Representing gene/UMI counts per spot

01 June, 2021

Abstract

Here, we read in the SM-omics manuscript data and for the 'brain' data, extract the gene/umi per spot and plot

```
library(tidyverse) # Load tidyverse pacakges
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

data files needed to run are available at SCP: https://singlecell.broadinstitute.org/single_cell/study/SCP979/ please download: 10005CN48_C1_downsamp.tsv 10005CN48_D1_downsamp.tsv 10005CN48_E1_downsamp.tsv 10015CN60_E2_downsamp.tsv 10015CN84_C2_downsamp.tsv 10015CN84_D2_downsamp.tsv 10005CN48_C1_stdata_under_tissue_IDs.txt 10005CN48_D1_stdata_under_tissue_IDs.txt 10005CN48_E1_stdata_under_tissue_IDs.txt 10015CN60_E2_stdata_under_tissue_IDs.txt 10015CN84_C2_stdata_under_tissue_IDs.txt 10015CN84_D2_stdata_under_tissue_IDs.txt

```
path <- '../smomics_data/' # set path to results directory
```

Define sample information.

```
# Sample-condition mapper
cond_sm <- list('10005CN48_C1'='ST',
                '10005CN48_D1'='ST',
                '10005CN48_E1'='ST',
                '10015CN84_D2'='SM-Omics',
                '10015CN84_C2'='SM-Omics',
                '10015CN60_E2'='SM-Omics')

# Sample:seq-depth mapper
seq_depth <- list('10015CN84_D2'=23113376,
                  '10015CN60_E2'=33876251,
                  '10015CN84_C2'=28174220,
                  '10005CN48_C1'=71022006,
```

```
                  '10005CN48_D1'=75643539,
                  '10005CN48_E1'=58631758)
```

Read in tsv counts files.

```
samples_list <- c('10005CN48_C1', '10005CN48_D1', '10005CN48_E1', '10015CN84_D2', '10015CN84_C2'

files_dict <- samples_list %>% lapply(function(filename){
  read.table(file.path(path, paste0(filename, '_downsamp_stdata.tsv.gz')), row.names=1) %>% t()
})

names(files_dict) <- samples_list
```

Identify spots 'inside' tissues. Seems this can be done using the '_stdata_under_tissue_IDs.txt.gz' files and taking the first row as spots inside tissue.

```
under_tissues <- samples_list %>% lapply(function(filename){

  con <- file(file.path(path, paste0(filename, '_stdata_under_tissue_IDs.txt.gz')),"r")

  #
  if(filename=='10005CN48_E1'){ # this file requires specific handling
    first_line <- readLines(con,n=1) %>% strsplit('\t') # get first line and split by tab delimi


    spots <- first_line[[1]] %>% lapply(function(spot){
      # for each element, split into two by '_' delimiter
      spot <- strsplit(spot, '_')[[1]]

      # round first and second values
      spot1 <- round(as.numeric(spot[[1]]))
      spot2 <- round(as.numeric(spot[[2]]))

      # recombine with 'x' delimiter
      sprintf('%sx%s', spot1, spot2)
    }) %>% unlist()

  } else{
    # get first line, replace '_' with 'x' and split by tab delimiter
    first_line <- readLines(con,n=1) %>% gsub(pattern = '_', replacement = 'x') %>% strsplit('\t
    spots <- first_line[[1]]

  }

  close(con)

  return(spots)

})
```

```
names(under_tissues) <- samples_list

# check number of spots in tissues and total number of spots.
under_tissues %>% sapply(length)
```

```
## 10005CN48_C1 10005CN48_D1 10005CN48_E1 10015CN84_D2 10015CN84_C2 10015CN60_E2
##          279          271          203          235          267          276
```

```
files_dict %>% sapply(ncol)
```

```
## 10005CN48_C1 10005CN48_D1 10005CN48_E1 10015CN84_D2 10015CN84_C2 10015CN60_E2
##         1007         1007         1007         1007          997         1007
```

Define a function to extract features (genes/umis) per spot.

Note that the thresholding is slightly different to the sm-omics jupyter notebooks:

- Column thresholding is > seq-depth/1E6. This is slightly more precise than the notebooks which use approximately this value
- Row thresholding value (100) is applied to all spots rather than separately inside and outside

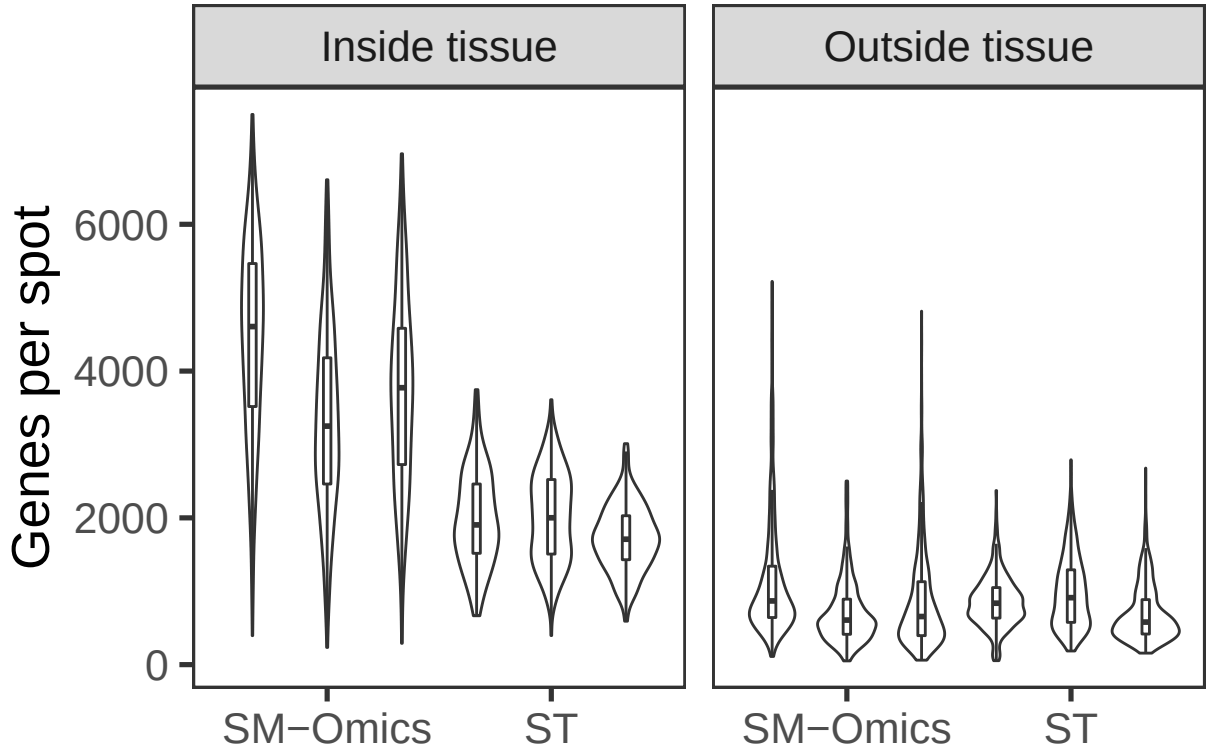
```
get_features_per_spot <- function(feature='gene', threshold=500){

  samples_list %>% lapply(function(sample){
    all_counts = files_dict[[sample]]

    # Remove spots (columns) with too small total count
    threshold <- seq_depth[[sample]]/1E6
    counts_above_thresh <- all_counts[,colSums(all_counts)>threshold]

    spots_inside <- under_tissues[[sample]]

    # remove genes (rows) with too small total count
    grouped_features <- counts_above_thresh[rowSums(counts_above_thresh)>threshold,] %>%
      data.frame() %>%
      tibble::rownames_to_column('gene_id') %>% # move rowname to column to preserve
      pivot_longer(cols=-gene_id, names_to='spot', values_to='count') %>% # pivot into longer fo
      filter(count>0) %>% # remove rows with zero count
      mutate(spot=gsub('^X', '', spot)) %>% # rename spot to remove leading 'X' from data.frame
      mutate(inside=ifelse(spot %in% spots_inside, 'Inside tissue', 'Outside tissue')) %>% # add
      group_by(spot, inside)

    if(feature=='gene'){ # just tally instances
      out <- grouped_features %>%
        tally()
    } else if(feature=='umi'){ # sum counts
      out <- grouped_features %>%
        summarise(n=sum(count))
    } else{
```

```
      stop('feature must be gene or umi')
    }

    out %>% mutate(sample_name=sample) # add sample name info

  }) %>%
    bind_rows() %>%
    mutate(condition=recode(sample_name, !!!cond_sm)) # recode sample name to make condition col
}

genes_per_spot <- get_features_per_spot('gene')
umis_per_spot <-  get_features_per_spot('umi')
```

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

Function to plot results

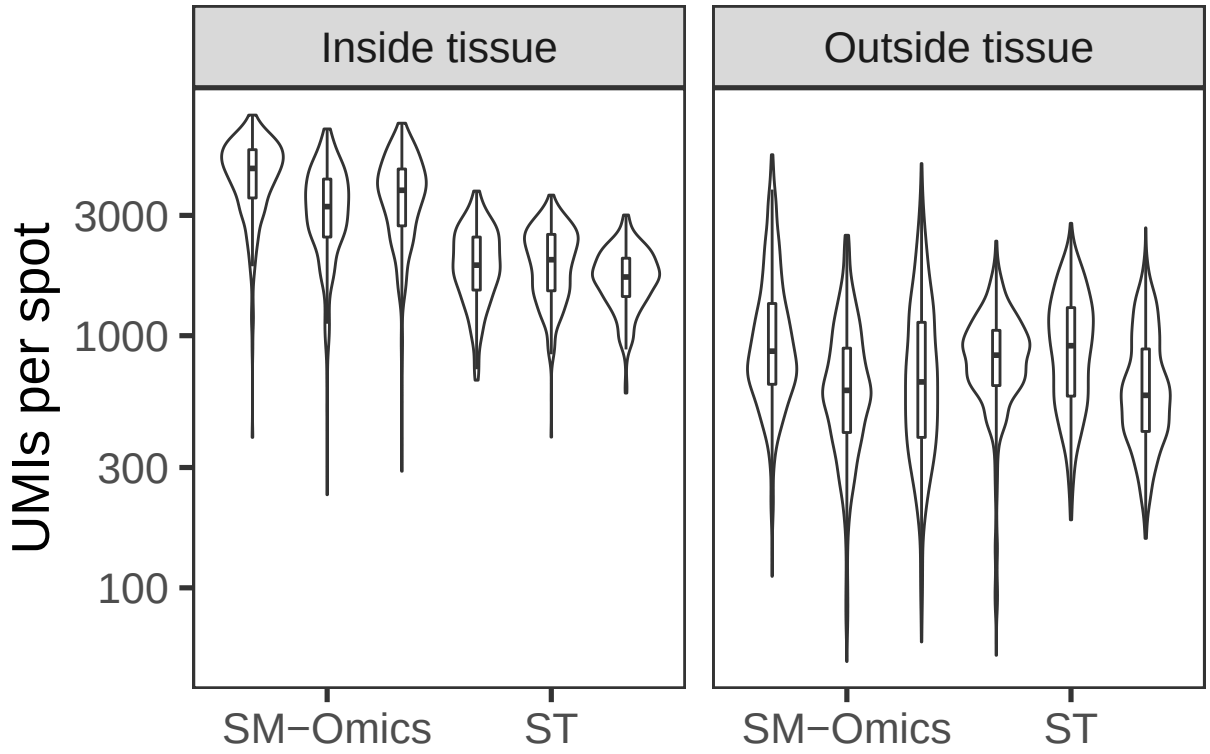
```
plot_feature_per_spot <- function(feature_per_spot){
  feature_per_spot %>%
    ggplot(aes(condition, n, group=sample_name)) +
    geom_violin(position=position_dodge(width=1)) +
    geom_boxplot(width=0.1, outlier.shape=NA, position=position_dodge(width=1)) +
    facet_wrap(~inside) +
    theme_bw(base_size=20) +
    theme(panel.grid=element_blank()) +
    xlab('')
}


p <- plot_feature_per_spot(genes_per_spot) + ylab('Genes per spot')
print(p)
```




```
ggsave('./gene_per_spot_dist.png')
```

```
## Saving 6.5 x 4.5 in image
```

```
p <- plot_feature_per_spot(genes_per_spot) + scale_y_log10() + ylab('UMIs per spot')
print(p)
```



```
ggsave('./umis_per_spot_dist.png')
```

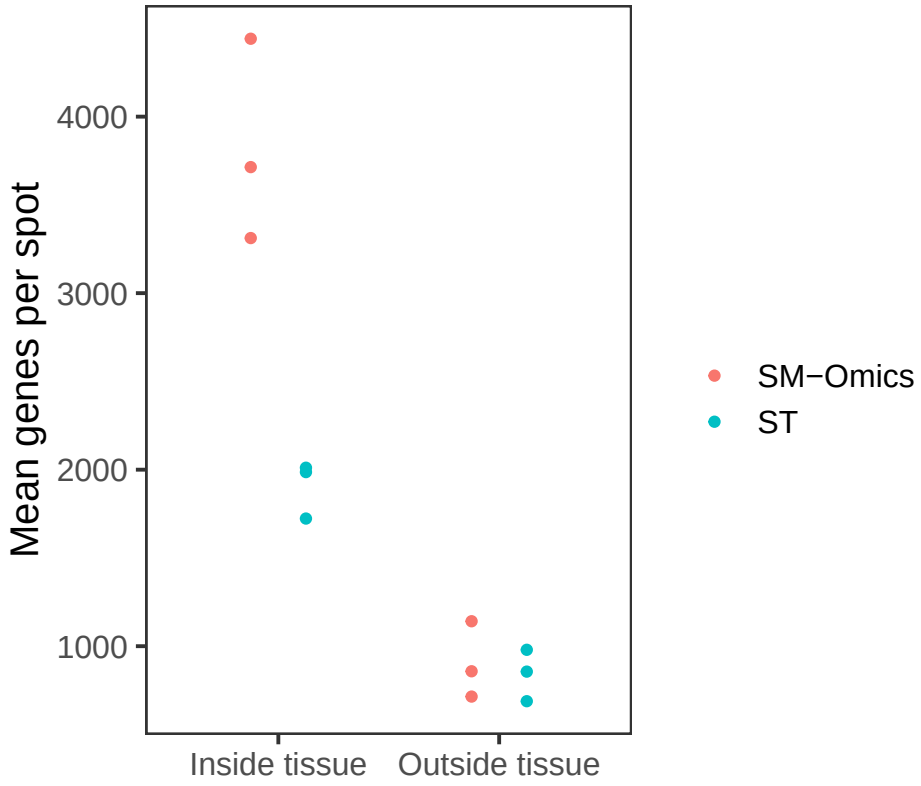
```
## Saving 6.5 x 4.5 in image
```

Plot aggregated (mean) genes per spot for comparison.

```
p <- genes_per_spot %>%
  bind_rows() %>%
  group_by(sample_name, inside) %>%
  summarise(mean_n=mean(n)) %>%
  mutate(condition=recode(sample_name, !!!cond_sm)) %>%
  ggplot(aes(inside, mean_n, colour=condition)) +
  geom_point(position=position_dodge(width=0.5)) +
  theme_bw(base_size=15) +
  theme(aspect.ratio=1.5,
        panel.grid=element_blank()) +
  scale_color_discrete(name='') +
  xlab('') +
  ylab('Mean genes per spot')
```

```
## `summarise()` has grouped output by 'sample_name'. You can override using the `.groups` argum
```

```
print(p)
```




```
ggsave('./gene_per_spot_mean.png')
```

```
## Saving 6.5 x 4.5 in image
```