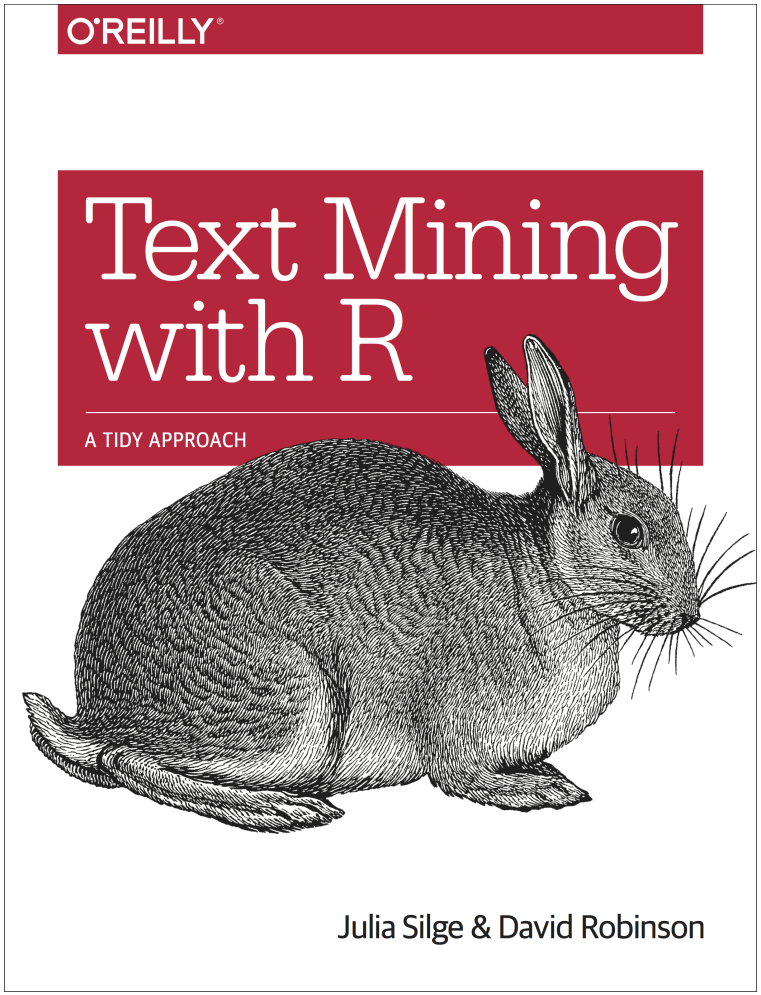
Socio-Informatics 348

Text Analysis
Tidy Text

Dr Lisa Martin

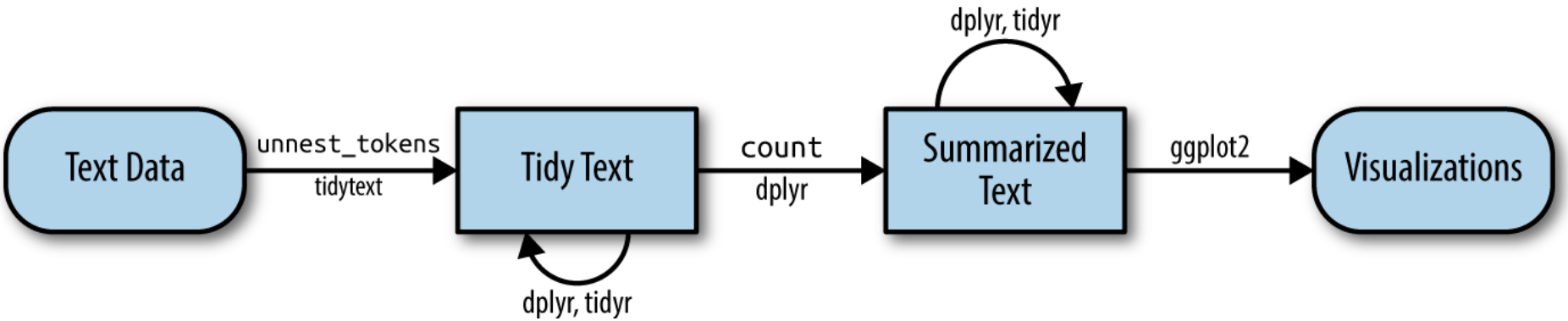
Department of Information Science
Stellenbosch University

Today's Reading



Text Mining with R, Chapter 1

Tidy data & text



Tidy data & text

- Tidy data principle (Wickham):
  - Each variable is a column
  - Each observation is a row
  - Each type of observational unit is a table
- **Tidy text format**: one token per row (word, n-gram, sentence, etc.)
- This contrasts with other text structures:
  - Strings / character vectors
  - Corpus objects
  - Document-term matrices

Tokenization with `unnest_tokens`

A token is a meaningful unit of text, most often a word, that we are interested in using for further analysis, and tokenization is the process of splitting text into tokens.

- The function `unnest_tokens(output, input)` splits a text column into tokens
- Default: word-level tokens, lowercased, punctuation stripped
- Other token types: characters, n-grams, sentences, paragraphs, custom regex

Example: Emily Dickinson poem

```
text <- c("Because I could not stop for Death -",
          "He kindly stopped for me -",
          "The Carriage held but just Ourselves -",
          "and Immortality")

text
#> [1] "Because I could not stop for Death -"
#> [2] "He kindly stopped for me -"
#> [3] "The Carriage held but just Ourselves -"
#> [4] "and Immortality"
```

Example: Emily Dickinson poem

```
library(dplyr)
text_df <- tibble(line = 1:4, text = text)

text_df
#> # A tibble: 4 × 2
#>    line text
#>   <int> <chr>
#> 1     1 Because I could not stop for Death -
#> 2     2 He kindly stopped for me -
#> 3     3 The Carriage held but just Ourselves -
#> 4     4 and Immortality
```

Example: Emily Dickinson poem

One-token-per-row format:

```
library(tidytext)

text_df %>%
  unnest_tokens(word, text)
#> # A tibble: 20 × 2
#>     line word
#>    <int> <chr>
#>  1     1 because
#>  2     1 i
#>  3     1 could
#>  4     1 not
#>  5     1 stop
#>  6     1 for
#>  7     1 death
#>  8     2 he
#>  9     2 kindly
#> 10     2 stopped
```

Tidying Jane Austen's works

- Use `janeaustenr::austen_books()` — each line one row
- Add metadata: line number, chapter (via regex detection)

```
library(janeaustenr)
library(dplyr)
library(stringr)

original_books <- austen_books() %>%
  group_by(book) %>%
  mutate(linenumber = row_number(),
         chapter = cumsum(str_detect(text,
                                     regex("^chapter [\\divxlc]",
                                           ignore_case = TRUE)))) %>%
  ungroup()
```

Tidying Jane Austen's works

- Use `janeaustenr::austen_books()` — each line one row
- Add metadata: line number, chapter (via regex detection)

```
original_books
#> # A tibble: 73,422 × 4
#>    text                  book                linenumber chapter
#>    <chr>                 <fct>                    <int>   <int>
#>  1 "SENSE AND SENSIBILITY" Sense & Sensibility          1       0
#>  2 ""                    Sense & Sensibility          2       0
#>  3 "by Jane Austen"      Sense & Sensibility          3       0
#>  4 ""                    Sense & Sensibility          4       0
#>  5 "(1811)"              Sense & Sensibility          5       0
#>  6 ""                    Sense & Sensibility          6       0
#>  7 ""                    Sense & Sensibility          7       0
#>  8 ""                    Sense & Sensibility          8       0
#>  9 ""                    Sense & Sensibility          9       0
#> 10 "CHAPTER 1"           Sense & Sensibility         10       1
#> # ℹ 73,412 more rows
```

Tidying Jane Austen's works

- Then apply `unnest_tokens(word, text)` ß one-token-per-row
- Result: tidy_books with columns (book, linenumber, chapter, word)

```
library(tidytext)
tidy_books <- original_books %>%
  unnest_tokens(word, text)

tidy_books
#> # A tibble: 725,064 × 4
#>    book                linenumber chapter word
#>    <fct>                    <int>   <int> <chr>
#>  1 Sense & Sensibility          1       0 sense
#>  2 Sense & Sensibility          1       0 and
#>  3 Sense & Sensibility          1       0 sensibility
#>  4 Sense & Sensibility          3       0 by
#>  5 Sense & Sensibility          3       0 jane
#>  6 Sense & Sensibility          3       0 austen
#>  7 Sense & Sensibility          5       0 1811
#>  8 Sense & Sensibility         10       1 chapter
```

Removing stop words & counting

- Use dataset `stop_words` from the tidytext package
- Remove: `tidy_books > anti_join(stop_words)`

```
data(stop_words)

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

Removing stop words & counting

- Count term frequencies:

```
tidy_books %>%
  count(word, sort = TRUE)
#> # A tibble: 13,910 × 2
#>    word       n
#>    <chr>  <int>
#>  1 miss    1855
#>  2 time    1337
#>  3 fanny    862
#>  4 dear     822
#>  5 lady     817
#>  6 sir      806
#>  7 day      797
#>  8 emma     787
#>  9 sister   727
#> 10 house    699
#> # ℹ 13,900 more rows
```

Removing stop words & counting

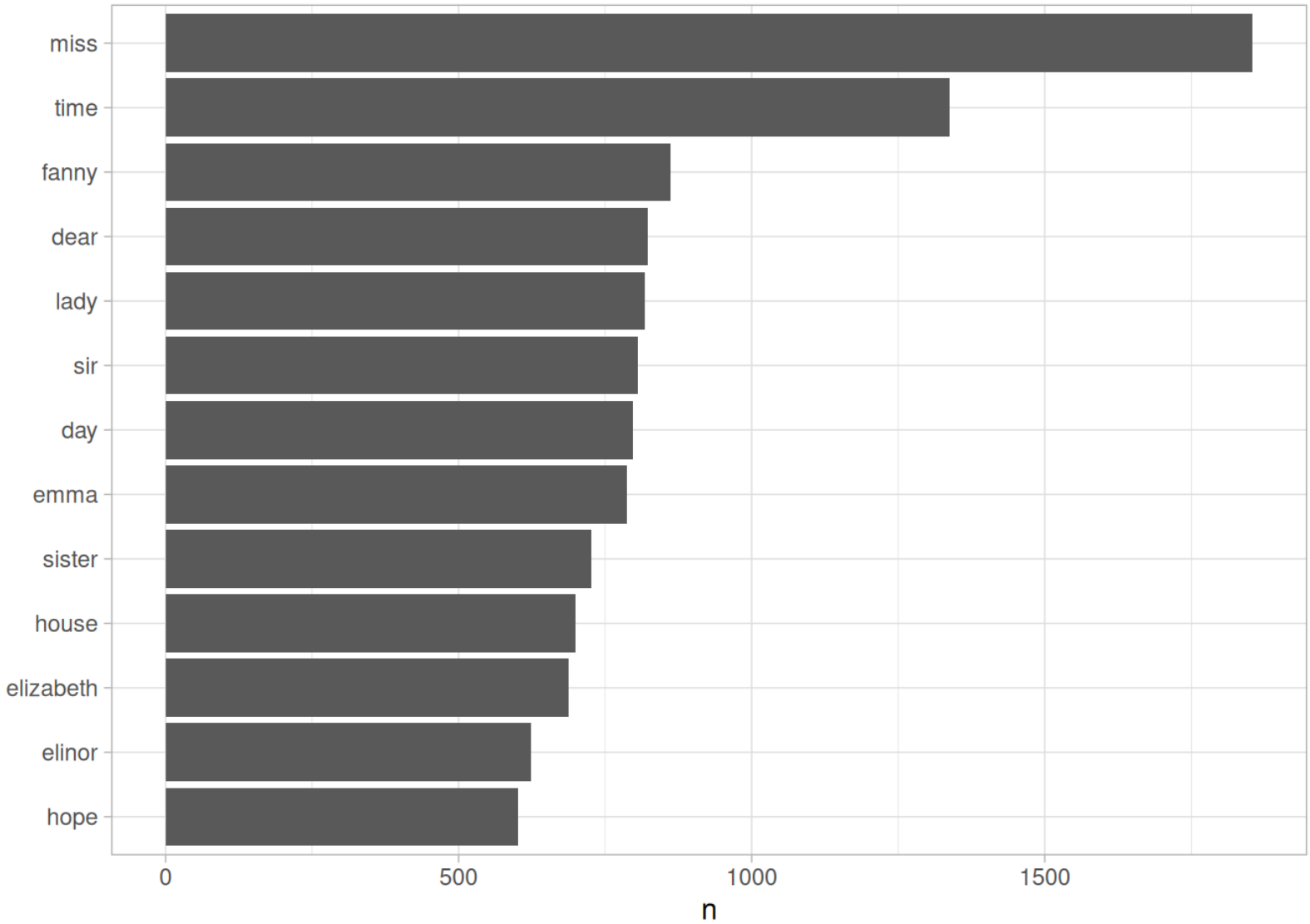
- Because data is tidy, you can pipe directly into `ggplot2`, etc.

```
library(ggplot2)

tidy_books %>%
  count(word, sort = TRUE) %>%
  filter(n > 600) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```

Removing stop words & counting

- Because data is tidy, you can pipe directly into `ggplot2`, etc.



Using Project Gutenberg texts

- Package: `gutenbergr`
- `gutenberg_download()` to fetch public domain books
- Strips header/footer metadata

Comparing word frequencies

Use the Project Gutenberg ID numbers for each novel:

```
library(gutenbergr)


hgwells <- gutenberg_download(c(35, 36, 5230, 159))
```

```
tidy_hgwells <- hgwells %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)
```

Comparing word frequencies

```
tidy_hgwells %>%
  count(word, sort = TRUE)
#> # A tibble: 11,768 × 2
#>    word       n
#>    <chr>  <int>
#>  1 time     454
#>  2 people   302
#>  3 door     260
#>  4 heard    249
#>  5 black    232
#>  6 stood    229
#>  7 white    222
#>  8 hand     218
#>  9 kemp     213
#> 10 eyes     210
#> # ℹ 11,758 more rows
```

Comparing word frequencies

Use the Project Gutenberg ID numbers for each novel:

```
bronte <- gutenberg_download(c(1260, 768, 969, 9182, 767))
```

```
tidy_bronte <- bronte %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)
```

Comparing word frequencies

```
tidy_bronte %>%
  count(word, sort = TRUE)
#> # A tibble: 23,042 × 2
#>    word       n
#>    <chr>  <int>
#>  1 time    1065
#>  2 miss     855
#>  3 day      827
#>  4 hand     768
#>  5 eyes     713
#>  6 night    647
#>  7 heart    638
#>  8 looked   602
#>  9 door     592
#> 10 half     586
#> # ℹ 23,032 more rows
```

Comparing word frequencies across authors

- Combine tidy datasets for different authors (e.g. Austen, Brontë, Wells)
- Compute relative frequencies:

$$\text{proportion} = \frac{n}{\sum n}$$

- Reshape (e.g. via `pivot_wider`, `pivot_longer`)
- Plot comparisons
- Correlation of frequencies: Austen vs Brontë $>$ Austen vs Wells

Comparing word frequencies across authors

```
library(tidyr)

frequency <- bind_rows(mutate(tidy_bronte, author = "Brontë Sisters"),
                       mutate(tidy_hgwells, author = "H.G. Wells"),
                       mutate(tidy_books, author = "Jane Austen")) %>%
  mutate(word = str_extract(word, "[a-z']+")) %>%
  count(author, word) %>%
  group_by(author) %>%
  mutate(proportion = n / sum(n)) %>%
  select(-n) %>%
  pivot_wider(names_from = author, values_from = proportion) %>%
  pivot_longer(`Brontë Sisters`:`H.G. Wells`,
               names_to = "author", values_to = "proportion")
```

Comparing word frequencies across authors

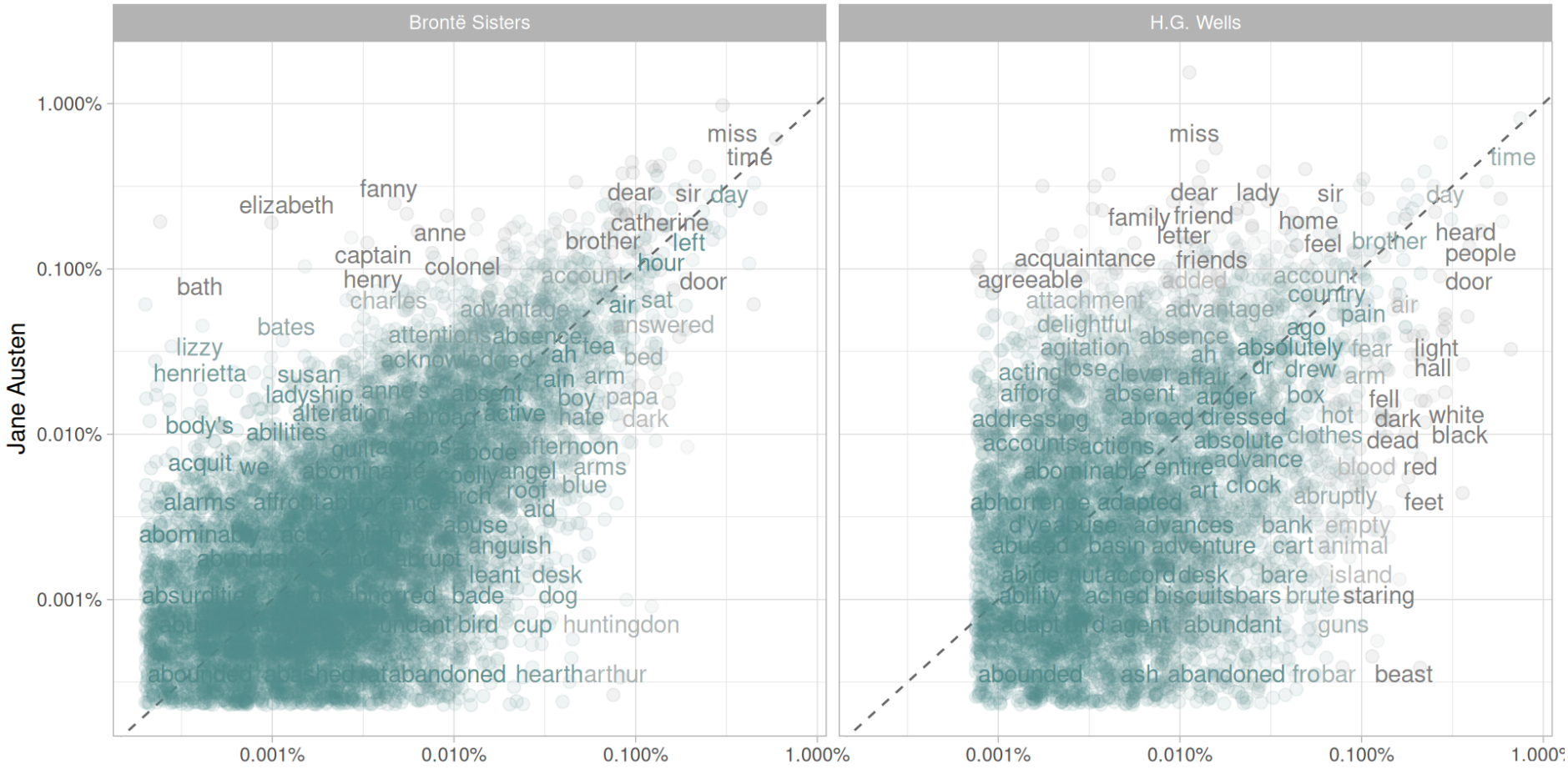
```
frequency
#> # A tibble: 57,812 × 4
#>    word    `Jane Austen` author          proportion
#>    <chr>           <dbl> <chr>                <dbl>
#>  1 a          0.00000919 Brontë Sisters  0.00000797
#>  2 a          0.00000919 H.G. Wells     NA
#>  3 a'most    NA          Brontë Sisters  0.0000159
#>  4 a'most    NA          H.G. Wells     NA
#>  5 aback     NA          Brontë Sisters  0.00000398
#>  6 aback     NA          H.G. Wells      0.0000150
#>  7 abaht     NA          Brontë Sisters  0.00000398
#>  8 abaht     NA          H.G. Wells     NA
#>  9 abandon   NA          Brontë Sisters  0.0000319
#> 10 abandon   NA          H.G. Wells      0.0000150
#> # ℹ 57,802 more rows
```

Comparing word frequencies across authors

```
library(scales)

# expect a warning about rows with missing values being removed
ggplot(frequency, aes(x = proportion, y = `Jane Austen`,
                      color = abs(`Jane Austen` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  facet_wrap(~author, ncol = 2) +
  theme(legend.position="none") +
  labs(y = "Jane Austen", x = NULL)
```

Comparing word frequencies across authors



Correlation of word usage

- Compute Pearson correlation between proportions of words between corpora
- Example results:
  - Austen vs Brontë: correlation ≈ 0.76
  - Austen vs Wells: correlation ≈ 0.42
- Interpretation: vocabularies are more similar between more related authors

Correlation of word usage

```
cor.test(data = frequency[frequency$author == "Brontë Sisters",],
         ~ proportion + `Jane Austen`)
#>
#>  Pearson's product-moment correlation
#>
#> data:  proportion and Jane Austen
#> t = 119.65, df = 10404, p-value < 2.2e-16
#> alternative hypothesis: true correlation is not equal to 0
#> 95 percent confidence interval:
#>  0.7527854 0.7689628
#> sample estimates:
#>       cor
#> 0.7609924
```

Correlation of word usage

```
cor.test(data = frequency[frequency$author == "H.G. Wells",],
         ~ proportion + `Jane Austen`)
#>
#>  Pearson's product-moment correlation
#>
#> data:  proportion and Jane Austen
#> t = 36.436, df = 6052, p-value < 2.2e-16
#> alternative hypothesis: true correlation is not equal to 0
#> 95 percent confidence interval:
#>  0.4032663 0.4445889
#> sample estimates:
#>       cor
#> 0.4241484
```