Assignment 5: Word Relationships

Desik Somasundaram

```
library(tidyr) #text analysis in R
library(pdftools)
library(lubridate) #working with date data
library(tidyverse)
library(tidytext)
library(readr)
library(quanteda)
library(readtext) #quanteda subpackage for reading pdf
library(quanteda.textstats)
library(quanteda.textplots)
library(ggplot2)
library(forcats)
library(stringr)
library(quanteda.textplots)
library(widyr)# pairwise correlations
library(igraph) #network plots
library(ggraph)
library(here)
```

Import EPA EJ Data

```
setwd("dat/")
files <- list.files(pattern = "*pdf$")

files <- str_subset(files, pattern="EPA")

ej_reports <- lapply(files, pdf_text)

ej_pdf <- readtext(files, docvarsfrom = "filenames",
                    docvarnames = c("type", "subj", "year"),
                    sep = "_")

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

```
## Corpus consisting of 6 documents, showing 6 documents:
##
##             Text Types Tokens Sentences type subj year
##  EPA_EJ_2015.pdf  2136   8944       263  EPA   EJ 2015
##  EPA_EJ_2016.pdf  1599   7965       176  EPA   EJ 2016
```

```
## EPA_EJ_2017.pdf 3973 30564 653 EPA EJ 2017
## EPA_EJ_2018.pdf 2774 16658 447 EPA EJ 2018
## EPA_EJ_2019.pdf 3773 22648 672 EPA EJ 2019
## EPA_EJ_2020.pdf 4493 30523 987 EPA EJ 2020
```

```
#I'm adding some 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)
```

Tidying and Exploring Data

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)
```

```
## Joining, by = "year"
```

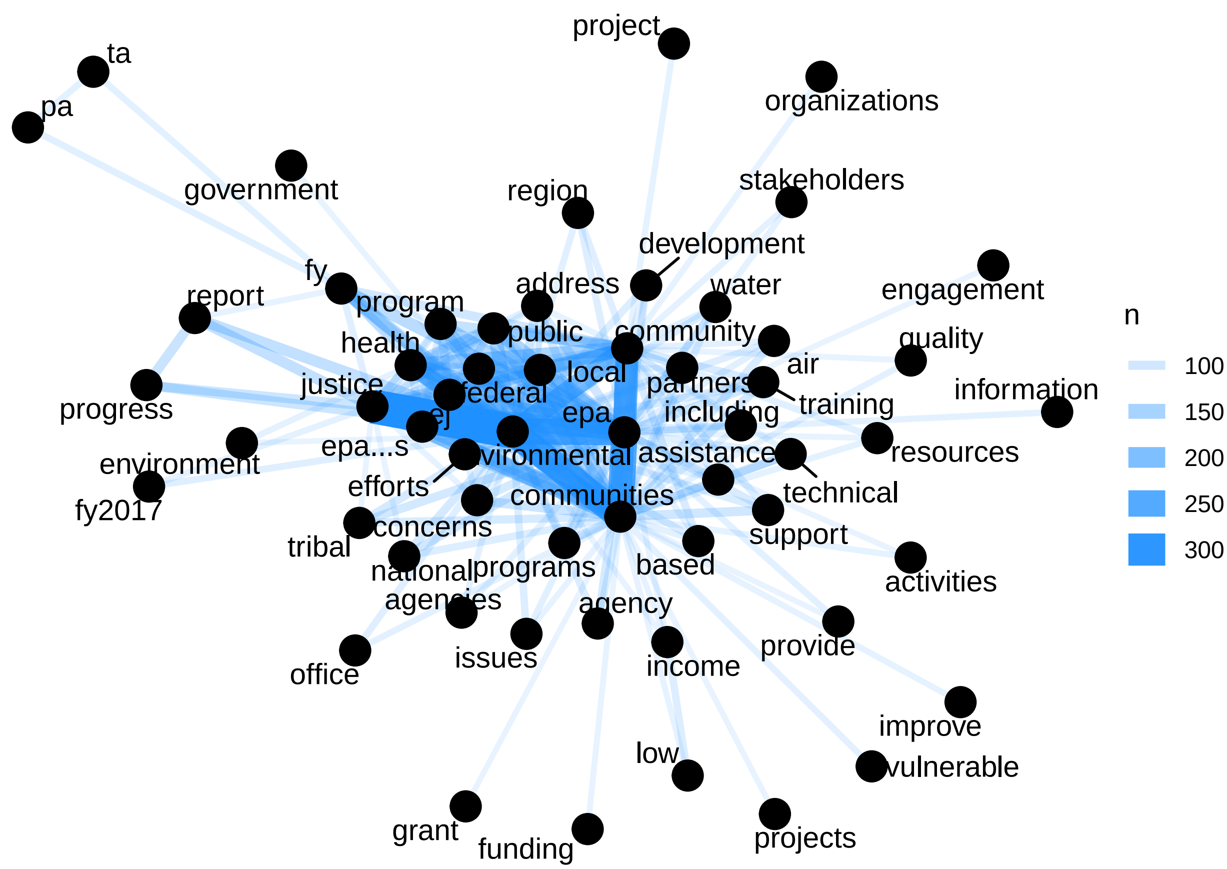
```
par_tokens <- unnest_tokens(raw_text, output = paragraphs, input = text, token = "paragraphs")

par_tokens <- par_tokens %>%
 mutate(par_id = 1:n())

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"))
```

Now we can visualize

```r
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()
```



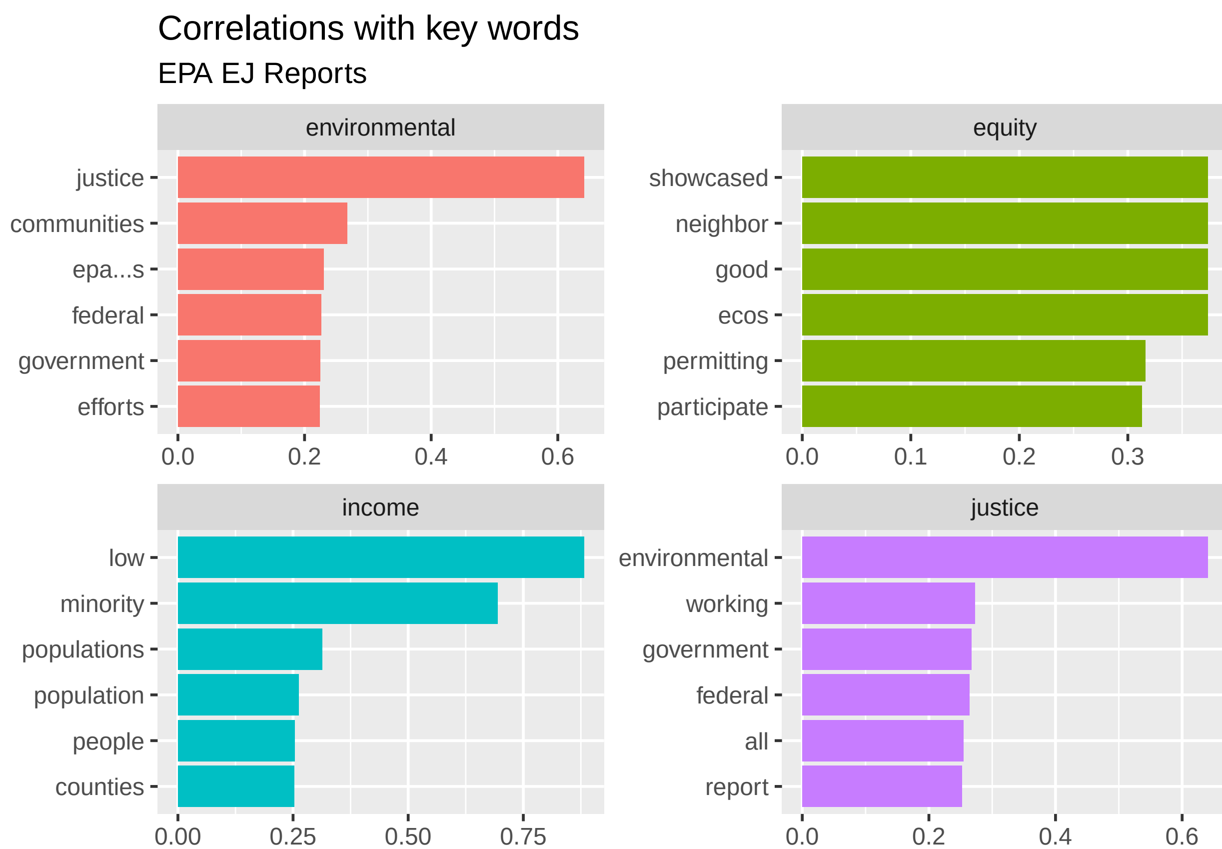
Hmm, interesting, but maybe we further subset the word pairs to get a cleaner picture of the most common ones by raising the cutoff for number of occurrences (n).

Pairs like "environmental" and "justice" are the most common co-occurring words, but that doesn't give us the full picture as they're also the most common individual words. We can also look at correlation among words, which tells us how often they appear together relative to how often they appear separately.

```r
word_cors <- par_words %>%
  add_count(par_id) %>%
  filter(n >= 50) %>%
  select(-n) %>%
  pairwise_cor(word, par_id, sort = TRUE)

just_cors <- word_cors %>%
  filter(item1 == "justice")
```

```
word_cors %>%
filter(item1 %in% c("environmental", "justice", "equity", "income"))%>%
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")
```

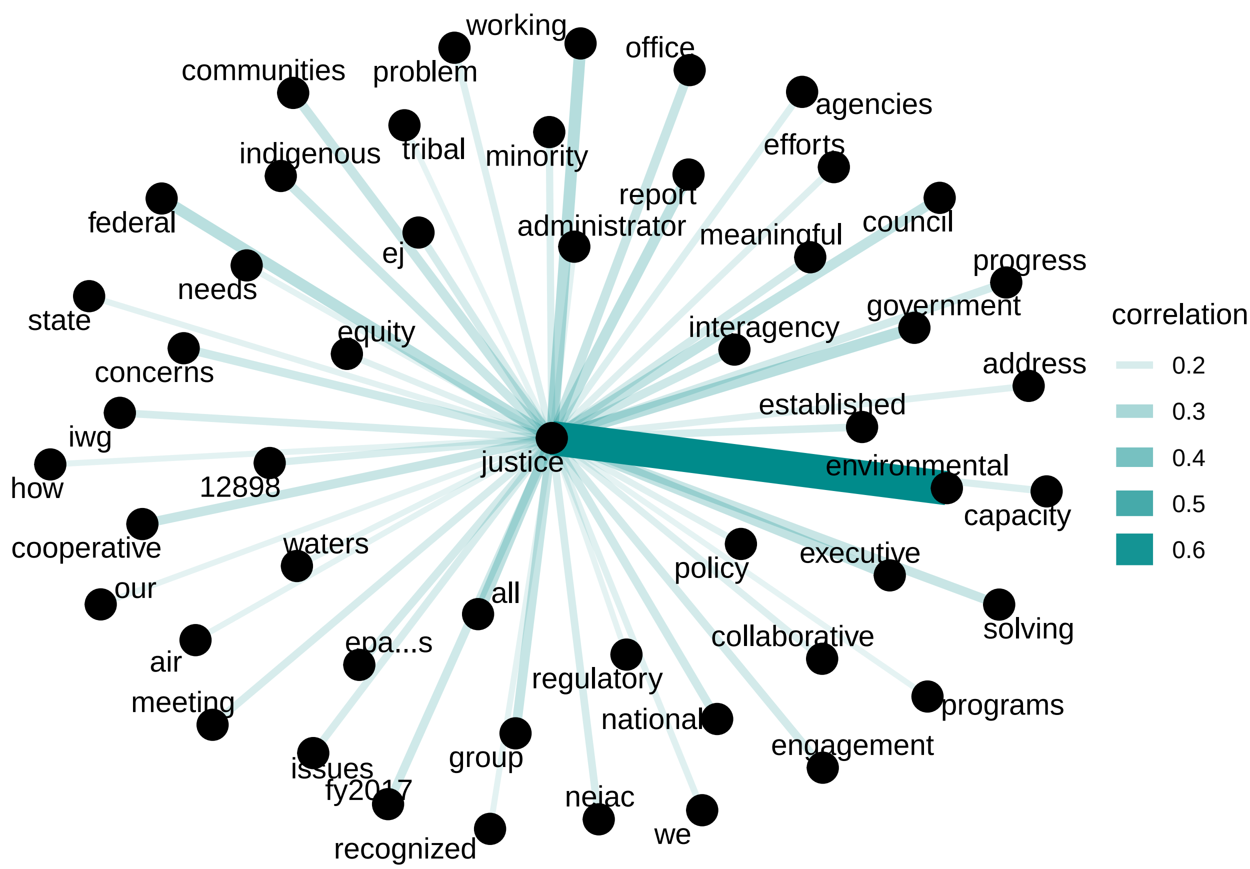
```
## Selecting by correlation
```




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

Not surprisingly, the correlation between "environmental" and "justice" is by far the highest, which makes sense given the nature of these reports. How might we visualize these correlations to develop of sense of the context in which justice is discussed here?

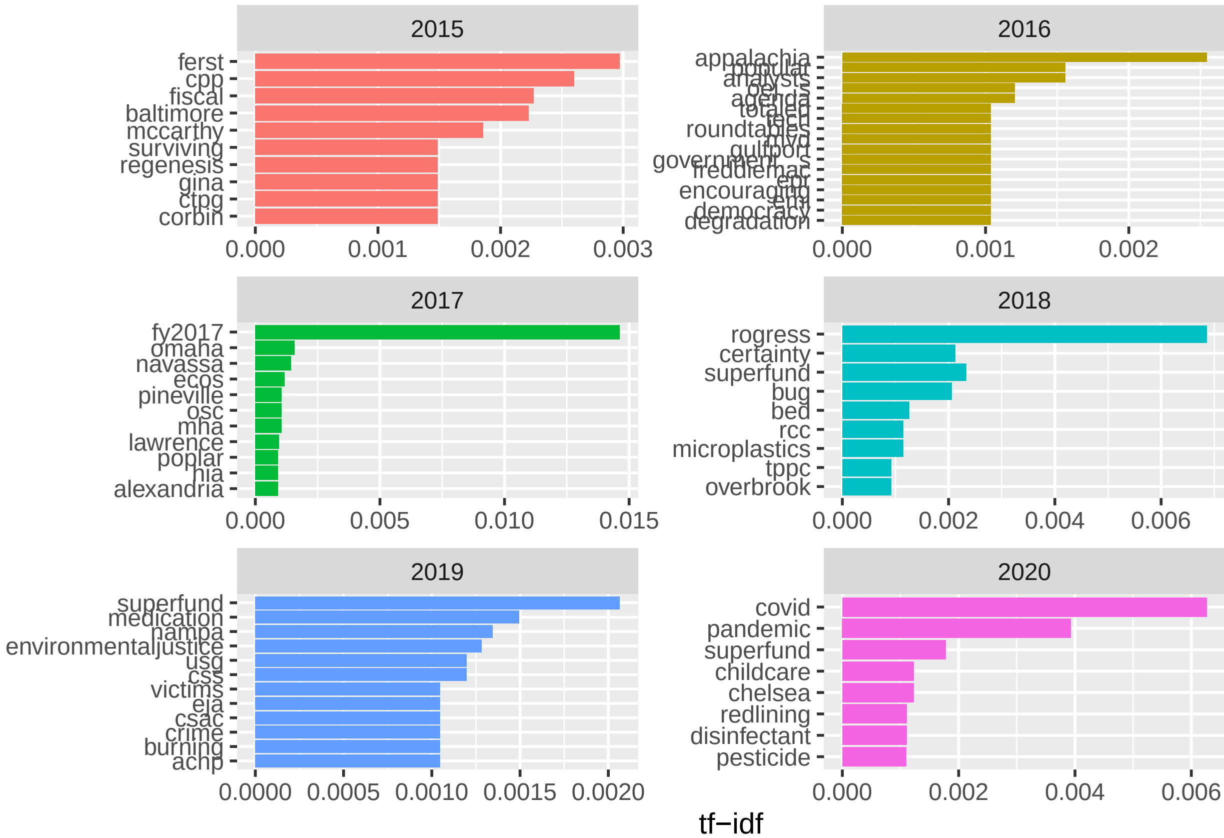
```
justice_cors  %>%
  filter(n <= 50) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = correlation, edge_width = correlation), edge_colour = "cyan4") +
  geom_node_point(size = 5) +
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding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



Now let's look at the tf-idf term we talked about. Remember, this statistic goes beyond simple frequency calculations within a document to control for overall commonality across documents

```
report_tf_idf <- report_words %>%
  bind_tf_idf(word, year, n) %>%
  select(-total) %>%
  arrange(desc(tf_idf))

report_tf_idf %>%
  group_by(year) %>%
  slice_max(tf_idf, n = 10) %>%
  ungroup() %>%
  filter(nchar(word) > 2)%>%
```

```
  ggplot(aes(tf_idf, fct_reorder(word, tf_idf), fill = year)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~year, ncol = 2, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```



So that gives an idea which words are frequent and unique to certain documents.

Now let's switch gears to quanteda for some additional word relationship tools. We'll also get into some ways to assess the similarity of documents.

```
tokens <- tokens(epa_corp, remove_punct = TRUE)
toks1<- tokens_select(tokens, min_nchar = 3)
toks1 <- tokens_tolower(toks1)
toks1 <- tokens_remove(toks1, pattern = (stop_vec))
dfm <- dfm(toks1)

#first the basic frequency stat
tstat_freq <- textstat_frequency(dfm, n = 5, groups = year)
head(tstat_freq, 10)
```

```
##          feature frequency rank docfreq group
## 1  environmental       127    1       1  2015
## 2    communities        99    2       1  2015
## 3            epa        92    3       1  2015
## 4        justice        84    4       1  2015
## 5      community        47    5       1  2015
```

```
## 6  environmental       109    1        1  2016
## 7    communities        85    2        1  2016
## 8        justice        71    3        1  2016
## 9            epa        48    4        1  2016
## 10       federal        31    5        1  2016
```

Another useful word relationship concept is that of the n-gram, which essentially means tokenizing at the multi-word level

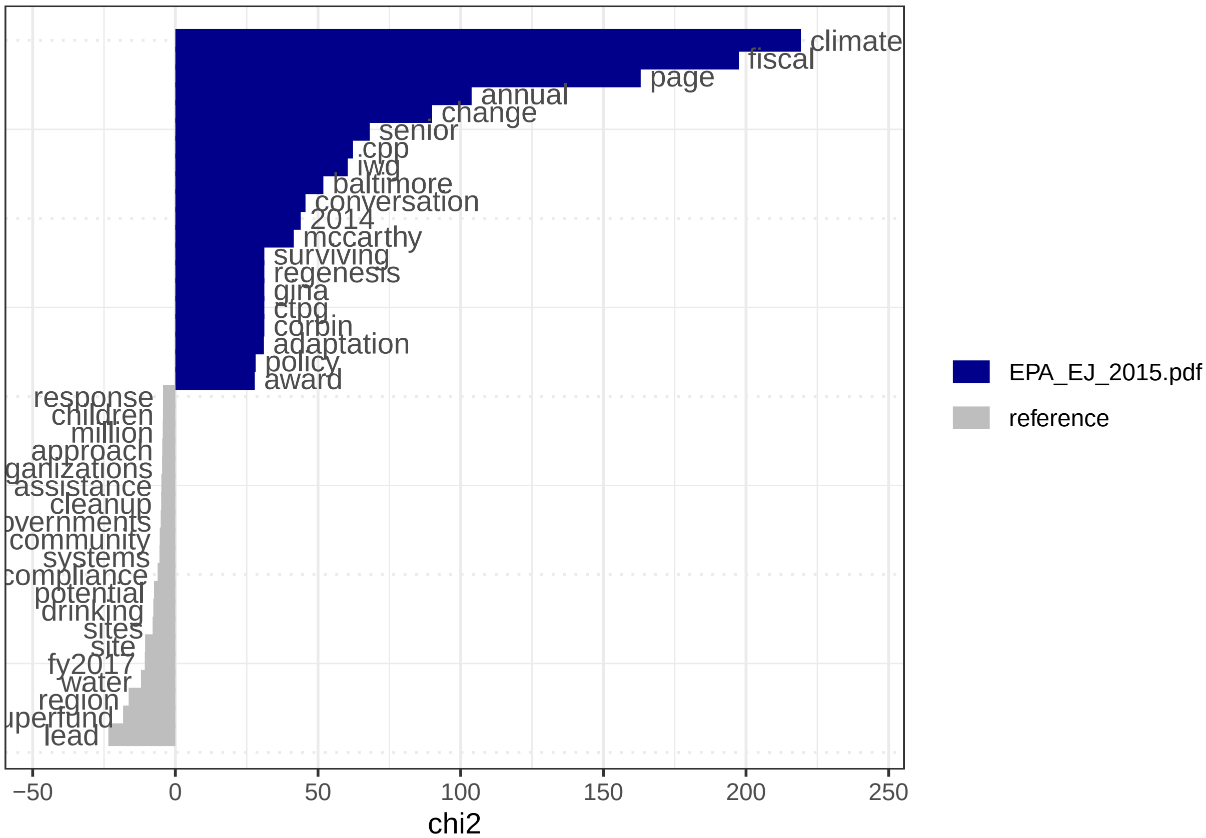
```
toks2 <- tokens_ngrams(toks1, n=2)
dfm2 <- dfm(toks2)
dfm2 <- dfm_remove(dfm2, pattern = c(stop_vec))
freq_words2 <- textstat_frequency(dfm2, n=20)
freq_words2$token <- rep("bigram", 20)
freq_words2
```

```
##                    feature frequency rank docfreq group  token
## 1     environmental_justice       556    1       6   all bigram
## 2      technical_assistance       139    2       6   all bigram
## 3            drinking_water       133    3       6   all bigram
## 4             public_health       123    4       6   all bigram
## 5           progress_report       108    5       6   all bigram
## 6               air_quality        73    6       6   all bigram
## 7             water_systems        66    7       6   all bigram
## 8    vulnerable_communities        65    8       6   all bigram
## 9                epa_region        62    9       5   all bigram
## 10     environmental_public        57   10       6   all bigram
## 11          federal_agencies        56   11       6   all bigram
## 12   national_environmental        51   12       6   all bigram
## 13            justice_fy2017        51   12       1   all bigram
## 14           fy2017_progress        51   12       1   all bigram
## 15           superfund_sites        48   15       4   all bigram
## 16        indigenous_peoples        46   16       6   all bigram
## 17              civil_rights        46   16       5   all bigram
## 18        local_governments        45   18       6   all bigram
## 19              urban_waters        44   19       6   all bigram
## 20 overburdened_communities        43   20       6   all bigram
```

```
#tokens1 <- tokens_select(tokens1,pattern = stopwords("en"), selection = "remove")
```

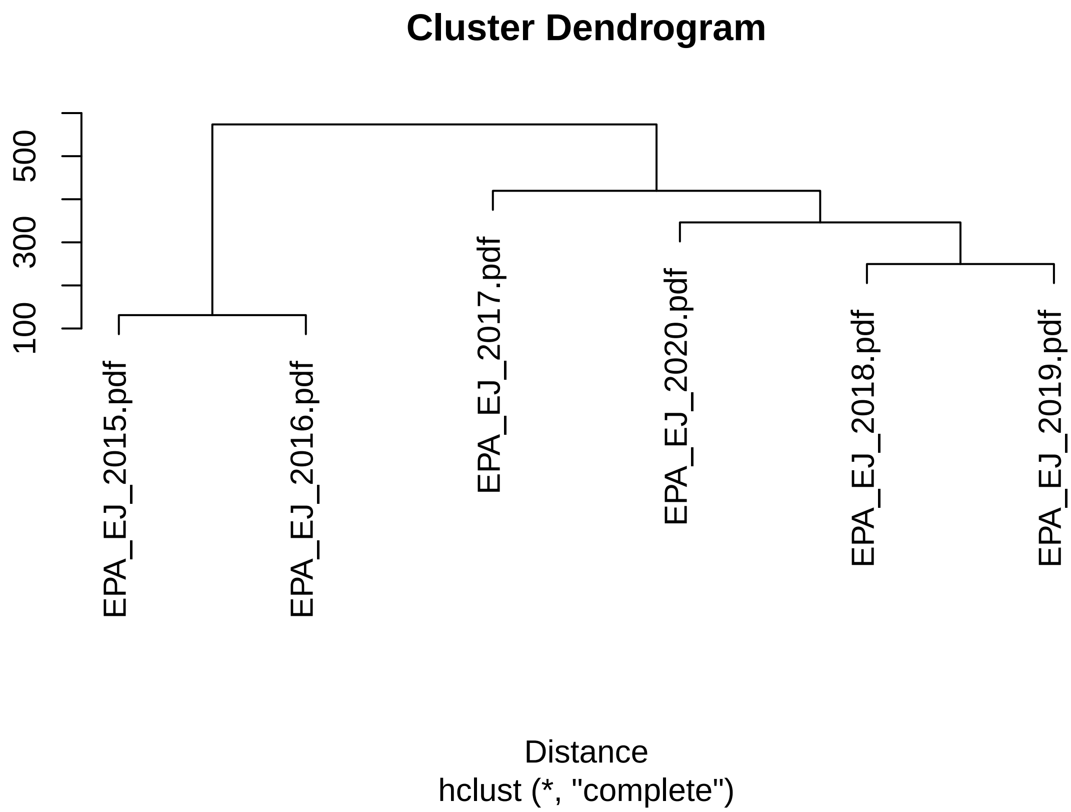
Now we can upgrade that by using all of the frequencies for each word in each document and calculating a chi-square to see which words occur significantly more or less within a particular target document

```
keyness <- textstat_keyness(dfm, target = 1)
textplot_keyness(keyness)
```



And finally, we can run a hierarchical clustering algorithm to assess document similarity. This tends to be more informative when you are dealing with a larger number of documents, but we'll add it here for future reference.

```
dist <- as.dist(textstat_dist(dfm))
clust <- hclust(dist)
plot(clust, xlab = "Distance", ylab = NULL)
```



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?

```
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)
freq_words2
```

```
##                     feature frequency rank docfreq group  token
## 1     environmental_justice       556    1       6   all bigram
## 2      technical_assistance       139    2       6   all bigram
## 3            drinking_water       133    3       6   all bigram
## 4             public_health       123    4       6   all bigram
## 5           progress_report       108    5       6   all bigram
## 6               air_quality        73    6       6   all bigram
## 7             water_systems        66    7       6   all bigram
## 8    vulnerable_communities        65    8       6   all bigram
## 9                epa_region        62    9       5   all bigram
## 10     environmental_public        57   10       6   all bigram
## 11         federal_agencies        56   11       6   all bigram
```

```
## 12   national_environmental        51    12        6   all bigram
## 13            justice_fy2017        51    12        1   all bigram
## 14           fy2017_progress        51    12        1   all bigram
## 15           superfund_sites        48    15        4   all bigram
## 16        indigenous_peoples        46    16        6   all bigram
## 17              civil_rights        46    16        5   all bigram
## 18         local_governments        45    18        6   all bigram
## 19              urban_waters        44    19        6   all bigram
## 20 overburdened_communities        43    20        6   all bigram
```

```
freq_words3
```

```
##                                feature frequency rank docfreq group   token
## 1             justice_fy2017_progress        51    1       1   all trigram
## 2              fy2017_progress_report        51    1       1   all trigram
## 3         environmental_public_health        50    3       6   all trigram
## 4        environmental_justice_fy2017        50    3       1   all trigram
## 5      national_environmental_justice        37    5       6   all trigram
## 6        office_environmental_justice        32    6       6   all trigram
## 7         epa's_environmental_justice        32    6       6   all trigram
## 8      environmental_justice_progress        30    8       4   all trigram
## 9             justice_progress_report        30    8       4   all trigram
## 10     environmental_justice_concerns        30    8       5   all trigram
## 11             drinking_water_systems        29   11       5   all trigram
## 12       annual_environmental_justice        27   12       5   all trigram
## 13     environmental_justice_advisory        27   12       6   all trigram
## 14        fiscal_annual_environmental        25   14       3   all trigram
## 15           justice_advisory_council        24   15       6   all trigram
## 16       environmental_justice_grants        22   16       5   all trigram
## 17    technical_assistance_communities       20   17       6   all trigram
## 18 communities_environmental_justice        20   17       5   all trigram
## 19                 safe_drinking_water       19   19       5   all trigram
## 20      technical_assistance_services        19   19       5   all trigram
```

```
#tokens1 <- tokens_select(tokens1,pattern = stopwords("en"), selection = "remove")
```

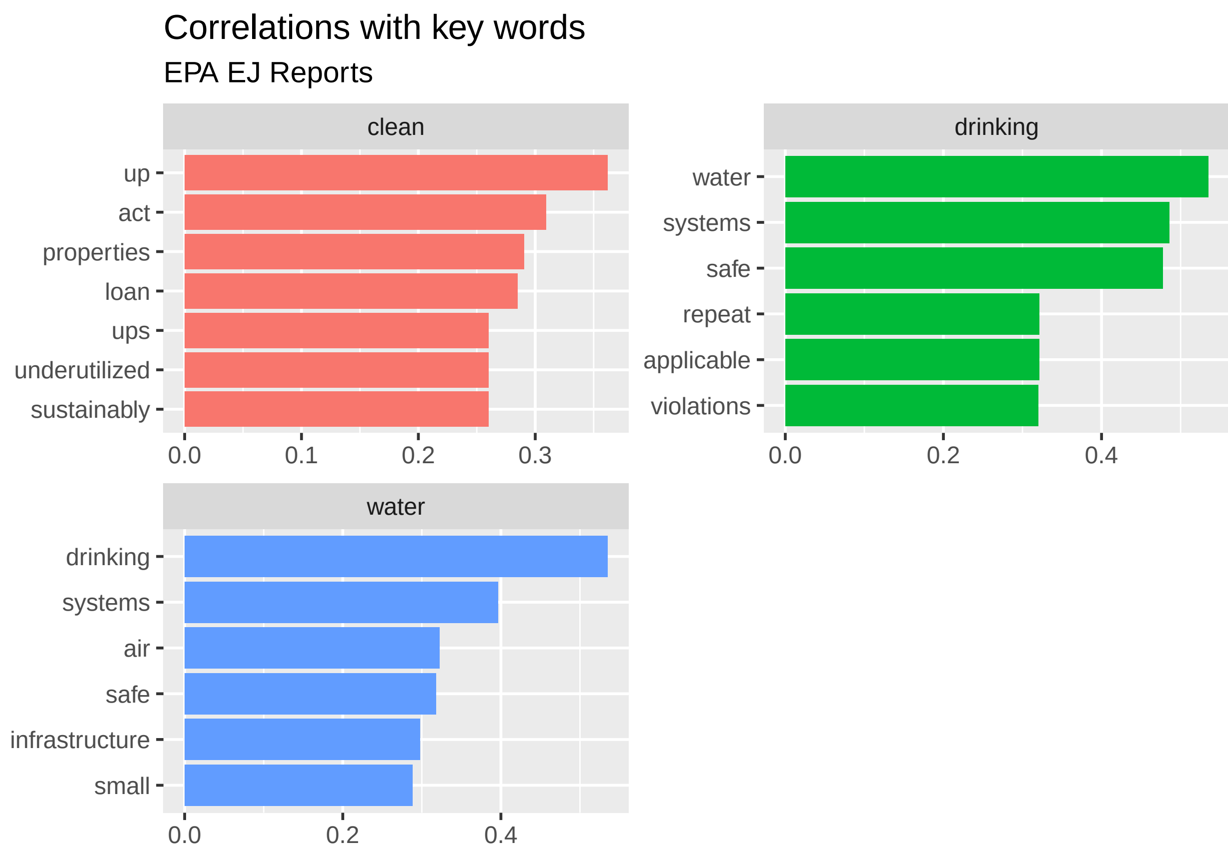
The most popular tri-grams are justice_fy2017_progress, fy2017_progress_report and environmental_public_health. The most popular bi-grams are environmental_justice, technical_assistance and drinking_water. In this case, the bi-grams seem much more useful because the tri-grams contain a lot of noise with most trigrams contains repeats of same keywords arranged differently.

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!

```
word_cors <- par_words %>%
  add_count(par_id) %>%
  filter(n >= 50) %>%
```

```
  select(-n) %>%
  pairwise_cor(word, par_id, sort = TRUE)

wat_cors <- word_cors %>%
  filter(item1 == "water")

  word_cors %>%
  filter(item1 %in% c("clean", "water", "scarcity", "drinking"))%>%
  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")
```

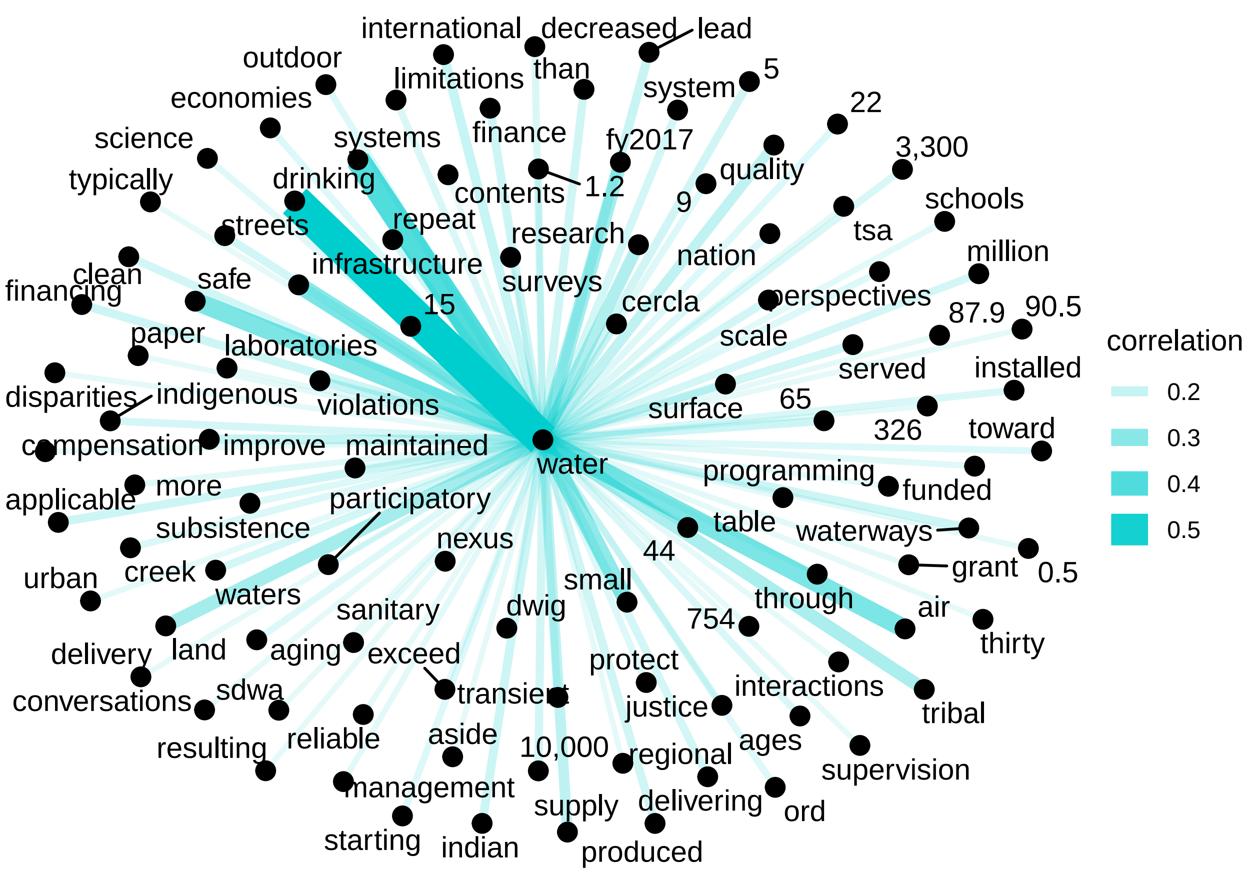
```
## Selecting by correlation
```



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

Water as a focal term

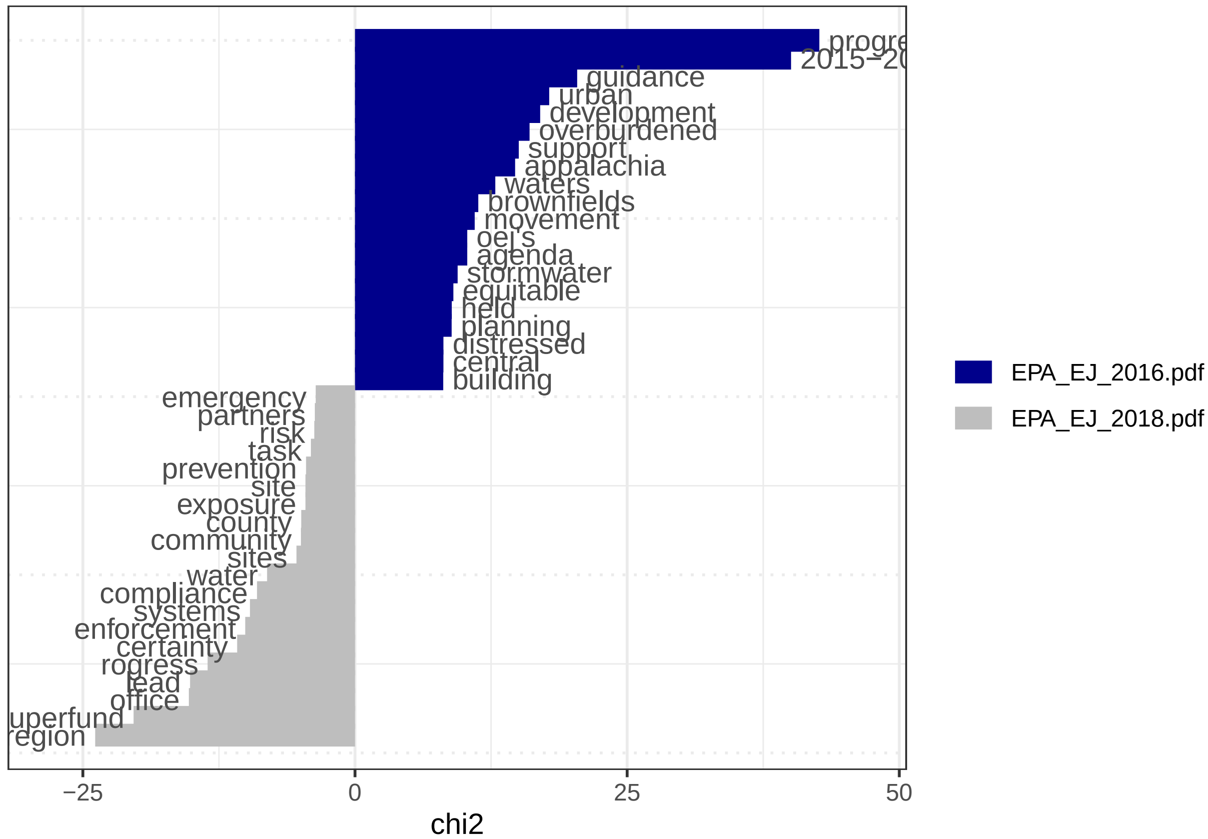
```r
water_cors  %>%
  filter(n <= 100) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = correlation, edge_width = correlation), edge_colour = "cyan3") +
  geom_node_point(size = 3) +
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding = unit(0.3, "lines")) +
  theme_void()
```



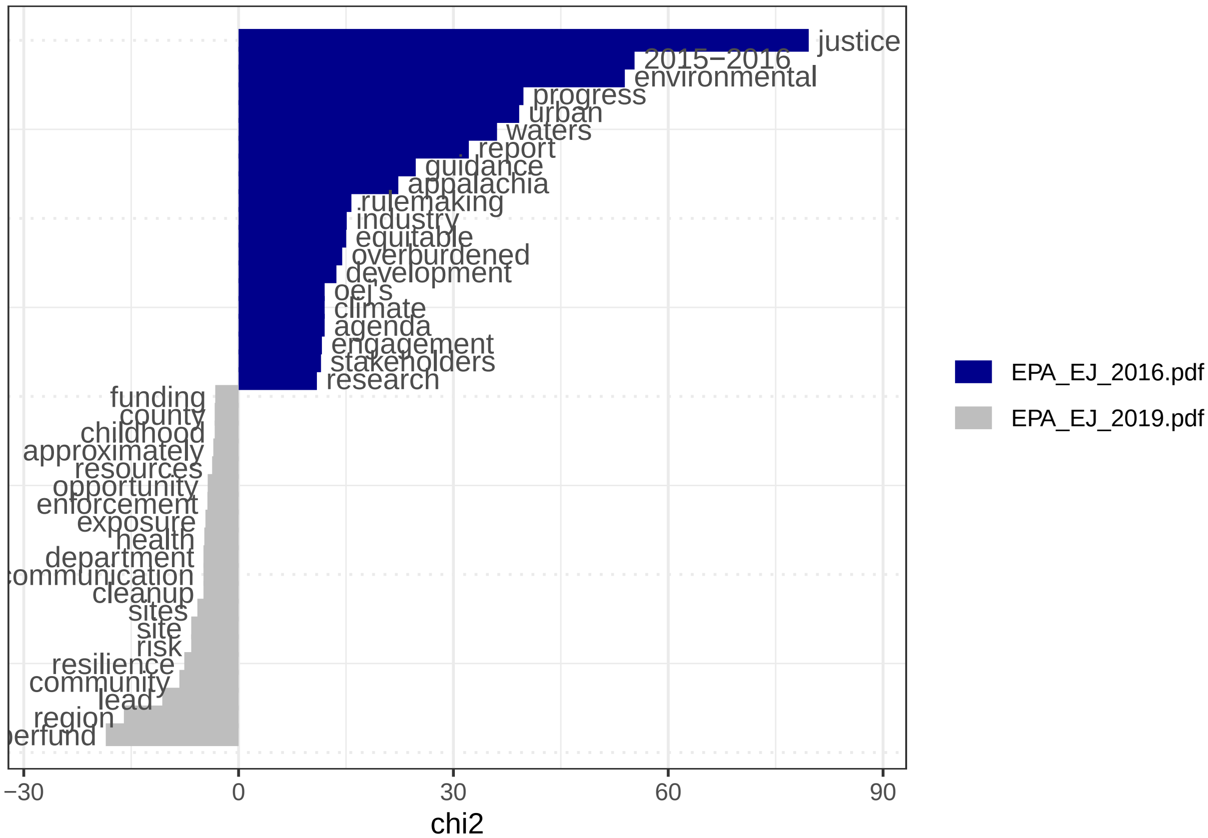
3. Write a function that allows you to conduct a keyness analysis to compare two individual EPA reports (hint: that means target and reference need to both be individual reports). Run the function on 3 pairs of reports, generating 3 keyness plots.

```
keyness_comp <- function(report1, report2){
  files <- c(report1,report2)
  reports <- lapply(files, pdf_text)
  pdf <- readtext(files, docvarsfrom = "filenames",
                  docvarnames = c("type", "subj", "year"),
                  sep = "_")

  #creating an initial corpus containing our data
  corp <- corpus(x = pdf, text_field = "text" )
  tokens <- tokens(corp, remove_punct = TRUE)
  toks1<- tokens_select(tokens, min_nchar = 3)
  toks1 <- tokens_tolower(toks1)
  toks1 <- tokens_remove(toks1, pattern = (stop_vec))
  dfm <- dfm(toks1)
  keyness <- textstat_keyness(dfm, target = 1)

  return(keyness)
}
```

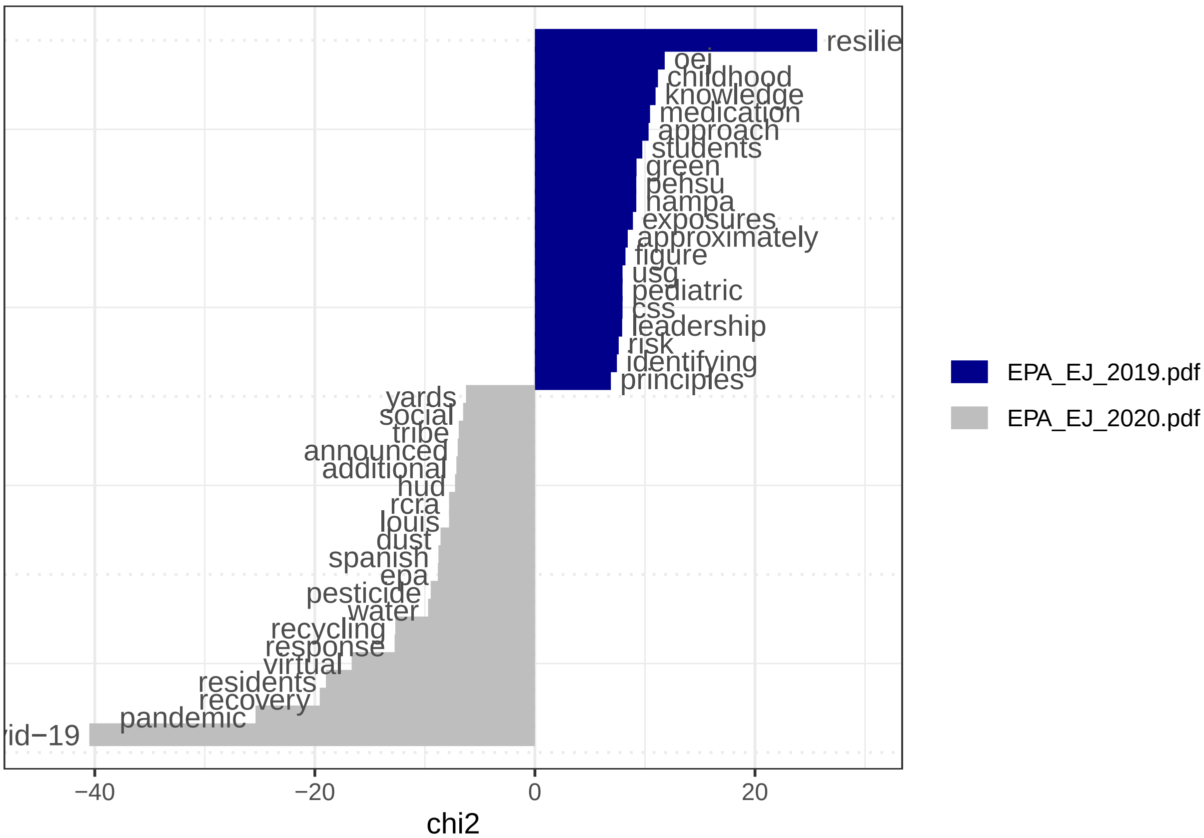
```
keyness_computed_1<-keyness_comp(here("dat", "EPA_EJ_2016.pdf"), here("dat", "EPA_EJ_2018.pdf"))
textplot_keyness(keyness_computed_1)
```




```
keyness_computed_2<-keyness_comp(here("dat", "EPA_EJ_2016.pdf"), here("dat", "EPA_EJ_2019.pdf"))
textplot_keyness(keyness_computed_2)
```




```
keyness_computed_3<-keyness_comp(here("dat", "EPA_EJ_2019.pdf"), here("dat", "EPA_EJ_2020.pdf"))
textplot_keyness(keyness_computed_3)
```



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?

Water Quality as multi-word term of interest

```
water_quality <- c("water", "water quality")
toks_inside <- tokens_keep(tokens, pattern = water_quality, window = 10)
toks_inside <- tokens_remove(toks_inside, pattern = water_quality) # remove the keywords
toks_outside <- tokens_remove(tokens, pattern = water_quality, window = 10)
```

```
dfmat_inside <- dfm(toks_inside)
dfmat_outside <- dfm(toks_outside)

tstat_key_inside <- textstat_keyness(rbind(dfmat_inside, dfmat_outside),
                                     target = seq_len(ndoc(dfmat_inside)))
head(tstat_key_inside, 25)
```

```
##          feature       chi2            p n_target n_reference
## 1       drinking 1983.86441 0.000000e+00      138           3
```

```
## 2          systems 1026.45693 0.000000e+00        92           25
## 3   infrastructure  197.22605 0.000000e+00        36           45
## 4            small  192.98029 0.000000e+00        43           69
## 5           system  182.53698 0.000000e+00        27           23
## 6             safe  178.65850 0.000000e+00        24           17
## 7     health-based  161.52831 0.000000e+00        13            1
## 8             land  115.86858 0.000000e+00        23           30
## 9            clean  113.60064 0.000000e+00        30           58
## 10          served   75.98434 0.000000e+00         9            4
## 11          tribal   75.41803 0.000000e+00        49          198
## 12      applicable   73.25277 0.000000e+00         6            0
## 13         quality   60.87612 6.106227e-15        33          118
## 14             air   54.92440 1.252332e-13        44          202
## 15          supply   52.24150 4.908296e-13         6            2
## 16     supervision   47.42411 5.717538e-12         5            1
## 17         finance   45.26037 1.725042e-11         6            3
## 18   non-community   44.17991 2.995404e-11         4            0
## 19       utilities   39.68904 2.977918e-10         6            4
## 20         surface   39.40271 3.448182e-10         5            2
## 21      delivering   37.70655 8.222780e-10         7            7
## 22      violations   37.70655 8.222780e-10         7            7
## 23       financing   33.97142 5.592754e-09         4            1
## 24          repeat   33.40363 7.488399e-09         5            3
## 25      wastewater   32.04017 1.510176e-08         9           16
```

The tokens inside the 10-word window are the target while the tokens outside the 10-word window are the reference.