smFISH analysis

Lorena Garcia-Perez (lorena.garcia-perez@crick.ac.uk)

07 October, 2018, 16:37

Contents

Set up:

Load libraries

```
library(ggplot2)
library(ggbeeswarm)
require(plyr) # I use it for mapValues in pairwise.scatter custom function
library(tidyverse)
# library(tibble) #for add_column etc
library(knitr)
library(rmarkdown)
library(pander)
library(jsonlite)
library(gridExtra)
library(LaCroixColoR)
```

Global output doc settings

```
opts_chunk$set(
  panderOptions("table.split.table", 120),
  fig.align = "center",
```

```
  fig.width = 7
)
# options(dplyr.print_min = 6, dplyr.print_max = 6)
```

Import file that loads data and contains user-defined parameters

```
wd <- getwd()
setwd("..")
parent <- getwd()

# Based on the output .csv table obtained with the Python script
file <- read.csv("./input/Diff26_c2_data.csv")

setwd(wd)

# To remove any row where the area = 0 due to clicking on a point during cell segmentation
file <- file[file$cell.area != 0, ]
pander(head(file), style = "rmarkdown")
```

channel	day	image	cell.id	cell.area	spot.id	x	y	area	max.int	mean.int
0	4	10	148	3899	1	226	258	1	1947	1947
0	4	10	148	3899	2	223.5	258	2	1979	1968
0	4	10	148	3899	3	224	256	1	2030	2030
0	4	10	148	3899	4	234	255.3	3	2421	2212
0	4	10	148	3899	5	228	255	1	2049	2049
0	4	10	148	3899	6	220	254	1	2369	2369

```
# Colors

#colourPalette <- c(
#  "#000000", "#009E73", "#e79f00", "#9ad0f3", "#0072B2", "#D55E00",
#  "#CC79A7", "#F0E442"
#)

#colourPalette <- lacroix_palette("PeachPear", type = "discrete")

gg_color_hue <- function(n) {
  hues = seq(15, 375, length = n + 1)
  hcl(h = hues, l = 65, c = 100)[1:n]
}

cols <- gg_color_hue(3)
col.custom <- c(cols[3], cols[2], cols[1])

days <- c(4, 5, 6)
days.names <- c("day 4", "day 5", "day 6")

channels <- c(0, 1, 2)
genes <- c("Irx3", "Sox2", "Olig2")

progenitors.gene <- genes[2]

# For spot selection as TX spot
MaxI.alpha <- 2 # This value should lay between 1.5 to 3
```

```
area.alpha <- 2 # This value should lay between 1.5 to 3

# With 60X objective in BX61 microscope and previous ANDOR camera:
# Pixel size = 0.1107x0.1107 µm^2 = 0.01225 µm^2
#pixel.size <- 0.01225 # µm^2 !!!!
pixel.size <- 0.2734^2 # µm^2. MAYBE THIS WAS TEH CORRECT ONE FOR BEFORE?

ploidy <- 3

# Correction factor that takes into account
# probe localization over gene:
# Irx3 reverse starnd transcribed, Sox2 and Olig2
correction.factor <- c(0.34, 0.5, 0.34)

genes.length <- c(4822, 3311, 4400) # channel 0, 1, 2 (Irx3, Sox2, Olig2)
vPol.chosen <- 2040 # bp/min, as in Bahar Halpern
```

Import file with custom functions

Custom functions for plotting

1. Themes

Chose white or dark themes as part of the plotting funtion inputs

```
basicTheme <- list(
  theme(
    plot.title = element_text(hjust = 0.5),
    plot.subtitle = element_text(hjust = 0.5, margin = margin(t = 0, r = 0, b = 0, l = 0)),
    axis.title.x = element_text(margin = margin(t = 10, r = 0, b = 0, l = 0)),
    axis.title.y = element_text(margin = margin(t = 0, r = 10, b = 0, l = 0)),
    strip.text.x = element_text(colour = "black", size = 12),
    panel.spacing = unit(1, "lines"),
    panel.background = element_rect(fill = "transparent"),
    plot.background = element_rect(fill = "transparent", colour = NA),
    legend.background = element_rect(fill = "transparent"),
    legend.box.background = element_rect(fill = "transparent"),
    strip.background = element_rect(fill = "transparent", colour = "transparent"),
    panel.grid.minor.x = element_blank(),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_line(size = 0.25, color = "gray85"),
    panel.grid.major.y = element_line(size = 0.25, color = "gray85"),
    axis.ticks=element_blank()
  )
)

whiteTheme <- list(theme())

darkTheme <- list(
  theme(
    axis.text = element_text(colour = "white"),
    axis.title = element_text(colour = "white"),
    strip.text.x = element_text(colour = "white"),
    panel.border = element_rect(color = "white")
```

```
  )
)
```

2. Box plots: RNA copy number and RNA concentration

```
box.plots <-
  function(data, point.color = "gray40", Theme = whiteTheme) {

    data$day <- as.factor(data$day)
    levels(data$day) <- days.names

    p.copy <-
      ggplot(data, aes(day, copy, group = day)) +
      geom_quasirandom(shape = 20, color = point.color,
                       size = 1.25, alpha = 0.7, stroke = 0) +
      geom_boxplot(aes(color = channel),
                   outlier.shape = NA, fill = NA, lwd = 0.5) +
      scale_color_manual(values = col.custom) +
      scale_y_continuous(expand = c(0,0),
                         breaks = seq(0, 150, 25),
                         limits = c(0, #max(data$copy)
                                                  150)) +
      labs (#subtitle = "RNA copy number",
            x = "",
            y = "RNAs / cell") +
      theme_linedraw(base_size = 12) +
      basicTheme +
      Theme +
      theme(legend.position="none",
            panel.grid.major.x = element_blank()) +
      #theme(plot.subtitle=element_text(colour = colourPalette[1])) +
      facet_wrap(~ channel)
    #print(p.copy)

    p.conc <-
      ggplot(data, aes(day, density.volume, group = day)) +
      geom_quasirandom(shape = 20, colour = point.color,
                       size = 1.25, alpha = 0.7, stroke = 0) +
      geom_boxplot(aes(color = channel),
                   outlier.shape = NA, fill = NA, lwd = 0.5) +
      scale_color_manual(values = col.custom,
                         labels = c(" ", " ", " ")) +
      scale_y_continuous(expand = c(0,0),
                         breaks = seq(0, max(data$density.volume), 0.02),
                         limits = c(0, max(data$density.volume))) +
      labs (#subtitle = "RNA concentration",
            x = "",
            y = bquote("RNAs / " ~  mu*m^3)) +
      theme_linedraw(base_size = 12) +
      basicTheme +
      Theme +
      theme(legend.position="none",
            strip.text.x = element_blank(),
            panel.grid.major.x = element_blank()) +
      #theme(plot.subtitle=element_text(colour = colourPalette[5])) +
      facet_wrap(~ channel)
```

```
    panel <- grid.arrange(p.copy, p.conc, nrow = 2, heights=c(1.15,1))
    ggsave(paste("./figures/copy_concentration_box_plots_",
                 deparse(substitute(Theme)), ".pdf", sep=""),
           panel, width = 9, height = 6)
  }

box.plot.area.volume <-
  function(data, point.color = "gray40", Theme = whiteTheme) {

    data$day <- as.factor(data$day)
    levels(data$day) <- days.names

    p.cell.area <-
      ggplot(filter(data, channel == genes[1]), aes(day, cell.area, group = day)) +
      geom_quasirandom(shape = 20, color = point.color,
                       size = 1.25, alpha = 0.7, stroke = 0) +
      geom_boxplot(aes(color = channel),
                   outlier.shape = NA, fill = NA, lwd = 0.5) +
      scale_color_manual(values = "tan1") +
      scale_y_continuous(expand = c(0,0),
                         breaks = seq(0, max(data$cell.area), 50),
                         limits = c(0, max(data$cell.area))) +
      labs (#subtitle = "Cell area"",
            x = "",
            y = bquote("Cell area [" ~  mu*m^2 ~ "]")) +
      theme_linedraw(base_size = 12) +
      basicTheme +
      Theme +
      theme(legend.position="none",
            strip.text.x = element_blank(),
            panel.grid.major.x = element_blank())

     p.cell.volume <-
      ggplot(filter(data, channel == genes[1]), aes(day, cell.volume, group = day)) +
      geom_quasirandom(shape = 20, color = point.color,
                       size = 1.25, alpha = 0.7, stroke = 0) +
      geom_boxplot(aes(color = channel),
                   outlier.shape = NA, fill = NA, lwd = 0.5) +
      scale_color_manual(values = "tan2") +
      scale_y_continuous(expand = c(0,0),
                         breaks = seq(0, max(data$cell.volume), 1000),
                         limits = c(0, max(data$cell.volume))) +
      labs (#subtitle = "Cell volume"",
            x = "",
            y = bquote("Cell volume [" ~  mu*m^3 ~ "]")) +
      theme_linedraw(base_size = 12) +
      basicTheme +
      Theme +
      theme(legend.position="none",
            strip.text.x = element_blank(),
            panel.grid.major.x = element_blank())

    panel <- grid.arrange(p.cell.area, p.cell.volume, ncol = 2)
    ggsave(paste("./figures/area_volume_box_plot_",
             deparse(substitute(Theme)), ".pdf", sep=""),
```

```
       panel, width = 6, height = 3)
}
```

3. Scatter plots: RNA copy number and RNA concentration

channel.1 and channel.2 must be supplied as strings
data must have channel as factor

```
pairwise.scatter.plot <-
  # Could try to make axis limit specific for pair of genes being plot each time
  function(data, channel.1, channel.2,
           point.color = "gray40", Theme = whiteTheme) {
    data.formatted <-
      data %>%
      mutate(day = factor(day)) %>%
      mutate(day = mapvalues(day, from = days, to = days.names))

    data.copy <-
      data.formatted %>%
      select(channel, day, image, cell.id, cell.area, copy) %>%
      spread(channel, copy)

    p.copy <-
      ggplot(data.copy, aes_string(channel.2, channel.1)) +
      geom_point(shape = 20, colour = point.color,
                 size = 1.25, alpha = 0.7, stroke = 0) +
      geom_density_2d(size = 0.25, bins = 15, color = "tan1") +
      labs(#title = "RNA copy number",
           x = bquote(atop(.(channel.2) ~ " [ RNAs / cell ]", " ")),
           y = bquote(.(channel.1) ~ " [ RNAs / cell ]")) +
      scale_x_continuous(
        #breaks = seq(0, max(data$copy), 25),
        #limits = c(0, max(data$copy)),
        expand = c(0,0)
      ) +
      scale_y_continuous(
        #breaks = seq(0, max(data$copy), 25),
        #limits = c(0, max(data$copy),
        expand = c(0,0)
      ) +
      theme_linedraw(base_size = 12) +
      basicTheme +
      Theme +
      #theme(plot.title = element_text(colour = colourPalette[1])) +
      facet_wrap(~day)

    data.conc <-
      data.formatted %>%
      select(channel, day, image, cell.id, cell.area, density) %>%
      spread(channel, density)

    p.conc <-
      ggplot(data.conc, aes_string(channel.2, channel.1)) +
      geom_point(shape = 20, colour = point.color,
                 size = 1.25, alpha = 0.7, stroke = 0) +
      geom_density_2d(size = 0.25, bins = 15, color = "tan1") +
```

```
      labs(#title = "RNA concentration",
           x = bquote(.(channel.2) ~ " [ RNAs /" ~ mu * m^2 ~ "]"),
           y = bquote(.(channel.1) ~ " [ RNAs /" ~ mu * m^2 ~ "]")) +
      scale_x_continuous(
        #breaks = seq(0, max(data$density), 1),
        #limits = c(0, max(data$density)),
        expand = c(0,0)
      ) +
      scale_y_continuous(
        #breaks = seq(0, max(data$density), 1),
        #limits = c(0, max(data$density)),
        expand = c(0,0)
      ) +
      theme_linedraw(base_size = 12) +
      basicTheme +
      Theme +
      theme(strip.text.x = element_blank()) +
      #theme(plot.title = element_text(colour = colourPalette[5])) +
      facet_wrap(~day)

    panel <- grid.arrange(p.copy, p.conc, nrow = 2, heights=c(1.25,1))
    ggsave(
      paste("./figures/copy_concentration_pairwise_scatter_plots_",
            deparse(substitute(Theme)), "_", channel.2, "_", channel.1, ".pdf",
            sep = ""),
      panel, width = 9, height = 6)
}
```

Custom functions to estimate RNA half-life

Trancription probability

```
prob.TX.fn <- function(data, day.chosen, gene.chosen, ploidy) {
  # day.chosen and gene.schosen must be input as character (for example, "4"
  # and "Olig2")
  # ploidy must be input as numeric
  day.gene.TX.subset <-
    pull(filter(data, day == day.chosen & channel == gene.chosen), TX.number)
  return(sum(day.gene.TX.subset) / (ploidy * length(day.gene.TX.subset)))
}
```

Mean RNA copy number

```
mean.copy.fn <- function(data, day.chosen, gene.chosen) {
  # day.chosen and gene.schosen must be input as character (for example, "4"
  # and "Olig2")
  day.gene.copy.subset <-
    pull(filter(data, day == day.chosen & channel == gene.chosen), copy)
  return(mean(day.gene.copy.subset))
}
```

Mean number of RNA pol = mean number of molecules at TX spot

```
mean.nPol.fn <- function(data, day.chosen, gene.chosen) {
  # day.chosen and gene.schosen must be input as character (for example, "4"
  # and "Olig2")
  day.gene.nPol.subset <-
    pull(filter(data, day == day.chosen & channel == gene.chosen), TX.mol)
```

```
  return(mean(day.gene.nPol.subset))
}
```

RNA copies, concentration and transcription per cell

RNA copy number and concentration

```
#detach("package:plyr", unload=TRUE)

copy.density.cell.nonZero <-
  file %>%
  mutate(cell.area = cell.area * pixel.size) %>% # Convert cell area in pixels to cell area in µm^2
  filter(spot.id != 0) %>%
  group_by(channel, day, image, cell.id, cell.area) %>%
  summarise(copy = n()) %>% # RNA copy number per cell
  mutate(density = round(copy / cell.area, 3)) # RNA densities in terms of RNA molecules per µm^2

copy.density.cell.Zero <-
  file %>%
  mutate(cell.area = cell.area * pixel.size) %>% # Convert cell area in pixels to cell area in µm^2
  filter(spot.id == 0) %>%
  group_by(channel, day, image, cell.id, cell.area) %>%
  summarise(copy = 0) %>% # RNA copy number per cell
  mutate(density = round(copy / cell.area, 3)) # RNA densities in terms of RNA molecules per µm^2

copy.density.cell <- full_join(copy.density.cell.nonZero, copy.density.cell.Zero)
```

```
## Joining, by = c("channel", "day", "image", "cell.id", "cell.area", "copy", "density")
```

```
copy.density.cell <-
  copy.density.cell %>%
  mutate(cell.volume = 4/3 * pi * (sqrt(cell.area/pi))^3) %>%
  mutate(density.volume = copy / cell.volume)

pander(head(copy.density.cell), style = "rmarkdown")
```

channel	day	image	cell.id	cell.area	copy	density	cell.volume	density.volume
0	4	3	36	152	7	0.046	1409	0.004967
0	4	3	37	184.6	15	0.081	1887	0.007949
0	4	3	38	118	14	0.119	963.7	0.01453
0	4	3	39	230.1	42	0.183	2625	0.016
0	4	3	40	241.7	32	0.132	2827	0.01132
0	4	3	41	200.7	13	0.065	2139	0.006078

Transcriptional parameters

To determine which spots can be considered transcription spots, we need to know the area and maximum intensity values for all the spots. Then we normalize according to each channel, day and image

```
TX.threholds <-
  file %>%
  group_by(channel, day, image) %>%
  # Choose threholds specifically for each image and channel
  summarise(
```

```
    MaxI.median = median(max.int), MaxI.mad = mad(max.int),
    area.median = median(area), area.mad = mad(area)
  ) %>%
  mutate(
    MaxI.thres = MaxI.median + 3 * MaxI.alpha * MaxI.mad,
    area.thres = area.median + 3 * area.alpha * area.mad
  )
pander(head(TX.threholds), style = "rmarkdown")
```

channel	day	image	MaxI.median	MaxI.mad	area.median	area.mad	MaxI.thres	area.thres
0	4	3	1894	282.4	2	1.483	3588	10.9
0	4	4	1547	198.7	2	1.483	2739	10.9
0	4	5	1755	296.5	2	1.483	3534	10.9
0	4	6	1727	209	2	1.483	2981	10.9
0	4	7	1454	222.4	2	1.483	2788	10.9
0	4	9	1628	247.6	2	1.483	3114	10.9

```
TX.number.cell <-
  file %>%
  group_by(channel, day, image) %>%
  left_join(TX.threholds) %>%
  mutate(is.TX.spot = max.int > MaxI.thres & area > area.thres) %>%
  group_by(channel, day, image, cell.id) %>%
  summarise(TX.number = sum(is.TX.spot))
```

```
## Joining, by = c("channel", "day", "image")
```

```
pander(head(TX.number.cell), style = "rmarkdown")
```

channel	day	image	cell.id	TX.number
0	4	3	36	0
0	4	3	37	0
0	4	3	38	0
0	4	3	39	0
0	4	3	40	0
0	4	3	41	0

To estimate how many RNA molecules are within each TX spot, we take into account the mean intensity and area of each TX spot, and divide by the median values of all the spots in the same channel, day and image. A correction factor that takes into account probe location along the RNA is also applied.

```
TX.mol <-
  file %>%
  group_by(channel, day, image) %>%
  left_join(TX.threholds) %>%
  mutate(is.TX.spot = max.int > MaxI.thres & area > area.thres) %>%
  mutate(meanInt.median = median(mean.int)) %>%
  filter(is.TX.spot == TRUE) %>%
  mutate(probe.loc = channel) %>%
  mutate(probe.loc = as.numeric(recode(probe.loc,
    "0" = correction.factor[1],
    "1" = correction.factor[2],
    "2" = correction.factor[3]
  ))) %>%
  mutate(TX.mol = mean.int * area / (probe.loc * meanInt.median * area.median))
```

```
## Joining, by = c("channel", "day", "image")
```

```
TX.mol$channel <- as.factor(TX.mol$channel)
levels(TX.mol$channel) <- genes
pander(head(TX.mol), style = "rmarkdown")
```

Table 5: Table continues below

channel	day	image	cell.id	cell.area	spot.id	x	y	area	max.int	mean.int
Irx3	5	0	1287	2372	28	195.2	316.4	5	4837	3606
Irx3	5	0	1314	1584	8	258	108.2	4	4412	3444
Irx3	5	0	1316	1436	25	140.5	107.5	4	5530	5109
Irx3	5	1	1338	1401	11	258.5	305.5	4	3615	3212
Irx3	5	1	1327	2463	19	137.3	327.3	3	3410	2680
Irx3	5	2	1415	1920	25	383.2	101.6	5	4024	3125

Table 6: Table continues below

MaxI.median	MaxI.mad	area.median	area.mad	MaxI.thres	area.thres	is.TX.spot	meanInt.median
2168	329.9	1	0	4147	1	TRUE	2145
2168	329.9	1	0	4147	1	TRUE	2145
2168	329.9	1	0	4147	1	TRUE	2145
1740	266.9	1	0	3341	1	TRUE	1710
1740	266.9	1	0	3341	1	TRUE	1710
1826	344	1	0	3890	1	TRUE	1810

probe.loc	TX.mol
0.34	24.72
0.34	18.89
0.34	28.02
0.34	22.09
0.34	13.82
0.34	25.4

Joining datasets and subsetting cell population

All cells

```
single.cell.data <-
  inner_join(copy.density.cell, TX.number.cell) %>%
  ungroup()
```

```
## Joining, by = c("channel", "day", "image", "cell.id")
```

```
single.cell.data$channel <- as.factor(single.cell.data$channel)
levels(single.cell.data$channel) <- genes

setwd(parent)
write.csv(single.cell.data, "./ouput/single_cell_data.csv", row.names=FALSE)
setwd(wd)

pander(head(single.cell.data), style = "rmarkdown")
```

channel	day	image	cell.id	cell.area	copy	density	cell.volume	density.volume	TX.number
Irx3	4	3	36	152	7	0.046	1409	0.004967	0
Irx3	4	3	37	184.6	15	0.081	1887	0.007949	0
Irx3	4	3	38	118	14	0.119	963.7	0.01453	0
Irx3	4	3	39	230.1	42	0.183	2625	0.016	0
Irx3	4	3	40	241.7	32	0.132	2827	0.01132	0
Irx3	4	3	41	200.7	13	0.065	2139	0.006078	0

```
single.cell.data.nested <-
  single.cell.data %>%
  group_by(day, image, cell.id, cell.area, channel) %>% nest() %>%
  group_by(day, image, cell.id, cell.area) %>% nest() %>%
  group_by(day) %>% nest()

myjson <- toJSON(single.cell.data.nested, pretty = TRUE, auto_unbox = TRUE)

setwd(parent)
write(myjson, "./ouput/single_cell_data_with_TX.json")
setwd(wd)

#json_data <- fromJSON(file = "./ouput/single_cell_data_with_TX.json")

# This is the more complete, non subseted, dataset
# But, if a cell has two TX spots, it will appear on two rows and so on
withTXmol.single.cell.data <-
  TX.mol %>%
  select(channel, day, image, cell.id, spot.id, TX.mol) %>%
  full_join(single.cell.data) %>%
  rename(spot.id.TX = spot.id) %>%
  arrange(channel, day, image) %>%
  replace_na(list(TX.mol = 0))

## Joining, by = c("channel", "day", "image", "cell.id")

pander(head(withTXmol.single.cell.data), style = "rmarkdown")
```

Table 9: Table continues below

channel	day	image	cell.id	spot.id.TX	TX.mol	cell.area	copy	density	cell.volume
Irx3	4	3	36	NA	0	152	7	0.046	1409
Irx3	4	3	37	NA	0	184.6	15	0.081	1887
Irx3	4	3	38	NA	0	118	14	0.119	963.7
Irx3	4	3	39	NA	0	230.1	42	0.183	2625
Irx3	4	3	40	NA	0	241.7	32	0.132	2827
Irx3	4	3	41	NA	0	200.7	13	0.065	2139

density.volume	TX.number
0.004967	0
0.007949	0
0.01453	0
0.016	0
0.01132	0
0.006078	0

```
two.alleles.data <-
  TX.mol %>%
  select(channel, day, image, cell.id, TX.mol) %>%
  group_by(channel, day, image, cell.id) %>%
  mutate(freq = n()) %>%
  ungroup() %>%
  filter(freq == 2) %>%
  select(-freq) %>%
  mutate(TX.id = rep(c(1, 2), length.out = length(.$channel))) %>%
  # Create a column with appropiate number of rows
  spread(TX.id, TX.mol) %>%
  rename(TX.id.1 = "1", TX.id.2 = "2")
two.alleles.data$channel <- as.factor(two.alleles.data$channel)
levels(two.alleles.data$channel) <- genes
pander(head(two.alleles.data), style = "rmarkdown")
```

channel	day	image	cell.id	TX.id.1	TX.id.2
Irx3	5	8	1171	32.79	23.44
Irx3	5	9	1207	28.36	22.51
Irx3	6	9	2083	29.24	26.39
Sox2	4	5	136	22.51	27.95
Sox2	4	5	146	17.37	31.55
Sox2	4	5	152	18.81	11.57

Cells expressing certain genes

single.cell.expressing.data <- single.cell.data %>% filter(density >= 1) pander(head(single.cell.expressing.data), style = "rmarkdown")

single.cell.sox2.expressing.data <- # This could be combined within single.cell.progenitors.data with filter_if()? single.cell.data %>% filter(channel == progenitors.gene) %>% filter(density >= 1) # Require sox2 density >= 1 molecule / µm^2 pander(head(single.cell.sox2.expressing.data), style = "rmarkdown")

Neural progenitor cells

This dataset will be used for posterior analysis

single.cell.progenitors.data <- # Select progenitors according to sox2 density >= 1 molecule / µm^2 semi_join(single.cell.data, single.cell.sox2.expressing.data, by = c("day", "image", "cell.id", "cell.area")) pander(head(single.cell.progenitors.data), style = "rmarkdown")

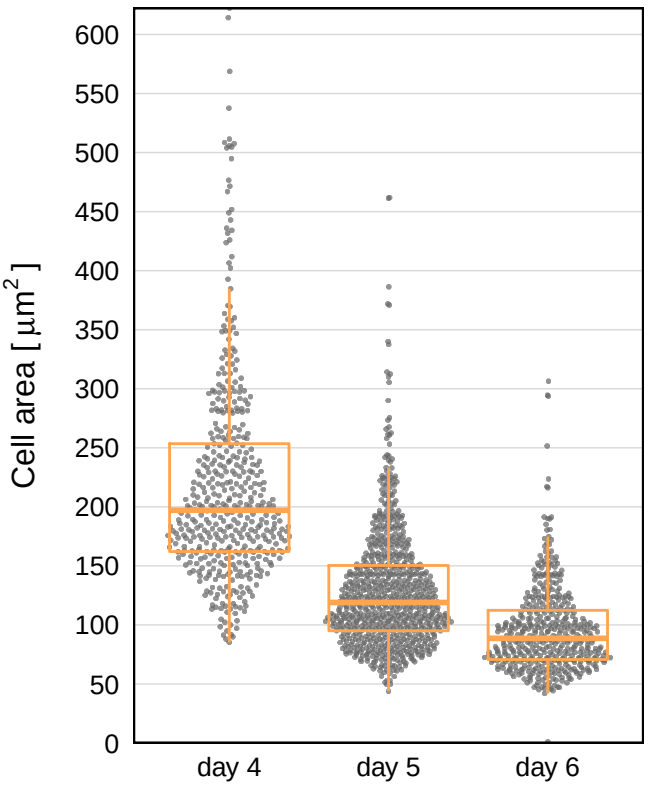
Plotting RNA copy number and concentration for progenitor cells

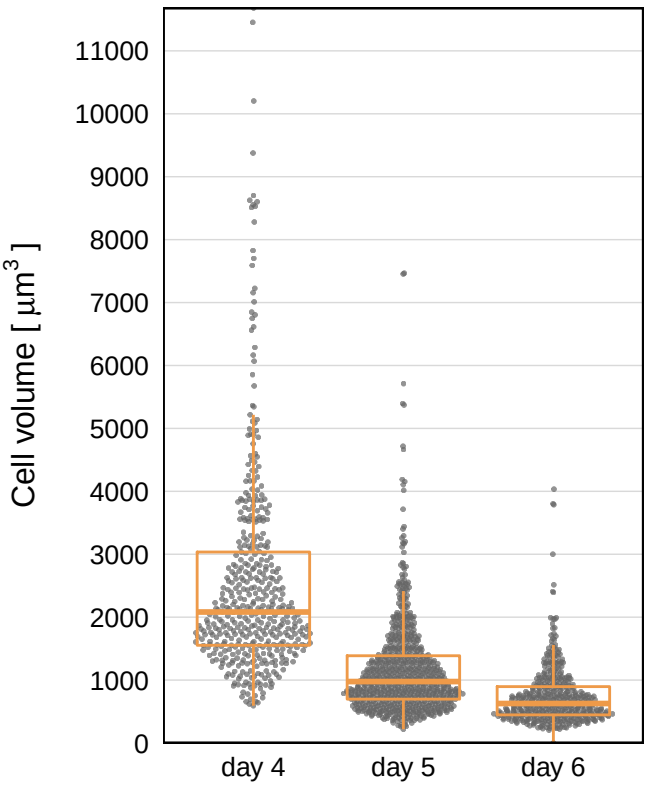
```
genes.pairwise <-
  tibble(genes) %>%
  rename(gene.1 = genes) %>%
  mutate(gene.2 = gene.1) %>%
  expand(gene.1, gene.2) %>%
  filter(gene.1 < gene.2) # To get unique combinations

# opts_knit$set(root.dir = parent)
setwd(parent)

box.plot.area.volume(single.cell.data
```

```
          #,"white",darkTheme
          )
```

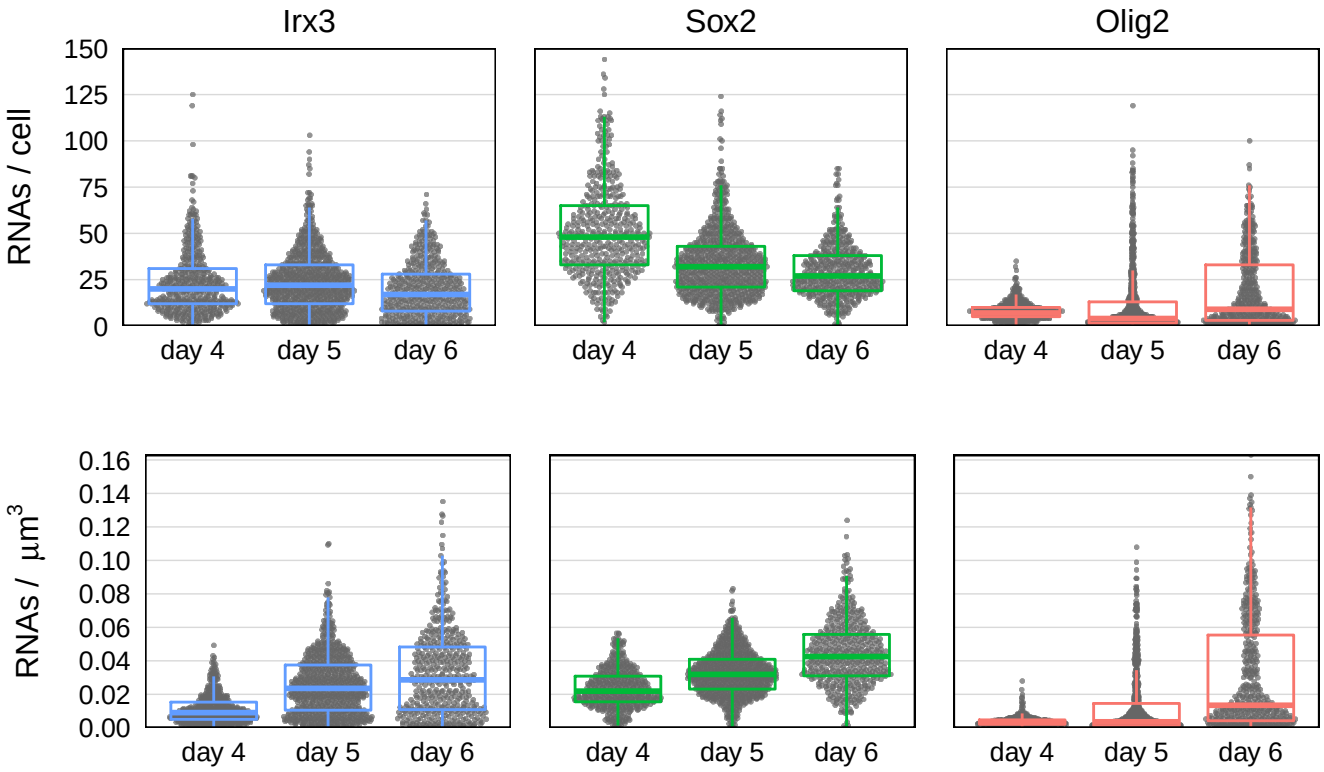





```
# copy and concentration box plots
box.plots(single.cell.data
          #,"white",darkTheme
          )
```

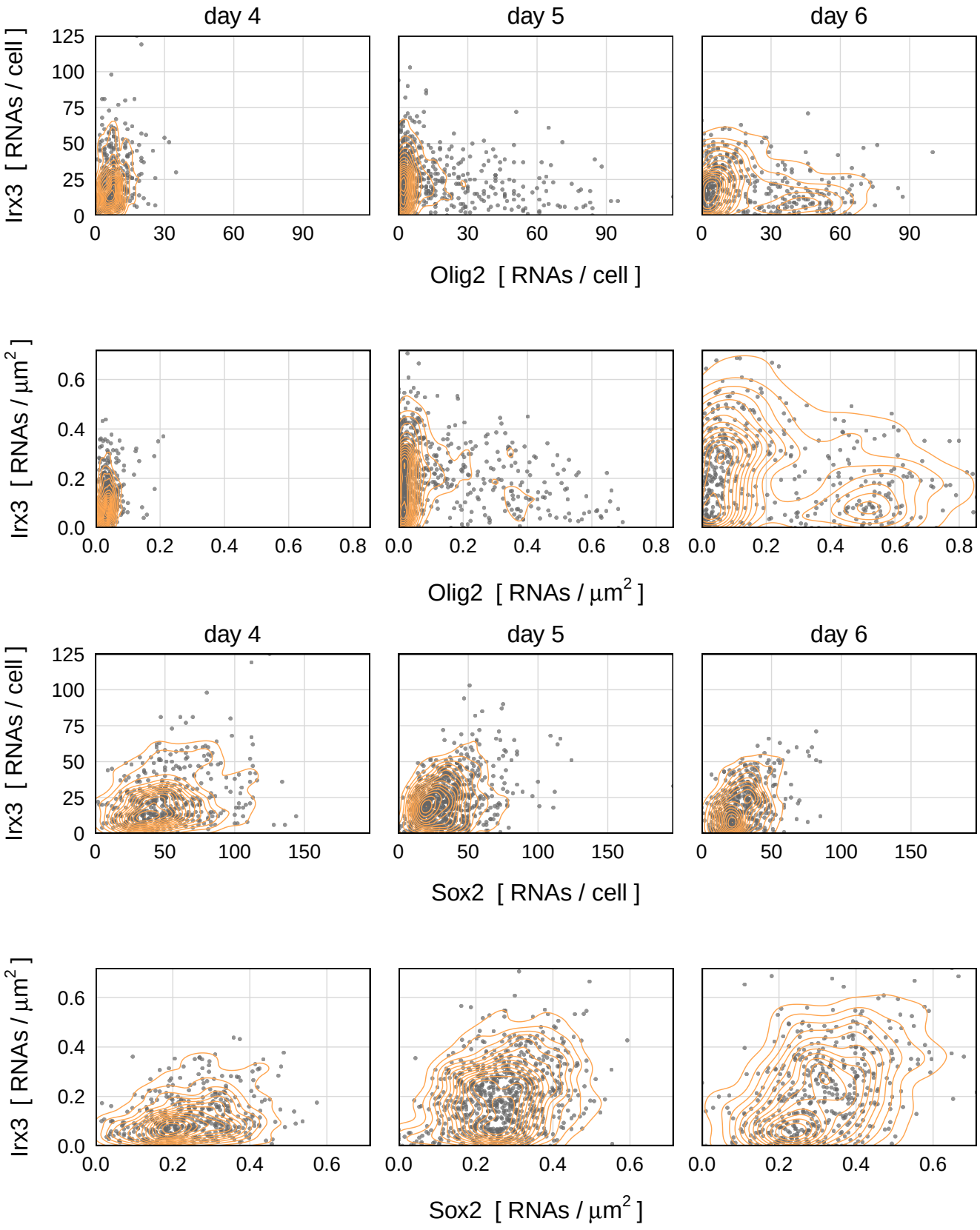
```
## Warning: Removed 1 rows containing non-finite values (stat_boxplot).
```

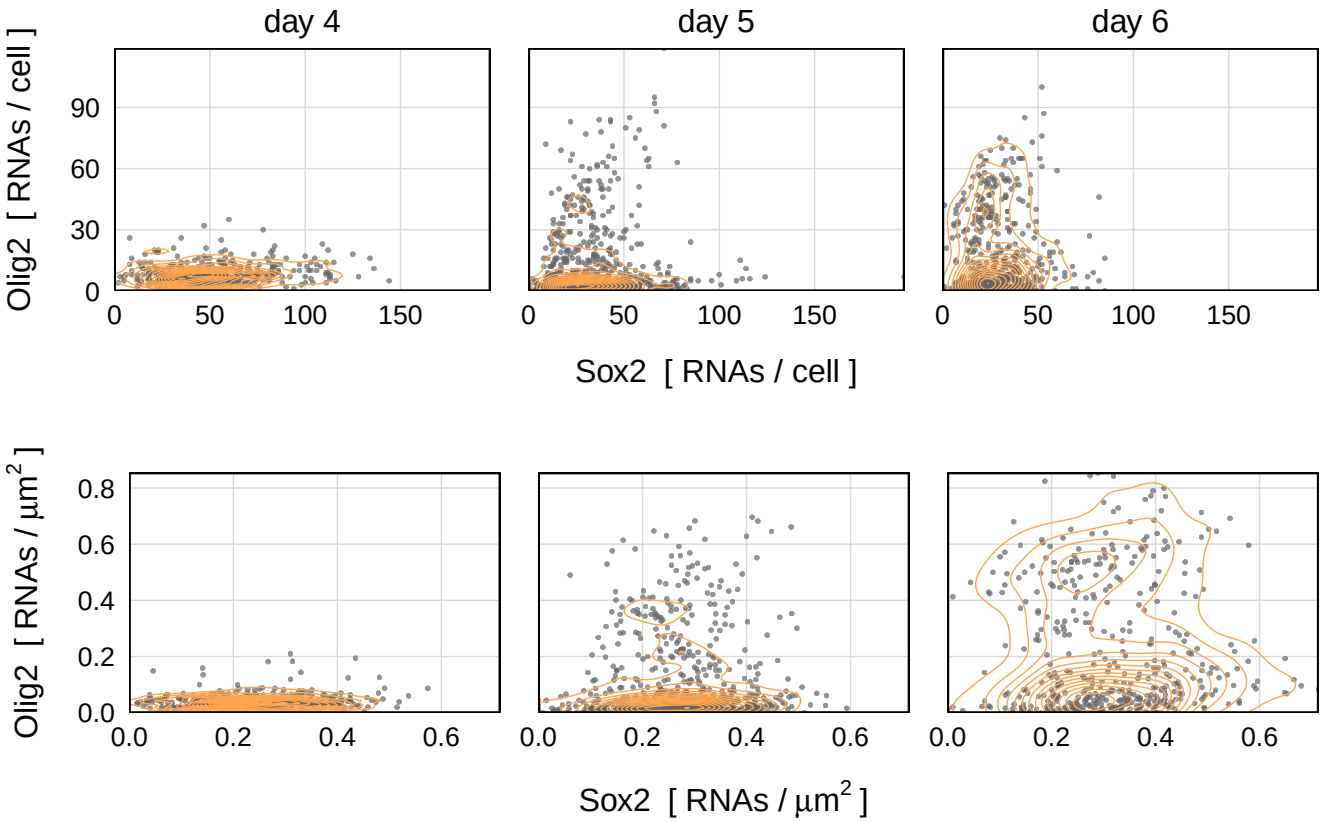
```
## Warning: Removed 1 rows containing missing values (position_quasirandom).
```




```
setwd(parent)

for (g in c(1, 2, 3)) {
  pairwise.scatter.plot(
    single.cell.data,
    genes.pairwise$gene.1[g], genes.pairwise$gene.2[g]
    #, "white", darkTheme
    )
}
```


day 4
day 5
day 6
Irx3 [RNAs / cell]
Olig2 [RNAs / cell]
Irx3 [RNAs / µm2]
Olig2 [RNAs / µm2]
Sox2 [RNAs / cell]
Sox2 [RNAs / µm2]



Calculating RNA half-life based on smFISH data

single.cell.progenitors.data is used, but results with single.cell.data are quite similar

```
mean.properties.data <-
  tibble(
    day = character(), gene = character(), prob.TX = as.numeric(),
    mean.copy = as.numeric(), mean.nPol = as.numeric()
  )

for (d in days) {
  for (g in genes) {
    mean.properties.data <-
      add_row(mean.properties.data,
        day = d, gene = g,
        prob.TX =
          round(prob.TX.fn(single.cell.data, d, g, ploidy), 3),
        mean.copy =
          round(mean.copy.fn(single.cell.data, d, g), 3),
        mean.nPol =
          round(mean.nPol.fn(TX.mol, d, g), 3)
      )
  }
}
mean.properties.data <- mean.properties.data %>% replace_na(list(mean.nPol = 0))

mean.properties.data <-
  mean.properties.data %>%
  mutate(
```

```
    vPol = vPol.chosen, gene.length = rep(genes.length, 3),
    ploidy = ploidy
  ) %>%
  mutate(TX.rate = mean.nPol * vPol / gene.length) %>%
  mutate(deg.rate = prob.TX * TX.rate * ploidy / mean.copy) %>%
  mutate(half.life = log(2) / deg.rate) %>% # in min
  arrange(gene, day)

pander(mean.properties.data, style = "rmarkdown")
```

day	gene	prob.TX	mean.copy	mean.nPol	vPol	gene.length	ploidy	TX.rate	deg.rate	half.life
4	Irx3	0	23.83	0	2040	4822	3	0	0	Inf
5	Irx3	0.014	23.96	23.27	2040	4822	3	9.846	0.01726	40.17
6	Irx3	0.032	19.86	24.41	2040	4822	3	10.33	0.04993	13.88
4	Olig2	0.007	8.152	38.7	2040	4400	3	17.94	0.04622	15
5	Olig2	0.02	11.75	33.23	2040	4400	3	15.4	0.07869	8.809
6	Olig2	0.024	18.73	41.08	2040	4400	3	19.05	0.07323	9.465
4	Sox2	0.034	51.16	18.89	2040	3311	3	11.64	0.0232	29.87
5	Sox2	0.036	34.53	16.76	2040	3311	3	10.32	0.03229	21.47
6	Sox2	0.046	29.38	17.45	2040	3311	3	10.75	0.05051	13.72

Effect of noise

```
prob.TX.pairs.data <-
  mean.properties.data %>%
  select(day, gene, prob.TX) %>%
  rename(gene.1 = gene, prob.TX.1 = prob.TX) %>%
  mutate(gene.2 = gene.1) %>%
  expand(nesting(day, gene.1, prob.TX.1), gene.2) %>%
  group_by(day) %>%
  mutate(prob.TX.2 = rep(unique(prob.TX.1), times = 3)) %>%
  filter(gene.1 < gene.2) # To get unique combinations
pander(head(prob.TX.pairs.data), style = "rmarkdown")
```

day	gene.1	prob.TX.1	gene.2	prob.TX.2
4	Irx3	0	Olig2	0.007
4	Irx3	0	Sox2	0.034
4	Olig2	0.007	Sox2	0.034
5	Irx3	0.014	Olig2	0.02
5	Irx3	0.014	Sox2	0.036
5	Olig2	0.02	Sox2	0.036

TO BE FINISHED…

for (d in days) { for (g in genes) { prob.TX.pairs.data$prob.TX.both <- add_row(prob.TX.pairs.data$prob.TX.both, prob.TX.pairs.fn())

```
prob.TX.pairs.fn <-
  function(data, day.chosen, gene.chosen.1, gene.chosen.2, ploidy) {
    # day.chosen and gene.schosen must be input as character (for example, "4"
    # and "Olig2")
    # ploidy must be input as numeric
    day.gene.TX.subset <-
```

```
    filter(data, day == day.chosen &
      (channel == gene.chosen.1 | channel == gene.chosen.2))
  dual.TX <-
    day.gene.TX.subset %>%
    select(channel, image, day, cell.id, TX.number) %>%
    filter(TX.number != 0) %>%
    group_by(day, image, cell.id) %>%
    mutate(freq = n()) %>%
    ungroup() %>%
    filter(freq == 2) %>%
    select(-freq) %>%
    spread(channel, TX.number) %>%
    return(nrow(dual.TX) / (nrow(day.gene.TX.subset)))
  # Here I do not make use of the ploidy concept...
}
```

two.alleles.data$channel <- as.factor(two.alleles.data$channel) levels(two.alleles.data$channel) <- genes

TX.probability.all[,6] <- TX.probability.all$TX.prob.pair/(TX.probability.all$TX.prob.oneC * TX.probability.all$TX.prob.otherC colnames(TX.probability.all)[6] <- c("extrinsic.noise")

To measure the influence of extrinsic noise:
THIS SHOULD BE DONE WITH A CONSTITUTIVE GENE TO COMPARE!!!

source("26_cell_area.R")

file$channel <- as.factor(file$channel) levels(file$channel) <- genes

rect <- data.frame(xmin=threshold.area, xmax=Inf, ymin=threshold.MaxInt, ymax=Inf, channel=c(0,1,2)) rect$channel <- as.factor(rect$channel) levels(rect$channel) <- genes

source("26_spot_area_maxInt.R")

source("26_TXnumber_vs.R")

source("26_barTXprob.R")

source("26_barTXprobExtrinsic.R")

source("26_boxTXmolecules.R")

source("26_TX2alleles.R")

plots <- c(hist.spots.area, hist.spots.MaxInt, p.spots.AreaVsMaxInt, p.TX.numberVsCopy, p.TX.numberVsDensity, p.TX.numberVsCellArea, bar.TX.prob, bar.TX.prob.extrinsic, box.TX.molecules, p.TX.2alleles.molecules.final)

Heterogeneity analysis

var(spots.cell$copy.C1)/mean(spots.cell$copy.C1)

#Fano factor = variance / mean = sigma^2 / mean FF.expressing <- data.frame(FF.C0=as.numeric(), FF.C1=as.numeric(), FF.C2=as.numeric()) i <- 1 for (d in days) { temp <- spots.cell.expressing[[1]] temp2 <- spots.cell.expressing[[2]] temp3 <- spots.cell.expressing[[3]] FF.expressing[i,] <- c(var(temp$copy.C0[temp$day==d]) / mean(temp$copy.C0[temp$day==d]), var(temp2$copy.C1[temp2$day==d]) / mean(temp2$copy.C1[temp2$day==d]), var(temp3$copy.C2[temp3$day==d]) / mean(temp3$copy.C2[temp3$day==d])) i <- i + 1 }

Save extracted data to file

write.csv(spots.cell, "spots_cell.csv", row.names=FALSE) write.csv(spots.cell.2, "spots_cell2.csv", row.names=FALSE)

write.csv(spots.cell.pro, "spots_cell_pro.csv", row.names=FALSE) write.csv(spots.cell.2.pro, "spots_cell2_pro.csv", row.names=FALSE)

write.csv(TX.molecules, "TX_molecules.csv", row.names=FALSE)