DSST 389: Final Project

Anna Phillips

2025-04-22

Introduction: Stephen Sondheim

Stephen Sondheim is one of Broadway's most beloved and successful lyricists/composers. He is a Tony Award, Academy Award, and Grammy Award winner, with a corpus of work ranging from Broadway musicals to television and movie scores. Sondheim's shows are known for their exploration of relationships, morality, and love – heavily facilitated by Sondheim's lyrical verbosity and musical complexity. In an interview with *The New York Times Magazine*, Sondheim reflected that "[he had] always conscientiously tried not to do the same thing twice," which was no easy feat when competing in a Broadway market that had turned "corporate" and "cookie-cutter."[1] Because of this, even though Sondheim grapples with similar themes in his shows, his characters and narratives feel brand new every time, ranging from homicidal barbers in the 18th century, to fairytale characters lost together in the forest, to musings on marriage in affluent New York social circles, to the complex inner world of 19th-century artists and their struggles to devote time outside of their work.

So what?

Sondheim's shows grapple with life's hard questions, and his lyrics often trend toward the existential,[2] allowing the audience to sympathize with broad themes – despite the often hyper-specific nature of Sondheim's shows. Interestingly, through initial data exploration, I identified 'love' as the most frequent term in the corpus as a whole, and I will use multiple textual analysis methods to explore the significance of 'love' in Sondheim's corpus. First, I will use tf-idf to explore important terms for each show, and then I will use sentiment analysis to provide more context for the shows and their relationship to each other before moving to topic modeling. With topic modeling, I will return to the topic of 'love' and explore important terms across

[1] Rich, Frank. 2000. "Conversations with Sondheim." *The New York Times Magazine*. https://www.nytimes.com/2000/03/12/magazine/conversations-with-sondheim.html

[2] Kohn, Alfie. 2000. "The Case for Sondheim as Existentialist." *The Sondheim Review*. https://www.alfiekohn.org/article/case-sondheim-existentialist/

the entire corpus, as well as between the contrasting individual shows. Sondheim is known for exploring dark themes and subject matter, so it may be surprising that love is the most prevalent topic – but when you explore his lyrics, you'll find that Sondheim explores multiple facets of love in his work (which includes darkness, revenge, and murder), and this complexity is what makes him so compelling as a lyricist.

Data Description

In order to perform my planned textual analysis, I had to create my own corpus of Sondheim musicals. To do this, I manually copy-and-pasted lines from eight shows into their own separate data sets and imported them to R. In terms of criteria, I chose his most famous shows, in combination with shows that have won the Tony Award for Best Musical.[3] In my data tidying, I removed stop words and character names, and I also edited iterations of similar words for clarity of analysis (e.g., all "pies" became "pie").

> **Note**
>
> The custom DSST 389 ChatGPT was utilized to help with my understanding of topic model limitations and most effective uses: https://chatgpt.com/share/6807c556-1fd8-800b-82fc-8ad04479347c

Data Tidying

```
# Read in data
merrily <- read.csv("../data/Merrily.csv")

woods <- read.csv("../data/Woods.csv")

sweeney <- read.csv("../data/Sweeney.csv")

company <- read.csv("../data/Company.csv")

sunday <- read.csv("../data/Sunday2.csv")

night <- read.csv("../data/Night.csv")

passion <- read.csv("../data/Passion.csv")
```

[3] A notable absence is *A Funny Thing Happened on the Way to the Forum*, which also won Best Musical, but I already had an even number of shows and I didn't want to disrupt the balance.

```r
follies <- read.csv("../data/Follies.csv")
```

```r
# Tidy datasets
merrily <- merrily |>
  mutate(linenumber = row_number()) |>
  rename(musical = Musical, song = Song, text = Text)

woods <- woods |>
  mutate(linenumber = row_number()) |>
  rename(musical = Musical, song = Song, text = Text)

sweeney <- sweeney |>
  mutate(linenumber = row_number()) |>
  rename(musical = Musical, song = Song, text = Text) |>
  mutate(song = if_else(song == "", "The Contest", song))

company <- company |>
  mutate(linenumber = row_number()) |>
  rename(musical = Musical, song = Song, text = Text) |>
  mutate(text = str_replace_all(text, "[’]", "'"))

sunday <- sunday |>
  mutate(linenumber = row_number()) |>
  rename(musical = Musical, song = Song, text = Text)

night <- night |>
  mutate(linenumber = row_number()) |>
  mutate(musical = if_else(musical == "", "A Little Night Music", musical))

passion <- passion |>
  mutate(linenumber = row_number())

follies <- follies |>
  mutate(linenumber = row_number())
```

Data Tokenization

I made a list of names from each show in order to avoid skewing the data based on the "rareness" of the character names. I also created a list of common Sondheim "stop words" like "ah" and "la."

```r
# Create list of common names for each show
names <- c(
  "charley", "frank", "mary", "beth", "sweeney", "fleet", "pirelli",
  "todd", "beadle", "johanna", "deedle", "toby", "anthony", "turpin",
  "lovett", "robert", "lucy", "bobby", "bubi", "bob", "harry", "amy",
  "paul", "joanne", "robby", "peter", "david", "susan", "sarah", "george",
  "louis", "louis's", "dot", "giorgio", "clara", "fosca", "jessie",
  "buddy", "tony's", "buddy's", "ben", "margie", "sally", "fredrik",
  "henrik", "anne", "petra", "desiree", "charlotte", "signora"
)

# Create list of common Sondheim stop words
sondheim_stop <- c(
  "la", "ah", "bah", "whadaya",
  "ding", "dong", "em", "gonna",
  "shh", "psst", "da", "bzzz"
)

# Tokenize and tidy data sets
merrily_tidy <- merrily |>
  unnest_tokens(word, text) |>
  anti_join(stop_words, join_by(word))

woods_tidy <- woods |>
  unnest_tokens(word, text) |>
  anti_join(stop_words, join_by(word))

sweeney_tidy <- sweeney |>
  unnest_tokens(word, text) |>
  anti_join(stop_words, join_by(word))

company_tidy <- company |>
  unnest_tokens(word, text) |>
  anti_join(stop_words, join_by(word))

sunday_tidy <- sunday |>
  unnest_tokens(word, text) |>
  anti_join(stop_words, join_by(word))

night_tidy <- night |>
  unnest_tokens(word, text) |>
  anti_join(stop_words, join_by(word))
```

```
passion_tidy <- passion |>
  unnest_tokens(word, text) |>
  anti_join(stop_words, join_by(word))

follies_tidy <- follies |>
  unnest_tokens(word, text) |>
  anti_join(stop_words, join_by(word))
```

```
# Create Sondheim corpus
sondheim_corpus <- bind_rows(
  merrily_tidy, woods_tidy,
  sweeney_tidy, company_tidy,
  sunday_tidy, night_tidy,
  passion_tidy, follies_tidy
)

# Edit strings for clearer analysis
sondheim_corpus <- sondheim_corpus |>
  filter(!(word %in% names), !(word %in% sondheim_stop)) |>
  mutate(word = str_replace_all(word, "'s", "")) |>
  mutate(word = str_replace_all(word, "pies", "pie")) |>
  anti_join(stop_words, join_by(word))

# Factor shows for order
sondheim_corpus <- sondheim_corpus |>
  mutate(musical = factor(musical,
    levels = c(
      "Follies", "Passion", "Company",
      "Into the Woods", "A Little Night Music",
      "Merrily We Roll Along",
      "Sunday in the Park with George",
      "Sweeney Todd: The Demon Barber of Fleet Street"
    )
  ))
```

```
# Create list of show lengths
sondheim_lengths <- sondheim_corpus |>
  group_by(musical) |>
  summarize(article_length = n()) |>
  ungroup()
```

```r
sondheim_lengths |>
  arrange(desc(article_length))
```

Table 1: Length (in words) of each Sondheim musical

musical	article_length
Sweeney Todd: The Demon Barber of Fleet Street	3255
Into the Woods	2537
Merrily We Roll Along	2400
Sunday in the Park with George	1975
Company	1718
A Little Night Music	1712
Passion	1482
Follies	1478

Data Exploration

```r
# Data exploration: most common words
word_ranks <- sondheim_corpus |>
  count(musical, word, sort = TRUE) |>
  group_by(musical) |>
  mutate(rank = row_number())

word_ranks |>
  group_by(musical) |>
  slice_head(n = 5)
```

Table 2: Most common words in each Sondheim musical

musical	word	n	rank
Follies	love	40	1
Follies	girls	24	2
Follies	leave	23	3
Follies	mirror	21	4
Follies	babe	17	5
Passion	love	90	1
Passion	live	23	2
Passion	happiness	22	3

musical	word	n	rank
Passion	day	20	4
Passion	feel	20	5
Company	married	37	1
Company	love	35	2
Company	person	27	3
Company	baby	24	4
Company	people	20	5
Into the Woods	woods	112	1
Into the Woods	mother	30	2
Into the Woods	festival	29	3
Into the Woods	prince	25	4
Into the Woods	moment	24	5
A Little Night Music	weekend	39	1
A Little Night Music	country	36	2
A Little Night Music	remember	19	3
A Little Night Music	life	13	4
A Little Night Music	day	12	5
Merrily We Roll Along	friends	40	1
Merrily We Roll Along	day	34	2
Merrily We Roll Along	time	33	3
Merrily We Roll Along	dreams	28	4
Merrily We Roll Along	mutter	28	5
Sunday in the Park with George	sunday	48	1
Sunday in the Park with George	art	38	2
Sunday in the Park with George	blue	33	3
Sunday in the Park with George	hat	27	4
Sunday in the Park with George	red	25	5
Sweeney Todd: The Demon Barber of Fleet Street	sir	61	1
Sweeney Todd: The Demon Barber of Fleet Street	pie	36	2
Sweeney Todd: The Demon Barber of Fleet Street	pretty	28	3
Sweeney Todd: The Demon Barber of Fleet Street	god	27	4
Sweeney Todd: The Demon Barber of Fleet Street	world	24	5

```
sondheim_corpus |>
  count(word, sort = TRUE) |>
  slice_head(n = 10)
```

Table 3: Most common words in entire Sondheim corpus

word	n
love	218
day	119
woods	112
time	99
life	80
sir	72
feel	70
wait	68
woman	66
beautiful	64

In an interview with *The Paris Review*, Sondheim said, "Two of the hardest words in the language to rhyme are life and love. Of all words!"[4] Unfortunately for Sondheim, as shown in Table 3, 'love' and 'life' are both in the top five most-used words of his lyrical corpus. As discussed in the introduction, love is one of Sondheim's most explored themes, so this is not a surprise. Notably, because 'love' and 'life' have high term frequencies, when we compare terms across each show (document) using tf-idf, these terms will have low scores. This will boost terms specific to each corpus to help us showcase how the characters/narratives in each show are unique. After exploring the data with tf-idf, I will use sentiment analysis to help accentuate the differences in sentiment between shows, in order to prepare for interpreting topic models.

tf-idf

```
# tf-idf preparation
n_docs <- sondheim_corpus |>
  select(musical) |>
  n_distinct()

idf <- sondheim_corpus |>
  distinct(musical, word) |>
  count(word, name = "word_docs") |>
  mutate(idf = log(n_docs / word_docs))
```

[4]Lipton, James. 1997. "Stephen Sondheim, The Art of the Musical." *The Paris Review*. https://www.theparisreview.org/interviews/1283/the-art-of-the-musical-stephen-sondheim

```r
# Top five tf-idf by show
labels <- sondheim_corpus |>
  count(musical, word) |>
  left_join(sondheim_lengths, join_by(musical)) |>
  mutate(tf = n / article_length) |>
  left_join(idf, join_by(word)) |>
  mutate(tf_idf = tf * idf) |>
  group_by(musical) |>
  arrange(desc(tf_idf)) |>
  slice_head(n = 5)

# Visualize tf-idf
sondheim_corpus |>
  count(musical, word) |>
  left_join(sondheim_lengths, join_by(musical)) |>
  mutate(tf = n / article_length) |>
  left_join(idf, join_by(word)) |>
  mutate(tf_idf = tf * idf) |>
  arrange(desc(tf_idf)) |>
  filter(tf_idf > 0) |>
  ggplot(aes(x = musical, y = tf_idf, fill = musical)) +
  geom_violin() +
  geom_label_repel(data = labels, aes(label = word), max.overlaps = 20) +
  scale_y_log10() +
  scale_fill_brewer(palette = "Set2") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  guides(fill = guide_legend(override.aes = aes(color = NA))) +
  theme(legend.position = "none") +
  labs(title = "tf-idf of popular Sondheim musicals", x = "Musical")
```

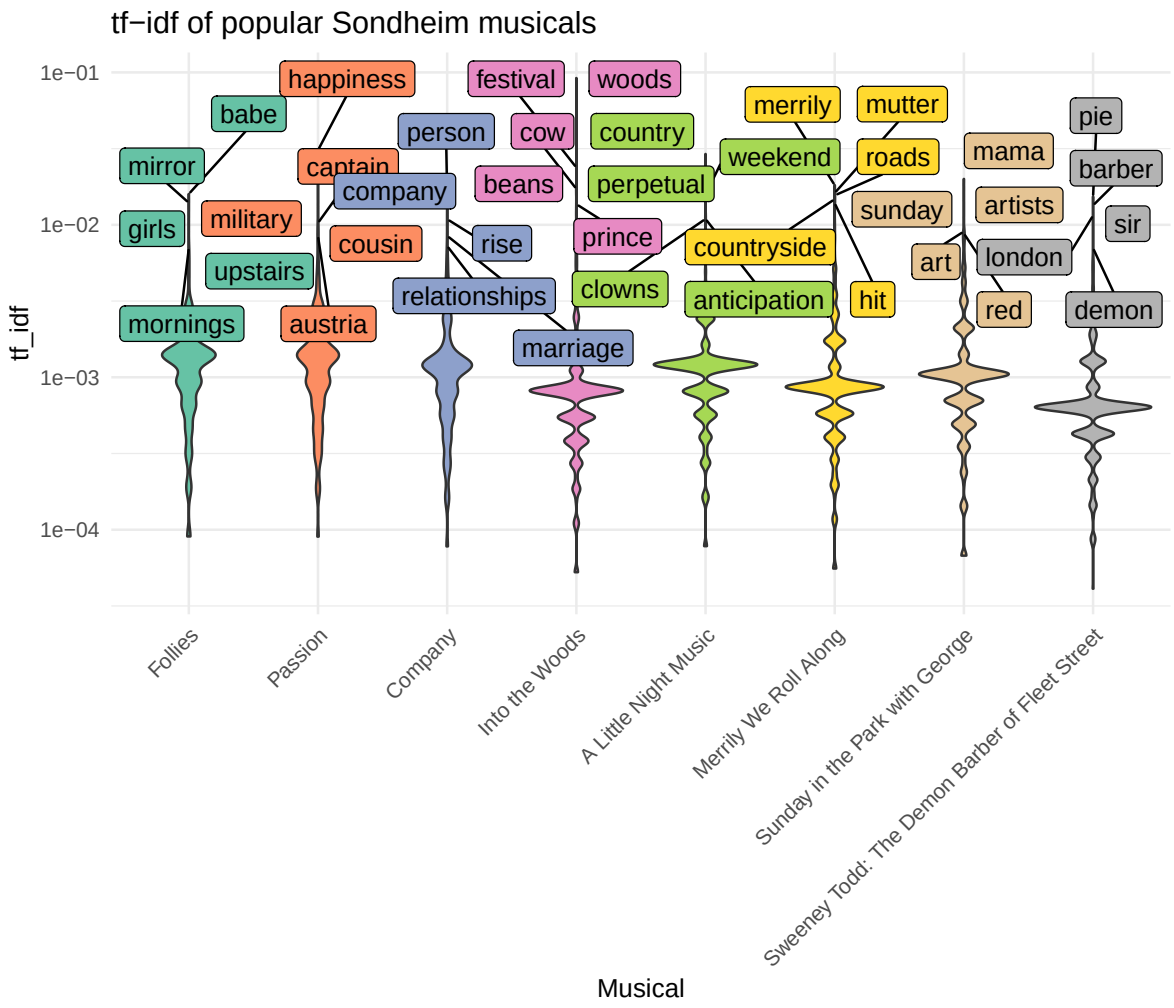

Figure 1: tf-idf scores for Sondheim's musicals.

As shown in Figure 1, the results of the tf-idf point to the highly specific nature of Sondheim's shows (e.g., *Sweeney Todd* and *Into the Woods*). Other shows have intriguing terms with high scores like "perpetual" in *A Little Night Music* or "mutter" in *Merrily We Roll Along*. However, these scores can be attributed to one song in each show skewing the entire corpus evaluation (i.e., "Perpetual Anticipation" and "Franklin Shepard, Inc.," respectively). Next, I'll use sentiment analysis to explore each show's overall sentiment to help provide further context for the topic models I'll create later.

Sentiment Analysis: afinn

```
# Load sentiment lexicon
afinn <- get_sentiments("afinn")

# Compute sentiment scores for each song
sentiment_scores <- sondheim_corpus |>
  left_join(afinn, join_by(word)) |>
  group_by(song, musical) |>
  summarise(
    afinn_net = sum(value, na.rm = TRUE),
    lyric_len = n(),
    afinn_normalized = afinn_net / lyric_len, .groups = "drop"
  ) |>
  ungroup()
```

```
# Compute the overall average sentiment
overall_avg_sentiment <- sentiment_scores |>
  summarize(
    avg_sentiment = mean(afinn_normalized, na.rm = TRUE),
    .groups = "drop"
  ) |>
  round(4) |>
  pull(avg_sentiment)

# Get sentiment at Album level
musical_sentiment <- sentiment_scores |>
  group_by(musical) |>
  summarize(
    avg_sentiment = mean(afinn_normalized, na.rm = TRUE),
    .groups = "drop"
  )

musical_sentiment |>
  arrange(desc(avg_sentiment))
```

Table 4: Average sentiment of Sondheim musicals

musical	avg_sentiment
Passion	0.1337627

musical	avg_sentiment
Follies	0.1221282
Sunday in the Park with George	0.0728613
Company	0.0566370
Merrily We Roll Along	0.0564251
A Little Night Music	0.0471659
Sweeney Todd: The Demon Barber of Fleet Street	0.0099184
Into the Woods	0.0058883

```
sweeney_sentiment <- sentiment_scores |>
  filter(musical == "Sweeney Todd: The Demon Barber of Fleet Street") |>
  summarize(
    avg_sentiment = mean(afinn_normalized, na.rm = TRUE),
    .groups = "drop"
  ) |>
  round(4) |>
  pull(avg_sentiment)

passion_sentiment <- sentiment_scores |>
  filter(musical == "Passion") |>
  summarize(
    avg_sentiment = mean(afinn_normalized, na.rm = TRUE),
    .groups = "drop"
  ) |>
  round(4) |>
  pull(avg_sentiment)


musical_avg_sentiment <- musical_sentiment |>
  pull(avg_sentiment)
```

Sondheim's shows portray a wide variety of sentiment, with his average sentiment landing at 0.0596. This is somewhat surprising as shows like *Into the Woods* and *Sweeney Todd* are known for their violence and darker themes, yet neither of these shows average in the negatives. Below, in Figure 2, the average scores of each Sondheim musical are visualized against the average sentiment of the entire corpus.

```
# Visualize average sentiment of Sondheim musicals
musical_sentiment |>
  mutate(
    musical = factor(
```

```
    musical,
    levels = c(
      "Follies", "Passion", "Company",
      "Into the Woods", "A Little Night Music",
      "Sunday in the Park with George",
      "Sweeney Todd: The Demon Barber of Fleet Street"
    )
  )
) |>
ggplot(aes(x = musical, y = avg_sentiment, fill = musical)) +
geom_col() +
geom_hline(
  yintercept = overall_avg_sentiment,
  color = "red", linetype = 2,
  linewidth = 0.8
) +
theme(
  legend.position = "none",
  axis.text.x = element_text(angle = 45, hjust = 1)
) +
scale_fill_brewer(palette = "Set2") +
labs(
  title = "Average sentiment of Sondheim musicals",
  x = "Musical",
  y = "Average Sentiment (normalized)"
)
```

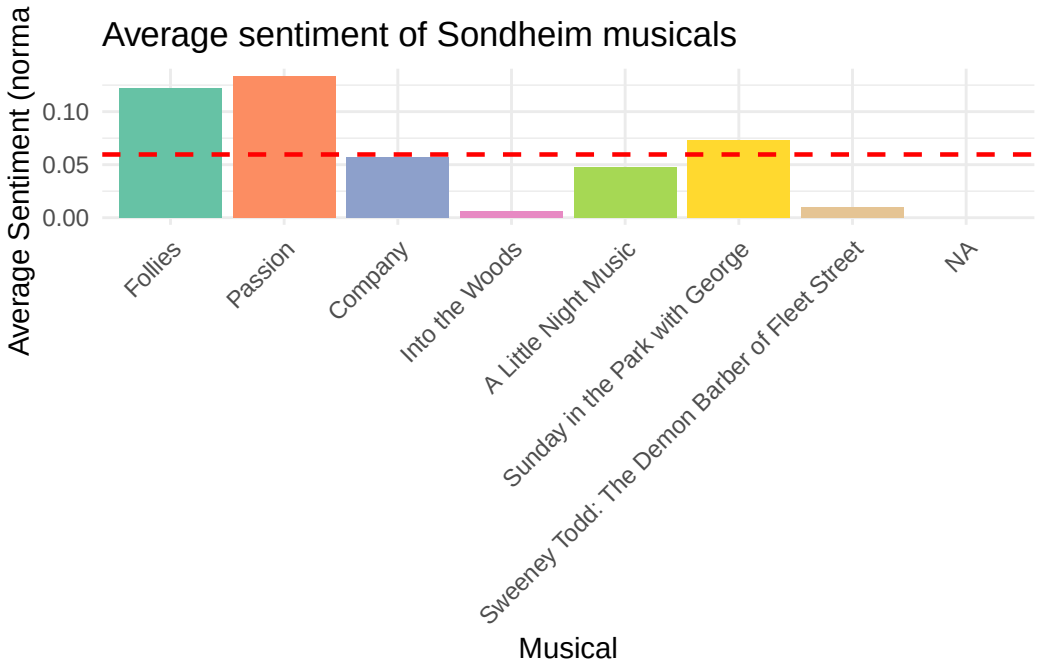

Figure 2: Average sentiment for each Sondheim musical

Next, let's run a quick ANOVA test to see if these differences in sentiment are statistically significant:

```
# Run ANOVA on sentiment scores by album
aov_sentiment <- sentiment_scores |>
  aov(afinn_normalized ~ musical, data = _)

summary(aov_sentiment)
```

```
             Df Sum Sq Mean Sq F value Pr(>F)
musical       7  0.354 0.05050   0.987  0.443
Residuals   163  8.342 0.05118
```

As shown in the ANOVA output, the differences between average mean sentiment for each Sondheim musical are not statistically significant ($p = 0.443$). One thing to consider is that how we perceive the "tone" of a show is heavily influenced by the music and orchestration. *Sweeney Todd* has many songs written in minor keys, whereas shows like *Company* use more major keys. Therefore, even if the lyrics themselves don't differ as much in sentiment, this analysis is unable to account for the influence of the music itself.

Next, I'll use topic modeling to explore the topic of 'love' in Sondheim's corpus – first in a global model and then in two individual show models.

Topic Modeling

```
# Set up Sondheim dtm
sondheim_untidy <- bind_rows(
  merrily, woods, sweeney, company,
  sunday, night, passion, follies
)

sondheim_tracks <- sondheim_corpus |>
  select(musical, song) |>
  group_by(musical) |>
  distinct() |>
  mutate(track = row_number())

sondheim_songs <- sondheim_untidy |>
  left_join(sondheim_tracks, join_by(musical, song)) |>
  group_by(musical, song, track) |>
  summarize(full_text = glue_collapse(text, sep = " ")) |>
  ungroup() |>
  mutate(doc_id = row_number()) |>
  arrange(doc_id)

# Tidy new Sondheim corpus
sondheim_words <- sondheim_songs |>
  unnest_tokens(word, full_text) |>
  mutate(word = str_replace_all(word, "'s", "")) |>
  anti_join(stop_words, join_by(word)) |>
  filter(!(word %in% names), !(word %in% sondheim_stop)) |>
  mutate(word = str_replace_all(word, "pies", "pie")) |>
  filter(str_length(word) > 3)

sondheim_dtm <- sondheim_words |>
  count(doc_id, word) |>
  cast_dtm(document = doc_id, term = word, value = n)
```

```
# Create lda model
k_best <- 10
```

```
lda_best <- LDA(
  sondheim_dtm,
  method = "Gibbs",
  k = k_best,
  control = list(seed = 123)
)
```

```
lda_best |>
  tidy(matrix = "beta") |>
  group_by(topic) |>
  slice_max(beta, n = 10) |>
  ungroup() |>
  mutate(term = reorder_within(term, beta, topic)) |>
  ggplot(aes(beta, term, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  scale_y_reordered() +
  labs(title = "Top Terms per Topic (k = 10)")
```

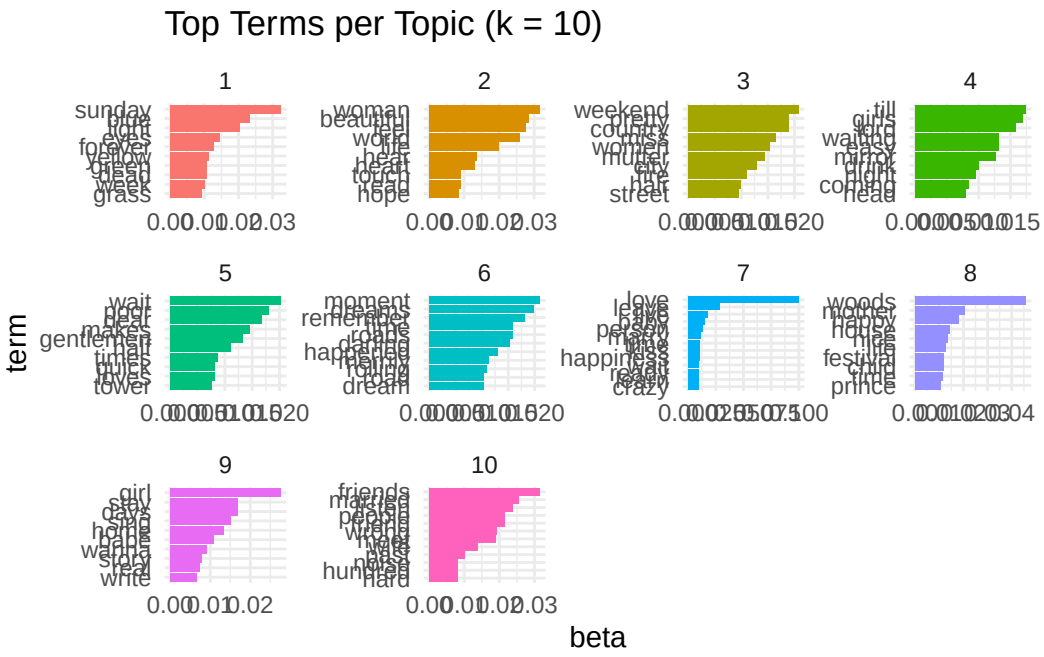

Figure 3: Top topics and terms in entire Sondheim corpus

```r
# Create visualization of top terms in topic 'love'
tidy_lda_best <- tidy(lda_best, matrix = "beta")

love_topics <- tidy_lda_best |>
  group_by(topic) |>
  arrange(desc(beta)) |>
  mutate(topic_rank = row_number()) |>
  filter(topic_rank <= 3, term == "love") |>
  pull(topic)

tidy_lda_best |>
  filter(topic %in% love_topics) |>
  group_by(topic) |>
  slice_max(beta, n = 10) |>
  ungroup() |>
  mutate(term = reorder_within(term, beta, topic)) |>
  ggplot(aes(beta, term, fill = as_factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  scale_y_reordered()
```

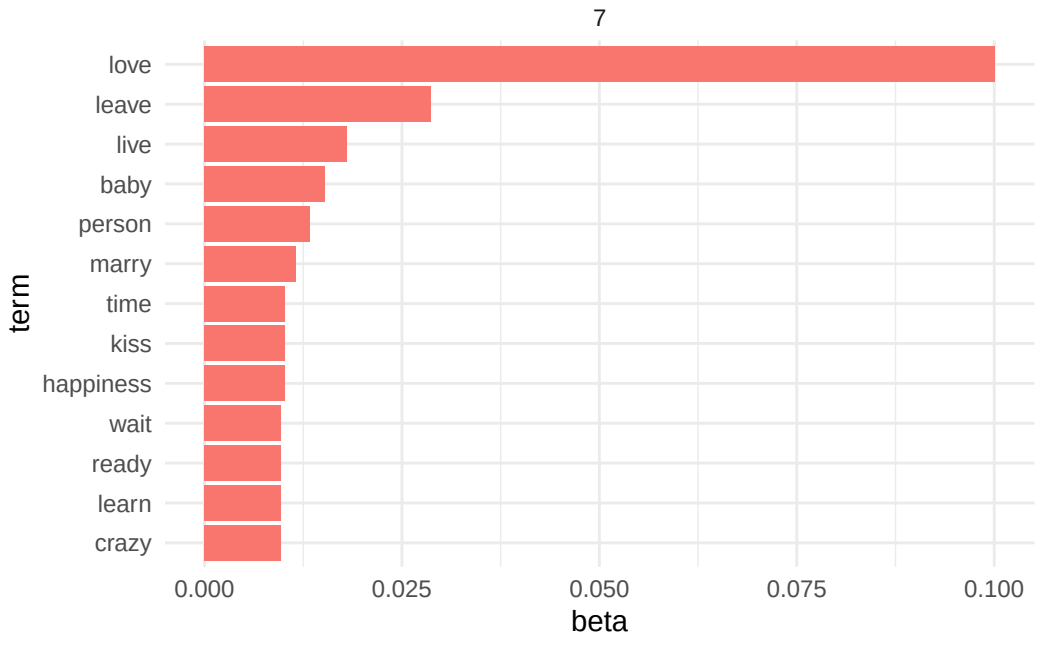

Figure 4: Top terms for topic ‘love’ in entire Sondheim corpus

After exploring between 5 and 15 topics, I decided to choose 10 because it provided the most cohesive overview of 'love' in Sondheim's corpus (in my somewhat qualified opinion). A couple of shows definitely skew the top terms, but this is to be expected due to inequalities in tf of 'love' between shows. As shown in Figure 5, *Passion*, *Follies*, and *Company* all contributed the most instances of 'love' to this model. For example, "marry" and "ready" can be attributed to *Company* because grappling with the implications of marriage, loneliness, and love is a central theme of the show.

```
sondheim_corpus |>
  group_by(musical) |>
  filter(word == "love") |>
  count(word) |>
  ggplot(aes(x = musical, y = n, fill = musical)) +
  geom_col() +
  theme(
    legend.position = "none",
    axis.text.x = element_text(angle = 45, hjust = 1)
  ) +
  scale_fill_brewer(palette = "Set2") +
  labs(
    title = "Instances of the term 'love' in Sondheim musicals",
    x = "Musical"
  )
```

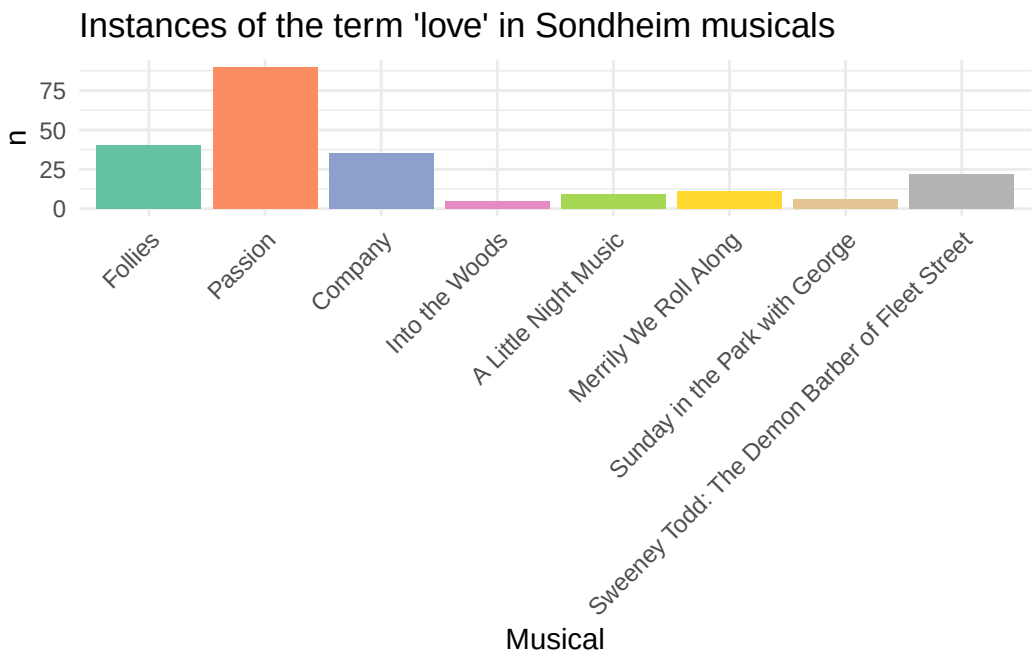

Figure 5: Instances of the term 'love' in Sondheim musicals

Now, I'll create topic models for two shows that have higher frequencies of 'love' but differ highly in sentiment. *Passion* has the highest frequency of 'love' (n = 90) and the highest average sentiment (0.1338). For the best contrast, I'll compare *Passion* to *Sweeney Todd*, which has (n = 22) and an average sentiment of 0.0099.

> **Note**
>
> The k-values for both musicals were chosen through trial-and-error in order to find the most representative list of terms (again, through my somewhat qualified opinion).

Topic Modeling by Musical: *Passion*

```
# Create visualization of top terms in topic 'love'
tidy_lda_passion <- tidy(lda_passion, matrix = "beta")

love_passion <- tidy_lda_passion |>
  group_by(topic) |>
  arrange(desc(beta)) |>
  mutate(topic_rank = row_number()) |>
```

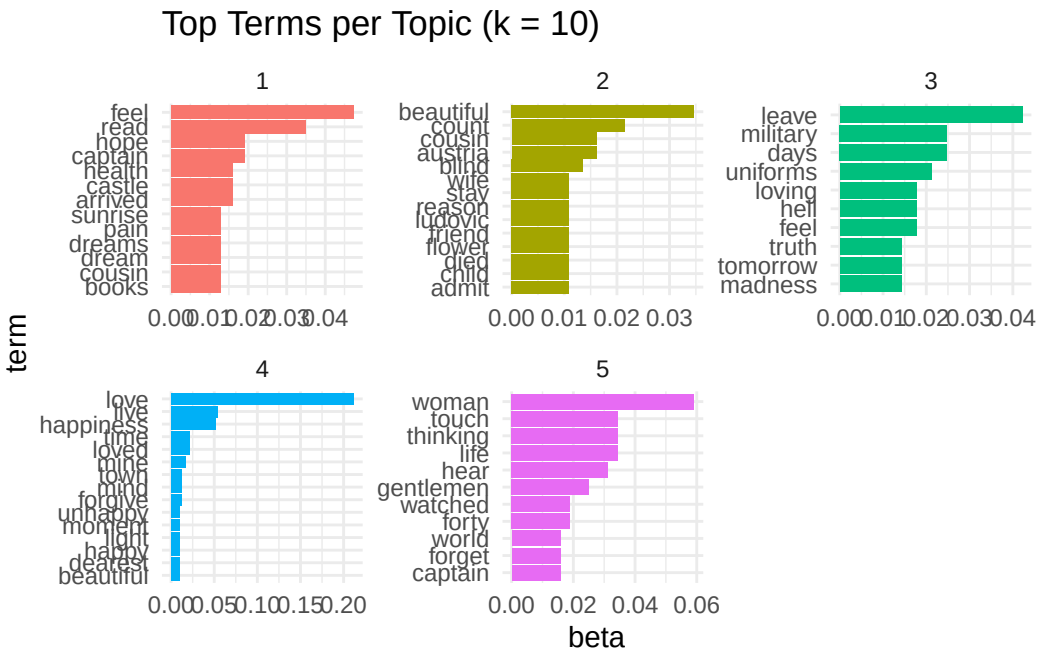

Figure 6: Top topics and terms in *Passion*

```
  filter(topic_rank <= 3, term == "love") |>
  pull(topic)

tidy_lda_passion |>
  filter(topic %in% love_passion) |>
  group_by(topic) |>
  slice_max(beta, n = 10) |>
  ungroup() |>
  mutate(term = reorder_within(term, beta, topic)) |>
  ggplot(aes(beta, term, fill = as_factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  scale_y_reordered()
```

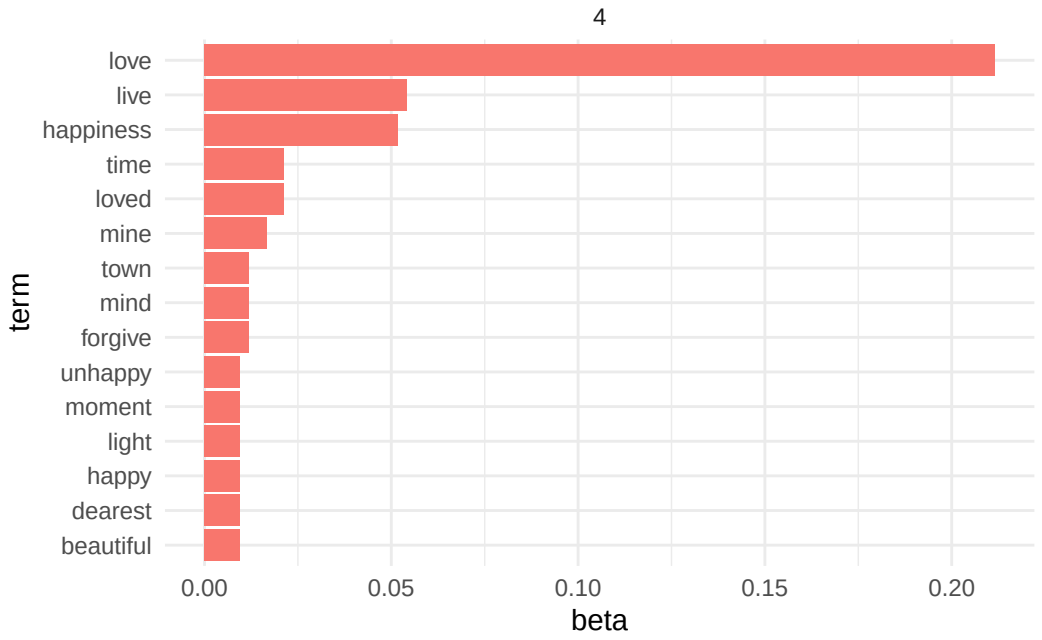

Figure 7: Top terms for topic 'love' in *Passion*

As shown in Figure 7, *Passion*'s exploration of love is broader, more 'cliche', and contains terms that one would expect (e.g., "happiness," "mine," "beautiful", etc.). However, there are hints of darker themes through words like "unhappy" and "forgive," suggesting a more complex attitude toward love than apparent at first glance.

Topic Modeling by Musical: *Sweeney Todd*

```
# Create visualization of top terms in topic 'love'
tidy_lda_sweeney <- tidy(lda_sweeney, matrix = "beta")

love_sweeney <- tidy_lda_sweeney |>
  group_by(topic) |>
  arrange(desc(beta)) |>
  mutate(topic_rank = row_number()) |>
  filter(topic_rank <= 3, term == "love") |>
  pull(topic)

tidy_lda_sweeney |>
  filter(topic %in% love_sweeney) |>
```

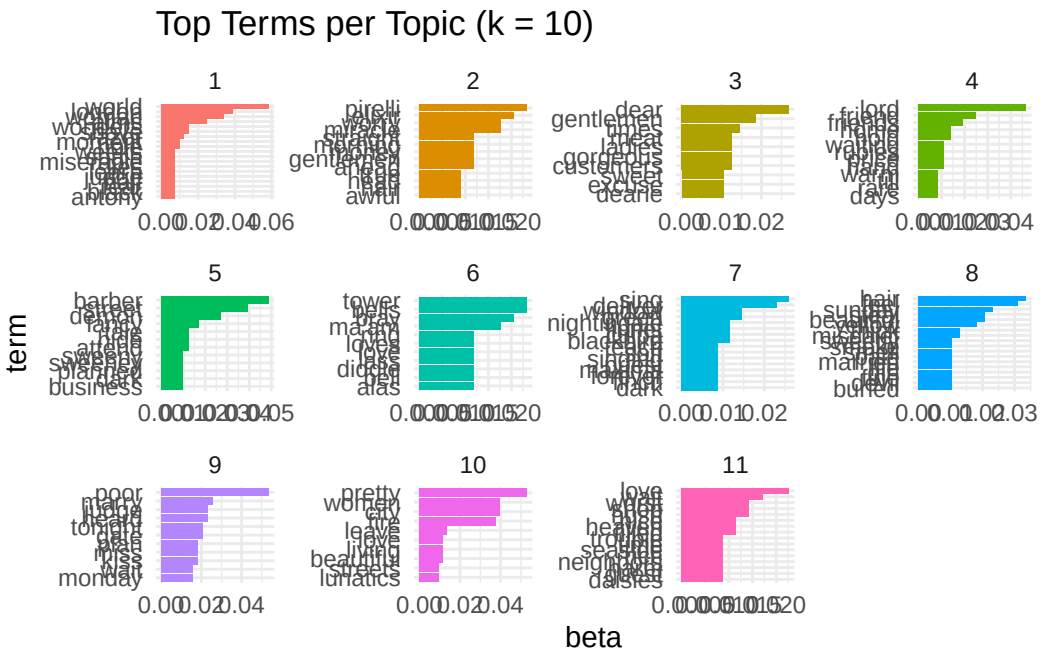

Figure 8: Top topics and terms in *Sweeney Todd*

```
  group_by(topic) |>
  slice_max(beta, n = 10) |>
  ungroup() |>
  mutate(term = reorder_within(term, beta, topic)) |>
  ggplot(aes(beta, term, fill = as_factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  scale_y_reordered()
```

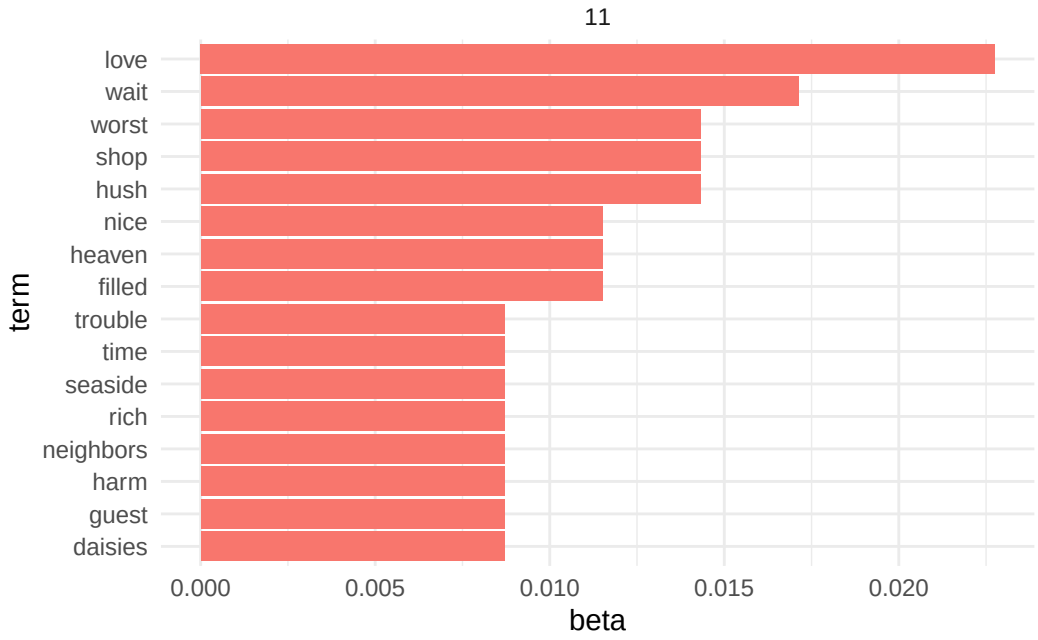

Figure 9: Top terms for topic 'love' in *Sweeney Todd*

Contrastly, as shown in Figure 9, in *Sweeney Todd*, 'love' is associated with a plethora of terms, obvious and not. For example, "heaven," "wait," and "nice" are fairly innocent, but the topic of love in *Sweeney Todd* becomes more complex when we consider words like "worst," "trouble," and "harm." The presence of these negative words speaks to the pain that Sweeney (and other characters) feel when grappling with love, and Sondheim uses *Sweeney Todd* to explore how love and morality intertwine – for good and ill.

Conclusion

As explored above, Sondheim's lyrical corpus is diverse and his characters/narration are often highly specific. Despite this (or perhaps because of this), Sondheim's work brilliantly navigates broad themes of love and life in accessible, meaningful, and heart-wrenching ways. 'Love' emerged as the most common word in Sondheim's corpus, and through tf-idf, sentiment analysis, and topic modeling, I highlighted the varied ways in which Sondheim explores love through unique characters and narration. Shows with a more positive average sentiment (e.g., *Passion*) still approached love with complexity, balancing 'cliche' terms (e.g., "happiness" and "mine") with more brooding ones (e.g., "unhappy" and "forgive"). Similarly, shows with more negative average sentiment (e.g., *Sweeney Todd*) included both hopeful and melancholy terms associated with love (e.g., "heaven" vs. "harm" and "nice" vs. "trouble"), illustrating the intricate nature of Sondheim's lyric writing and storytelling. Importantly, as previously

mentioned, lyrics only tell part of the story when examining love – without the context of orchestration, staging, and acting, we cannot explicate the full story of Sondheim's narrative intent. Nonetheless, analyzing his lyrics separately provides important insight into his lyrical genius.