# EDS_231_assignment_1: New York Times API

Marie Rivers

4/10/2022

## New York Times API

This assignment looks at New York Times articles that contain the term ‘PFAS’ which refers to perfluoroalkyl and polyfluoroalky compounds found in over 4,000 man-made chemicals. This class of compounds is often referred to as ‘forever chemicals’ because they do not breakdown in the environment. PFAS chemicals are know or suspected to cause a wide range of health problems such as cancer, weakened immune system, and thyroid disease. PFAS has been detected in drinking water supplies throughout the United States.

This text analysis uses the New York Times API to look at the frequency of PFAS related articles between 2000 and 2022 and common words contained in those articles.

```
library(jsonlite) #convert results from API queries into R-friendly formats
library(tidyverse)
library(tidytext) #text data management and analysis
library(ggplot2) #plot word frequencies and publication dates
library(here)
library(kableExtra)
```

### Connect to the New York Times API, set parameters, and send a query

```
api_key <- "GW3whn8dTpvpdcAD1AIAiUMix03szFDn" # article search api
term <- "pfas" # Need to use + to string together separate words
begin_date <- "20000101" # YYYYMMDD
end_date <- "20220401" #YYYYMMDD

#construct the query url using API operators
baseurl <- paste0("http://api.nytimes.com/svc/search/v2/articlesearch.json?q=",
                  term, "&begin_date=",begin_date, "&end_date=", end_date,
                  "&facet_filter=true&api-key=", api_key, sep="")
```

```
# this code allows for obtaining multiple pages of query results
initialQuery <- fromJSON(baseurl)
maxPages <- round((initialQuery$response$meta$hits[1] / 10) - 1) # might time out at 8 or 9 pages

pages <- list()
for(i in 0:maxPages){
  nytSearch <- fromJSON(paste0(baseurl, "&page=", i), flatten = TRUE) %>% data.frame()
  message("Retrieving page ", i)
  pages[[i+1]] <- nytSearch
  Sys.sleep(6) # change Sys.sleep to 6
}
```

```
## Retrieving page 0

## Retrieving page 1

## Retrieving page 2

## Retrieving page 3
```

```
class(pages) # this is a list
```

```
## [1] "list"
```

```
class(nytSearch) # this is a data.frame
```

```
## [1] "data.frame"
```

```
#Inspect the data
dim(nytSearch) # how big is it?
```

```
## [1] 10 33
```

```
names(nytSearch) # what variables are we working with?
```

```
##  [1] "status"
##  [2] "copyright"
##  [3] "response.docs.abstract"
##  [4] "response.docs.web_url"
##  [5] "response.docs.snippet"
##  [6] "response.docs.lead_paragraph"
##  [7] "response.docs.print_section"
##  [8] "response.docs.print_page"
##  [9] "response.docs.source"
## [10] "response.docs.multimedia"
## [11] "response.docs.keywords"
## [12] "response.docs.pub_date"
## [13] "response.docs.document_type"
## [14] "response.docs.news_desk"
## [15] "response.docs.section_name"
## [16] "response.docs.type_of_material"
## [17] "response.docs._id"
## [18] "response.docs.word_count"
## [19] "response.docs.uri"
## [20] "response.docs.subsection_name"
## [21] "response.docs.headline.main"
## [22] "response.docs.headline.kicker"
## [23] "response.docs.headline.content_kicker"
## [24] "response.docs.headline.print_headline"
## [25] "response.docs.headline.name"
## [26] "response.docs.headline.seo"
## [27] "response.docs.headline.sub"
## [28] "response.docs.byline.original"
## [29] "response.docs.byline.person"
## [30] "response.docs.byline.organization"
## [31] "response.meta.hits"
## [32] "response.meta.offset"
## [33] "response.meta.time"
```

```
#need to bind the pages and create a tibble
nytDat <- rbind_pages(pages)
class(nytDat)
```

```
## [1] "data.frame"
```

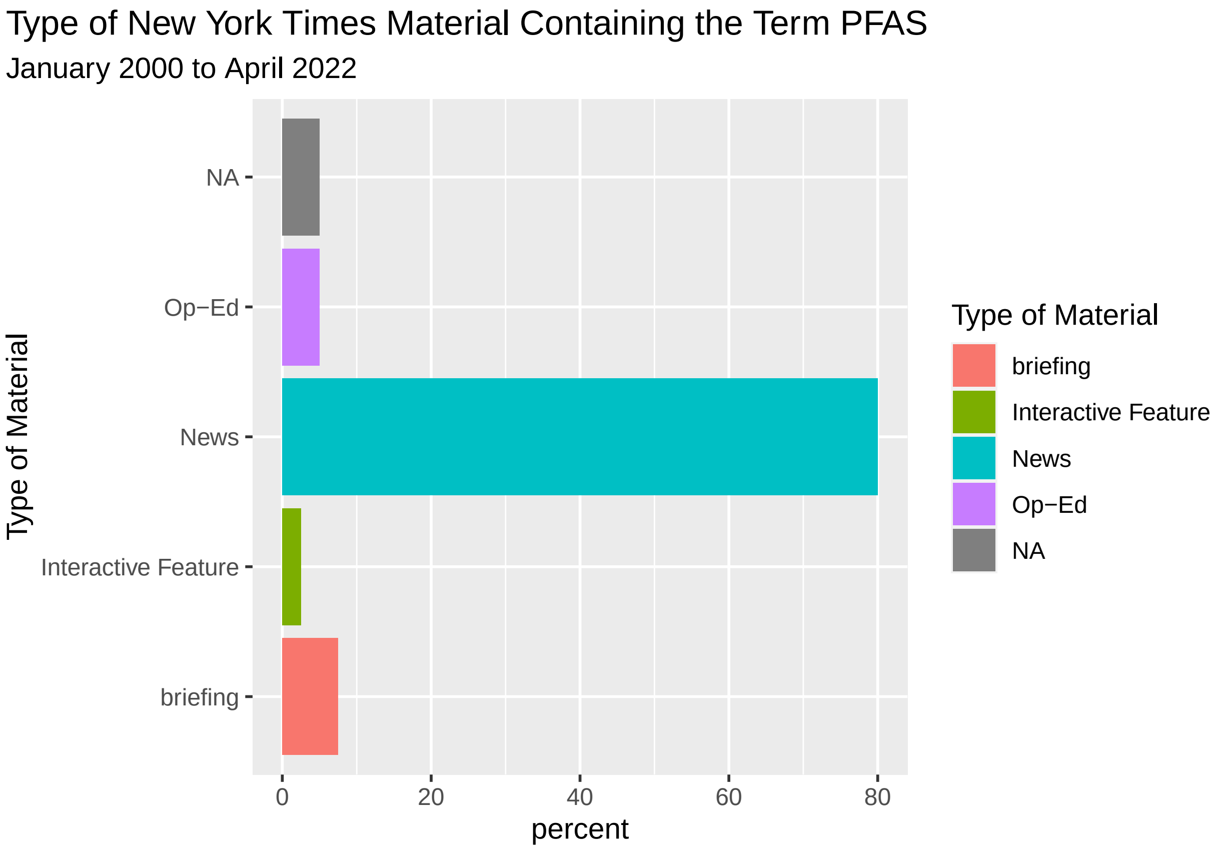
```
# this might be a good place to export to csv as a backup in case the api times out
write_csv(nytDat, here("data", "nytDat.csv"))
```

```
# backup
#nytDat <- read_csv("data/nytDat.csv")
```

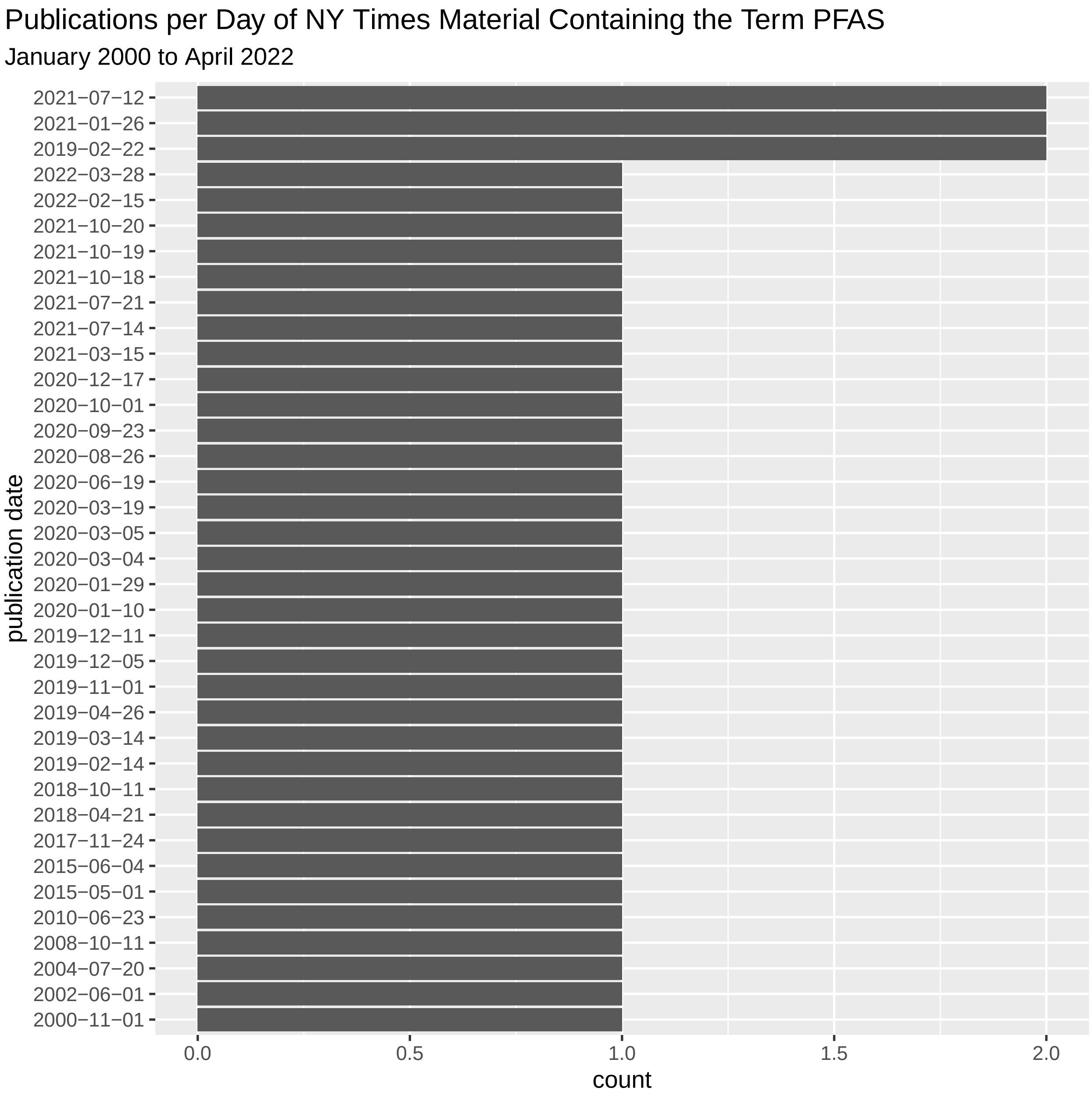
```
nytDat_count <- nytDat %>%
  summarize(count=n()) %>%
  as.numeric()

news_type_pct <- nytDat %>%
  group_by(response.docs.type_of_material) %>%
  summarize(count=n()) %>%
  mutate(percent = (count / sum(count))*100) %>%
  filter(response.docs.type_of_material == "News") %>%
  select(percent) %>%
  as.numeric()
```

This search resulted in 40 articles and 80% of these articles were news articles.

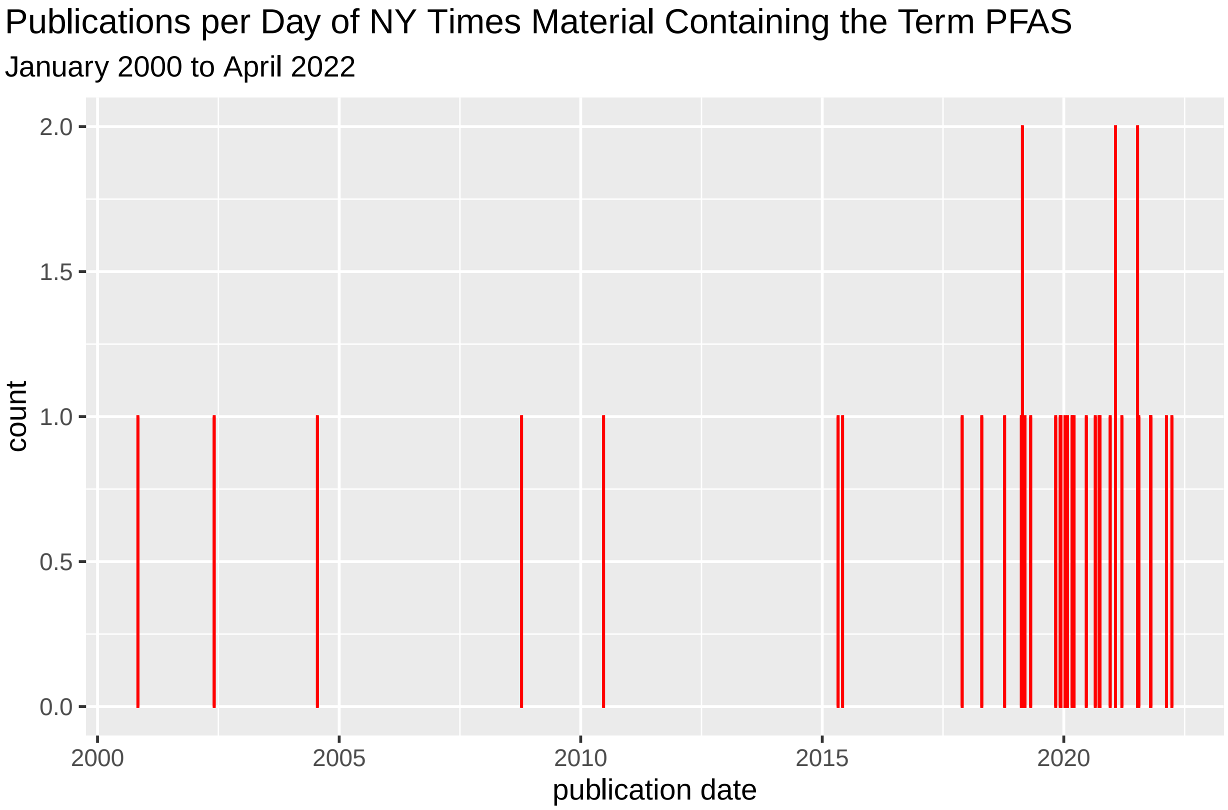
```
nytDat %>%
  group_by(response.docs.type_of_material) %>%
  summarize(count=n()) %>%
  mutate(percent = (count / sum(count))*100) %>%
  ggplot() +
  geom_bar(aes(y=percent, x=response.docs.type_of_material, fill=response.docs.type_of_material), stat =
  labs(x = "Type of Material", fill = "Type of Material",
       title = "Type of New York Times Material Containing the Term PFAS",
       subtitle = "January 2000 to April 2022") +
   theme(plot.title.position = "plot")
```



```
nytDat %>%
  mutate(pubDay=gsub("T.*","",response.docs.pub_date)) %>%
  group_by(pubDay) %>%
  summarise(count=n()) %>%
  filter(count >= 1) %>% # change this if needed, based on search results
  ggplot() +
  geom_bar(aes(x=reorder(pubDay, count), y=count), stat="identity") + coord_flip() +
  labs(x = "publication date",
       title = "Publications per Day of NY Times Material Containing the Term PFAS",
       subtitle = "January 2000 to April 2022") +
   theme(plot.title.position = "plot")
```



```
nytDat %>%
  mutate(pubDay=gsub("T.*","",response.docs.pub_date)) %>%
  group_by(pubDay) %>%
  summarise(count=n()) %>%
  filter(count >= 1) %>% # change this if needed, based on search results
  mutate(pubDay = as.Date(pubDay)) %>%
  ggplot() +
  geom_bar(aes(x=pubDay, y=count), stat="identity", color = "red") +
  labs(x = "publication date",
       title = "Publications per Day of NY Times Material Containing the Term PFAS",
       subtitle = "January 2000 to April 2022",
       caption = "As this figure shows, after 2017 there was a significant increase in publication of n
   theme(plot.title.position = "plot",
         plot.caption = element_text(hjust = 0),
         plot.caption.position = "plot")
```



As this figure shows, after 2017 there was a significant increase in publication of new articles referring to PFAS

```
# example sentence
nytDat$response.docs.snippet[9]
```

```
## [1] "Waste from a shoe factory has tainted groundwater in a Grand Rapids suburb. Some residents are
```

```
# $response refers to the 'response column' from the api results
# snippets (from the NY Times) are sentences. `snippet[9]` pulls the 9th sentence
```

```
# example paragraph
nytDat$response.docs.lead_paragraph[9]
```

```
## [1] "PLAINFIELD CHARTER TOWNSHIP, Mich. - They found pollutants in the water at the National Guard a
```

The New York Times doesn't make full text of the articles available through the API. But we can use the first paragraph of each article. The NY Times includes 33 variables (paragraph, author info,. . . ) Add a 34th column for 'word' as part of the unnesting process (start as 1 row per paragraph, then make a row per word)

```
names(nytDat)
```

```
##  [1] "status"
##  [2] "copyright"
##  [3] "response.docs.abstract"
##  [4] "response.docs.web_url"
##  [5] "response.docs.snippet"
##  [6] "response.docs.lead_paragraph"
##  [7] "response.docs.print_section"
##  [8] "response.docs.print_page"
##  [9] "response.docs.source"
## [10] "response.docs.multimedia"
## [11] "response.docs.keywords"
## [12] "response.docs.pub_date"
## [13] "response.docs.document_type"
## [14] "response.docs.news_desk"
## [15] "response.docs.section_name"
## [16] "response.docs.type_of_material"
## [17] "response.docs._id"
## [18] "response.docs.word_count"
## [19] "response.docs.uri"
## [20] "response.docs.subsection_name"
## [21] "response.docs.headline.main"
## [22] "response.docs.headline.kicker"
## [23] "response.docs.headline.content_kicker"
## [24] "response.docs.headline.print_headline"
## [25] "response.docs.headline.name"
## [26] "response.docs.headline.seo"
## [27] "response.docs.headline.sub"
## [28] "response.docs.byline.original"
## [29] "response.docs.byline.person"
## [30] "response.docs.byline.organization"
## [31] "response.meta.hits"
## [32] "response.meta.offset"
## [33] "response.meta.time"
```

### First Paragraph

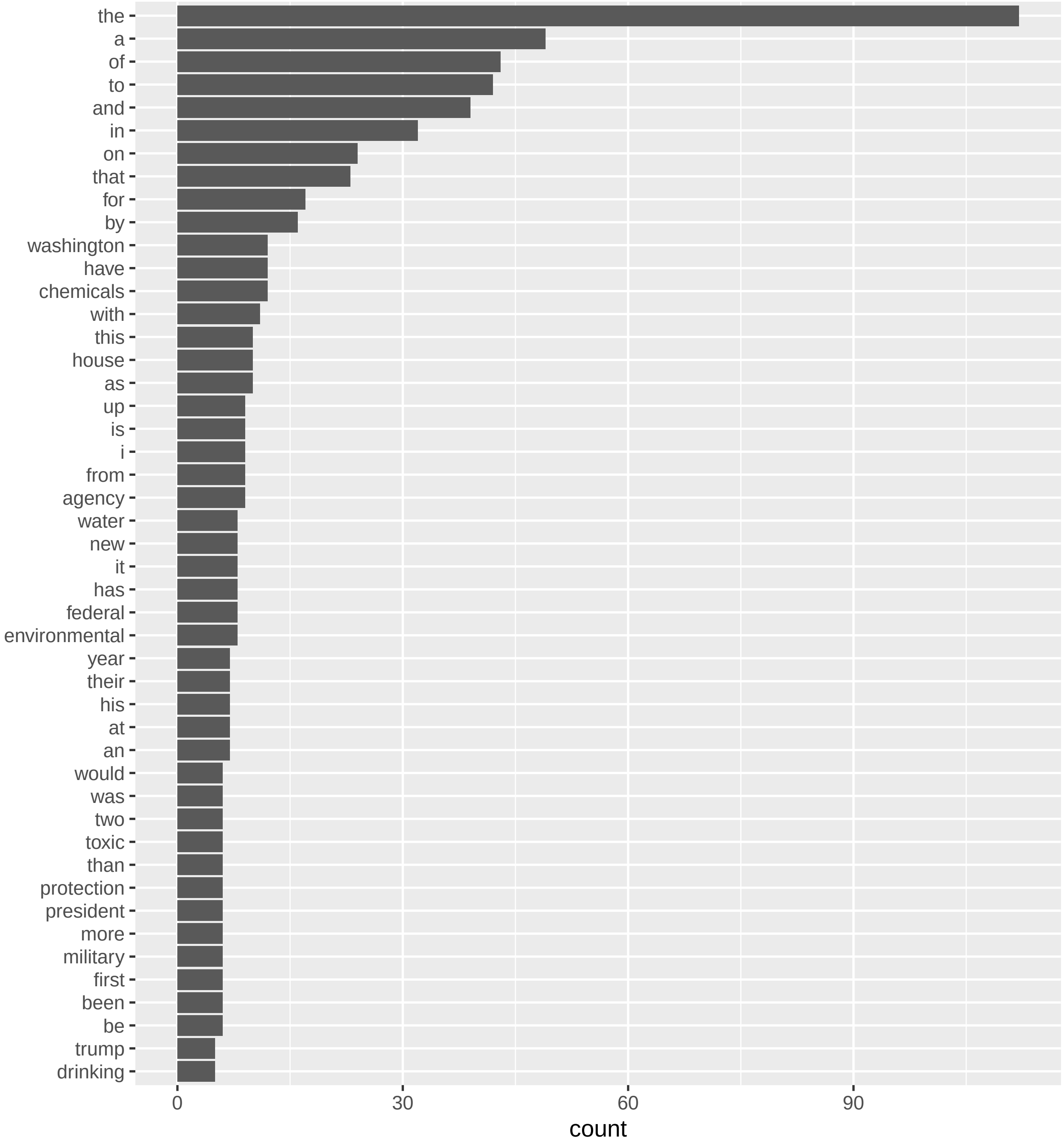
Create plots of publications per day and word frequency using the first paragraph variable

```
paragraph <- names(nytDat)[6] #The 6th column, "response.doc.lead_paragraph", is the one we want here.
tokenized_paragraph <- nytDat %>%
  unnest_tokens(word, paragraph)
```

```
names(tokenized_paragraph)
```

```
##  [1] "status"
##  [2] "copyright"
##  [3] "response.docs.abstract"
##  [4] "response.docs.web_url"
##  [5] "response.docs.snippet"
##  [6] "response.docs.lead_paragraph"
##  [7] "response.docs.print_section"
##  [8] "response.docs.print_page"
##  [9] "response.docs.source"
## [10] "response.docs.multimedia"
## [11] "response.docs.keywords"
## [12] "response.docs.pub_date"
## [13] "response.docs.document_type"
## [14] "response.docs.news_desk"
## [15] "response.docs.section_name"
## [16] "response.docs.type_of_material"
## [17] "response.docs._id"
## [18] "response.docs.word_count"
## [19] "response.docs.uri"
## [20] "response.docs.subsection_name"
## [21] "response.docs.headline.main"
## [22] "response.docs.headline.kicker"
## [23] "response.docs.headline.content_kicker"
## [24] "response.docs.headline.print_headline"
## [25] "response.docs.headline.name"
## [26] "response.docs.headline.seo"
## [27] "response.docs.headline.sub"
## [28] "response.docs.byline.original"
## [29] "response.docs.byline.person"
## [30] "response.docs.byline.organization"
## [31] "response.meta.hits"
## [32] "response.meta.offset"
## [33] "response.meta.time"
## [34] "word"
```

```r
tokenized_paragraph %>%
  count(word, sort = TRUE) %>%
  filter(n > 4) %>% #illegible with all the words displayed, consider increasing threshold to 10
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL, x = "count",
       title = "Word Frequency Plot Using First Paragraph (includes stop words)",
       subtitle = "NY Times Material Containing the Term PFAS, January 2000 to April 2022") +
   theme(plot.title.position = "plot")
```



Word Frequency Plot Using First Paragraph (includes stop words)

NY Times Material Containing the Term PFAS, January 2000 to April 2022

## Remove Stop Words

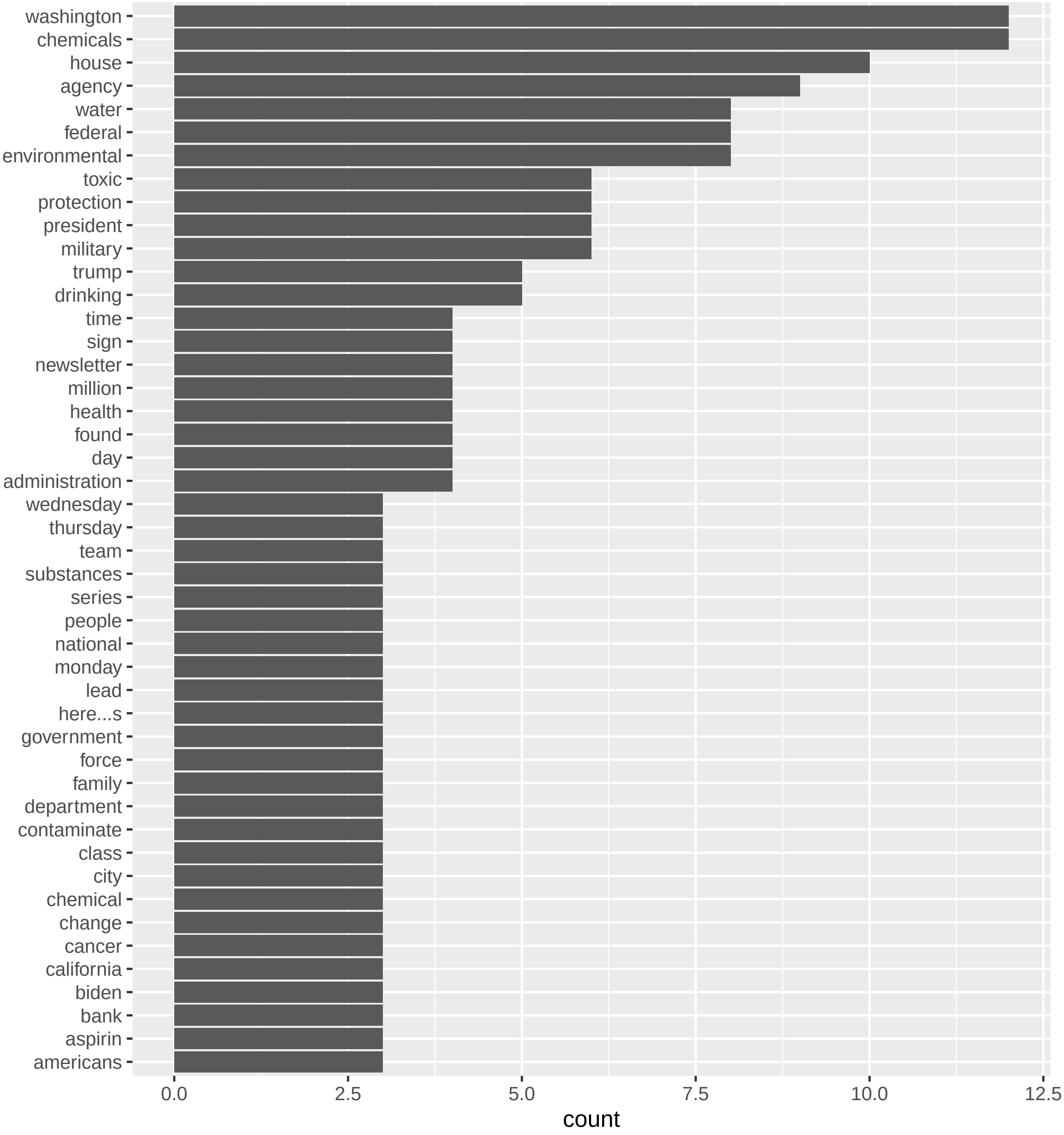
```
data(stop_words)

tokenized_paragraph_no_stop <- tokenized_paragraph %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```
tokenized_paragraph_no_stop %>%
  count(word, sort = TRUE) %>%
  filter(n > 2) %>% # adjust this based on results
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL, x = "count",
       title = "Word Frequency Plot Using First Paragraph (stop words removed)",
       subtitle = "NY Times Material Containing the Term PFAS, January 2000 to April 2022") +
   theme(plot.title.position = "plot")
```

Word Frequency Plot Using First Paragraph (stop words removed)

NY Times Material Containing the Term PFAS, January 2000 to April 2022



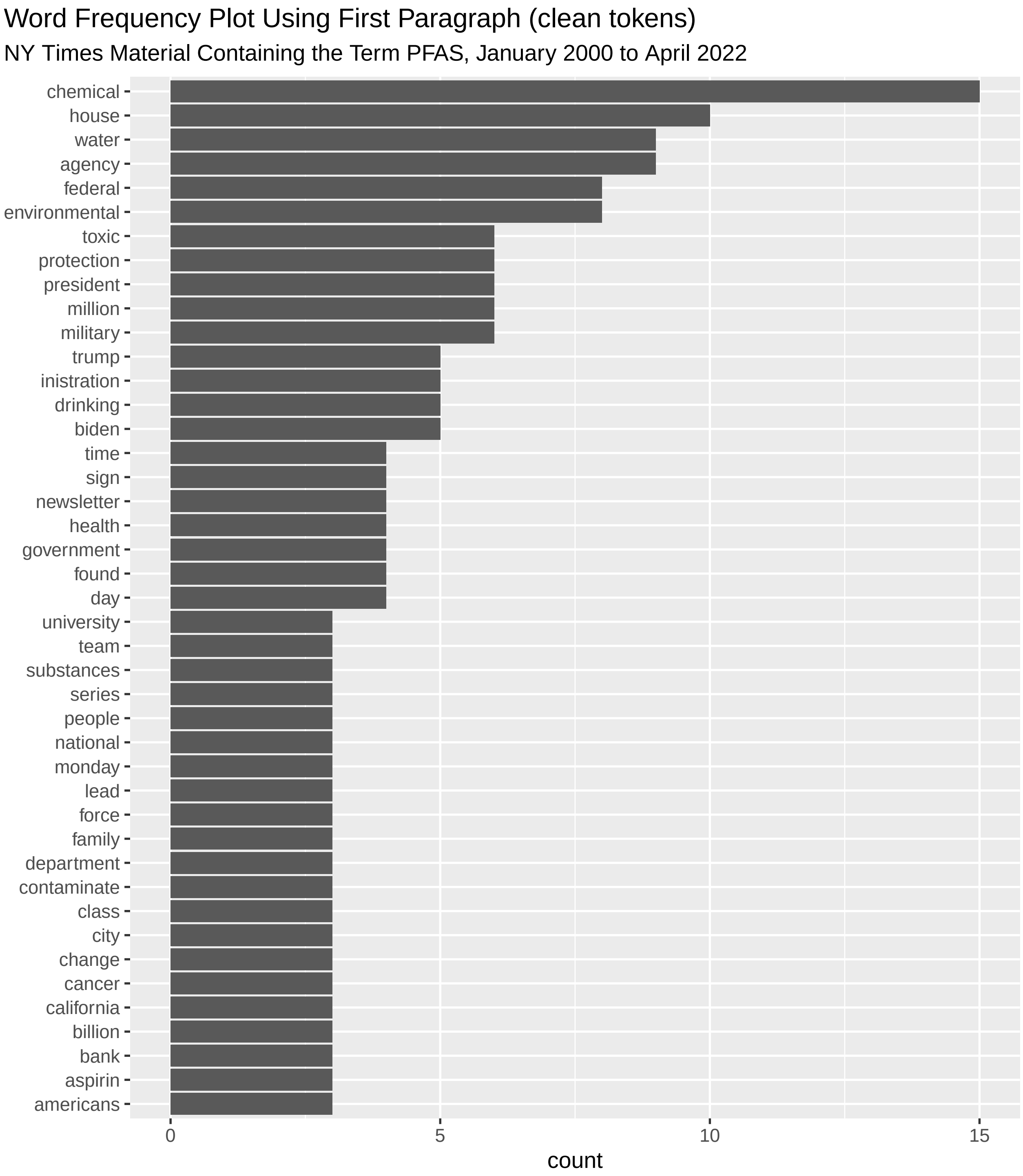
## Clean Tokens

Several steps were taken to clean tokens. Words such as ‘administration’s’ and ‘biden’s’ were cleaned to remove the ‘s. Numbers, which included single numbers and years, were removed. The word ’washington’ was removed because it was generally used to refer to the location tag at the beginning of each article.

```
clean_tokens <- str_remove_all(tokenized_paragraph_no_stop$word, "[:digit:]") #remove all numbers
clean_tokens <- gsub("’s", '', clean_tokens) # gsub is used to replace all the matches of a pattern fro
clean_tokens <- gsub(",", '', clean_tokens) # gsub is used to replace all the matches of a pattern from
clean_tokens <- gsub("chemicals", 'chemical', clean_tokens) # gsub is used to replace all the matches o
clean_tokens <- gsub("governments", 'government', clean_tokens)
clean_tokens <- gsub("’d", '', clean_tokens)
clean_tokens <- gsub("millions", 'million', clean_tokens)
clean_tokens <- gsub("billions", 'billion', clean_tokens)
clean_tokens <- gsub("residents", 'resident', clean_tokens)
clean_tokens <- gsub("waters", 'water', clean_tokens)
clean_tokens <- str_remove_all(clean_tokens, "washington") # removed 'washingon' because it referred to
clean_tokens <- str_remove_all(clean_tokens, "thursday")
clean_tokens <- str_remove_all(clean_tokens, "wednesday")
clean_tokens <- str_remove_all(clean_tokens, "here") # additional stop word
clean_tokens <- str_remove_all(clean_tokens, "a.m") # additional stop word
#clean_tokens <- str_replace_all(clean_tokens,"land[a-z,A-Z]*","land") #stem tribe words

tokenized_paragraph_no_stop$clean <- clean_tokens
```

```
#remove the empty strings
tib <-subset(tokenized_paragraph_no_stop, clean!="")

#reassign
tokenized_paragraph_clean <- tib
```

```
tokenized_paragraph_clean %>%
  count(clean, sort = TRUE) %>%
  filter(n > 2) %>% # adjust based on results
  mutate(clean = reorder(clean, n)) %>%
  ggplot(aes(n, clean)) +
  geom_col() +
  labs(y = NULL, x = "count",
       title = "Word Frequency Plot Using First Paragraph (clean tokens)",
       subtitle = "NY Times Material Containing the Term PFAS, January 2000 to April 2022") +
  theme(plot.title.position = "plot")
```

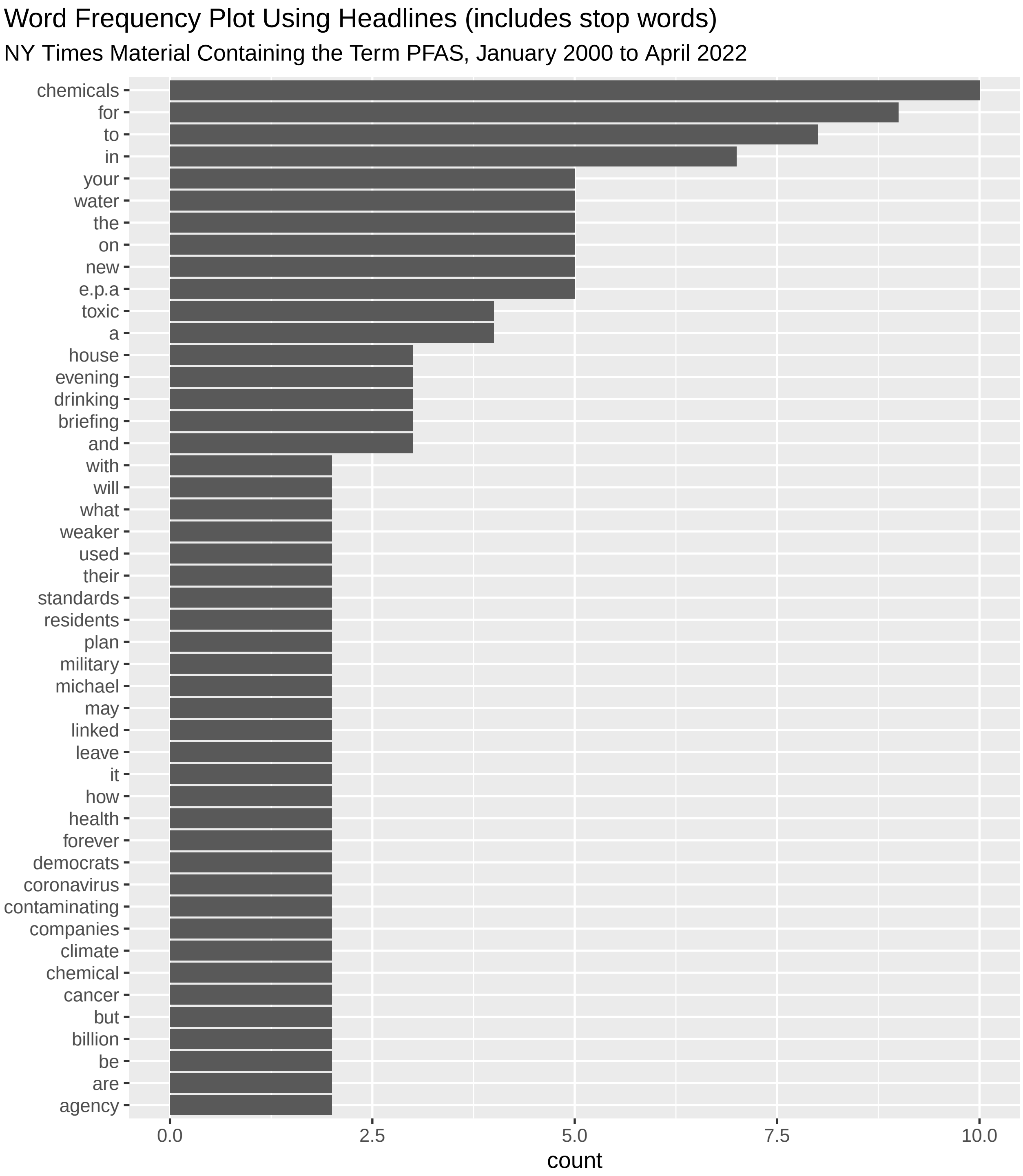


## Headlines

Create plots of publications per day and word frequency using the headline variable

```r
headlines <- names(nytDat)[21] #The 21th column, "rresponse.docs.headline.main", is the one we want her
tokenized_headlines <- nytDat %>%
  unnest_tokens(word, headlines)
```

```
tokenized_headlines %>%
  count(word, sort = TRUE) %>%
  filter(n > 1) %>% #illegible with all the words displayed, consider increasing threshold to 10
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL, x = "count",
       title = "Word Frequency Plot Using Headlines (includes stop words)",
       subtitle = "NY Times Material Containing the Term PFAS, January 2000 to April 2022") +
   theme(plot.title.position = "plot")
```


Word Frequency Plot Using Headlines (includes stop words)

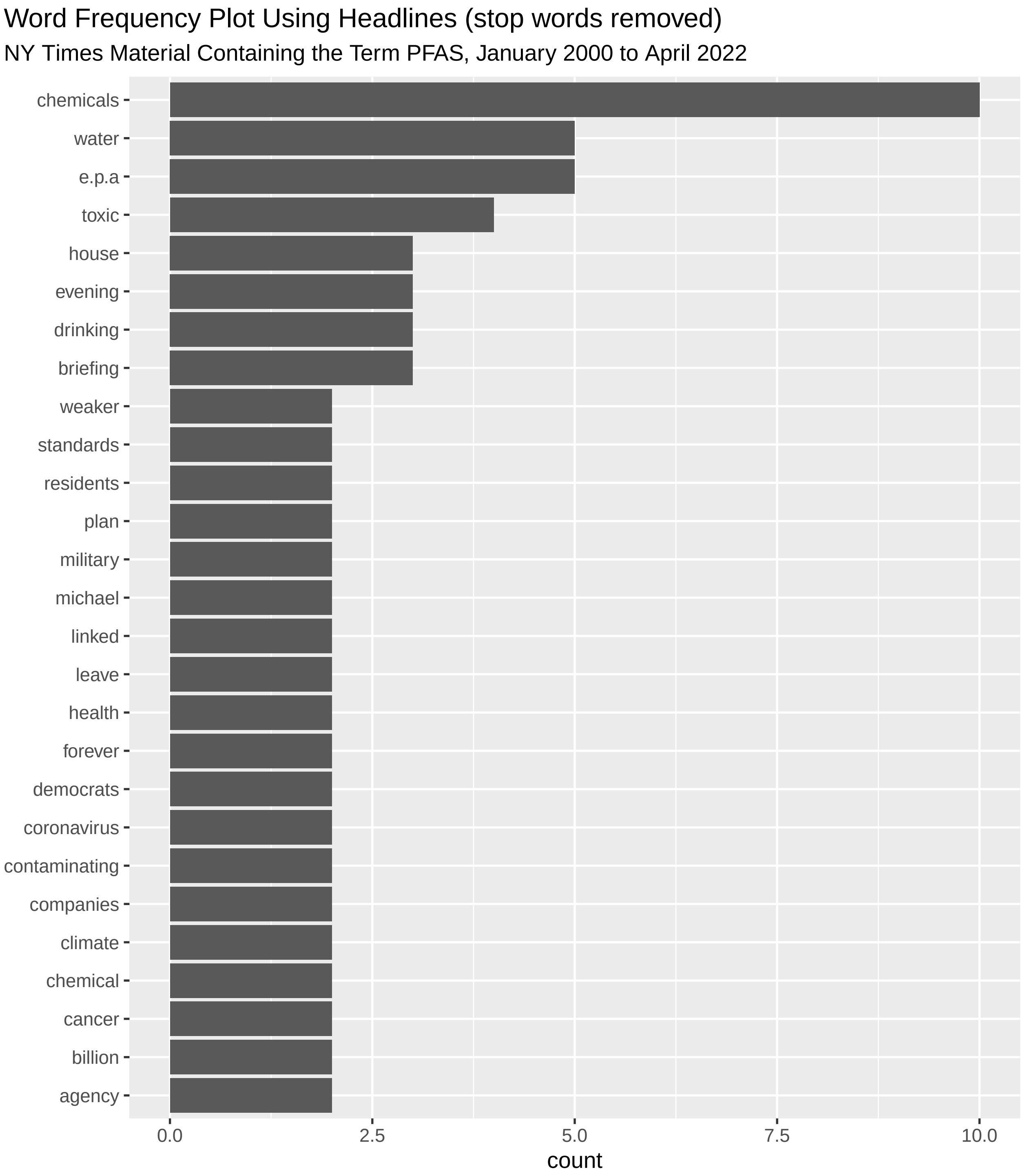
NY Times Material Containing the Term PFAS, January 2000 to April 2022

### Remove Stop Words

```
tokenized_headlines_no_stop <- tokenized_headlines %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```
tokenized_headlines_no_stop %>%
  count(word, sort = TRUE) %>%
  filter(n > 1) %>% # adjust this based on results
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL, x = "count",
       title = "Word Frequency Plot Using Headlines (stop words removed)",
       subtitle = "NY Times Material Containing the Term PFAS, January 2000 to April 2022") +
   theme(plot.title.position = "plot")
```


Word Frequency Plot Using Headlines (stop words removed)

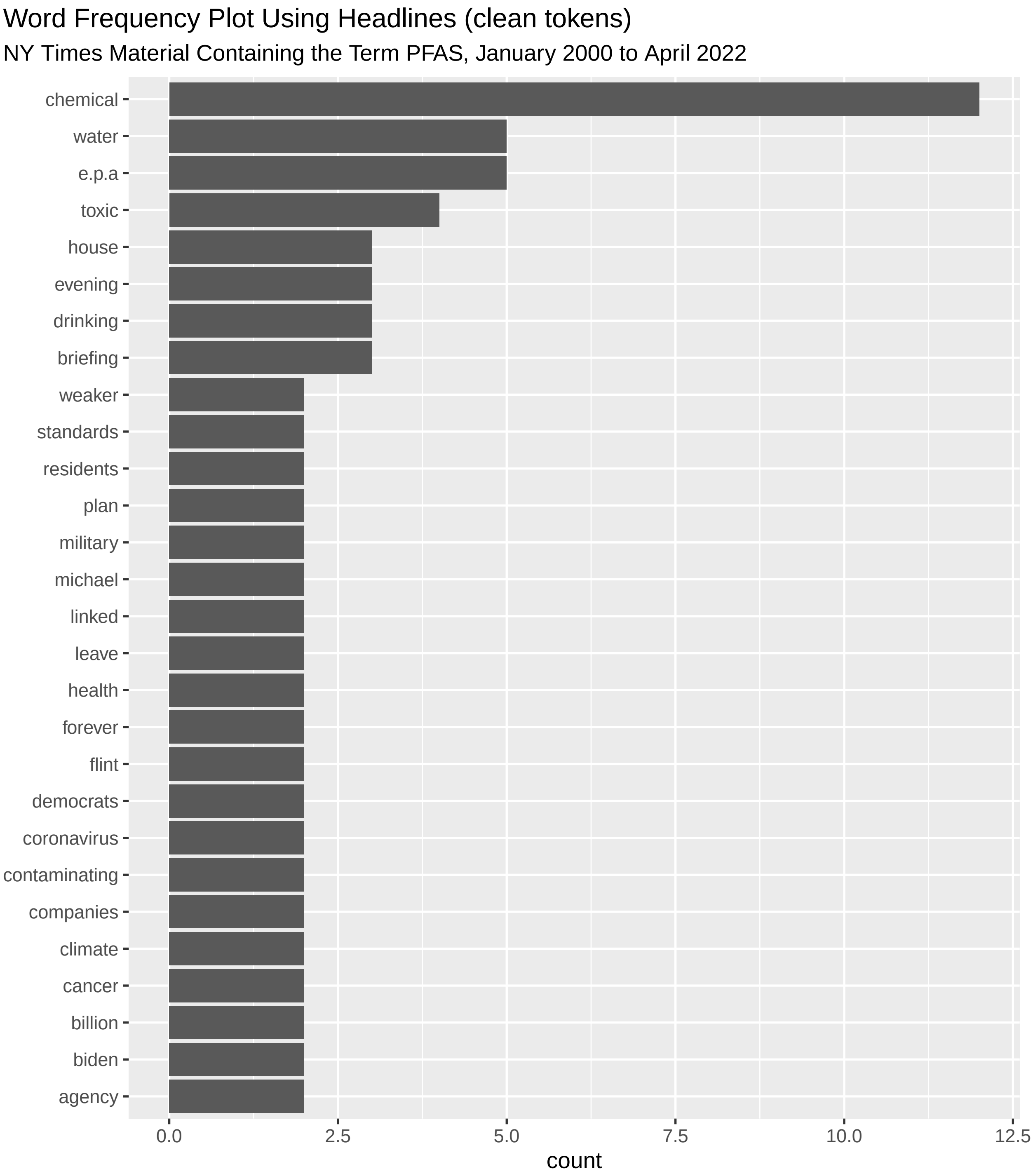
NY Times Material Containing the Term PFAS, January 2000 to April 2022

## Clean Tokens

```
clean_tokens <- str_remove_all(tokenized_headlines_no_stop$word, "[:digit:]") #remove all numbers
clean_tokens <- gsub("’s", '', clean_tokens) # gsub is used to replace all the matches of a pattern fro
clean_tokens <- gsub(",", '', clean_tokens) # gsub is used to replace all the matches of a pattern from
clean_tokens <- gsub("chemicals", 'chemical', clean_tokens) # gsub is used to replace all the matches o
clean_tokens <- str_remove_all(clean_tokens, "here") # additional stop word
clean_tokens <- str_remove_all(clean_tokens, "isn't") # additional stop word
clean_tokens <- str_remove_all(clean_tokens, "won't") # additional stop word

tokenized_headlines_no_stop$clean <- clean_tokens
```

```
#remove the empty strings
tib <-subset(tokenized_headlines_no_stop, clean!="")

#reassign
tokenized_headlines_clean<- tib
```

```
tokenized_headlines_clean %>%
  count(clean, sort = TRUE) %>%
  filter(n > 1) %>% # adjust based on results
  mutate(clean = reorder(clean, n)) %>%
  ggplot(aes(n, clean)) +
  geom_col() +
  labs(y = NULL, x = "count",
       title = "Word Frequency Plot Using Headlines (clean tokens)",
       subtitle = "NY Times Material Containing the Term PFAS, January 2000 to April 2022") +
   theme(plot.title.position = "plot")
```



## Summary

```
pfas_first_paragraph <- tokenized_paragraph_clean %>%
  count(clean) %>%
  filter(clean == "pfas") %>%
  select(n) %>%
  as.numeric()
```

```
pfas_headlines <- tokenized_headlines_clean %>%
  count(clean) %>%
  filter(clean == "pfas") %>%
  select(n) %>%
  as.numeric()
```

```
top_words_paragraph <- tokenized_paragraph_clean %>%
  count(clean, sort = TRUE) %>%
  head(28) %>%
  select(clean) %>%
  rename(top_words_paragraph = clean)

top_words_headlines <- tokenized_headlines_clean %>%
  count(clean, sort = TRUE) %>%
  filter(n > 1) %>%
  select(clean) %>%
  rename(top_words_headlines = clean)

top_words <- data.frame(top_words_paragraph, top_words_headlines)
```

```
top_words_table <- top_words %>%
  kable(col.names = c("Top Words - First Paragraph", "Top Words - Headlines")) %>%
  kable_paper(full_width = FALSE)
top_words_table
```

```
# identify common words in the list of top words from the first paragraph and headlines
common_words <- as.data.frame(intersect(top_words$top_words_paragraph, top_words$top_words_headlines))

common_words_table <- common_words %>%


  kable(col.names = "Top Words Common to First Paragraph and Headlines") %>%
  kable_paper(full_width = FALSE)
common_words_table
```

```
unique_words_paragraph <- as.data.frame(setdiff(top_words$top_words_paragraph, top_words$top_words_headl
unique_words_headlines <- as.data.frame(setdiff(top_words$top_words_headlines, top_words$top_words_parag
```

Top words (occurring 2 or more times) found in both the first paragraph and headlines include: chemical, house, agency, water, toxic, biden, drinking, military, health, cancer, and billion. Top words unique to the first paragraph include: environmental, federal, million, president, protection, trump, administration, day, found, government, newsletter, sign, time, americans, aspirin, bank, and california. Top words unique to the headlines include: e.p.a, briefing, evening, climate, companies, contaminating, coronavirus, democrats, flint, forever, leave, linked, michael, plan, residents, standards, and weaker.

It is interesting to note that the word 'pfas' only appeared 2 times in the first paragraphs and 0 times in the headlines. Headlines tended to mention chemicals in a general sense, only specifically mentioning PFAS later in the article. Headlines also tended to hint at scary health effects from everyday products in an attempt to get readers' attention.

The plot of publications per day is the same for first paragraphs and headlines.

The table below shows how the headlines were often written to stir fear and curiosity in the article.

| Top Words - First Paragraph | Top Words - Headlines |
|---|---|
| chemical | chemical |
| house | e.p.a |
| agency | water |
| water | toxic |
| environmental | briefing |
| federal | drinking |
| military | evening |
| million | house |
| president | agency |
| protection | biden |
| toxic | billion |
| biden | cancer |
| drinking | climate |
| inistration | companies |
| trump | contaminating |
| day | coronavirus |
| found | democrats |
| government | flint |
| health | forever |
| newsletter | health |
| sign | leave |
| time | linked |
| americans | michael |
| aspirin | military |
| bank | plan |
| billion | residents |
| california | standards |
| cancer | weaker |

| Top Words Common to First Paragraph and Headlines |
|---|
| chemical |
| house |
| agency |
| water |
| military |
| toxic |
| biden |
| drinking |
| health |
| billion |
| cancer |

```
headlines_table <- nytDat %>%
  select(response.docs.headline.main) %>%
  kable(col.names = "PFAS headlines") %>%
  kable_paper(full_width = FALSE)
headlines_table
```

| PFAS headlines |
|---|
| How Chemical Companies Avoid Paying for Pollution |
| A Move to Rein In Cancer-Causing ‘Forever Chemicals’ |
| These Everyday Toxins May Be Hurting Pregnant Women and Their Babies |
| Firefighters Battle an Unseen Hazard: Their Gear Could Be Toxic |
| Government Studying Widely Used Chemicals Linked to Health Issues |
| E.P.A. Approved Toxic Chemicals for Fracking a Decade Ago, New Files Show |
| E.P.A. Will Study Limits on Cancer-Linked Chemicals. Critics Say the Plan Delays Action. |
| Pentagon Pushes for Weaker Standards on Chemicals Contaminating Drinking Water |
| Miles From Flint, Residents Turn Off Taps in New Water Crisis |
| E.P.A. Proposes Weaker Standards on Chemicals Contaminating Drinking Water |
| States Are Doing What Scott Pruitt Won’t |
| Toxic ‘Forever Chemicals’ in Drinking Water Leave Military Families Reeling |
| House Democrats Push Environmental Bills, but Victories Are Few |
| Diller Scofidio + Renfro to Design Berkeley Museum |
| My Quest for Pure Water |
| Commonly Used Chemicals Come Under New Scrutiny |
| Chemical Industry Executive Nominated to Lead Consumer Watchdog Agency |
| New Threats Put Wildfire Fighters’ Health on the Line |
| Chicago’s Big Climate Problem |
| The 3 Scariest Chemicals to Watch Out For in Your Home |
| Toxic Ghosts |
| U.S. Navy Relaxes Rules on Hair Length Amid Coronavirus Outbreak |
| House Democrats to Unveil $760 Billion Infrastructure Plan |
| 2 CENTS WORTH / Commentary: In for a (pretty) penny |
| For Some, Aspirin May Not Help Hearts |
| Michael Cohen, Robert Kraft, R. Kelly: Your Friday Evening Briefing |
| Chemicals in Your Popcorn? |
| New E.P.A. Head Says Agency Has Climate Regulations Underway |
| The Austin Bungalow Had Charm. But It ‘Needed Everything.’ |
| Naming of Foreigner for Top Job Upsets Many, Including the Charlton Brothers : England Appoints Swede to Coach Tea |
| Why ‘Biodegradable’ Isn’t What You Think |
| Doping notes |
| House Passes $738 Billion Military Bill With Space Force and Parental Leave |
| Impeachment, Coronavirus, Spring Training: Your Tuesday Evening Briefing |
| With a Center-Leaning Budget, Biden Bows to Political Reality |
| Vaccines, Haiti, Texas: Your Monday Evening Briefing |
| Maine Will Make Companies Pay for Recycling. Here’s How It Works. |
| Michael Regan, Biden’s E.P.A. Pick, Faces ‘Massive Reconstruction and Rebuilding’ |
| Senate Confirms Califf as F.D.A. Chief in Tight Vote |
| Michigan Governor’s Race Tests Flint’s Jaded Residents |