TOPIC MODELING IN R

Linking words to topics

Pavel Oleinikov
Associate Director

LDA and random numbers

- LDA call

```
mod = LDA(x=dtm, k=2,
method="Gibbs",control=list(alpha=1, delta=0.1,
 seed=10005, iter=2000, thin=1))
```

- Random search through the space of parameters

- Optimization goal - find the model with the largest log-likelihood
- Likelihood - plausibility of parameters in the model given the data

Random search

- Gibbs sampling - a type of Monte Carlo Markov Chain (MCMC) algorithm.

```
method="Gibbs"
```

- Tries different combinations of probabilities of topics in documents, and probabilities of words in topics: e.g. (0.5, 0.5) vs. (0.8, 0.2)

- The combinations are influenced by parameters alpha and delta

```
control=list(alpha=1, delta=0.1)
```

Random search - controlling the iterations

- Argument `seed` sets the starting point for the pseudo-random number generator

```
control=list(seed=10005)
```

- Ensures *replication* of results between runs
- Argument `iter` controls the number of iterations of algorithm

```
control=list(iter=1000)
```

- Default is 2000

Effect of seed value

- Same corpus of five short sentences

```
mod = LDA(x=dtm, k=2,
   method="Gibbs",
   control=list(alpha=1,
    seed=10005, thin=1))
mod@gamma
```

- Prevalence of topics in documents

```
          [,1]       [,2]
[1,] 0.1538462 0.84615385
[2,] 0.2777778 0.72222222
[3,] 0.8750000 0.12500000
[4,] 0.9230769 0.07692308
[5,] 0.5000000 0.50000000
```

- Different `seed` value

```
mod <- LDA(x=dtm, k=2,
  method="Gibbs",
  control=list(alpha=1,
  seed=678910, thin=1))
mod@gamma
```

- Similar proportions, flipped topics

```
          [,1]      [,2]
[1,] 0.6153846 0.3846154
[2,] 0.7222222 0.2777778
[3,] 0.1250000 0.8750000
[4,] 0.4615385 0.5384615
[5,] 0.3888889 0.6111111
```

Handling intermediate results

- `topicmodels` calls a piece of code written in C
- Argument `thin` specifies how often to return the result of search

```
control=list(thin=1)
```

- Setting `thin=1` will return result for every step, and the best one will be picked.
- Most efficient, but slows down the execution.

Most probable words in topics

- LDA model object contains matrix `beta` with probabilities of words in topics
  - Use function `tidy` to extract
- If we want to get top 5 words from each topic:
  - Retrieve the matrix by calling `tidy(model, matrix="beta")` and sort by probabilities, filter by row number

Using tidy() to get most probable words

```
tidy(mod, matrix="beta") %>%
  group_by(topic) %>%
  arrange(desc(beta)) %>%
  filter(row_number() <=3) %>%
  ungroup() %>%
  arrange(topic, desc(beta))
```

```
  topic term          beta
  <int> <chr>        <dbl>
1     1 the         0.0831
2     1 you         0.0831
3     1 loans       0.0695
4     2 restaurant 0.0804
5     2 will        0.0647
6     2 opened      0.0647
```

Using function terms()

- Function `terms` from `topicmodels` will return either top `k` words or all words with probability above `threshold`

```
terms(mod, k=5)

     Topic 1 Topic 2
[1,] "the"   "restaurant"
[2,] "you"   "will"
[3,] "loans" "opened"
[4,] "to"    "a"
[5,] "pay"   "new"

terms(mod, threshold=0.05)

$`Topic 1`
[1] "loans" "pay"   "the"   "to"    "you"
$`Topic 2`
[1] "will"       "opened"     "restaurant"
```

TOPIC MODELING IN R

Time to practice

TOPIC MODELING IN R

Manipulating the vocabulary

Pavel Oleinikov
Associate Director
Quantitative Analysis Center
Wesleyan University

Possible operations

Two situations:

1. Knowing what words we don't want
2. Knowing what words we do want

Similar actions, differ based on how much we know:

1. removing stop words
2. keeping needed words

Removing stopwords

- What are stopwords?
  - Service words that are considered as noise and must be removed
- They obscure word associations in topics
- Example from previous lesson:

```
topic term          beta
<int> <chr>        <dbl>
1      1 will       0.0928
2      1 opened     0.0928
3      1 restaurant 0.0928
4      2 the        0.153
5      2 you        0.153
6      2 to         0.123
```

Using anti_join()

- `inner_join` in `dplyr` keeps the rows that matched in both tables
- `anti_join` drops the rows matched in both tables
- `tidytext` comes with a table `stop_words` containing stop words from several lexicons

```
d = data.frame(term=c("we", "went", "fishing", "slept"),
               count=c(2, 1, 3, 1),
               stringsAsFactors = F)
d %>% anti_join(stop_words, by=c("term"="word"))
```

```
     term count
1 fishing     3
2   slept     1
```

Keeping the needed words in

- `inner_join` offers a way to keep the needed words in the corpus.
  - Some literature scholars prefer to keep only nouns.
  - We will later keep only verbs.
- Example of making a dtm with vocabulary of two words:

```
d = data.frame(term=c("we", "went", "fishing", "slept"),
               count=c(2, 1, 3, 1),
               stringsAsFactors = F)
dictionary = data.frame(term=c("fishing", "slept"), stringsAsFactors = F)
d %>% inner_join(dictionary, by="term")

     term count
1 fishing     1
2   slept     1
```

TOPIC MODELING IN R

Time to practice

TOPIC MODELING IN R

Word clouds

Pavel Oleinikov
Associate Director
Quantitative Analysis Center
Wesleyan University

Word clouds

- Bar plots do not look good when the number of words is large
- `wordcloud` will draw a cloud of text labels, with font size proportionate to the frequency of the word
- Required arguments - a vector of words, and the vector of word frequencies
- No need to sort the words by frequency
- Package `wordcloud`

Top 20 words

- Count the frequencies over the whole corpus

```
word_frequencies <- corpus %>%
    unnest_tokens(input=text, output=word) %>%
    count(word)
```

- In a call to `wordcloud`:
  - Specify number of words shown `max.words`
  - Specify the range of word frequencies, `min.freq` and `max.freq`

```
library(wordcloud)

wordcloud(words=word_frequencies$word, freq=word_frequencies$n,
            min.freq=1, max.words=20)
```


you
face
if
to
loans
in
want
bank
on
fines
pay
the
due
new
warwick
off
a
opened
will
restaurant

Adding color and rotations

- Two more arguments to control appearance
- `colors` takes a vector of colors.
- `rot.per` is percentage of rotated words. Default is 0.1

```
word_frequencies <- corpus %>%
    unnest_tokens(input=text, output=word) %>%
    count(word)
wordcloud(words=word_frequencies$word, freq=word_frequencies$n,
            min.freq=1,
    colors=c("DarkOrange", "CornflowerBlue", "DarkRed"),
    rot.per=0.3,
    max.words=20)
```


you
the
to
loans
street
will
pay
in
fines
there
need
is
a
new
bank
bad
are
off
opened
restaurant

Wordclouds with results of LDA

- `wordcloud` expects integer values for word frequencies
- `LDA` returns probabilities - decimal fractions
- Solution: multiply by a large number, truncate the fractional part

```
# Fit a topic model with k=2
mod <- LDA(x=dtm, k=2,
           method="Gibbs",
           control=list(alpha=1, thin=1, seed=10005))

# Multiply probabilities by 10000
word_frequencies <- tidy(mod, matrix="beta") %>%
      mutate(n = trunc(beta * 10000)) %>%
      filter(topic == 1)

## display word cloud
wordcloud(words=word_frequencies$term,
          freq=word_frequencies$n,
          max.words=20,
          colors=c("DarkOrange", "CornflowerBlue", "DarkRed"),
          rot.per=0.3)
```


opened
restaurant
will
a
late
new
agreed
in
for
there
that
on
downtown
need
is
off
face
want
just
street

TOPIC MODELING IN R

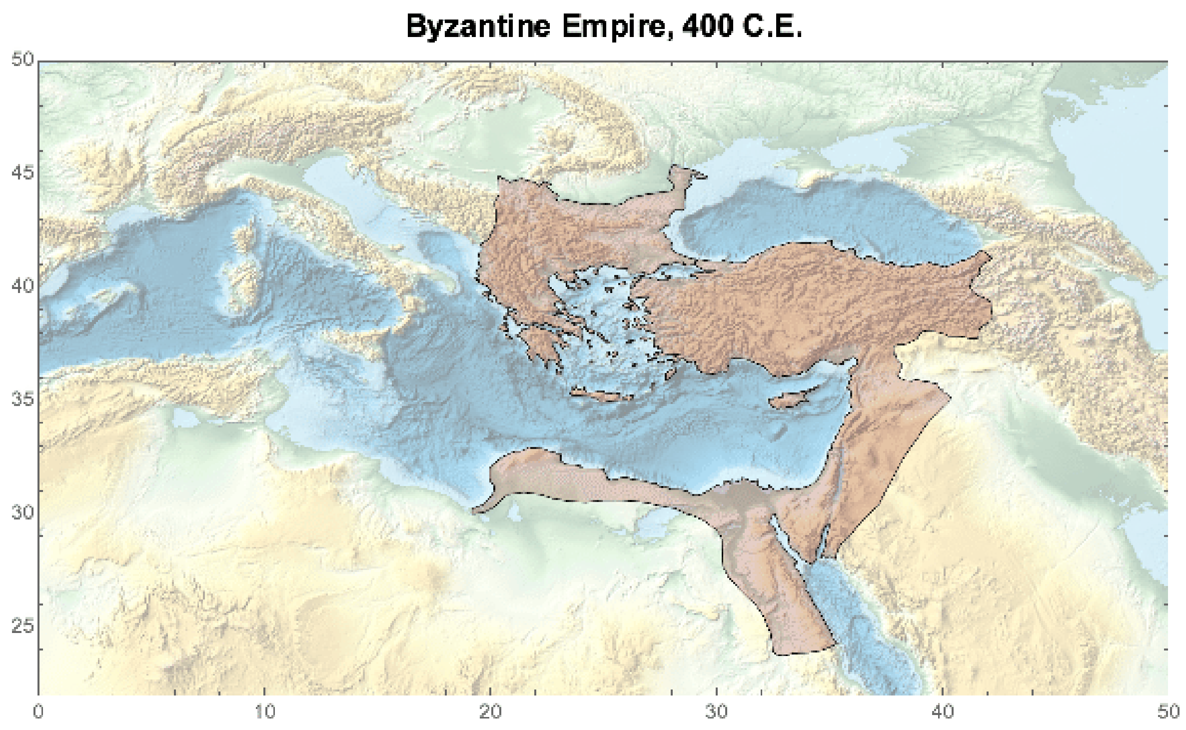
Let's practice

TOPIC MODELING IN R

History of the Byzantine Empire

Pavel Oleinikov
Associate Director
Quantitative Analysis Center
Wesleyan University

Byzantine Empire



- Byzantine Empire - East Roman empire
  - Founded in 330 C.E.
  - Fell in 1453 C.E.
  - Capital in Constantinople (Istanbul)
  - The "second Rome"

The text

- The text: *The Byzantine Empire*, by Charles Oman, printed in 1902, available from Project Guttenberg (https://www.gutenberg.org/)
  - Twenty six chapters arranged in chronological order
- Package `gutenbergr` enables direct download of texts
  - Dataframe with lines of text
- Dataframe `history` with two columns: `text` and `chapter`

The Plan

- Fit a topic model, find the predominant themes in specific periods.
  - Prepare a document-term matrix
  - Fit a simple model (four topics).
  - Examine the topics. Repeat text pre-processing and re-run the model, if necessary.
  - Visualize with ggplot.
- Compare topics with outside knowledge

TOPIC MODELING IN R

Let's jump in.