Part 5 code

## 1. Store gene count data, and annotation information, for each cell in a SCEset object from scater package

# Load necessary libraries and install necessary packages  
library(scater, quietly = TRUE)  
library(monocle)  
library(scran)  
library("SC3")

# Read in data of raw count expression values matrix  
molecules <- read.table("Pb.readcounts.txt", sep = "\t", header = TRUE, row.names=1)  
  
# Read in annotations for the cells  
anno <- read.table("PbM\_meta.txt", sep = "\t", header = TRUE)  
  
# Read in data that contains description of the features   
genes <- read.table("berg.desc", header=FALSE, row.names=1, quote = "", sep="\t")  
  
# Read in the phenotypic data for each cell  
pheno\_data <- new("AnnotatedDataFrame", anno)  
rownames(pheno\_data) <- pheno\_data$sample\_id  
  
# Store the data in SCEset object  
PbM <- SingleCellExperiment(assays = list(counts = as.matrix(molecules), logcounts=log(as.matrix(molecules+1))), colData=anno)  
  
# Adding row data of gene/transcript names to SCEset object (expression data set)  
rowData(PbM)$feature\_symbol <- row.names(molecules)

How many genes are expressed in average per cell

mean(molecules[,1])  
df <- data.frame(molecules)  
summary(molecules[,1])  
head(sapply(df, function(x) max(x)))  
mean(sapply(df, function(x) length(x[x>5])))

## 

## 2. Quality control

# Filter genes (features) with no counts  
keep\_feature <- rowSums(counts(PbM) > 0) > 0  
PbM <- PbM[keep\_feature, ]  
  
# Calculate quality control metrics   
PbM <- scater::calculateQCMetrics(PbM)

Visualise the distributions of quality metrics before fiiltering and normalization

# Visualise distribution of cell library size  
hist(PbM$total\_counts, xlab = "Library sizes", main ="", breaks=30, col = "grey", ylab = "Number of cells")  
  
# Visualise distribution of expressed features  
hist(PbM$total\_features, xlab = "Number of expressed genes", main ="", breaks=30, col = "grey", ylab = "Number of cells")

PCA and tSNE Plot before filtering out low quality cells (perhaps helps as a guidance)

scater::plotPCA(PbM, size\_by = "total\_features", colour\_by = "otto", exprs\_values = "logcounts")  
scater::plotTSNE(PbM, size\_by = "total\_features", colour\_by = "otto", exprs\_values = "logcounts")

# Filter cells with low counts(small library size)  
filter\_by\_total\_counts <- (PbM$total\_counts > 25000)  
table(filter\_by\_total\_counts)  
  
# Filter cells with low numbers of expressed features  
filter\_by\_expr\_features <- (PbM$total\_features > 1000)  
table(filter\_by\_expr\_features)  
  
# Use filtered data on expression matrix, but don't use controls  
PbM$use <- (filter\_by\_expr\_features & filter\_by\_total\_counts & !colData(PbM)$is\_control)  
  
# Filter out low abundance genes (genes with <10 reads in 5 cells)  
filter\_genes <- apply(counts(PbM[ , colData(PbM)$use]), 1, function(x) length(x[x >= 10]) >= 5)  
table(filter\_genes)  
  
rowData(PbM)$use <- filter\_genes  
  
dim(PbM[rowData(PbM)$use, colData(PbM)$use])  
  
# Create the filtered epression matrix  
PbM.clean <- PbM[rowData(PbM)$use, colData(PbM)$use]

Visualise the distributions of quality metrics after fiiltering

# Visualise distribution of cell library size  
hist(PbM.clean$total\_counts, xlab = "Library sizes", main ="", breaks=30, col = "grey", ylab = "Number of cells")  
  
# Visualise distribution of expressed features  
hist(PbM.clean$total\_features, xlab = "Number of expressed genes", main ="", breaks=30, col = "grey", ylab = "Number of cells")

## 

## 3. Normalization of read counts to remove cell specific biases, using the scran package

# Group cells into clusters of similar expression  
qclust <- scran::quickCluster(PbM.clean, min.size = 30)  
  
# Compute size factors for each cluster, which are used to scale the counts in each cell  
PbM.clean <- scran::computeSumFactors(PbM.clean, sizes = 15, clusters = qclust, positive=TRUE)  
summary(sizeFactors(PbM.clean))  
  
# Plot size factors against library sizes for all cells, to observe source of systematic differences between cells  
plot(sizeFactors(PbM.clean), PbM.clean$log10\_total\_counts, log="x", ylab = "Log10 total counts", xlab="Size Factor")   
  
# Normalize between clusters  
PbM.clean <- scater::normalize(PbM.clean)  
  
# Normalize expression values  
PbM.clean <- scater::normaliseExprs(PbM.clean)

Dimentionality reduction after filtering and Normalizing expression data

scater::plotPCA(PbM.clean, size\_by = "total\_features", colour\_by = "otto", exprs\_values = "logcounts")  
scater::plotTSNE(PbM.clean, size\_by = "total\_features", colour\_by = "otto", exprs\_values = "logcounts")

## 4. Determining distinct groups of single-cells based on their expression pattern, using SC3 package

library("SC3")  
library("pkgmaker")  
  
  
# Estimate the number of clusters needed  
PbM.clean <- sc3\_estimate\_k(PbM.clean)  
metadata(PbM.clean)$sc3$k\_estimation  
  
# Perform k-means clustering  
PbM.clean <- sc3(PbM.clean, ks = 6, biology = TRUE)

Looking at Biomarkers

png("Marker cluster\_3.png", width = 1000, height = 800, pointsize=5)  
sc3\_plot\_markers(PbM.clean, k = 6, show\_pdata = c("otto", "hoo", "sc3\_6\_log2\_outlier\_score", "sc3\_6\_clusters", "consensus"))  
#sc3\_interactive(PbM.clean)  
dev.off()