DE analysis for the 2016-17 cohort

EJC

Nov 2020, based on code from June 2020 + May 2019

Setup

The Babraham compute cluster does not contain a global tex installation, so a local tex is added to $PATH to allow knitting to pdf.

```
Sys.setenv(PATH=paste(Sys.getenv("PATH"),
                      "/bi/home/carre/texlive/2017/bin/x86_64-linux/",sep=":"))
```

```
library(dplyr)
library(SingleCellExperiment)
library(scater)
library(scran)
library(org.Hs.eg.db)
```

```
##
```

```
load(file = "../cohort_2016_17/data/SCE_QC_pass_finalised.RData")
```

```
########
# Get day 42 cells
# Aggregate by PID + UMAP clusters
# Use edgeR to calculate differentially expressed genes within each UMAP cluster for young vs old.
##########


summed <- sce[,sce$day == "d42"] %>%
  aggregateAcrossCells(.,
                     id=DataFrame(
                               cluster=.$clusters,
                               age=.$age, samples = .$PID))

summed
```

```
## class: SingleCellExperiment
## dim: 58051 87
## metadata(0):
## assays(1): counts
## rownames(58051): ENSG00000223972 ENSG00000227232 ... ENSG00000277475
##   ENSG00000268674
## rowData names(0):
## colnames: NULL
## colData names(48): lane i5 ... age samples
## reducedDimNames(2): PCA UMAP
## spikeNames(0):
```

```
## altExpNames(0):
```

```
#
# This combines several colData columns.
# This may be useful for index sort data (eg average cell width or for MFIs)
# But I have not explored exactly what data manipulation is taking place.
# Therefore treat the colData with extreme caution.
#

library(edgeR)
```

```
## Loading required package: limma

##
## Attaching package: 'limma'

## The following object is masked from 'package:scater':
##
##     plotMDS

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

##
## Attaching package: 'edgeR'

## The following object is masked from 'package:SingleCellExperiment':
##
##     cpm
```

```
### Loop for all clusters / labels

de.results.d42.YvsO <- list()

for (i in levels(factor(summed$clusters))) {
  current <- summed[,i==summed$clusters]
  y <- DGEList(counts(current), samples=colData(current))

  discarded <- isOutlier(colSums(counts(current)), log=TRUE, type="lower")
  y <- y[,!discarded]
  y <- y[filterByExpr(y, min.count = 1, min.total.count =10, min.prop = 0.1),]
  y <- calcNormFactors(y)

  design <- try(
    model.matrix(~  factor(age), y$samples),
    silent=TRUE
  )
  if (is(design, "try-error") ||
      qr(design)$rank==nrow(design) ||
      qr(design)$rank < ncol(design))
  {
    # Skipping labels without contrasts or without
    # enough residual d.f. to estimate the dispersion.
    next
  }

  y <- estimateDisp(y, design)
  fit <- glmQLFit(y, design)
  res <- glmQLFTest(fit, coef=ncol(design))
  de.results.d42.Yvs0[[i]] <- res
}

#
summaries.d42.Yvs0 <- lapply(de.results.d42.Yvs0, FUN=function(x) summary(decideTests(x, adjust.method =


sum.tab.d42.Yvs0 <- do.call(rbind, summaries.d42.Yvs0)
sum.tab.d42.Yvs0
```

```
##   Down NotSig Up
## 1    0   7486  0
## 2    0   4023  0
## 3    0   5427  0
## 4    0   2841  0
## 5    0  10062  0
```

```
########
# Get day 0 cells
# Aggregate by PID + UMAP clusters
# Use edgeR to calculate differentially expressed genes within each UMAP cluster for young vs old.
##########


summed <- sce[,sce$day == "d0"] %>%
  aggregateAcrossCells(.,
                       id=DataFrame(
                         cluster=.$clusters,
                         age=.$age, samples = .$PID))

summed
```

```
## class: SingleCellExperiment
## dim: 58051 58
## metadata(0):
## assays(1): counts
## rownames(58051): ENSG00000223972 ENSG00000227232 ... ENSG00000277475
##   ENSG00000268674
## rowData names(0):
## colnames: NULL
## colData names(48): lane i5 ... age samples
## reducedDimNames(2): PCA UMAP
## spikeNames(0):
## altExpNames(0):
```

```
#
# This combines several colData columns.
# This may be useful for index sort data (eg average cell width or for MFIs)
# But I have not explored exactly what data manipulation is taking place.
# Therefore treat the colData with extreme caution.
#

library(edgeR)
```

```
### Loop for all clusters / labels

de.results.d0.YvsO <- list()

for (i in levels(factor(summed$clusters))) {
  current <- summed[,i==summed$clusters]
  y <- DGEList(counts(current), samples=colData(current))

  discarded <- isOutlier(colSums(counts(current)), log=TRUE, type="lower")
  y <- y[,!discarded]
  y <- y[filterByExpr(y, min.count = 1, min.total.count =10, min.prop = 0.1),]
  y <- calcNormFactors(y)

  design <- try(
    model.matrix(~  factor(age), y$samples),
    silent=TRUE
  )
  if (is(design, "try-error") ||
      qr(design)$rank==nrow(design) ||
      qr(design)$rank < ncol(design))
  {
    # Skipping labels without contrasts or without
    # enough residual d.f. to estimate the dispersion.
    next
  }

  y <- estimateDisp(y, design)
  fit <- glmQLFit(y, design)
  res <- glmQLFTest(fit, coef=ncol(design))
  de.results.d0.YvsO[[i]] <- res
}

#
summaries.d0.YvsO <- lapply(de.results.d0.YvsO, FUN=function(x) summary(decideTests(x, adjust.method =


sum.tab.d0.YvsO <- do.call(rbind, summaries.d0.YvsO)
sum.tab.d0.YvsO
```

```
##   Down NotSig Up
## 1    0    897  0
## 2    0   1144  0
## 3    0    901  0
## 4    0   1216  0
## 5    0   2315  0
```

```
########
# Get young cells
# Aggregate by PID + UMAP clusters
# Use edgeR to calculate differentially expressed genes within each UMAP cluster for young d0 vs young
##########


summed <- sce[,sce$age == "young"] %>%
  aggregateAcrossCells(.,
                       id=DataFrame(
                         cluster=.$clusters,
                         day=.$day, samples = .$PID))

summed
```

```
## class: SingleCellExperiment
## dim: 58051 71
## metadata(0):
## assays(1): counts
## rownames(58051): ENSG00000223972 ENSG00000227232 ... ENSG00000277475
##   ENSG00000268674
## rowData names(0):
## colnames: NULL
## colData names(48): lane i5 ... day samples
## reducedDimNames(2): PCA UMAP
## spikeNames(0):
## altExpNames(0):
```

```
#
# This combines several colData columns.
# This may be useful for index sort data (eg average cell width or for MFIs)
# But I have not explored exactly what data manipulation is taking place.
# Therefore treat the colData with extreme caution.
#

library(edgeR)
### Loop for all clusters / labels

de.results.Y.d0vsd42 <- list()

for (i in levels(factor(summed$clusters))) {
  current <- summed[,i==summed$clusters]
  y <- DGEList(counts(current), samples=colData(current))

  discarded <- isOutlier(colSums(counts(current)), log=TRUE, type="lower")
  y <- y[,!discarded]
  y <- y[filterByExpr(y, min.count = 1, min.total.count =10, min.prop = 0.1),]
  y <- calcNormFactors(y)

  design <- try(
    model.matrix(~  factor(day), y$samples),
    silent=TRUE
  )
  if (is(design, "try-error") ||
      qr(design)$rank==nrow(design) ||
      qr(design)$rank < ncol(design))
  {
    # Skipping labels without contrasts or without
    # enough residual d.f. to estimate the dispersion.
    next
  }
```

```r
  y <- estimateDisp(y, design)
  fit <- glmQLFit(y, design)
  res <- glmQLFTest(fit, coef=ncol(design))
  de.results.Y.d0vsd42[[i]] <- res
}

#
summaries.Y.d0vsd42 <- lapply(de.results.Y.d0vsd42, FUN=function(x) summary(decideTests(x, adjust.metho


sum.tab.Y.d0vsd42 <- do.call(rbind, summaries.Y.d0vsd42)


########
# Get old cells
# Aggregate by PID + UMAP clusters
# Use edgeR to calculate differentially expressed genes within each UMAP cluster for old d0 vs old d42.
##########


summed <- sce[,sce$age == "old"] %>%
  aggregateAcrossCells(.,
                      id=DataFrame(
                        cluster=.$clusters,
                        day=.$day, samples = .$PID))

summed
```

```
## class: SingleCellExperiment
## dim: 58051 74
## metadata(0):
## assays(1): counts
## rownames(58051): ENSG00000223972 ENSG00000227232 ... ENSG00000277475
##   ENSG00000268674
## rowData names(0):
## colnames: NULL
## colData names(48): lane i5 ... day samples
## reducedDimNames(2): PCA UMAP
## spikeNames(0):
## altExpNames(0):
```

```r
#
# This combines several colData columns.
# This may be useful for index sort data (eg average cell width or for MFIs)
# But I have not explored exactly what data manipulation is taking place.
# Therefore treat the colData with extreme caution.
#

library(edgeR)
### Loop for all clusters / labels

de.results.O.d0vsd42 <- list()

for (i in levels(factor(summed$clusters))) {
  current <- summed[,i==summed$clusters]
  y <- DGEList(counts(current), samples=colData(current))

  discarded <- isOutlier(colSums(counts(current)), log=TRUE, type="lower")
  y <- y[,!discarded]
  y <- y[filterByExpr(y, min.count = 1, min.total.count =10, min.prop = 0.1),]
  y <- calcNormFactors(y)

  design <- try(
    model.matrix(~  factor(day), y$samples),
    silent=TRUE
  )
  if (is(design, "try-error") ||
      qr(design)$rank==nrow(design) ||
      qr(design)$rank < ncol(design))
  {
    # Skipping labels without contrasts or without
    # enough residual d.f. to estimate the dispersion.
    next
  }

  y <- estimateDisp(y, design)
  fit <- glmQLFit(y, design)
  res <- glmQLFTest(fit, coef=ncol(design))
  de.results.0.d0vsd42[[i]] <- res
}

#
summaries.0.d0vsd42 <- lapply(de.results.0.d0vsd42, FUN=function(x) summary(decideTests(x, adjust.metho


sum.tab.0.d0vsd42 <- do.call(rbind, summaries.0.d0vsd42)


########
# Get all cells
# Aggregate by PID, day + UMAP clusters
# Use edgeR to calculate differentially expressed genes within each UMAP cluster for day0 and day 42 (i
##########


summed <- sce %>%
  aggregateAcrossCells(.,
                       id=DataFrame(
                                  cluster=.$clusters,
                                  day=.$day, samples = .$PID))

summed
```

```
## class: SingleCellExperiment
## dim: 58051 145
## metadata(0):
## assays(1): counts
## rownames(58051): ENSG00000223972 ENSG00000227232 ... ENSG00000277475
##   ENSG00000268674
## rowData names(0):
## colnames: NULL
## colData names(48): lane i5 ... day samples
## reducedDimNames(2): PCA UMAP
## spikeNames(0):
## altExpNames(0):
```

```r
#
# This combines several colData columns.
# This may be useful for index sort data (eg average cell width or for MFIs)
# But I have not explored exactly what data manipulation is taking place.
# Therefore treat the colData with extreme caution.
#

library(edgeR)
### Loop for all clusters / labels

de.results.d42vsd0 <- list()

for (i in levels(factor(summed$clusters))) {
  current <- summed[,i==summed$clusters]
  y <- DGEList(counts(current), samples=colData(current))

  discarded <- isOutlier(colSums(counts(current)), log=TRUE, type="lower")
  y <- y[,!discarded]
  y <- y[filterByExpr(y, min.count = 1, min.total.count =10, min.prop = 0.1),]
  y <- calcNormFactors(y)

  design <- try(
    model.matrix(~  factor(day), y$samples),
    silent=TRUE
  )
  if (is(design, "try-error") ||
      qr(design)$rank==nrow(design) ||
      qr(design)$rank < ncol(design))
  {
    # Skipping labels without contrasts or without
    # enough residual d.f. to estimate the dispersion.
    next
  }

  y <- estimateDisp(y, design)
  fit <- glmQLFit(y, design)
  res <- glmQLFTest(fit, coef=ncol(design))
  de.results.d42vsd0[[i]] <- res
}

#
summaries.d42vsd0 <- lapply(de.results.d42vsd0, FUN=function(x) summary(decideTests(x, adjust.method =


sum.tab.d42vsd0 <- do.call(rbind, summaries.d42vsd0)
sum.tab.d42vsd0
```

```
##   Down NotSig Up
## 1    0   8060 44
## 2    0   5708  2
## 3    0   7767  0
## 4    0   5605  1
## 5    0  10412 12
```

```
### Overall:
sum.tab.d42vsd0
```

```
##   Down NotSig Up
## 1    0   8060 44
## 2    0   5708  2
## 3    0   7767  0
## 4    0   5605  1
## 5    0  10412 12
```

```
sum.tab.Y.d0vsd42
```

```
##   Down NotSig Up
## 1    0   2505  0
## 2    0   2593  0
## 3    0   3781  0
## 4    0   2050  0
## 5    0   6892  0
```

```
sum.tab.O.d0vsd42
```

```
##   Down NotSig Up
## 1    0   2587  0
## 2    0   2053  0
## 3    0   3029  0
## 4    0   2182  0
## 5    0   5583  0
```

```
sum.tab.d0.YvsO
```

```
##   Down NotSig Up
## 1    0    897  0
## 2    0   1144  0
## 3    0    901  0
## 4    0   1216  0
## 5    0   2315  0
```

```
sum.tab.d42.YvsO
```

```
##   Down NotSig Up
## 1    0   7486  0
## 2    0   4023  0
## 3    0   5427  0
## 4    0   2841  0
## 5    0  10062  0
```

```
library(org.Hs.eg.db)
lapply(de.results.d42vsd0, function(z) topTags(z,n = 100, p.value = 0.05)) %>%
  lapply(., function(x) {
    if(nrow(x) >0){
      mapIds(org.Hs.eg.db, keys=rownames(x),
       keytype="ENSEMBL", column="SYMBOL", multiVals = "first")}
})
```

```
## 'select()' returned 1:many mapping between keys and columns

## 'select()' returned 1:1 mapping between keys and columns
## 'select()' returned 1:1 mapping between keys and columns
## 'select()' returned 1:1 mapping between keys and columns

## $`1`
## ENSG00000095485 ENSG00000018408 ENSG00000135845 ENSG00000129347 ENSG00000110697
##       "CWF19L1"         "WWTR1"          "PIGC"          "KRI1"       "PITPNM1"
## ENSG00000142252 ENSG00000183495 ENSG00000089048 ENSG00000116704 ENSG00000005194
##        "GEMIN7"         "EP400"          "ESF1"       "SLC35D1"       "CIAPIN1"
## ENSG00000115738 ENSG00000135801 ENSG00000176438 ENSG00000166004 ENSG00000204120
##           "ID2"         "TAF5L"         "SYNE3"        "CEP295"        "GIGYF2"
## ENSG00000108344 ENSG00000261052 ENSG00000149591 ENSG00000198089 ENSG00000189007
##         "PSMD3"       "SULT1A3"         "TAGLN"          "SFI1"         "ADAT2"
## ENSG00000116863 ENSG00000176986 ENSG00000135686 ENSG00000115514 ENSG00000236675
##       "ADPRHL2"        "SEC24C"        "KLHL36"        "TXNDC9"              NA
## ENSG00000113108 ENSG00000104973 ENSG00000165609 ENSG00000133028 ENSG00000111737
##         "APBB3"         "MED25"         "NUDT5"          "SCO1"         "RAB35"
## ENSG00000162341 ENSG00000143458 ENSG00000078142 ENSG00000146282 ENSG00000168924
##         "TPCN2"        "GABPB2"        "PIK3C3"         "RARS2"         "LETM1"
## ENSG00000181666 ENSG00000102531 ENSG00000170266 ENSG00000066084 ENSG00000122884
##        "ZNF875"        "FNDC3A"          "GLB1"         "DIP2B"         "P4HA1"
## ENSG00000134899 ENSG00000164631 ENSG00000204560 ENSG00000079134
##         "ERCC5"         "ZNF12"         "DHX16"         "THOC1"
##
## $`2`
## ENSG00000196850 ENSG00000112941
##         "PPTC7"        "TENT4A"
##
## $`3`
## NULL
##
## $`4`
## ENSG00000269335
##         "IKBKG"
##
## $`5`
## ENSG00000167112 ENSG00000143851 ENSG00000174996 ENSG00000047932 ENSG00000275418
##         "TRUB2"         "PTPN7"          "KLC2"          "GOPC"              NA
## ENSG00000101187 ENSG00000171813 ENSG00000004766 ENSG00000088038 ENSG00000171466
##       "SLCO4A1"        "PWWP2B"         "VPS50"         "CNOT3"        "ZNF562"
## ENSG00000120519 ENSG00000120647
##       "SLC10A7"        "CCDC77"
```