

# 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)
```

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(bootstrap_options = c("striped", "hover"),
               latex_options = "HOLD_position") #hold the table position when knitting
```

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

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(bootstrap_options = c("striped", "hover"),
               latex_options = "HOLD_position")
```

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

After comparing the 10 most common bigrams and trigrams, the additional third word does not seem to add much more meaning to the results. For example, “environmental\_justice” is just as informative as “national\_environmental\_justice.” The third word can even be more confusing. For instance “page\_fiscal\_annual” is an unclear trigram. Thus, the bigrams seem more informative to me.

**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!**

Put the text into tidy format and extract individual words from each paragraph

```

#convert to tidy format
raw_text <- tidy(epa_corp)

#paragraph tokens (tibble)
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())

#extract individual words from each paragraph
par_words <- unnest_tokens(par_tokens, output = word, input = paragraphs, token = "words")

```

Identify which words tend to occur closely together in the text

```

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"))

#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 and graph words correlated with the term “urban”

```

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

#create facet_wrap labels
supp.labs <- c("Waters",
              "Flooding",
              "Housing",
              "Urban")
names(supp.labs) <- c("waters",
                    "flooding",
                    "housing",
                    "urban")

#graph correlation coefficients for words correlated with 4 key terms based on the correlation coefficient
word_cors %>%
  filter(item1 %in% c("waters",
                    "flooding",
                    "housing",
                    "urban")) %>%
  group_by(item1) %>%
  top_n(5) %>% #display top 5 terms
  ungroup() %>%
  mutate(item1 = as.factor(item1),

```

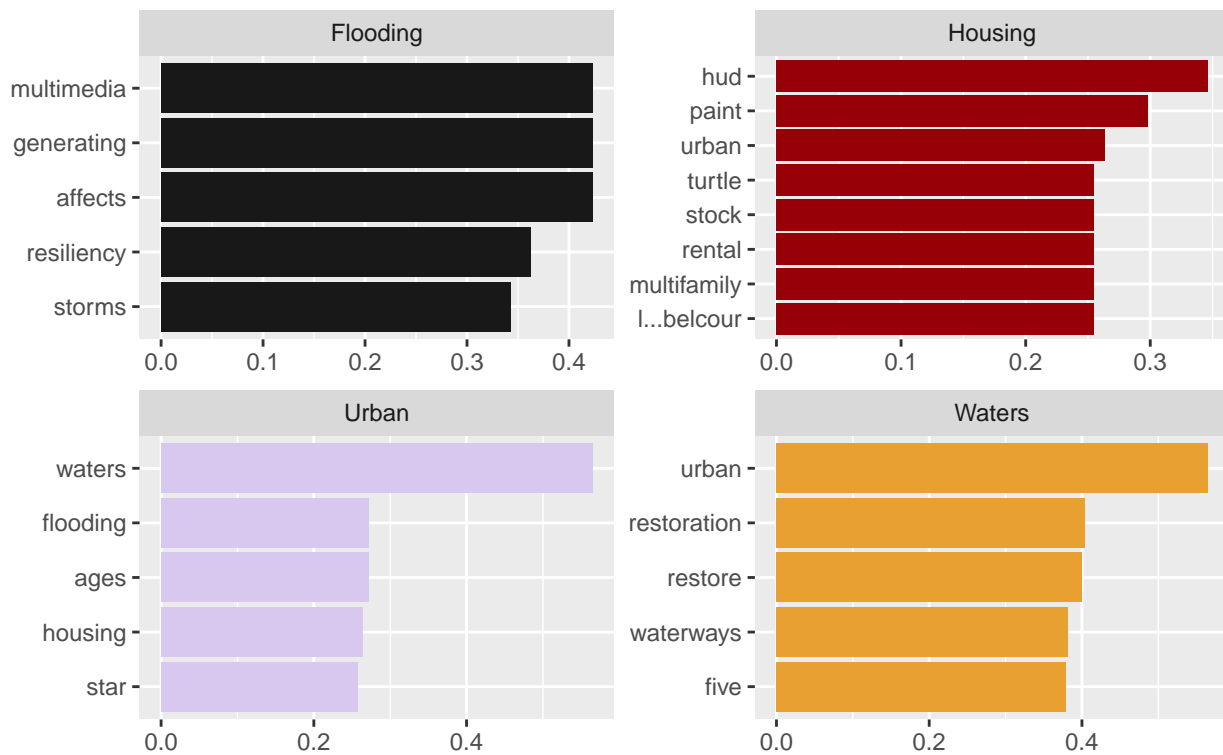
```

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",
           labeller = labeller(item1 = supp.labs))+
scale_y_reordered() +
labs(y = NULL,
     x = NULL,
     title = "Word correlation strength with 4 key words",
     subtitle = "Data taken from 2015 - 2020 EPA EJ Reports") +
palettetown::scale_fill_poke(pokemon = "Beedrill", spread = 4) #select a Pokemon themed color palette

```

## Word correlation strength with 4 key words

Data taken from 2015 – 2020 EPA EJ Reports



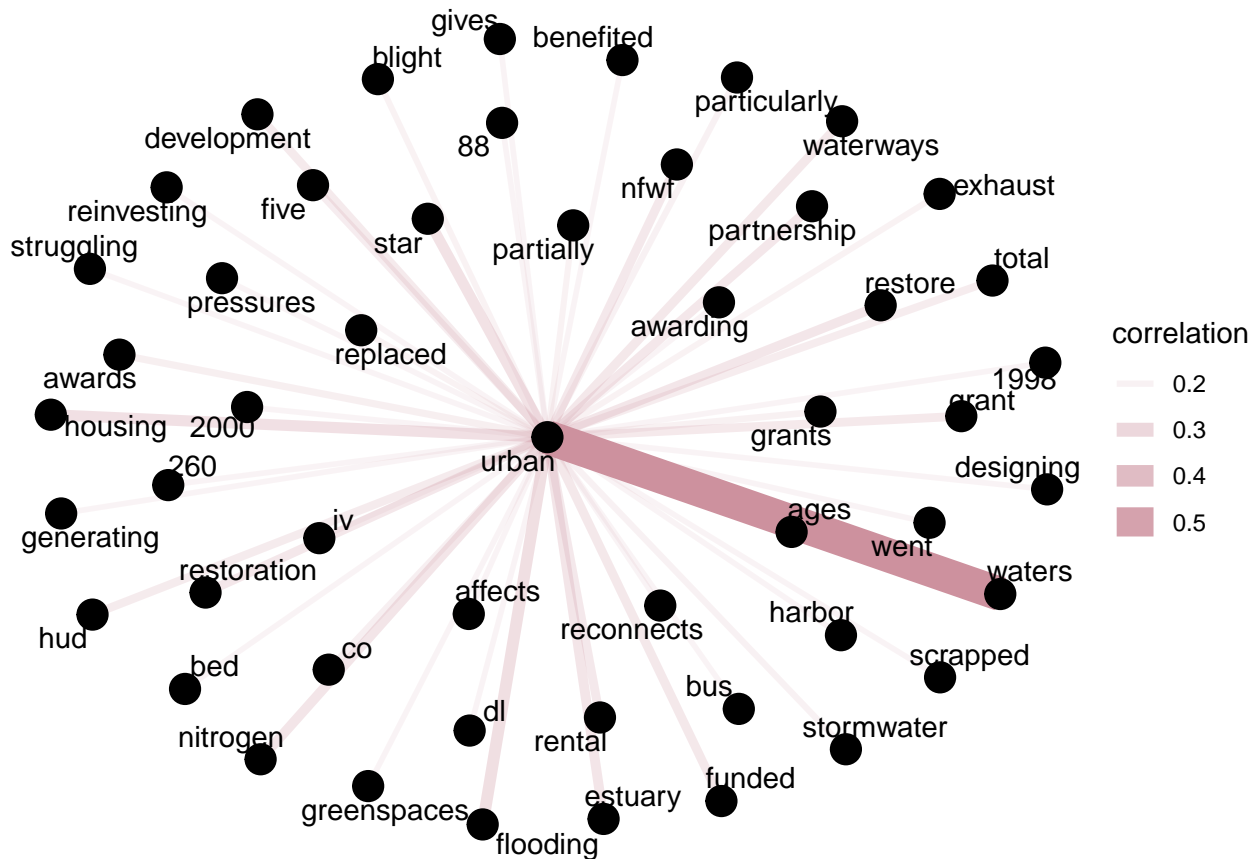
Zoom in on just the key term “urban” and the 50 most related words

```

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 = "pink3") +

```

```
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.

Create the function

```
keyness_function <- function(report_number) {

  for (i in (1:(length(epa_corp) -1))) { #1:5 to make sure the last list doesn't throw an error
    report_test <- epa_corp[i: (i+1)] #create a list of 2 reports
    #print each list to check that all pairs of reports are accounted for
    #print(report_test)

    #tokenize each list
    tokens_test <- tokens(report_test, remove_punct = TRUE)
    tokens_test <- tokens_select(tokens_test, min_nchar = 3)
```

```

tokens_test <- tokens_tolower(tokens_test)
tokens_test <- tokens_remove(tokens_test, pattern = (stop_vec))

#create a document feature matrix
dfm_test <- dfm(tokens_test)

#use the textstat_keyness function where the target is the first report in the list, so the second
keyness <- textstat_keyness(dfm_test, target = 1)

#print the results
print(textplot_keyness(keyness))

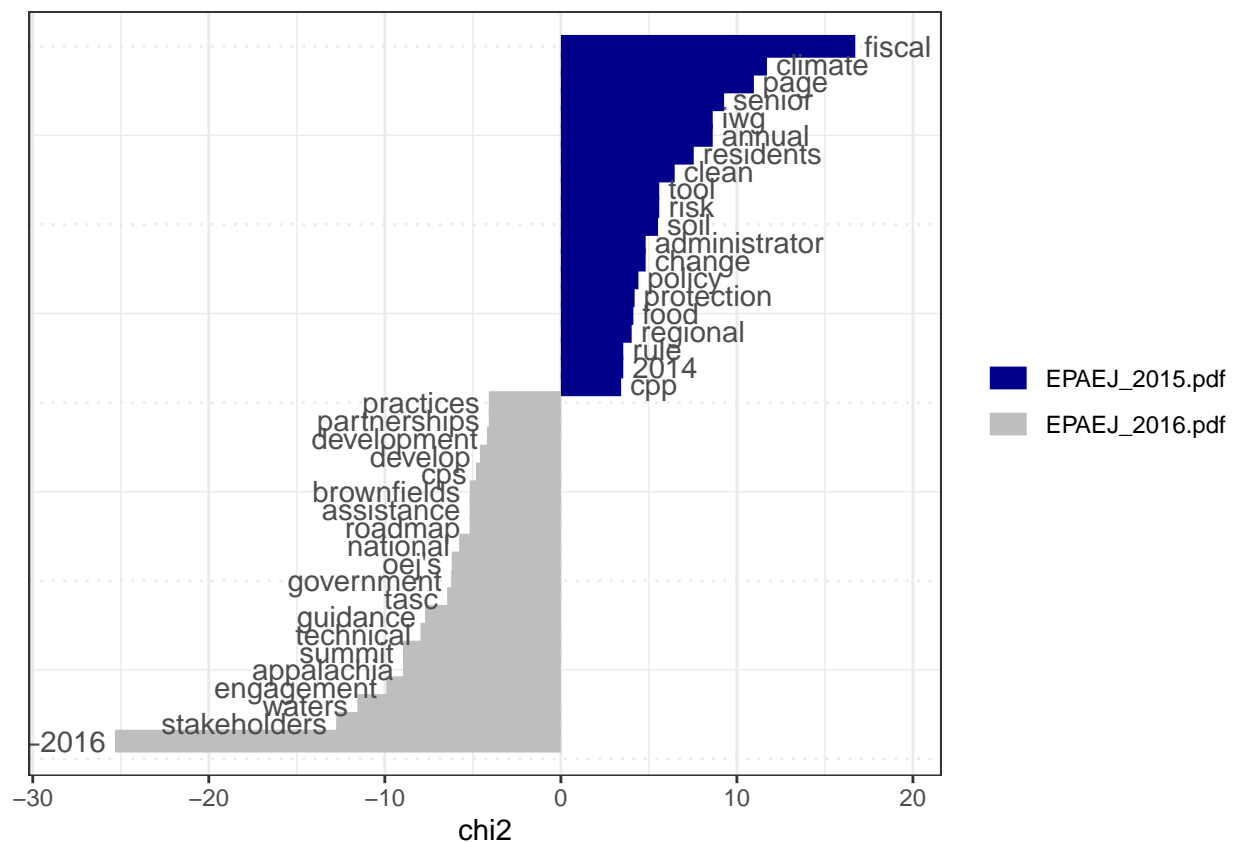
}

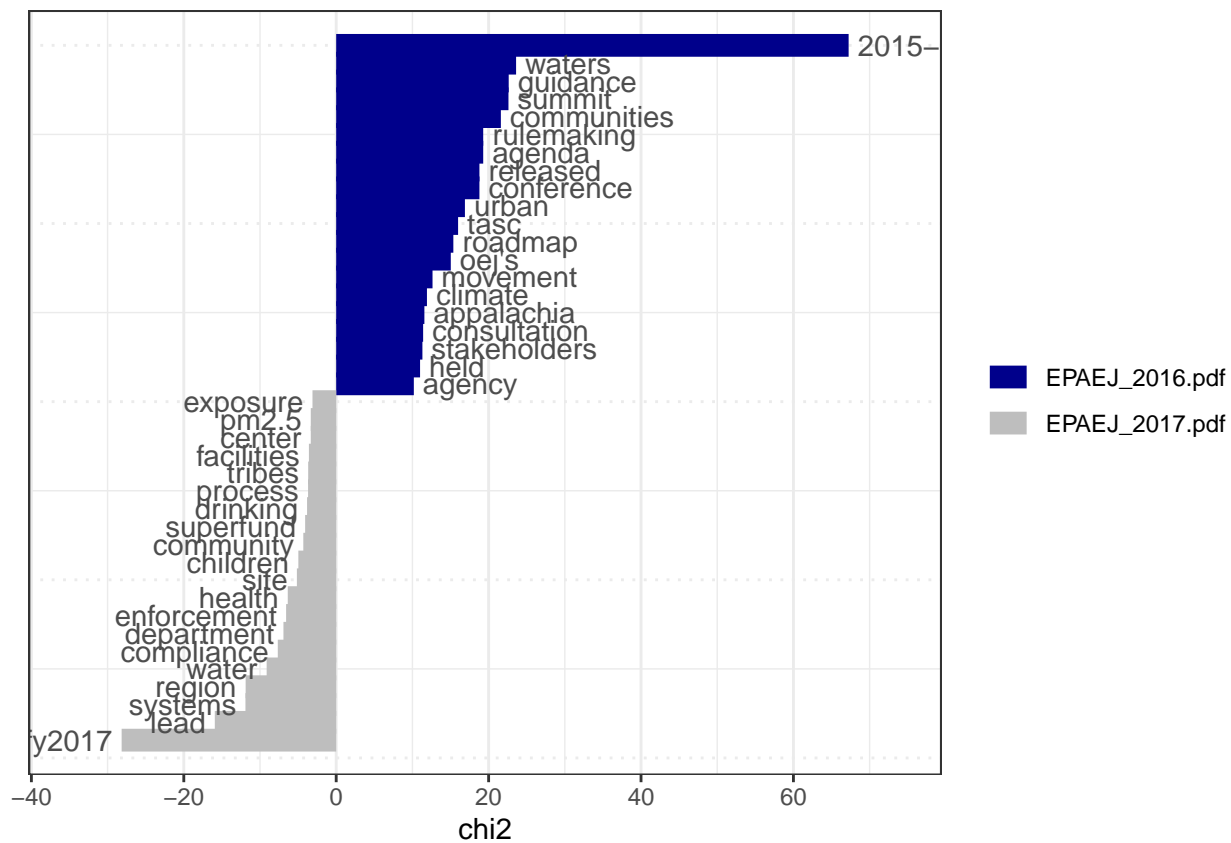
}

```

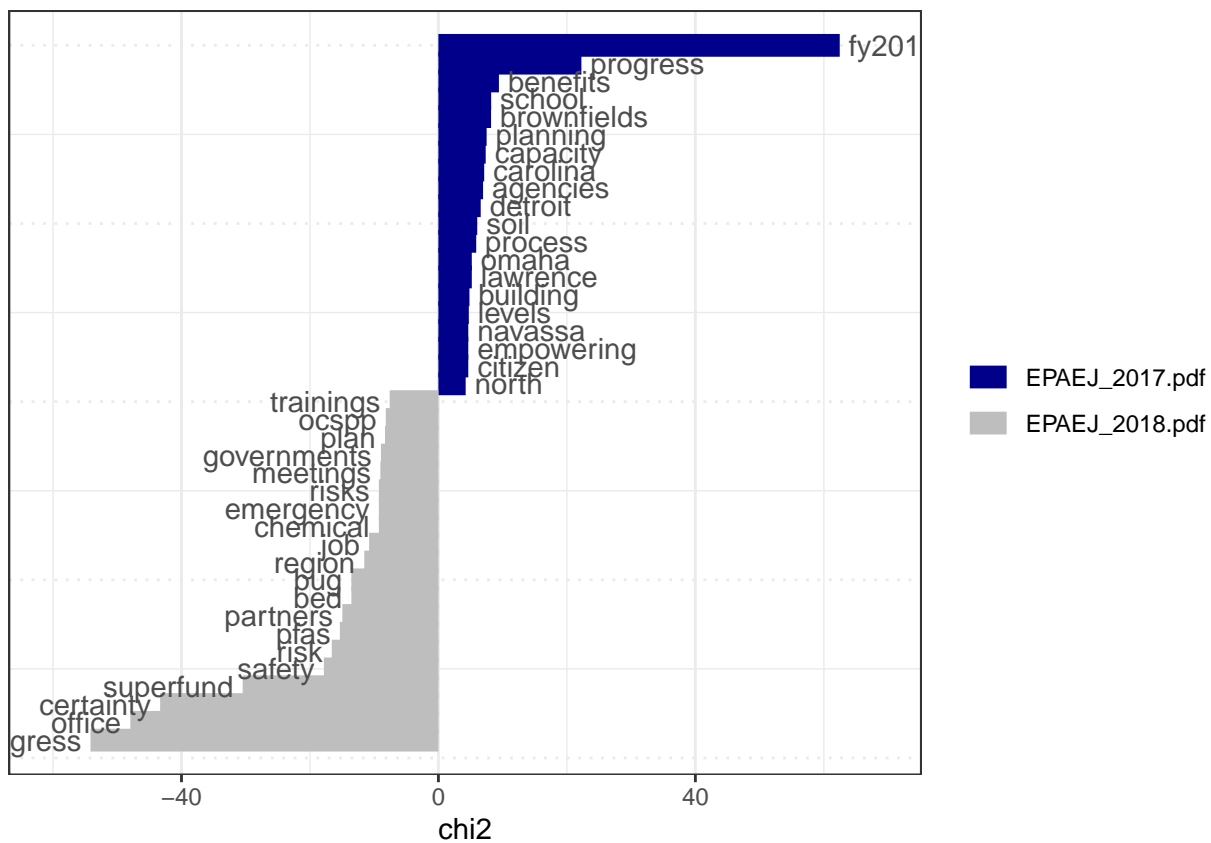
Test the function

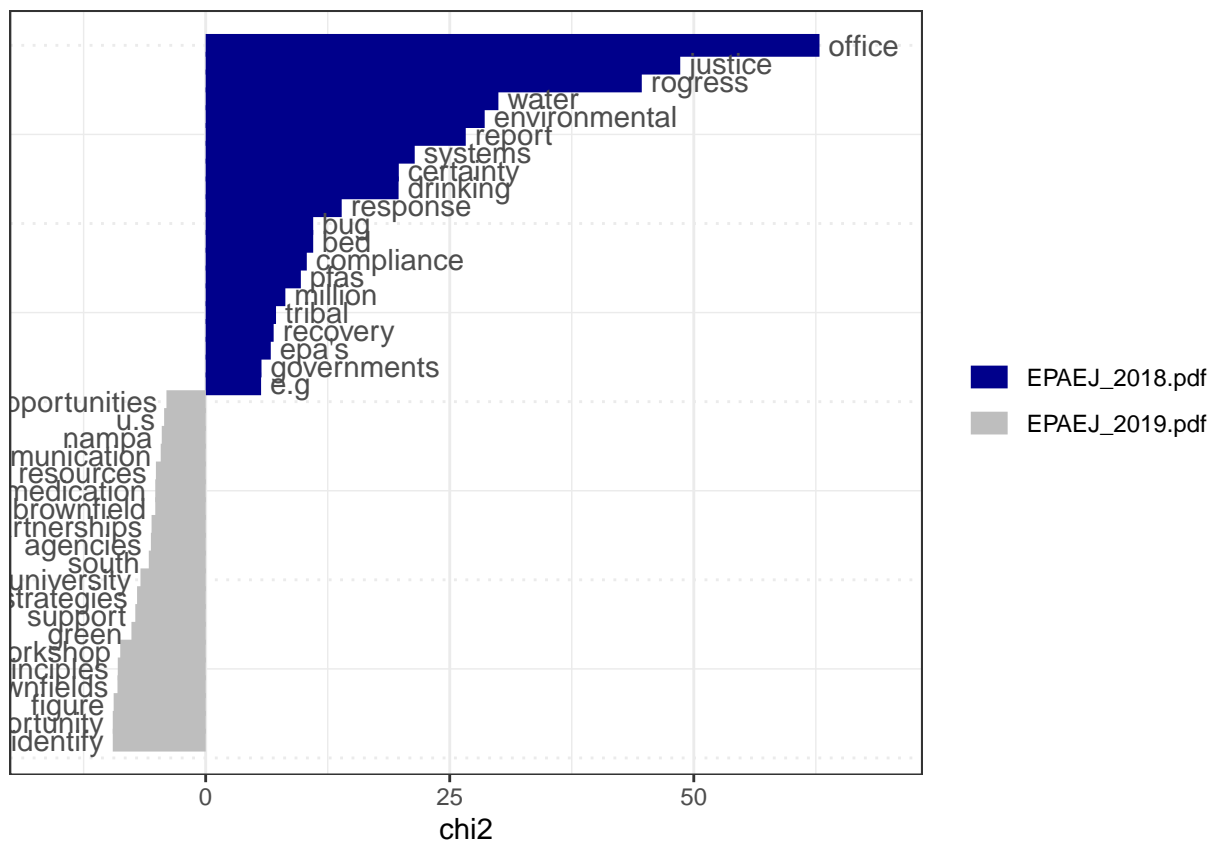
```
keyness_function(report_number = 1)
```

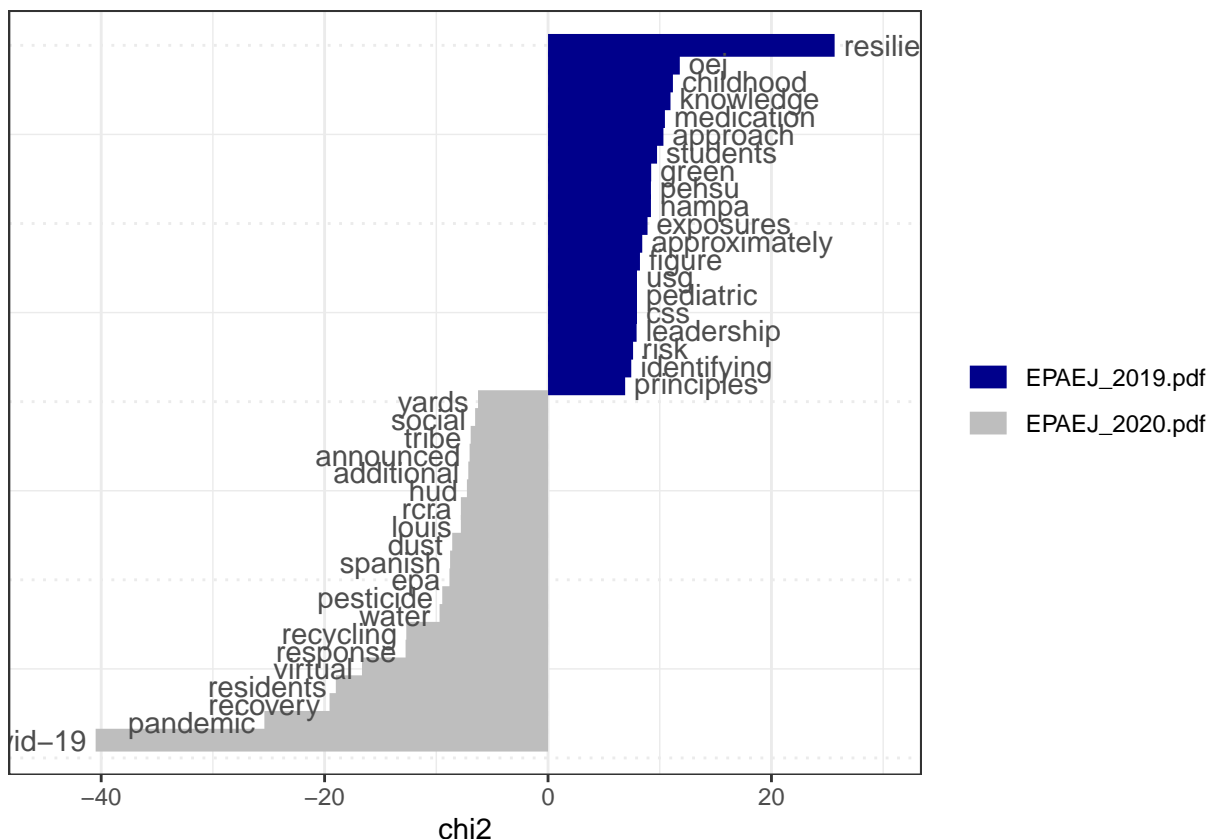












4. Select a word or multi-word term of interest and identify words related to it using windowing and keyness comparison. To do this you will create two objects: one containing all words occurring within a 10-word window of your term of interest, and the second object containing all other words. Then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

Start with the initial corpus and tokenized words

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

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
```

Select 2 token objects for words (1) inside vs (2) outside a 10-word window of the keyword “health

```

#identify keyword
keyword <- c("health")

#1. token object for words inside the 10-word window
toks_inside <- tokens_keep(toks1,
                           pattern = keyword,
                           window = 10)
toks_inside <- tokens_remove(toks_inside,
                             pattern = keyword) # remove the keywords
#create document feature matrix
dfmat_inside <- dfm(toks_inside)

#2. token object for words outside the 10-word window
toks_outside <- tokens_remove(toks1,
                              pattern = keyword, window = 10)
#create document feature matrix
dfmat_outside <- dfm(toks_outside)

tstat_key_inside <- textstat_keyness(rbind(dfmat_inside, dfmat_outside),
                                     target = seq_len(ndoc(dfmat_inside)))
tstat_key_inside[1:10] %>%
  rename("Related Words" = feature,
         "Chi Squared" = chi2) %>%
  kbl() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
               latex_options = "HOLD_position")

```

Related Words	Chi Squared	p	n_target	n_reference
environment	187.48080	0	68	40
public	177.92777	0	137	189
human	154.41312	0	44	16
impacts	106.38418	0	55	52
children's	64.90879	0	18	5
care	63.62066	0	24	15
disparities	61.13043	0	21	10
exposures	60.40295	0	22	13
risks	56.07863	0	24	18
childhood	52.03349	0	23	18