HackBio: Data Viz

Qahhar

2023-10-26

Introduction

The HackBio *DataScience4Life* contest required it's participant to complete a weekly task

"In this section, you are provided with a dataset and final figures. Your task is to reproduce the figures using the dataset alone". These words mark the exact instructions to be followed for this aspect of the contest

We have also been advised to "Use only base R functions. Do not use any library or package such as ggplot2 to solve the tasks" as we proceed, yet I chose not to. Rationale?

- I am not familiar of other methods to use nor do I find them to be an efficient way to approach coding
- Only participants who have registered for the Data Science with R course hosted by HackBio are eligible to be graded/ranked. As you can guess, this doesn't include me

In that light, here is my approcach to the weekly task

Firstly:

We load in the very important package 'tidyverse' This library serves as a container for other necessary and helpful libraries in today's data science world

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Secondly:

We retrieve the dataset to be worked on

```
url <- "https://raw.githubusercontent.com/HackBio-Internship/public_datasets/main/R/datasets/Contests/V

# Create a destination and file name for the retirieved data and assign it to a variable x
x <- "/cloud/project/Hackbio/fig_One_a_e.dat"
x
```

```
## [1] "/cloud/project/Hackbio/fig_One_a_e.dat"
```

```
#Downloading the dataset, from the url to it's destination path
download.file(url, destfile = x)
```

Thirdly:

Read the retrieved data into your environment. Make sure that your working directory is the same as the destination path of the file

- To check this

```
getwd()
```

```
## [1] "/cloud/project/Hackbio"
```

To read in this file, we utilize the read.table function, given a few arguments

- x: Basically the downloaded csv
- header: This command when sets to true(T) keeps the header of any given txt file when being read in as a table or a dataframe
- sep: Every plain txt file to be read into R's console has a delimeter, The very thing that defines their structure, For our .dat file, the delimeter is a tab, thus the '' notation

```
fig_data <- read.table(x, header= T, sep = '\t')
```

Now the stage is set for the visualization task

It is good practice to convert dataframes or tables to a more convienient form for wranglingm which as 'tibbles' These tibbles are created from the library, Tibbles, and has been read in as one of the packages under the umbrella of the tidyverse

```
#Converting the table to a more convenient data holder, a Tibble
fig_tib <- as.tibble(fig_data)
```

```
## Warning: `as.tibble()` was deprecated in tibble 2.0.0.
## i Please use `as_tibble()` instead.
## i The signature and semantics have changed, see `?as_tibble`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
head(fig_tib, 5)
```

```
## # A tibble: 5 x 7
##   depth tech        TSS_enrichment Unique_nr_frag_in_re~1 X._unique_nr_frag_in~2
##   <int> <chr>                <dbl>                  <dbl>                   <dbl>
## 1     5 10xmultiome           23.8                   936.                   0.251
## 2     5 10xv1                 25.9                  1018.                   0.221
## 3     5 10xv11                20.5                  1094.                   0.256
## 4     5 10xv11c               21.6                  1770.                   0.382
## 5     5 10xv2                 23.7                  2467.                   0.514
## # i abbreviated names: 1: Unique_nr_frag_in_regions,
## #   2: X._unique_nr_frag_in_regions_in_cells
## # i 2 more variables: median_cell_type_pred_score <dbl>, fc__B_cell <dbl>
```

Dataset Previewing

Understanding the dimensions and structure of the dataset being worked with gives the analyst as good idea on what to expect as he/she goes down on the process

```
dim(fig_tib) # Returns the dimension of our tibble
```

```
## [1] 80  7
```

```
colnames(fig_tib) #This returns the no of columns in the given dataset
```

```
## [1] "depth"
## [2] "tech"
## [3] "TSS_enrichment"
## [4] "Unique_nr_frag_in_regions"
## [5] "X._unique_nr_frag_in_regions_in_cells"
## [6] "median_cell_type_pred_score"
## [7] "fc__B_cell"
```

```
glimpse(fig_tib) #Returns a glimpse view of our dataset, as the name suggests
```

```
## Rows: 80
## Columns: 7
## $ depth                                <int> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 10~
## $ tech                                 <chr> "10xmultiome", "10xv1", "10xv11"~
## $ TSS_enrichment                       <dbl> 23.822727, 25.898505, 20.499234,~
## $ Unique_nr_frag_in_regions            <dbl> 935.75, 1018.50, 1094.50, 1770.2~
## $ X._unique_nr_frag_in_regions_in_cells <dbl> 0.25137184, 0.22082997, 0.255948~
## $ median_cell_type_pred_score          <dbl> 0.5454979, 0.6108661, 0.6161470,~
## $ fc__B_cell                           <dbl> 8.155354, 7.986556, 12.019779, N~
```

```
summary(fig_data) #Returns a descriptive summary of the entire dataset
```

```
##      depth           tech           TSS_enrichment   Unique_nr_frag_in_regions
##  Min.   : 5.00   Length:80          Min.   : 3.697   Min.   :  169.5
##  1st Qu.:13.75   Class :character   1st Qu.:21.307   1st Qu.: 1457.0
##  Median :22.50   Mode  :character   Median :22.646   Median : 3211.2
##  Mean   :22.50                      Mean   :23.327   Mean   : 3593.0
##  3rd Qu.:31.25                      3rd Qu.:27.607   3rd Qu.: 4852.4
##  Max.   :40.00                      Max.   :34.824   Max.   :10765.0
##
##  X._unique_nr_frag_in_regions_in_cells median_cell_type_pred_score
##  Min.   :0.04135                       Min.   :0.3855
##  1st Qu.:0.12403                       1st Qu.:0.6106
##  Median :0.18326                       Median :0.7234
##  Mean   :0.19684                       Mean   :0.6955
##  3rd Qu.:0.25594                       3rd Qu.:0.7959
##  Max.   :0.51445                       Max.   :0.8819
##
##    fc__B_cell
##  Min.   : 1.773
##  1st Qu.: 7.551
##  Median :12.809
##  Mean   :13.846
##  3rd Qu.:20.558
##  Max.   :31.550
##  NA's   :17
```

We get a statistical overview of the dataset.

- A majority of the columns are of number types with only one column/variable being a character/string type denoted by **chr**

In order to explore the 'tech' column, we can use the tabyl from from the janitor package in R to give us a high end description of the columns values

```
library(janitor)
```

```
##
## Attaching package: 'janitor'
```

```
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
tabyl(fig_tib$tech)
```

```
##  fig_tib$tech n percent
##   10xmultiome 8     0.1
##         10xv1 8     0.1
##        10xv11 8     0.1
##       10xv11c 8     0.1
##         10xv2 8     0.1
##         ddseq 8     0.1
##        hydrop 8     0.1
##      mtscatac 8     0.1
##  mtscatacfacs 8     0.1
##        s3atac 8     0.1
```

From this, we find that the tech column has 10 unique values each of which are 8 in number, in line with the length of the dataset.

This column would be used as the basis of subsetting the dataset in visualization.

Visualization Analysis

As stated previously, the task is to replicate template graphs. Here is the link for reference

Setting up a theme e.g color, font size and whatnot saves a ton of stress

```
tech_colors <- rainbow(10) #Setting the colors of the unique labels of each group that are subjecting t
# rainbow(n) is equal to 10 because we have 10 groups
```

PROBLEM 1

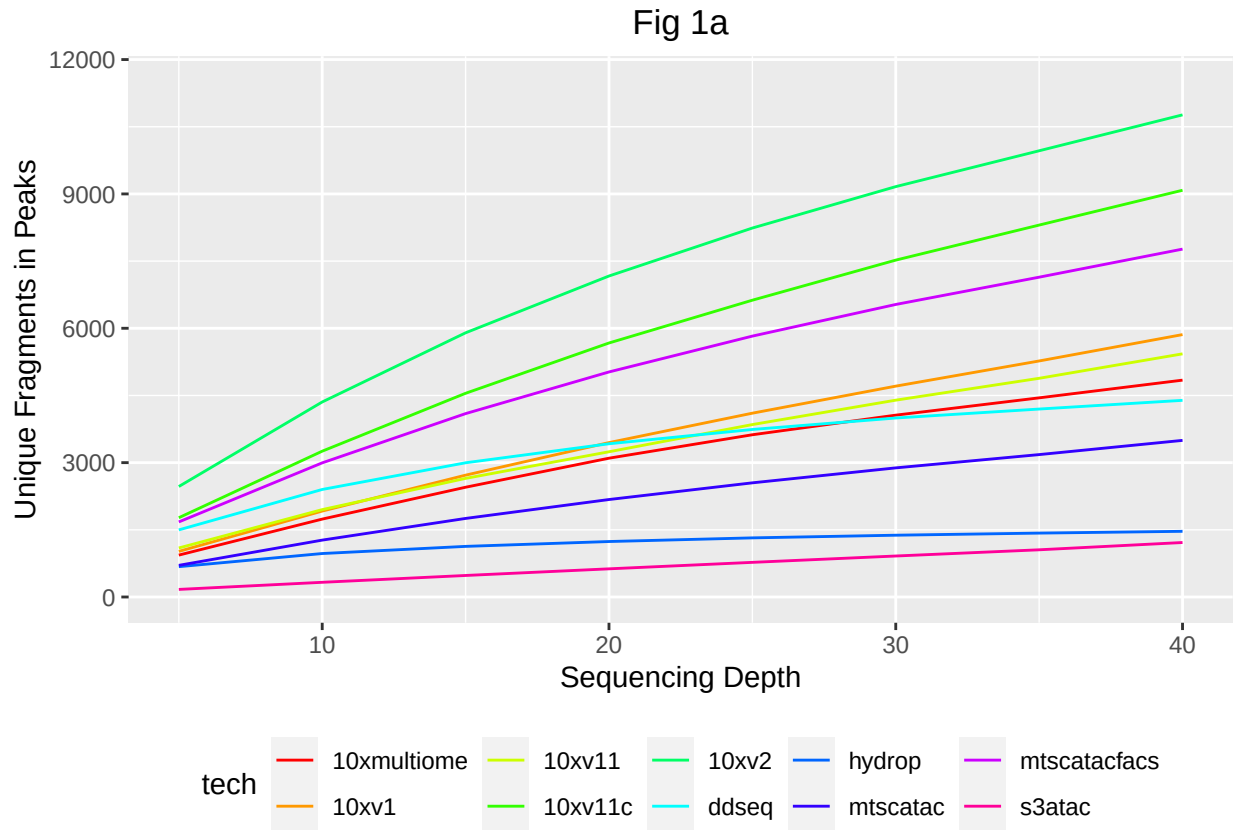
This problem seems to require the geom function (geom_line), and to understand the syntax of the argument to be passed, this was used:

```
?geom_line() # Understanding the geom function we are working on
```

Creating a plot in R using an important package 'ggplot2',under the umbrella of the tidyverse, has some basic things to define a plot

- gglot(data =) defines the plot
- aes(): This function maps what variables/columns would be plotted against one another. This then creates a grid of X and Y
- geom func : is used to determine the plot type that the data points must follow

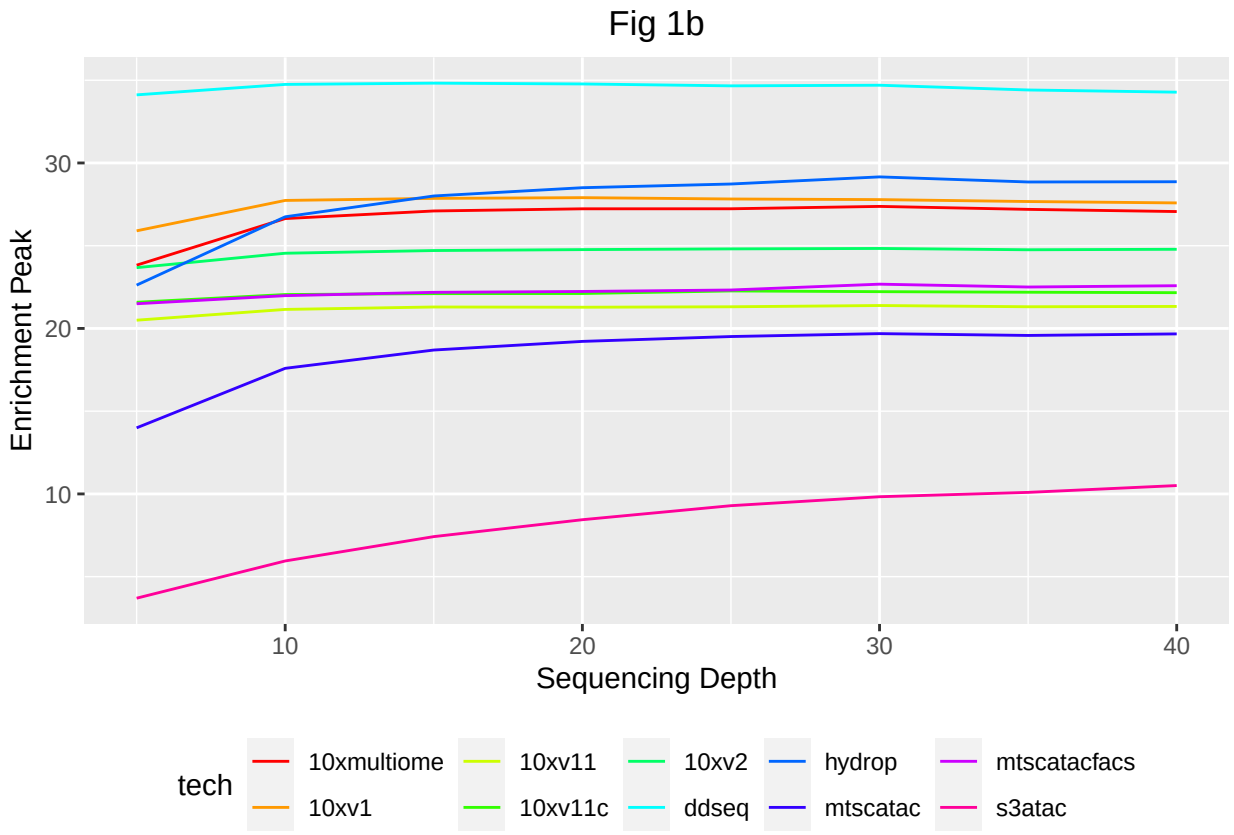
```
ggplot(fig_tib, aes(x= depth, y = Unique_nr_frag_in_regions, color = tech)) +
  geom_line() +
    scale_color_manual(values = tech_colors) + #This function states that for each line, they would be
  ylim(0, 11500)+ #y lim sets the y axis range of the plot. 12,000 was the desired peak
  labs(title = 'Fig 1a', x = "Sequencing Depth", y = "Unique Fragments in Peaks") + #the labs function
  theme(legend.position = 'bottom') + # Legend position is set to the bottom
  theme(plot.title = element_text(hjust = 0.5)) # This centralizes the plot title
```



For the next four graphs, we would use the same syntax with a difference of variables being plotted against one another

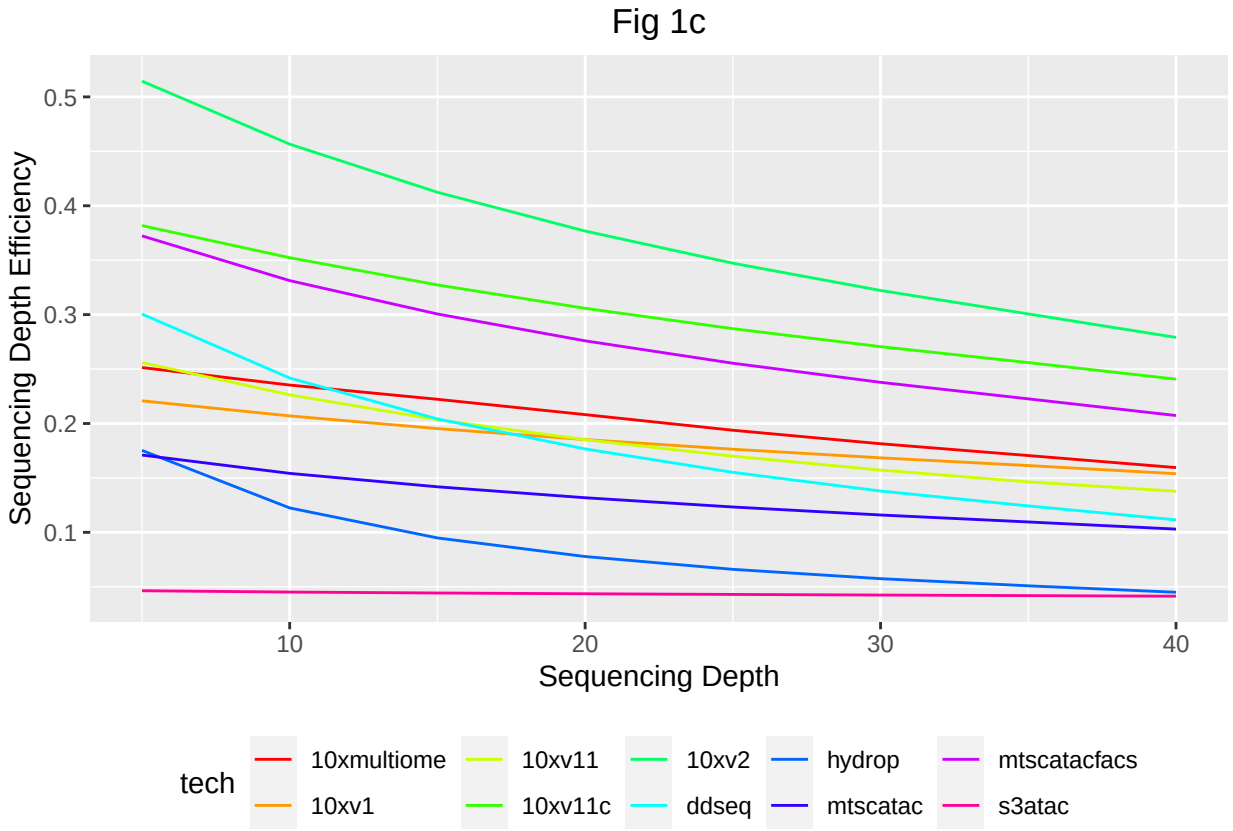
PROBLEM 2

```
ggplot(fig_tib, aes(x= depth, y = TSS_enrichment, color = tech)) +
geom_line() +
  scale_color_manual(values = tech_colors) +
  labs(title = 'Fig 1b', x = "Sequencing Depth", y = "Enrichment Peak") +
  theme(legend.position = 'bottom') +
  theme(plot.title = element_text(hjust = 0.5))
```



PROBLEM 3

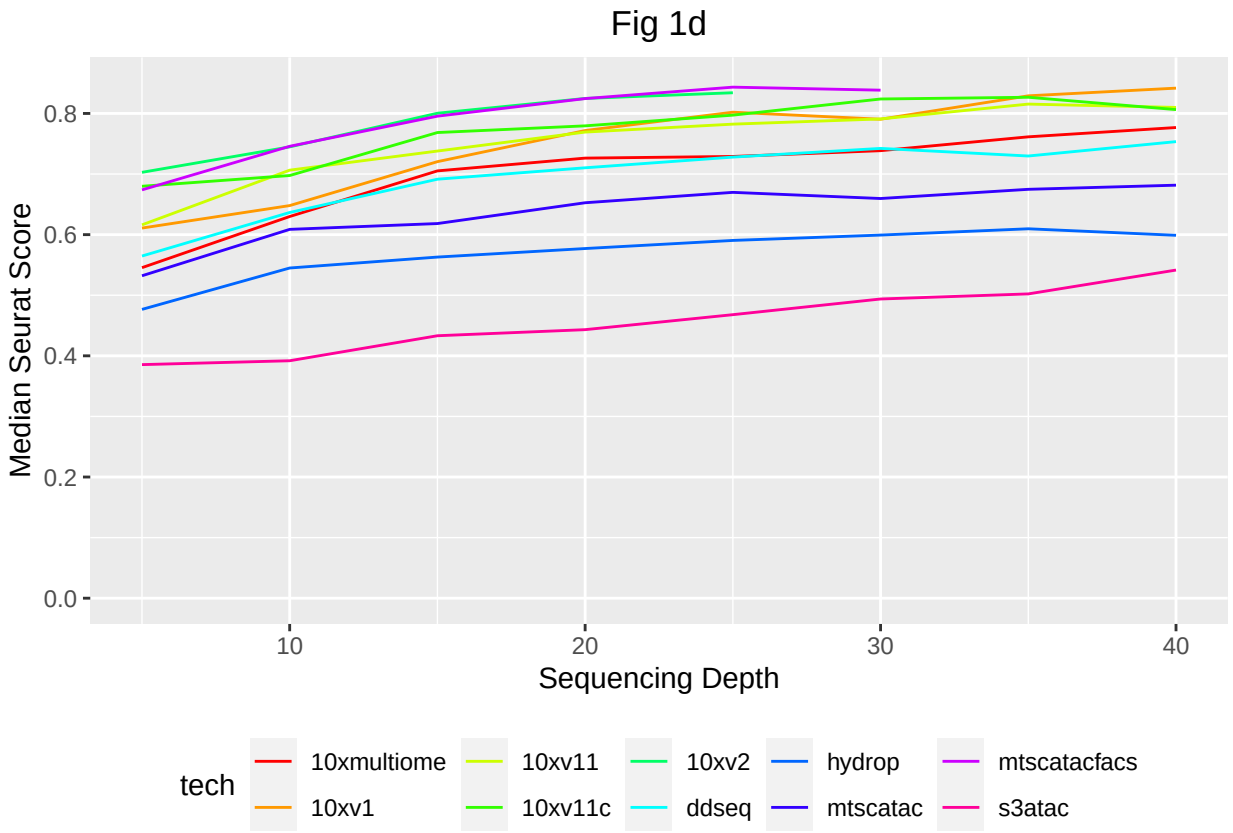
```
ggplot(fig_tib, aes(x= depth, y = X._unique_nr_frag_in_regions_in_cells, color = tech)) +
  geom_line() +
  scale_color_manual(values = tech_colors) +
  labs(title = 'Fig 1c', x = "Sequencing Depth", y = "Sequencing Depth Efficiency") +
  theme(legend.position = 'bottom') +
  theme(plot.title = element_text(hjust = 0.5)) # This centralizes the plot title
```



PROBLEM 4

```
ggplot(fig_tib, aes(x= depth, y = median_cell_type_pred_score, color = tech)) +
  geom_line() + ylim(0, 0.85) +
  scale_color_manual(values = tech_colors) +
  labs(title = 'Fig 1d', x = "Sequencing Depth", y = "Median Seurat Score") +
  theme(legend.position = 'bottom') +
  theme(plot.title = element_text(hjust = 0.5))
```

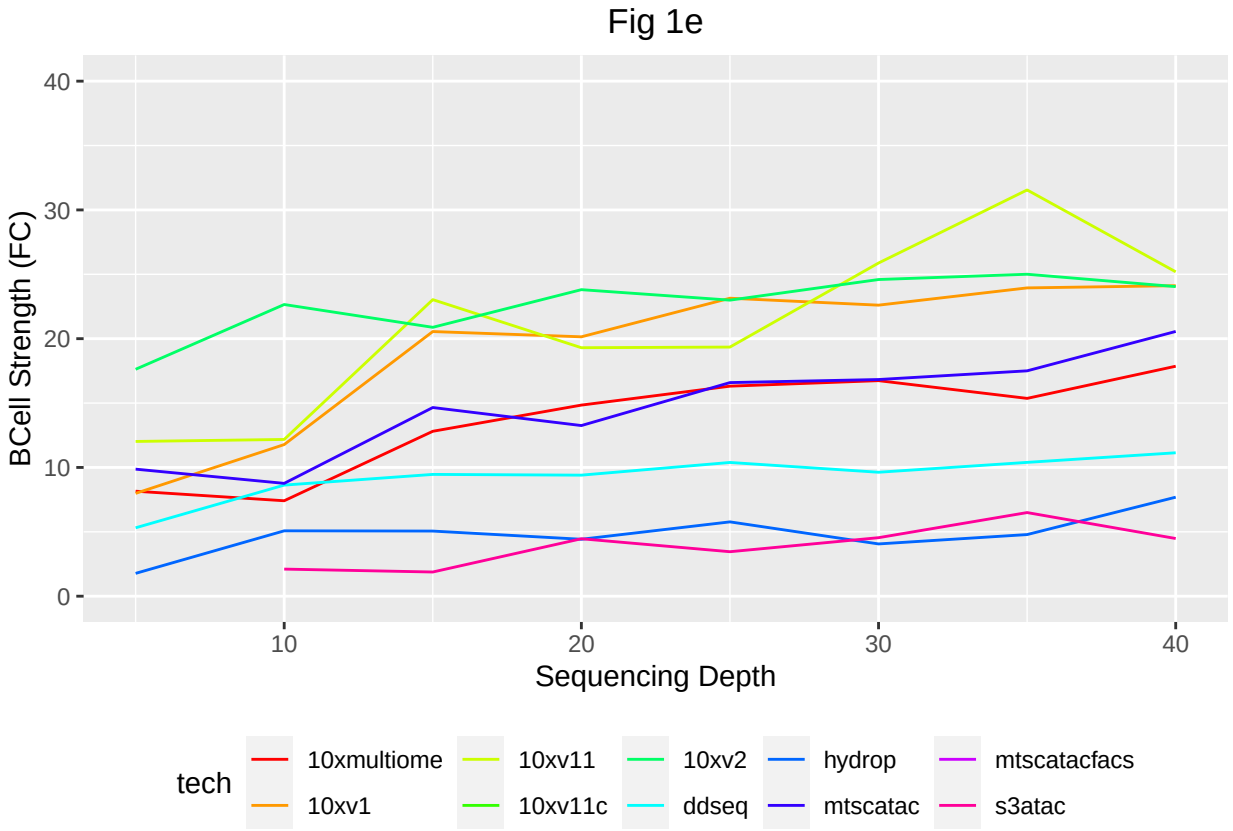
```
## Warning: Removed 5 rows containing missing values (`geom_line()`).
```



PROBLEM 5

```
ggplot(fig_tib, aes(x= depth, y = fc__B_cell, color = tech)) +
  geom_line() + ylim(0, 40) +
  scale_color_manual(values = tech_colors) +
  labs(title = 'Fig 1e', x = "Sequencing Depth", y = "BCell Strength (FC)") +
  theme(legend.position = 'bottom') +
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Warning: Removed 17 rows containing missing values (`geom_line()`).
```



Sidenote: There are various ways in which the color scale for each plot could be changed. I just chose to stick with the given instruction.

See you next week. Shalom!