SCE assembly for the 2016-17 cohort

EJC

25/06/2020, based on code from 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=":"))
```

Load Rsubread counts

```
load(file = "data/Rsubread_counts.RData")
```

SingleCellExperiment sample annotation

Simple annotation from the bam filename

```
# Some basic info is stored in the bam filename, which
# becomes the colname of the Rsubread count matrix:

library(dplyr)
```

```
##
## 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
```

```
annotation <- colnames(counts$counts) %>% # Remove the end of the bam filename:
gsub(., pattern = ".L001.*.", replacement = "") %>% # Split into separate strings based on the '.'
sapply(., strsplit, split = "\\.") %>% # Make a dataframe
as.data.frame(.) %>% # Re-orientate so samples are rows
t(.) %>% # Keep the first 7 columns
.[, 1:7] %>% # Ensure a dataframe
as.data.frame(.)
```

```
colnames(annotation) <- c("lane", "i5", "i7", "lib_plate", "lib_well",
    "PID", "day")
rownames(annotation) <- colnames(counts$counts)
annotation$short.name <- rownames(annotation)


#### Fix some formatting 1 plate is called 'lib' rest are 'cDNA'
annotation$lib_plate <- annotation$lib_plate %>% as.character(.) %>%
    gsub(pattern = "lib", replacement = "cDNA", .) %>% factor(.)
```

Index sort information

```
##### Add in the index sort data Read in the CSV This includes
##### age
index.flow <- read.csv(file = "ALFNA16_indexed_scB_flow_with_lib_wells_from_flowsinglecell.csv",
    row.names = 1)


# number of rows is the total number of index sorted cells.
# We sorted slightly more cells than were sequenced.
dim(index.flow)
```

```
## [1] 1070   29
```

```
# change the lib_plate factor so it shares levels with the
# corresponding factor in the 'annotation' object:
index.flow$lib_plate <- index.flow$lib.name %>% as.character(.) %>%
    gsub(pattern = "lib", replacement = "cDNA", .) %>% factor(.)

# Remove the old factor:
index.flow <- index.flow[, !colnames(index.flow) %in% c("lib.name")]

# Change column names:
colnames(index.flow) <- gsub(colnames(index.flow), pattern = "sampleID",
    replacement = "PID")  # sampleID -> PID
colnames(index.flow) <- gsub(colnames(index.flow), pattern = "age.group",
    replacement = "age")  # age
colnames(index.flow) <- gsub(colnames(index.flow), pattern = "lib.well",
    replacement = "lib_well")  # lib_well
colnames(index.flow) <- gsub(colnames(index.flow), pattern = "^name",
    replacement = "fcs_name")  # fcs filename

# Change the fluorescence columns to shorter names They all
# end with nm.A (and nothing else does) they have 15
# characters of filters + lasers that clutters plots:

colnames(index.flow)[grepl(colnames(index.flow), pattern = "nm.A$")] <- colnames(index.flow)[grepl(colna
    pattern = "nm.A$")] %>% substr(., 1, stop = nchar(.) - 15) %>%
    gsub(., pattern = "_515.30", replacement = "")

# Make 'day' formatting and class [factor] the same:
index.flow$day <- factor(paste0("d", as.character(index.flow$day)))
```

```
# remove XLoc + YLoc columns (these are duplicated in the
# fcs.XLoc / fcs.YLoc columns) XLoc / YLoc on their own could
# refer to the library well or the fcs well.  Remove now to
# minimise this risk of confusion.
index.flow <- index.flow[, !colnames(index.flow) %in% c("XLoc",
    "YLoc")]
```

Identify empty [NTC] wells

These are excluded at blind QC steps, but it is important to highlight their existence by design. Their presence in QC step skews calculations of median & MAD, so other cells also fail.

```
# Do left_join.
index.annot <- dplyr::left_join(annotation, index.flow, by.x = c("PID",
    "day", "lib_plate", "lib_well"), by.y = c("PID", "day", "lib_plate",
    "lib_well"), all.x = T, all.y = F)
```

```
## Joining, by = c("lib_plate", "lib_well", "PID", "day")
```

```
all(index.annot$short.name == annotation$short.name)  # this should be true.
```

```
## [1] TRUE
```

```
## This is important.  There are 8 NTC wells.  (in submitting
## the sequencing to Sierra [our sequencing pipeline manager],
## I gave 96 index combinations and labelled by nearest
## PID/day. This gives bam files implying a PID)#

summary(index.annot$age)  # there are 8 empty library wells, which have 'NA' as age.
```

```
##   old young  NA's
##   412   540     8
```

```
# Set the PID and day for these wells to NA to reflect their
# NTC status

index.annot[is.na(index.annot$age), ]$PID <- NA
index.annot[is.na(index.annot$age), ]$day <- NA
```

SCE building

```
library(SingleCellExperiment)
```

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

## Loading required package: S4Vectors

##
## Attaching package: 'S4Vectors'

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

## 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

## 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")'.

## Loading required package: DelayedArray

## Loading required package: matrixStats

##
## Attaching package: 'matrixStats'
```

```
## The following objects are masked from 'package:Biobase':
##
##     anyMissing, rowMedians
## The following object is masked from 'package:dplyr':
##
##     count
## Loading required package: BiocParallel
##
## Attaching package: 'DelayedArray'
## The following objects are masked from 'package:matrixStats':
##
##     colMaxs, colMins, colRanges, rowMaxs, rowMins, rowRanges
## The following objects are masked from 'package:base':
##
##     aperm, apply, rowsum
```

```
library(scater)
```

```
## Loading required package: ggplot2
```

```
library(scran)

sce <- SingleCellExperiment(assays = list(counts = counts$counts),
    colData = index.annot)
```

SCE transcript labelling

```
library(AnnotationHub)
```

```
## Loading required package: BiocFileCache
## Loading required package: dbplyr
##
## Attaching package: 'dbplyr'
## The following objects are masked from 'package:dplyr':
##
##     ident, sql
##
## Attaching package: 'AnnotationHub'
## The following object is masked from 'package:Biobase':
##
##     cache
```

```
library(ensembldb)
```

```
## Loading required package: GenomicFeatures
## Loading required package: AnnotationDbi
##
## Attaching package: 'AnnotationDbi'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## Loading required package: AnnotationFilter
```

```
##
## Attaching package: 'ensembldb'
```

```
## The following object is masked from 'package:dplyr':
##
##     filter
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
ens.hs.v38.87 <- AnnotationHub()[["AH53321"]]
```

```
## snapshotDate(): 2019-10-29
```

```
## loading from cache
```

```
## Importing File into R ..
```

```
## require("rtracklayer")
```

```
txdb <- makeTxDbFromGRanges(ens.hs.v38.87)

# Retrieve chromosomal locations of each transcript:
location <- mapIds(txdb, keys = rownames(sce), keytype = "GENEID",
    column = "TXCHROM")
```

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

```
is.mito <- which(location == "MT")


# Handy to save the is.mito object alongside the SCE
save(sce, is.mito, file = "data/SCE_incl_NTC.RData")
```

SCE quality control

```
load("data/SCE_incl_NTC.RData")
# Remove the known NTC wells these wells already have low
# counts/features so would get excluded anyway but they skew
# MAD calculations, so affect which 'borderline' cells are
# also excluded.
sce <- sce[, !is.na(sce$PID)]

# Calculate QC stats:
qcstats <- perCellQCMetrics(sce, subsets = list(Mito = is.mito))

# Define the groups WITHIN which to determine median and
# 3xMADs:
sce$phenotype <- paste(sce$age, sce$day, sep = " ")
# Identify QC fails:
```

```
batch.4grp.reasons <- quickPerCellQC(qcstats, percent_subsets = c("subsets_Mito_percent",
    "percent_top_50"), batch = sce$phenotype)


# Table to explain QC fail distribution:
colSums(as.matrix(batch.4grp.reasons))
```

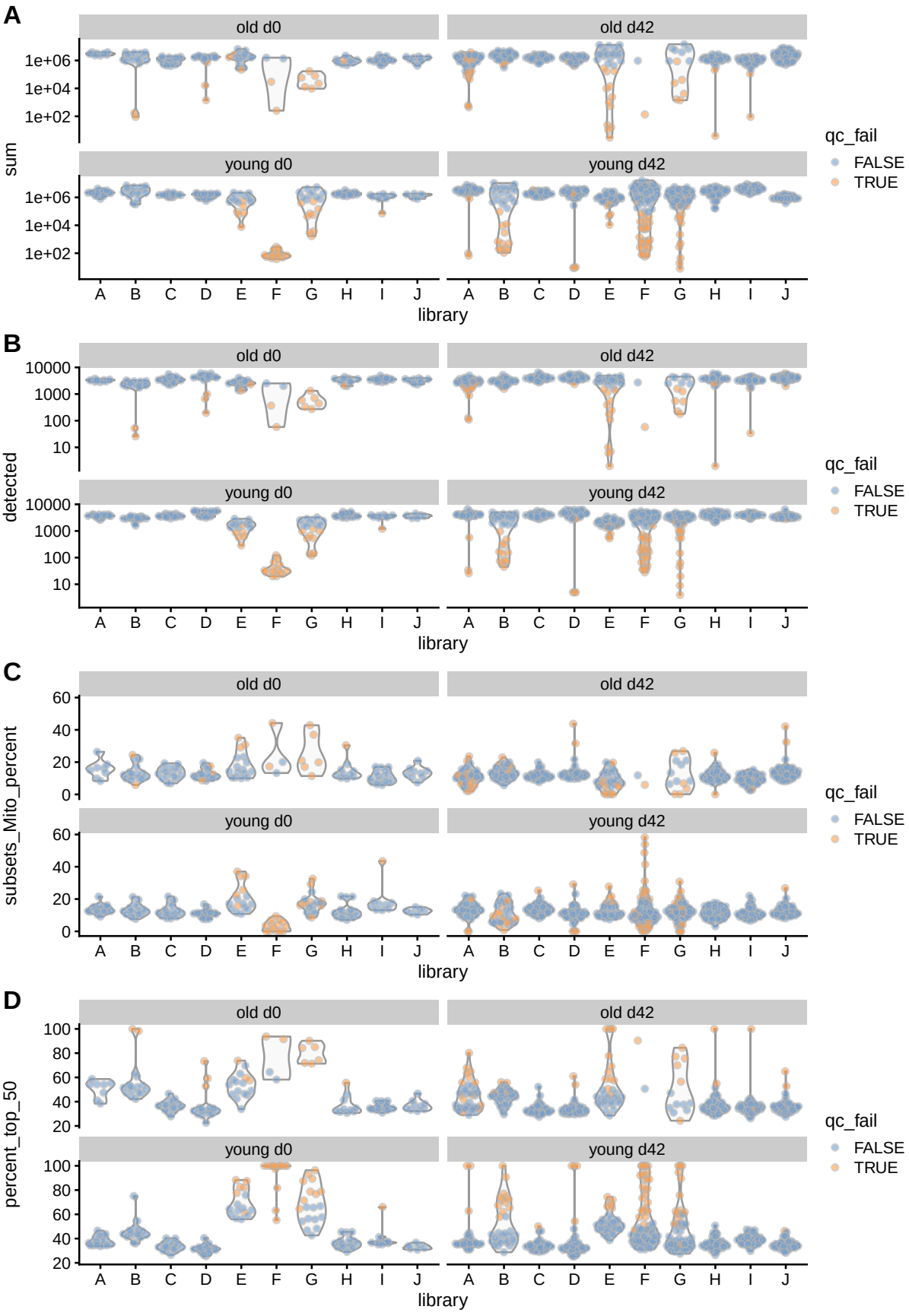
```
##              low_lib_size            low_n_features high_subsets_Mito_percent
##                       117                       130                        36
##       high_percent_top_50                   discard
##                       119                       163
```

```
# Now to plot the QC metrics, highlighting pass/fails:

sce <- addPerCellQC(sce, subsets = list(Mito = is.mito))
sce$qc_fail <- batch.4grp.reasons$discard

# Replace the long, full text 'lib_plate' with A-J:
sce$library <- LETTERS[1:10][sce$lib_plate]

cowplot::plot_grid(plotColData(sce, x = "library", y = "sum",
    colour_by = "qc_fail", other_fields = "phenotype") + facet_wrap(~phenotype) +
    scale_y_log10(), plotColData(sce, x = "library", y = "detected",
    colour_by = "qc_fail", other_fields = "phenotype") + facet_wrap(~phenotype) +
    scale_y_log10(), plotColData(sce, x = "library", y = "subsets_Mito_percent",
    colour_by = "qc_fail", other_fields = "phenotype") + facet_wrap(~phenotype),
    plotColData(sce, x = "library", y = "percent_top_50", colour_by = "qc_fail",
        other_fields = "phenotype") + facet_wrap(~phenotype),
    labels = "AUTO", align = "hv", vjust = 1, ncol = 1)
```


A
old d0
old d42
young d0
young d42
sum
1e+06
1e+04
1e+02
A B C D E F G H I J
library
qc_fail
FALSE
TRUE
B
detected
10000
1000
100
10
C
subsets_Mito_percent
60
40
20
0
D
percent_top_50
100
80
60
40
20

Save final SCE: NTC removed, QC passing cells

```
sce <- sce[, !sce$qc_fail]

save(sce, file = "data/SCE_QC_pass.RData")
```

SessionInfo

```
sessionInfo()
```

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: CentOS Linux 7 (Core)
##
## Matrix products: default
## BLAS:   /bi/apps/R/3.6.1/lib64/R/lib/libRblas.so
## LAPACK: /bi/apps/R/3.6.1/lib64/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8 LC_NUMERIC=C         LC_TIME=C
##  [4] LC_COLLATE=C         LC_MONETARY=C        LC_MESSAGES=C
##  [7] LC_PAPER=C           LC_NAME=C            LC_ADDRESS=C
## [10] LC_TELEPHONE=C       LC_MEASUREMENT=C     LC_IDENTIFICATION=C
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] rtracklayer_1.46.0          ensembldb_2.10.2
##  [3] AnnotationFilter_1.10.0     GenomicFeatures_1.38.0
##  [5] AnnotationDbi_1.48.0        AnnotationHub_2.18.0
##  [7] BiocFileCache_1.10.2        dbplyr_1.4.2
##  [9] scran_1.14.5                scater_1.14.5
## [11] ggplot2_3.3.2               SingleCellExperiment_1.8.0
## [13] SummarizedExperiment_1.16.0 DelayedArray_0.12.0
## [15] BiocParallel_1.20.0         matrixStats_0.55.0
## [17] Biobase_2.46.0              GenomicRanges_1.38.0
## [19] GenomeInfoDb_1.22.0         IRanges_2.20.1
## [21] S4Vectors_0.24.1            BiocGenerics_0.32.0
## [23] dplyr_1.0.2
##
## loaded via a namespace (and not attached):
##  [1] ggbeeswarm_0.6.0              colorspace_1.4-1
##  [3] ellipsis_0.3.0                XVector_0.26.0
##  [5] BiocNeighbors_1.4.1           farver_2.0.1
##  [7] bit64_0.9-7                   interactiveDisplayBase_1.24.0
##  [9] knitr_1.26                    Rsamtools_2.2.1
## [11] shiny_1.4.0                   BiocManager_1.30.10
## [13] compiler_3.6.1                httr_1.4.1
## [15] dqrng_0.2.1                   assertthat_0.2.1
```

```
## [17] Matrix_1.2-17                fastmap_1.0.1
## [19] lazyeval_0.2.2               limma_3.42.0
## [21] later_1.0.0                  BiocSingular_1.2.0
## [23] formatR_1.7                  htmltools_0.4.0
## [25] prettyunits_1.0.2            tools_3.6.1
## [27] rsvd_1.0.2                   igraph_1.2.4.2
## [29] gtable_0.3.0                 glue_1.4.2
## [31] GenomeInfoDbData_1.2.2       rappdirs_0.3.1
## [33] Rcpp_1.0.3                   vctrs_0.3.6
## [35] Biostrings_2.54.0            DelayedMatrixStats_1.8.0
## [37] xfun_0.11                    stringr_1.4.0
## [39] mime_0.7                     lifecycle_0.2.0
## [41] irlba_2.3.3                  statmod_1.4.32
## [43] XML_3.98-1.20                edgeR_3.28.0
## [45] zlibbioc_1.32.0              scales_1.1.0
## [47] BSgenome_1.54.0              hms_0.5.2
## [49] promises_1.1.0               ProtGenerics_1.18.0
## [51] yaml_2.2.0                   curl_4.3
## [53] memoise_1.1.0                gridExtra_2.3
## [55] biomaRt_2.42.0               stringi_1.4.3
## [57] RSQLite_2.1.4                BiocVersion_3.10.1
## [59] rlang_0.4.10                 pkgconfig_2.0.3
## [61] bitops_1.0-6                 evaluate_0.14
## [63] lattice_0.20-38              purrr_0.3.3
## [65] labeling_0.3                 GenomicAlignments_1.22.1
## [67] cowplot_1.0.0                bit_1.1-14
## [69] tidyselect_1.1.0             magrittr_1.5
## [71] R6_2.4.1                     generics_0.0.2
## [73] DBI_1.1.0                    pillar_1.4.7
## [75] withr_2.1.2                  RCurl_1.95-4.12
## [77] tibble_3.0.4                 crayon_1.3.4
## [79] rmarkdown_2.0                viridis_0.5.1
## [81] progress_1.2.2               locfit_1.5-9.1
## [83] grid_3.6.1                   blob_1.2.0
## [85] digest_0.6.23                xtable_1.8-4
## [87] httpuv_1.5.2                 openssl_1.4.1
## [89] munsell_0.5.0                beeswarm_0.2.3
## [91] viridisLite_0.3.0            vipor_0.4.5
## [93] askpass_1.1
```