Midterm

HD Sheets

2024-09-11

##DSE 5001 Mid Term, Spring 1, 2025

We will use some data from the fivethirtyeight site, and from some other sources

Prep for the exam by making sure you can do all these problems, and that you understand the content of each data set.

You may want to do some research on your own to understand these data sets a bit, and to be sure you understand the ideas in each problem.

Prepare an RMD with answers to all of these problems. You may use the RMD and other sources of information during the exam. You can look things up during the exam, but you cannot asking questions of another living person, or of LLM models like ChatGPT or Claude.

(If you want to try to ask questions of non-living person's that's your call, but most ghosts just don't know much R, as far as I can tell).

Turn in your prepared RMD with all answers prepared and annotated at the start of the exam. The preparation is 40% of the exam grade.

Then you will get 3 randomly chosen questions from this set of questions to answer on the exam during the one-hour exam window.

The questions during the live exam will be modified, so while you can (and should) cut and paste answers from your preparation RMD, you will need to modify your answers to reflect the changes in the exam question

I might alter the exam question by

a.) changing the data set b.) changing what variables I want to see plotted, or in a table c.) changing ranges of values

I will use the Quiz function in Canvas to randomly give you three problems, cut and paste these into an RMD and then use your prepared code to answer them. You can just paste the altered problems right into your prepared RMD if that helps.

I will answer questions about the problem statements, but will not tell you if you have a problem correct or not.

Most of the questions are derived from homework and/or PairProgramming examples from Modules 1 to 4

Module 5 content is not covered on this exam, the final will be Modules 5 to 7

This exam thus has a "take home preparation" portion and a "live" portion. Due to the sheer volume of material in the class and the rapid pace, this format will reward extensive preparation work, and lower the impact of work during the one hour exam window. At the same time, you do have to be able to execute the code quickly, demonstrating that you understand how it works.

Exam Time

The exam will open at 5 pm EST Feb 13 and close at 9pm Friday Feb 14, you must complete the exam in a one-hour long block within that time frame

Note: I have be available on Zoom from 7 to 9 pm EST on Thursday Feb 13 and 7-9 pm EST on Friday Feb 14, so that you can jump on zoom if you have to ask me questions. I strongly urge you to take the exam during one of the times when I am online. I cannot guarantee you will be able to ask me questions at other times during the exam window. If you feel you will want to be able to ask questions, try to plan on taking the exam between 7-9 pm EST on Thursday or Friday.

Exploratory Analysis

It is wise to do a bit of basic exploratory data analysis on these data sets, even if I don't explicitly ask for it.

```
library("fivethirtyeight")
```

```
## Some larger datasets need to be installed separately, like senators and
## house_district_forecast. To install these, we recommend you install the
## fivethirtyeightdata package by running:
## install.packages('fivethirtyeightdata', repos =
## 'https://fivethirtyeightdata.github.io/drat/', type = 'source')
```

```
library("tidyverse")
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.1     ✓ tibble    3.2.1
## ✓ lubridate 1.9.4     ✓ tidyr     1.3.1
## ✓ purrr     1.0.2
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
library("dplyr")
```

#Airline Safety, From the 538 data, Problem 1

Airline safety Data

```
head(airline_safety)
```

```
## # A tibble: 6 × 9
##   airline           incl_reg_subsidiaries avail_seat_km_per_week incidents_85_99
##   <chr>             <lgl>                                 <dbl>           <int>
## 1 Aer Lingus        FALSE                             320906734               2
## 2 Aeroflot          TRUE                             1197672318              76
## 3 Aerolineas Argen… FALSE                             385803648               6
## 4 Aeromexico        TRUE                              596871813               3
## 5 Air Canada        FALSE                            1865253802               2
## 6 Air France        FALSE                            3004002661              14
## # ℹ 5 more variables: fatal_accidents_85_99 <int>, fatalities_85_99 <int>,
## #   incidents_00_14 <int>, fatal_accidents_00_14 <int>, fatalities_00_14 <int>
```

a.) How many distinct airlines are there?

```
length(unique(airline_safety$airline))
```

```
## [1] 56
```

a.) Which airlines (top 5) had the most:

1.) incidents from 85-99?

```
airlines <- airline_safety |>
  group_by(airline) |>
  arrange(desc(incidents_85_99))

head(airlines, 5)
```

```
## # A tibble: 5 × 9
## # Groups:   airline [5]
##   airline           incl_reg_subsidiaries avail_seat_km_per_week incidents_85_99
##   <chr>             <lgl>                                 <dbl>           <int>
## 1 Aeroflot          TRUE                             1197672318              76
## 2 Ethiopian Airlin… FALSE                             488560643              25
## 3 Delta / Northwest TRUE                             6525658894              24
## 4 American          TRUE                             5228357340              21
## 5 United / Contine… TRUE                             7139291291              19
## # ℹ 5 more variables: fatal_accidents_85_99 <int>, fatalities_85_99 <int>,
## #   incidents_00_14 <int>, fatal_accidents_00_14 <int>, fatalities_00_14 <int>
```

2.) incidents from 00-14?

```
airlines <- airline_safety |>
  group_by(airline) |>
  arrange(desc(incidents_00_14))

head(airlines, 5)
```

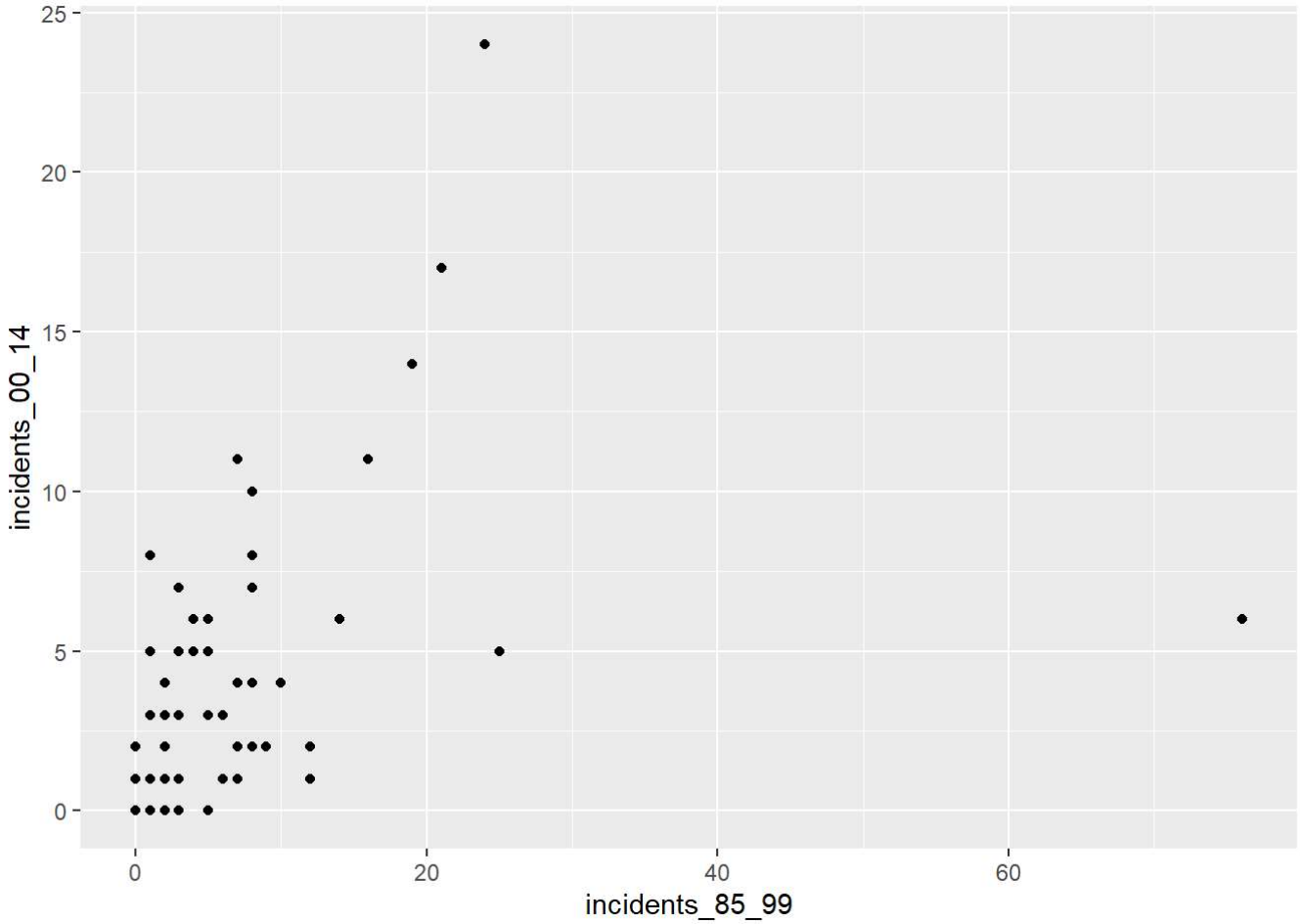
```
## # A tibble: 5 × 9
## # Groups:   airline [5]
##   airline           incl_reg_subsidiaries avail_seat_km_per_week incidents_85_99
##   <chr>             <lgl>                                 <dbl>           <int>
## 1 Delta / Northwest TRUE                             6525658894              24
## 2 American          TRUE                             5228357340              21
## 3 United / Contine… TRUE                             7139291291              19
## 4 Saudi Arabian     FALSE                             859673901               7
## 5 US Airways / Ame… TRUE                             2455687887              16
## # ℹ 5 more variables: fatal_accidents_85_99 <int>, fatalities_85_99 <int>,
## #   incidents_00_14 <int>, fatal_accidents_00_14 <int>, fatalities_00_14 <int>
```

3.) how much overlap was there of the two lists

Delta / Northwest, American, and United / Continental were all overlapping

b.) Create a graph that shows whether or not incidents from 85-99 predicts the number of incidents from 00-14. Find the correlation

```
#create a plot
ggplot(airline_safety,aes(x=incidents_85_99,y=incidents_00_14))+geom_point()
```



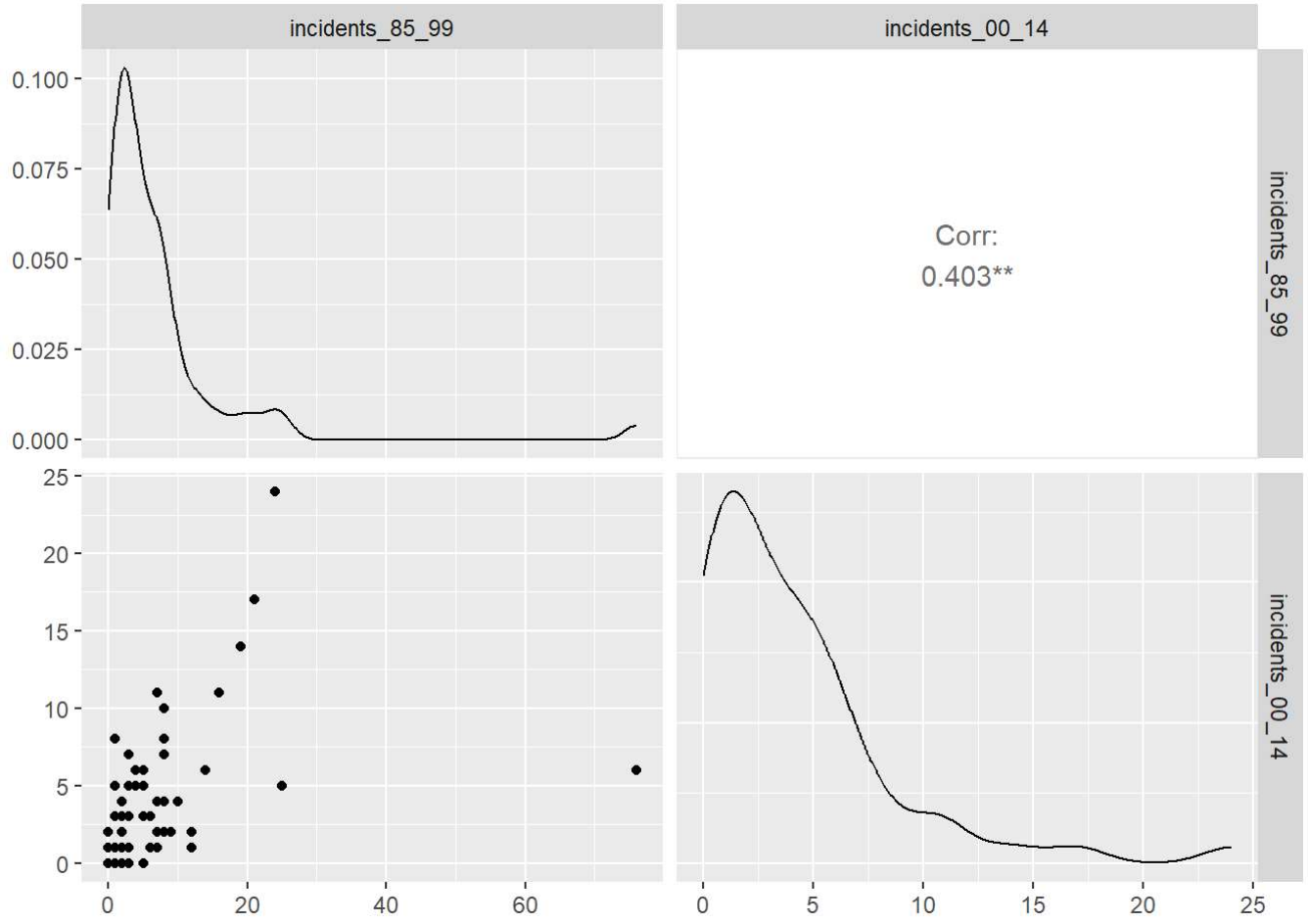

```
incidents <- airline_safety |>
  select(incidents_85_99, incidents_00_14)
incidents
```

```
## # A tibble: 56 × 2
##    incidents_85_99 incidents_00_14
##              <int>           <int>
##  1               2               0
##  2              76               6
##  3               6               1
##  4               3               5
##  5               2               2
##  6              14               6
##  7               2               4
##  8               3               5
##  9               5               5
## 10               7               4
## # ℹ 46 more rows
```

```
library('GGally')
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggpairs(incidents)
```



c.) Find a KPI or measure that computes incidents per avail_seat_km_per_week, compute this for 85_99 and 00_14

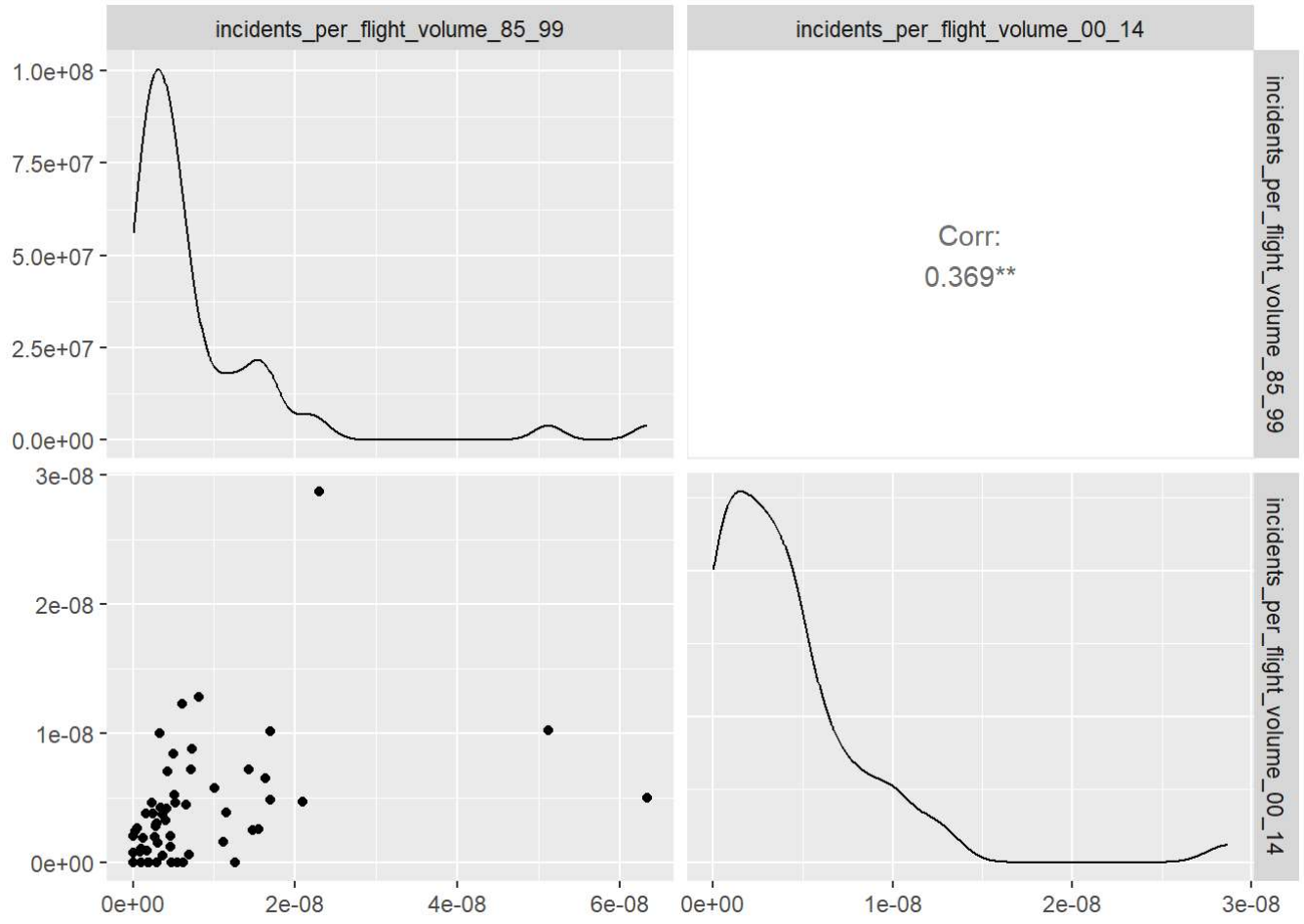
This sounds like we want to normalize incident counts based on how often an airline flies. A safer airline would have fewer incidents per available seat kilometers flown every week. The avail_seat_km_per_week variable sounds like it has the plane size data compressed into it, i.e. a larger plane with more seats has a higher avail_seat_km_per_week per kilometer flown. This is probably in an attempt to normalize across smaller private planes and larger commercial jets. Based on this analysis, the KPI I would use is incidents/avail_seat_km_per_week

```
incidents_per_flight_volume <- airline_safety |>
  mutate(incidents_per_flight_volume_85_99=(incidents_85_99/avail_seat_km_per_week)) |>
  mutate(incidents_per_flight_volume_00_14=(incidents_00_14/avail_seat_km_per_week))
incidents_per_flight_volume
```

```
## # A tibble: 56 × 11
##    airline          incl_reg_subsidiaries avail_seat_km_per_week incidents_85_99
##    <chr>            <lgl>                                 <dbl>           <int>
##  1 Aer Lingus       FALSE                              320906734               2
##  2 Aeroflot         TRUE                              1197672318              76
##  3 Aerolineas Arge… FALSE                              385803648               6
##  4 Aeromexico       TRUE                               596871813               3
##  5 Air Canada       FALSE                             1865253802               2
##  6 Air France       FALSE                             3004002661              14
##  7 Air India        TRUE                               869253552               2
##  8 Air New Zealand  TRUE                               710174817               3
##  9 Alaska Airlines  TRUE                               965346773               5
## 10 Alitalia         FALSE                              698012498               7
## # ℹ 46 more rows
## # ℹ 7 more variables: fatal_accidents_85_99 <int>, fatalities_85_99 <int>,
## #   incidents_00_14 <int>, fatal_accidents_00_14 <int>, fatalities_00_14 <int>,
## #   incidents_per_flight_volume_85_99 <dbl>,
## #   incidents_per_flight_volume_00_14 <dbl>
```

d.) Create a plot that shows whether high incidents per avail_seat_km_per_week in 85-99 predicts the number of incidents from 00-14

```
incident_correlation <- incidents_per_flight_volume |>
  select(incidents_per_flight_volume_85_99, incidents_per_flight_volume_00_14)
ggpairs(incident_correlation)
```



e.) What are the top 5 and bottom 5 airlines, based on avail_seat_km_per week?

```
top_5 <- airline_safety |>
  group_by(airline) |>
  arrange(desc(avail_seat_km_per_week))
head(top_5, 5)
```

```
## # A tibble: 5 × 9
## # Groups:   airline [5]
##   airline           incl_reg_subsidiaries avail_seat_km_per_week incidents_85_99
##   <chr>             <lgl>                                 <dbl>           <int>
## 1 United / Contine… TRUE                             7139291291              19
## 2 Delta / Northwest TRUE                             6525658894              24
## 3 American          TRUE                             5228357340              21
## 4 Lufthansa         TRUE                             3426529504               6
## 5 Southwest Airlin… FALSE                            3276525770               1
## # ℹ 5 more variables: fatal_accidents_85_99 <int>, fatalities_85_99 <int>,
## #   incidents_00_14 <int>, fatal_accidents_00_14 <int>, fatalities_00_14 <int>
```

```
bottom_5 <- airline_safety |>
  group_by(airline) |>
  arrange(avail_seat_km_per_week)
head(bottom_5, 5)
```

```
## # A tibble: 5 × 9
## # Groups:   airline [5]
##   airline         incl_reg_subsidiaries avail_seat_km_per_week incidents_85_99
##   <chr>           <lgl>                                 <dbl>           <int>
## 1 TACA            FALSE                             259373346               3
## 2 Kenya Airways   FALSE                             277414794               2
## 3 Royal Air Maroc FALSE                             295705339               5
## 4 Gulf Air        FALSE                             301379762               1
## 5 Aer Lingus      FALSE                             320906734               2
## # ℹ 5 more variables: fatal_accidents_85_99 <int>, fatalities_85_99 <int>,
## #   incidents_00_14 <int>, fatal_accidents_00_14 <int>, fatalities_00_14 <int>
```

e.) What different factors go into available_seat_km_per_week? How do different operational factors in an airline influence this measure? What specific information (ie additional data variables) would let you improve this analysis? This question is on the preparation section only.

The following factors go into available_seat_km_per_week: plane size, flight duration, flight frequency, number of available planes in the airline, and population density of the country of origin. These operational factors are all positively correlated with this measure, as they all affect the number of seats, number of flights, or distance traveled in a given week. One way to improve this analysis is to add the following variables: per capita GDP of home country (I found this in the article about the data set (https://fivethirtyeight.com/features/should-travelers-avoid-flying-airlines-that-have-had-crashes-in-the-past/)), total flights per week (from this, we could deduce the average number of km per flight), and number of staff (from this we can better understand the safety implications of under staffing airlines).

Comma Survey from 538, Problem 2

This is a set of survey results related to the “Oxford Comma”

```
head(comma_survey)
```

```
## # A tibble: 6 × 13
##   respondent_id gender age   household_income  education                location
##           <dbl> <chr>  <ord> <ord>             <ord>                    <chr>
## 1    3292953864 Male   30-44 $50,000 - $99,999 Bachelor degree          South A…
## 2    3292950324 Male   30-44 $50,000 - $99,999 Graduate degree          Mountain
## 3    3292942669 Male   30-44 <NA>              <NA>                     East No…
## 4    3292932796 Male   18-29 <NA>              Less than high school d… Middle …
## 5    3292932522 <NA>   <NA>  <NA>              <NA>                     <NA>
## 6    3292926586 Male   18-29 $25,000 - $49,999 Some college or Associa… New Eng…
## # ℹ 7 more variables: more_grammar_correct <chr>, heard_oxford_comma <lgl>,
## #   care_oxford_comma <ord>, write_following <chr>, data_singular_plural <lgl>,
## #   care_data <ord>, care_proper_grammar <ord>
```

a.) Look up what the “Oxford Comma” is and why it seems to be a concern (prepartion only)

The oxford comma a comma used prior the word “and” in a written list. For example, left, right, and center utilizes the oxford comma, but left, right and center does not. I am personally concerned with the oxford comma, and my very strong opinion is that it is a mechanism to provide important context in a list. Without the oxford comma, it could be interpreted that independent factors are related. For example, let’s say someone is describing the primary

colors of several independent images. With the oxford comma, they would list purple, orange, black, and white. This clearly delimits the color black from white, proving to the reader that each color is with respect to an individual image. Without the oxford comma, they would list orange, purple, black and white. From the reader's perspective, there is no way to determine if there are four images, 1.orange 2.purple 3.black 4.white, or three images, 1.orange 2.purple 3.black and white, and what the respective colors of those images are, unless it is explicitly stated otherwise. The oxford comma is a simple way to increase the information density of a written text, and erases ambiguity that could otherwise be detrimental.

b.) For the variable "more_grammar_correct", how many different answers are there? Make these factors

```
length(unique(comma_survey$more_grammar_correct))
```

```
## [1] 2
```

```
comma_survey <- comma_survey |>
  mutate(more_grammar_correct_as_factor=as.factor(comma_survey$more_grammar_correct))
```

c.) Which of responses to more_grammar_correct" is the Oxford commma? Create a new column that is TRUE or FALSE for "more_grammar_correct" choosing the Oxford comma.

The oxford comma response is "It's important for a person to be honest, kind, and loyal."

Source for counting character occurrences: https://www.geeksforgeeks.org/count-number-of-occurrences-of-certain-character-in-string-in-r/ (https://www.geeksforgeeks.org/count-number-of-occurrences-of-certain-character-in-string-in-r/) Source for conditional mutate: https://stackoverflow.com/questions/24459752/can-dplyr-package-be-used-for-conditional-mutating (https://stackoverflow.com/questions/24459752/can-dplyr-package-be-used-for-conditional-mutating)

```
#There are plenty of ways to do this, but I wanted to learn more about string parsing in R, so I
chose to do it this way
oxford_parsing <- comma_survey |>
  mutate(oxford_comma_parse = ifelse(lengths(regmatches(more_grammar_correct, gregexpr(",", more
_grammar_correct)))<2, FALSE, TRUE), .after = more_grammar_correct)
oxford_parsing
```

```
## # A tibble: 1,129 × 15
##    respondent_id gender age   household_income  education               location
##            <dbl> <chr>  <ord> <ord>             <ord>                   <chr>
##  1    3292953864 Male   30-44 $50,000 - $99,999 Bachelor degree         South A…
##  2    3292950324 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
##  3    3292942669 Male   30-44 <NA>              <NA>                    East No…
##  4    3292932796 Male   18-29 <NA>              Less than high school … Middle …
##  5    3292932522 <NA>   <NA>  <NA>              <NA>                    <NA>
##  6    3292926586 Male   18-29 $25,000 - $49,999 Some college or Associ… New Eng…
##  7    3292908135 Male   18-29 $0 - $24,999      Some college or Associ… Pacific
##  8    3292869879 Male   18-29 $25,000 - $49,999 Some college or Associ… East No…
##  9    3292863455 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
## 10    3292860428 Male   30-44 $150,000+         Bachelor degree         Pacific
## # ℹ 1,119 more rows
## # ℹ 9 more variables: more_grammar_correct <chr>, oxford_comma_parse <lgl>,
## #   heard_oxford_comma <lgl>, care_oxford_comma <ord>, write_following <chr>,
## #   data_singular_plural <lgl>, care_data <ord>, care_proper_grammar <ord>,
## #   more_grammar_correct_as_factor <fct>
```

```
#another cool way to do this would be to evaluate the length of the string, since the oxford com
ma string will always be longer, in this case
oxford_str_len <- comma_survey |>
  mutate(oxford_comma_str = ifelse(nchar(more_grammar_correct)<max(nchar(more_grammar_correct)),
FALSE, TRUE), .after = more_grammar_correct)
oxford_str_len
```

```
## # A tibble: 1,129 × 15
##    respondent_id gender age   household_income  education               location
##            <dbl> <chr>  <ord> <ord>             <ord>                   <chr>
##  1    3292953864 Male   30-44 $50,000 - $99,999 Bachelor degree         South A…
##  2    3292950324 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
##  3    3292942669 Male   30-44 <NA>              <NA>                    East No…
##  4    3292932796 Male   18-29 <NA>              Less than high school … Middle …
##  5    3292932522 <NA>   <NA>  <NA>              <NA>                    <NA>
##  6    3292926586 Male   18-29 $25,000 - $49,999 Some college or Associ… New Eng…
##  7    3292908135 Male   18-29 $0 - $24,999      Some college or Associ… Pacific
##  8    3292869879 Male   18-29 $25,000 - $49,999 Some college or Associ… East No…
##  9    3292863455 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
## 10    3292860428 Male   30-44 $150,000+         Bachelor degree         Pacific
## # ℹ 1,119 more rows
## # ℹ 9 more variables: more_grammar_correct <chr>, oxford_comma_str <lgl>,
## #   heard_oxford_comma <lgl>, care_oxford_comma <ord>, write_following <chr>,
## #   data_singular_plural <lgl>, care_data <ord>, care_proper_grammar <ord>,
## #   more_grammar_correct_as_factor <fct>
```

```
#lastly, the least elegant way to do this (in my opinion) is to check for a complete match again
st the more_grammar_correct variable
oxford_complete_match <- comma_survey |>
  mutate(oxford_comma_match=ifelse(more_grammar_correct_as_factor=="It's important for a person
to be honest, kind and loyal.", FALSE, TRUE), .after = more_grammar_correct)
oxford_complete_match
```

```
## # A tibble: 1,129 × 15
##    respondent_id gender age   household_income  education               location
##            <dbl> <chr>  <ord> <ord>             <ord>                   <chr>
##  1    3292953864 Male   30-44 $50,000 - $99,999 Bachelor degree         South A…
##  2    3292950324 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
##  3    3292942669 Male   30-44 <NA>              <NA>                    East No…
##  4    3292932796 Male   18-29 <NA>              Less than high school … Middle …
##  5    3292932522 <NA>   <NA>  <NA>              <NA>                    <NA>
##  6    3292926586 Male   18-29 $25,000 - $49,999 Some college or Associ… New Eng…
##  7    3292908135 Male   18-29 $0 - $24,999      Some college or Associ… Pacific
##  8    3292869879 Male   18-29 $25,000 - $49,999 Some college or Associ… East No…
##  9    3292863455 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
## 10    3292860428 Male   30-44 $150,000+         Bachelor degree         Pacific
## # ℹ 1,119 more rows
## # ℹ 9 more variables: more_grammar_correct <chr>, oxford_comma_match <lgl>,
## #   heard_oxford_comma <lgl>, care_oxford_comma <ord>, write_following <chr>,
## #   data_singular_plural <lgl>, care_data <ord>, care_proper_grammar <ord>,
## #   more_grammar_correct_as_factor <fct>
```

c.) Many other variables are really factors, some are listed as chr some are already set as ordinal. For variables that should be factors, set them as factors. Try to use NA as a valid factor entry, as the missing value may be indicative of some attitude. You may have to read up on how to do this.

I found this neat source on the mutate function: https://sparkbyexamples.com/r-programming/replace-using-dplyr-package-in-r/ (https://sparkbyexamples.com/r-programming/replace-using-dplyr-package-in-r/). I am going to replace all chr with factors, as they can all be broken into discrete categories. It seems like this implementation just works for NA as a factor entry.

```
factorized_comma_survey <- comma_survey |>
  mutate_if(is.character, as.factor)
factorized_comma_survey
```

```
## # A tibble: 1,129 × 14
##    respondent_id gender age   household_income  education               location
##            <dbl> <fct>  <ord> <ord>             <ord>                   <fct>
##  1    3292953864 Male   30-44 $50,000 - $99,999 Bachelor degree         South A…
##  2    3292950324 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
##  3    3292942669 Male   30-44 <NA>              <NA>                    East No…
##  4    3292932796 Male   18-29 <NA>              Less than high school … Middle …
##  5    3292932522 <NA>   <NA>  <NA>              <NA>                    <NA>
##  6    3292926586 Male   18-29 $25,000 - $49,999 Some college or Associ… New Eng…
##  7    3292908135 Male   18-29 $0 - $24,999      Some college or Associ… Pacific
##  8    3292869879 Male   18-29 $25,000 - $49,999 Some college or Associ… East No…
##  9    3292863455 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
## 10    3292860428 Male   30-44 $150,000+         Bachelor degree         Pacific
## # ℹ 1,119 more rows
## # ℹ 8 more variables: more_grammar_correct <fct>, heard_oxford_comma <lgl>,
## #   care_oxford_comma <ord>, write_following <fct>, data_singular_plural <lgl>,
## #   care_data <ord>, care_proper_grammar <ord>,
## #   more_grammar_correct_as_factor <fct>
```

```
#test NA as factor
test <- factorized_comma_survey |>
  group_by(gender)
test
```

```
## # A tibble: 1,129 × 14
## # Groups:   gender [3]
##    respondent_id gender age   household_income  education               location
##            <dbl> <fct>  <ord> <ord>             <ord>                   <fct>
##  1    3292953864 Male   30-44 $50,000 - $99,999 Bachelor degree         South A…
##  2    3292950324 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
##  3    3292942669 Male   30-44 <NA>              <NA>                    East No…
##  4    3292932796 Male   18-29 <NA>              Less than high school … Middle …
##  5    3292932522 <NA>   <NA>  <NA>              <NA>                    <NA>
##  6    3292926586 Male   18-29 $25,000 - $49,999 Some college or Associ… New Eng…
##  7    3292908135 Male   18-29 $0 - $24,999      Some college or Associ… Pacific
##  8    3292869879 Male   18-29 $25,000 - $49,999 Some college or Associ… East No…
##  9    3292863455 Male   30-44 $50,000 - $99,999 Graduate degree         Mountain
## 10    3292860428 Male   30-44 $150,000+         Bachelor degree         Pacific
## # ℹ 1,119 more rows
## # ℹ 8 more variables: more_grammar_correct <fct>, heard_oxford_comma <lgl>,
## #   care_oxford_comma <ord>, write_following <fct>, data_singular_plural <lgl>,
## #   care_data <ord>, care_proper_grammar <ord>,
## #   more_grammar_correct_as_factor <fct>
```

d.) Create a table that shows the counts of Male, Female and NA values by whether or not the

d1.) Had heard of the Oxford Comma

```
heard_by_gender <- factorized_comma_survey |>
  group_by(gender) |>
  summarize(
    heard_count = sum(heard_oxford_comma==TRUE, na.rm=TRUE),
    not_heard_count = sum(heard_oxford_comma==FALSE, na.rm=TRUE)
            )
heard_by_gender
```

```
## # A tibble: 3 × 3
##   gender heard_count not_heard_count
##   <fct>        <int>           <int>
## 1 Female         333             215
## 2 Male           289             200
## 3 <NA>            33              29
```

d2.) Preferred the Oxford Comma (using your answer/results from C)

```
preferred_by_gender <- oxford_parsing |>
  mutate_if(is.character, as.factor) |>
  group_by(gender) |>
  summarize(
    preferred_count = sum(oxford_comma_parse==TRUE, na.rm=TRUE),
    not_preferred_count = sum(oxford_comma_parse==FALSE, na.rm=TRUE)
            )
preferred_by_gender
```

```
## # A tibble: 3 × 3
##   gender preferred_count not_preferred_count
##   <fct>            <int>               <int>
## 1 Female             314                 234
## 2 Male               280                 209
## 3 <NA>                47                  45
```

Explain what this tabular data means.

Across the three groups, it appears that men, women, and the NA category all prefer the oxford comma. Interestingly, at first glance, this also looks to be proportional to whether or not each group has heard of the oxford comma.

e.) Add the location to your table in part d.

```
heard_by_gender <- factorized_comma_survey |>
  group_by(gender, location) |>
  summarize(
    heard_count = sum(heard_oxford_comma==TRUE, na.rm=TRUE),
    not_heard_count = sum(heard_oxford_comma==FALSE, na.rm=TRUE)
            )
```

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

```
heard_by_gender
```

```
## # A tibble: 21 × 4
## # Groups:   gender [3]
##    gender location           heard_count not_heard_count
##    <fct>  <fct>                    <int>           <int>
##  1 Female East North Central          62              36
##  2 Female East South Central          15               6
##  3 Female Middle Atlantic             39              33
##  4 Female Mountain                    31              11
##  5 Female New England                 23              18
##  6 Female Pacific                     55              35
##  7 Female South Atlantic              50              39
##  8 Female West North Central          33              17
##  9 Female West South Central          23              18
## 10 Female <NA>                         2               2
## # ℹ 11 more rows
```

```
preferred_by_gender <- oxford_parsing |>
  mutate_if(is.character, as.factor) |>
  group_by(gender,location) |>
  summarize(
    preferred_count = sum(oxford_comma_parse==TRUE, na.rm=TRUE),
    not_preferred_count = sum(oxford_comma_parse==FALSE, na.rm=TRUE)
            )
```

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

```
preferred_by_gender
```

```
## # A tibble: 21 × 4
## # Groups:   gender [3]
##    gender location           preferred_count not_preferred_count
##    <fct>  <fct>                        <int>               <int>
##  1 Female East North Central              64                  34
##  2 Female East South Central              12                   9
##  3 Female Middle Atlantic                 35                  37
##  4 Female Mountain                        23                  19
##  5 Female New England                     23                  18
##  6 Female Pacific                         49                  41
##  7 Female South Atlantic                  57                  32
##  8 Female West North Central              25                  25
##  9 Female West South Central              24                  17
## 10 Female <NA>                             2                   2
## # ℹ 11 more rows
```

Explain what this table means.

Each table breaks down the respective counts for men, women, and NA by region in the US, giving the viewer a top level overview of how gender, region, or both affect the counts.

#Bechdel test results,Problem 3 EXAM

https://en.wikipedia.org/wiki/Bechdel_test (https://en.wikipedia.org/wiki/Bechdel_test)

This is a data set that looks at whether movies with higher performance on the Bechdel test (essentially looking at the dialogue spoken by women) impacts on the profitability of the movie.

Read both the Wikipedia article above and the help information for this data

```
data("bechdel")
head(bechdel)
```

```
## # A tibble: 6 × 15
##    year imdb      title   test  clean_test binary budget domgross intgross code
##   <int> <chr>     <chr>   <chr> <ord>      <chr>   <int>    <dbl>    <dbl> <chr>
## 1  2013 tt1711425 21 & O… nota… notalk     FAIL   1.3 e7 25682380   4.22e7 2013…
## 2  2012 tt1343727 Dredd … ok-d… ok         PASS   4.50e7 13414714   4.09e7 2012…
## 3  2013 tt2024544 12 Yea… nota… notalk     FAIL   2   e7 53107035   1.59e8 2013…
## 4  2013 tt1272878 2 Guns  nota… notalk     FAIL   6.1 e7 75612460   1.32e8 2013…
## 5  2013 tt0453562 42      men   men        FAIL   4   e7 95020213   9.50e7 2013…
## 6  2013 tt1335975 47 Ron… men   men        FAIL   2.25e8 38362475   1.46e8 2013…
## # ℹ 5 more variables: budget_2013 <int>, domgross_2013 <dbl>,
## #   intgross_2013 <dbl>, period_code <int>, decade_code <int>
```

a.) Decide what criteria you should use to decide how successful a movie was, based on budget, domestic gross and international gross (ie a KPI). Consider several alternatives and explain your choice of the best measure. Add a column to the data set that holds this KPI to the data set

I believe the best metric would be the ratio between the budget and the sum of gross earnings (i.e. (domestic gross + international gross)/(budget)). An alternative measure could be the ratio between domestic gross and international gross, which would eliminate budget as an influence. The thought here is that the more successful a

movie is domestically, the more it will be marketed for international sale, which is a much larger audience than the US alone. A high success movie would have a high (intgross/domgross) value. This, however, is a worse metric than (budget)/(domestic gross + international gross) because a movie could have very few sales, but do far better in an international market, which would lead to an incorrect conclusion.

I changed this from my initial submission in the exam prep. The KPI I had used prior would work, but this makes it easier to identify performance as the highest values are the best.

```
bechdel <- bechdel |>
  mutate(kpi=(intgross+domgross)/budget, .after = intgross)
head(bechdel)
```

```
## # A tibble: 6 × 16
##    year imdb      title  test  clean_test binary budget domgross intgross    kpi
##   <int> <chr>     <chr>  <chr> <ord>      <chr>   <int>    <dbl>    <dbl>  <dbl>
## 1  2013 tt1711425 21 & … nota… notalk     FAIL   1.3 e7 25682380   4.22e7  5.22
## 2  2012 tt1343727 Dredd… ok-d… ok         PASS   4.50e7 13414714   4.09e7  1.21
## 3  2013 tt2024544 12 Ye… nota… notalk     FAIL   2   e7 53107035   1.59e8 10.6
## 4  2013 tt1272878 2 Guns nota… notalk     FAIL   6.1 e7 75612460   1.32e8  3.41
## 5  2013 tt0453562 42     men   men        FAIL   4   e7 95020213   9.50e7  4.75
## 6  2013 tt1335975 47 Ro… men   men        FAIL   2.25e8 38362475   1.46e8  0.819
## # ℹ 6 more variables: code <chr>, budget_2013 <int>, domgross_2013 <dbl>,
## #   intgross_2013 <dbl>, period_code <int>, decade_code <int>
```

b.) Compute the KPI again, but using budget_2013, domgross_2013, intgross_2013, which should be normalized to 2013 data. Call his KPI2013

```
bechdel <- bechdel |>
  mutate(kpi_2013=(intgross_2013+domgross_2013)/(budget_2013), .after = intgross_2013)
head(bechdel)
```

```
## # A tibble: 6 × 17
##    year imdb      title  test  clean_test binary budget domgross intgross    kpi
##   <int> <chr>     <chr>  <chr> <ord>      <chr>   <int>    <dbl>    <dbl>  <dbl>
## 1  2013 tt1711425 21 & … nota… notalk     FAIL   1.3 e7 25682380   4.22e7  5.22
## 2  2012 tt1343727 Dredd… ok-d… ok         PASS   4.50e7 13414714   4.09e7  1.21
## 3  2013 tt2024544 12 Ye… nota… notalk     FAIL   2   e7 53107035   1.59e8 10.6
## 4  2013 tt1272878 2 Guns nota… notalk     FAIL   6.1 e7 75612460   1.32e8  3.41
## 5  2013 tt0453562 42     men   men        FAIL   4   e7 95020213   9.50e7  4.75
## 6  2013 tt1335975 47 Ro… men   men        FAIL   2.25e8 38362475   1.46e8  0.819
## # ℹ 7 more variables: code <chr>, budget_2013 <int>, domgross_2013 <dbl>,
## #   intgross_2013 <dbl>, kpi_2013 <dbl>, period_code <int>, decade_code <int>
```

c.) Which movie did best on your KPI based on the 2013 data? on the KPI from a?

```
best_movie_2013 <- bechdel |>
  arrange(desc(kpi_2013))
head(best_movie_2013, 1)
```

```
## # A tibble: 1 × 17
##    year imdb  title test  clean_test binary budget domgross intgross   kpi code
##   <int> <chr> <chr> <chr> <ord>      <chr>   <int>    <dbl>    <dbl> <dbl> <chr>
## 1  2007 tt11… Para… dubi… dubious    FAIL   450000   1.08e8   1.94e8  671. 2007…
## # ℹ 6 more variables: budget_2013 <int>, domgross_2013 <dbl>,
## #   intgross_2013 <dbl>, kpi_2013 <dbl>, period_code <int>, decade_code <int>
```

Based on 2013 data, paranormal activity performed best.

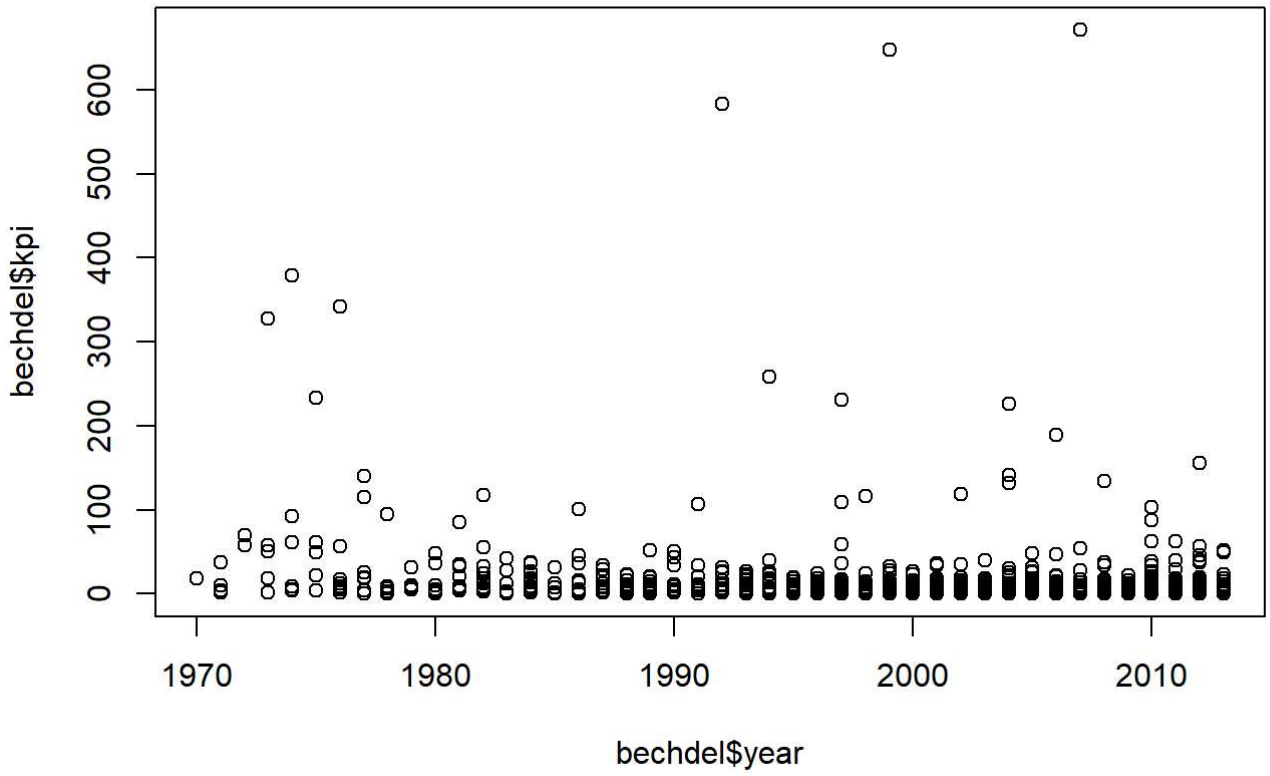
```
best_movie <- bechdel |>
  arrange(desc(kpi))
head(best_movie, 1)
```

```
## # A tibble: 1 × 17
##    year imdb  title test  clean_test binary budget domgross intgross   kpi code
##   <int> <chr> <chr> <chr> <ord>      <chr>   <int>    <dbl>    <dbl> <dbl> <chr>
## 1  2007 tt11… Para… dubi… dubious    FAIL   450000   1.08e8   1.94e8  671. 2007…
## # ℹ 6 more variables: budget_2013 <int>, domgross_2013 <dbl>,
## #   intgross_2013 <dbl>, kpi_2013 <dbl>, period_code <int>, decade_code <int>
```

Based on the kpi data from part a, paranormal activity performed best.

d.) Produce a graph that shows your success measure KPI (a) as a function of the year. Did movies become more or less profitable over this time period? Or did they not change?

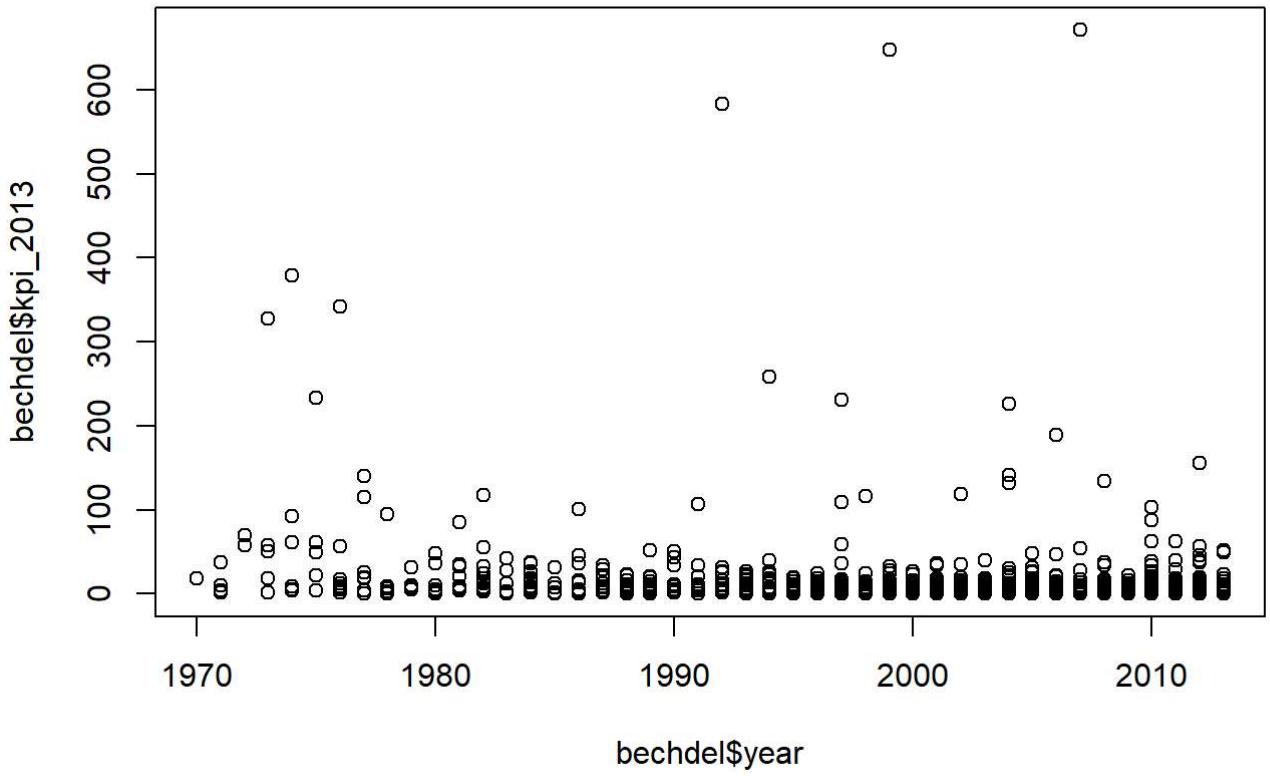
```
plot(bechdel$year, bechdel$kpi)
```



It appears that movies became less profitable over time, as the density of low kpi value movies increases by year.

e.) Produce a graph that shows your success measure KPI 2013, the standardized version (a) as a function of the year. Did movies become more or less profitable over this time period? Or did they not change?

```
plot(bechdel$year, bechdel$kpi_2013)
```



It appears that movies became less profitable over time, as the density of low kpi value movies increases by year.

Palmer Penguins Data Set Problem 4 EXAM

```
library("palmerpenguins")
```

```
head(penguins)
```

```
## # A tibble: 6 × 8
##   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
##   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
## 1 Adelie  Torgersen           39.1          18.7               181        3750
## 2 Adelie  Torgersen           39.5          17.4               186        3800
## 3 Adelie  Torgersen           40.3          18                 195        3250
## 4 Adelie  Torgersen           NA            NA                  NA          NA
## 5 Adelie  Torgersen           36.7          19.3               193        3450
## 6 Adelie  Torgersen           39.3          20.6               190        3650
## # ℹ 2 more variables: sex <fct>, year <int>
```

a.) Restrict your data set to male penguins only.

```
penguins_filtered <- penguins |>
  filter(sex=="male")
head(penguins_filtered)
```

```
## # A tibble: 6 × 8
##   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
##   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
## 1 Adelie  Torgersen           39.1          18.7               181        3750
## 2 Adelie  Torgersen           39.3          20.6               190        3650
## 3 Adelie  Torgersen           39.2          19.6               195        4675
## 4 Adelie  Torgersen           38.6          21.2               191        3800
## 5 Adelie  Torgersen           34.6          21.1               198        4400
## 6 Adelie  Torgersen           42.5          20.7               197        4500
## # ℹ 2 more variables: sex <fct>, year <int>
```

a.) Create a long version of the data set, with species and island as indices (variables), the bill length, bill depth, flipper length and body mass should be in the names_to setting and the values in values, year should be dropped.

```
penguins_long <- penguins_filtered |>
  select(!sex & !year) |>
  pivot_longer(!species & !island, names_to="Variable", values_to="Value")
head(penguins_long)
```

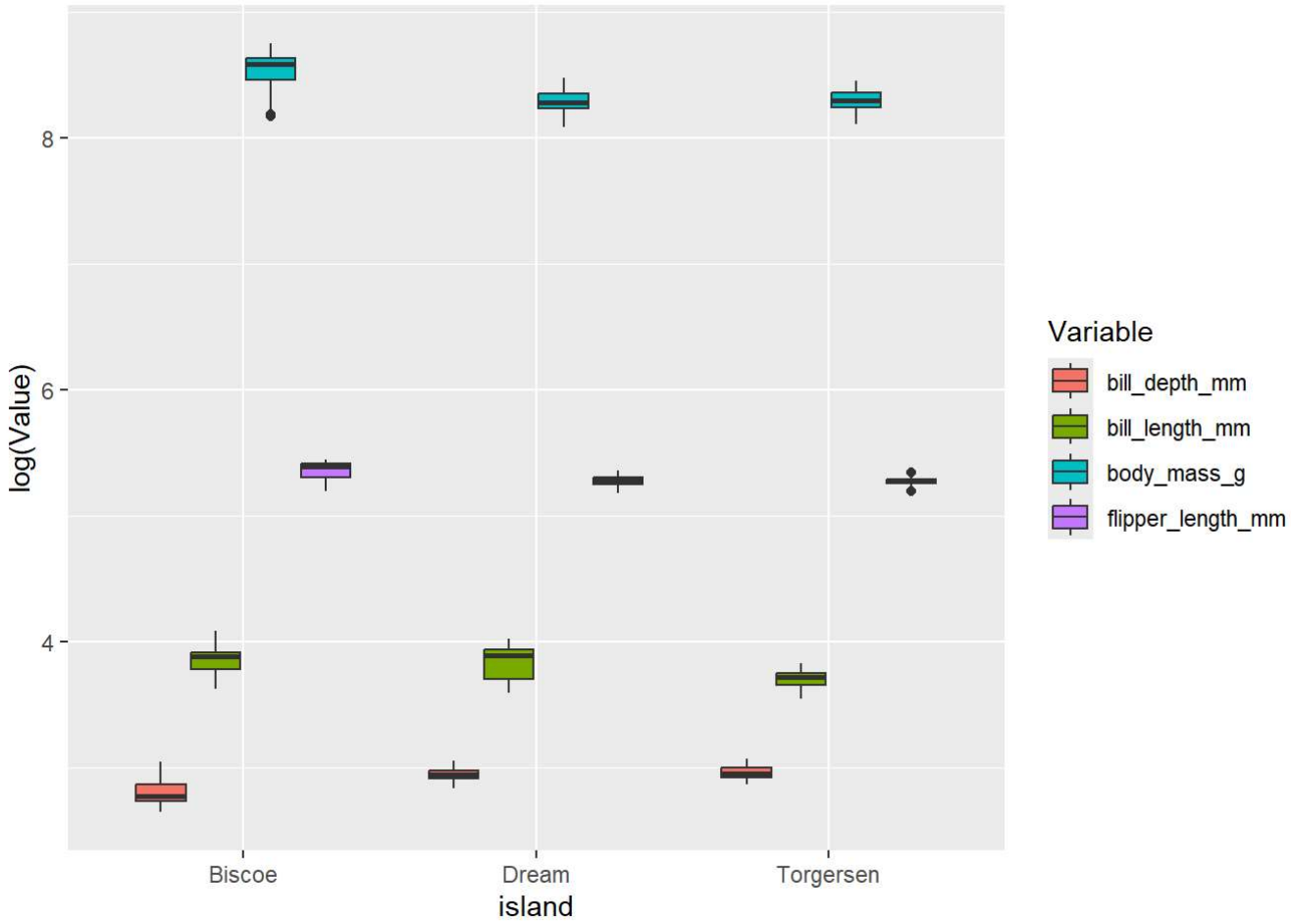
```
## # A tibble: 6 × 4
##   species island    Variable          Value
##   <fct>   <fct>     <chr>             <dbl>
## 1 Adelie  Torgersen bill_length_mm     39.1
## 2 Adelie  Torgersen bill_depth_mm      18.7
## 3 Adelie  Torgersen flipper_length_mm  181
## 4 Adelie  Torgersen body_mass_g       3750
## 5 Adelie  Torgersen bill_length_mm     39.3
## 6 Adelie  Torgersen bill_depth_mm      20.6
```

It should look like this (but without the sex column)

Species island sex Variable Value Adelie Torgerson male bill_lenth_mm 39.1 Adelie Torgerson male bill_depth_mm 18.7 Adelie Torgerson male flipper_length_mm 181 Adelie Torgerson male body_mass_g 3750 Adelie Torgerson female bill_lenth_mm 39.5 Adelie Torgerson female bill_depth_mm 17.4 Adelie Torgerson female flipper_length_mm 18 Adelie Torgerson female body_mass_g 3800

b.) Produce a box plot that shows the all the Variables (bill_length, bill_depth,flipper_length and body_mass) for each island, using the long data

```
ggplot(
  data=penguins_long,
  mapping=aes(x=island, y=log(Value), fill=Variable)
) + geom_boxplot()
```

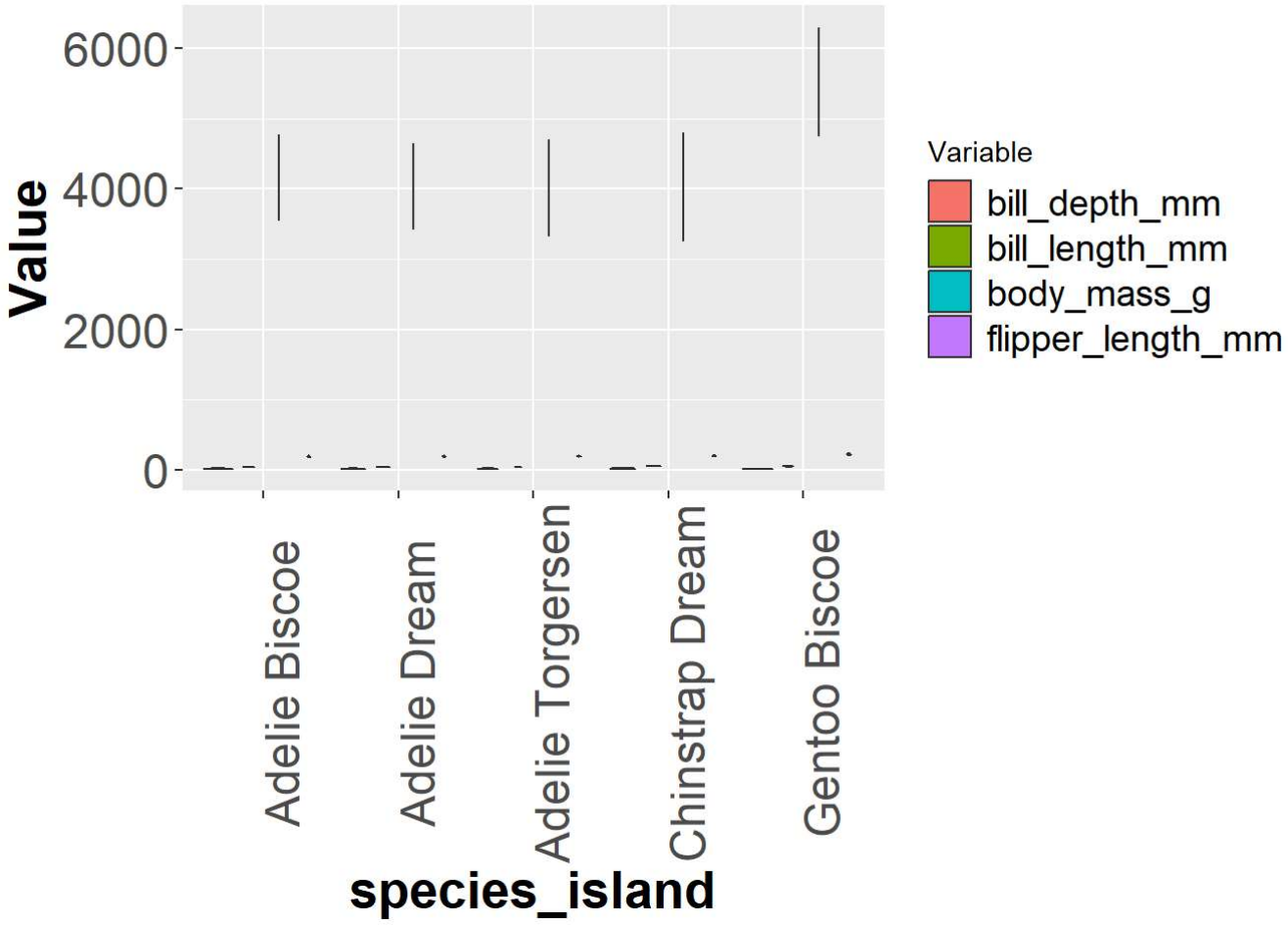


c.) Show a violin plot that shows all the variables (bill_length, bill_depth,flipper_length and body_mass) for each species and island, using the long data

```
#create a column that combines sex and species
penguins_long_species_island <- penguins_long |>
  mutate(species_island=as.factor(paste0(species, " ", island)), .before=Variable) |>
  select(!species & !island) |>
  filter(!is.na(Value))

#verify this did what I want
head(penguins_long_species_island)
```

```
## # A tibble: 6 × 3
##   species_island   Variable           Value
##   <fct>            <chr>              <dbl>
## 1 Adelie Torgersen bill_length_mm      39.1
## 2 Adelie Torgersen bill_depth_mm       18.7
## 3 Adelie Torgersen flipper_length_mm  181
## 4 Adelie Torgersen body_mass_g       3750
## 5 Adelie Torgersen bill_length_mm      39.3
## 6 Adelie Torgersen bill_depth_mm       20.6
```

```
#plot the violin plot
ggplot(
  data=penguins_long_species_island,
  mapping=aes(x=species_island, y=Value, fill=Variable)
) + geom_violin() + theme(
  legend.text = element_text(size = 14),
  axis.title = element_text(size = 20,face="bold"),
  axis.text = element_text(size = 18),
  axis.text.x = element_text(angle = 90)
  )
```



After running this, it looks like body mass is messing up the y-axis. Normalize all values from 0-1 to make this a little easier to understand.

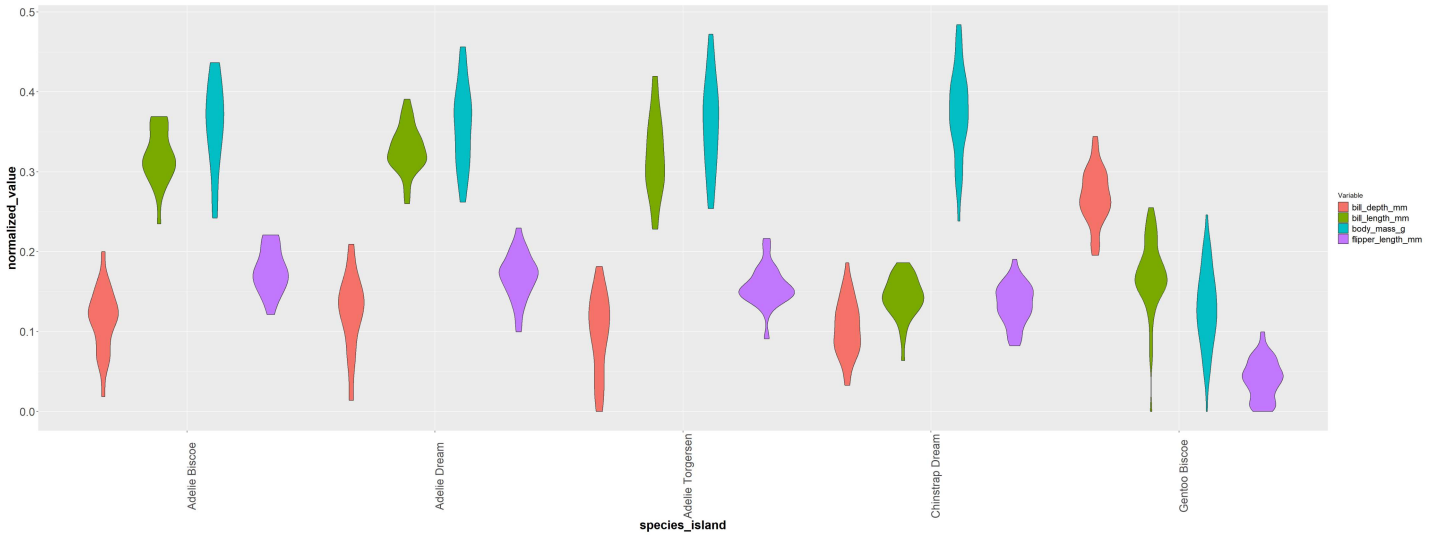
```
#Find the max value of each variable to normalize against
variables <- as.vector(unique(penguins_long_species_island$Variable))
max_list <- vector()

#store all max values to a vector
for (i in 1:length(variables)) {
  filtered_df <- penguins_long_species_island |>
    filter(Variable==variables[i])
  max_list[i] <- max(filtered_df$Value)
}

#check that the values have been captured
max_list
```

```
## [1]   59.6   21.5  231.0 6300.0
```

```
#Create a column that conditionally normalizes the data
penguins_long_species_island_normalized <- penguins_long_species_island |>
  mutate(normalized_value=ifelse(
    #condition
    Variable==variables[1],
    #True
    ((max_list[1]-Value)/max_list[1]),
    #False
    ifelse(
      #condition
      Variable==variables[2],
      #True
      ((max_list[2]-Value)/max_list[2]),
    #False
    ifelse(
      #condition
      Variable==variables[3],
      #True
      ((max_list[3]-Value)/max_list[3]),
    #False
    ((max_list[4]-Value)/max_list[4])
    )
    )
    )
    )

#There was a lot of conditional calculation there...
#verify this did what I want
penguins_long_species_island_normalized
```

```
## # A tibble: 672 × 4
##    species_island   Variable          Value normalized_value
##    <fct>            <chr>             <dbl>            <dbl>
##  1 Adelie Torgersen bill_length_mm      39.1           0.344
##  2 Adelie Torgersen bill_depth_mm       18.7           0.130
##  3 Adelie Torgersen flipper_length_mm  181             0.216
##  4 Adelie Torgersen body_mass_g       3750             0.405
##  5 Adelie Torgersen bill_length_mm      39.3           0.341
##  6 Adelie Torgersen bill_depth_mm       20.6           0.0419
##  7 Adelie Torgersen flipper_length_mm  190             0.177
##  8 Adelie Torgersen body_mass_g       3650             0.421
##  9 Adelie Torgersen bill_length_mm      39.2           0.342
## 10 Adelie Torgersen bill_depth_mm       19.6           0.0884
## # ℹ 662 more rows
```

```
#plot the violin plot
ggplot(
  data=penguins_long_species_island_normalized,
  mapping=aes(x=species_island, y=normalized_value, fill=Variable)
) + geom_violin() + theme(
  legend.text = element_text(size = 14),
  axis.title = element_text(size = 20,face="bold"),
  axis.text = element_text(size = 18),
  axis.text.x = element_text(angle = 90)
  )
```



NOTE: I found this source (https://bookdown.org/yihui/rmarkdown-cookbook/figure-size.html) when completing the midterm prep.

Pulitzer Prize Finalists and Newspaper circulation Problem 5

Does the number of Pulitzer prize nominations influence circulation

```
head(pulitzer)
```

```
## # A tibble: 6 × 7
##   newspaper           circ2004 circ2013 pctchg_circ num_finals1990_2003
##   <chr>                  <dbl>    <dbl>       <int>                <int>
## 1 USA Today            2192098  1674306         -24                    1
## 2 Wall Street Journal  2101017  2378827          13                   30
## 3 New York Times       1119027  1865318          67                   55
## 4 Los Angeles Times     983727   653868         -34                   44
## 5 Washington Post       760034   474767         -38                   52
## 6 New York Daily News   712671   516165         -28                    4
## # ℹ 2 more variables: num_finals2004_2014 <int>, num_finals1990_2014 <int>
```

a.) How many papers are there?

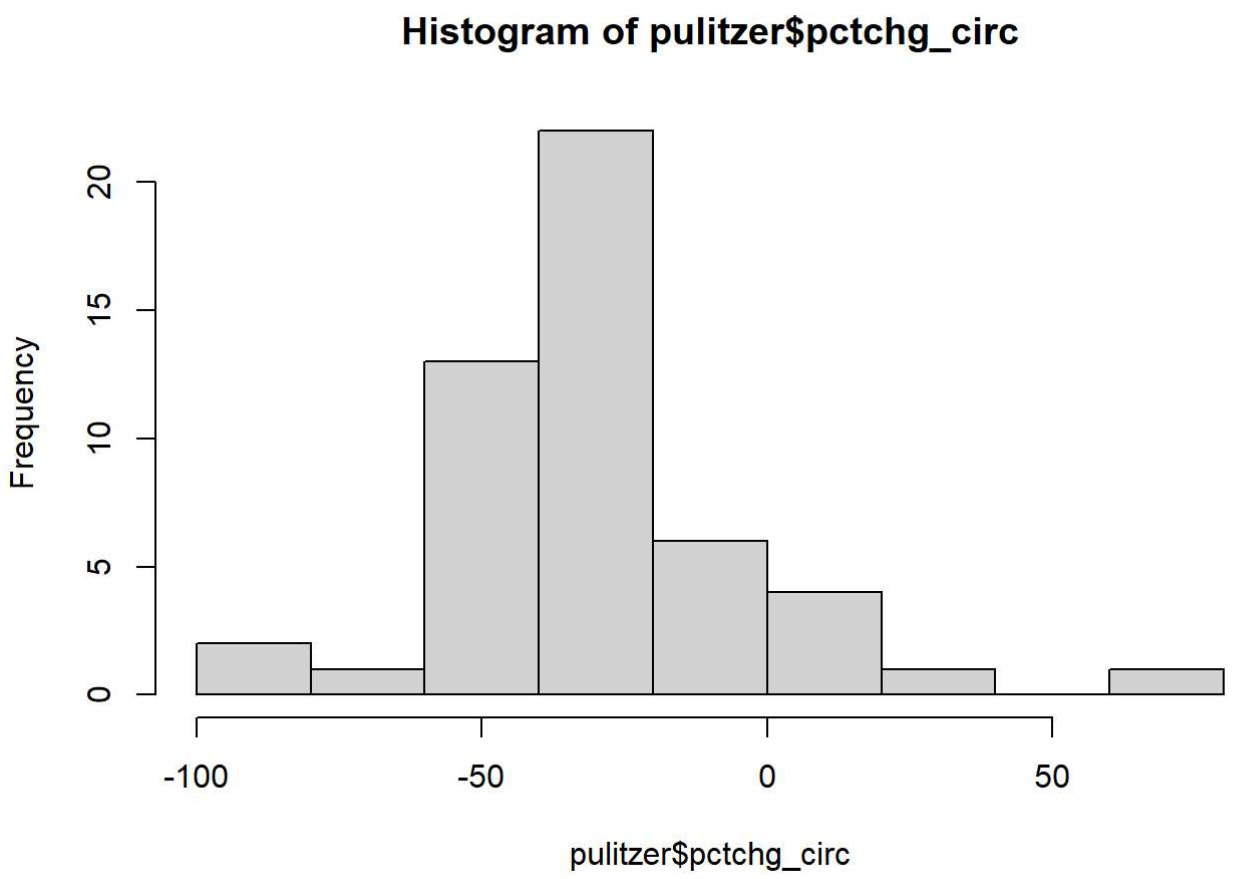
```
length(unique(pulitzer$newspaper))
```

```
## [1] 50
```

There are 50 papers.

b.) Show a histogram of the distribution of percentage change in circulation from 2004 to 2013 over all newspapers.

```
hist(pulitzer$pctchg_circ)
```



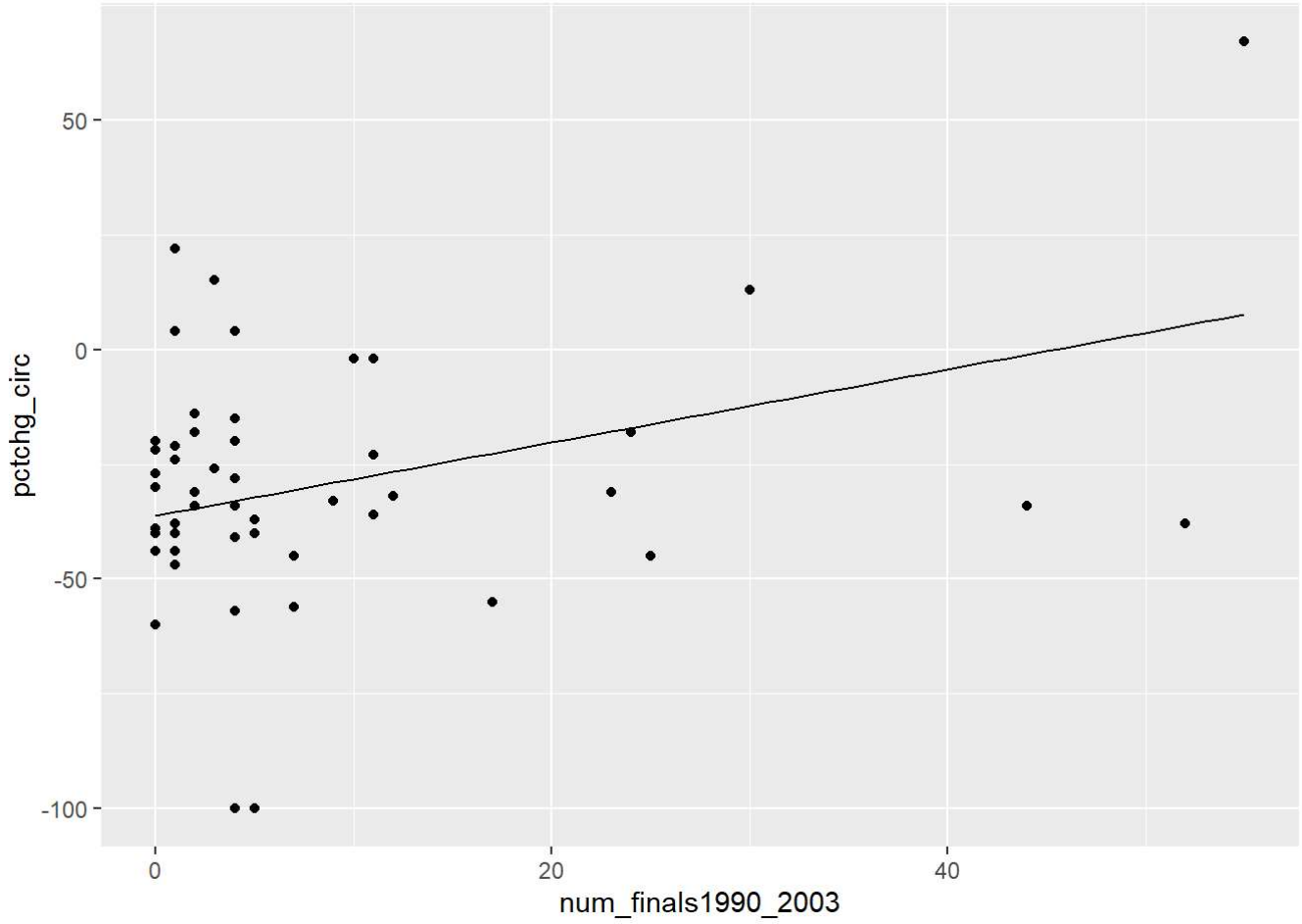
c.) Show a plot that indicates how the number of finals 1990_2003 might have influenced the percent change in circulation.

```
#determine how correlated the two values are, then get a range over the x axis values
correlation <- cor(pulitzer$num_finals1990_2003, pulitzer$pctchg_circ)
x_range <- seq(min(pulitzer$num_finals1990_2003),max(pulitzer$num_finals1990_2003))

#Filter ctchg_circ when num_finals1990_2003=0 so we can then find the y intercept of the correla
tion line
x_0 <- pulitzer |>
  filter(num_finals1990_2003==0)

#determine a scaling factor for the correlation based on the standard deviation of the y data
scale_factor <- 0.1*sd(pulitzer$pctchg_circ)

#Create a data frame with the x and y correlation data
correlation_data <- data.frame(x_cor=x_range, y_cor=(correlation*x_range*scale_factor)+mean(x_0
$pctchg_circ))
correlation_data
```

```
##    x_cor       y_cor
## 1      0 -36.2222222
## 2      1 -35.4254159
## 3      2 -34.6286096
## 4      3 -33.8318032
## 5      4 -33.0349969
## 6      5 -32.2381906
## 7      6 -31.4413842
## 8      7 -30.6445779
## 9      8 -29.8477716
## 10     9 -29.0509652
## 11    10 -28.2541589
## 12    11 -27.4573526
## 13    12 -26.6605462
## 14    13 -25.8637399
## 15    14 -25.0669336
## 16    15 -24.2701272
## 17    16 -23.4733209
## 18    17 -22.6765146
## 19    18 -21.8797083
## 20    19 -21.0829019
## 21    20 -20.2860956
## 22    21 -19.4892893
## 23    22 -18.6924829
## 24    23 -17.8956766
## 25    24 -17.0988703
## 26    25 -16.3020639
## 27    26 -15.5052576
## 28    27 -14.7084513
## 29    28 -13.9116449
## 30    29 -13.1148386
## 31    30 -12.3180323
## 32    31 -11.5212259
## 33    32 -10.7244196
## 34    33  -9.9276133
## 35    34  -9.1308069
## 36    35  -8.3340006
## 37    36  -7.5371943
## 38    37  -6.7403879
## 39    38  -5.9435816
## 40    39  -5.1467753
## 41    40  -4.3499690
## 42    41  -3.5531626
## 43    42  -2.7563563
## 44    43  -1.9595500
## 45    44  -1.1627436
## 46    45  -0.3659373
## 47    46   0.4308690
## 48    47   1.2276754
## 49    48   2.0244817
## 50    49   2.8212880
## 51    50   3.6180944
```

```
## 52    51   4.4149007
## 53    52   5.2117070
## 54    53   6.0085134
## 55    54   6.8053197
## 56    55   7.6021260
```

```
#plot the scatter plot with the correlation overlayed on top
ggplot(
  data=pulitzer,
  mapping=aes(x=num_finals1990_2003, y=pctchg_circ)
) +
  geom_point() +
  geom_line(
    data = correlation_data,
    mapping=aes(x=x_cor, y=y_cor)
  )
```



d.) Show a plot of the relationship of the circulation in 2004 to the number of finals 2004 to 2014

```
#determine how correlated the two values are, then get a range over the x axis values
correlation <- cor(pulitzer$num_finals2004_2014, pulitzer$circ2004)
correlation
```

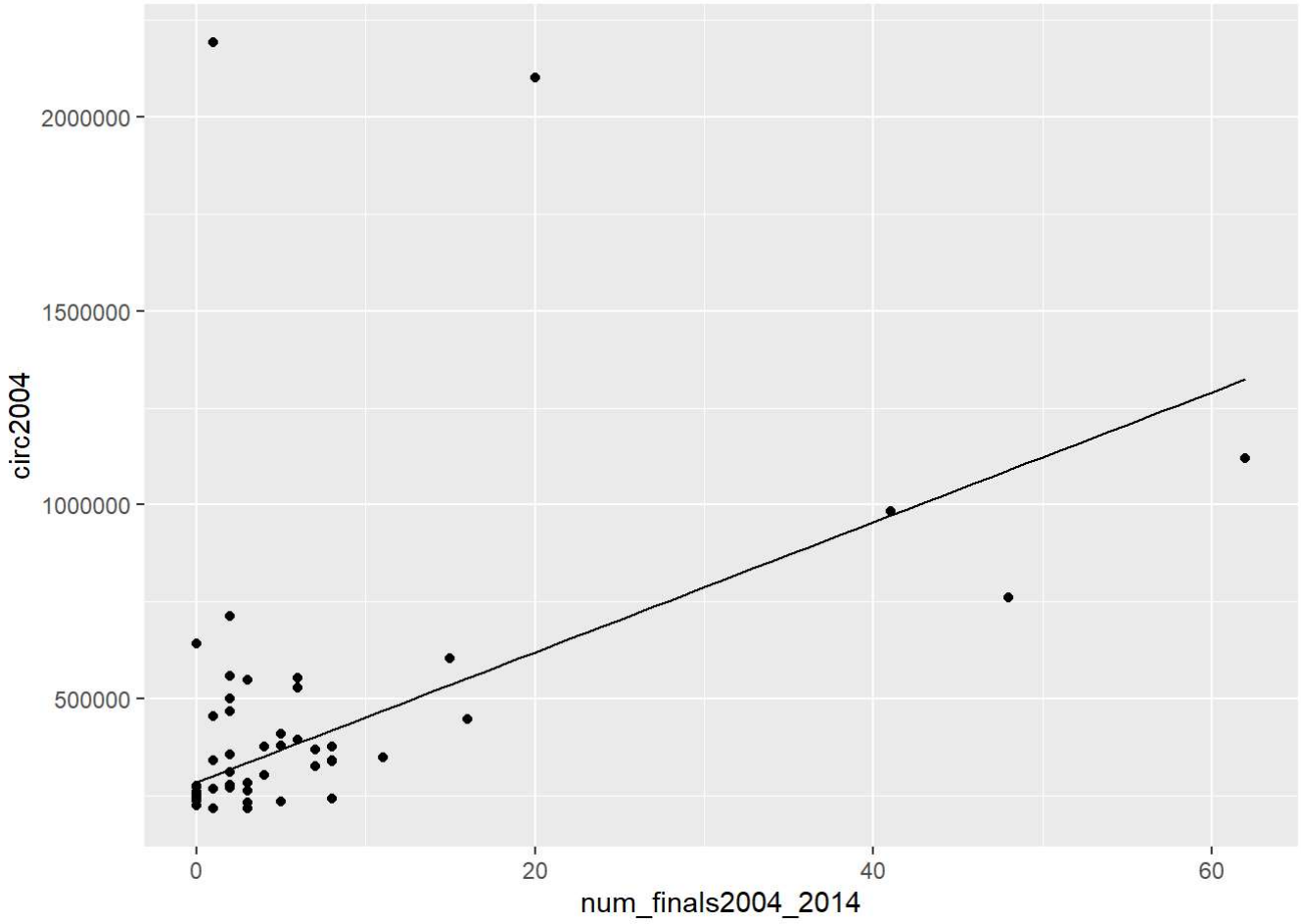
```
## [1] 0.4228815
```

```
x_range <- seq(min(pulitzer$num_finals2004_2014),max(pulitzer$num_finals2004_2014))

#Filter ctchg_circ when num_finals2004_2014=0 so we can then find the y intercept of the correla
tion line
x_0 <- pulitzer |>
  filter(num_finals2004_2014==0)

#determine a scaling factor for the correlation based on the standard deviation of the y data
scale_factor <- 0.1*sd(pulitzer$circ2004)

#Create a data frame with the x and y correlation data
correlation_data <- data.frame(x_cor=x_range, y_cor=(correlation*x_range*scale_factor)+mean(x_0
$circ2004))
correlation_data
```

```
##    x_cor     y_cor
## 1      0  284366.3
## 2      1  301137.1
## 3      2  317907.9
## 4      3  334678.8
## 5      4  351449.6
## 6      5  368220.5
## 7      6  384991.3
## 8      7  401762.1
## 9      8  418533.0
## 10     9  435303.8
## 11    10  452074.7
## 12    11  468845.5
## 13    12  485616.3
## 14    13  502387.2
## 15    14  519158.0
## 16    15  535928.9
## 17    16  552699.7
## 18    17  569470.5
## 19    18  586241.4
## 20    19  603012.2
## 21    20  619783.0
## 22    21  636553.9
## 23    22  653324.7
## 24    23  670095.6
## 25    24  686866.4
## 26    25  703637.2
## 27    26  720408.1
## 28    27  737178.9
## 29    28  753949.8
## 30    29  770720.6
## 31    30  787491.4
## 32    31  804262.3
## 33    32  821033.1
## 34    33  837803.9
## 35    34  854574.8
## 36    35  871345.6
## 37    36  888116.5
## 38    37  904887.3
## 39    38  921658.1
## 40    39  938429.0
## 41    40  955199.8
## 42    41  971970.7
## 43    42  988741.5
## 44    43 1005512.3
## 45    44 1022283.2
## 46    45 1039054.0
## 47    46 1055824.8
## 48    47 1072595.7
## 49    48 1089366.5
## 50    49 1106137.4
## 51    50 1122908.2
```

```
## 52    51 1139679.0
## 53    52 1156449.9
## 54    53 1173220.7
## 55    54 1189991.6
## 56    55 1206762.4
## 57    56 1223533.2
## 58    57 1240304.1
## 59    58 1257074.9
## 60    59 1273845.8
## 61    60 1290616.6
## 62    61 1307387.4
## 63    62 1324158.3
```

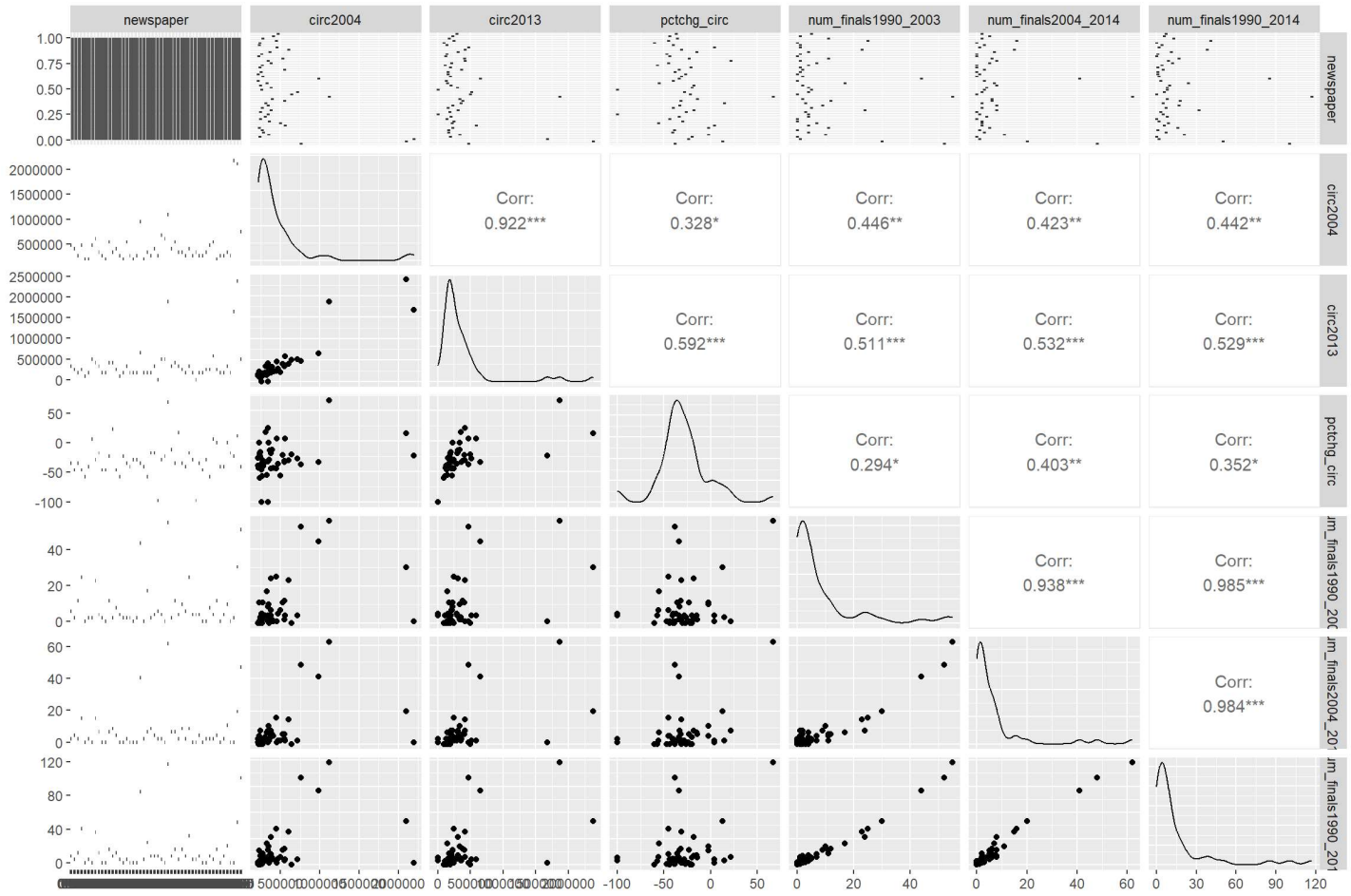
```
#plot the scatter plot with the correlation overlayed on top
ggplot(
  data=pulitzer,
  mapping=aes(x=num_finals2004_2014, y=circ2004)
) +
  geom_point() +
  geom_line(
    data = correlation_data,
    mapping=aes(x=x_cor, y=y_cor)
  )
```



e.) Show a ggpairs plot of this data set. Which variables are most highly correlated? What do you think this means?

```
ggpairs(pulitzer, cardinality_threshold = length(unique(pulitzer$newspaper)))
```

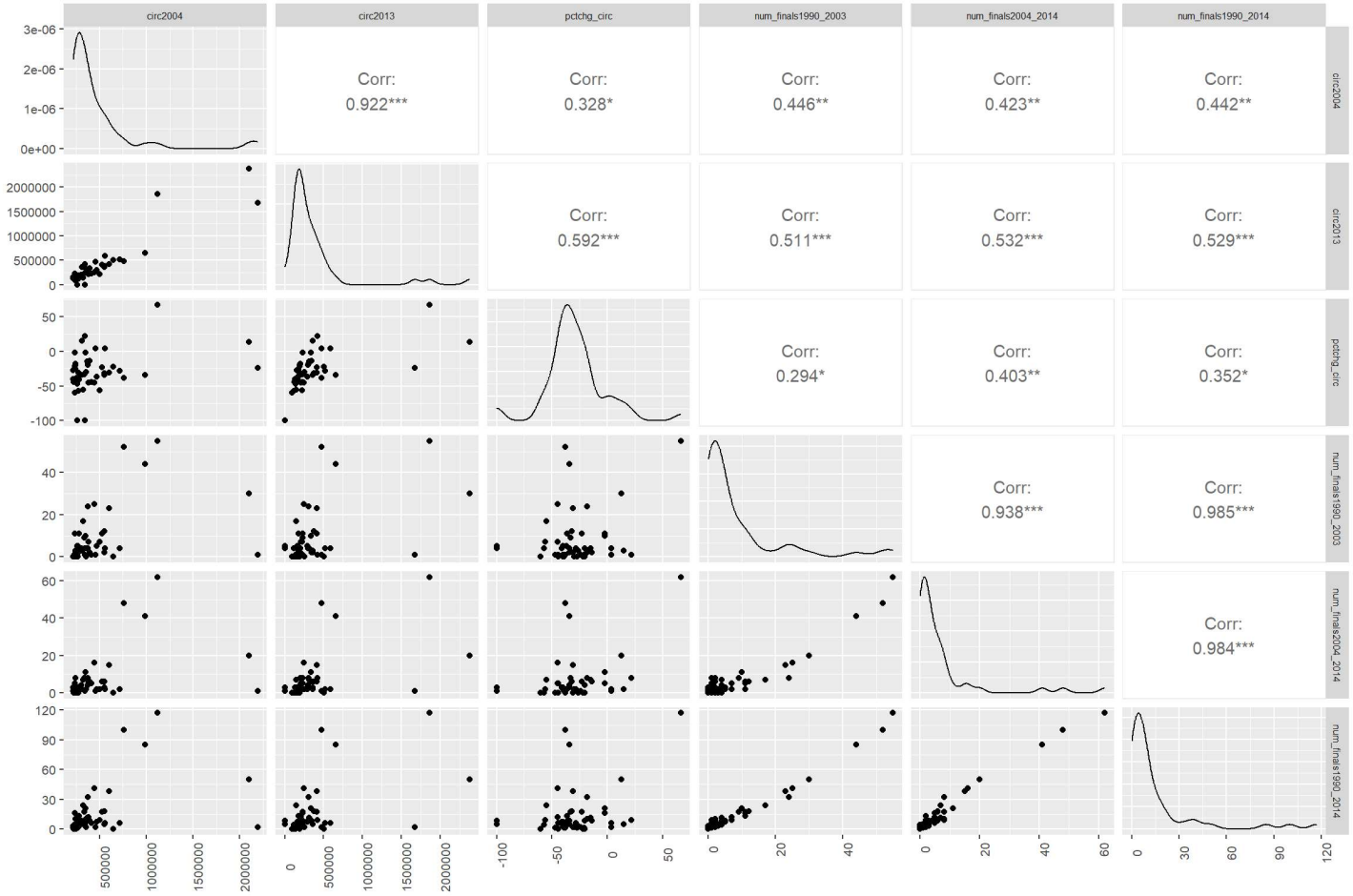
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



This doesn't quite look right… It seems like omitting the newspaper may help. Perhaps the cardinality threshold is there for a reason.

Formatting Sources: https://bookdown.org/yihui/rmarkdown-cookbook/figure-size.html (https://bookdown.org/yihui/rmarkdown-cookbook/figure-size.html)
https://stackoverflow.com/questions/73289397/ggpairs-change-axis-label-font-size (https://stackoverflow.com/questions/73289397/ggpairs-change-axis-label-font-size)

```
#Remove newspaper from the selected columns
pulitzer_minus_np <- pulitzer |>
  select(!newspaper)

#Perform ggpairs with some reformatting
ggpairs(pulitzer_minus_np) + theme(
  strip.text.x = element_text(size = 6),
  strip.text.y = element_text(size = 6),
  axis.text = element_text(size = 8),
  axis.text.x = element_text(angle = 90)
  )
```



Ahhhh, much better. We don't care about the newspaper much anyway, these should be global trends across the press industry. The highest correlations are between num_finals1990_2014 and num_finals_1990_2003, num_finals1990_2014 and num_finals2004_2014, num_finals2004_2014 and num_finals_1990_2003, and circ2013 and circ2004, respectively. The Pulitzer Prize is a prestigious award, so it is unlikely that the number of Pulitzer prizes will dramatically increase over time, hence the strong correlation between the number of finalists in all of the different time ranges. Additionally, the number of papers in circulation daily should also be strongly correlated, especially in the time frames being analyzed. Media between 2004 and 2013 had begun to shift to a digital format, which likely stagnated the number of newspapers in favor of other news and media outlets. This would explain the near linear relationship between the number of papers in circulation from 2004 to 2013.

Exponential distribution Problem 6 EXAM

We will be looking at the exponential distribution, which is a continuous distribution much like the normal or poison

It has only one parameter a rate.

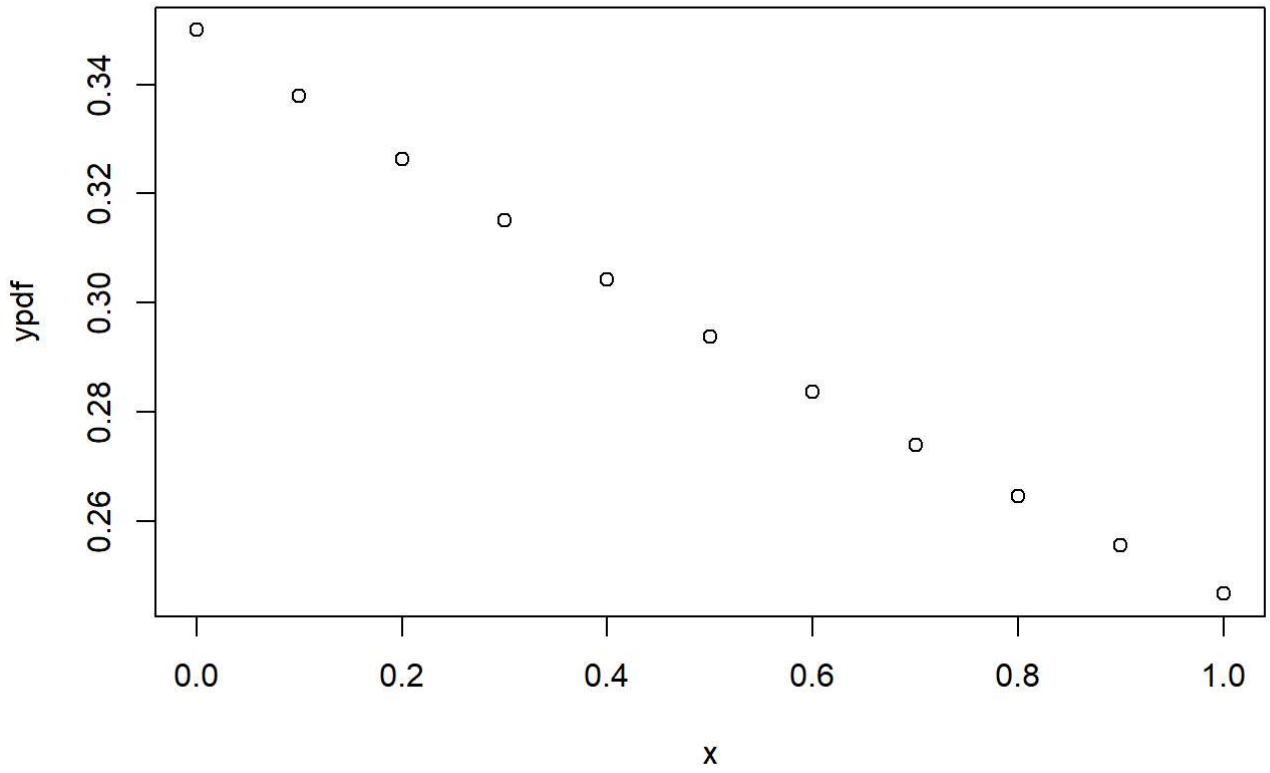
We will look at an exponential distribution with a rate of .35

Using a sequence of x values form 0 to 1 in steps of 0.1 (set up in code block below)

```
x<-seq(0,1,0.1)
```

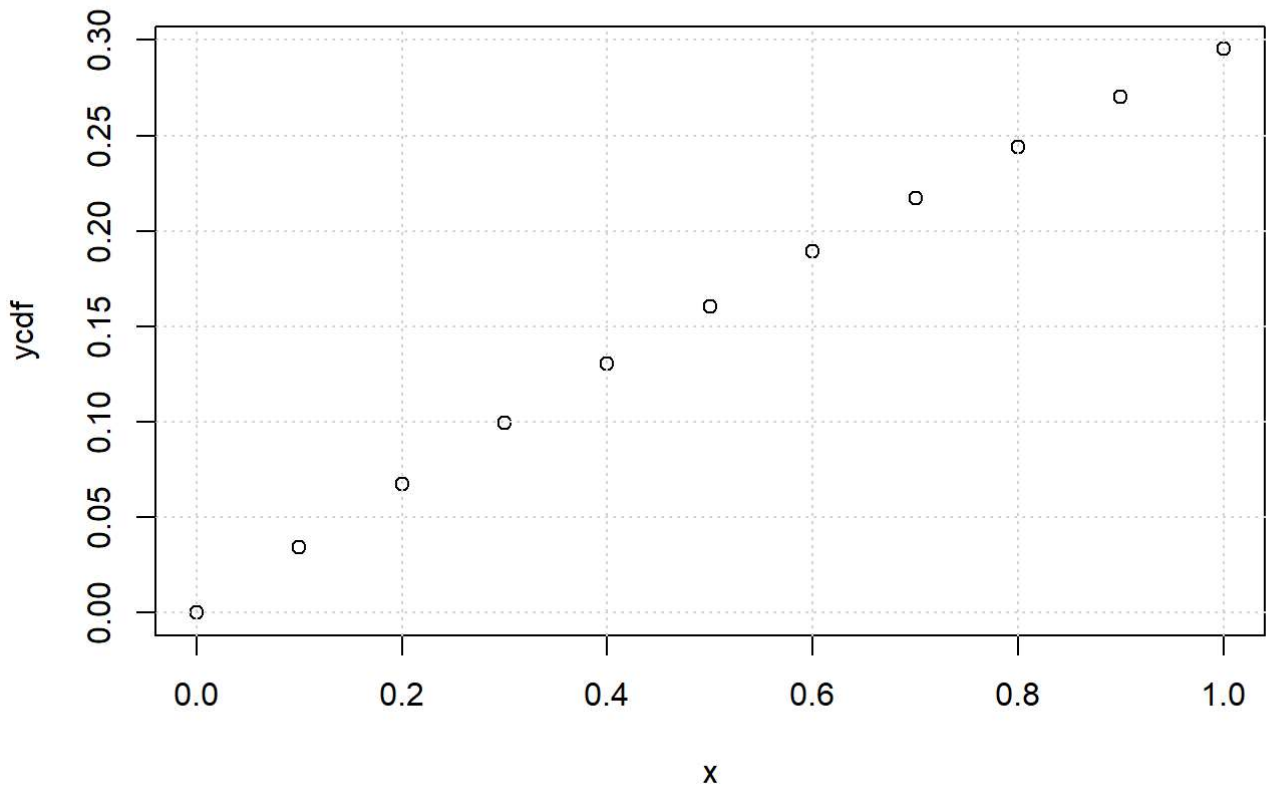
a.) Plot the probability distribution function (pdf) for an exponential distribution with rate 0.35

```
ypdf=dexp(x, rate=0.35)
plot(x, ypdf)
```



b.) Plot the cummulative probability distribution (cdf) for an exponential distribution with rate 0.35, add a grid to this

```
ycdf=pexp(x,0.35)
plot(x,ycdf)
grid()
```



c.) Find the mean and standard deviation of this distribution

```
mean(ycdf)
```

```
## [1] 0.155392
```

```
sd(ycdf)
```

```
## [1] 0.09791693
```

d.) What is the probablity that a value (x) from this distribution is less than 4?

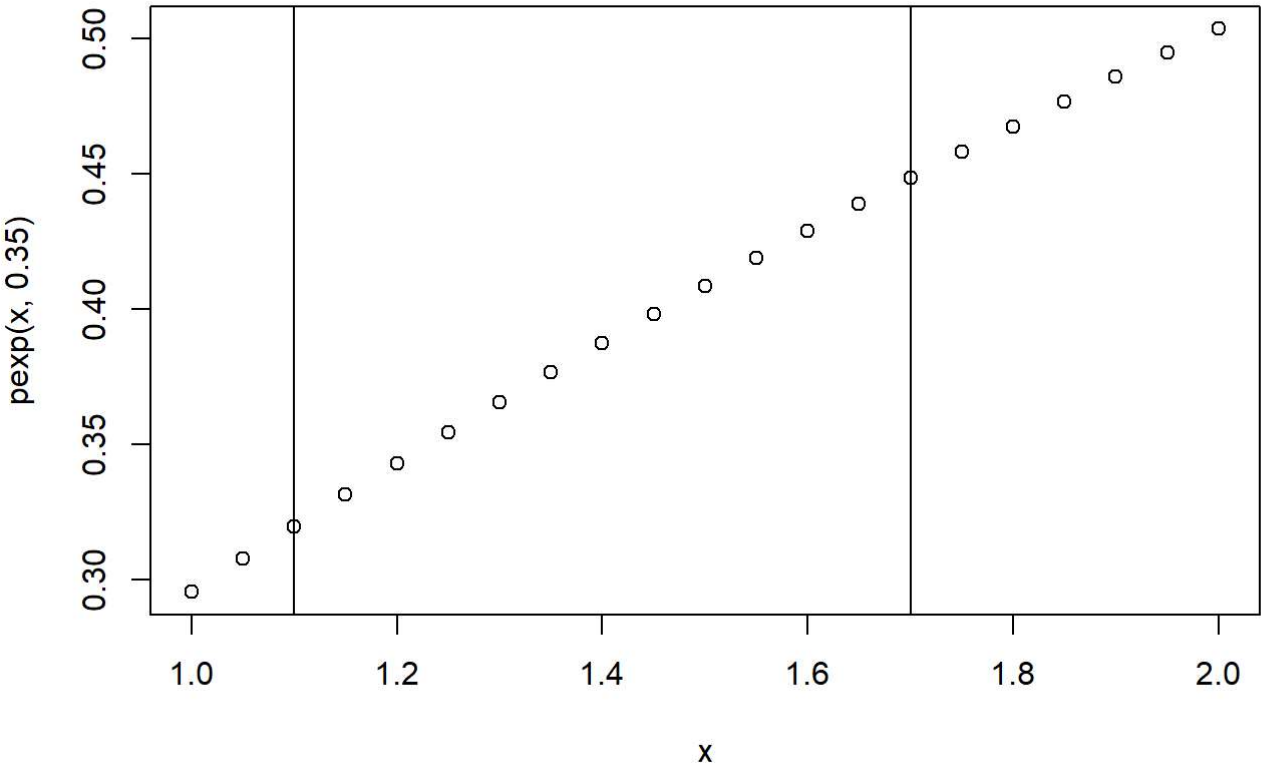
```
prob <- pexp(4, rate=0.35)
prob
```

```
## [1] 0.753403
```

About 75.3%. This series is constrained to less than 1, but in the actual distribution, x goes to infinity, so 4 is a valid input for x.

e.) What is the probability that value (x) from this distribution is in the range 1.1 to 1.7?

```
#Plot this to get a better intuitive understanding.
x=seq(1,2,0.05)
plot(x, pexp(x, 0.35))
abline(v=1.1)
abline(v=1.7)
```




```
upper <- pexp(1.7,0.35)
lower <- pexp(1.1,0.35)
upper-lower
```

```
## [1] 0.1288881
```

About 12.9%

Log normal Distribution Problem #7

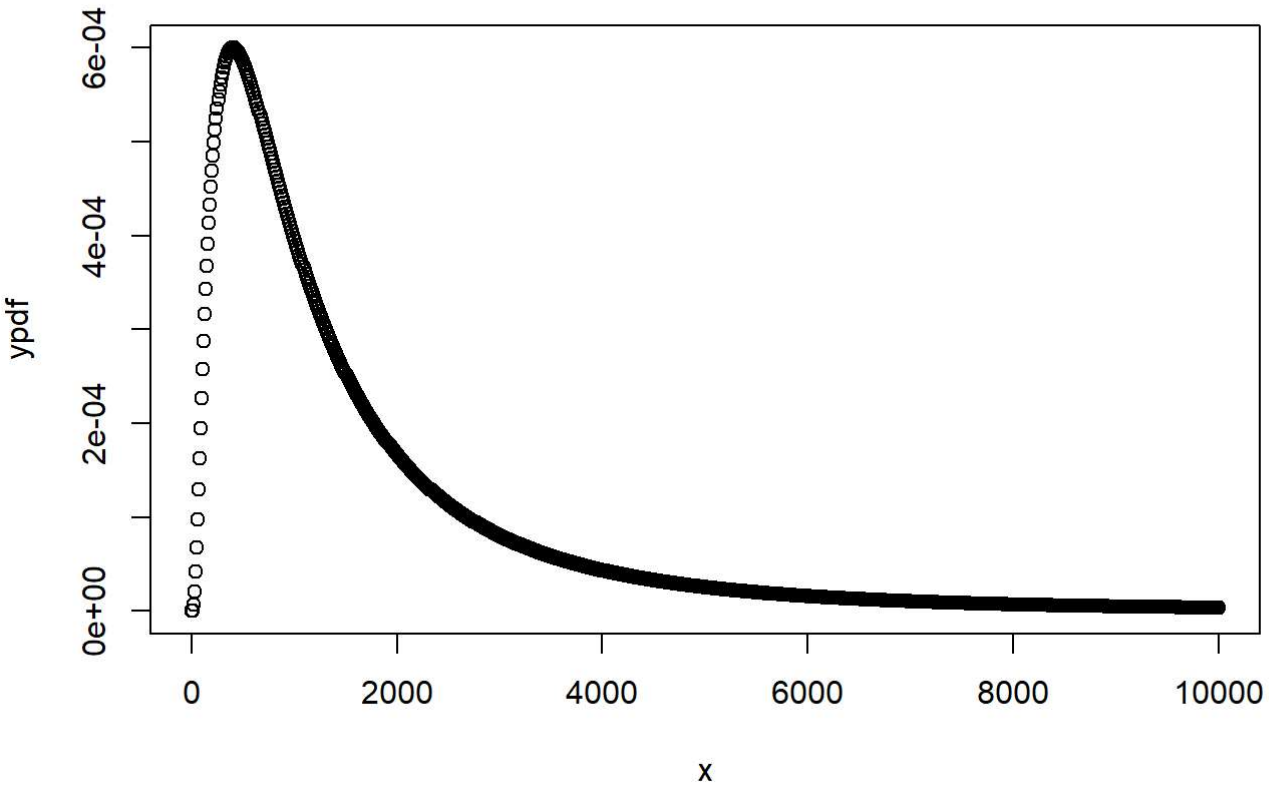
The log normal has parameters meanlog and sdlog

We will set meanlog=7, sdlog=1

a.) Plot the log normal probability distribution function (pdf) for x=0 to 10,000 using the x sequence list set up below

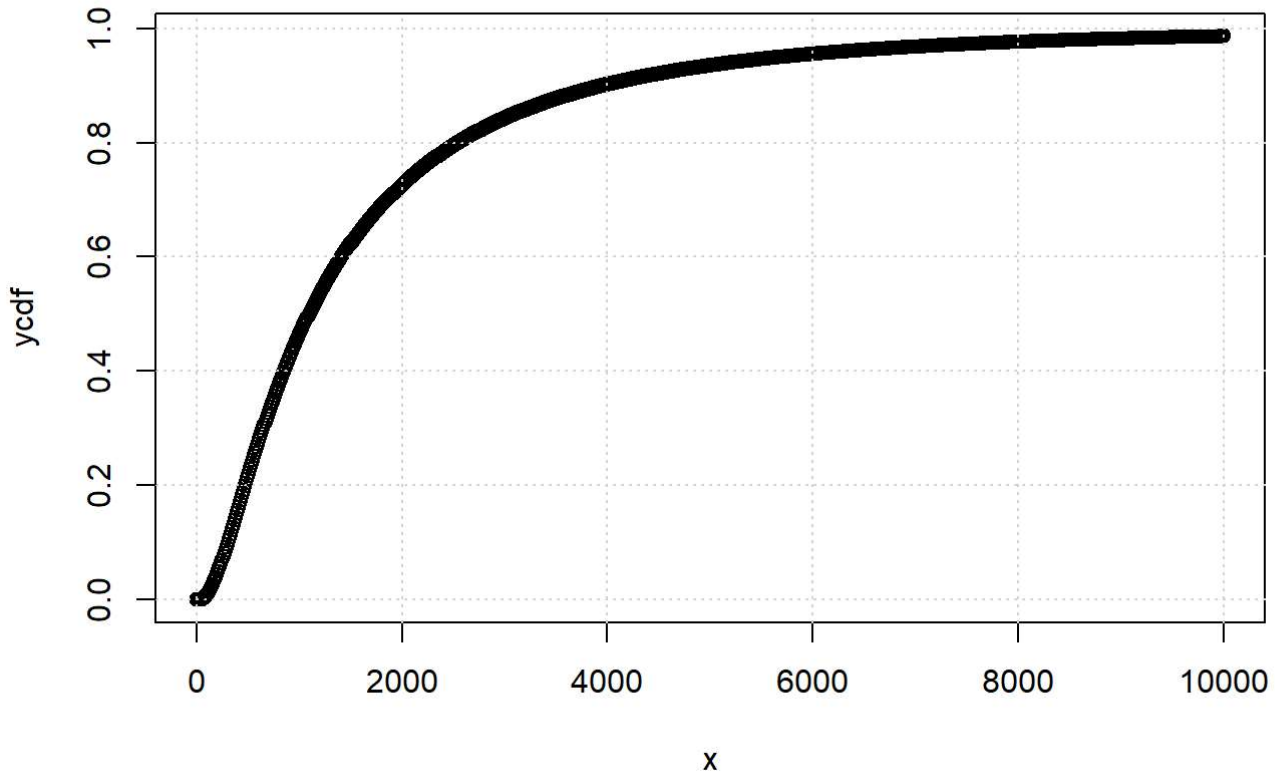
Source: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/Lognormal (https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/Lognormal)

```
x=seq(0,10000,10)
ypdf=dlnorm(x, mean = 7, sd = 1)
plot(x, ypdf)
```



b.) Plot the log normal cumulative distribution function (cdf) for the same range of x. Add a grid to this plot

```
x=seq(0,10000,10)
ycdf=plnorm(x, mean = 7, sd = 1)
plot(x, ycdf)
grid()
```



c.) What are the mean and standard deviation of this distribution

```
mean(ycdf)
```

```
## [1] 0.8257667
```

```
sd(ycdf)
```

```
## [1] 0.2382591
```

d.) What is the maximum of the pdf?

```
max(ypdf)
```

```
## [1] 0.0005997636
```

e.) What is the probability of a random x from this distribution being less than 500?

```
plnorm(500, mean = 7, sd = 1)
```

```
## [1] 0.2161119
```

f.) What is the probability of a random x from this distribution being less than 200 OR more than 1500?

```
x=seq(0,10000,10)

upper <- plnorm(1500, mean = 7, sd = 1)
lower <- plnorm(200, mean = 7, sd = 1)

#I am doing this to omit the portion of the distribution in the range 200:1500
1-(upper-lower)
```

```
## [1] 0.4214641
```