Quantitative Text Analysis Overview and Example

This document is a brief introduction to Quantitative Text Analysis (QTA) methods in R. Doing QTA in R generally follows the steps below:

1. Find or Generate Initial Dataset – this often entails using web or document scraping techniques, including working with Optical Character Recognition technologies. In this example, I am using a dataset provided to me by Andy Eggers of obituaries of former Members of Parliament. Note in the example that you want to tell R not to read the text strings as factor variables:

```
parli = read.csv("http://bit.ly/2Flquph", stringsAsFactors = F)
str(parli)
```

'data.frame': 11485 obs. of 8 variables: $ election_id : int 35779 35779 35779 35779 35780 35780 35780 35781 35781 35781 ... $ date : chr "1950-02-23" "1950-02-23" "1950-02-23" "1950-02-23" ... $ constituency.name: chr "Battersea North" "Battersea North" "Battersea North" "Battersea North" ... $ sname : chr "Jay" "Maddan" "Handscombe" "Mahon" ... $ party : chr "Lab." "C" "L." "Comm." ... $ votes : int 24762 9084 1090 655 16142 15774 2949 26018 5964 1852 ... $ winner : int 1 0 0 0 1 0 0 1 0 0 ... $ bio : chr "Mr. Douglas Jay, who was appointed Economic Secretary to the Treasury in December, 1947, was born in 1907, and"| **truncated** "Mr. M. Madden, aged 29, was educated at Fettes and Brasenose College, Oxford, where he read politics, philosoph"| **truncated** "Mr. E. R. Handscombe is a Fulham builder who began life as a barrister's clerk. Aged 48, he went to St. Martin"'| **truncated** "Mr. J. Mahon, a member of the Communist Party since 1921, is the author of "British Trade Unionism," and a nu"| **truncated** ...

2. Getting Your Dataset into Workable Condition – this entails loading your dataset into R and then pre-processing it – finding and removing artifacts or typos, and sometimes finding and rooming text that might distort your findings (most commonly, articles like 'the' and 'a' or other such language). For example, Snowball has a list of common english stopwords, which you could edit out of a dataset using a package like grepl:

```
library(SnowballC)
library(tm)
sw=stopwords('english')
sample(sw,10)
```

```
##  [1] "into"       "out"        "about"      "themselves" "ours"
##  [6] "they'll"    "theirs"     "he'll"      "isn't"      "by"
```

Some QTA packages will have you generate a corpora or another such specialized data object. Below we generate a Corpora and then pre-process it to remove everything except relevant text.

```
library(stm)
library(slam)
library(servr)
library(tm)
textpar=parli$bio
CorpusTM=Corpus(VectorSource(textpar))
CorpusTM=tm_map(CorpusTM,removePunctuation)
CorpusTM=tm_map(CorpusTM,removeNumbers)
CorpusTM=tm_map(CorpusTM,tolower)
CorpusTM=tm_map(CorpusTM,removeWords,stopwords('english'))
CorpusTM=tm_map(CorpusTM,stripWhitespace)
CorpusTM=tm_map(CorpusTM,stemDocument)
TDM=TermDocumentMatrix(CorpusTM)
```

3. Analyzing Your Dataset – Once you have loaded and processed your data, you can begin to analyze it.

Your dataset will generally consists of columns of variables, some of which will be the text strings which interest you. Various packages can also manipulate corpora to create digital objects (like a matrix of terms appearing in the documents) that can themselves be analyzed. We can explore frequency of terms.

```
sample(findFreqTerms(TDM,lowfreq=4,highfreq=10),20)
```

```
##  [1] "stevedor"  "pavilion"  "toy"       "braintre"  "reynold"
##  [6] "elain"     "sinn"      "eventu"    "ridg"      "mate"
## [11] "lucton"    "crieff"    "porter"    "hockey"    "biolog"
## [16] "nixon"     "carterjon" "plant"     "hals"      "phd"
```

But more importantly, we can build topic models and other analytical tools that explore what words are co-located. You can use more sophisticated models that also incorporate variables. Below, we put our corpus into the slm format and then run a function that generates a Structural Topic Model. STMs will take a document-term matrix and an arbitrary number of topics (K below) and return K number of topics fitted to the parameters.

```
TDMS=removeSparseTerms(TDM,0.99)
out= readCorpus(TDMS, type="slam")
names = paste(parli$constituency.name,parli$sname)
out=prepDocuments(out$documents, out$vocab, names)
summary(out)
```

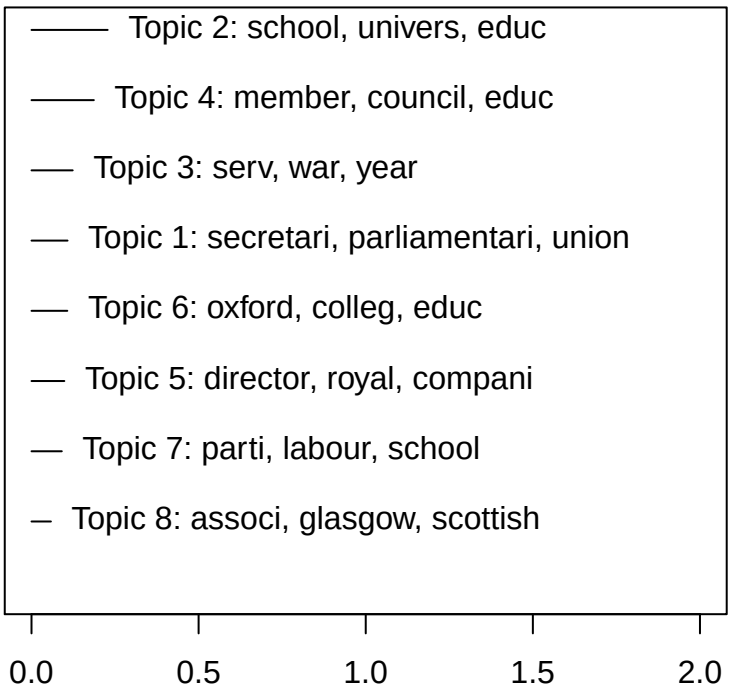
```
##                Length Class  Mode
## documents      11464  -none- list
## vocab            381  -none- character
## meta           11485  -none- character
## words.removed      0  -none- character
## docs.removed       0  -none- NULL
## tokens.removed     1  -none- numeric
## wordcounts       381  -none- numeric
```

```
TMResult=stm(out$documents, out$vocab, K=8, max.em.its = 75, data = out$meta)
```

```
## Beginning Spectral Initialization
##   Calculating the gram matrix...
##   Finding anchor words...
##      ........
##   Recovering initialization...
##      ...
## Initialization complete.
## ..................................................................................................
## Completed E-Step (3 seconds).
## Completed M-Step.
## Completing Iteration 1 (approx. per word bound = -5.236)
## ..................................................................................................
## Completed E-Step (3 seconds).
## Completed M-Step.
## Completing Iteration 2 (approx. per word bound = -5.178, relative change = 1.105e-02)
## ..................................................................................................
## Completed E-Step (2 seconds).
## Completed M-Step.
## Completing Iteration 3 (approx. per word bound = -5.153, relative change = 4.906e-03)
## ..................................................................................................
## Completed E-Step (2 seconds).
## Completed M-Step.
```

```
## Completing Iteration 4 (approx. per word bound = -5.140, relative change = 2.514e-03)
## ..........................................................................................
## Completed E-Step (2 seconds).
## Completed M-Step.
## Completing Iteration 5 (approx. per word bound = -5.133, relative change = 1.322e-03)
## Topic 1: secretari, parliamentari, nation, union, minist
##  Topic 2: univers, school, educ, colleg, born
##  Topic 3: war, serv, year, armi, educ
##  Topic 4: council, member, school, educ, born
##  Topic 5: royal, colleg, born, educ, director
##  Topic 6: oxford, colleg, educ, born, barrist
##  Topic 7: labour, parti, school, educ, born
##  Topic 8: glasgow, associ, scottish, edinburgh, born
## ..........................................................................................
## Completed E-Step (2 seconds).
## Completed M-Step.
## Completing Iteration 6 (approx. per word bound = -5.130, relative change = 7.017e-04)
## ..........................................................................................
## Completed E-Step (2 seconds).
## Completed M-Step.
## Completing Iteration 7 (approx. per word bound = -5.128, relative change = 3.673e-04)
## ..........................................................................................
## Completed E-Step (2 seconds).
## Completed M-Step.
## Completing Iteration 8 (approx. per word bound = -5.127, relative change = 1.836e-04)
## ..........................................................................................
## Completed E-Step (2 seconds).
## Completed M-Step.
## Completing Iteration 9 (approx. per word bound = -5.126, relative change = 7.568e-05)
## ..........................................................................................
## Completed E-Step (2 seconds).
## Completed M-Step.
## Model Converged
```

```
plot.STM(TMResult, type="summary", xlim=c(0,2))
```



We can then do all kinds of interesting things, such as modeling these topics' relationships with each other. Below is a static graph depicting this, but R also supports software that does really beautiful interactive modelling (but that won't work in a pdf).

```
mod.out.corr=topicCorr(TMResult)
library(igraph)
plot.topicCorr(mod.out.corr)
```