R Notebook

#Time to try a different approach to pseudotime. It's clear the way it seems to be heading is clusters 2, 1, 3, 0.   
  
#using this tutorial: https://rnabioco.github.io/cellar/5\_trajectories.html  
  
#make new seurat object:  
#working directory, change if using on system other than authors to folder with.rds file in.   
setwd("~/OneDrive - Queen Mary, University of London/QMUL/Lab/Coding/data/R/Seurat/SandLKerato")  
  
  
  
#load all libraries.   
library(Seurat)

## Attaching SeuratObject

library(tidyverse)

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

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──

## ✓ ggplot2 3.3.5 ✓ purrr 0.3.4  
## ✓ tibble 3.1.4 ✓ dplyr 1.0.7  
## ✓ tidyr 1.1.3 ✓ stringr 1.4.0  
## ✓ readr 2.0.1 ✓ forcats 0.5.1

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(ggplot2)  
library(DEGreport)  
library(Matrix)

##   
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':  
##   
## expand, pack, unpack

library(slingshot)

## Loading required package: princurve

library(tradeSeq)  
library(SingleCellExperiment)

## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

##   
## Attaching package: 'matrixStats'

## The following object is masked from 'package:dplyr':  
##   
## count

##   
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':  
##   
## colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,  
## colCounts, colCummaxs, colCummins, colCumprods, colCumsums,  
## colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,  
## colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,  
## colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,  
## colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,  
## colWeightedMeans, colWeightedMedians, colWeightedSds,  
## colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,  
## rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,  
## rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,  
## rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,  
## rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,  
## rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,  
## rowWeightedMads, rowWeightedMeans, rowWeightedMedians,  
## rowWeightedSds, rowWeightedVars

## Loading required package: GenomicRanges

## Loading required package: stats4

## Loading required package: BiocGenerics

## Loading required package: parallel

##   
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':  
##   
## clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,  
## clusterExport, clusterMap, parApply, parCapply, parLapply,  
## parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:dplyr':  
##   
## combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':  
##   
## IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':  
##   
## anyDuplicated, append, as.data.frame, basename, cbind, colnames,  
## dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,  
## grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,  
## order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,  
## rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,  
## union, unique, unsplit, which.max, which.min

## Loading required package: S4Vectors

##   
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:Matrix':  
##   
## expand

## The following objects are masked from 'package:dplyr':  
##   
## first, rename

## The following object is masked from 'package:tidyr':  
##   
## expand

## The following object is masked from 'package:base':  
##   
## expand.grid

## Loading required package: IRanges

##   
## Attaching package: 'IRanges'

## The following objects are masked from 'package:dplyr':  
##   
## collapse, desc, slice

## The following object is masked from 'package:purrr':  
##   
## reduce

## Loading required package: GenomeInfoDb

## Loading required package: Biobase

## Welcome to Bioconductor  
##   
## Vignettes contain introductory material; view with  
## 'browseVignettes()'. To cite Bioconductor, see  
## 'citation("Biobase")', and for packages 'citation("pkgname")'.

##   
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':  
##   
## rowMedians

## The following objects are masked from 'package:matrixStats':  
##   
## anyMissing, rowMedians

##   
## Attaching package: 'SummarizedExperiment'

## The following object is masked from 'package:SeuratObject':  
##   
## Assays

## The following object is masked from 'package:Seurat':  
##   
## Assays

library(RColorBrewer)  
library(gam)

## Loading required package: splines

## Loading required package: foreach

##   
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':  
##   
## accumulate, when

## Loaded gam 1.20

library(cowplot)  
library(scales)

##   
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':  
##   
## discard

## The following object is masked from 'package:readr':  
##   
## col\_factor

library(gridExtra)

##   
## Attaching package: 'gridExtra'

## The following object is masked from 'package:Biobase':  
##   
## combine

## The following object is masked from 'package:BiocGenerics':  
##   
## combine

## The following object is masked from 'package:dplyr':  
##   
## combine

library(destiny)

##   
## Attaching package: 'destiny'

## The following object is masked from 'package:SummarizedExperiment':  
##   
## distance

## The following object is masked from 'package:GenomicRanges':  
##   
## distance

## The following object is masked from 'package:IRanges':  
##   
## distance

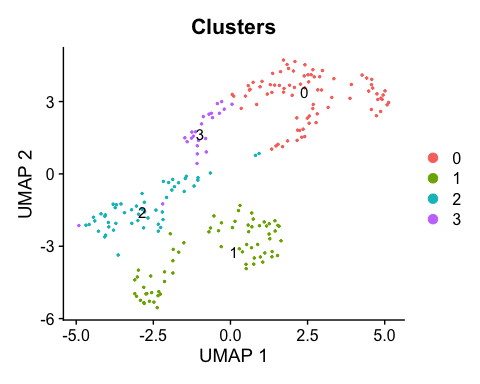
library(ComplexHeatmap)

## Loading required package: grid

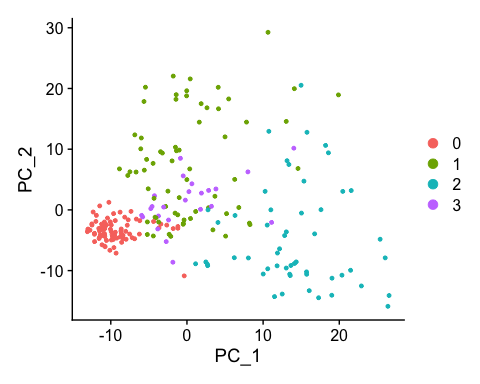
## ========================================  
## ComplexHeatmap version 2.6.2  
## Bioconductor page: http://bioconductor.org/packages/ComplexHeatmap/  
## Github page: https://github.com/jokergoo/ComplexHeatmap  
## Documentation: http://jokergoo.github.io/ComplexHeatmap-reference  
##   
## If you use it in published research, please cite:  
## Gu, Z. Complex heatmaps reveal patterns and correlations in multidimensional   
## genomic data. Bioinformatics 2016.  
##   
## This message can be suppressed by:  
## suppressPackageStartupMessages(library(ComplexHeatmap))  
## ========================================

theme\_set(theme\_cowplot())  
  
  
#Load in .rds object containing filtered and clustered data.  
kerato <- readRDS(file = "smalllargekera.rds")

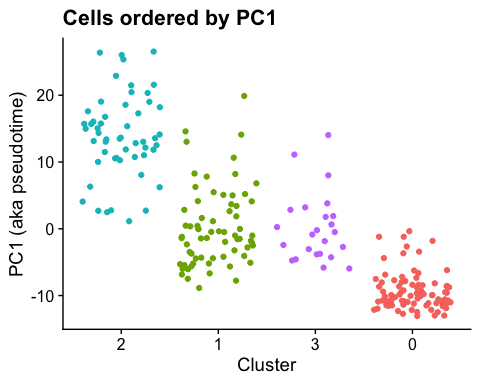
#seurat object already created, but need to run PCA reduction:  
  
newpseudokerato <- RunPCA(kerato, verbose = FALSE)  
  
DimPlot(newpseudokerato, reduction = "umap", label = TRUE, pt.size = 0.5) + ggtitle('Clusters') +  
 xlab('UMAP 1') +   
 ylab('UMAP 2') +   
 theme(plot.title = element\_text(hjust = 0.5)) #centre title



# plot the PCA reduction  
DimPlot(newpseudokerato, reduction = "pca")



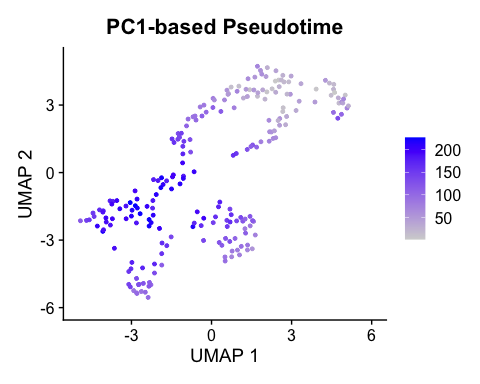
# get the PC1 values for each cell, and store in a data frame  
plt\_dat <- FetchData(newpseudokerato, c("PC\_1", "seurat\_clusters"))  
  
# reorder cell\_type based on known developmental time  
cell\_type <- factor(plt\_dat$seurat\_clusters,  
 levels = c("2",  
 "1",  
 "3",  
 "0"))  
  
# add the reordered cell\_type info as a new column to plt\_data  
plt\_dat$cell\_type <- cell\_type  
  
# plot the cells ordered by pseudotime (PC1)  
ggplot(plt\_dat, aes(cell\_type, PC\_1)) +  
 geom\_jitter(aes(color = seurat\_clusters)) +  
 labs(y = "PC1 (aka pseudotime)", x= 'Cluster') +  
 ggtitle("Cells ordered by PC1") + theme(legend.position="none")



# get PC1 values and rank to generate a 'pseudotime'  
ptime <- FetchData(newpseudokerato, "PC\_1")  
ptime$ptime <- rank(ptime$PC\_1)  
  
  
# add pseudotime to the metadata  
newpseudokerato <- AddMetaData(newpseudokerato,   
 ptime$ptime,   
 col.name = "Pseudotime\_manual")  
  
#save new pseudo as a .rds file  
saveRDS(newpseudokerato, file = "newpseudokera.rds")

#reset working directory  
setwd("~/OneDrive - Queen Mary, University of London/QMUL/Lab/Coding/data/R/Seurat/SandLKerato")  
  
#load in new newpseudokerato from .rds file  
newpseudokerato <- readRDS(file = "newpseudokera.rds")

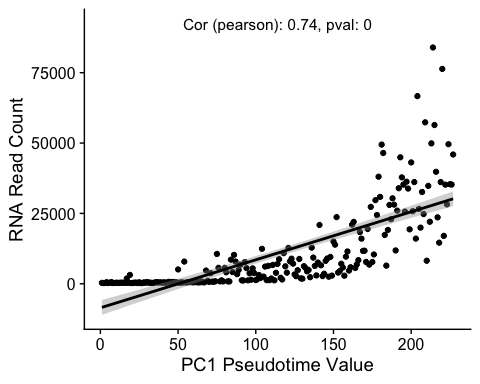
#plots   
#plot showing pseudotime over clusters  
FeaturePlot(newpseudokerato, features = ('Pseudotime\_manual')) + xlab('UMAP 1') +  
 ylab('UMAP 2') + ggtitle('PC1-based Pseudotime')



#plot for n\_count RNA over pseudo  
newpseudokerato %>%  
 tidyseurat::ggplot(aes(x=Pseudotime\_manual, y=nCount\_RNA)) +   
 geom\_point() +   
 stat\_smooth(method="lm", se=TRUE, colour = 'black', plot.cor = TRUE) +   
 geom\_cor(method = "pearson") + xlab('PC1 Pseudotime Value') +  
 ylab('RNA Read Count')

## Warning: Ignoring unknown parameters: plot.cor

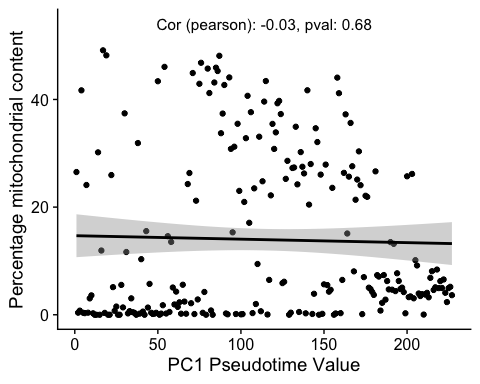
## `geom\_smooth()` using formula 'y ~ x'



#plot for % mitochondrial RNA over pseudo  
newpseudokerato %>%  
 tidyseurat::ggplot(aes(x=Pseudotime\_manual, y=percent.mt)) + geom\_point() +   
 stat\_smooth(method="lm", se=TRUE, colour = 'black', plot.cor = TRUE) +   
 geom\_cor(method = "pearson") + xlab('PC1 Pseudotime Value') +  
 ylab('Percentage mitochondrial content')

## Warning: Ignoring unknown parameters: plot.cor

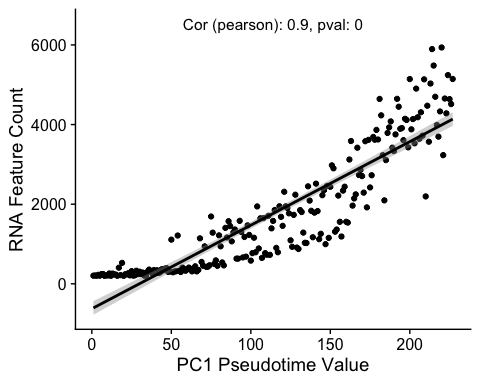
## `geom\_smooth()` using formula 'y ~ x'



#plot for n\_feature RNA over pseudo  
newpseudokerato %>%  
 tidyseurat::ggplot(aes(x=Pseudotime\_manual, y=nFeature\_RNA)) +   
 geom\_point() +   
 stat\_smooth(method="lm", se=TRUE, colour = 'black', plot.cor = TRUE) +   
 geom\_cor(method = "pearson") + xlab('PC1 Pseudotime Value') +  
 ylab('RNA Feature Count')

## Warning: Ignoring unknown parameters: plot.cor

## `geom\_smooth()` using formula 'y ~ x'

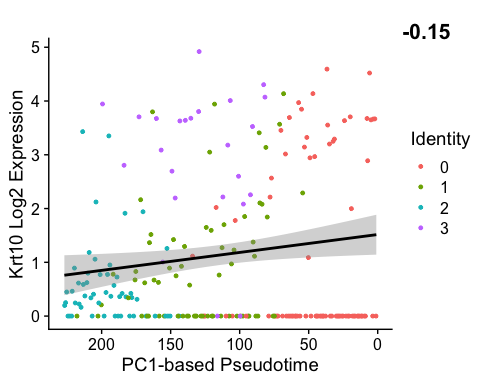


#reset working directory  
setwd("~/OneDrive - Queen Mary, University of London/QMUL/Lab/Coding/data/R/Seurat/SandLKerato")  
  
#load in new newpseudokerato from .rds file  
newpseudokerato <- readRDS(file = "newpseudokera.rds")

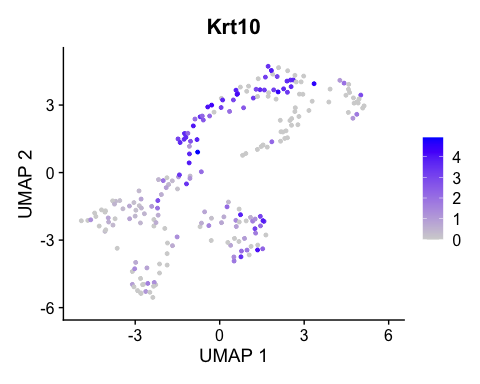
# this section builds lists containing gene IDs. These IDs are sorted into lists related to their function and/or their research area associated with them.   
  
# kerato\_marker\_genes contain general keratinocyte differentiation markers.  
kerato\_marker\_genes <- c("Krt10", "Krt5", "Krt14", "Sprr1a", "Sprr1b", "Evpl", "Dsp", "Ppl", "Klk11", "Spink5", "Klf4", "Klk6", "Klk7", "Klk8", "Klk10")  
  
# cell\_state\_markers contains genes used to mark the cell cycle state of cells, in this case currently proliferative markers.   
cell\_state\_markers <- c("Mki67","Pcna")  
  
# actin\_markers contains all genes associated with the actin cytoskeleton.   
actin\_markers <- c("Arhgdia","Camsap2", "Anxa1", "Rflnb", "Tmsb4x", "Arpc1b", "Sbsn","Dsg3", "Ahnak", "Cdh1", "Col3a1", "Tpt1", "Lgals7", "Rack1", "S100a11", "Eef1b2", "Ran", "Sprr1a", "Klf8", "Ptprf")  
  
actin\_GO\_markers <- c("Anxa1", "Sptbn2", "Rflnb", "Cavin3", "Tmsb4x", "Phpt1", "Cav1", "Ctsl", "Amotl2", "Hsp90b1", "Slc2a1", "Gmfg", "Dstn", 'Arhgdia')  
  
# plmn\_markers has putative lamin bodies markers investigated.   
plmn\_markers <- c("Ipo7", "Ran", "Snupn", "Matr3", "Nup153")  
  
# SGPB\_markers contains genes associated with condensates, primarly stress granules and P bodies.  
SGPB\_markers <- c("Ddx6", "Eif4e", "Nxf1", "Lsm14a", "Caprin1", "Csde1", "Pum1", "Zfp36")  
  
# other\_markers contains miscellaneous markers, notes below variable show reasoning.   
other\_markers <- c("Snhg11", "Fabp5", "Slc25a4", "Lgals7", "Tacstd2", "Fosb", "Cd44", "Trpv4", "Trpm4" )  
  
JAKSTATmarkers <- c("Jak1", "Jak2", "Jak3", "Stat2","Stat3","Stat4","Stat6")  
  
Request1 <- c("Pigu", "Ggnbp2", "Notch1", "Cav1", "Hsp90b1", "Akr1b1", "Adipor1")  
  
list\_of\_features <- c(kerato\_marker\_genes, cell\_state\_markers, actin\_markers, plmn\_markers, SGPB\_markers, other\_markers)

#gene of interest plots  
#gene of interest  
pseudotime\_scatter\_gene <- 'Krt10'  
  
#pseudotime graphs  
FeatureScatter(newpseudokerato, feature1 = "Pseudotime\_manual", feature2 = (pseudotime\_scatter\_gene)) +  
 scale\_x\_reverse() +  
 theme(plot.title = element\_text(hjust = 1.2, vjust = -2)) + # adjust pearsons coefficient to right hand side  
 stat\_smooth(method="lm", se=TRUE, colour = 'black') + # regression line  
 #labels  
 xlab('PC1-based Pseudotime') +   
 ylab(sprintf('%s Log2 Expression', pseudotime\_scatter\_gene)) +  
 labs(fill = "Cluster")

## `geom\_smooth()` using formula 'y ~ x'



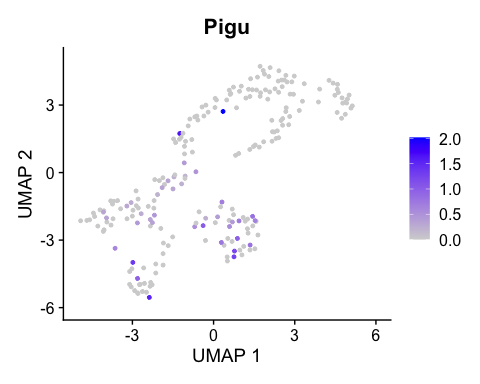
#cluster graph  
FeaturePlot(newpseudokerato, features = (pseudotime\_scatter\_gene)) + xlab('UMAP 1') +  
 ylab('UMAP 2')



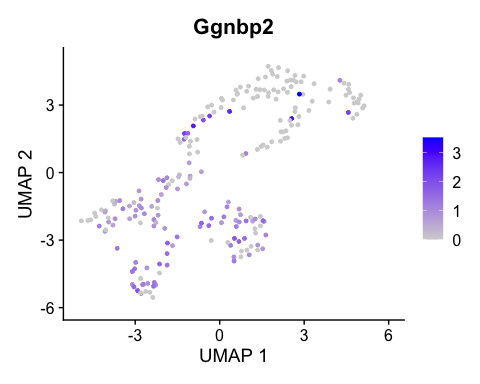
#set what batch of genes being turned to images:  
images\_gene\_list <- Request1

setwd("~/OneDrive - Queen Mary, University of London/QMUL/Lab/Coding/data/R/Seurat/SandLKerato/images")  
  
for (gene in images\_gene\_list){  
 #create cluster map  
 p <- FeaturePlot(newpseudokerato, features = (gene)) + xlab('UMAP 1') +  
 ylab('UMAP 2')  
 #print plot to console  
 print(p)  
   
 #this saves as a .png in the images folder with default settings.  
 ggsave(filename = (sprintf('%s cluster', gene)), plot = p, device = 'png')  
}

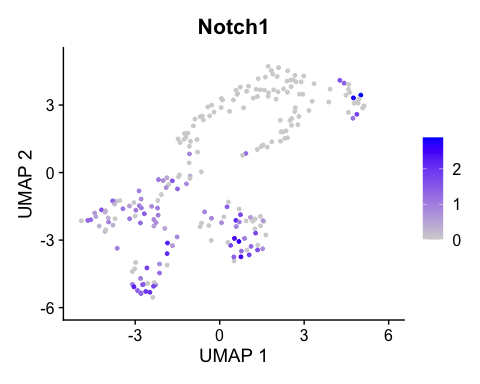
## Saving 5 x 4 in image



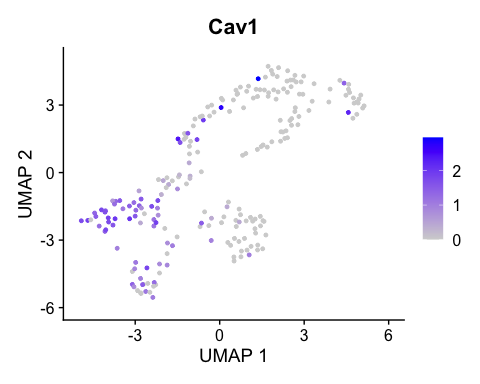
## Saving 5 x 4 in image



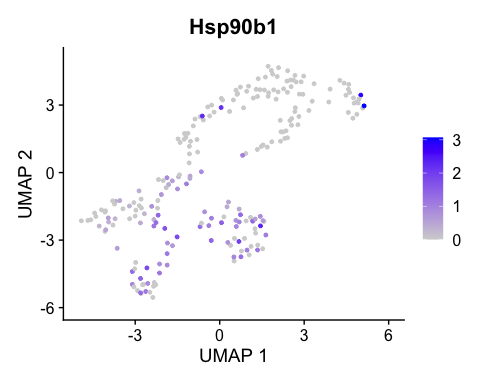
## Saving 5 x 4 in image



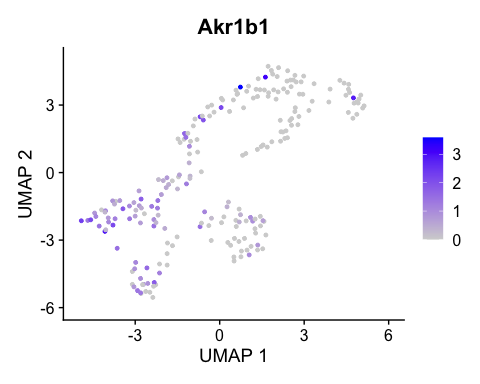
## Saving 5 x 4 in image



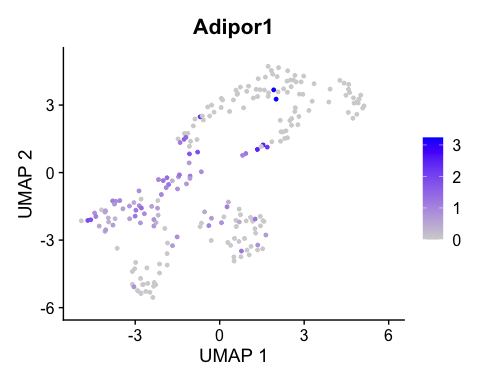
## Saving 5 x 4 in image



## Saving 5 x 4 in image



## Saving 5 x 4 in image

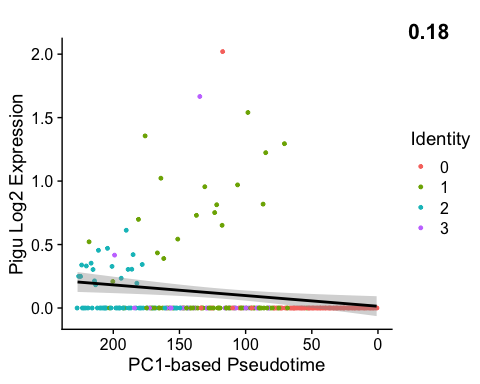


# chunk plotting all pseudotime line graphs for genes in a character vector  
setwd("~/OneDrive - Queen Mary, University of London/QMUL/Lab/Coding/data/R/Seurat/SandLKerato/images")  
  
for (gene in images\_gene\_list){  
  
 #build initial feature scatter plot  
 p <- FeatureScatter(newpseudokerato, feature1 = "Pseudotime\_manual", feature2 = (gene))  
 #add overlays  
 p <- p + theme(plot.title = element\_text(hjust = 1.2, vjust = -2)) + # adjust pearsons coefficient to right hand side  
 stat\_smooth(method="lm", se=TRUE, colour = 'black') + # regression line  
 scale\_x\_reverse() +  
 #labels  
 xlab('PC1-based Pseudotime') +  
 ylab(sprintf('%s Log2 Expression', gene)) +  
 labs(fill = "Cluster")  
 #print whole plot 'p'  
 print(p)  
   
 #this saves as a .png in the images folder with default settings.  
 ggsave(filename = (sprintf('%s line', gene)), plot = p, device = 'png')  
}

## `geom\_smooth()` using formula 'y ~ x'

## Saving 5 x 4 in image

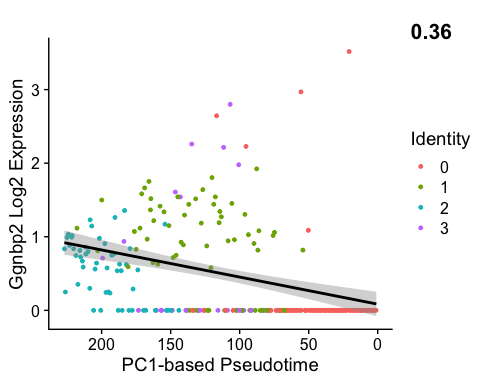
## `geom\_smooth()` using formula 'y ~ x'



## `geom\_smooth()` using formula 'y ~ x'

## Saving 5 x 4 in image

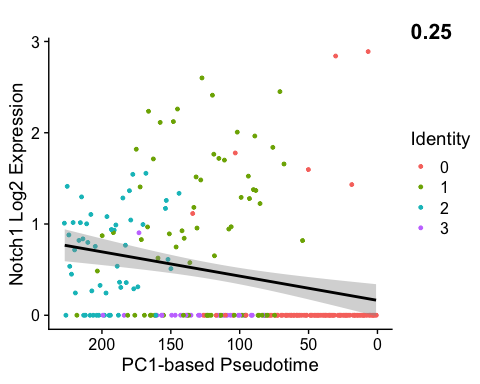
## `geom\_smooth()` using formula 'y ~ x'



## `geom\_smooth()` using formula 'y ~ x'

## Saving 5 x 4 in image

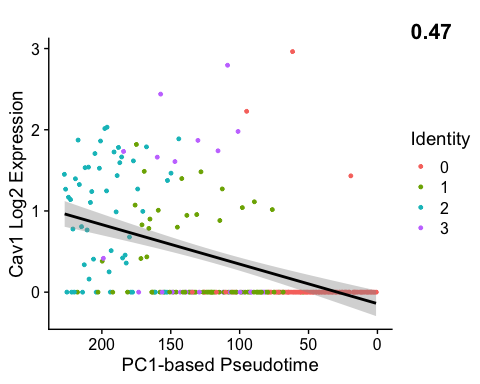
## `geom\_smooth()` using formula 'y ~ x'



## `geom\_smooth()` using formula 'y ~ x'

## Saving 5 x 4 in image

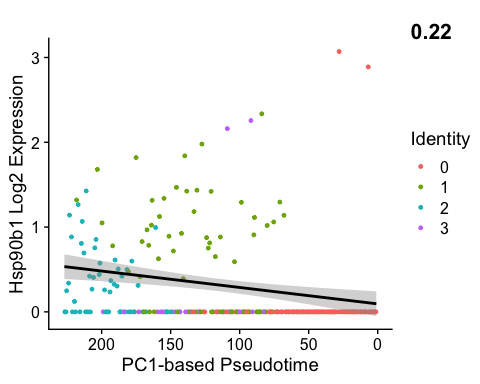
## `geom\_smooth()` using formula 'y ~ x'



## `geom\_smooth()` using formula 'y ~ x'

## Saving 5 x 4 in image

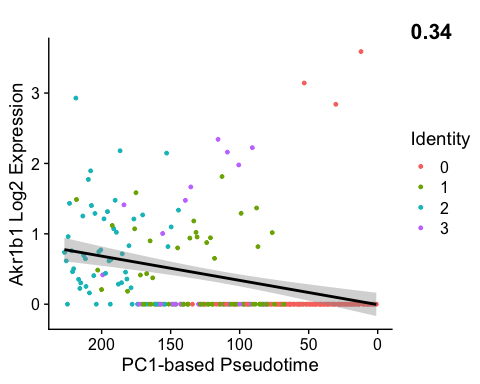
## `geom\_smooth()` using formula 'y ~ x'



## `geom\_smooth()` using formula 'y ~ x'

## Saving 5 x 4 in image

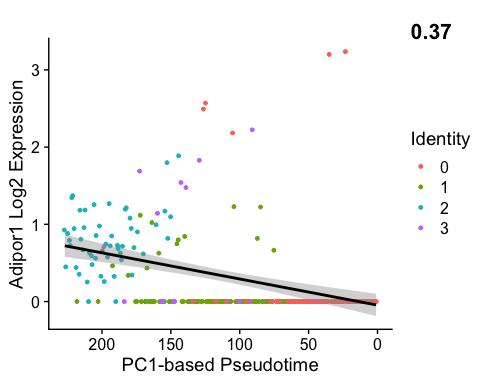
## `geom\_smooth()` using formula 'y ~ x'



## `geom\_smooth()` using formula 'y ~ x'

## Saving 5 x 4 in image

## `geom\_smooth()` using formula 'y ~ x'



metadataframe <- newpseudokerato@meta.data  
   
grouped\_qcs <- metadataframe %>% group\_by(seurat\_clusters)   
  
summary(newpseudokerato@meta.data)

## orig.ident nCount\_RNA nFeature\_RNA percent.mt   
## Kerato:227 Min. : 245 Min. : 201.0 Min. : 0.0000   
## 1st Qu.: 608 1st Qu.: 372.5 1st Qu.: 0.7492   
## Median : 4710 Median :1286.0 Median : 5.5753   
## Mean :10898 Mean :1765.8 Mean :13.9546   
## 3rd Qu.:14239 3rd Qu.:2797.0 3rd Qu.:26.2964   
## Max. :83967 Max. :5936.0 Max. :49.1361   
## RNA\_snn\_res.0.4 seurat\_clusters Pseudotime\_manual  
## 0:82 0:82 Min. : 1.0   
## 1:68 1:68 1st Qu.: 57.5   
## 2:53 2:53 Median :114.0   
## 3:24 3:24 Mean :114.0   
## 3rd Qu.:170.5   
## Max. :227.0

grouped\_qcs %>% summarise(  
 mito = mean(percent.mt),  
 counts = mean(nCount\_RNA),  
 features = mean(nFeature\_RNA)  
)

## # A tibble: 4 × 4  
## seurat\_clusters mito counts features  
## <fct> <dbl> <dbl> <dbl>  
## 1 0 8.22 791. 375.  
## 2 1 32.2 10078. 2072.  
## 3 2 5.58 31063. 3828.  
## 4 3 0.320 3226. 1098.

grouped\_qcs %>% summarise(  
 mito = median(percent.mt),  
 counts = median(nCount\_RNA),  
 features = median(nFeature\_RNA)  
)

## # A tibble: 4 × 4  
## seurat\_clusters mito counts features  
## <fct> <dbl> <dbl> <dbl>  
## 1 0 1.84 510 311   
## 2 1 31.0 7742 1842.  
## 3 2 4.80 28195 3894   
## 4 3 0.208 2020. 906.

grouped\_qcs %>% summarise(  
 mito = min(percent.mt),  
 counts = min(nCount\_RNA),  
 features = min(nFeature\_RNA)  
)

## # A tibble: 4 × 4  
## seurat\_clusters mito counts features  
## <fct> <dbl> <dbl> <int>  
## 1 0 0 245 201  
## 2 1 8.40 3106 941  
## 3 2 0.0366 3567 1256  
## 4 3 0 822 463

grouped\_qcs %>% summarise(  
 mito = max(percent.mt),  
 counts = max(nCount\_RNA),  
 features = max(nFeature\_RNA)  
)

## # A tibble: 4 × 4  
## seurat\_clusters mito counts features  
## <fct> <dbl> <dbl> <int>  
## 1 0 49.1 5082 1110  
## 2 1 48.1 49453 5143  
## 3 2 25.1 83967 5936  
## 4 3 2.00 19342 3428

grouped\_qcs %>% summarise(  
 mito = quantile(percent.mt),  
 counts = quantile(nCount\_RNA),  
 features = quantile(nFeature\_RNA)  
)

## `summarise()` has grouped output by 'seurat\_clusters'. You can override using the `.groups` argument.

## # A tibble: 20 × 4  
## # Groups: seurat\_clusters [4]  
## seurat\_clusters mito counts features  
## <fct> <dbl> <dbl> <dbl>  
## 1 0 0 245 201   
## 2 0 0.282 364. 241   
## 3 0 1.84 510 311   
## 4 0 11.3 666. 406.  
## 5 0 49.1 5082 1110   
## 6 1 8.40 3106 941   
## 7 1 25.7 5647. 1439.  
## 8 1 31.0 7742 1842.  
## 9 1 40.0 10669 2368.  
## 10 1 48.1 49453 5143   
## 11 2 0.0366 3567 1256   
## 12 2 3.90 22030 3419   
## 13 2 4.80 28195 3894   
## 14 2 6.47 36273 4448   
## 15 2 25.1 83967 5936   
## 16 3 0 822 463   
## 17 3 0.0832 1300. 675   
## 18 3 0.208 2020. 906.  
## 19 3 0.328 2914. 1174.  
## 20 3 2.00 19342 3428