Text Mining Gutenberg Assignment Corpus Linguistics

Hongxu Zhou

February 2024

Goal

The goal of this assignment is

- to familiarize yourself with the tidytext environment and to install and load the basic libraries
- to produce and visualize word frequency statistics for a given text

Text Mining with R

Read the first chapter of Text Mining with R.

If you want to be able to actually run and modify the code in the examples, you can download all the code (Rmarkdown files and more) from github.

The examples use specialized packages that are not included by default in your R distribution. Install these packages using install.packages(). (I find it easiest to do this from the command line and not include it in the Rmarkdown, as installation needs to be done only once but does produce quite a bit of output.)

Gutenberg

In this assignment you will produce a word frequency list for one of the authors included in the Gutenberg Project.

- Install and load the gutenbergr package. See the gutenbergr tutorial for some examples of how to search the metadata and download the actual texts.
- Select an author such that the first letter of his/her last name matches the first letter of your last name. You can use gutenberg_works(author == ...) or string matching over the author field to find (lists of) authors. (Note that the str_detect method from the stringr library takes a regular expression as second argument. The regular expression "^B" for instance, matches only with strings beginning with B).
- Download (some of) the texts of your selected author and store this in a variable. (You can give a list of gutenberg_id's as argument to the gutenberg_download function.)

```
# get metadata
books <- gutenberg_metadata

# get the authors with their names staring with Z and in English
filtered_books <- books %>%
  filter(
```

```
    str_starts(author, "Z"), # author name starts with Z
    language == "en", # language is English
    has_text == TRUE # some books have no text why?
  )

# show the first three rows
head(filtered_books, 5)
```

```
## # A tibble: 5 x 8
##   gutenberg_id title     author gutenberg_author_id language gutenberg_bookshelf
##          <int> <chr>     <chr>                <int> <chr>    <chr>
## 1          338 Old Indi~ Zitka~                 188 en       "Native America"
## 2         1445 Aeroplan~ Zerbe~                635 en       ""
## 3         6304 Without ~ Zangw~               1987 en       ""
## 4         6626 Theresa ~ Zola,~                528 en       ""
## 5         7806 A Boy's ~ Zolli~               2536 en       ""
## # i 2 more variables: rights <chr>, has_text <lgl>
```

I will use the ** first three** books above as the dataset for this assignment.

Preprocessing and word counting

- Apply the chapter detection from section 1.3 of Text Mining with R to your texts. For this, ensure that the texts you chose have chapter headings (otherwise, pick another author). Adapt the regular expression if needed. (Note that the result of downloading a set of texts is already a data frame, that can be used directly with functions such as unnest_tokens.) Do some spot checking to see that

  (a) the detected chapter headings are correct and (b) you don't miss chapter headings; show some examples. Does the number of chapters match the number reported in the table of contents?

- Create a word frequency list for your author. Filter stop words. What are the ten most frequent words in the texts of your author? The workflow for this section is:

```
library(tidytext)
library(dplyr)

# download the three books
book_zitkala <- gutenberg_download(338)
book_zerbe <- gutenberg_download(1445)
book_zangwill <- gutenberg_download(6304)
```

Each of the three books has its own format of chapter headings. So, I will process them respectively before combining them into a single dataset.

By checking the three books, we can see both `zangwill` and `zerbe` use chapter headings in the format of `Chapter I`, `Chapter II`, etc, while the chapter headings of `zangwill` are nested. However, the chapters in `zitkala` are like *IKTOMI AND THE DUCKS* and *IKTOMI'S BLANKET*. The different format of chapter headings requires different regular expressions to detect them.

Each of the three books needs to be processed separately, then combined into one dataset.

Procesing Zangwill's book

`Zangwill` is comprised of three **parts**. Each part contains multiple subsections. For the convenience of processing and analysis, I will regard each part as a chapter.

```
zangwill_clean <- book_zangwill %>%
  mutate(text = iconv(text, "latin1", "UTF-8", sub = "")) %>% # Convert the encoding
  filter(nchar(text) > 0) %>% # Remove empty lines
  mutate(
    book = "WITHOUT PREJUDICE",
    chapter = cumsum(str_detect(text, "^PART [IVX]+")), # Detect the chapter headings
  )

# Make all chapter numbers as characters
zangwill_clean <- zangwill_clean %>%
  mutate(chapter = as.character(chapter))
```

Processing Zerbe's book

The format of `Zerbe` is the most complicated among the three books. The chapter section in the front matter is in the format of `CHAPTER I`, `CHAPTER II`, etc., but also contains abstracts of each chapter. It makes the regex to detect the chapter headings more complicated.

To make the regex easier and more robust, the front matter of the books will be removed. The main text starts at line 332 of the df, so I'll use it as the starting point for slicing.

```
zerbe_clean <- book_zerbe %>%
  # Start from line 332
  slice(332:n()) %>%
  # Keep the text column
  select(gutenberg_id, text)

# Now the regex will be very simple because the chapter headings are in the format of "Chapter I", "Cha

# Add the columns of book title and chapter number
zerbe_clean <- zerbe_clean %>%
  mutate(
    book = "AEROPLANES",
    chapter = cumsum(str_detect(text, "(?i)^chapter [IVXLC]+"))
  )

zerbe_clean <- zerbe_clean %>%
  mutate(chapter = as.character(chapter))
```

Processing Zitkala's book

`Zitkala` does not have chapter numbers, but the chapter headings are in the format of `IKTOMI AND THE DUCKS`, `IKTOMI'S BLANKET`, etc. For this book, I need to manually add the chapter numbers.

```
zitkala_clean <- book_zitkala %>%
  filter(nchar(text) > 0) %>% # Remove empty lines
  mutate(book = "OLD INDIAN LEGENDS") %>% # Add the column of book title
```

```
  # Start calculating the chapter number by the structure
  mutate(
    chapter_num = case_when(
      # for the front matter, let the chapter number be 0
      row_number() <= which(text == "CONTENTS") |
        str_detect(text, "^     [A-Z]") ~ 0, # the format of content is all caps with 5 spaces indent
      # Main text part
      TRUE ~ cumsum( # only +1 when there is a real chapter heading
        str_detect(text, "^[A-Z][A-Z\\s,\\-]+$") & # chapter headings in main text don't use indent
        !str_detect(text, "^     ") & # double check the indent
        text != "OLD INDIAN LEGENDS" # exclude the book title
      )
    )
  ) %>%
  # fill the chapter number for the main text
  group_by(chapter_num) %>%
  fill(chapter_num) %>%
  ungroup() %>%
  # final chapter number
  mutate(
    chapter = if_else(chapter_num == 0,
                      "0",
                      paste(chapter_num))
  ) %>%
  select(gutenberg_id, text, book, chapter)
```

We can do a quick check to see if the chapter headings are correctly detected.

```
head(zangwill_clean, 5)
```

```
## # A tibble: 5 x 4
##   gutenberg_id text                                                book  chapter
##          <int> <chr>                                               <chr> <chr>
## 1         6304 "William Fishburne, Charles Aldarondo, and the Onl~ WITH~ 0
## 2         6304 "Proofreading Team."                                WITH~ 0
## 3         6304 "WITHOUT PREJUDICE"                                 WITH~ 0
## 4         6304 "BY I. ZANGWILL"                                    WITH~ 0
## 5         6304 "Author Of \"The Master,\" \"Children Of The Ghett~ WITH~ 0
```

```
tail(zangwill_clean, 5)
```

```
## # A tibble: 5 x 4
##   gutenberg_id text                                                book  chapter
##          <int> <chr>                                               <chr> <chr>
## 1         6304 "seems! Even the sandwich-men have lost their dole~ WITH~ 3
## 2         6304 "stirring in their boards. They are dreaming of th~ WITH~ 3
## 3         6304 "when they edited magazines or played \"Hamlet.\" ~ WITH~ 3
## 4         6304 "pen again to tell you how I dropped it, let me no~ WITH~ 3
## 5         6304 "bidding you a fond and last farewell--without pre~ WITH~ 3
```

```r
head(zerbe_clean, 5)
```

```
## # A tibble: 5 x 4
##   gutenberg_id text                                book       chapter
##          <int> <chr>                               <chr>      <chr>
## 1         1445 "CHAPTER I"                         AEROPLANES 1
## 2         1445 ""                                  AEROPLANES 1
## 3         1445 "THEORIES AND FACTS ABOUT FLYING"   AEROPLANES 1
## 4         1445 ""                                  AEROPLANES 1
## 5         1445 ""                                  AEROPLANES 1
```

```r
tail(zerbe_clean, 5)
```

```
## # A tibble: 5 x 4
##   gutenberg_id text                                                book  chapter
##          <int> <chr>                                               <chr> <chr>
## 1         1445 "Warping. The twist given to certain portions of p~ AERO~ 15
## 2         1445 "cause the air to aet against the warped portions." AERO~ 15
## 3         1445 ""                                                  AERO~ 15
## 4         1445 "Weight. The measure of the force which gravity ex~ AERO~ 15
## 5         1445 "objects."                                          AERO~ 15
```

```r
head(zitkala_clean, 5)
```

```
## # A tibble: 5 x 4
##   gutenberg_id text                          book               chapter
##          <int> <chr>                         <chr>              <chr>
## 1          338 "OLD INDIAN LEGENDS"          OLD INDIAN LEGENDS 0
## 2          338 "Retold By Zitkala-Sa"        OLD INDIAN LEGENDS 0
## 3          338 "ITKALA-SA."                  OLD INDIAN LEGENDS 0
## 4          338 "CONTENTS"                    OLD INDIAN LEGENDS 0
## 5          338 "     IKTOMI AND THE DUCKS"   OLD INDIAN LEGENDS 0
```

```r
tail(zitkala_clean, 5)
```

```
## # A tibble: 5 x 4
##   gutenberg_id text                                                book  chapter
##          <int> <chr>                                               <chr> <chr>
## 1          338 He struck his knife upward in the Eater's stomach,~ OLD ~ 14
## 2          338 out drowned those people of the village.            OLD ~ 14
## 3          338 Now when the great water fell into its own bed, th~ OLD ~ 14
## 4          338 came to the shore. They went home painted victors ~ OLD ~ 14
## 5          338 singers.                                            OLD ~ 14
```

```r
# Combine all books
all_books <- bind_rows(
  zangwill_clean,
  zerbe_clean,
  zitkala_clean
)

glimpse(all_books)
```

```
## Rows: 20,088
## Columns: 4
## $ gutenberg_id <int> 6304, 6304, 6304, 6304, 6304, 6304, 6304, 6304, 6304, 630~
## $ text         <chr> "William Fishburne, Charles Aldarondo, and the Online Dis~
## $ book         <chr> "WITHOUT PREJUDICE", "WITHOUT PREJUDICE", "WITHOUT PREJUD~
## $ chapter      <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0~
```

The result shows the dataset is securely combined. The nest step is to tokenise it and make the one-token-per-row format.

```
tidy_books <- all_books %>%
  unnest_tokens(word, text)
```

Remove stop words

```
data(stop_words)

tidy_books <- tidy_books %>%
  anti_join(stop_words)
```

Calculate the word frequency & Visualisation

- Now create a comparison plot as in section 1.5 of Text Mining with R. In your plot, compare your author against Jane Austen and another author of your choice. Include the plot, and also report the results of a correlation test, as in section 1.5. Are the results what you would expect?

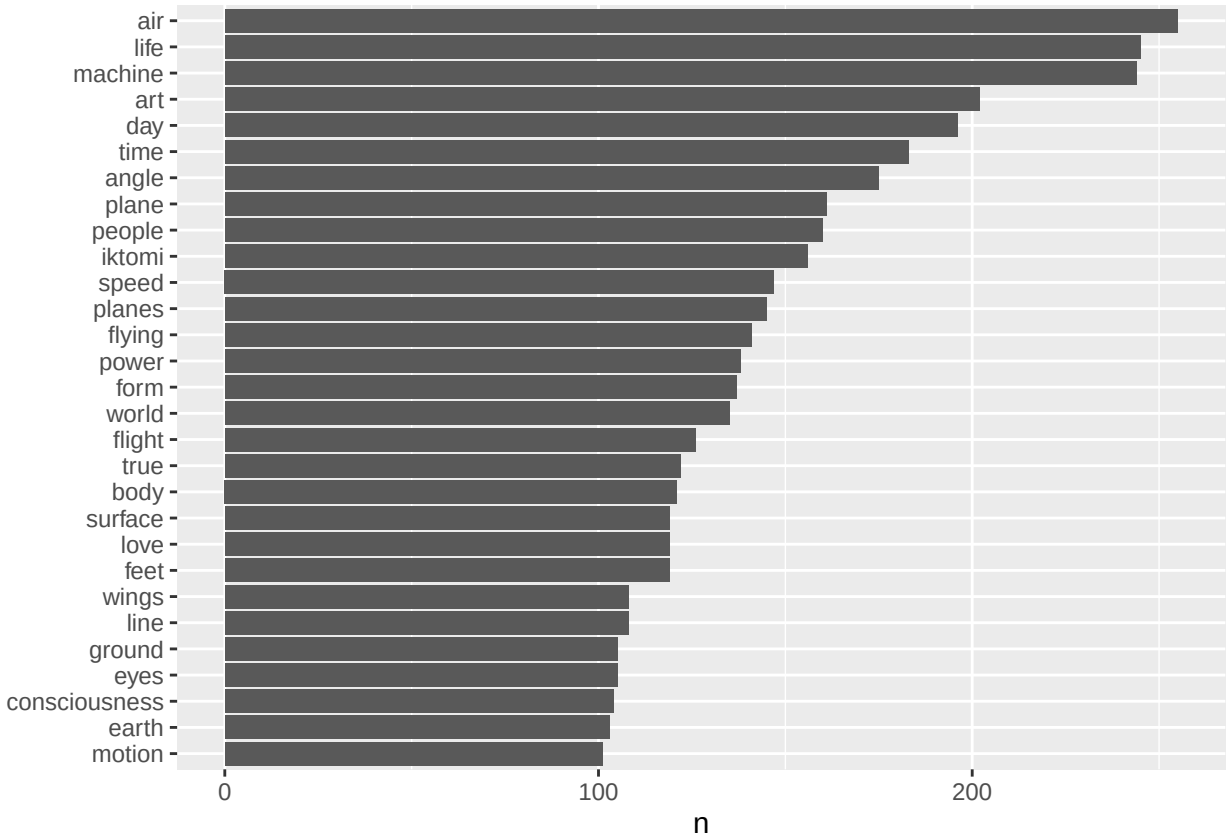
First, check the general word frequency

```
tidy_books %>%
  count(word, sort = TRUE)
```

```
## # A tibble: 17,836 x 2
##    word        n
##    <chr>   <int>
##  1 air       255
##  2 life      245
##  3 machine   244
##  4 art       202
##  5 day       196
##  6 time      183
##  7 angle     175
##  8 plane     161
##  9 people    160
## 10 iktomi    156
## # i 17,826 more rows
```

```
tidy_books %>%
  count(word, sort = TRUE) %>%
  filter(n > 100) %>%  #filter the words with frequency > 600
```

```
  mutate(word = reorder(word, n)) %>% #reorder the words by the frequency
  ggplot(aes(n, word))+
           geom_col()+
           labs(y = NULL)
```



The first 9 out of 10 most frequent words are all commonly used. The 10th one, *Iktomi*, is the name of a god in the book *OLD INDIAN LEGENDS*.

For author comparison, I will first compare the three authors with Jane Austen, then compare the three authors with each other.

Compare the three authors with Jane Austen

Step 1: prepare the Jane Austen dataset

```
book_ja <- austen_books() %>%
  group_by(book) %>%
  mutate(
    chapter = cumsum(str_detect(text, regex("^chapter [\\divxlc]",
                                            ignore_case = TRUE)))
  ) %>%
  ungroup() %>%
  unnest_tokens(word, text, drop = TRUE) %>%
  anti_join(stop_words)
```

```
book_ja <- book_ja %>%
  mutate(chapter = as.character(chapter))
```

The shapes of two datasets are different. The dataset of `all_books` has an extra column of *gutenberg_id* which book_ja does no have. I need to normalise the Jane Austen dataset to make it comparable with the three authors dataset.

I think it is better to add a new column of *id* to the Jane Austen dataset instead of removing it from the three authors dataset. Since the column of gutenberg_is is not directly useful for frequency analysis, I will use **4242** as the id for Jane Austen.

```
book_ja <- book_ja %>%
  mutate(gutenberg_id = 4242) %>%  # maintain data consistency
  relocate(gutenberg_id)
```

Step 2: combine the Jane Austin dataset with the three authors dataset

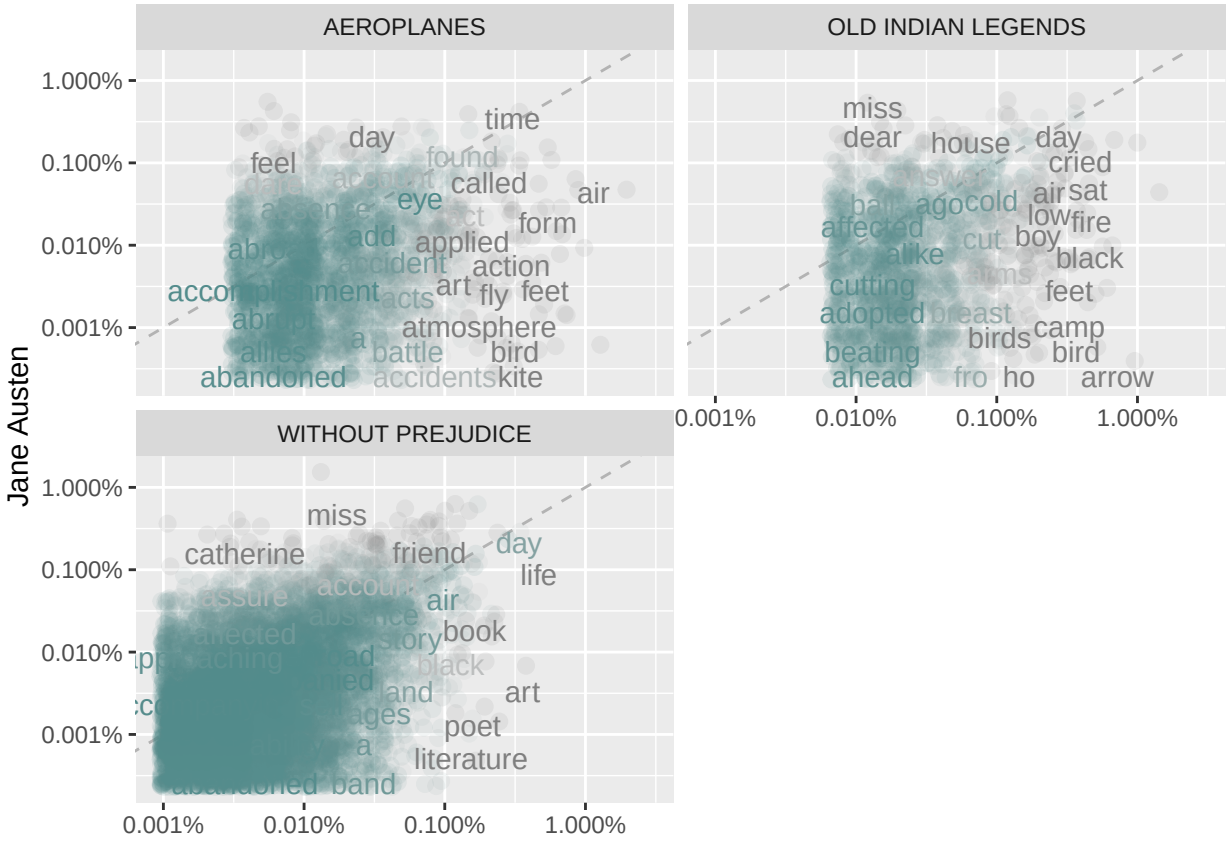
```
freq_comparison <- bind_rows(
  # Add tags for the three books
  tidy_books %>%
    filter(book %in% c("WITHOUT PREJUDICE",
                       "AEROPLANES",
                       "OLD INDIAN LEGENDS")) %>%
    mutate(source = book),  # use their book titles as tags

  # add Jane Austen's data as a reference
  book_ja %>%
    mutate(source = "Jane Austen")  # Universal tag for Jane Austen
) %>%

  # Extract only the words
  mutate(word = str_extract(word, "[a-z]+")) %>%
  # Remove NA values
  filter(!is.na(word)) %>%
  # calculate the frequency of each word
  count(source, word) %>%
  # Calculate the proportion based on the source
  group_by(source) %>%
  mutate(proportion = n / sum(n)) %>%
  # remove the original count column and keep the proportion column
  select(-n) %>%
  # Data reshaping -- long to wide
  pivot_wider(names_from = source, values_from = proportion) %>%
  # Data reshaping -- wide to long
  pivot_longer(
    c("WITHOUT PREJUDICE", "AEROPLANES", "OLD INDIAN LEGENDS"),
    names_to = "book",
    values_to = "proportion"
  )

# show results
freq_comparison %>%
  # Remove NA values
```

```
  filter(!is.na(proportion), !is.na(`Jane Austen`)) %>%
  # Plot the results
  mutate(difference = proportion - `Jane Austen`) %>%
  head(20)  # check the first 20 rows
```

```
## # A tibble: 20 x 5
##    word       `Jane Austen` book              proportion difference
##    <chr>              <dbl> <chr>                  <dbl>      <dbl>
##  1 a             0.0000138  WITHOUT PREJUDICE  0.000266   0.000252
##  2 a             0.0000138  AEROPLANES         0.000243   0.000229
##  3 abandoned     0.00000460 WITHOUT PREJUDICE  0.0000380  0.0000334
##  4 abandoned     0.00000460 AEROPLANES         0.0000607  0.0000561
##  5 ability       0.0000138  WITHOUT PREJUDICE  0.0000760  0.0000622
##  6 ability       0.0000138  AEROPLANES         0.000486   0.000472
##  7 abroad        0.000166   WITHOUT PREJUDICE  0.000152  -0.0000136
##  8 abroad        0.000166   AEROPLANES         0.0000607 -0.000105
##  9 abrupt        0.0000230  AEROPLANES         0.0000607  0.0000377
## 10 absence       0.000511   WITHOUT PREJUDICE  0.000266  -0.000245
## 11 absence       0.000511   AEROPLANES         0.000121  -0.000389
## 12 absolute      0.000120   WITHOUT PREJUDICE  0.0000950 -0.0000246
## 13 absolute      0.000120   AEROPLANES         0.0000607 -0.0000589
## 14 absolutely    0.000437   WITHOUT PREJUDICE  0.000247  -0.000190
## 15 absolutely    0.000437   AEROPLANES         0.000850   0.000413
## 16 absorbing     0.00000460 WITHOUT PREJUDICE  0.0000190  0.0000144
## 17 absorbing     0.00000460 AEROPLANES         0.0000607  0.0000561
## 18 access        0.00000460 WITHOUT PREJUDICE  0.0000190  0.0000144
## 19 access        0.00000460 AEROPLANES         0.0000607  0.0000561
## 20 accident      0.000110   WITHOUT PREJUDICE  0.000190   0.0000796
```

```
# I copied and pasted the code from the book with necessary modifications
ggplot(freq_comparison, aes(x = proportion,
                            y = `Jane Austen`,
                            color = abs(`Jane Austen` - proportion))) +

  # Add a diagonal line for reference
  geom_abline(color = "gray70", lty = 2) +

  # Add data points with random noise
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +

  # Add text labels to the words
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +

  # Dont understand this part
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +

  # Set up colour
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = 'gray75') +

  # Creating separate plots for each book
  facet_wrap(~book, ncol = 2) +
```

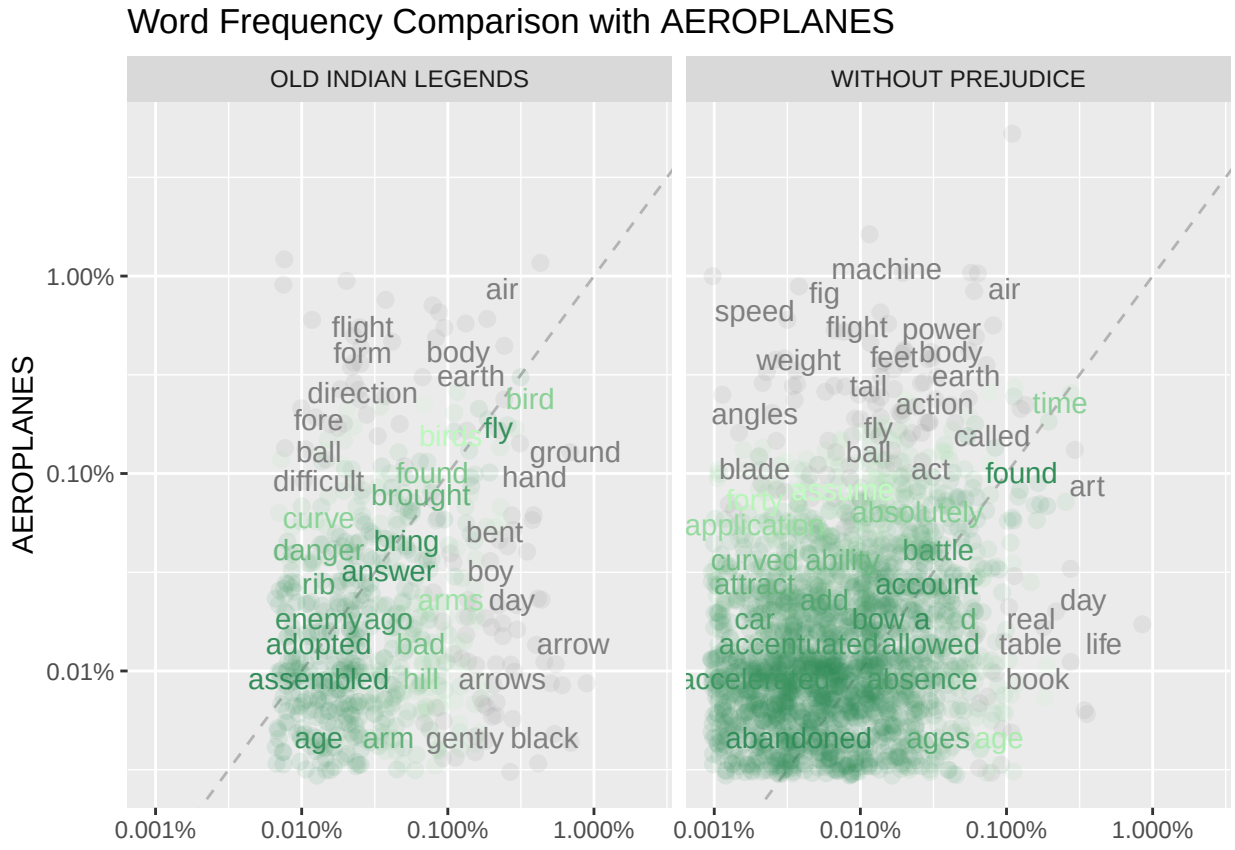
```
  # Removing the colour legend and setting axis labels
  theme(legend.position = "none") +
  labs(y = "Jane Austen", x = NULL)
```



Step 3: Use the book `zerbe` as the reference to compare the other two books

```
# Reshape the data to use AEROPLANES as reference
freq_aero_comparison <- bind_rows(
    tidy_books %>%
    filter(book %in% c("WITHOUT PREJUDICE", "AEROPLANES", "OLD INDIAN LEGENDS")) %>%
    mutate(source = book)
) %>%
    mutate(word = str_extract(word, "[a-z]+")) %>%
    count(source, word) %>%
    group_by(source) %>%
    mutate(proportion = n / sum(n)) %>%
    select(-n) %>%
    pivot_wider(names_from = source, values_from = proportion) %>%
    pivot_longer(
        c("WITHOUT PREJUDICE", "OLD INDIAN LEGENDS"),
        names_to = "book",
        values_to = "proportion"
    )

# Use the same vis code
```

```r
ggplot(freq_aero_comparison,
       aes(x = proportion,
           y = AEROPLANES, # Proportion in AEROPLANES (new reference)
           color = abs(AEROPLANES - proportion))) +

    # Add a diagonal line for reference
    geom_abline(color = "gray70", lty = 2) +

    # Add data points with random noise
    geom_jitter(alpha = 0.1,
                size = 2.5,
                width = 0.3,
                height = 0.3) +

    # Add text labels to the words
    geom_text(aes(label = word),
              check_overlap = TRUE,
              vjust = 1.5) +

    # Dont understand this part
    scale_x_log10(labels = percent_format()) +
    scale_y_log10(labels = percent_format()) +

    # Set up colour
    scale_color_gradient(limits = c(0, 0.001),
                         low = "seagreen4",
                         high = 'darkseagreen1') + # Changed the colour

    # Create separate plots for each comparison book
    facet_wrap(~book, ncol = 2) +

    # Remove legend and set axis labels
    theme(legend.position = "none") +
    labs(y = "AEROPLANES",
         x = NULL,
         title = "Word Frequency Comparison with AEROPLANES")
```



Step 4: Correlation Test The textbook uses tidy formula notation being different from the one in R Doc.

```
# tidy style cor test
co_wp <- cor.test(data = freq_comparison[freq_comparison$book == "WITHOUT PREJUDICE", ],
        ~ proportion + `Jane Austen`)

co_oil <- cor.test(data = freq_comparison[freq_comparison$book == "OLD INDIAN LEGENDS", ],
        ~ proportion + `Jane Austen`)

co_aero <- cor.test(data = freq_comparison[freq_comparison$book == "AEROPLANES", ],
        ~ proportion + `Jane Austen`)

# Print the results
cat("\nThe correlation report of book Without Prejudice is:\n")
```

```
##
## The correlation report of book Without Prejudice is:
```

```
print(co_wp)
```

```
##
##  Pearson's product-moment correlation
##
## data:  proportion and Jane Austen
## t = 39.78, df = 6911, p-value < 2.2e-16
```

```
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.4122654 0.4506325
## sample estimates:
##       cor
## 0.4316441
```

```
cat("\nThe correlation report of book Old Indian Legends is:\n")
```

```
##
## The correlation report of book Old Indian Legends is:
```

```
print(co_oil)
```

```
##
##  Pearson's product-moment correlation
##
## data:  proportion and Jane Austen
## t = 9.4062, df = 1470, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1894800 0.2858785
## sample estimates:
##      cor
## 0.238266
```

```
cat("\nThe correlation report of book Aeroplanes is:\n")
```

```
##
## The correlation report of book Aeroplanes is:
```

```
print(co_aero)
```

```
##
##  Pearson's product-moment correlation
##
## data:  proportion and Jane Austen
## t = 6.9221, df = 2186, p-value = 5.826e-12
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1051958 0.1872117
## sample estimates:
##       cor
## 0.1464554
```

- We extracted the chapter headings, but haven't used them yet. Think of some analysis or visualization that uses the chapter numbers; e.g., you could plot the chapter lengths.

For this task, I will calculate the distribution of modal verbs across the chapters of the three books. Modal verbs are a type of auxiliary verbs that express necessity, possibility, permission, or ability. The modal verbs

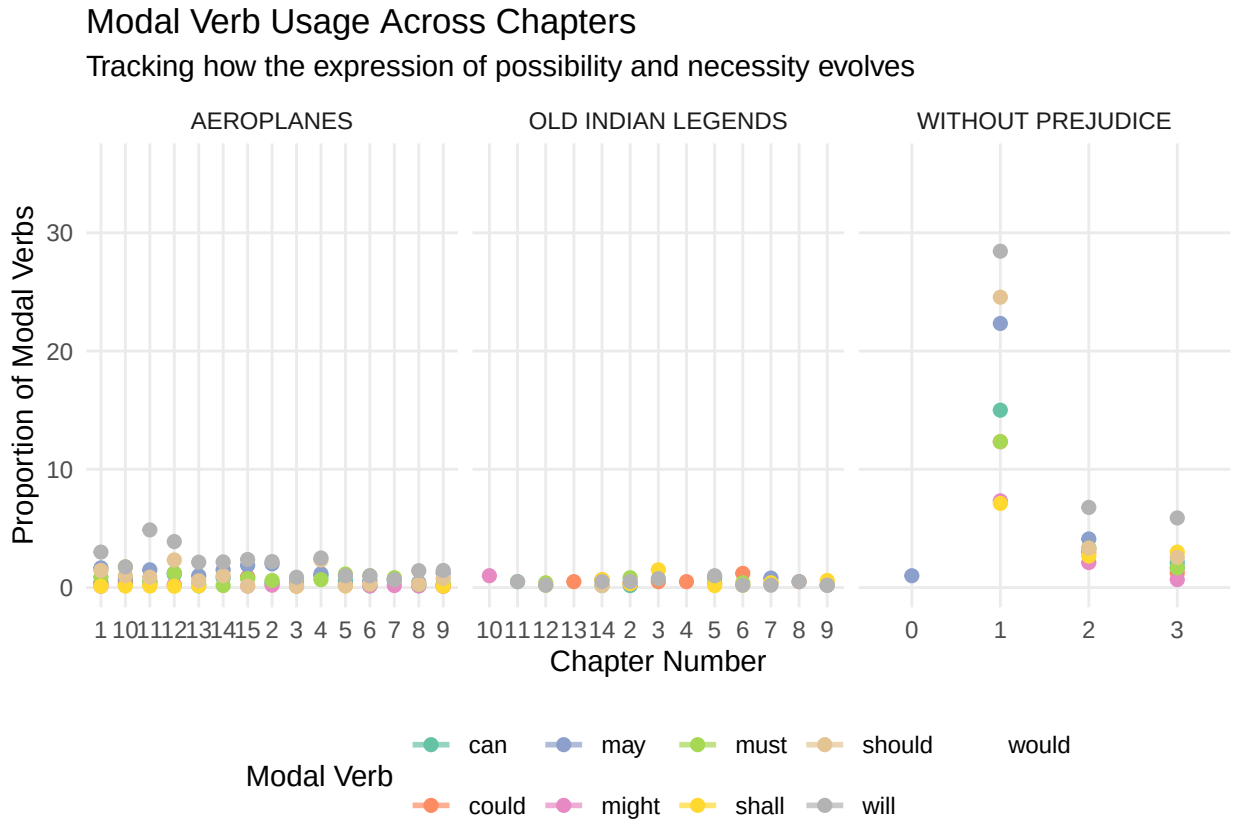
include *can, could, may, might, shall, should, will, would, must.* Ought to* is considered as a semi-modal verb and so not included in this analysis.

To do this analysis, the stop words need to be kept.

```
# Set up a df for modal verbs analysis
tidy_books_modal <- all_books %>%
  unnest_tokens(word, text)
```

```
# Set up the vector of modal verbs
modal_verbs <- c("can", "could", "will",
                 "would", "shall", "should",
                 "may", "might", "must")

# Analysis by chapter
modal_analysis <- tidy_books_modal %>%
  filter(book %in% c("WITHOUT PREJUDICE",
                     "AEROPLANES",
                     "OLD INDIAN LEGENDS")) %>%
  # Only keep the modal verbs
  filter(word %in% modal_verbs) %>%
  # Count occurrences by book, chapter, and specific modal verb
  count(book, chapter, word) %>%
  # Add book length information for normalisation
  group_by(book, chapter) %>%
  mutate(
    # Calculate the total words in each chapter for normalisation
    chapter_total = n(),
    # Calculate the proportion of each modal verb
    proportion = n / chapter_total
  ) %>%
  ungroup()

# Visualisation
ggplot(modal_analysis,
       aes(x = chapter,
           y = proportion,
           color = word)) +
  # Use lines to show the progression
  geom_line(size = 1, alpha = 0.7) +
  # Add points for specific values
  geom_point(size = 2) +
  # Separate plots for each book
  facet_wrap(~book, scales = "free_x") +
  scale_color_brewer(palette = "Set2") +
  theme_minimal() +
  labs(
    title = "Modal Verb Usage Across Chapters",
    subtitle = "Tracking how the expression of possibility and necessity evolves",
    x = "Chapter Number",
    y = "Proportion of Modal Verbs",
    color = "Modal Verb"
  ) +
  # Improve readability
```

```
  theme(
    legend.position = "bottom",
    panel.grid.minor = element_blank()
  )
```



The End