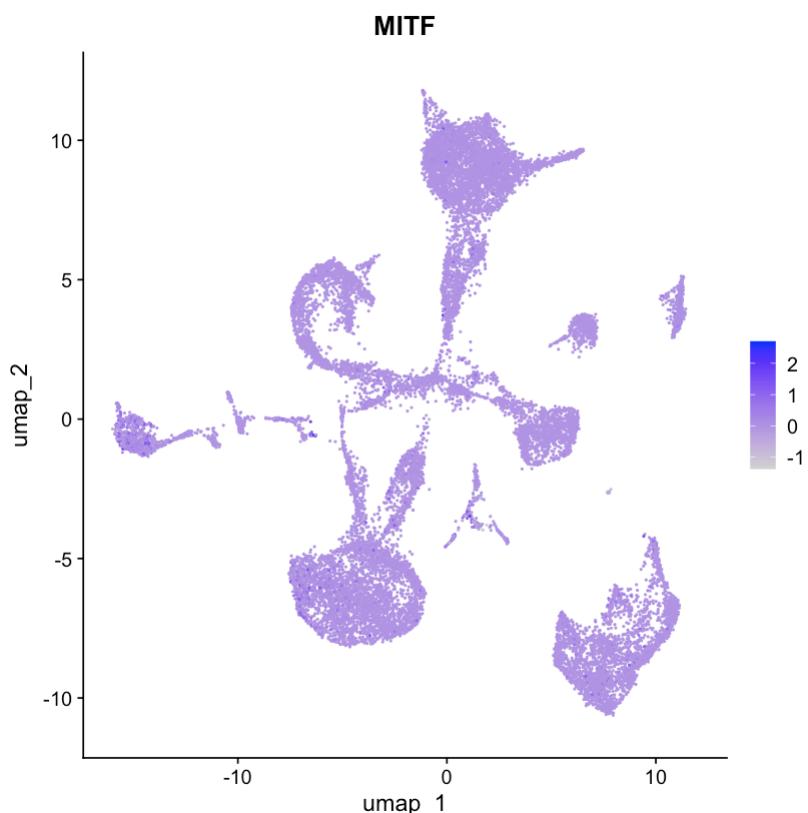
```
FeaturePlot(integrated, c("ELAVL3", "ELAVL4", "PHOX2B", "TH"), ncol=2, raster.dpi = c(800,100))
```



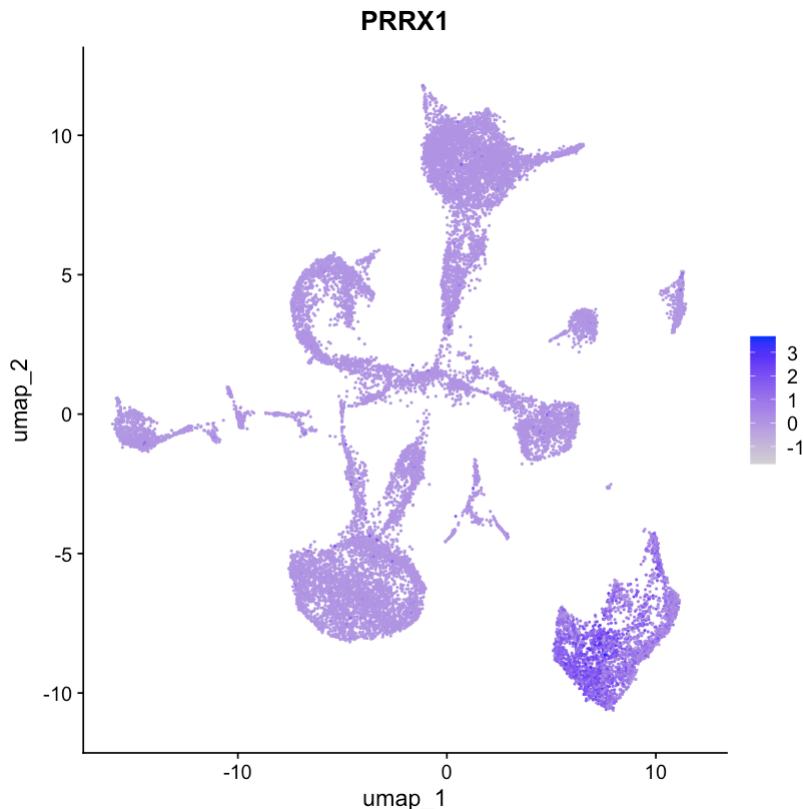
```
FeaturePlot(integrated, c("STMN2"), ncol=1, raster.dpi = c(800,100))
```



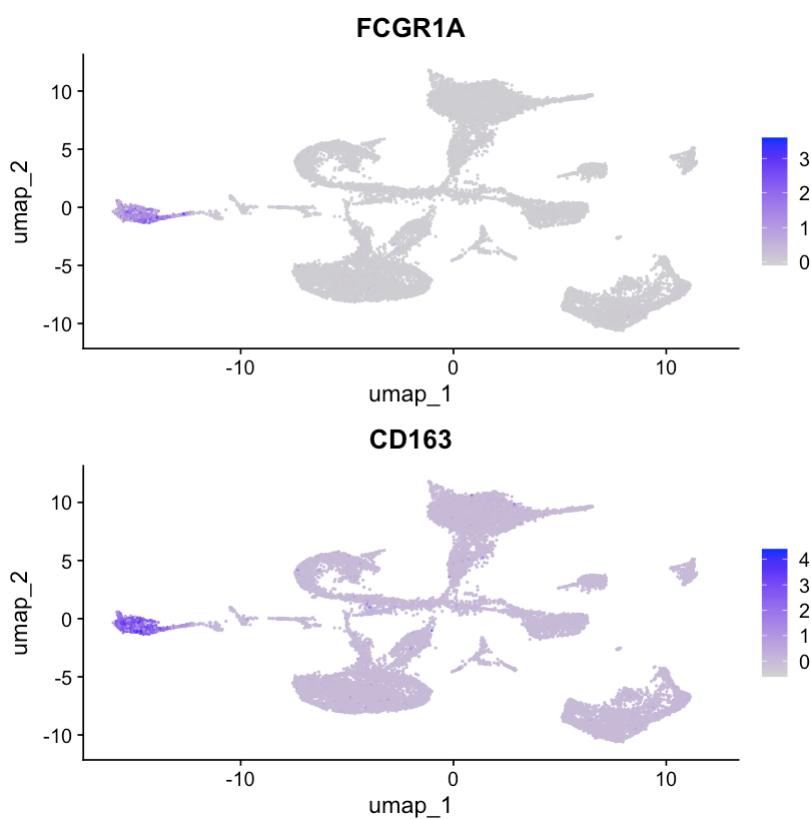
```
FeaturePlot(integrated, c("MITF"), ncol=1, raster.dpi = c(800,100))
```



```
FeaturePlot(integrated, c("PRRX1"), ncol=1, raster.dpi = c(800,100))
```



```
FeaturePlot(integrated, c("FCGR1A", "CD163"), ncol=1, raster.dpi = c(800,100))
```



```

cluster <- list(
  "11" = "SCPs", #= c("SOX10", "PLP1", "FOXD3"),
  "8" = "Chromaffin cells", #= c("ELAVL3", "ELAVL4", "PHOX2B", "TH"),
  "5" = "Sympathoblasts", #= c("STMN2"),
  "1" = "Adrenal gland cortex", #= c("NR5A1"),
  # "6" = "Melanocytes", #= c("MITF"),
  # "Kidney" = c("PAX2"),
  "2" = "Subepicardial and abdominal mesenchyme", #= c("PRRX1"),
  "0" = "Endothelium", #= c("PECAM1", "KDR"),
  "3" = "Intermediate mesoderm", #= c("GATA4", "HAND2"),
  # "Liver" = c("HNF4A", "AHSG"),
  "7" = "HSCs", #= c("SPINK2"), # AZU1
  "6" = "Immune cells", #= c("FCGR1A", "CD163"),
  "4" = "Erythroid cells" #= c("HBA2", "HBB")
)

```

Mapping clusters to cell types

```

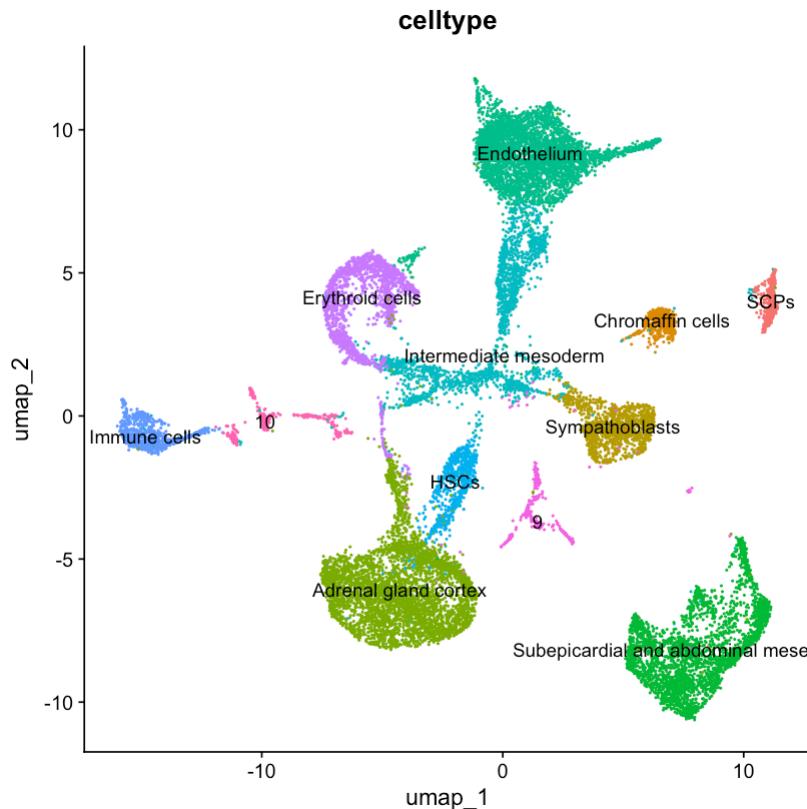
# Convert list to a character vector for easier handling
cluster <- unlist(cluster)
# Rename the identities
integrated <- RenameIdents(integrated, cluster)
# Save the renamed cluster identities
integrated$celltype <- Idents(integrated)

```

```

umap_ann <- DimPlot(integrated, group.by = "celltype", label = TRUE) + NoLegend()
umap_ann

```



```
ggsave("plots/umap_annotated.pdf", umap_ann, width = 10, height = 10)
```

Automatic cluster annotation

did not work properly

```
# Score each cell for each cell type based on marker genes
integrated <- AddModuleScore(
  integrated,
  features = cell_type_markers,
  name = names(cell_type_markers),
  ctrl = 20,
  replace = TRUE
)
```

```
install.packages('devtools')
devtools::install_github('immunogenomics/presto')
```

The downloaded binary packages are in
`/var/folders/wl/jrkngsm57b944tj7rtjg1200000gn/T//Rtmp3lmekZ/downloaded_packages`

Using GitHub PAT from the git credential store.

Skipping install of 'presto' from a github remote, the SHA1 (7636b3d0) has not changed since last install
 Use `force = TRUE` to force installation

Dot Plot for top5 markers for each cluster

```
# Find all markers for each cluster
markers <- FindAllMarkers(object = integrated,
                           only.pos = TRUE,      # Only consider positive markers
                           min.pct = 0.25,       # Minimum detection fraction
                           logfc.threshold = 0.25) # Minimum log fold change
```

Calculating cluster SCPs

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster Chromaffin cells

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster Sympathoblasts

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster Adrenal gland cortex

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster Subepicardial and abdominal mesenchyme

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster Endothelium

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster Intermediate mesoderm

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster HSCs

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster Immune cells

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster Erythroid cells

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):
"NaNs produced"
```

Calculating cluster 9

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):  
"NaNs produced"  
Calculating cluster 10
```

```
Warning message in mean.fxn(object[features, cells.1, drop = FALSE]):  
"NaNs produced"  
Warning message in mean.fxn(object[features, cells.2, drop = FALSE]):  
"NaNs produced"
```

```
head(markers)
```

A data.frame: 6 x 7

| | p_val <dbl> | avg_log2FC <dbl> | pct.1 <dbl> | pct.2 <dbl> | p_val_adj <dbl> | cluster <fct> | gene <chr> |
|--------|----------------|---------------------|----------------|----------------|--------------------|------------------|---------------|
| PTPRZ1 | 1.063855e-243 | 6.412992 | 0.990 | 0.783 | 2.127709e-240 | SCPs | PTPRZ1 |
| ERBB3 | 1.696545e-233 | 8.156112 | 0.972 | 0.566 | 3.393090e-230 | SCPs | ERBB3 |
| DST | 5.987636e-231 | 3.781085 | 1.000 | 0.891 | 1.197527e-227 | SCPs | DST |
| PLP1 | 7.904205e-229 | 7.297576 | 0.970 | 0.727 | 1.580841e-225 | SCPs | PLP1 |
| TRPM3 | 3.513331e-227 | 7.403232 | 0.970 | 0.622 | 7.026663e-224 | SCPs | TRPM3 |
| MPZ | 1.720136e-225 | 8.064614 | 0.965 | 0.670 | 3.440272e-222 | SCPs | MPZ |

```
# Get top 5 markers per cluster
top5 <- markers %>%
  group_by(cluster) %>%
  top_n(n = 5, wt = avg_log2FC)%>%
  arrange(cluster, desc(avg_log2FC))
```

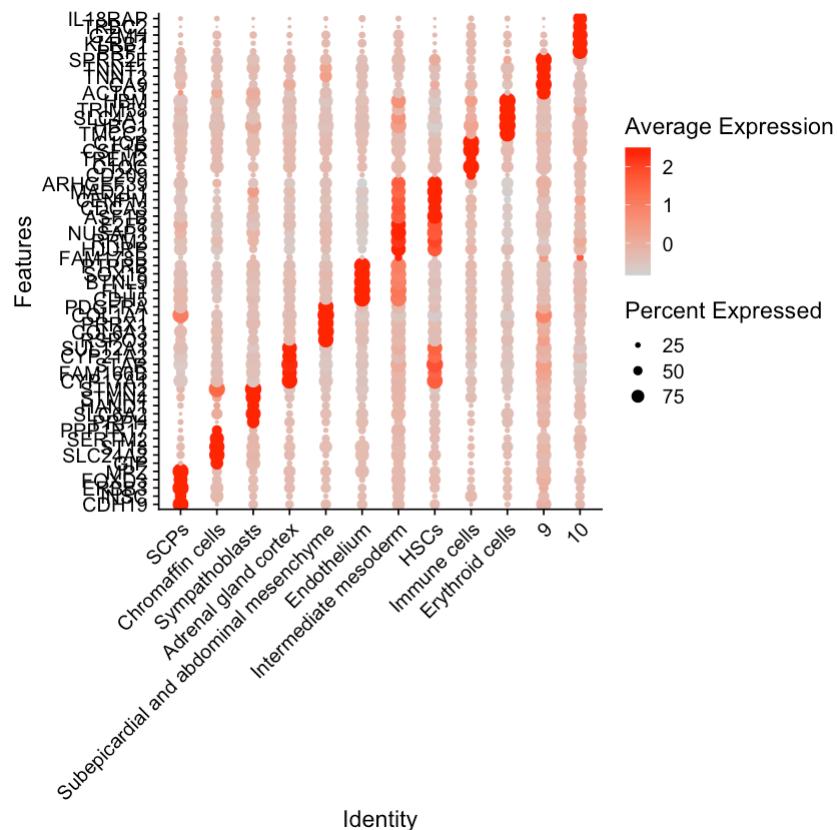
```
head(top5)
```

A grouped_df: 6 x 7

| p_val <dbl> | avg_log2FC <dbl> | pct.1 <dbl> | pct.2 <dbl> | p_val_adj <dbl> | cluster <fct> | gene <chr> |
|----------------|---------------------|----------------|----------------|--------------------|------------------|---------------|
| 1.150230e-221 | 8.924888 | 0.952 | 0.516 | 2.300459e-218 | SCPs | CDH19 |
| 5.514697e-88 | 8.242869 | 0.656 | 0.437 | 1.102939e-84 | SCPs | INSC |
| 1.696545e-233 | 8.156112 | 0.972 | 0.566 | 3.393090e-230 | SCPs | ERBB3 |
| 6.326074e-136 | 8.108642 | 0.824 | 0.519 | 1.265215e-132 | SCPs | FOXD3 |
| 1.720136e-225 | 8.064614 | 0.965 | 0.670 | 3.440272e-222 | SCPs | MPZ |

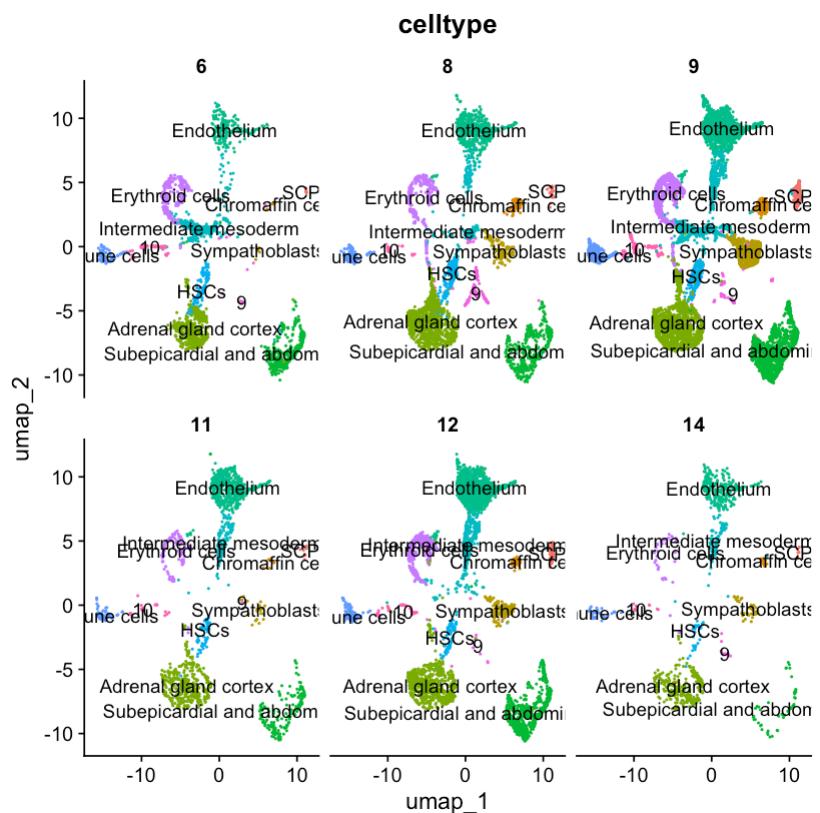
| p_val <dbl> | avg_log2FC <dbl> | pct.1 <dbl> | pct.2 <dbl> | p_val_adj <dbl> | cluster <fct> | gene <chr> |
|---------------|------------------|-------------|-------------|-----------------|------------------|------------|
| 1.815618e-113 | 8.241624 | 0.735 | 0.382 | 3.631235e-110 | Chromaffin cells | GIP |

```
# Create dot plot
dp <- DotPlot(object = integrated,
               features = unique(top5$gene), # unique genes top5
               cols = c("lightgrey", "red"), # to match the paper
               dot.scale = 4) +
  coord_flip() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
dp
```



```
ggsave("plots/dotplot_top5_perCluster.pdf", dp, width = 8, height = 16)
```

```
umap_ann_week <- DimPlot(integrated, group.by = "celltype", label = TRUE, split.by = "week", ncol=3)
ggsave("plots/umap_ann_week.png", width = 10, height = 10)
umap_ann_week
```



```
saveRDS(integrated, "data/processed/integrated_annotated.rds")
```

Adreanal Medulla Reclustering

```
library(Seurat)
library(ggplot2)
```

Loading required package: SeuratObject

Loading required package: sp

Attaching package: 'SeuratObject'

The following objects are masked from 'package:base':

intersect, t

Load the data

```
seu <- readRDS("data/processed/integrated_annotated.rds")
```

Subset AMC (Schwann cell precursors (SCPs), Chromaffin cells, Sympathoblasts)

```
unique(seu$celltype)
```

1. Adrenal gland cortex
2. Chromaffin cells
3. Endothelium
4. 10
5. Subepicardial and abdominal mesenchyme
6. Intermediate mesoderm
7. Sympathoblasts
8. Erythroid cells
9. HSCs
10. 9
11. Immune cells
12. SCPs

Levels: 1. ‘SCPs’ 2. ‘Chromaffin cells’ 3. ‘Sympathoblasts’ 4. ‘Adrenal gland cortex’ 5. ‘Subepicardial and abdominal mesenchyme’ 6. ‘Endothelium’ 7. ‘Intermediate mesoderm’ 8. ‘HSCs’ 9. ‘Immune cells’ 10. ‘Erythroid cells’ 11. ‘9’ 12. ‘10’

```
seu_subset <- subset(seu, subset = celltype %in% c("SCPs", "Chromaffin cells", "Sympathoblasts"))
```

```
unique(seu_subset$celltype)
```

1. Chromaffin cells
2. Sympathoblasts
3. SCPs

Levels: 1. ‘SCPs’ 2. ‘Chromaffin cells’ 3. ‘Sympathoblasts’ 4. ‘Adrenal gland cortex’ 5. ‘Subepicardial and abdominal mesenchyme’ 6. ‘Endothelium’ 7. ‘Intermediate mesoderm’ 8. ‘HSCs’ 9. ‘Immune cells’ 10. ‘Erythroid cells’ 11. ‘9’ 12. ‘10’

Recluster

```
seu_subset <- FindVariableFeatures(seu_subset)
```

```
Warning message in FindVariableFeatures.Assay(object = object[[assay]], selection.method = selection
"selection.method set to 'vst' but count slot is empty; will use data slot instead"
Warning message in eval(predvars, data, env):
"NaNs produced"
Warning message in hvf.info$variance.expected[not.const] <- 10^fit$fitted:
"number of items to replace is not a multiple of replacement length"
```

```
seu_subset <- ScaleData(seu_subset)
seu_subset <- RunPCA(seu_subset)
seu_subset <- FindNeighbors(seu_subset, dims = 1:30)
```

Centering and scaling data matrix

PC_ 1

Positive: STMN2, RGS5, CHGB, DBH, PCSK1N, EEF1A2, HAND2-AS1, EML5, CD24, BEX1
GATA3, HAND2, SYT1, MIAT, MAP1B, GATA2, CHGA, CNTN1, BASP1, PHOX2A
TUBB2B, ELAVL4, ISL1, CHRNA3, DPP6, SEZ6L2, RAMP1, RGS4, KIF21A, ELAVL3
Negative: PLP1, PTPRZ1, EDNRB, COL5A2, ERBB3, OLFML2A, MPZ, SPARC, S100B, CDH19
VCAN, TGFBR2, POSTN, MOXD1, TRPM3, NR2F2, ABCA8, GPM6B, TTYH1, PLAT
METRN, NID1, HSPG2, COL2A1, SOX10, LM04, COL1A1, LGI4, GAS7, PLEKHA4

PC_ 2

Positive: TUBB, SOX4, STMN1, TUBA1A, RTN1, GAP43, TUBB2B, TMSB15A, STMN4, ELAVL4
SOX11, TUBA1B, SLC6A2, HMGB1, SYNE2, SMC4, TMSB4X, ASPM, SLC38A1, MLLT11
KIF21A, PLXNA4, RBFOX1, VIM, PHOX2B, TOP2A, PRC1, MKI67, JPT1, TMPO
Negative: DLK1, PENK, SLC24A2, ST18, INSM1, CCSER1, PNMT, VWA5B2, NDUFA4L2, CHGA
ADM, ARC, SERTM2, C1QL1, CDKN1C, SCARB1, F10, TH, CCND2, HTATSF1
PCSK2, VEGFA, JUNB, ROBO2, DGKK, SCG2, CARTPT, HIST1H2AC, SMIM1, RALYL

PC_ 3

Positive: CENPF, MKI67, TOP2A, NUSAP1, CENPE, CDK1, ASPM, NUF2, GTSE1, TTK

RRM2, UBE2C, KIF23, CCNB1, TPX2, PBK, CCNA2, PIMREG, HMGB2, CDCA8
NCAPG, AURKB, PTTG1, CDKN3, PLK1, FOXM1, DLGAP5, BUB1, KIF11, BIRC5
Negative: SOX4, PRPH, STMN4, MAOA, TUBA1A, GAP43, CCND1, GAL, TUBB2B, NEFL
RBFOX1, RTN1, NPY, C4orf48, PLPPR3, PLXNA4, UCHL1, MAPT, DPYSL3, PHOX2B
ARHGDIG, SCN9A, SCN3B, TUBB2A, ANXA2, TBX2, STMN2, MAP1B, ANK2, HBG2

PC_ 4

Positive: TMEM88, PLVAP, CD93, CD34, EMCN, F8, ROBO4, TIE1, FLT1, SOX18
CD109, RASGRP3, CAVIN2, A2M, CDH5, ADGRF5, TFPI, CALCRL, VAMP5, BTNL9
LMO2, HLA-E, TM4SF18, KDR, NOSTRIN, PECAM1, LRRC32, ICAM2, LYVE1, PCDH12
Negative: NUF2, MKI67, CENPF, KIFC1, GAS2L3, ASPM, PTTG1, DST, SLITRK2, TROAP
DLGAP5, MXD3, KIF2C, FLRT3, KIF4A, UBE2C, RTKN2, ZEB2, AURKB, MNS1
EGFLAM, FOXM1, NCAPG, PLP1, TPX2, BIRC5, GTSE1, COL25A1, ERBB3, S100B

PC_ 5

Positive: NR5A1, MGARP, CYP11A1, FDXR, ASB4, SIGLEC11, STAR, APOE, TCEA3, ALDH1A2
GIPC2, MRAP, SNCG, CCDC141, MT3, MC2R, NRK, MCF2, ZG16B, AC020571.1
FDX1, DHCR24, SULT2A1, GRB14, RBM47, NOV, TNNI3, INHA, CYP17A1, MAP3K15
Negative: PLVAP, FLT1, CALCRL, CAVIN2, F8, PECAM1, CD93, TM4SF18, CLDN5, BTNL9
CDH5, CD34, ICAM2, SOX18, TMEM88, TIE1, TEK, ROBO4, PROCR, KDR
EMCN, ADGRF5, PCAT19, FGF23, CETP, RASGRP3, HLA-E, IRX3, EHD3, CEACAM1

Computing nearest neighbor graph

Computing SNN

```
seu_subset <- FindClusters(seu_subset, resolution = 0.1)
seu_subset <- RunUMAP(seu_subset, dims = 1:40)
```

Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck

Number of nodes: 2118

Number of edges: 94548

Running Louvain algorithm...

Maximum modularity in 10 random starts: 0.9551

Number of communities: 4

Elapsed time: 0 seconds

20:32:52 UMAP embedding parameters a = 0.9922 b = 1.112

20:32:52 Read 2118 rows and found 40 numeric columns

20:32:52 Using Annoy for neighbor search, n_neighbors = 30

20:32:52 Building Annoy index with metric = cosine, n_trees = 50

0% 10 20 30 40 50 60 70 80 90 100%

[----|----|----|----|----|----|----|----|----|

*
*

*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
|

20:32:53 Writing NN index file to temp file /var/folders/wl/jrkngsm57b944tj7rtjg12000000gn/T//Rtmpal

```

20:32:53 Searching Annoy index using 1 thread, search_k = 3000
20:32:53 Annoy recall = 100%
20:32:53 Commencing smooth kNN distance calibration using 1 thread
with target n_neighbors = 30
20:32:53 Initializing from normalized Laplacian + noise (using RSpectra)
20:32:53 Commencing optimization for 500 epochs, with 93720 positive edges
20:32:53 Using rng type: pcg
20:32:54 Optimization finished

```

```
DimPlot(seu_subset)
```

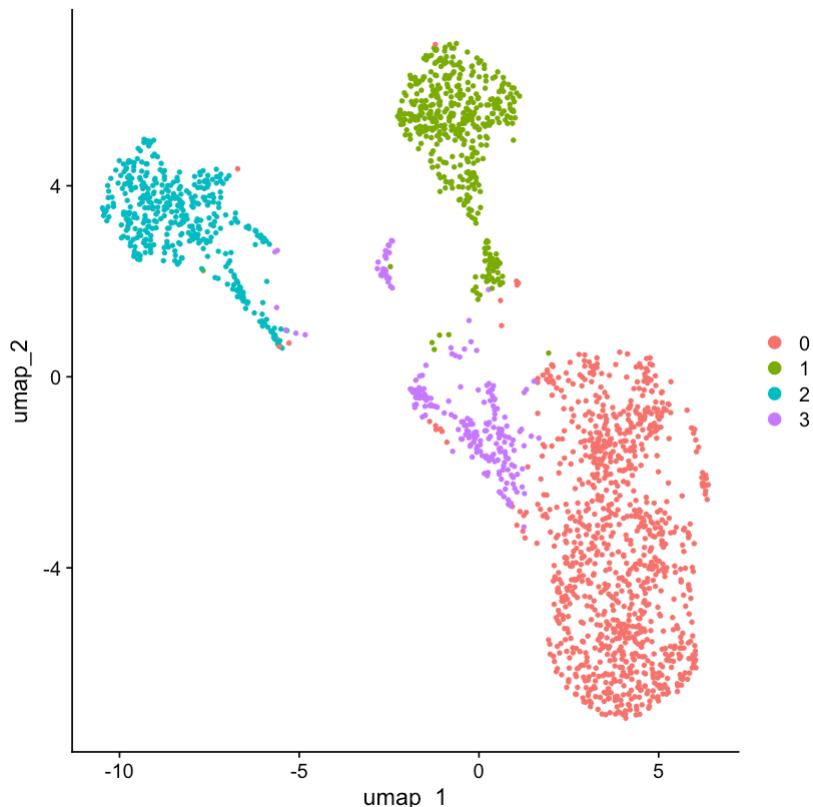


Fig 2 Markers from the paper

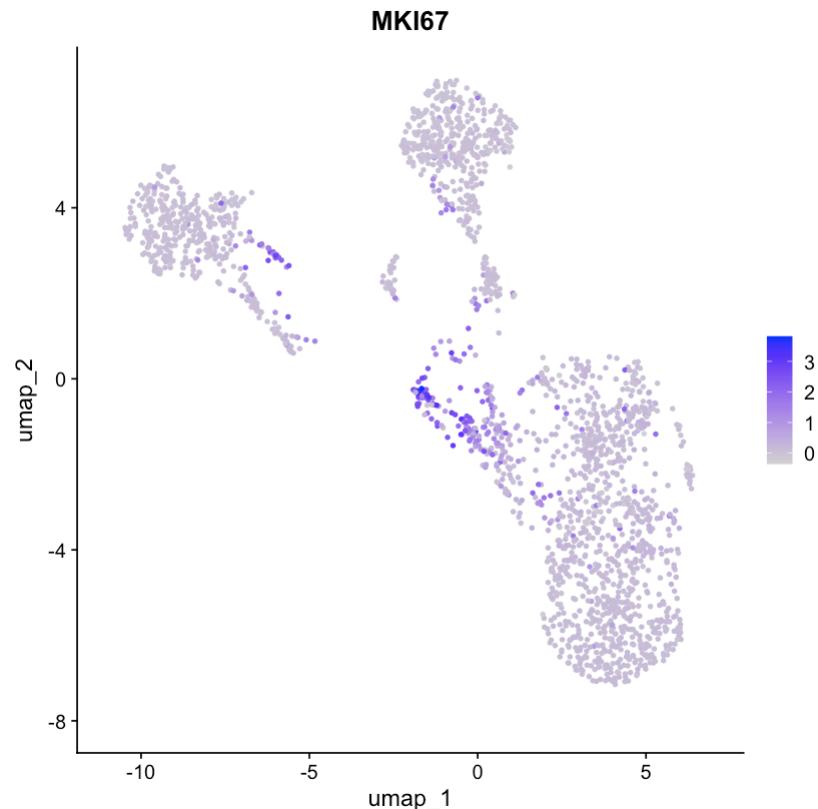
```

markers <- list(
  "Proliferating sympathoblasts" = "MKI67",
  "Sympathoblasts" = c("ELAVL4", "ISL1", "PRPH"),
  "SCPs" = c("SOX10", "PLP1"),
  "Chromaffin cells" = c("CHGA", "PNMT")
)

```

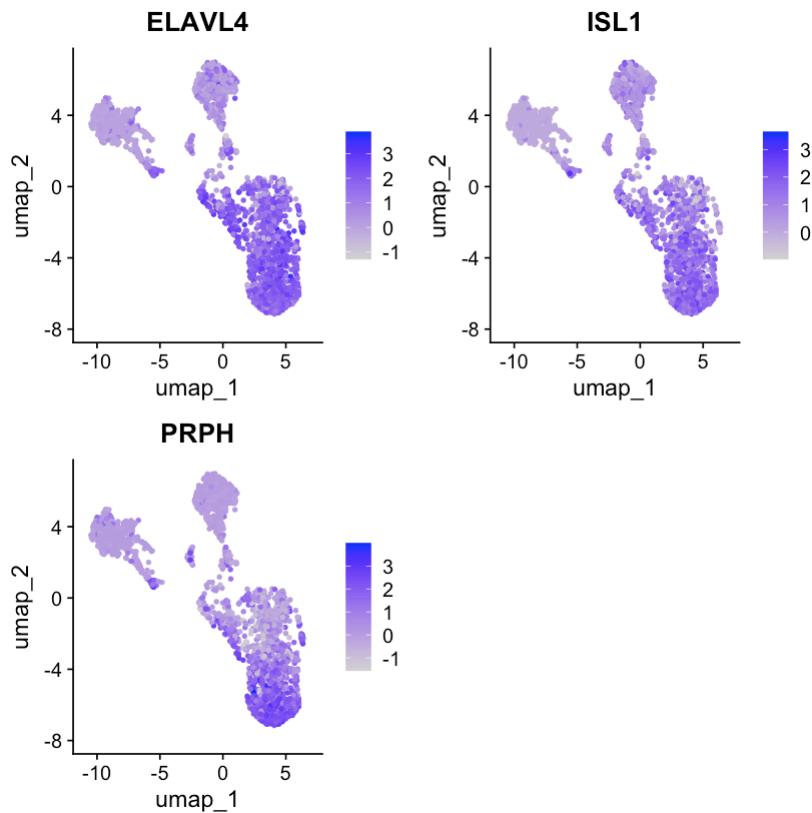
Markers for Prol. Sympathoblasts

```
FeaturePlot(seu_subset,markers[[1]])  
# cluster 3
```



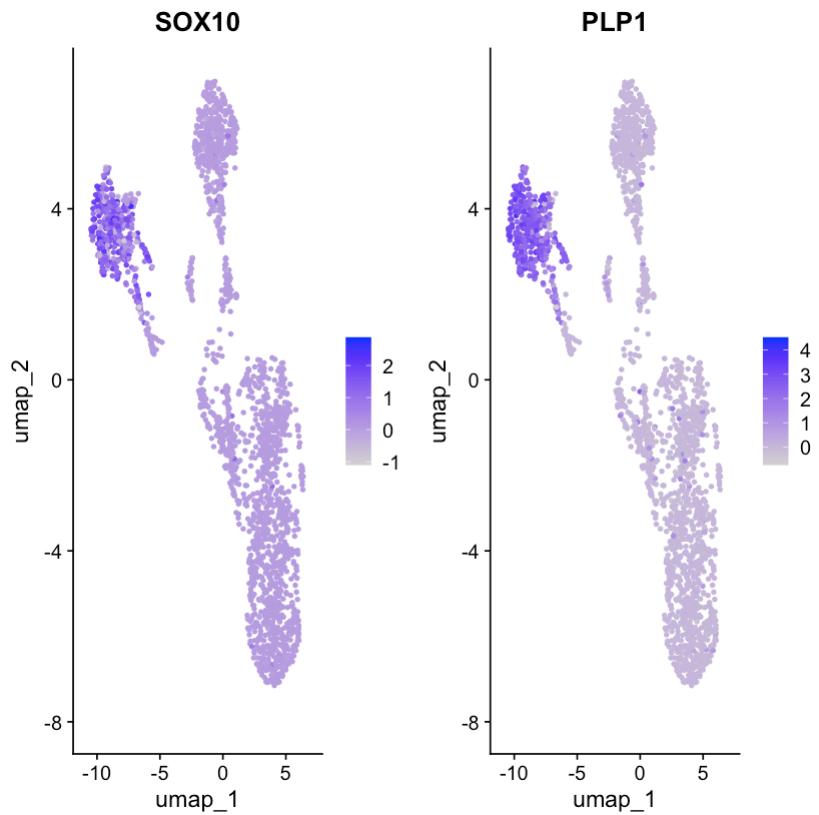
Markers for Sympathoblasts

```
FeaturePlot(seu_subset,markers[[2]])  
# cluster 0
```



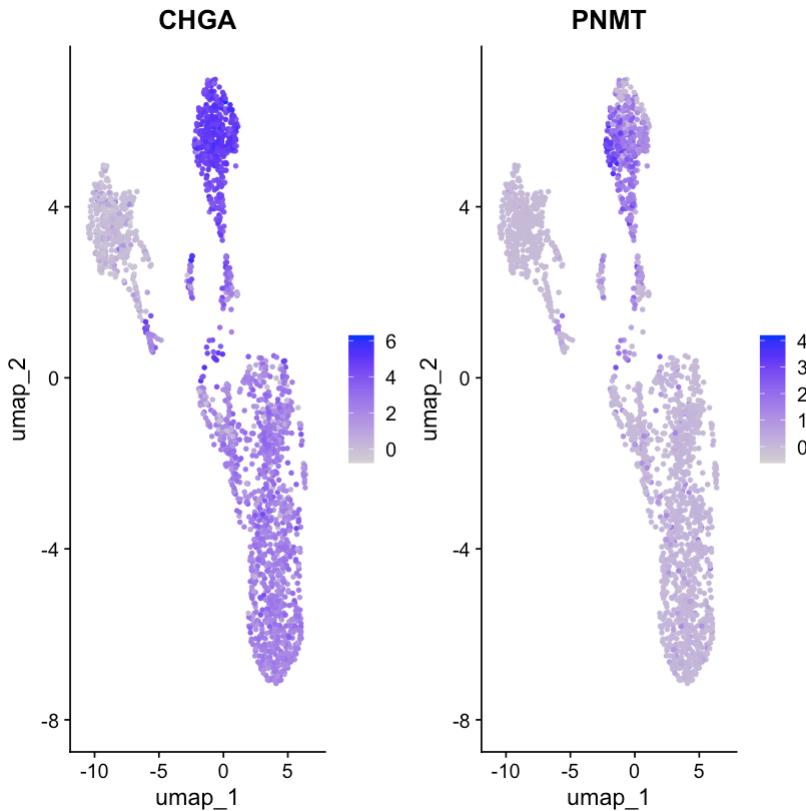
Markers for SCPs

```
FeaturePlot(seu_subset,markers[[3]], ncol=2)
# cluster 2
```



Chromaffin cells

```
FeaturePlot(seu_subset,markers[[4]], ncol=2)
# cluster 1
```



```
cluster <- list(
  "3" = "Prolifertating sympathoblasts", # = "MKI67",
  "0" = "Sympathoblasts", # = c("ELAVL4", "ISL1", "PRPH"),
  "2" = "SCPs", # = c("SOX10", "PLP1"),
  "1" = "Chromaffin cells" # = c("CHGA", "PNMT")
)
```

```
cluster <- unlist(cluster)
seu_subset <- RenameIdents(seu_subset, cluster)
seu_subset$celltype <- Idents(seu_subset)
```

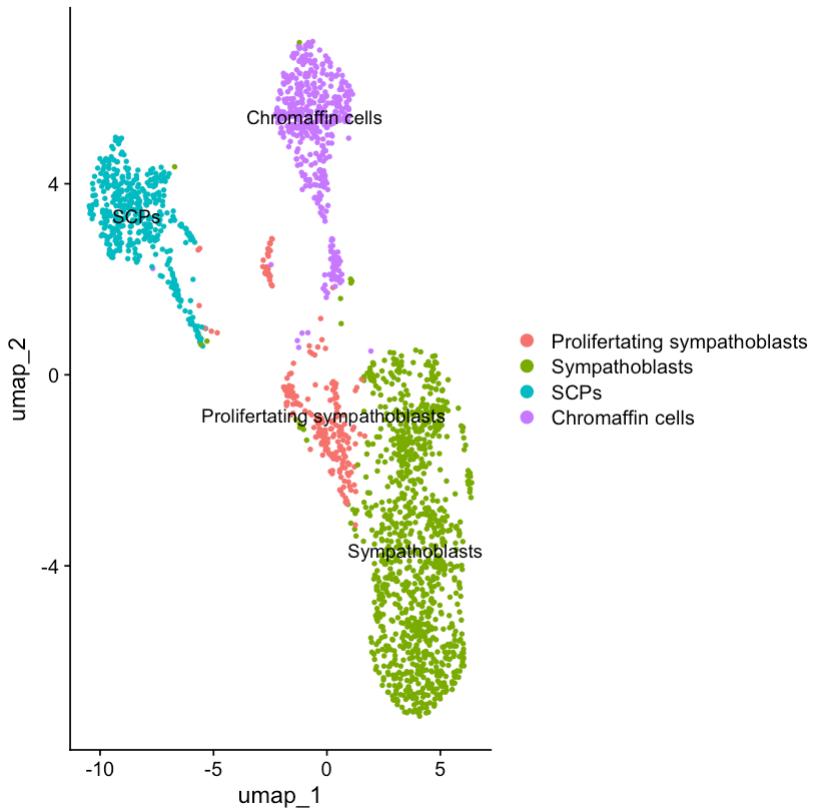
```
unique(seu_subset$celltype)
```

1. Chromaffin cells
2. Sympathoblasts
3. Prolifertating sympathoblasts
4. SCPs

Levels: 1. ‘Prolifertating sympathoblasts’ 2. ‘Sympathoblasts’ 3. ‘SCPs’ 4. ‘Chromaffin cells’

UMAP with Annotations

```
umap_ann_subset = DimPlot(seu_subset, label=TRUE)
umap_ann_subset
```



```
ggsave("plots/AMC_subset_UMAP_annotated.pdf",umap_ann_subset, width = 10, height = 8)
```

```
saveRDS(seu_subset,"data/processed/AMC_subset_annotated.rds")
```

Trajectory analysis

```
import scanpy as sc
import celestial as cl
from lets_plot import *

LetsPlot.setup_html()
```

Unable to display output for mime type(s): text/html

```
adata = sc.read("data/processed/AMC_subset_annotated.h5ad")
```

/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/anndata/compat/__init__.py:371: FutureWarning:

This is where adjacency matrices should go now.

```
warn(
```

```
adata.obs["celltype"] = adata.obs["celltype"].astype(str).astype("category")
```

Note seurat object to h5ad did not retain celltype info

```
adata.obs["celltype"].unique()
```

```
['3', '1', '0', '2']
Categories (4, object): ['0', '1', '2', '3']
```

Plot with Celestial

```
cl.umap(adata, "celltype", size=2, axis_type="arrow", legend_ondata="True")
```

```
<lets_plot.plot.core.PlotSpec at 0x35c640b00>
```

Assingn cell types

```

cluster = {
    "0": "Proliferating sympathoblasts", # = "MKI67",
    "1": "Sympathoblasts", # = c("ELAVL4", "ISL1", "PRPH"),
    "2": "SCPs", # = c("SOX10", "PLP1"),
    "3": "Chromaffin cells", # = c("CHGA", "PNMT")
}
adata.obs["cell_type"] = adata.obs["celltype"].map(cluster)
adata.obs["cell_type"].unique()

```

```

['Chromaffin cells', 'Sympathoblasts', 'Proliferating sympathoblasts', 'SCPs']
Categories (4, object): ['Proliferating sympathoblasts', 'Sympathoblasts', 'SCPs', 'Chromaffin cells']

```

```

cl.umaps(
    adata,
    ["celltype", "cell_type"],
    size=2,
    axis_type="arrow",
    legend_ondata="True",
    ncol=2,
)

```

```
<lets_plot.plot.subplots.SupPlotsSpec at 0x3517f1a90>
```

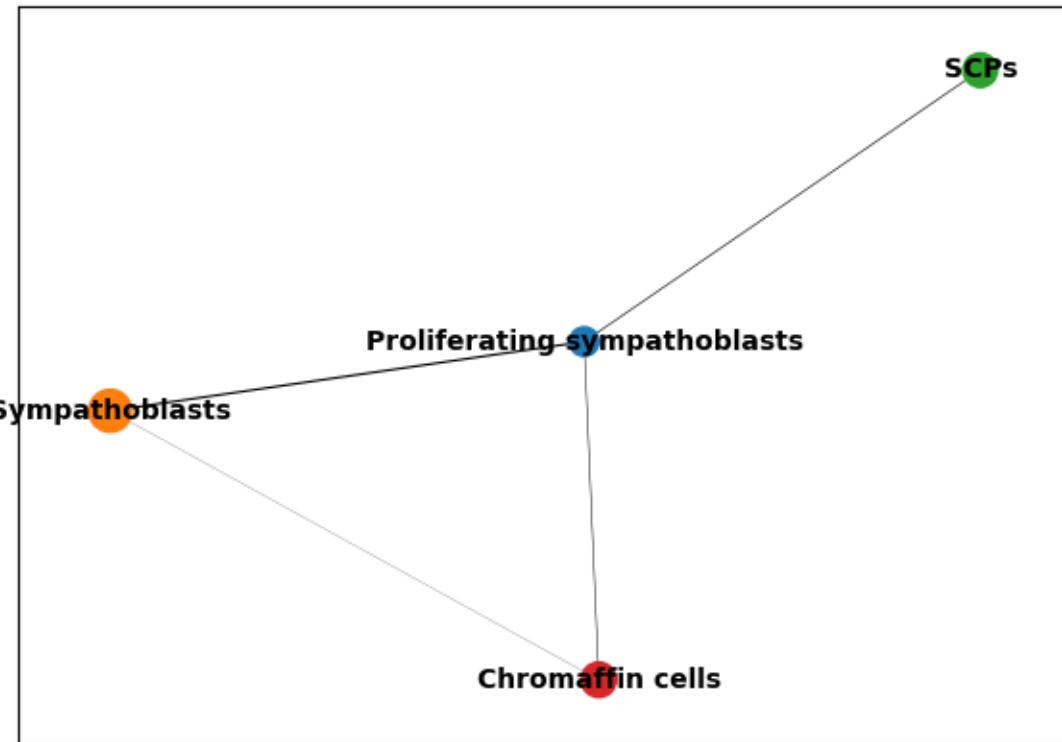
Overall Dataset

PAGA

```

sc.tl.paga(adata, groups="cell_type")
sc.pl.paga(adata, color="cell_type")

```



Re calculate Neighbors and UMAP

```
sc.pp.neighbors(adata, n_neighbors=15, n_pcs=40)
```

```
sc.tl.umap(adata, init_pos="paga")
umap_all = cl.umap(
    adata, key="cell_type", legend_ondata=True, axis_type="arrow", ondata_size=8, size=3
) + ggtitle("All AMC cells")
umap_all
```

```
<lets_plot.plot.core.PlotSpec at 0x30dfff140>
```

```
print(adata.obs["cell_type"].unique())
```

```
['Chromaffin cells', 'Sympathoblasts', 'Proliferating sympathoblasts', 'SCPs']
Categories (4, object): ['Proliferating sympathoblasts', 'Sympathoblasts', 'SCPs', 'Chromaffin cells']
```

Run DPT

```
root_cell = adata.obs[adata.obs["cell_type"] == "SCPs"].index[0]
adata.uns["iroot"] = adata.obs.index.get_loc(root_cell)
sc.tl.dpt(adata)
```

WARNING: Trying to run `tl.dpt` without prior call of `tl.diffmap`. Falling back to `tl.diffmap` with

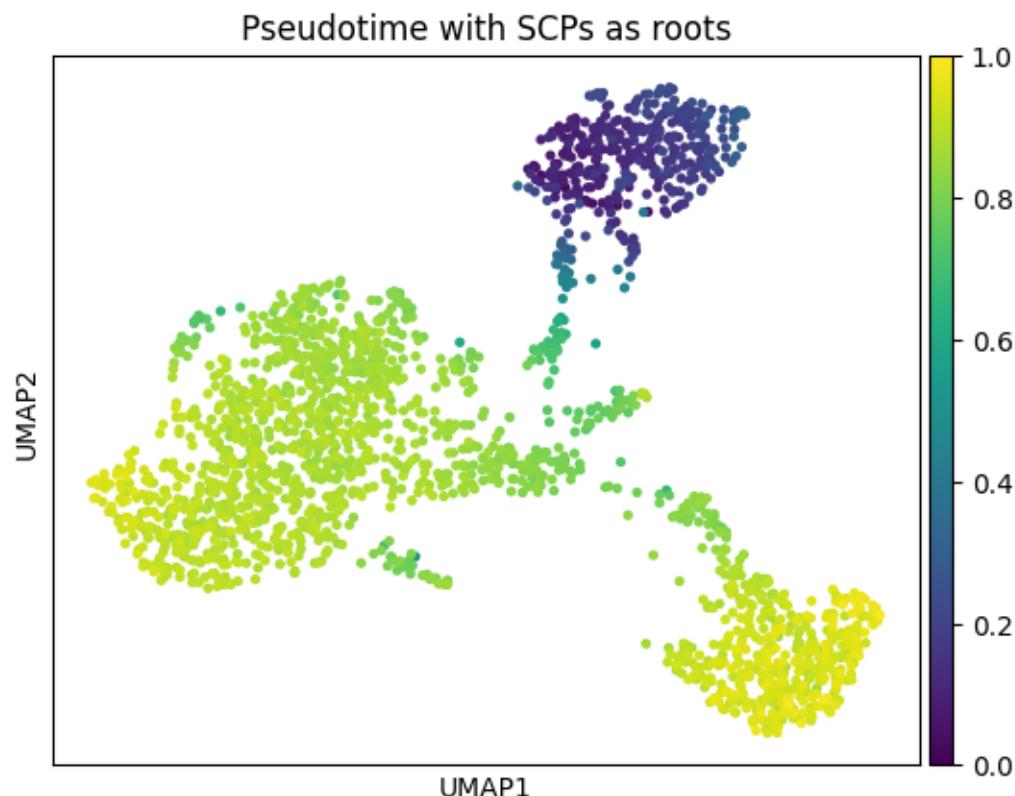
```

umap_all_dpt =(
    cl.umap(
        adata,
        key="dpt_pseudotime",
        size=3,
        axis_type="arrow",
        add_tooltips=["cell_type"],
    )
    + scale_color_viridis()
    + ggtitle("All AMC cells")
    + ggszie(600, 500)
)
umap_all_dpt

```

<lets_plot.plot.core.PlotSpec at 0x30dff2f0>

```
sc.pl.umap(adata, color="dpt_pseudotime", title="Pseudotime with SCPs as roots")
```



```

(
    cl.umap(
        adata,
        key="dpt_pseudotime",
        size=2,
        axis_type="arrow",
        add_tooltips=["cell_type"],
    )
)
```

```
+ scale_color_viridis()  
)
```

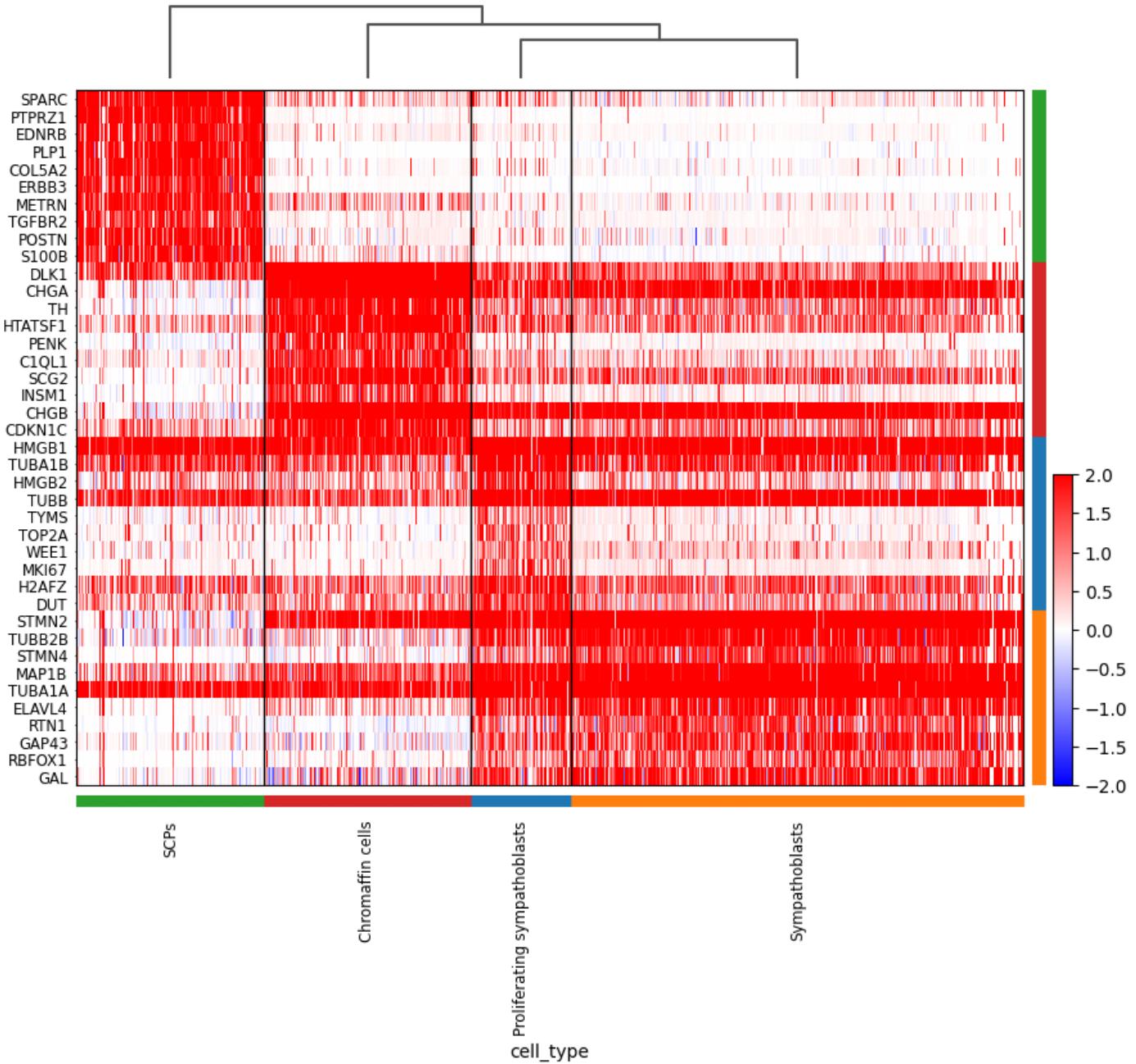
```
<lets_plot.plot.core.PlotSpec at 0x35ca15d90>
```

Heatmap

```
sc.tl.rank_genes_groups(adata, 'cell_type', method='t-test')
```

```
sc.pl.rank_genes_groups_heatmap(  
    adata,  
    n_genes=10,    # show top 10 per group  
    groupby='cell_type',  
    show_gene_labels=True,  
    cmap='bwr',  
    swap_axes=True,  
    vmin=-2, vmax=2    #  
)
```

WARNING: dendrogram data not found (using key=dendrogram_cell_type). Running `sc.tl.dendrogram` with



Subsets

1. SCPs to Sympathoblasts
2. SCPs to Chromaffin Cells
3. Chromaffin Cells to Sympathoblasts

subset1

```
subset1 = ["Proliferating sympathoblasts", "Sympathoblasts", "SCPs"]
adata1 = adata[adata.obs["cell_type"].isin(subset1)]
```

subset2

```
subset2 = ["Chromaffin cells", "SCPs", "Proliferating sympathoblasts"]
adata2 = adata[adata.obs["cell_type"].isin(subset2)]
```

subset3

```
subset3 = ["Chromaffin cells", "Proliferating sympathoblasts", "Sympathoblasts"]
adata3 = adata[adata.obs["cell_type"].isin(subset3)]
```

SCPs to sympathoblasts

```
adata1.obs["cell_type"].unique()
```

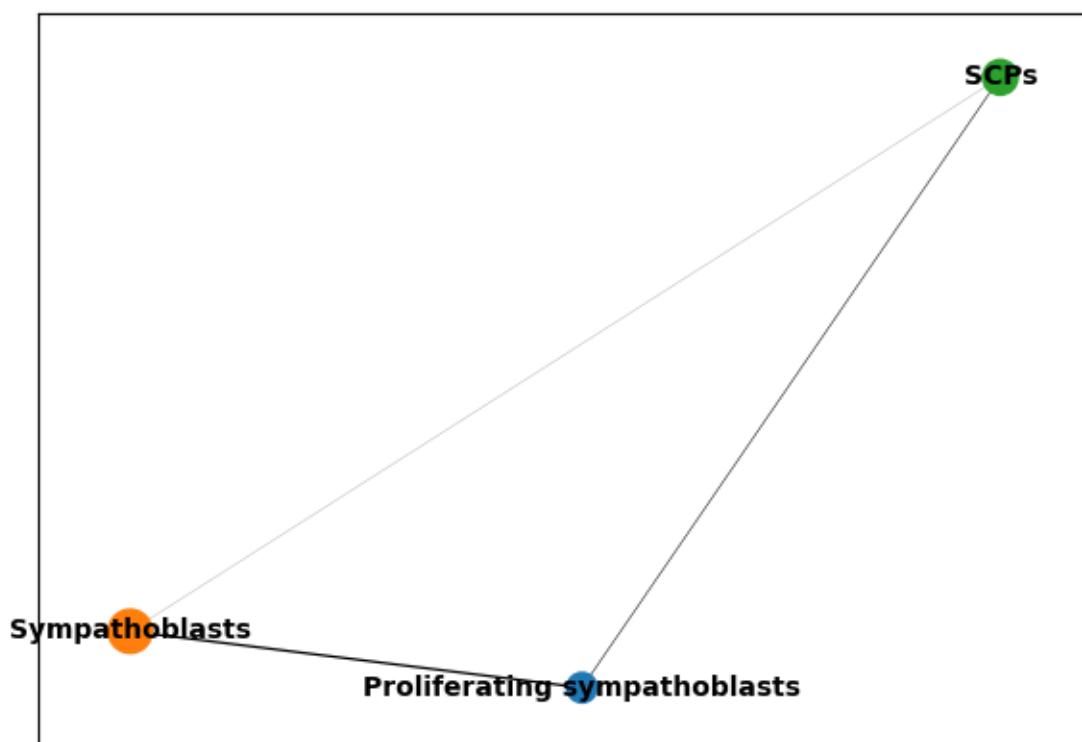
```
['Sympathoblasts', 'Proliferating sympathoblasts', 'SCPs']
Categories (3, object): ['Proliferating sympathoblasts', 'Sympathoblasts', 'SCPs']
```

PAGA workflow

```
sc.tl.paga(adata1, groups="cell_type")
```

```
/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/scanpy/tools/_paga.py:139: ImplicitModifi
adata.uns[groups + "_sizes"] = np.array(paga.ns)
```

```
sc.pl.paga(adata1, color='cell_type')
```



```
sc.pp.neighbors(adata1, n_neighbors=15, n_pcs=40)
```

```
sc.tl.umap(adata1, init_pos="paga")
```

```
umap1 =(
    cl.umap(
        adata1,
        key="cell_type",
        legend_ondata=True,
        axis_type="arrow",
        ondata_size=8,
        size=3,
    )
    + ggtitle("SCPs and sympathoblasts")
    + ggszie(600, 500)
)
umap1
```

```
<lets_plot.plot.core.PlotSpec at 0x30dfa6cc0>
```

Run DPT

```
root_cell = adata1.obs[adata1.obs["cell_type"] == "SCPs"].index[0]
adata1.uns["iroot"] = adata1.obs.index.get_loc(root_cell)
sc.tl.dpt(adata1)
```

```
umap1_dpt =(
    cl.umap(
        adata1,
        key="dpt_pseudotime",
        size=3,
        axis_type="arrow",
        add_tooltips=["cell_type"],
    )
    + scale_color_viridis()
    + ggtitle("SCPs and sympathoblasts ")
    + ggszie(600, 500)
)
umap1_dpt
```

```
<lets_plot.plot.core.PlotSpec at 0x30dff0b0>
```

Find changing genes along the trajectory

```
sc.tl.rank_genes_groups(adata1, 'cell_type', method='t-test')
```

```
top_genes = adata1.uns['rank_genes_groups']['names']
top_genes[:10]
```

```

rec.array([('HMGB2', 'STMN2', 'SPARC'), ('HMGN2', 'MIAT', 'PTPRZ1'),
          ('TUBA1B', 'CD24', 'EDNRB'), ('HMGB1', 'CHGB', 'PLP1'),
          ('TYMS', 'DBH', 'COL5A2'), ('DUT', 'HAND2-AS1', 'METRN'),
          ('TOP2A', 'TUBB2B', 'ERBB3'), ('MKI67', 'RGS5', 'TGFBR2'),
          ('HELLS', 'GATA2', 'S100B'), ('CENPF', 'PCSK1N', 'POSTN')],  

          dtype=[('Proliferating sympathoblasts', 'O'), ('Sympathoblasts', 'O'), ('SCPs', 'O')])

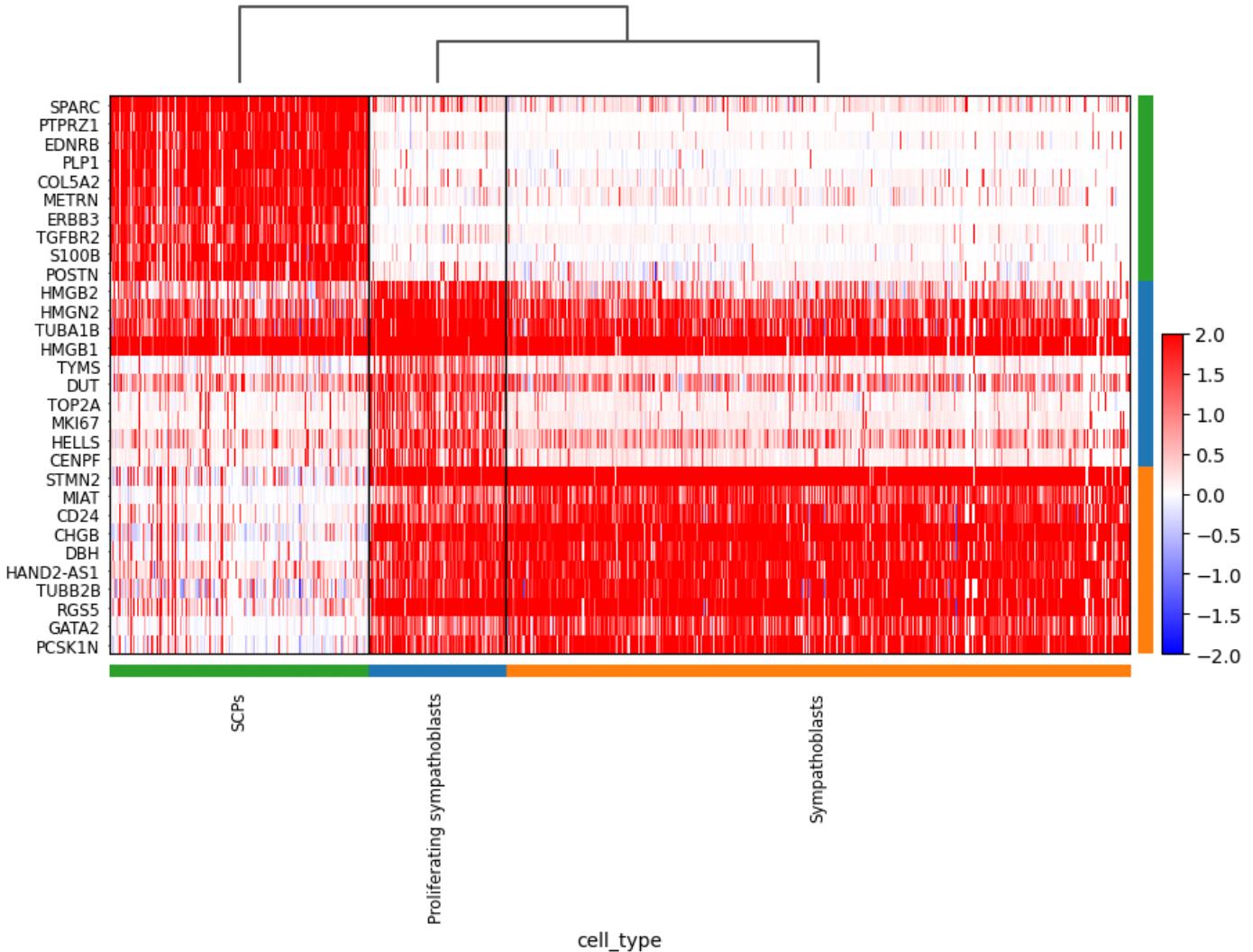
```

Heatmap of genes chaning along the trajectory

```
sc.tl.dendrogram(adata1, groupby="cell_type")
```

```
heatmap1 = sc.pl.rank_genes_groups_heatmap(  
    adata1,  
    n_genes=10,  
    groupby='cell_type',  
    show_gene_labels=True,  
    cmap='bwr',  
    swap_axes=True,  
    save='_scps_and_sympathoblasts.pdf',  
    vmin=-2, vmax=2)  
  
heatmap1
```

WARNING: saving figure to file figures/heatmap_scps_and_sympathoblasts.pdf



SCPs to Chromaffin Cells

```
adata2.obs["cell_type"].unique()
```

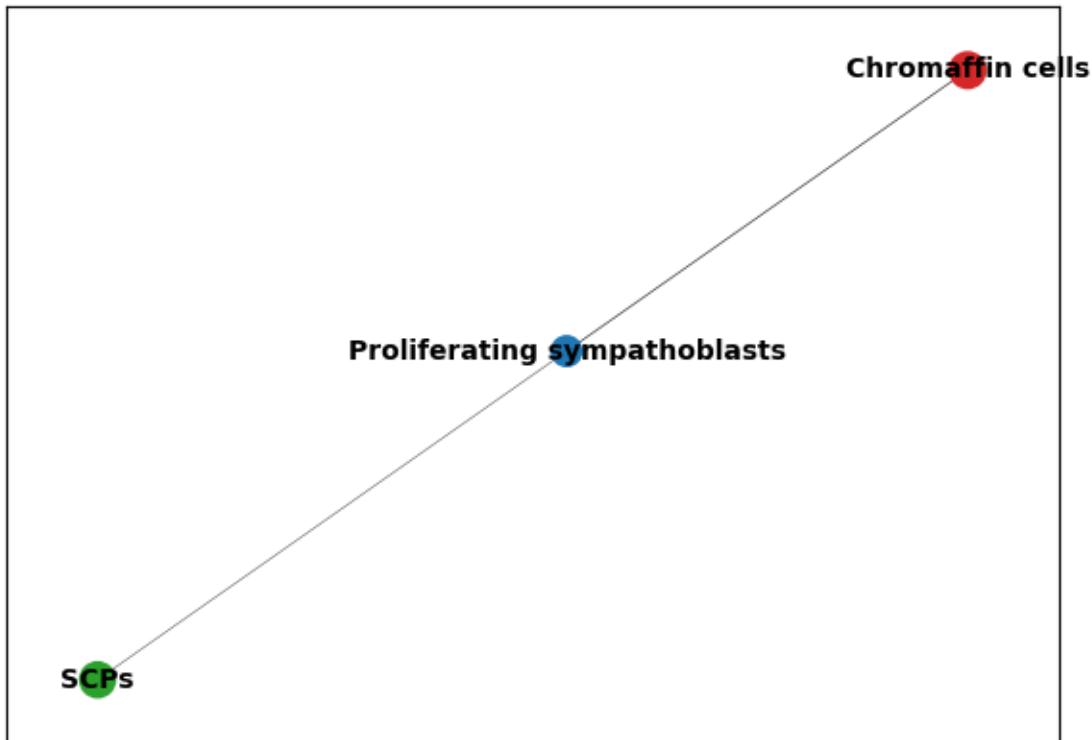
```
['Chromaffin cells', 'Proliferating sympathoblasts', 'SCPs']
Categories (3, object): ['Proliferating sympathoblasts', 'SCPs', 'Chromaffin cells']
```

PAGA

```
sc.tl.paga(adata2, groups="cell_type")
```

```
/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/scanpy/tools/_paga.py:139: ImplicitModifi
adata.uns[groups + "_sizes"] = np.array(paga.ns)
```

```
sc.pl.paga(adata2, color='cell_type')
```



```
sc.pp.neighbors(adata2, n_neighbors=15, n_pcs=40)
```

```
sc.tl.umap(adata2, init_pos="paga")
```

```
umap2 =(
    cl.umap(
        adata2,
        key="cell_type",
        legend_ondata=True,
        axis_type="arrow",
        ondata_size=8,
        size=3,
    )
    + ggtitle("SCPs and Chromaffin Cells")
    + ggszie(600, 500)
)
umap2
```

```
<lets_plot.plot.core.PlotSpec at 0x366175be0>
```

Run DPT

```
root_cell = adata2.obs[adata2.obs["cell_type"] == "SCPs"].index[0]
adata2.uns["iroot"] = adata2.obs.index.get_loc(root_cell)
sc.tl.dpt(adata2)
```

```

umap2_dpt =(
    cl.umap(
        adata2,
        key="dpt_pseudotime",
        size=3,
        axis_type="arrow",
        add_tooltips=["cell_type"],
    )
    + scale_color_viridis()
    + ggtitle("SCPs and Chromaffin Cells")
    + ggszie(600, 500)
)
umap2_dpt

```

<lets_plot.plot.core.PlotSpec at 0x334f2cc20>

Find changing genes along the trajectory

```
sc.tl.rank_genes_groups(adata2, 'cell_type', method='t-test')
```

```
/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/scanpy/tools/_rank_genes_groups.py:484:
    self.stats[group_name, "logfoldchanges"] = np.log2(
/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/scanpy/tools/_rank_genes_groups.py:484:
    self.stats[group_name, "logfoldchanges"] = np.log2(
/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/scanpy/tools/_rank_genes_groups.py:484:
    self.stats[group_name, "logfoldchanges"] = np.log2(
```

```
top_genes = adata2.uns['rank_genes_groups']['names']
top_genes[:10]
```

```
rec.array([('TUBB', 'PTPRZ1', 'CHGA'), ('MAP1B', 'PLP1', 'DLK1'),
           ('STMN1', 'SPARC', 'TH'), ('TUBB2B', 'COL5A2', 'CHGB'),
           ('STMN2', 'EDNRB', 'SCG2'), ('HMGB1', 'ANXA2', 'HTATSF1'),
           ('ELAVL4', 'ERBB3', 'RGS4'), ('BASP1', 'TGFB2', 'RAMP1'),
           ('KIF21A', 'MPZ', 'SLC18A1'), ('TUBA1B', 'OLFML2A', 'PENK')],
          dtype=[('Proliferating sympathoblasts', '0'), ('SCPs', '0'), ('Chromaffin cells', '0')])
```

Heatmap of genes chaning along the trajectory

```
sc.tl.dendrogram(adata2, groupby="cell_type")
```

```
heatmap2 = sc.pl.rank_genes_groups_heatmap(
    adata2,
    n_genes=10,
    groupby='cell_type',
    show_gene_labels=True,
    cmap='bwr',
```

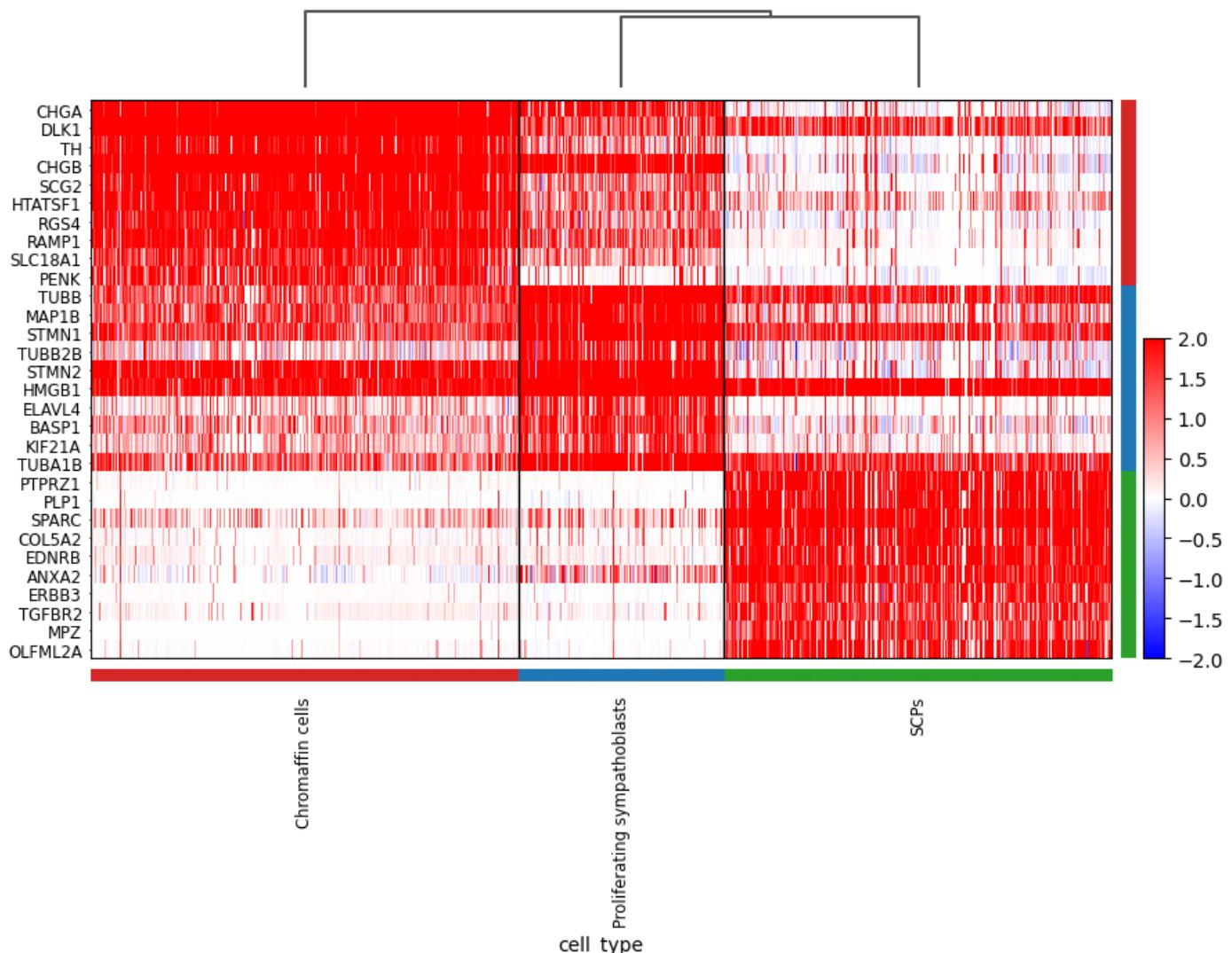
```

    swap_axes=True,
    save='_scps_and_chromaffin.pdf',
    vmin=-2, vmax=2)

```

heatmap2

WARNING: saving figure to file figures/heatmap_scps_and_chromaffin.pdf



Chromaffin Cells to Sympathoblasts

```
adata3.obs["cell_type"].unique()
```

```

['Chromaffin cells', 'Sympathoblasts', 'Proliferating sympathoblasts']
Categories (3, object): ['Proliferating sympathoblasts', 'Sympathoblasts', 'Chromaffin cells']

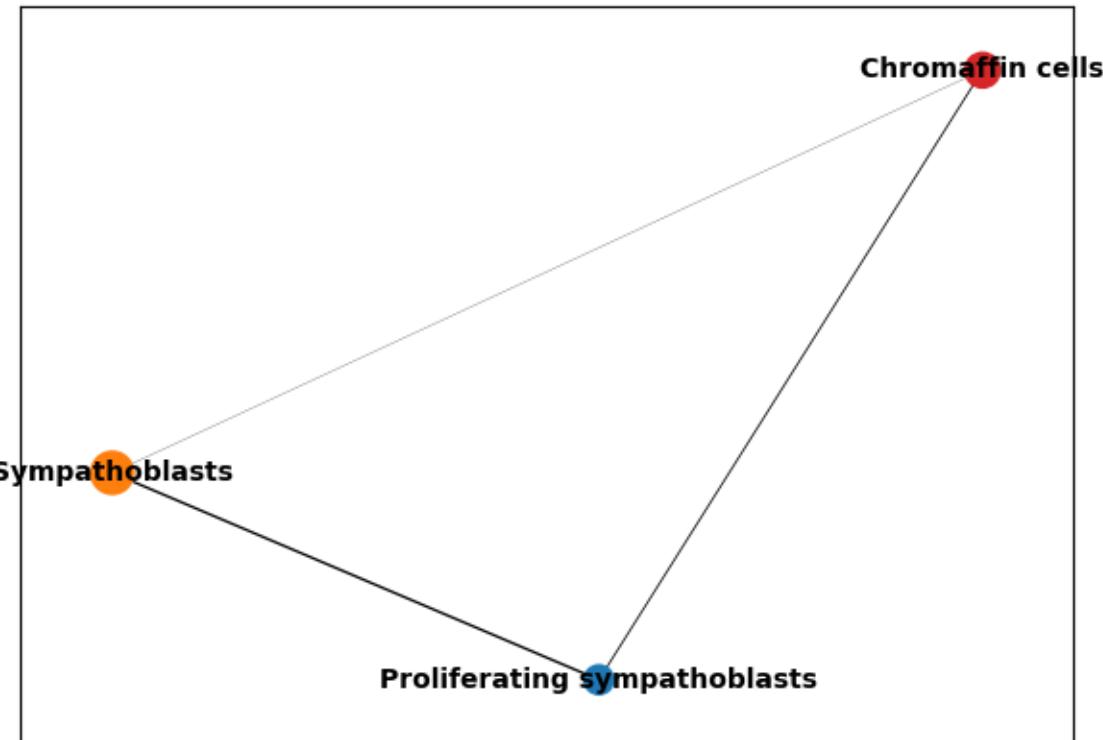
```

PAGA

```
sc.tl.paga(adata3, groups="cell_type")
```

```
/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/scanpy/tools/_paga.py:139: ImplicitModi  
adata.uns[groups + "_sizes"] = np.array(paga.ns)
```

```
sc.pl.paga(adata3, color="cell_type")
```



```
sc.pp.neighbors(adata3, n_neighbors=15, n_pcs=40)
```

```
sc.tl.umap(adata3, init_pos="paga")
```

```
umap3 = (  
    cl.umap(  
        adata3,  
        key="cell_type",  
        legend_ondata=True,  
        axis_type="arrow",  
        ondata_size=8,  
        size=3,  
    )  
    + ggtitle("Chromaffin Cells and Sympathoblasts")  
    + ggszie(600, 500)  
)
```

Run DPT

```
root_cell = adata3.obs[adata3.obs["cell_type"] == "Sympathoblasts"].index[0]
adata3.uns["iroot"] = adata3.obs.index.get_loc(root_cell)
sc.tl.dpt(adata3)
```

```
umap3_dpt = (
    cl.umap(
        adata3,
        key="dpt_pseudotime",
        size=3,
        axis_type="arrow",
        add_tooltips=["cell_type"],
    )
    + scale_color_viridis()
    + ggtitle("Chromaffin Cells and Sympathoblasts")
    + ggszie(600, 500)
)
umap3_dpt
```

<lets_plot.plot.core.PlotSpec at 0x335ef0200>

Find changing genes along the trajectory

```
sc.tl.rank_genes_groups(adata3, 'cell_type', method='t-test')
```

```
/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/scanpy/tools/_rank_genes_groups.py:484:
    self.stats[group_name, "logfoldchanges"] = np.log2(
/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/scanpy/tools/_rank_genes_groups.py:484:
    self.stats[group_name, "logfoldchanges"] = np.log2(
/Users/zaf4/dev/CCRItask/.venv/lib/python3.13/site-packages/scanpy/tools/_rank_genes_groups.py:484:
    self.stats[group_name, "logfoldchanges"] = np.log2(
```

```
top_genes = adata3.uns['rank_genes_groups']['names']
top_genes[:10]
```

```
rec.array([('HMGB1', 'SOX4', 'DLK1'), ('TUBA1B', 'STMN2', 'CHGA'),
           ('HMGB2', 'TUBB2B', 'PENK'), ('TOP2A', 'TUBA1A', 'C1QL1'),
           ('TYMS', 'STMN4', 'HTATSF1'), ('MKI67', 'RTN1', 'INSM1'),
           ('CDK1', 'MAP1B', 'TH'), ('CENPF', 'GAP43', 'SLC24A2'),
           ('H2AFZ', 'RBFOX1', 'CDKN1C'), ('TUBB', 'PHOX2B', 'ST18')],
          dtype=[('Proliferating sympathoblasts', '0'), ('Sympathoblasts', '0'), ('Chromaffin cells', '0')])
```

Heatmap of genes chaning along the trajectory

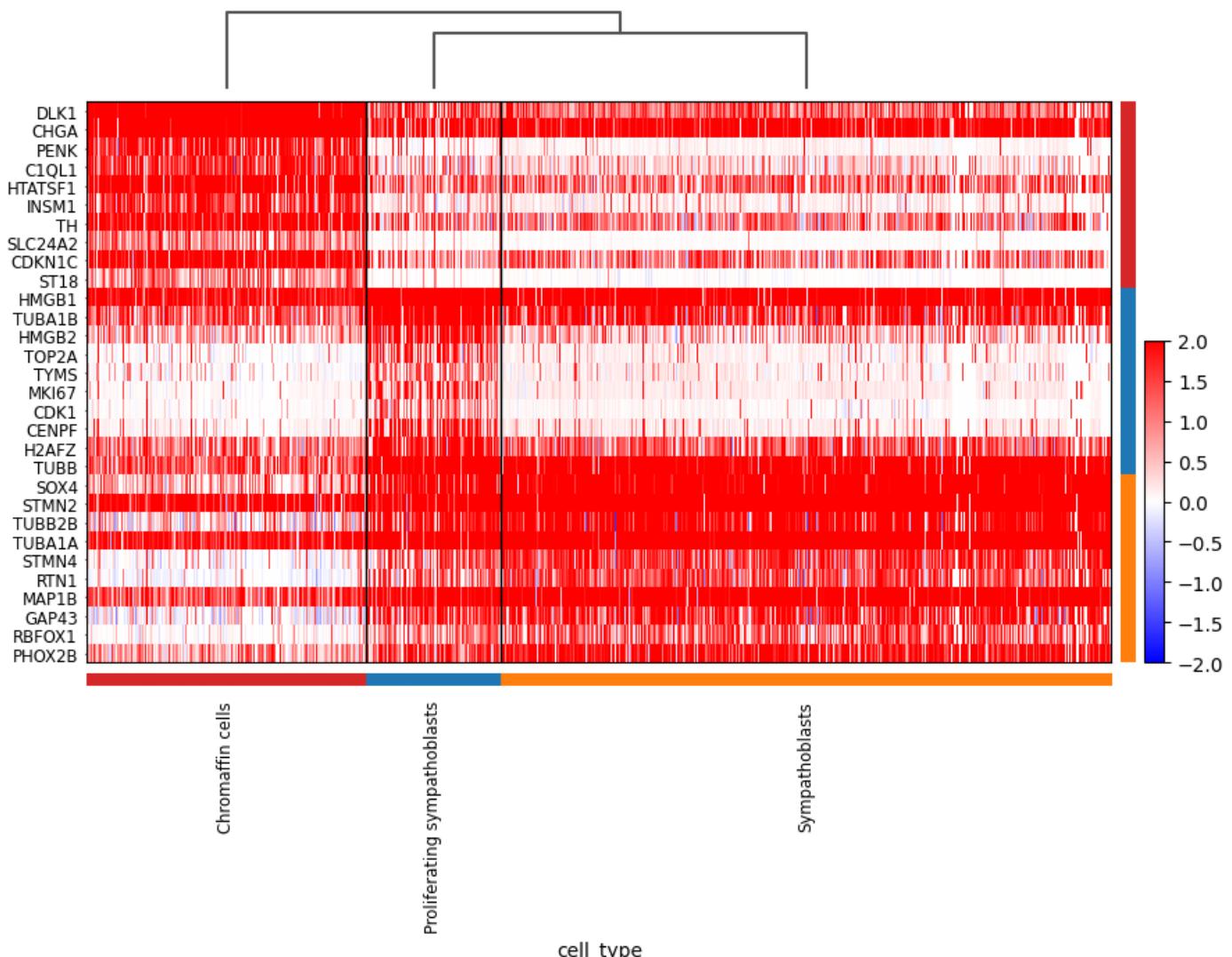
```
sc.tl.dendrogram(adata3, groupby="cell_type")
```

```

heatmap3 = sc.pl.rank_genes_groups_heatmap(
    adata3,
    n_genes=10,
    groupby='cell_type',
    show_gene_labels=True,
    cmap='bwr',
    swap_axes=True,
    save='_chromaffin_and_sympathoblasts.pdf',
    vmin=-2, vmax=2, return_fig=True)
heatmap3

```

WARNING: saving figure to file figures/heatmap_chromaffin_and_sympathoblasts.pdf



```

grid = gggrid([umap_all, umap_all_dpt, umap1, umap1_dpt, umap2, umap2_dpt, umap3, umap3_dpt], ncol=3)
grid

```

<lets_plot.plot.subplots.SupPlotsSpec at 0x3480a7000>

```
ggsave(grid, filename='plots/umap_pseudotime.svg', path='..')
```

```
'/Users/zaf4/dev/CCRItask/plots/umap_pseudotime.svg'
```