Single Cell RNA Sequence Analysis Pipeline Guide

Craig McDougall

October 2020

Introduction

This document details the stages of a bioinformatic pipeline, designed in R to analyse single cell RNA sequence data. The pipeline was constructed as part of an MSc project at the University of Edinburgh in Scotland.

The aim of the project was to investigate sexual dimorphism in the central stress response system; the Hypothalmic-pituitary-adrenal (HPA) axis. The pipeline was designed as a tool, to probe the transcriptome profiles of anterior pituitary cells. In the project it facilitated a study of sexual dimorphism in the gene expression of pituitary corticotroph cells, but could also be used on other cell types.

A similar method has been reported in a recent study of sexual dimorphism in pituitary cells. However, this study concentrated on gonadotroph and lactotroph pituitary cells [Fletcher et.al 2019]. An underlying limitation of this kind of analysis is that it relies on inferring protein levels by measuring transcription levels, it is not a direct measurement of the proteins present.

The pipeline files can be found on GitHub.

The Seurat R. package forms the foundation of this pipeline and is in particular based on the guided clustering tutorial, but updated to include the `SCTransform` normalisation function.

Example single cell RNA sequence data is available in public repositories. In the project the pipeline was designed for, the following data sets were examined:

- Chung et al.
- Mayran et al.
- Ho et al.
- Fletcher et al.

Details on the data acquisition of these datasets is described within the following publications:

- Chung et al.
- Mayran et al.
- Ho et al.
- Fletcher et al.

The pipeline is comprised of the following steps;

Step 1. Call Libraries & Set Directories

Before the main body of the pipeline is run, all of the required libraries are called;

```
library(dplyr)
library(Seurat)
library(patchwork)
library(sctransform)
library(ggplot2)
library(scales)
```

The working directories are then set:

```
#set the working directory to directory wish to read data from and save analysis results to
data.dir <- setwd(here::here())
#check working directory
getwd()
```

```
## [1] "/Users/craigmcdougall/Documents/Project 2/Data analysis/Project 2"
```

```
#list all the files in the data directory, to make sur this is where we expect to be
list.files(data.dir)
```

```
##  [1] "Cheung Data"                 "Cheung-Data-analysis.pdf"
##  [3] "Cheung-Data-analysis.Rmd"    "Ho Data"
##  [5] "HoFemale-Data-Analysis.pdf" "HoFemale-Data-Analysis.Rmd"
##  [7] "HoMale-Data-Anlysis.pdf"     "HoMale-Data-Anlysis.Rmd"
##  [9] "Mayran Data"                 "Mayran-Data-Analysis.pdf"
## [11] "Mayran-Data-Analysis.Rmd"    "Pipeline Guide.Rmd"
## [13] "Pipeline-Guide.pdf"          "Pipeline-Guide.Rmd"
## [15] "README.md"                   "scRNAseq_pipeline.R"
## [17] "scRNAseq_pipeline.Rproj"
```

```
#have now seen that the files are there and ok to proceed
```

The raw count and gene data is read in from the 10X sequence data files downloaded from the public repositories. Seurat expects `barcodes.tsv`, `matrix.tsv` and `matrix.mtx` files. These are then used to initialise a Seurat object for the transcriptome analysis.

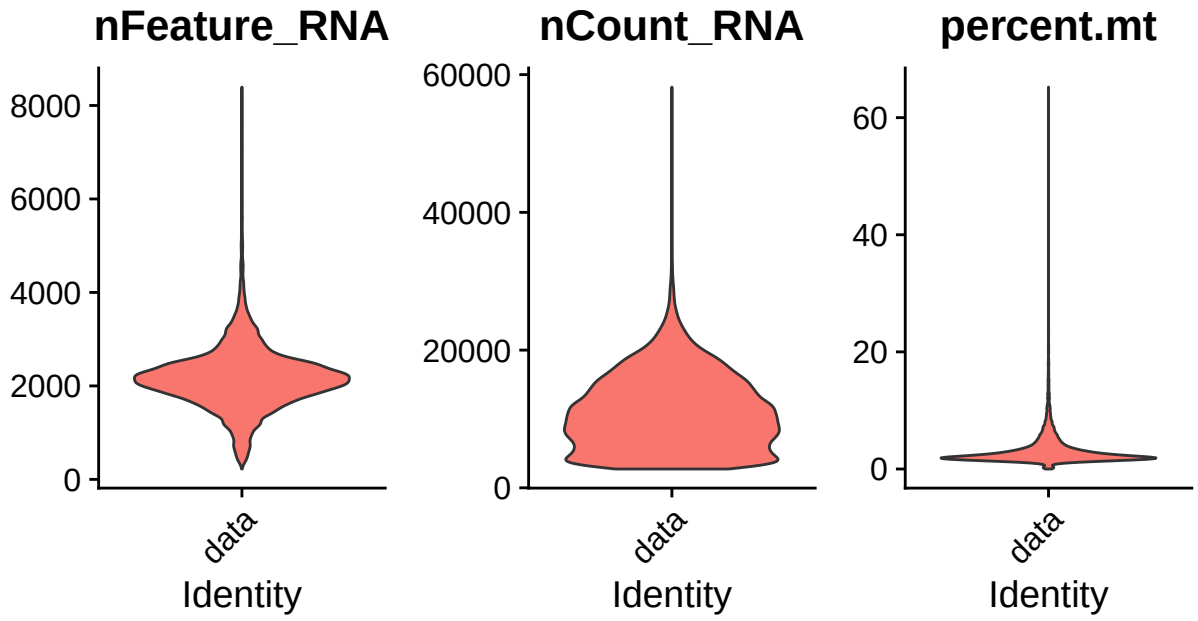
Please note the folder name will be relative to your own path.

```
#load raw data using Read10x, this will be relative to your path, name desired folder e.g.  "Cheung Dat
raw.data <- Read10X("Cheung Data")

#initialise a Seurat object with the raw (not normalised) data. This is a count matrix
data <- CreateSeuratObject(counts=raw.data, project = "data", min.cells = 3, min.features = 200)
```

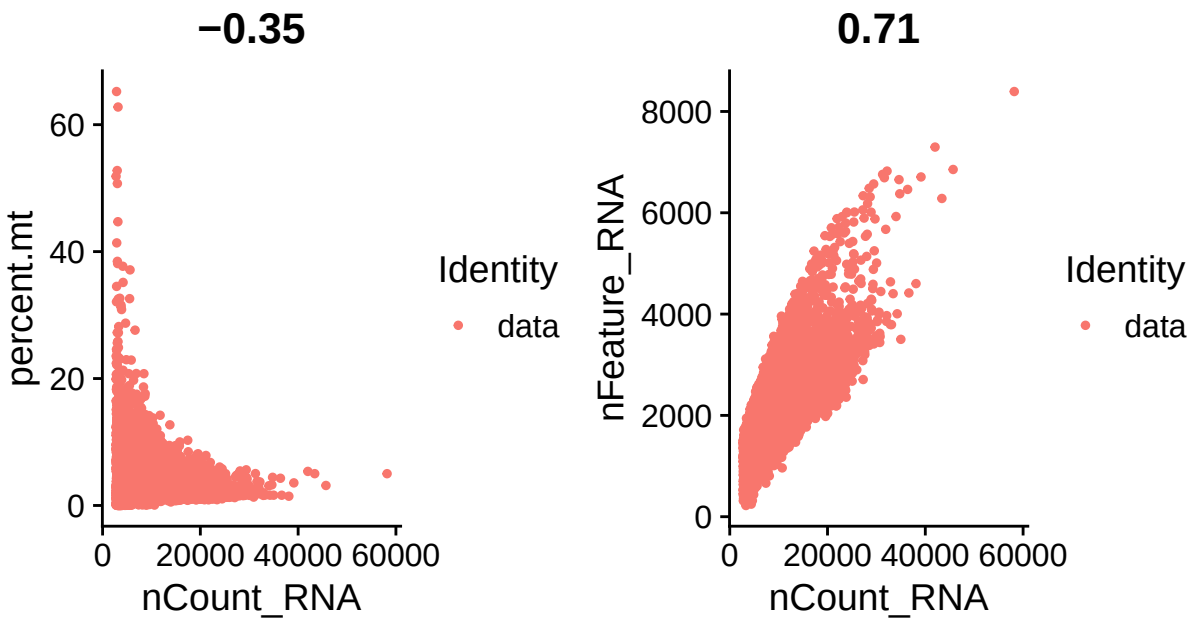
Step 2. Quality control plots

Before processing the data, some quality control plots are made to assess the quality of the data and explore confounding factors such as assessing the number of reads that map to mitochondrial genome.

```
#Seurat recommends percent.MT label for mitochondria data, however;
#Cheung, Ho & Mayran et.al. use "percent.mt"
data <- PercentageFeatureSet(data, pattern = "^mt-", col.name = "percent.mt")
#Fletcher uses "percent.Mt"
#data <- PercentageFeatureSet(data, pattern = "^mt-", col.name = "percent.MT")
#select appropriate method and comment out other option

#use QC metrics to filter the cells
#visualise the QC metrics - use a violin plot
VlnPlot(data, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3, pt.size=0)
```




```
#make the plots
plot1 <- FeatureScatter(data, feature1 = "nCount_RNA", feature2 = "percent.mt")
plot2 <- FeatureScatter(data, feature1 = "nCount_RNA", feature2 = "nFeature_RNA")
#then plot
plot1 + plot2
```



The QC plots to inform which cells should be excluded on quality basis (low quality, double counts). This is achieved using the `subset` function. For example to exclude cells with less than 200 counts and more than 2500 and greater than 5% mitochondrial DNA confounding factor;

```
data <- subset(data, subset = nFeature_RNA > 200 & nFeature_RNA <2500 & percent.mt < 5)
```

Step 3. Transform the data

With the data visualised and trimmed, it must be;

- Normalised to ensure all genes carry equal weight going forward and prevent heavily expressing genes from being over represented.
- Regressed to remove confounding factors such as areas mapped to the mitochondrial genome.
- Assessed for variable genes.

The SCTransform function performs all of these actions and greatly simplifies the code, negating the use of four separate functions.

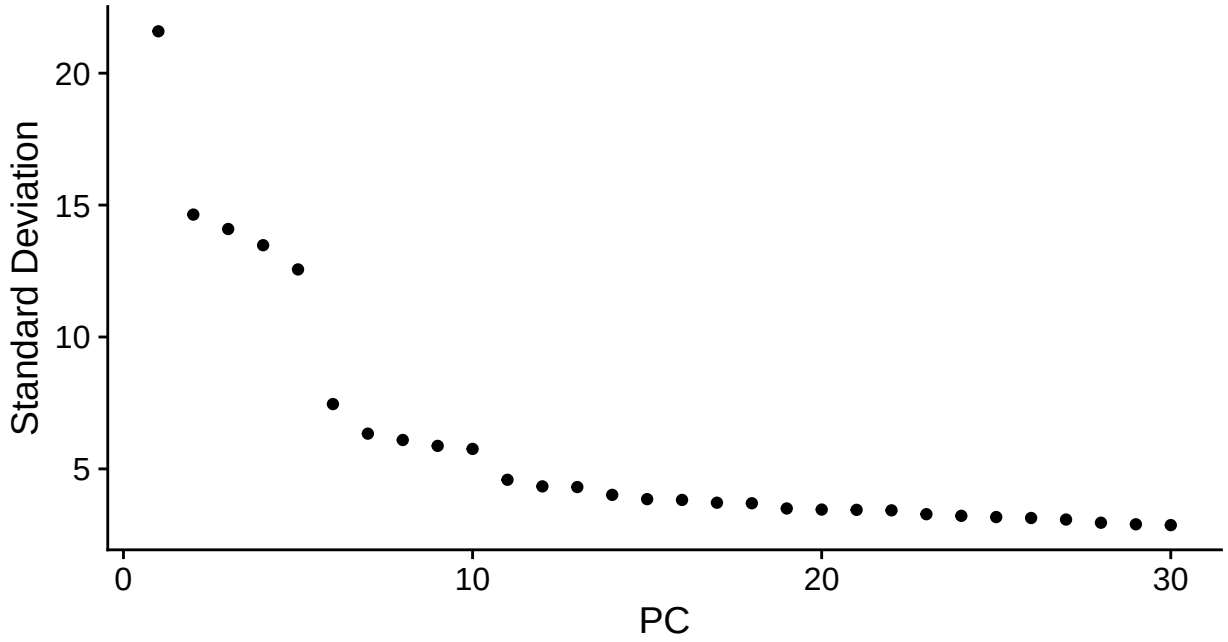
Step 4. Dimensional reduction

The first step in dimensional reduction is to conduct principal component analysis (PCA) on the scaled data, using the variable features as the input. PCA transforms the from the table into new features known as principal components, which capture the information of the dataset in a new way.

```
data <- RunPCA(data,verbose= FALSE)
```

An elbow plot can be used to assess which principal components contain the majority of the information and identify which may be removed from the dataset;

```
#Do we need to keep all PCs, or is majority of data captured by a certain point?
ElbowPlot(data, ndims=30)
```



Uniform Manifold Approximation and Projection (UMAP) is an algorithm for dimensional reduction and is applied on the principal components established in the first step.

```
#UMAP dimensional reduction of the identified PCs
data <- RunUMAP(data, dims=1:30, umap.method = "umap-learn", metric = "correlation", verbose=FALSE)
```

Step 5. Clustering

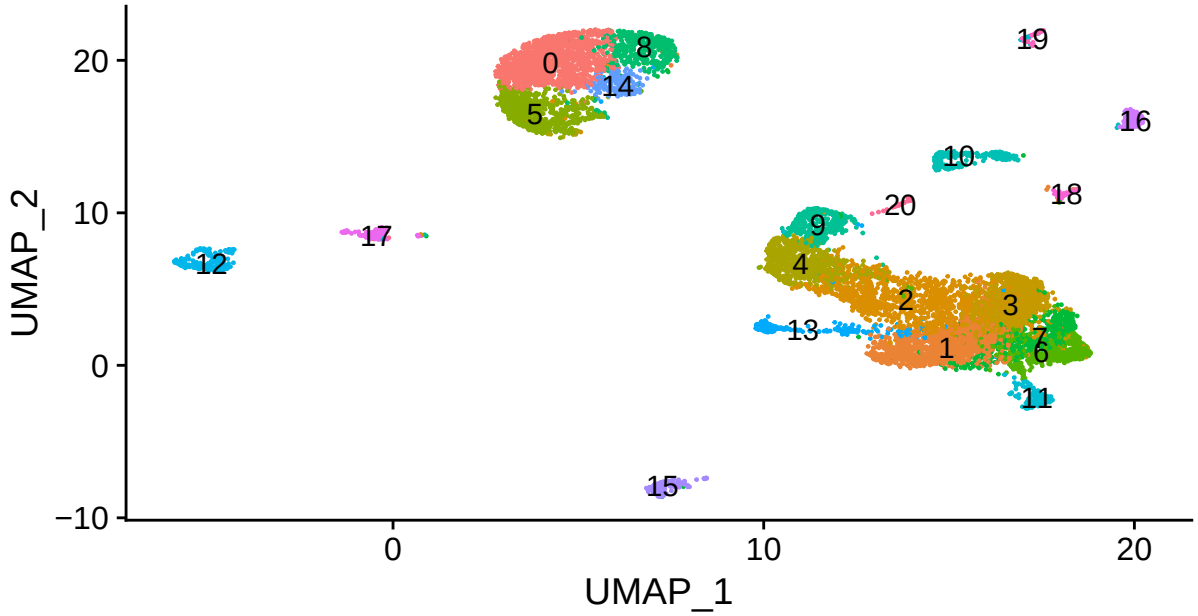
The first stage in clustering the cells is to establish a shared nearest neighbour (SNN) information for the data by calculating the overlap between each cell based on its k.parameters using a Seurat function.

The clusters cn then be identified based on the SNN using a clustering algorithm.

```
#Find nearest neighbours
data <- FindNeighbors(data, dims=1:30, verbose=FALSE)

#CLuster based on neighbour distances
data <- FindClusters(data, verbose=FALSE)
```

Once the clusters have been established, they can be projected into two lower dimensional space using a dimensional reduction plot for visualisation.

```
#plot the clusters
DimPlot(data, label=TRUE) + NoLegend()
```



Step 6. Cell Type assessment

The main question now, is how well do the clusters approximate cell type? It is difficult to make absolute claims when attributing cell types to empirical cluster data based purely on an *in silico* analysis. However we may approximate the cell type of each cluster based on known canonical expression markers [Fletcher et.al 2019] [Chung et.al 2018].

In the corticotroph study, hormone secreting pituitary cell types were identified based on the following canonical gene expression markers:

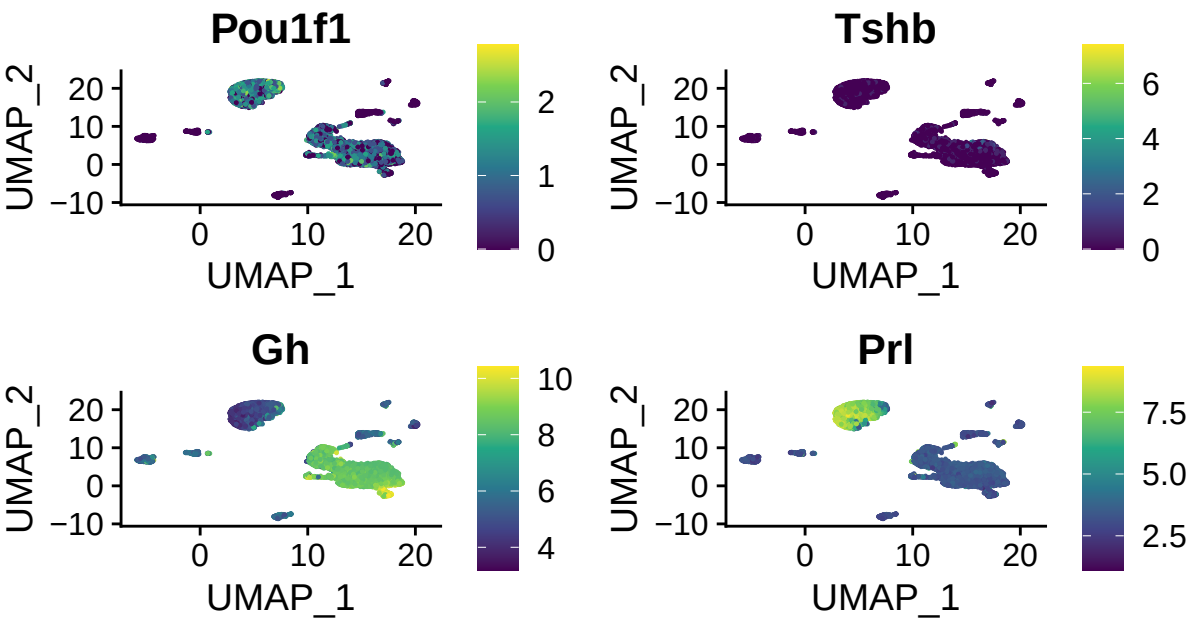
- Thyrotrophs (Pou1f1 + Tshb),
- Somatotrophs(Pou1f1 + Gh)
- Lactotrophs (Pou1f1 + Prl)
- Gonadotrophs (Lhb)
- Melanotrophs (Pomc + Pcsk2 + Pax7)
- Corticotrophs (Pomc + Crhr1 + Avpr1b + Gpc5 - Pcsk2 - Pax7)

Feature plots were used to visually assess gene expression levels of single cells. Cell type was assigned based on visual inspection of canonical markers [Fletcher et al, Chung et al].

First the thyrotophs (Pou1f1 + Tshb), somatotrophs (Pou1f1 + Gh) and lactotrops (Pou1f1 + Prl) may be excluded based on gene expression:

```
#Visualise the canonical marker genes

#first exclude Pou1f1 expressing cells (Thyrotrophs, Somatotrophs, Lactotrophs)
FeaturePlot(data, features = c("Pou1f1","Tshb", "Gh","Prl"), pt.size = 0.2)&scale_color_viridis_c()
```

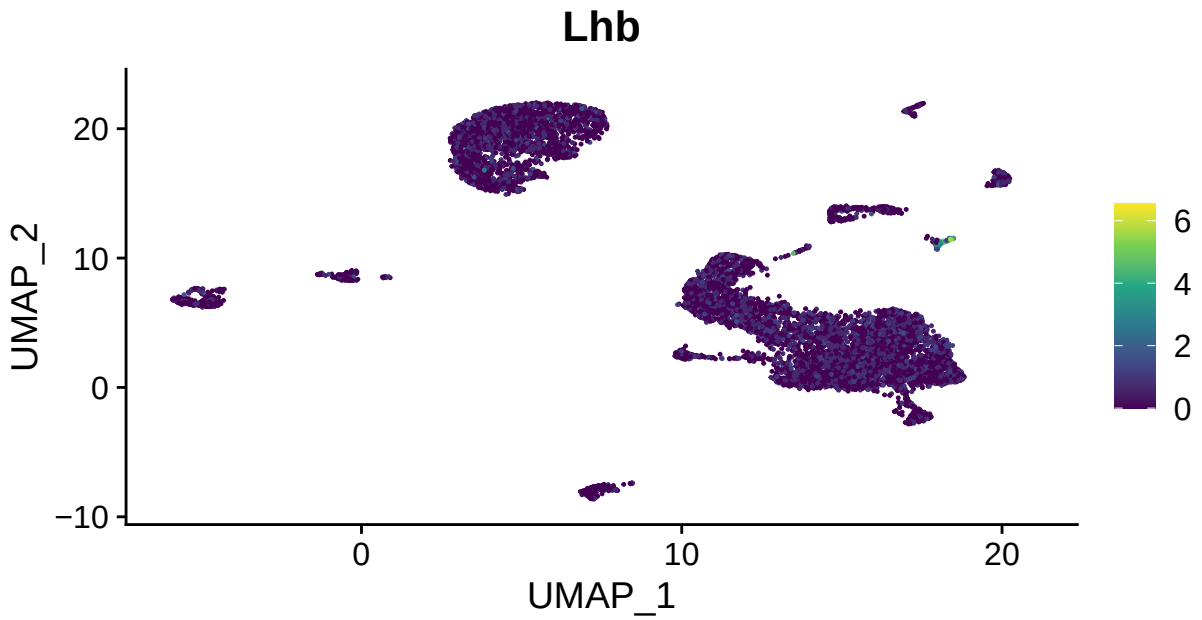
```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```


Pou1f1
Tshb
Gh
Prl
UMAP_1
UMAP_2
20
10
0
−10
0
10
20
2
1
0
6
4
2
0
10
8
6
4
7.5
5.0
2.5

Then the Gonadotoprhs may be excluded by Lhb expression;

```
#Visualise the canonical marker genes


#then exclude the gonadotrophs
FeaturePlot(data, features = c("Lhb"), pt.size = 0.2)&scale_color_viridis_c()
```

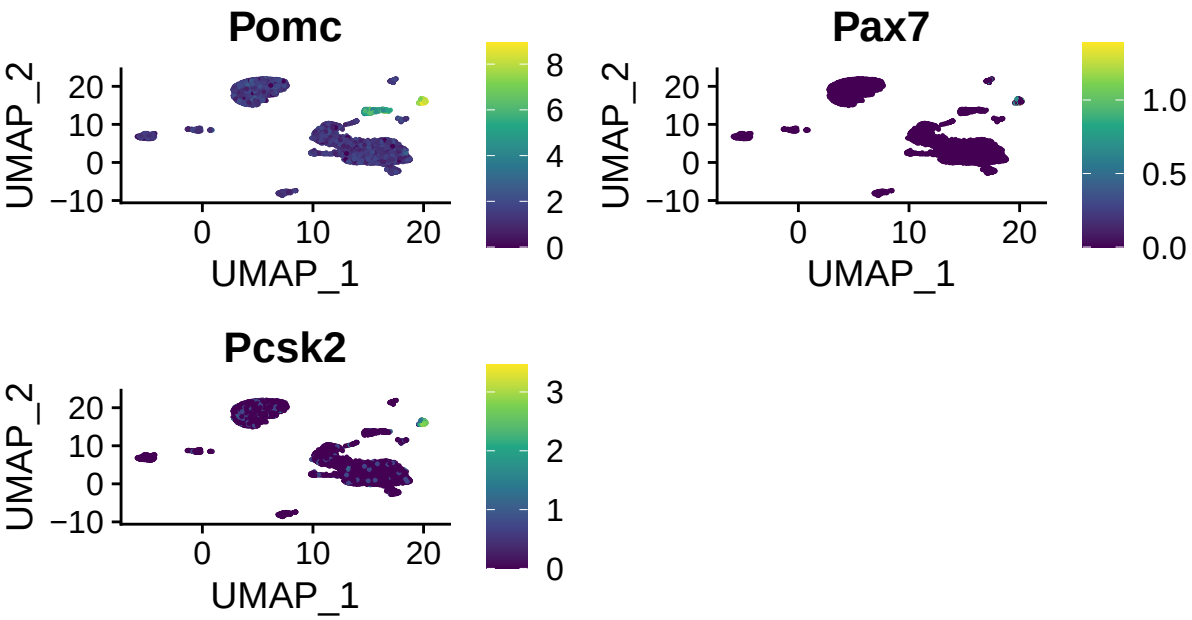
```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```



Then the melanotrophs may be excluded based on Pomc, Pcsk2 and Pax7 expression:

```
#Visualise the canonical marker genes
#then exclude the melanotrophs
FeaturePlot(data, features = c("Pomc","Pax7","Pcsk2"), pt.size = 0.2)&scale_color_viridis_c()
```

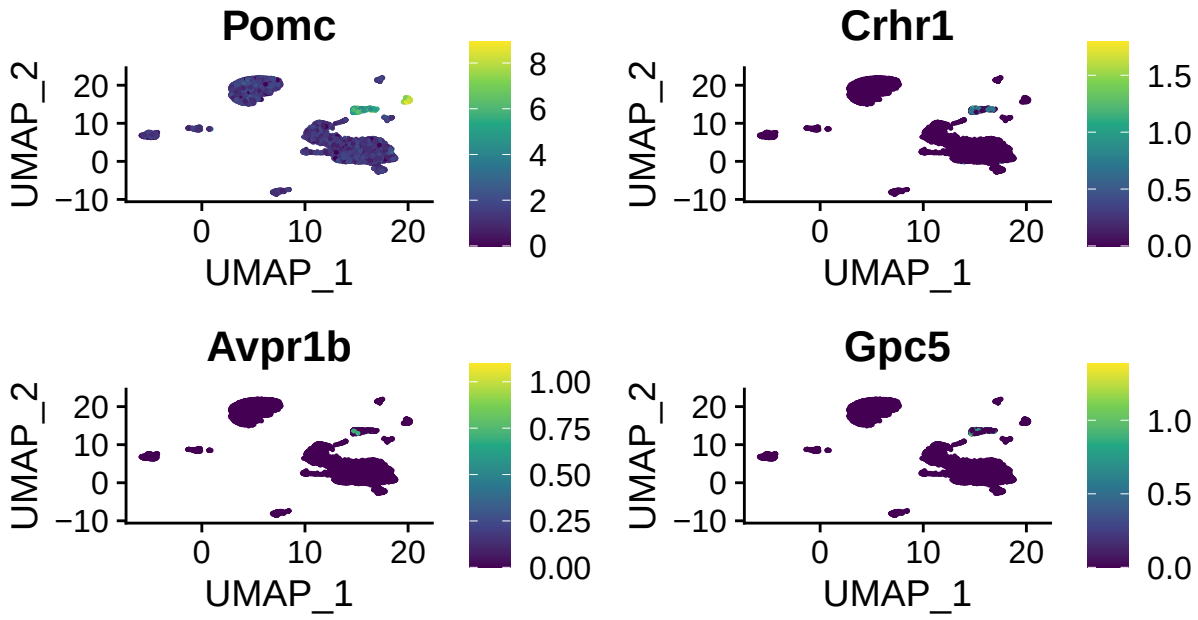
```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```


Pomc
UMAP_2
20
10
0
−10
0
10
20
UMAP_1
8
6
4
2
0
Pax7
UMAP_2
20
10
0
−10
0
10
20
UMAP_1
1.0
0.5
0.0
Pcsk2
UMAP_2
20
10
0
−10
0
10
20
UMAP_1
3
2
1
0

Then confirm the identification of the corticotrophs by Pomc, Crh1, Avpr1b and Gpc5.

```
#Visualise the canonical marker genes
#then id the corticotrophs
FeaturePlot(data, features = c("Pomc", "Crhr1","Avpr1b","Gpc5"), pt.size = 0.2)&scale_color_viridis_c()
```

```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```



These plots would suggest cluster 12 is the most likely corticotroph cluster.

Step 7. Isolate desired cell cluster

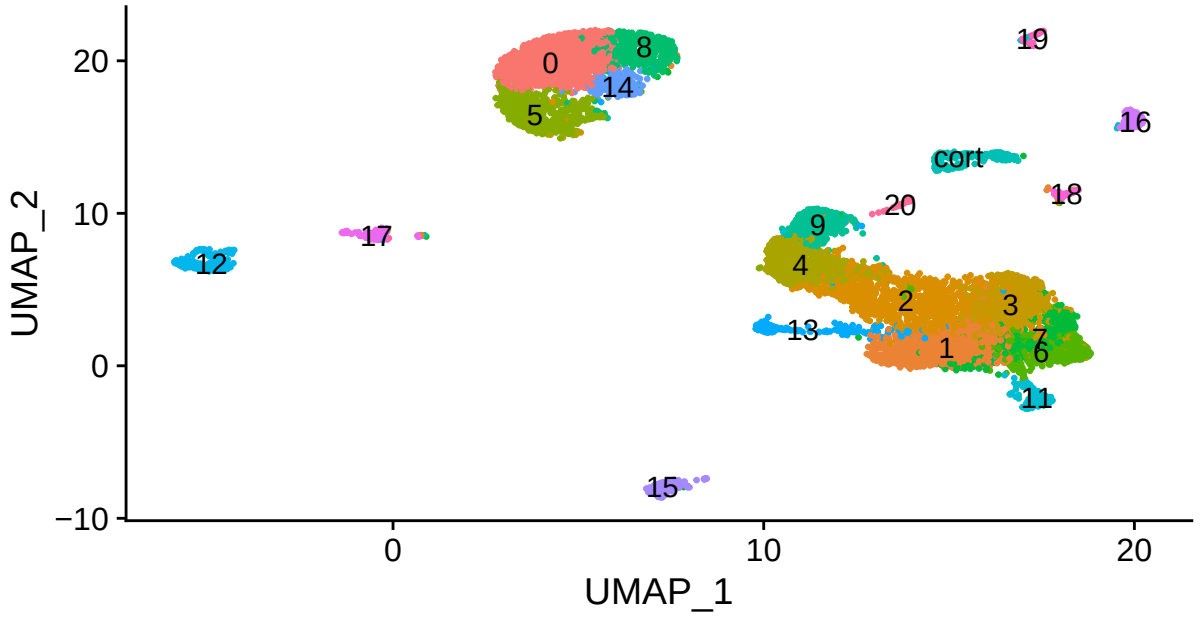
To examine cell homogeneity within the corticotrophs, the cluster identified in the previous step was pulled out as a subset.

The desired cluster must be manually entered upon visual inspection of the gene expression feature plots in the previous step.

```
#enter the cotricoph cell ids, found in visualisation. Note change to correct cluster #s or will miss-m
#new.cluster.ids <- c("0", "1", "2", "3", "4", "5", "6","7","8","9","10","11","12","13","14","15","16",
new.cluster.ids <- c("0", "1", "2", "3", "4", "5", "6","7","8","9","cort","11","12","13","14","15","16"

names(new.cluster.ids) <- levels(data)
```

As a sanity check that the correct cluster was selected, a dimensional reduction plot;

```
#Rename the cluster
data <- RenameIdents(data, new.cluster.ids)
#then put on the plot to confirm slected the correct one (sanity check)
DimPlot(data, reduction ="umap", label = TRUE, pt.size = 0.5)+NoLegend()
```



Find the biomarkers for the corticotroph cluster compared to all other cells, what gene expression makes it different from the others? For a gene to be included in the differential expression for the cluster vs all other clusters, it must be expressed in at least 25% of the cells in the cluster `min.pct=0.25`. `Pct.1` is the percentage of cells in the cluster where the gene is detected, `pct.2` is the percentage of cells on average in all the other clusters where the gene is detected.

```
#find all markers for cort cluster that make it different from all other clusters
cort.markers <- FindMarkers(data, ident.1="cort", min.pct=0.25)
```

```
## For a more efficient implementation of the Wilcoxon Rank Sum Test,
## (default method for FindMarkers) please install the limma package
```

```
## ------------------------------------------
## install.packages('BiocManager')
## BiocManager::install('limma')
## ------------------------------------------
## After installation of limma, Seurat will automatically use the more
## efficient implementation (no further action necessary).
## This message will be shown once per session
```

```
#display top 10 markers differentially expressed by the corticotroph cluster
head(cort.markers, n=10)
```

```
##                  p_val avg_logFC pct.1 pct.2     p_val_adj
## Tnnt1     0.000000e+00 1.1323338 0.482 0.010  0.000000e+00
## Atp1a2    0.000000e+00 0.6902586 0.547 0.022  0.000000e+00
## Sparcl1   0.000000e+00 0.6800784 0.615 0.028  0.000000e+00
## Tbx19     0.000000e+00 0.6034896 0.534 0.010  0.000000e+00
## Crhr1     0.000000e+00 0.5801306 0.522 0.002  0.000000e+00
## AW551984 1.598204e-297 1.3829092 0.927 0.140 2.660051e-293
## Baiap3   6.355815e-209 0.4952264 0.514 0.047 1.057862e-204
## Acbd7    3.875921e-202 0.3665982 0.368 0.023 6.451083e-198
## Tmem91   8.862934e-179 0.2995059 0.296 0.016 1.475147e-174
## Hn1      1.707399e-169 0.8742906 0.814 0.169 2.841794e-165
```

The data for the corticotroph cluster is extracted for further study using `subset`.

```
cortico <- subset(data, idents = "cort")
```

Step 8. Re-cluster corticotroph cluster and study cluster homogeneity

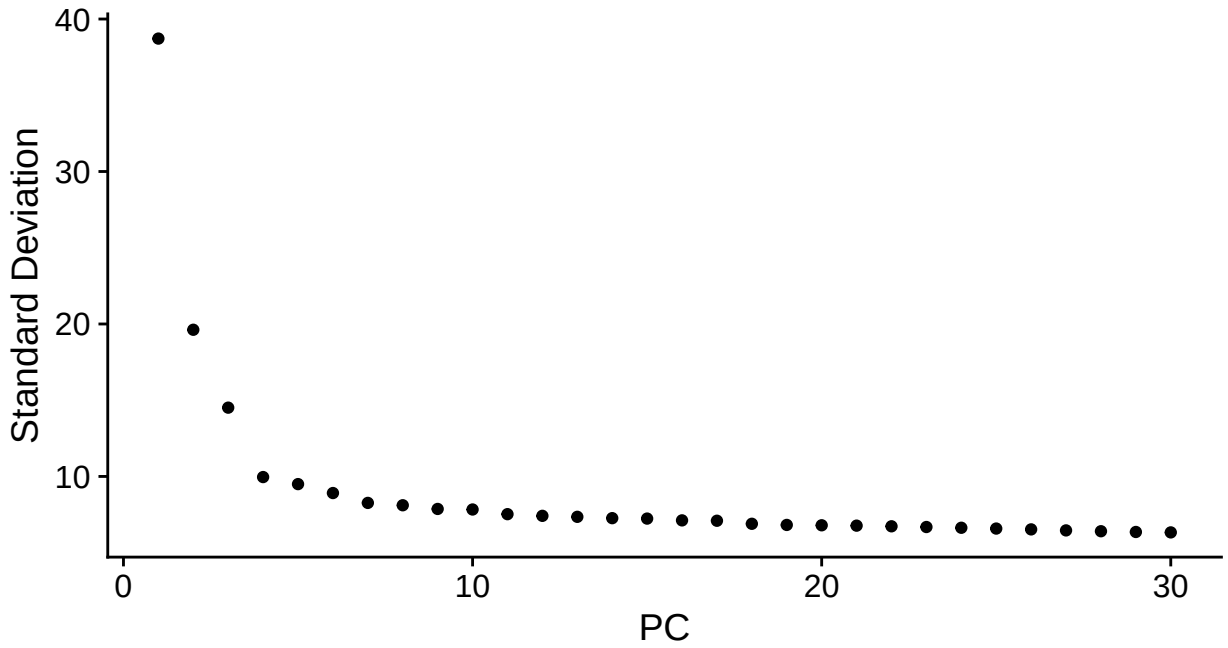
To investigate cell homogeneity within the cell cluster of interest, the same procedure may be repeated;

Data transformation, identification of principal components (PCA) Dimensional reduction (UMAP) *Identification of nearest neighbours (SNN)* Clustering using SNN.

```
#dimensional analysis
#PCA to identify PCs
cortico <- RunPCA(cortico,verbose= FALSE)
```

Elbow plot to establish dimensionality;

```
#Do we need to keep all PCs, or is majority of data captured by a certain point?
ElbowPlot(cortico, ndims=30)
```



Identify most of information retained in 12 PCs

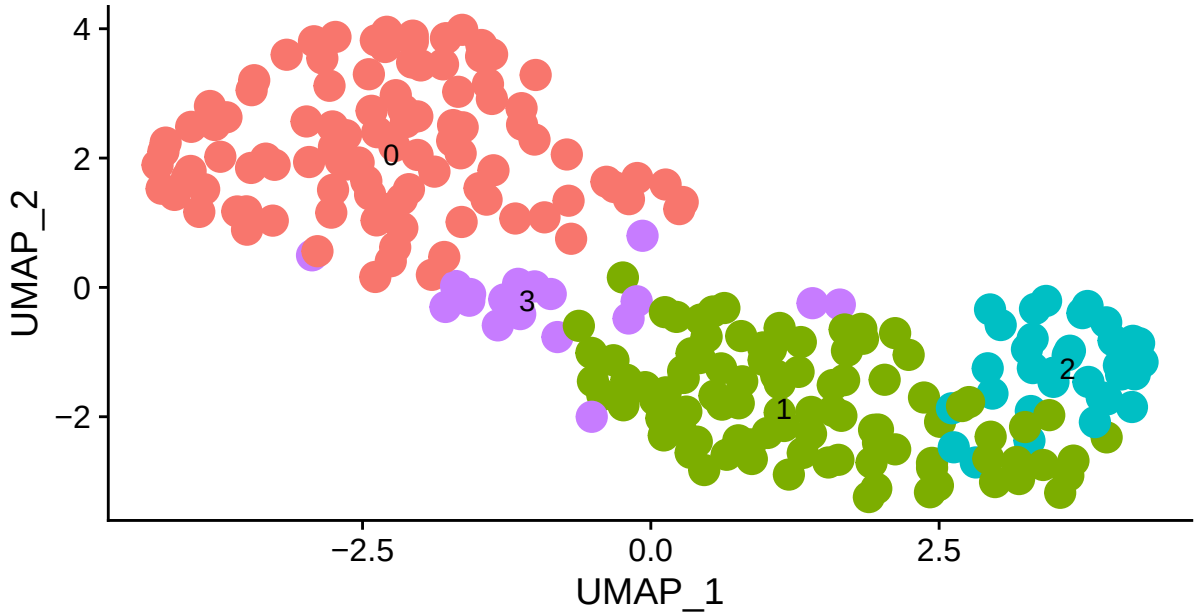
```
#dimensional reduction
#UMAP dimensional reduction of the identified PCs, change dims!
cortico <- RunUMAP(cortico, dims=1:12, verbose=FALSE) #note because SCTransform passes 3000 features ca
```

```
## Warning: The default method for RunUMAP has changed from calling Python UMAP via reticulate to the R
## To use Python UMAP via reticulate, set umap.method to 'umap-learn' and metric to 'correlation'
## This message will be shown once per session
```

```
#Find nearest neighbours, change dims!
cortico <- FindNeighbors(cortico, dims=1:12, verbose=FALSE)

#CLuster based on neighbour distances
cortico <- FindClusters(cortico, verbose=FALSE)
```

A dimensional reduction plot can then reveal any heterogeneity in the cluster;

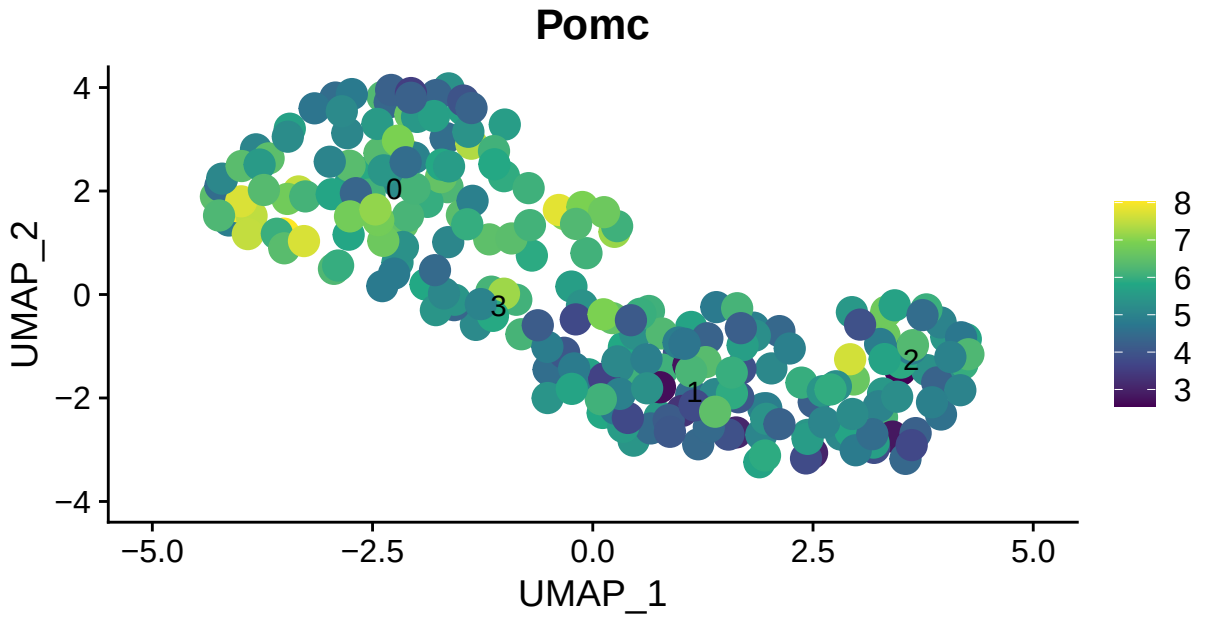
```
#plot the clusters
DimPlot(cortico, pt.size = 5, label=TRUE) + NoLegend()
```



A featureplot can be used to examine gene expression levels across these clusters within the corticotroph data;

```
#Visualise the canonical marker genes
FeaturePlot(cortico, features = c("Pomc"), pt.size = 5, label=TRUE)& scale_color_viridis_c()
```

```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```



Differential expression analysis can reveal differences between the clusters, for example the two top deferentially expressed genes for each cluster are;

```
#examine DE genes in the  corticotroph cluster
#find the markers for EVERY CLUSTER when compared to remaining cells, only +ve ones
cortico.markers <-FindAllMarkers(cortico, only.pos = TRUE, min.pct = 0.25, logfc.threshold = 0.25)
```

```
## Calculating cluster 0

## Calculating cluster 1

## Calculating cluster 2

## Calculating cluster 3
```

```
#examine top 2 deferentially expressed genes in each corticotroph cluster
df <- cortico.markers %>% group_by(cluster) %>% top_n(n=2, wt= avg_logFC)
```

```
head(df)
```

```
## Registered S3 method overwritten by 'cli':
##   method     from
##   print.boxx spatstat

## # A tibble: 6 x 7
## # Groups:   cluster [3]
##       p_val avg_logFC pct.1 pct.2 p_val_adj cluster gene
##       <dbl>     <dbl> <dbl> <dbl>     <dbl> <fct>   <chr>
## 1 6.51e-18     0.723 0.881 0.527  1.08e-13 0       Ndufa4
## 2 2.35e- 8     0.837 1     1      3.92e- 4 0       Pomc
```

```
## 3 3.46e-29     1.99  0.947 0.691  5.75e-25 1       Dnajb1
## 4 5.62e-25     1.67  0.874 0.336  9.36e-21 1       Npas4
## 5 1.20e-12     1.39  1     0.958  2.00e- 8 2       Mt1
## 6 3.02e-10     1.39  1     0.743  5.03e- 6 2       Mt2
```

9. Summary

This analysis based on 20 of 30 possible principal components identifies cluster 12 to be the corticotophs based on expression of canonical markers. Differential expression analysis showed the top differentially expressed gene found in 90% of corticodroh cells (and 16% of all other cells) was AW551984.

When re-clustered, the corticotrophs demonstrate cell heterogeneity with three visible sub clusters. A differential gene expression analysis of these three sub-clusters shows the top differentiated genes in each cluster to be Ndufa4 & Pomc in cluster zero, Dnajb1 & Hspb1 in cluster one and Mt1 & Mt2 in cluster three.

Future improvements

A future version of this script will include cell cycle considerations.

All suggestions for future improvements are welcome, as are comments on the current version. Please make these via the GitHub page.