Topic 6: Topic Analysis

Felicia Cruz, Mia Forsline, Cullen Molitor, Marie Rivers

2022-05-10

```
#install packages if necessary, then load libraries
if (!require(librarian)){
  install.packages("librarian")
  library(librarian)
}

librarian::shelf(
  ggraph,
  here,
  igraph, #network plots
  kableExtra,
  ldatuning,
  LDAvis,
  lubridate,
  palettetown,
  pdftools,
  quanteda,
  quanteda.textstats,
  quanteda.textplots,
  readtext, #quanted subpackage for reading pdfs
  reshape2,
  tidytext,
  tidyverse,
  tm,
  topicmodels,
  tsne,
  widyr
  )
```

Read in the data from PDF files in the "data" directory

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

#create a readtext/dataframe of all PDF scripts
scripts_pdf <- readtext(file = here("data", "*.pdf"),
```

Table 1: Summary of Movie Script Corpus

Text	Types	Tokens	Sentences
an_inconvenient_truth.pdf	2245	10936	685
before_the_flood.pdf	2540	13634	863
dont_look_up.pdf	4620	28016	2825

```
                              docvarsfrom = c("metadata", "filenames", "filepaths"),
                              sep = "_")
```

Create a clean corpus

```
#creating an initial corpus containing our data
script_corp <- corpus(x = scripts_pdf, text_field = "text" )

#check the corpus
summary(script_corp) %>%
  knitr::kable(caption = "Summary of Movie Script Corpus")
```

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 and remove stop words

```
toks <- tokens(script_corp,
               remove_punct = TRUE,
               remove_numbers = TRUE)

# xxx...add custom stop words such as character names from "Don't Look Up"
add_stops <- c(stopwords("en"),"xxx", "yyy", "zzz")

toks1 <- tokens_select(toks, pattern = add_stops, selection = "remove")
```

Convert to a document-feature matrix (dfm)

```
dfm_comm <- dfm(toks1, tolower = TRUE)
dfm <- dfm_wordstem(dfm_comm)
dfm <- dfm_trim(dfm,
                min_docfreq = 3, #keep terms that appear in at least 3 documents
                min_termfreq = 1) #keep terms that appear at least once

# dfm %>%
  # kbl() %>%
  # kable_styling(bootstrap_options = c("striped", "hover"),
  # latex_options = "HOLD_position") #hold the table position when knitting
```

Remove rows (AKA documents) with all zeroes - otherwise the topic model won't work

```
sel_idx <- slam::row_sums(dfm) > 0
dfm <- dfm[sel_idx, ]

# dfm %>%
  # kbl() %>%
  # kable_styling(bootstrap_options = c("striped", "hover"),
  # latex_options = "HOLD_position") #hold the table position when knitting
```

Choose a number of latent topics (k) and run the Gibbs topic analysis

```
k <- 7 #k is the number of topics

topicModel_k7 <- LDA(dfm,
                     k,
                     method="Gibbs",
                     control=list(iter = 500, verbose = 25))
```

```
## K = 7; V = 546; M = 3
## Sampling 500 iterations!
## Iteration 25 ...
## Iteration 50 ...
## Iteration 75 ...
## Iteration 100 ...
## Iteration 125 ...
## Iteration 150 ...
## Iteration 175 ...
## Iteration 200 ...
## Iteration 225 ...
## Iteration 250 ...
## Iteration 275 ...
## Iteration 300 ...
## Iteration 325 ...
## Iteration 350 ...
## Iteration 375 ...
## Iteration 400 ...
```

```
## Iteration 425 ...
## Iteration 450 ...
## Iteration 475 ...
## Iteration 500 ...
## Gibbs sampling completed!
```

```
tmResult <- posterior(topicModel_k7)
#attributes(tmResult) #terms and topics

beta <- tmResult$terms   # get beta from results
#dim(beta)                # K distributions over nTerms(DTM) terms

terms(topicModel_k7, 10) #view the terms from each topic
```

```
##       Topic 1 Topic 2    Topic 3 Topic 4  Topic 5     Topic 6  Topic 7
##  [1,] "can"   "time"     "just"  "presid" "year"      "world"  "now"
##  [2,] "peopl" "happen"   "go"    "orlean" "one"       "chang"  "right"
##  [3,] "come"  "work"     "back"  "look"   "ice"       "will"   "like"
##  [4,] "see"   "last"     "look"  "time"   "warm"      "know"   "want"
##  [5,] "thing" "citi"     "us"    "know"   "earth"     "start"  "think"
##  [6,] "make"  "never"    "say"   "got"    "atmospher" "get"    "yeah"
##  [7,] "like"  "point"    "new"   "night"  "ocean"     "realli" "planet"
##  [8,] "way"   "cours"    "show"  "two"    "problem"   "believ" "need"
##  [9,] "even"  "unit"     "said"  "one"    "lot"       "differ" "know"
## [10,] "take"  "scientif" "day"   "cut"    "went"      "well"   "around"
```

Calculate metrics from the data

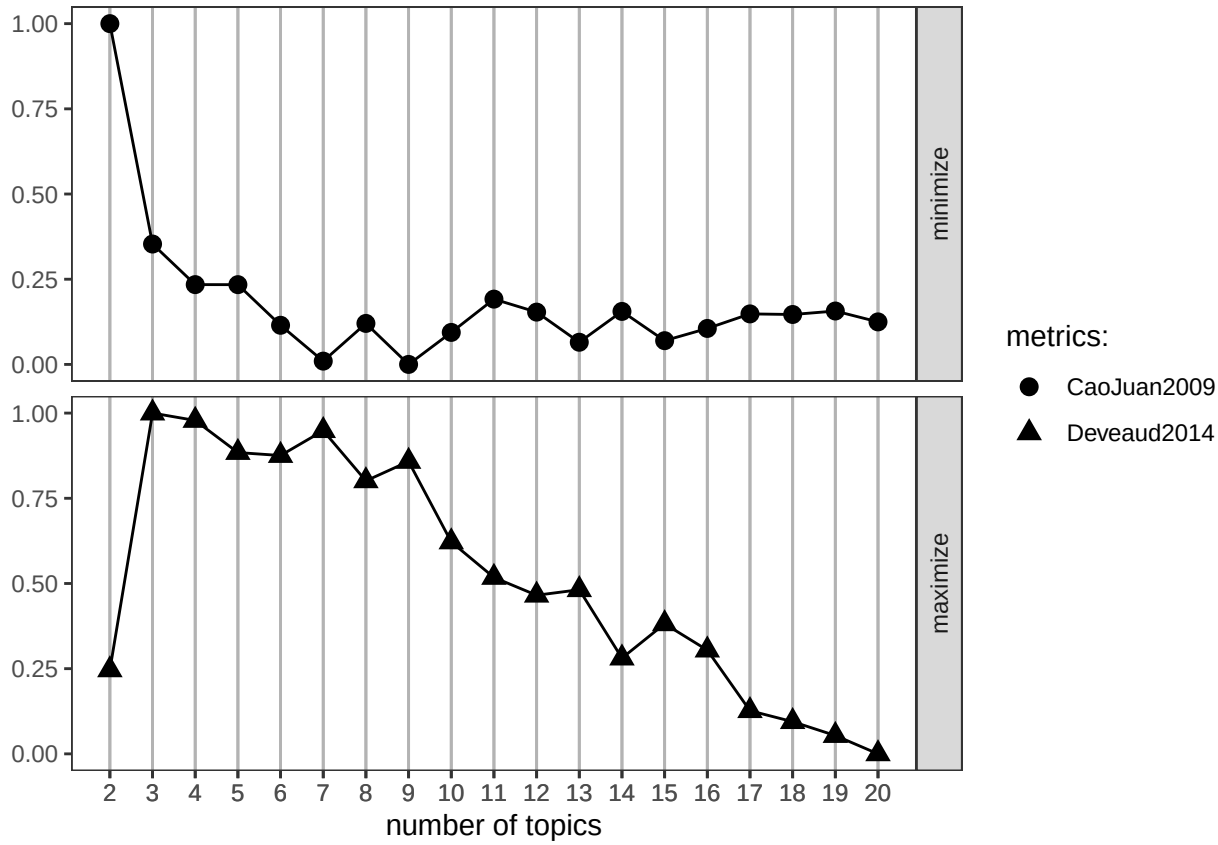
CaoJuan 2009 & Deveaud2014 method

It seems like 7 topics is the ideal number

```
result <- FindTopicsNumber(
  dfm,
  topics = seq(from = 2, to = 20, by = 1),
  metrics = c("CaoJuan2009",  "Deveaud2014"),
  method = "Gibbs",
  control = list(seed = 77),
  verbose = TRUE
)
```

```
## fit models... done.
## calculate metrics:
##   CaoJuan2009... done.
##   Deveaud2014... done.
```

```
FindTopicsNumber_plot(result)
```

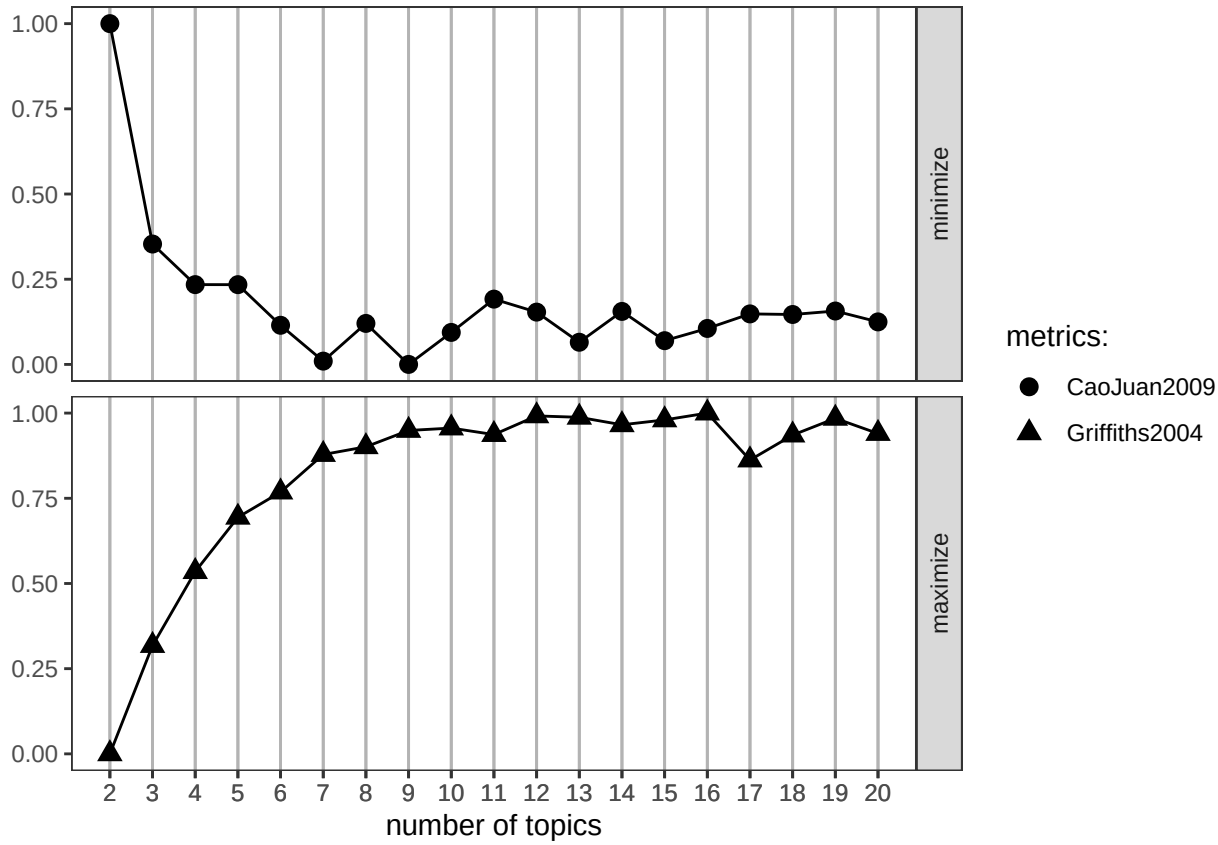


Griffiths2004 & CaoJuan2009 method

```
result <- FindTopicsNumber(
  dfm,
  topics = seq(from = 2, to = 20, by = 1),
  metrics = c("CaoJuan2009", "Griffiths2004"),
  method = "Gibbs",
  control = list(seed = 77),
  verbose = TRUE
)
```

```
## fit models... done.
## calculate metrics:
##   CaoJuan2009... done.
##   Griffiths2004... done.
```

```
FindTopicsNumber_plot(result)
```

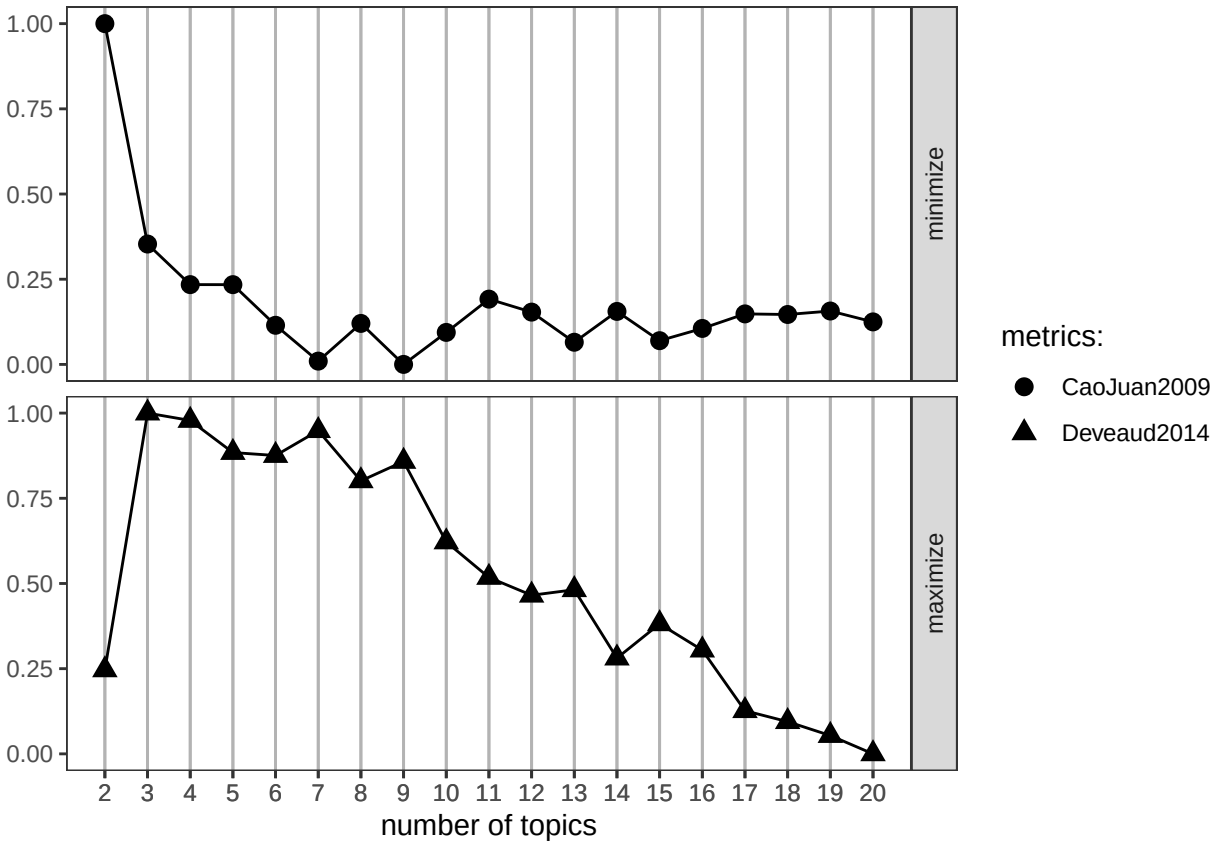


Deveaud2014 method

```
result <- FindTopicsNumber(
  dfm,
  topics = seq(from = 2, to = 20, by = 1),
  metrics = c("CaoJuan2009", "Deveaud2014"),
  method = "Gibbs",
  control = list(seed = 77),
  verbose = TRUE
)
```

```
## fit models... done.
## calculate metrics:
##   CaoJuan2009... done.
##   Deveaud2014... done.
```

```
FindTopicsNumber_plot(result)
```



Extract theta and beta values

```
theta <- tmResult$topics
beta <- tmResult$terms
vocab <- (colnames(beta))
```

Note that the beta value indicates how likely a term is to be in that particular topic

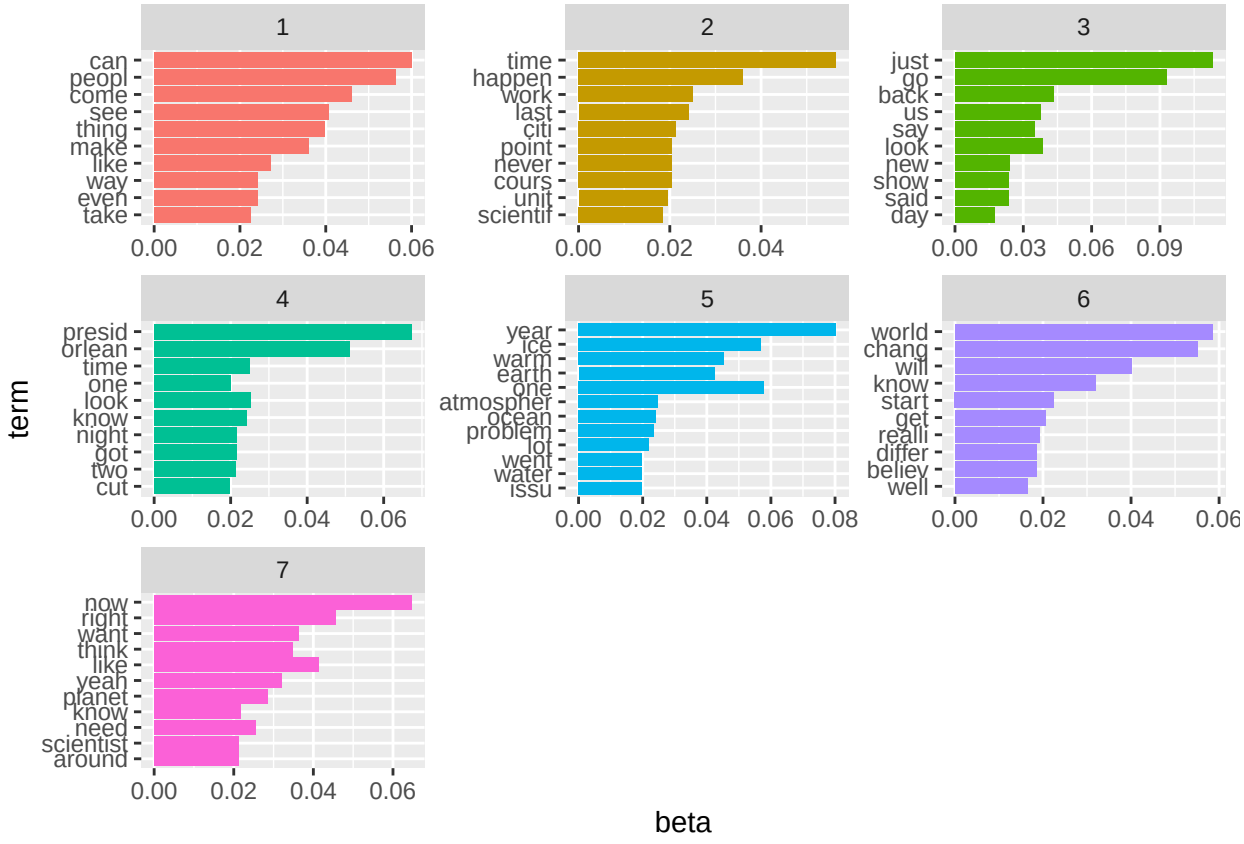
```
comment_topics <- tidy(topicModel_k7, matrix = "beta")

top_terms <- comment_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

top_terms
```

```
## # A tibble: 73 x 3
##    topic term    beta
##    <int> <chr>  <dbl>
##  1     1 can   0.0600
```

```
##  2     1 peopl 0.0564
##  3     1 come  0.0459
##  4     1 see   0.0407
##  5     1 thing 0.0397
##  6     1 make  0.0360
##  7     1 like  0.0272
##  8     1 way   0.0240
##  9     1 even  0.0240
## 10     1 take  0.0225
## # ... with 63 more rows
```

Graph the most likely words per topic

```
top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```



Assign topic names based on the first 5 words

```
top5termsPerTopic <- terms(topicModel_k7, 5)
topicNames <- apply(top5termsPerTopic, 2, paste, collapse=" ")
```

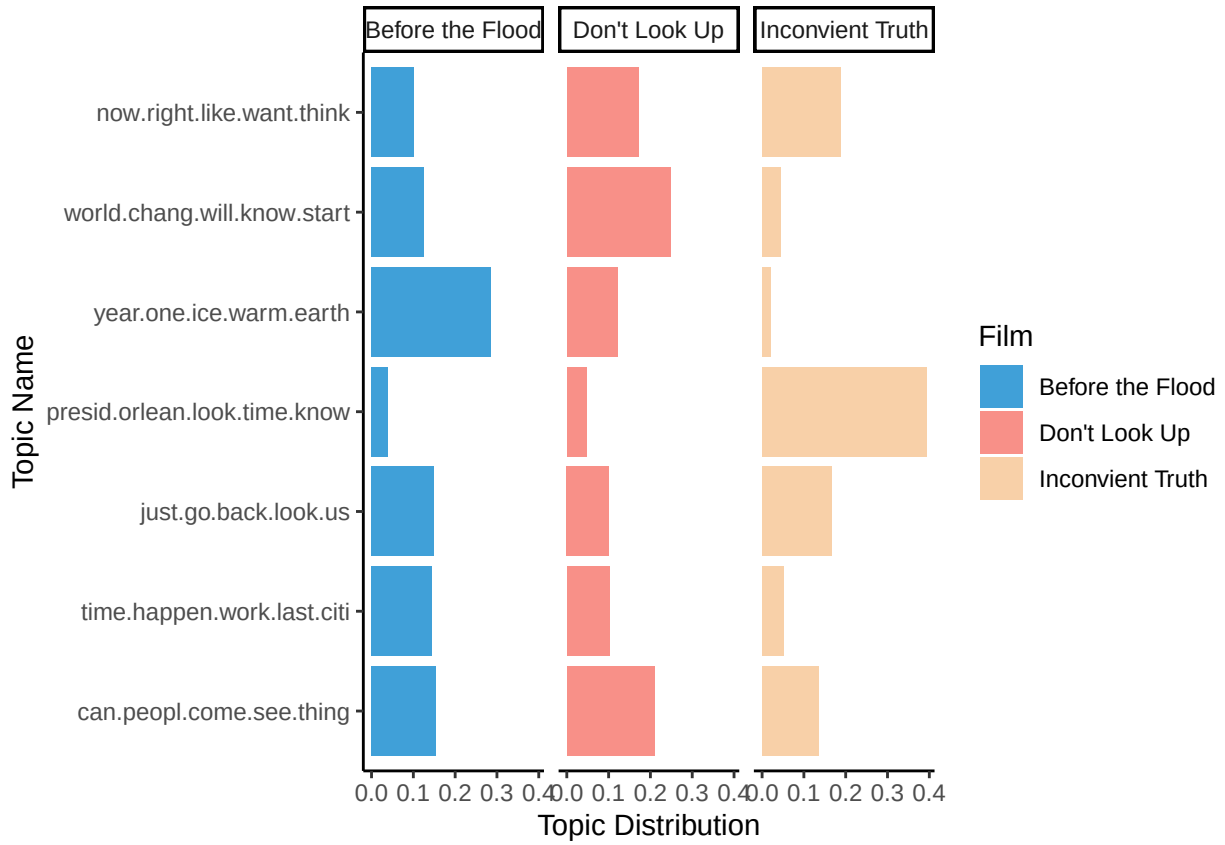
Explore the theta matrix

The theta matrix contains the distribution of each topic over each document

```
exampleIds <- c(1, 2, 3)
N <- length(exampleIds)

#lapply(epa_corp[exampleIds], as.character) #uncomment to view example text
# get topic proportions form example documents

topicProportionExamples <- theta[exampleIds,]

colnames(topicProportionExamples) <- topicNames

rownames(topicProportionExamples) <- c("Before the Flood",
                                       "Don't Look Up",
                                       "Inconvient Truth")

vizDataFrame <- melt(cbind(data.frame(topicProportionExamples),
                           document=factor(1:N)),
                           variable.name = "topic",
                           id.vars = "document")  %>%
    mutate(document = case_when(
    document == 1 ~ "Before the Flood",
    document == 2 ~ "Don't Look Up",
    document == 3 ~ "Inconvient Truth"))
```

Plot the distribution of each topic over each document

```
ggplot(data = vizDataFrame, aes(topic, value, fill = document)) +
  geom_bar(stat="identity") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  coord_flip() +
  facet_wrap(~ document, ncol = N) +
  labs(y = "Topic Distribution",
       x = "Topic Name",
       fill = "Film") +
  scale_fill_poke(pokemon = 137, spread = 3) +
  theme_classic()
```



Here's a neat JSON-based model visualizer

```
svd_tsne <- function(x) tsne(svd(x)$u)

json <- createJSON(
  phi = tmResult$terms,
  theta = tmResult$topics,
  doc.length = rowSums(dfm),
  vocab = colnames(dfm),
  term.frequency = colSums(dfm),
  mds.method = svd_tsne,
  plot.opts = list(xlab="", ylab="")
)

serVis(json)
```