FOMC Topic Modeling Summary

Tommy Jones

This document summarizes topic modeling on FOMC communications provided by *Wonseong Kim* and *Jan Spoerer*.

Data summary

The data set consists of 32,034 sentences of FOMC communications from January 3, 2006 to February 22, 2023. The communictions are drawn from meeting minutes of the FOMC. Of these 32 thousand sentences, 1,405 have been manually tagged for their sentiment on economic growth, employment, and inflation. The tagged sentences were not used for this analysis.

Text data preparation

I prepared the text of the sentences using the below procedure:

1. Stop words were *not* removed from a list. Instead, they were removed by frequency, as described below.
2. Each word was stemmed using Porter's word stemmer.
3. For each year separate document term matrices (where each row is a sentence) were created by:

    a) Constructing unigrams:
    - Remove unigrams that appear in half or more sentences per year
    - Remove unigrams that appear in 2 or fewer sentences per year

    b) Constructing bigrams:
    - Remove bigrams that don't start with a retained unigram in that year
    - Remove bigrams that appear in 5 or fewer sentences per year

This process has two implications that might want to be reconsidered before putting any version of this in production.

1. Vocabulary was constructed independently per year, instead of sequentially. As a result, we might have discarded words in later years that appeared in earlier years. In a production setting, we might wish to retain any words that were retained in earlier periods, even if they were infreqeuent in later periods.
2. Vocabulary was constructed on a per-year basis, but modeling was done on a per-date basis. As a result, there is a little bit of information leakage from the future in vocabulary construction. I doubt this is a significant issue.

In a production setting we'd want to have a vocabulary construction procedure that works on a single document at a time. (One could just retain all words, but over time, I think that would lead to a vocabulary blow up in the models, leading to an even larger memory footprint and longer training times.)

Topic modeling procedure

I use the tidylda package for the R language. tidylda implements an extension of Latent Dirichlet Allocation (LDA) called *transfer* LDA or tLDA. This allows users to fine tune an existing topic model with new data. I begin by topic modeling sentences with a date of January 3, 2006 and fine tune by adding more data at each subsequent date. The result is a chain of 131 topic models.

LDA models calculated by tidylda produce 3 relevant matrices for analysis. (Other objects produced are described as needed throughout this document.) These matrices are:

- $\boldsymbol{\Theta}$ whose d, k entries are $P(\text{sentence}_d|\text{topic}_k)$.
- $\boldsymbol{B}$ whose k, v entries are $P(\text{topic}_k|\text{token}_v)$.
- $\boldsymbol{\Lambda}$ whose k, v entries are $P(\text{topic}_k|\text{token}_v)$.

Selecting user-defined parameters

Topic modeling requires the user to set 3 parameters:

- K — the number of topics
- $\boldsymbol{\alpha}$ — the Dirichlet prior tuning the distribution of topics over documents (sentences in this case)
- $\boldsymbol{\eta}$ — the Dirichlet prior tuning the distribution of words over topics

In addition, a user must specify the number of total and burn in iterations for the Gibbs sampler, used to estimate the model. I chose 200 total iterations and 150 burn in iterations. As a result, tidylda averages the posterior parameters of the model in the last 50 iterations.

In using tLDA, I only need to set the parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ for the first model in the chain. Models in subsequent periods take their priors from the model in the previous period in the chain. tLDA itself has a user-defined parameter. I discuss it in the next section.

Selecting $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$

I select asymmetric priors for $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$. According to *this widely cited paper*, there are significant advantages to as asymmetric $\boldsymbol{\alpha}$. And *my own research* indicates that $\boldsymbol{\eta}$ should be asymmetric and proportional to a power law—or at the very least proportional to the word frequencies in the training corpus.

I set $\boldsymbol{\alpha}$ by sampling K times from a geometric distribution with parameter 0.05. (The choice of the geometric distribution and its parameter are to get a sufficiently asymmetric distribution. But the choice is largely arbitrary.) I then add a tiny bit to the vector to ensure all entries are non-zero using R's .Machine$double.eps, equivalent to the machine epsilon. Finally, I rescale the vector so that its magnitude is equivalent to $0.1 \cdot K$. tidylda's default value for $\boldsymbol{\alpha}$ is a symmetric vector where each entry is equal to 0.1. This keeps our vector the same magnitude as tidylda's default.

I set $\boldsymbol{\eta}$ to be proportional to the frequencies of each word in the initial corpus (sentences with a date of January 3, 2006). The magnitude of $\boldsymbol{\eta}$ is set to $0.05 \cdot V^{(t)}$, where $V^{(t)}$ is the number of unique tokens in the initial corpus. Again, this is so the vector is the same magnitude as the tidylda default.

Selecting the number of topics

I select a fixed number of topics for each model in the chain, though tidylda's implementation of tLDA allows users to add randomly initialized topics for new models to aid in topic discovery. I choose to keep the number of topics fixed to keep this early analysis simple. Though a future version may included adding—or perhaps even retiring—topics to account for the dynamic way topics come and go in real life discourse.

To select the fixed number of topics, I performed an analysis that modeled all sentences in a year independently. I fit models over a range of 5 to 200 topics, with a step size of 5 topics, with the same specifications for $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$, above. I ultimately used coherence (described in more detail, below) to select the number of topics. But I calculated several variables for analysis:

- *coherence*, averaged across all topics. Coherence falls between -1 and 1 and measures how much support a topic has in the observed data. Values close to 1 indicate the top 5 words in a topic ($P(token|topic)$) have a strong positive and statistically dependent relationship. Values close to 0 indicate the top 5 words in a topic are statistically-independent of each other. Values close to -1 (which in practice don't happen) indicate the top 5 words in a topic have a strong negative and statistically dependent relationship. A technical definition of this coherence metric can be found here
- R^2, a measure of goodness-of-fit for the topic model on the training data. See chapter 3 of my dissertation (linked above) for more details. But it's effectively the R^2 used for linear models generalized for outcomes in multiple dimensions. In this case, the outcome is the observed word frequencies.
- *log likelihood* of the model given the data. In this case, I average the log likelihood across the last 50 iterations of each model.
- *skewness* of the distribution of each $\boldsymbol{\beta}_k$, averaged across all topics.

I selected the number of topics procedurally, as described below. Admittedly, this process is somewhat arbitrary. However, eyeballing the data (see below) it looks like it's not bad. Moreover, an analysis of the resulting topics indicates that there are only a few "junk" topics, with the overwhelming majority finding support in the data. My experience with simulated data is that you can fit a model with a few "junk" topics (i.e., you specified more topics than the data actually have) and the good topics are useful. However, if you don't have any junk topics, then you might have specified too few topics in the model and it's pathologically misspecified.

The procedure for selecting the optimum number of topics for a year's sentences is as follows:

1. Fit models over a range of K from 5 to 200, with a step size of 200.
2. Fit a loess model of the mean coherence on K. (I use the loess to smooth the coherence vs. K curve.)
3. Take the K that maximizes the fitted value of the loess model.

The plot below illustrates this process for the 2012 data.

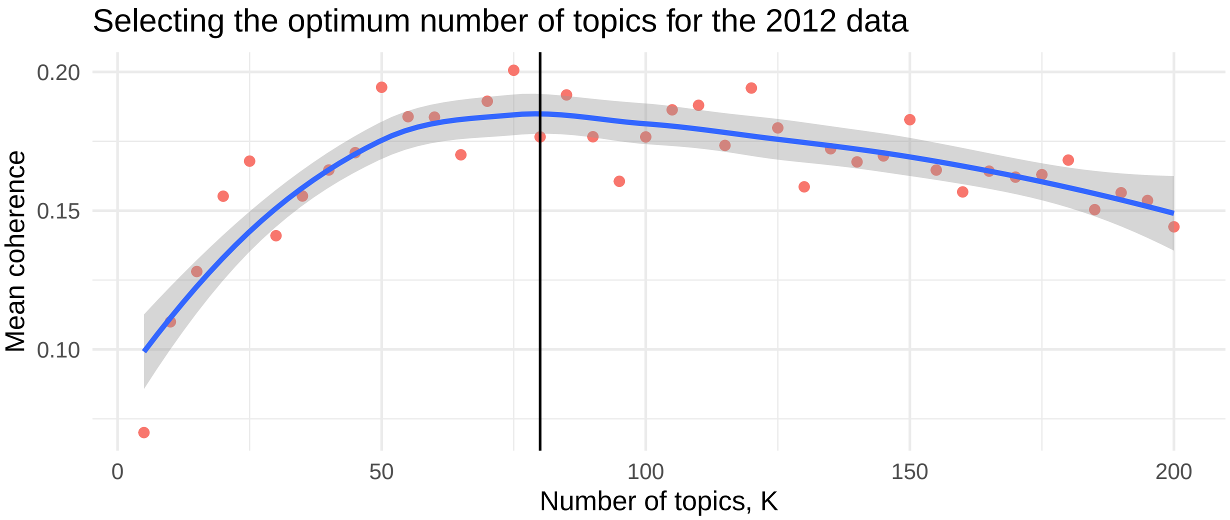


Figure 1: The "optimal" number of topics is the maximum of the blue loess line

The overall number is the average optimal number of topics across all years. A scatterplot of K vs. mean coherence is plotted below. The points are colored by each year. A blue loess line is plotted for illustrative purposes. The black vertical line is the selected number of topics. As you can see, with a couple exceptions, each year follows roughly the same path.

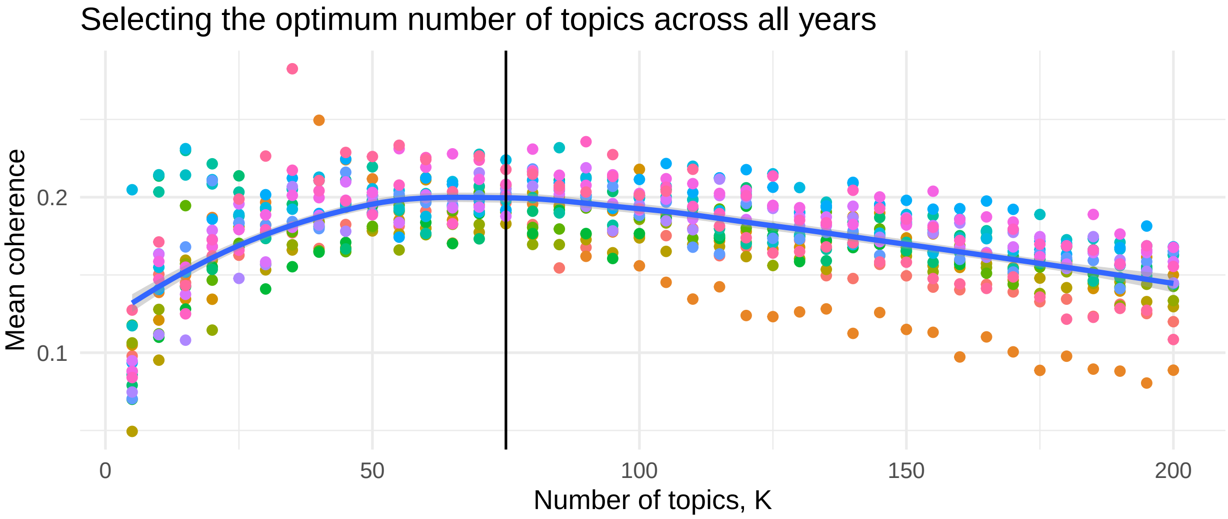


Figure 2: The selected number of topics is the average over all years.

It's worth noting that this method involves data from the future informing the past. I doubt it's a big issue for our purposes. However a rigorous evaluation of this approach (e.g., if you wanted to put this model into production) would be to recursively select the number of topics for each time period between year$_0$ and year$_t$, advancing t a year at a time and seeing if one detects evidence of model misspecification when fine tuning on data from time $t + 1$.

Constructing the chain of models

I partition the data so that sentences at each unique date becomes its own data set. I begin by training an initial LDA model on the sentences with date January 3, 2006 and then use tLDA to fine tune the model at each subsequent date. The result is a Markov chain of 131 toic models.

A technical overview of tLDA can be found *here*. A more in-depth evaluation can be found in chapter 5 of my dissertation, linked above.

tLDA has a few user defined parameters:

- **a**, called **prior_weight** in `tidylda`: this tunes the weight of the posterior of the model in time $t - 1$ used in the prior of a model at time t. When $a = 1$, a single word occurrence at all time periods prior to time t have the same weight in the posterior as a single word occurrence at time t. According to my research, an optimal value of a lies somewhere between 0.6 and 0.9. I set $a = 0.7$ for this model chain.
- **number of total and burn in iterations**: I set these to the same values used to select the number of topics, described above.
- **additional topics**: One can add randomly-initialized topics to the model to account for the possiblilty of a topic appearing at time t that was not present from time t_0 to $t - 1$. For this analysis, I choose not to add any additional topics as time goes by for simplicity.

Summary of outputs

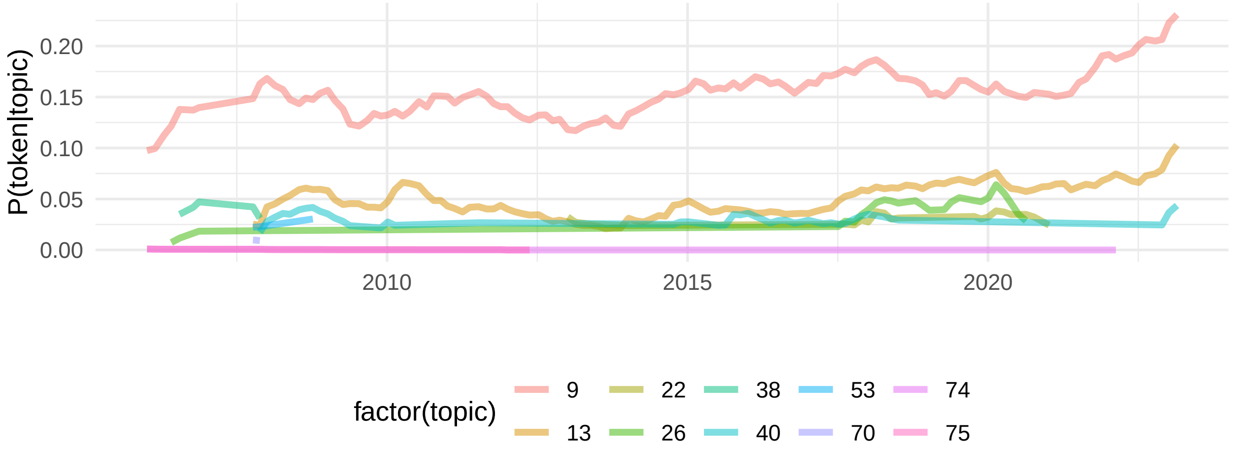
I've collected the outputs into several objects for downstream analysis. Most of these are provided as .csv text files with the orignal objects stored in an R format, .rds files. These are

- **model-list.rds**: A 1.7 Gb list of `tidylda` model objects. Each element is a topic model from one time period in the chain. Due to its size and non-tabular format, I've only provided the .rds file.
- **model-summaries.csv(.rds)**: Table of summary information for all models in the chain. Individual models are specified by the *date* field. Columns are
  - *topic* – the unique topic ID
  - *prevalence* – the percent of tokens ascribed to each topic. (i.e., The volume of a topic across all sentences for a time period.) Each value is taken from $\frac{\sum_d n_d \cdot \theta_{d,k}}{\sum_k \sum_d n_d \cdot \theta_{d,k}} \cdot 100$
  - *coherence* – measure between -1 and 1 of how much support a topic has in the observed data.
  - *top_terms* – the top 5 tokens as ordered by $P(\text{token}_v | \text{topic}_k)$, rows of the $\boldsymbol{B}$ matrix.
  - *top_terms_lambda* – the top 5 tokens as ordered by $P(\text{topic}_k | \text{token}_v)$, rows of the $\boldsymbol{\Lambda}$ matrix.
  - *skewness* – a measure of skewness of the distribution $\boldsymbol{\beta}_k$. Topics with low skewness are likely low-quality topics as human language follows a power law distribution, which is highly skewed.
  - *date* – the date of the document(s) for the model in the chain. Each unique date is a unique model.
- **model-summaries-monthly.csv(.rds)**: The *model-summaries* table, above, where values have been aggregated to a monthly frequency. The date is the last day of the month. This matrix can be more easily merged with monthly economic data for downstream modeling. In addition to the columns in the *model-summaries* table, *model-summaries-monthly* contains the following additional columns.
  - *prev_roll3m* – a rolling average of prevalence 3 months ending in the month in the *date* column.
  - *prev_roll6m* – a rolling average of prevalence 6 months ending in the month in the *date* column.
  - *prev_roll12m* – a rolling average of prevalence 12 months ending in the month in the *date* column.
  - *prev_pct3m* – a percent change of prevalence from 3 months prior.
  - *prev_pct6m* – a percent change of prevalence from 6 months prior.
  - *prev_pct12m* – a percent change of prevalence from 12 months prior.
- **agg-model-summary.csv(.rds)**: Some aggregate statistics of each model. In general, I expect these statistics to be largely independent of time and symmetrically distributed. (Eyeballing it, they mostly are. But I didn't do a rigorous analysis.) In several cases, coherence for a topic is *NaN*, or "not a number". This is when there are no words in a topic's top 5 words present in the corpus. (As a result of tLDA modeling.) The columns are:
  - *mean_coherence* – mean coherence across all topics
  - *var_coherence* – variance of coherence across all topics
  - *skew_coherence* – skewness of coherence across all topics
  - *var_prevalence* – variance of prevalence across all topics
  - *skew_prevalence* – skewness of prevalence across all topics
  - *r2* – The R^2 of the model
  - *date* – The date which identifies the model
- **tidy-beta.csv(.rds)**: a version of the $\boldsymbol{B}$ matrix that is reformatted for easier analysis. I limited this matrix to only include the top 10 tokens per topic, to limit its size. It has four columns:
  - *topic* – topic index
  - *token* – the token (stemmed unigrams and bigrams)
  - *beta* – the value $P(\text{token}_v | \text{topic}_k)$
  - *date* – similar to the summary tables, date identifies each model
- **tidy-lambda.csv(.rds)**: a version of the $\boldsymbol{\Lambda}$ matrix that is reformatted for easier analysis. I limited this matrix to only include the top 10 tokens per topic, to limit its size. It has four columns:

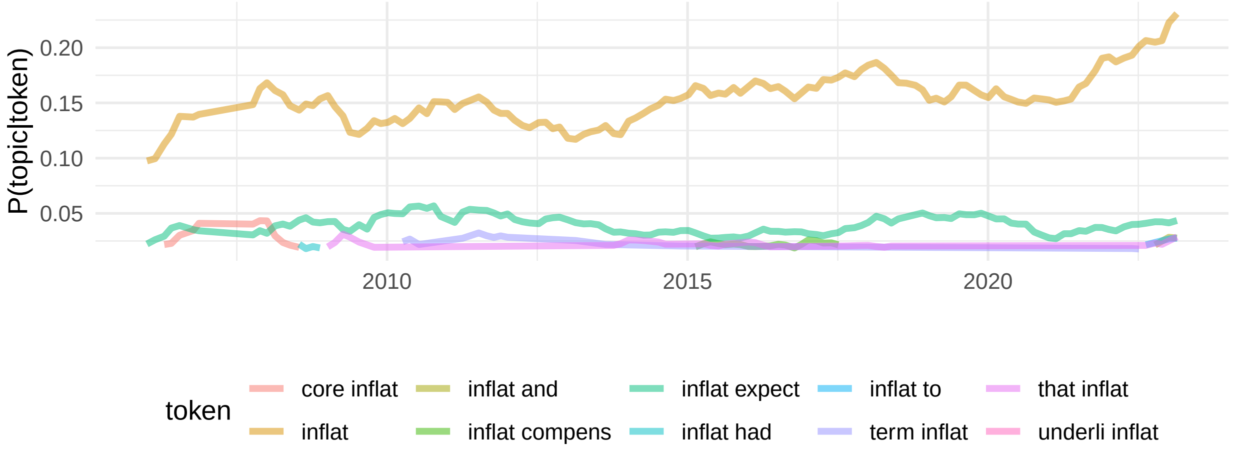
- *topic* – topic index
- *token* – the token (stemmed unigrams and bigrams)
- *beta* – the value $P(\text{topic}_k|\text{token}_v)$
- *date* – similar to the summary tables, date identifies each model

Example analyses

Prevalence of the token 'inflat' across topics

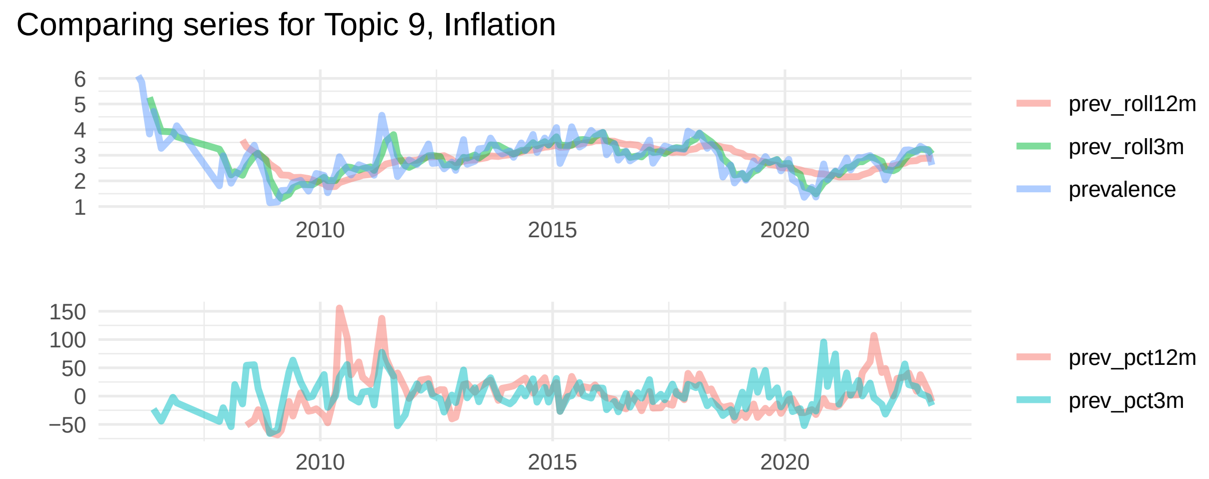


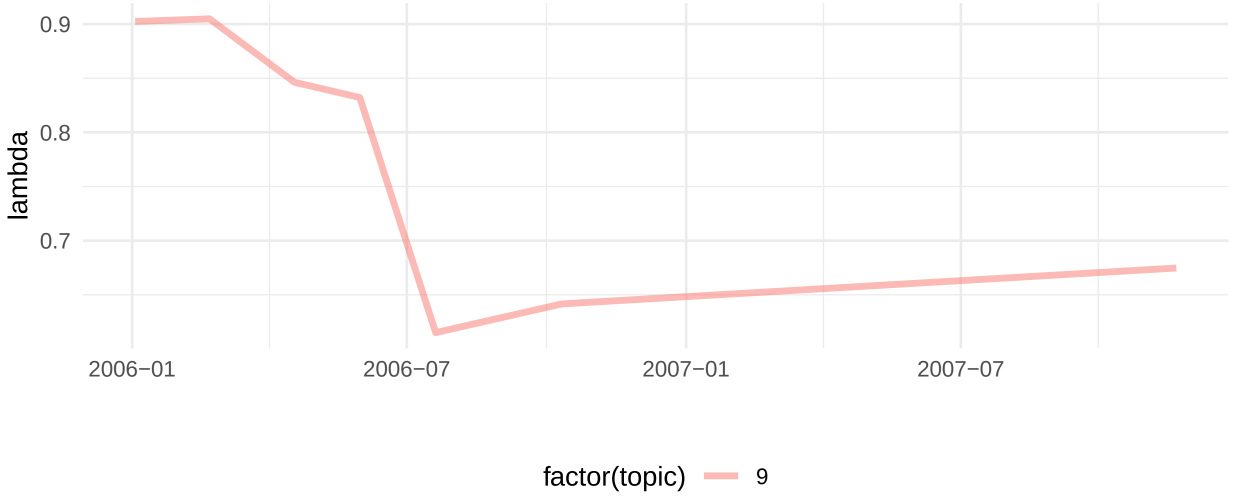
Prevalence of tokens containing 'inflat' in topic 9

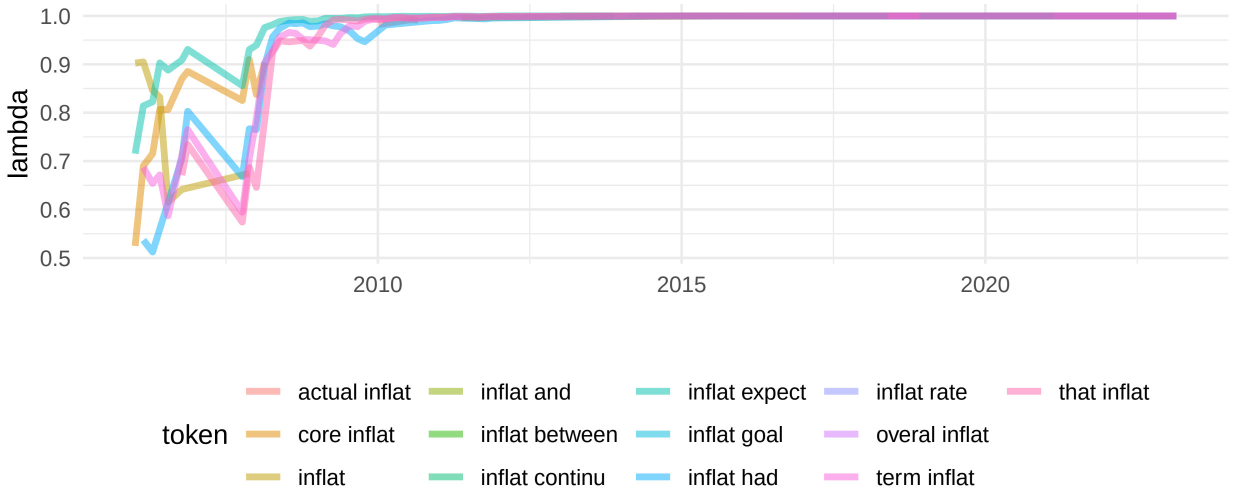


```
## Warning: Removed 13 rows containing missing values (‘geom_line()‘).

## Warning: Removed 15 rows containing missing values (‘geom_line()‘).
```


Comparing series for Topic 9, Inflation
6
5
4
3
2
1
2010
2015
2020
prev_roll12m
prev_roll3m
prevalence
150
100
50
0
−50
2010
2015
2020
prev_pct12m
prev_pct3m


lambda
0.9
0.8
0.7
2006−01
2006−07
2007−01
2007−07
factor(topic)
9


lambda
1.0
0.9
0.8
0.7
0.6
0.5
2010
2015
2020
token
actual inflat
core inflat
inflat
inflat and
inflat between
inflat continu
inflat expect
inflat goal
inflat had
inflat rate
overal inflat
term inflat
that inflat