Structural Topic Models

Bamberg Summer Institute in Computational Social Science

Carsten Schwemmer, University of Bamberg

2019-08-01

Many thanks to Brandon Stewart for providing material for this lecture

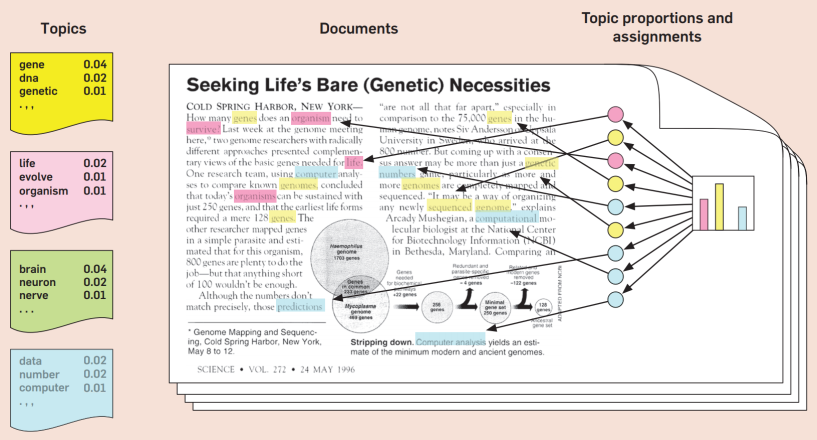
Introducing topic models

Vanilla topic modeling - latent dirichlet allocation (LDA)

> *"Topic models are algorithms for discovering the main themes that pervade a large and otherwise unstructured collection of documents. Topic models can organize the collection according to the discovered themes." (Blei 2012)*

- LDA is a mixed membership model and the name is based upon the dirichlet distribution
- idea: texts are generated by latent topics.
- each document is a mixture over topics. Each topic is a mixture over words
- we have to fix the number of topics ex ante

LDA - A statistical highlighter



Topics

gene	0.04
dna	0.02
genetic	0.01
. , ,	

life	0.02
evolve	0.01
organism	0.01
. , ,	

brain	0.04
neuron	0.02
nerve	0.01
. . .	

data	0.02
number	0.02
computer	0.01
. , ,	

Documents

Topic proportions and assignments

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

Haemophilus genome 1703 genes
Genes in common 233 genes
Mycoplasma genome 469 genes
Genes needed for biochemical pathways +22 genes
256 genes
Redundant and parasite-specific genes removed −4 genes
Minimal gene set 250 genes
Related and modern genes removed −122 genes
128 genes
Ancestral gene set
ADAPTED FROM NCBI

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

Blei (2012). Probabilistic Graphical Models.

LDA - disadvantages

- the initialization of LDA models is not deterministic: starting parameters are initially set to random values. This affects model stability and introduces repllication problems
- good implementations of LDA topic models often require dependencies beyond R (e.g. java for mallet)
- LDA is “blind” to context information. It only models term co-occurence without considering meta data

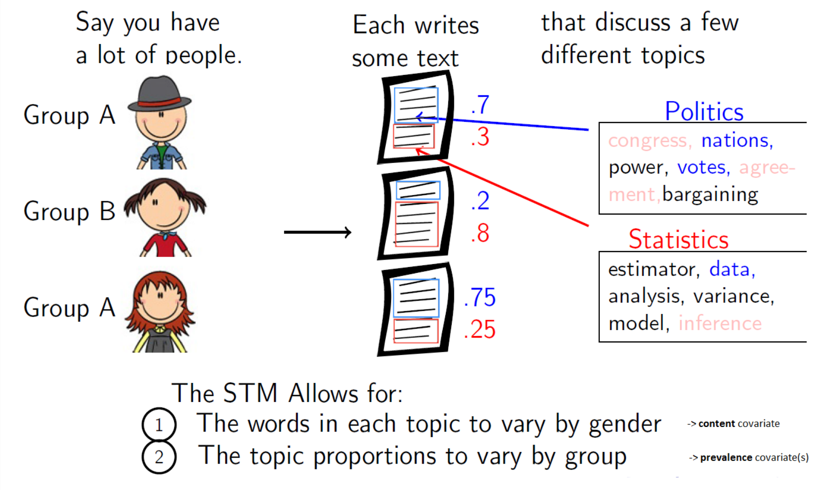
Topic modeling - extensions

correlated topic models, dynamic topic models, hierarchical LDA, pachinko allocation, nonparametric pachinko allocation,factorial LDA, gamma-poisson factorization, shared component topic models, dirichlet multinomial regresson topic models, expressed agenda model, structured topic model, nested hierarchical dirichlet process topic model, focused topic model, inverse regression topic model, ideal point topic model, discrete innite logistic normal topic model multilingual topic model, markov topic model, relational topic model, syntactic topic model, supervised latent dirichlet allocation, ...

Introducing the Structural Topic Model (STM)

- Text documents often include contextual information, e.g. time stamps or author attributes
- A huge variety of social science questions is motivated by connecting metadata with textual data
- STM provides two ways to include contextual information:
  - Topic prevelance can vary by metadata (e.g. Democrats talk more about education than Republicans)
  - Topic content can vary by metadata (e.g. Democrats are less likely to use the word life when talking about abortion than Republicans)
- you can find many publications and auxiliary packages for STM on `structuraltopicmodel.com`

STM - what this means in pictures



Say you have a lot of people.

Each writes some text

that discuss a few different topics

Politics

congress, nations, power, votes, agreement, bargaining

Statistics

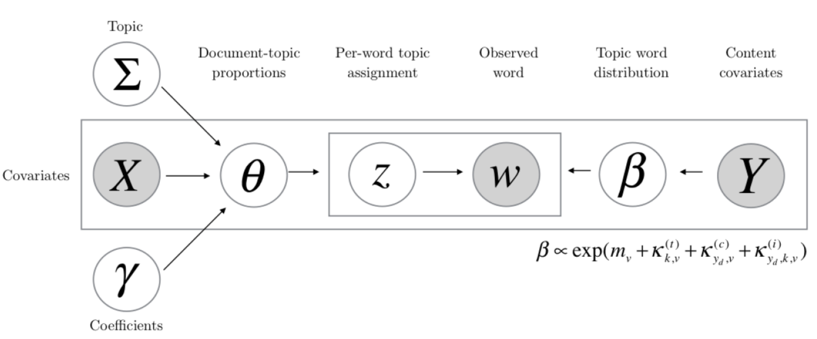
estimator, data, analysis, variance, model, inference

The STM Allows for:

1. The words in each topic to vary by gender -> **content** covariate
2. The topic proportions to vary by group -> **prevalence** covariate(s)

Stewart (2017). LDA and Beyond.

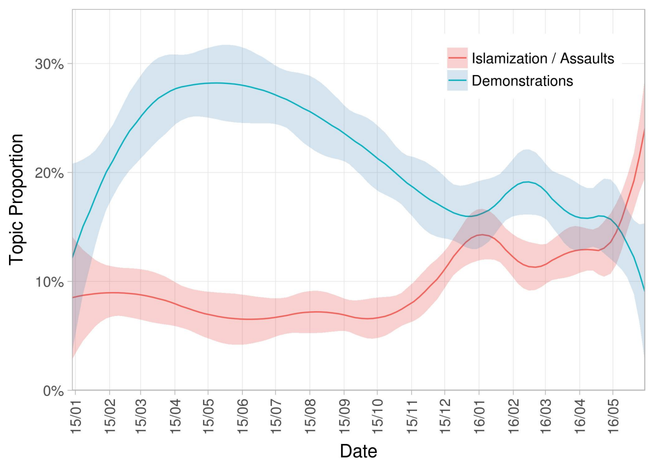
STM - graphical model



Topic
Document-topic proportions
Per-word topic assignment
Observed word
Topic word distribution
Content covariates
Covariates
Coefficients
Σ
X
θ
z
w
β
Y
γ
$\beta \propto \exp(m_v + \kappa^{(t)}_{k,v} + \kappa^{(c)}_{y_d,v} + \kappa^{(i)}_{y_d,k,v})$

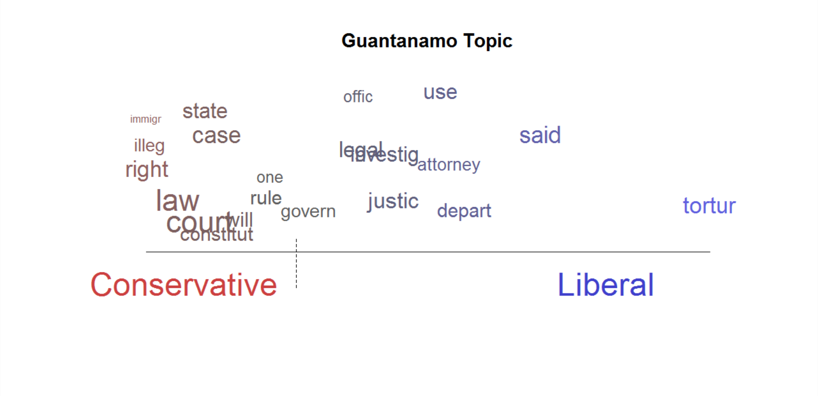
Stewart (2017). LDA and Beyond.

STM - prevalence example: time



Pegida Facebook Posts, December 2014 to May 2016.

STM - content example: ideology



Posts of US politicial blogs, 2008.

STM - R Package

- structural topic models can be estimated and interpreted with the R package `stm`
- the vignette introduces the functionality of the package and is worth a read
- package advantages:
  - very comprehensive documentation.
  - good starting values for hyperparameters
  - extensive functionality for validating and visualising models
- disadvantages: base R approach (no `ggplot2` graphics, no `tidy` data structures (we will fix that with stminsights)

Reload our DonorsChoose data

```r
library(tidyverse)
library(quanteda)
library(stm)
library(stminsights)
theme_set(theme_minimal())
load('data/dfm_donor.Rdata')
dim(dfm_donor)
```

```
## [1] 10000  6751
```

Prepare textual data for STM

- You can provide input data for the stm package in several ways:
  - via STM's own functions for text pre-processing
  - via directly passing quanteda dfm's
  - using quanteda's `convert()` function to prepare dfm's (recommended option)

```r
out <- convert(dfm_donor, to = 'stm')
names(out)
```

```
## [1] "documents" "vocab"     "meta"
```

STM - model fitting

For our first model, we will choose 30 topics and include school metro type (urban vs rural), teacher gender and a flexible spline for date as prevalence covariates:

```r
stm_30 <- stm(documents = out$documents,
      vocab = out$vocab,
      data = out$meta,
      K = 30,
      prevalence = ~ school_metro_type + gender + s(date_num),
      verbose = TRUE) # show progress

stm_effects30 <- estimateEffect(1:30 ~ school_metro_type +
      gender + s(date_num),
      stmobj = stm_30, metadata = out$meta)
```

Saving and restoring models

- depending on the number of documents and the vocabulary size, fitting STM models can require a lot of memory and computation time
- it can be useful to save model objects as R binaries and reload them as needed:

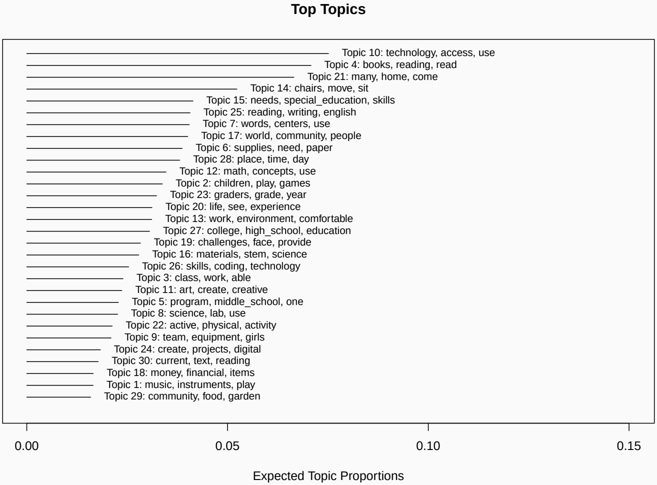
```
save(out, stm_30, stm_effects30,
     file = "data/stm_donor.RData")
```

```
load("data/stm_donor.RData") # reload data
```

Model validation and interactively exploring STM models

Model interpretation - topic proportions

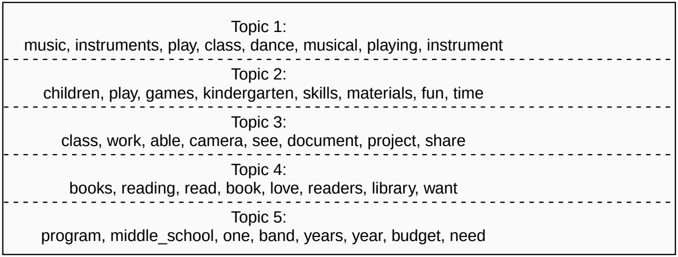
```
plot.STM(stm_30, type = 'summary', text.cex = 0.8)
```



Model interpretation - probability terms

`label` plots show terms for each topic with (again) the most likely terms as a default:

```
plot.STM(stm_30, type = 'labels', n = 8,
         text.cex = 0.8, width = 100, topics = 1:5)
```



Topic 1:
music, instruments, play, class, dance, musical, playing, instrument

Topic 2:
children, play, games, kindergarten, skills, materials, fun, time

Topic 3:
class, work, able, camera, see, document, project, share

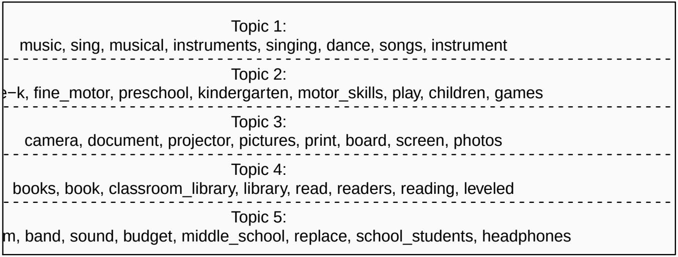
Topic 4:
books, reading, read, book, love, readers, library, want

Topic 5:
program, middle_school, one, band, years, year, budget, need

Model interpretation - frex terms

One strength of STM is that it also offers other metrics for topic terms. `frex` terms are both frequent and exclusive to a topic.

```
plot.STM(stm_30, type = 'labels', n = 8, text.cex = 0.8,
         width = 100, topics = 1:5, labeltype = 'frex')
```



Topic 1:
music, sing, musical, instruments, singing, dance, songs, instrument

Topic 2:
e−k, fine_motor, preschool, kindergarten, motor_skills, play, children, games

Topic 3:
camera, document, projector, pictures, print, board, screen, photos

Topic 4:
books, book, classroom_library, library, read, readers, reading, leveled

Topic 5:
m, band, sound, budget, middle_school, replace, school_students, headphones

Model interpretation - don't rely on terms only

- assigning labels for topics only by looking at the most likely terms is generally not a good idea
- sometimes these terms contain domain-specific stop words. Sometimes they are hard to make sense of by themselves
- recommendation:
    - use probability (most likely) terms
    - use frex terms
    - **qualitatively examine representative documents**

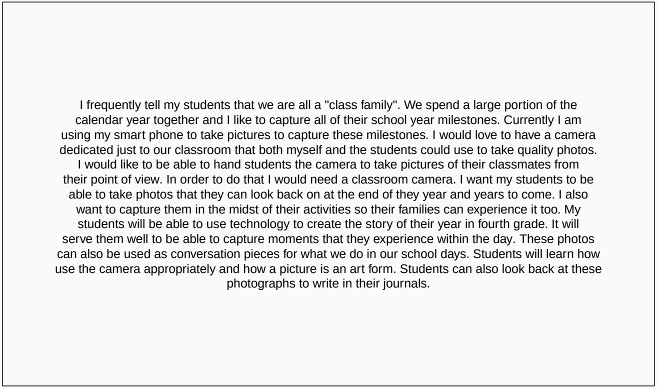
Model interpretation - representative documents

STM allows to find representative (unprocessed) documents for each topic with `findThoughts()`, which can then be plotted with `plotQuote()`:

```r
thoughts <- findThoughts(stm_30,
     texts = out$meta$text, # unprocessed documents
     topics = 1:3,  n = 2) # topics and number of documents
```

Model interpretation - representative documents

```r
plotQuote(thoughts$docs[[3]][1], # topic 3 thought 1
          width = 100)
```



I frequently tell my students that we are all a "class family". We spend a large portion of the calendar year together and I like to capture all of their school year milestones. Currently I am using my smart phone to take pictures to capture these milestones. I would love to have a camera dedicated just to our classroom that both myself and the students could use to take quality photos. I would like to be able to hand students the camera to take pictures of their classmates from their point of view. In order to do that I would need a classroom camera. I want my students to be able to take photos that they can look back on at the end of they year and years to come. I also want to capture them in the midst of their activities so their families can experience it too. My students will be able to use technology to create the story of their year in fourth grade. It will serve them well to be able to capture moments that they experience within the day. These photos can also be used as conversation pieces for what we do in our school days. Students will learn how use the camera appropriately and how a picture is an art form. Students can also look back at these photographs to write in their journals.

Interactive model validation - stminsights

You can interactively validate and explore structural topic models using the R package *stminsights*. What you need:

- one or several stm models and corresponding effect estimates
- the out object used to fit the models
- the example `stm_donor.RData` includes all required objects

```
run_stminsights()
```

For more details see `https://cschwem2er.github.io/stminsights`

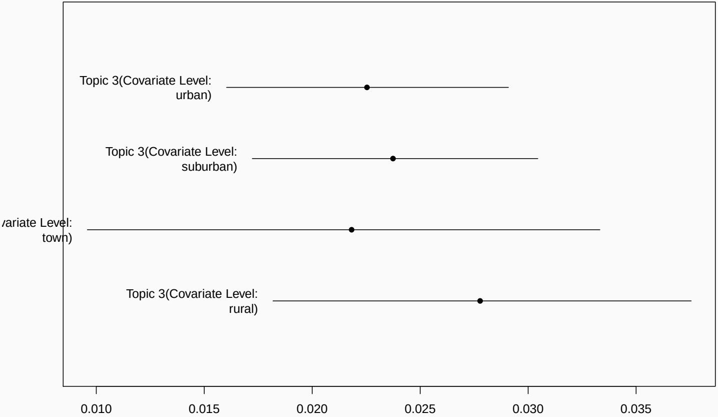
Interpreting and visualizing prevalence and content effects

Prevalence effects (stm package)

- prevalence covariates affect topic proportions
- they can be visualized in three ways:
  - `pointestimate`: pointestimates for categorical variables
  - `difference`: differences between topic proportions for two categories of one variable
  - `continuous`: line plots for continuous variables
- you can also visualize interaction effects if you integrated them in your STM model (see `?plot.estimateEffect()`)

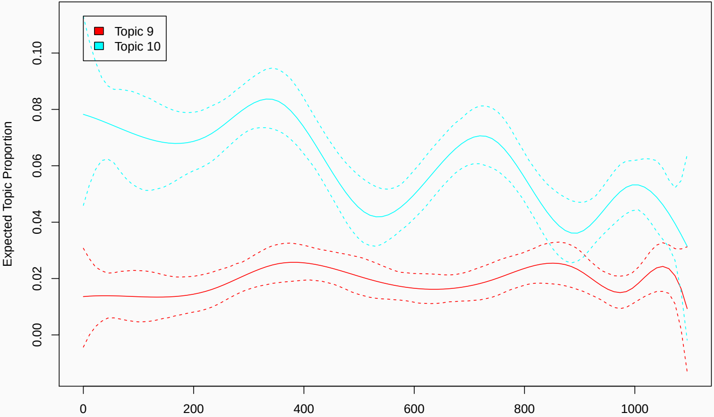
Prevalence effects - pointestimate

```
plot.estimateEffect(stm_effects30, topic = 3,
            covariate = 'school_metro_type', method = 'pointestimate')
```



Prevalence effects - continuous

```
plot.estimateEffect(stm_effects30, covariate = "date_num",
                    topics = c(9:10), method = "continuous")
```



Prevalence effects with stminsights

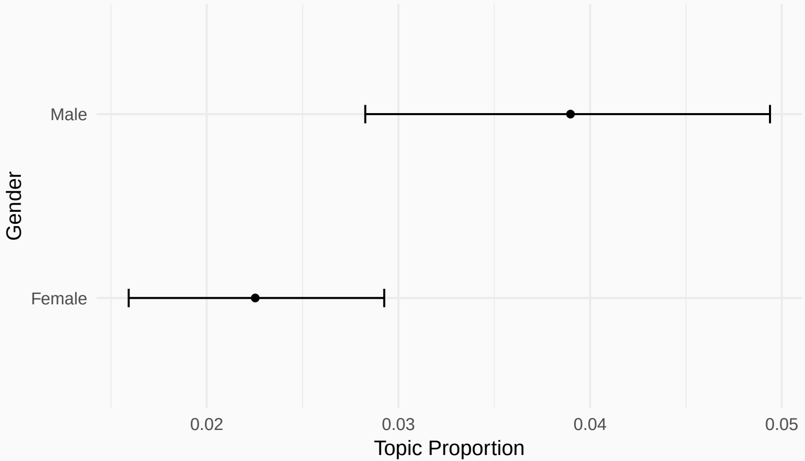
You can use get_effects() to store prevalence effects in a tidy data frame:

```
effects <- get_effects(estimates = stm_effects30,
                       variable = 'gender',
                       type = 'pointestimate')
```

Afterwards, effects can for instance be visualized with ggplot2

Prevalence effects with stminsights

```
effects %>% filter(topic == 3 & value %in% c('Female', 'Male')) %>%
ggplot(aes(x = value, y = proportion)) + geom_point() +
 geom_errorbar(aes(ymin = lower, ymax = upper), width = 0.1) +
 coord_flip() + labs(x = 'Gender', y = 'Topic Proportion')
```



STM content effects

- content effects allow covariates to affect word distributions **within a topic** (e.g. female teachers talk differently about sports in comparison to male teachers). Example model formula: `content = ~ gender`
- this feature is powerful but comes with some disadvantages:
  - you can only use one discrete variable for content effects
  - interpreting the model is more complicated (see 'labelTopics() and sageLabels()
  - we will focus on visualizing content effects with `perspective` plots

Fitting content models

- content models can (but do not have to) be combined with prevalence models. We fit a model with 20 topics and teacher gender as content covariate
- important note: this as a new model and can show different results, even if you compare it to a model with the same number of topics

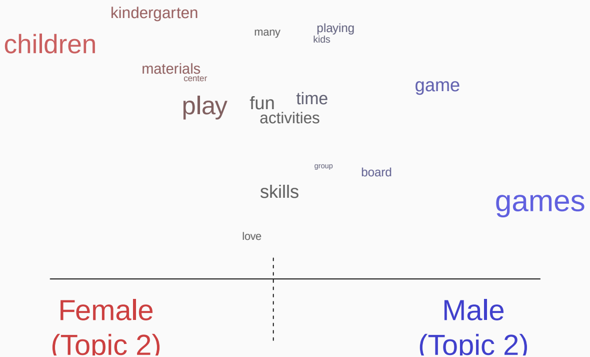
```r
stm_20_content <- stm(documents = out$documents,
      vocab = out$vocab,
      data = out$meta,
      K = 20,
      prevalence = ~ school_metro_type + gender + s(date_num),
      content = ~ gender,
      verbose = FALSE) # show progress
stm_effects20 <- estimateEffect(1:20 ~ school_metro_type +
      gender + s(date_num),
      stmobj = stm_20_content, metadata = out$meta)
save(stm_20_content, stm_effects20,file =  "data/stm_donor_content.RData")
```

Load content model

```r
load("data/stm_donor_content.RData") # reload data
```

Visalizing content effects

```
plot.STM(stm_20_content, topics = c(2), type = 'perspectives',
         covarlevels = c('Female', 'Male'))
```



Questions?