Topic 5: Word Relationships

Mia Forsline

2022-05-03

Import EPA environmental justice data

```
#character of full file paths for each PDF in the data folder
files <- list.files(path = here::here("data"),
                    pattern = "pdf$", full.names = TRUE)

#subset to include only EPA report PDFs
files <- str_subset(files, pattern="EPA")

#list
ej_reports <- lapply(files, pdf_text)

#readtext/df of all 6 PDf reports
ej_pdf <- readtext(file = files,
                   docvarsfrom = "filenames",
                   docvarnames = c("type","year"),
                   sep = "_")

#creating an initial corpus containing the data
epa_corp <- corpus(x = ej_pdf, text_field = "text" )

#check that the corpus contains all 6 reports
summary(epa_corp)
```

```
## Corpus consisting of 6 documents, showing 6 documents:
##
##            Text Types Tokens Sentences  type year
##  EPAEJ_2015.pdf  2136   8944       263 EPAEJ 2015
##  EPAEJ_2016.pdf  1599   7965       176 EPAEJ 2016
##  EPAEJ_2017.pdf  3973  30564       653 EPAEJ 2017
##  EPAEJ_2018.pdf  2774  16658       447 EPAEJ 2018
##  EPAEJ_2019.pdf  3773  22648       672 EPAEJ 2019
##  EPAEJ_2020.pdf  4493  30523       987 EPAEJ 2020
```

Add additional, context-specific stop words to stop word lexicon

```
more_stops <-c("2015",
               "2016",
               "2017",
               "2018",
               "2019",
```

```
                "2020",
                "www.epa.gov",
                "https")
add_stops <- tibble(word = c(stop_words$word, more_stops))
stop_vec <- as_vector(add_stops)
```

Tokenize the data into single words

```
tokens <- tokens(epa_corp, remove_punct = TRUE)

toks1<- tokens_select(tokens, min_nchar = 3)
toks1 <- tokens_tolower(toks1) #lowercase
toks1 <- tokens_remove(toks1, pattern = (stop_vec)) #remove stop words
dfm <- dfm(toks1) #create a data frequency matrix
```

1. What are the most frequent trigrams in the dataset? How does this compare to the most frequent bigrams? Which n-gram seems more informative here, and why?

Calculate the 10 most frequent bigrams

```
toks2 <- tokens_ngrams(toks1, n=2)
dfm2 <- dfm(toks2)
dfm2 <- dfm_remove(dfm2, pattern = c(stop_vec))
freq_words2 <- textstat_frequency(dfm2, n=20)
freq_words2$token <- rep("bigram", 20)

tstat_freq2 <- textstat_frequency(dfm2, n = 5, groups = year)

#display the top 10 rows in a table
tstat_freq2[1:10] %>%
  rename(Bigram = feature,
         "Report Year" = group,
         Frequency = frequency,
         Rank = rank,
         "Document Frequency" = docfreq) %>%
  kbl() %>%
  kable_styling()
```

Bigram	Frequency	Rank	Document Frequency	Report Year
environmental_justice	82	1	1	2015
progress_report	25	2	1	2015
fiscal_annual	23	3	1	2015
annual_environmental	23	3	1	2015
justice_progress	23	3	1	2015
environmental_justice	63	1	1	2016
progress_report	21	2	1	2016
report_2015-2016	18	3	1	2016
urban_waters	16	4	1	2016
equitable_development	8	5	1	2016

Trigram	Frequency	Rank	Document Frequency	Report Year
fiscal_annual_environmental	23	1	1	2015
annual_environmental_justice	23	1	1	2015
environmental_justice_progress	23	1	1	2015
justice_progress_report	23	1	1	2015
page_fiscal_annual	13	5	1	2015
progress_report_2015-2016	18	1	1	2016
environmental_justice_concerns	6	2	1	2016
epa's_environmental_justice	5	3	1	2016
urban_waters_program	5	3	1	2016
national_environmental_justice	4	5	1	2016

Calculate the most frequent trigrams

```
toks3 <- tokens_ngrams(toks1, n=3)
dfm3 <- dfm(toks3)
dfm3 <- dfm_remove(dfm3, pattern = c(stop_vec))
freq_words3 <- textstat_frequency(dfm3, n=20)
freq_words3$token <- rep("trigram", 20)

tstat_freq3 <- textstat_frequency(dfm3, n = 5, groups = year)

#display the top 10 rows in a table
tstat_freq3[1:10] %>%
  rename(Trigram = feature,
         "Report Year" = group,
         Frequency = frequency,
         Rank = rank,
         "Document Frequency" = docfreq) %>%
  kbl() %>%
  kable_styling()
```

The third word in the trigrams does not add a lot of value to the n-gram.

2. Choose a new focal term to replace "justice" and recreate the correlation table and network (see corr_paragraphs and corr_network chunks). Explore some of the plotting parameters in the cor_network chunk to see if you can improve the clarity or amount of information your plot conveys. Make sure to use a different color for the ties!

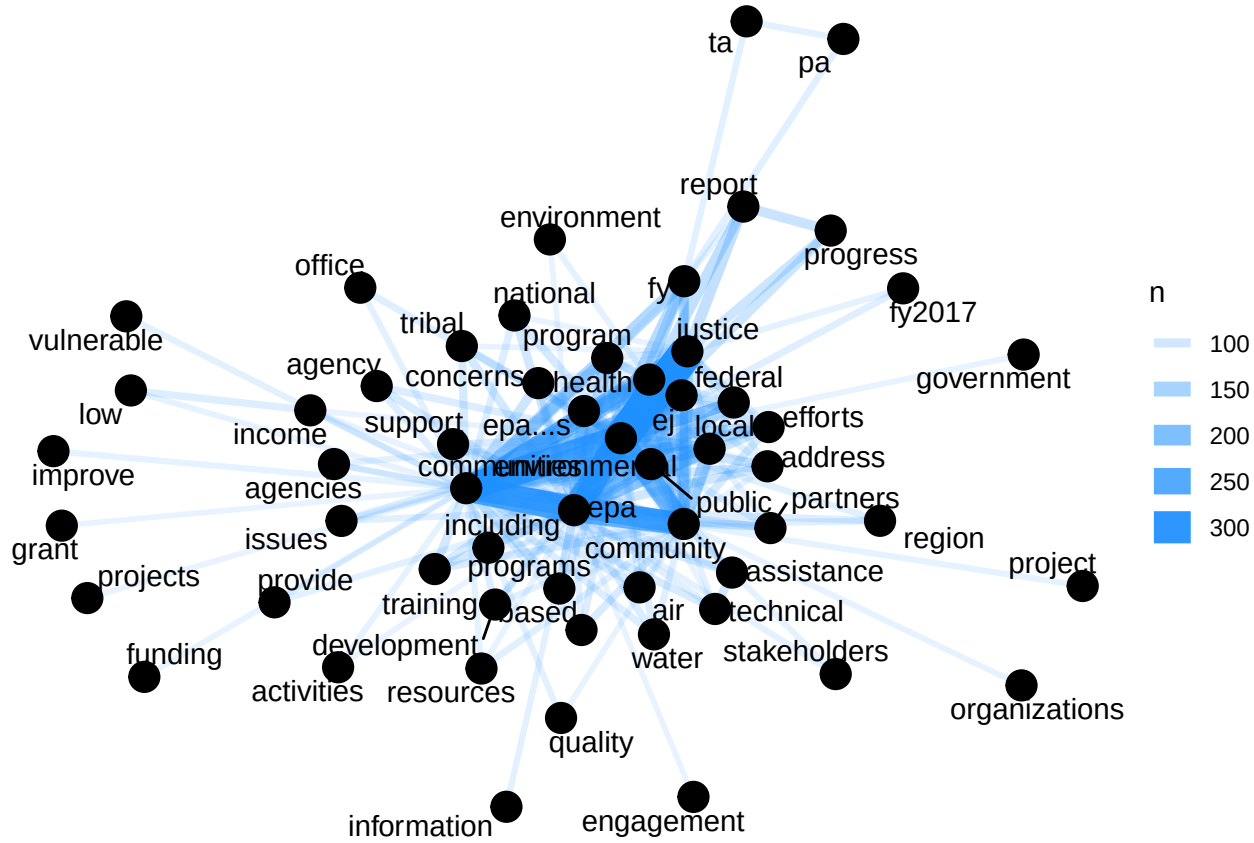
Now we'll create some different data objects that will set us up for the subsequent analyses

```
#convert to tidy format and apply my stop words
raw_text <- tidy(epa_corp)

#Distribution of most frequent words across documents
raw_words <- raw_text %>%
  mutate(year = as.factor(year)) %>%
  unnest_tokens(word, text) %>%
  anti_join(add_stops, by = 'word') %>%
  count(year, word, sort = TRUE)

#number of total words by document
total_words <- raw_words %>%
  group_by(year) %>%
  summarize(total = sum(n))

report_words <- left_join(raw_words, total_words)

#paragraph tokens
par_tokens <- unnest_tokens(raw_text, output = paragraphs, input = text, token = "paragraphs")

#give each paragraph an id number
par_tokens <- par_tokens %>%
 mutate(par_id = 1:n())

#individual words
par_words <- unnest_tokens(par_tokens, output = word, input = paragraphs, token = "words")
```

Let's see which words tend to occur close together in the text. This is a way to leverage word relationships (in this case, co-occurence in a single paragraph) to give us some understanding of the things discussed in the documents.

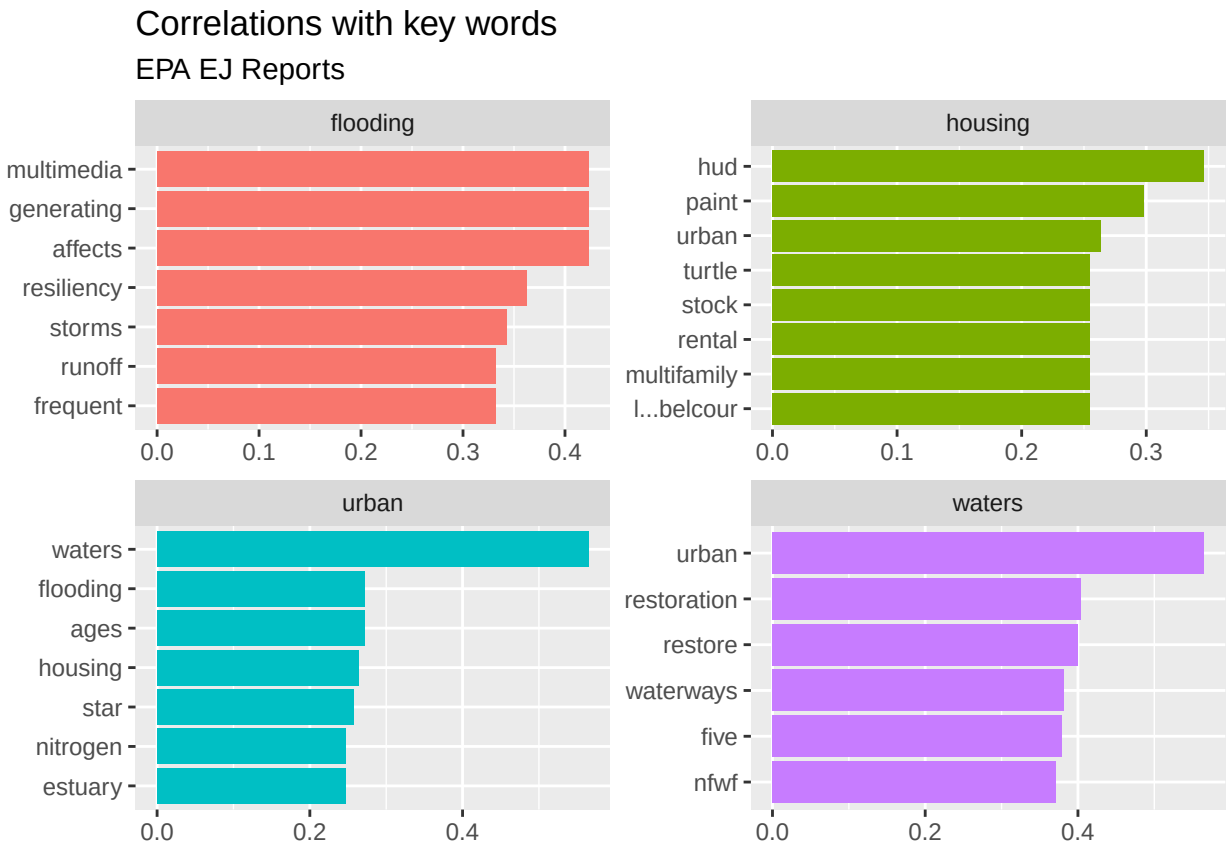
```
word_pairs <- par_words %>%
  pairwise_count(word, par_id, sort = TRUE, upper = FALSE) %>%
  anti_join(add_stops, by = c("item1" = "word")) %>%
  anti_join(add_stops, by = c("item2" = "word"))

word_pairs %>%
  filter(n >= 70) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), edge_colour = "dodgerblue") +
  geom_node_point(size = 5) +
```

```
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding = unit(0.2, "lines")) +
  theme_void()
```



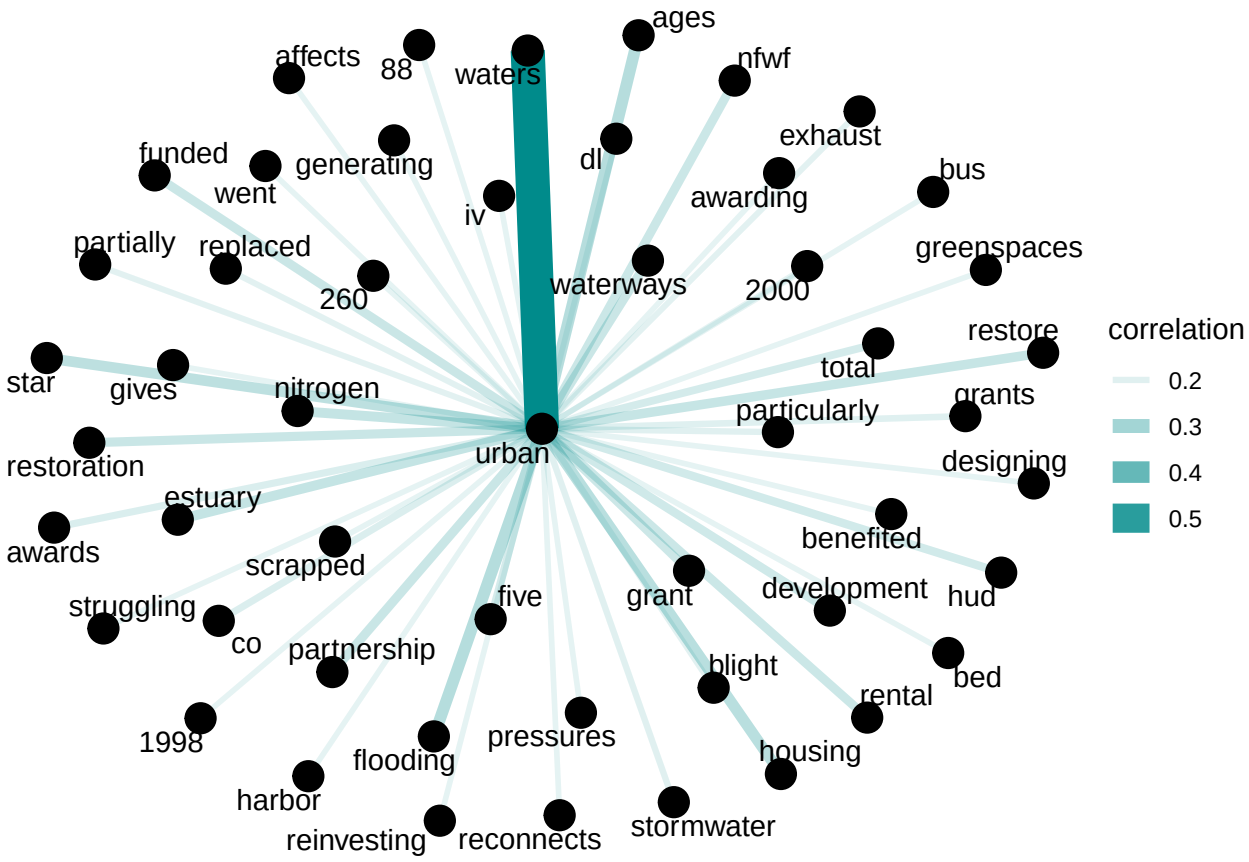

```
#calculate correlation coefficients between words
word_cors <- par_words %>%
  add_count(par_id) %>%
  filter(n >= 50) %>%
  select(-n) %>%
  pairwise_cor(word, par_id, sort = TRUE)

#search for words correlated with "justice"
urban_cors <- word_cors %>%
  filter(item1 == "urban")

#graph correlation coefficients for words correlated with 4 key terms
  word_cors %>%
  filter(item1 %in% c("waters",
                      "flooding",
                      "housing",
                      "urban"))%>%
  group_by(item1) %>%
  top_n(6) %>%
  ungroup() %>%
  mutate(item1 = as.factor(item1),
```

```
  name = reorder_within(item2, correlation, item1)) %>%
  ggplot(aes(y = name, x = correlation, fill = item1)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~item1, ncol = 2, scales = "free")+
  scale_y_reordered() +
  labs(y = NULL,
         x = NULL,
         title = "Correlations with key words",
         subtitle = "EPA EJ Reports")
```



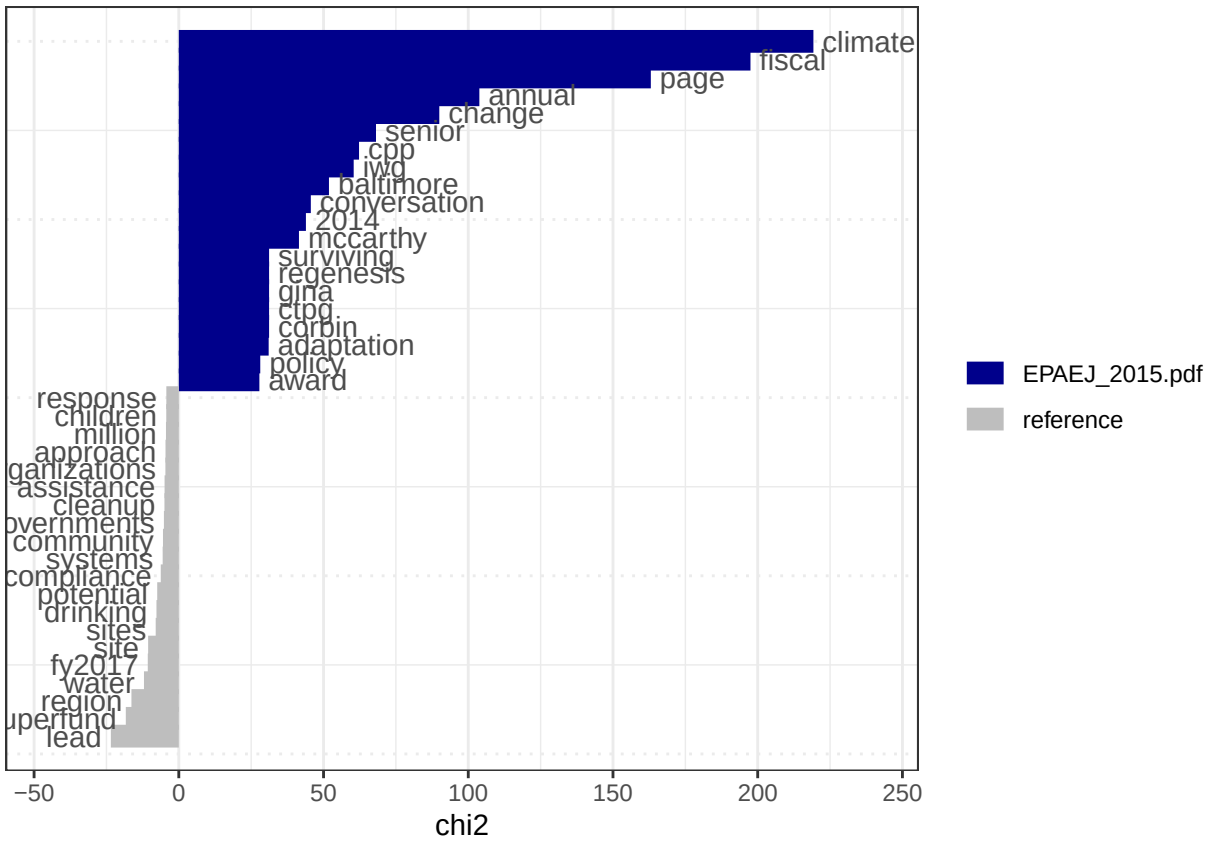

```
#let's zoom in on just one of our key terms
urban_cors <- word_cors %>%
  filter(item1 == "urban") %>%
  mutate(n = 1:n())
```

```
urban_cors  %>%
  filter(n <= 50) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = correlation, edge_width = correlation), edge_colour = "cyan4") +
  geom_node_point(size = 5) +
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding = unit(0.2, "lines")) +
  theme_void()
```



3. Write a function that allows you to conduct a keyness analysis to compare two individual EPA reports (hint: that means target and reference need to both be individual reports). Run the function on 3 pairs of reports, generating 3 keyness plots.

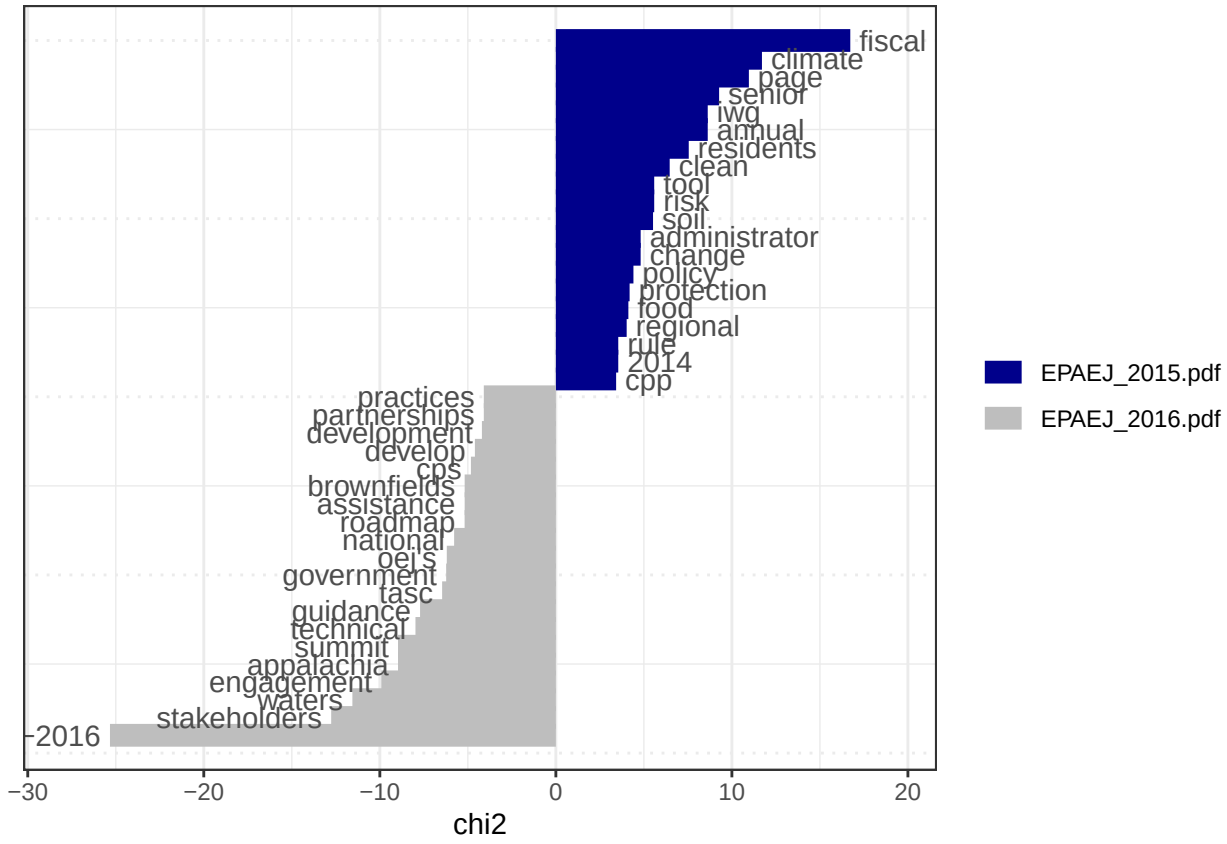
```
keyness <- textstat_keyness(dfm, target = 1) #target = 1 --> first document in the list, so in this cas
#target = 2 would use 2016 as the target document
textplot_keyness(keyness)
```



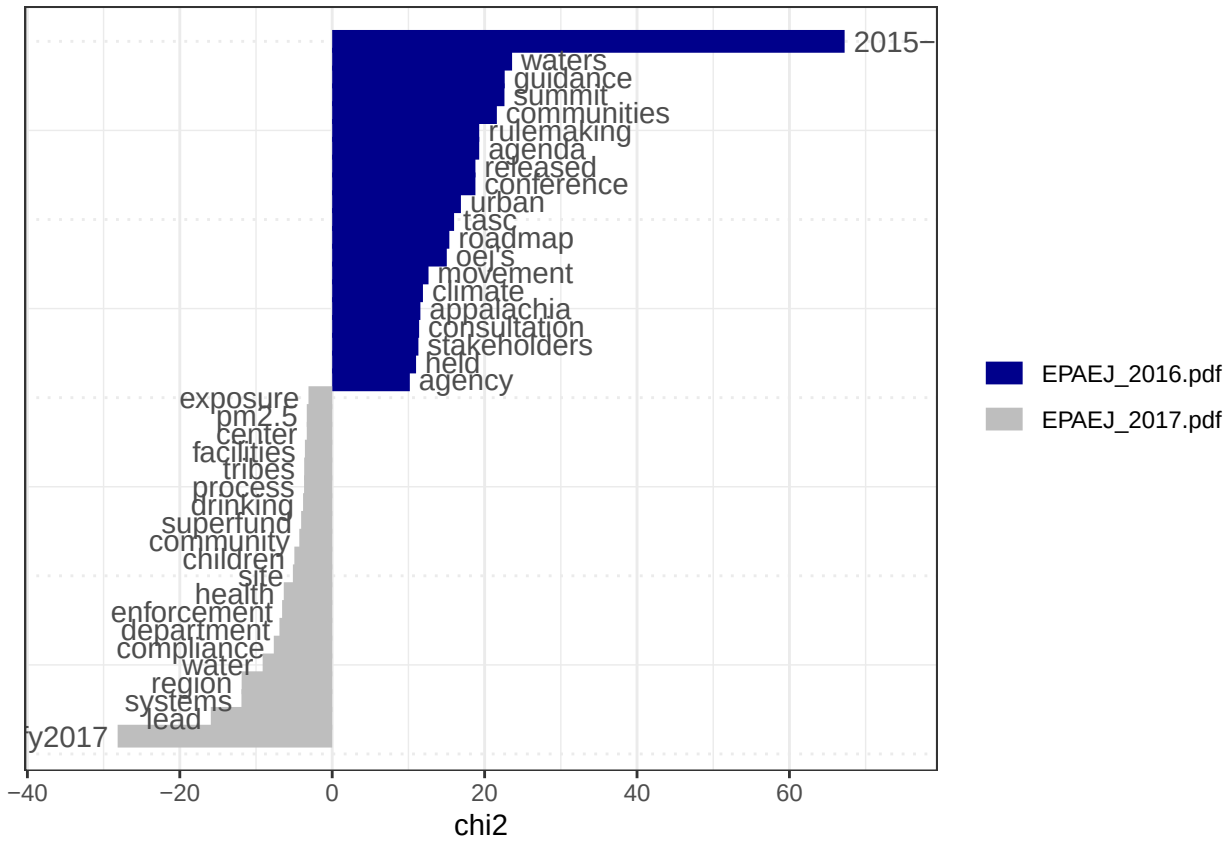

```
#chi-square values for each word
#in 2015, climate has the highest chi-square value, so climate occurred the most "more than expected" g
#words in blue = occured more than expected
#words in grey = occurred less than expected
#reference = includes all reports except 2015
```

```
keyness_function <- function(report_number) {

  for (i in (1:(length(epa_corp) -1))) {
    report_test <- epa_corp[i: (i+1)]
    print(report_test)
    tokens_test <- tokens(report_test, remove_punct = TRUE) #list split word by word
    tokens_test <- tokens_select(tokens_test, min_nchar = 3)
    tokens_test <- tokens_tolower(tokens_test)
    tokens_test <- tokens_remove(tokens_test, pattern = (stop_vec))
    dfm_test <- dfm(tokens_test) #document feature matrix

    keyness <- textstat_keyness(dfm_test, target = 1)
    print(textplot_keyness(keyness))

  }

}

keyness_function(report_number = 1)
```

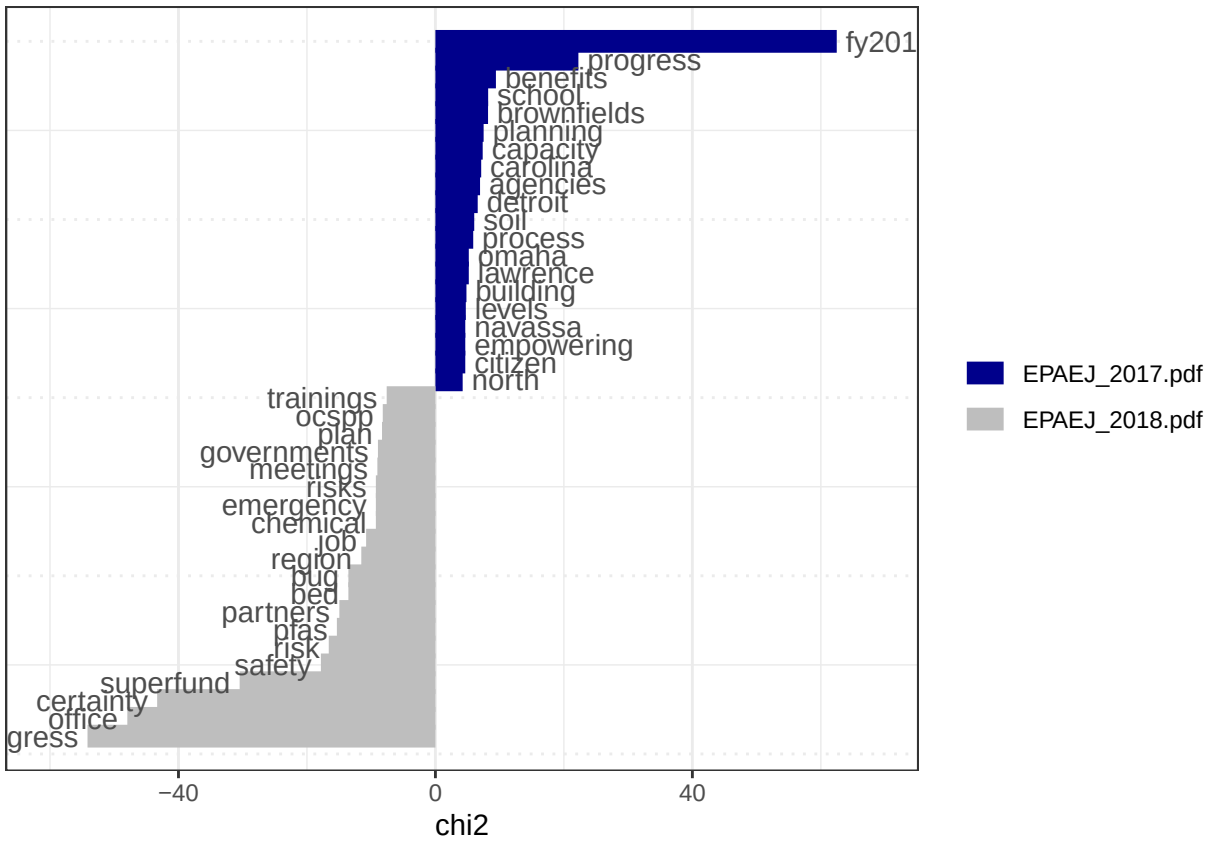
```
## Corpus consisting of 2 documents and 2 docvars.
## EPAEJ_2015.pdf :
## "  Clean water and clean air don't just happen, especially in..."
##
## EPAEJ_2016.pdf :
## " EPA-300-R-17-001 Table of Contents Preface  .  .  .  .  .  ..."
```



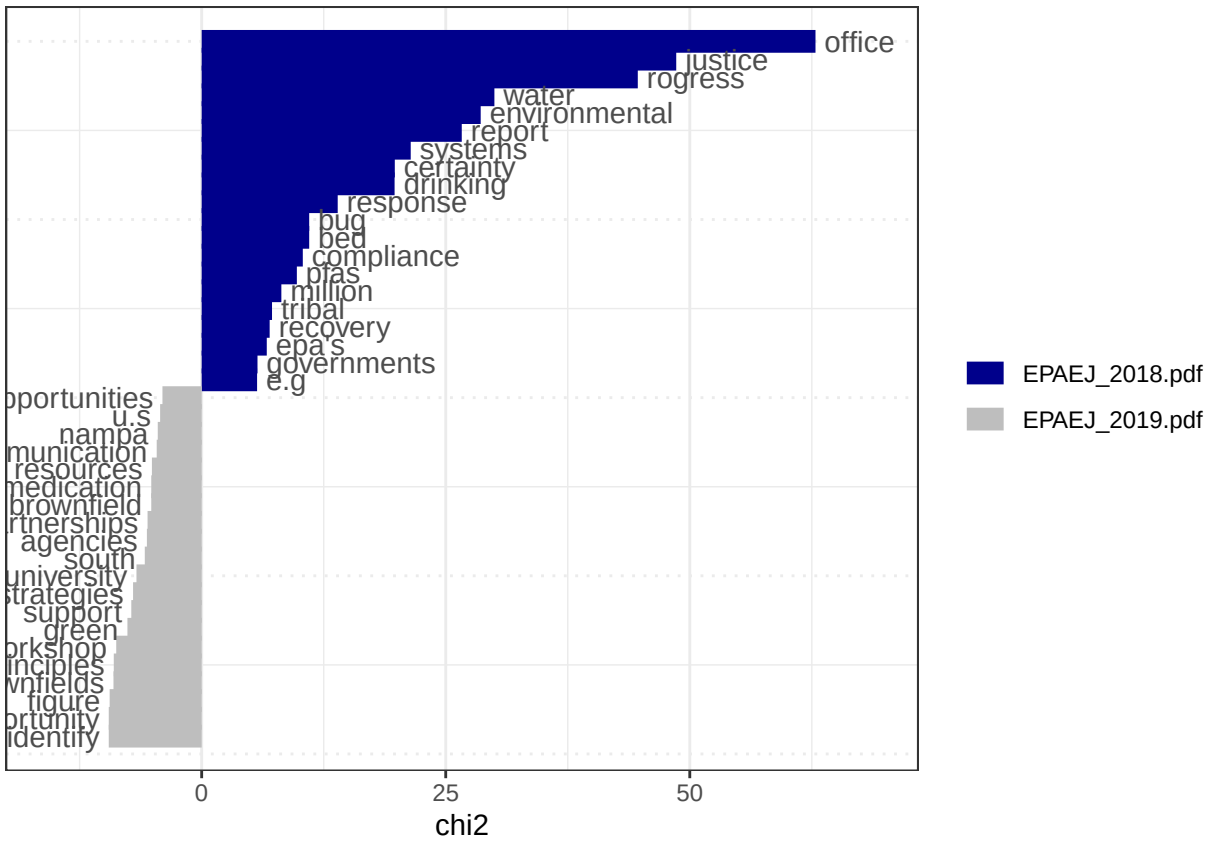

```
## Corpus consisting of 2 documents and 2 docvars.
## EPAEJ_2016.pdf :
## " EPA-300-R-17-001 Table of Contents Preface  .  .  .  .  .  ..."
##
## EPAEJ_2017.pdf :
## " Environmental Justice FY2017 Progress Report Blank Page    ..."
```



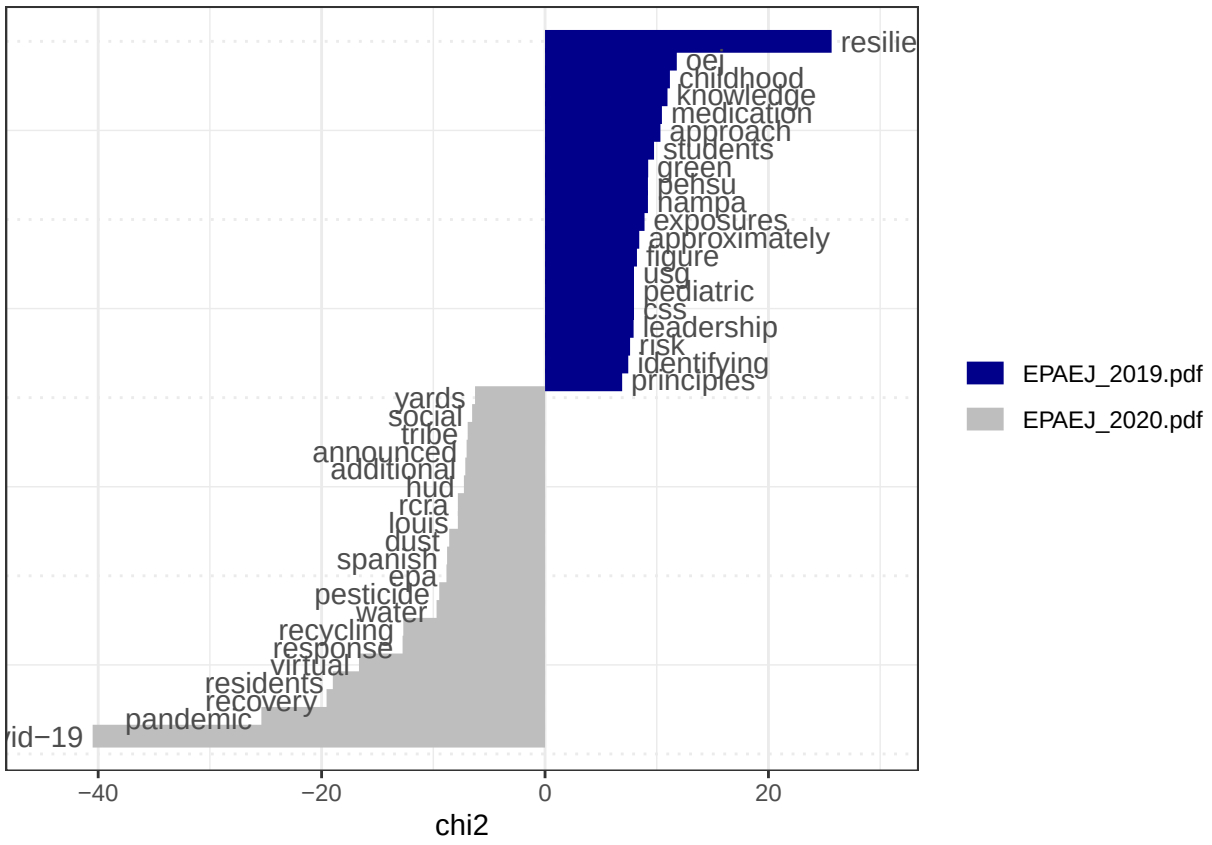

```
## Corpus consisting of 2 documents and 2 docvars.
## EPAEJ_2017.pdf :
## " Environmental Justice FY2017 Progress Report Blank Page    ..."
##
## EPAEJ_2018.pdf :
## " EPA Publication Number: 230R19002 Table of Contents LEADERS..."
```




```
## Corpus consisting of 2 documents and 2 docvars.
## EPAEJ_2018.pdf :
## " EPA Publication Number: 230R19002 Table of Contents LEADERS..."
##
## EPAEJ_2019.pdf :
## " ADMINISTRATOR FOREWORD This year marks the 25th anniversary..."
```




```
## Corpus consisting of 2 documents and 2 docvars.
## EPAEJ_2019.pdf :
## " ADMINISTRATOR FOREWORD This year marks the 25th anniversary..."
##
## EPAEJ_2020.pdf :
## " EPA Publication Number: 230R20002 Table of Contents MESSAGE..."
```



4. Select a word or multi-word term of interest and identify words related to it using windowing and keyness comparison. To do this you will create to objects: one containing all words occurring within a 10-word window of your term of interest, and the second object containing all other words. Then run a keyness comparison on these objects. Which one is the target, and which the reference? Hint