Just slightly adapted/condensed from David Mimno's "mallet" package vignette.

```
# First, we tell R use the "Mallet" library. Without it, the language
# has no idea what topic modeling *is*.

library(mallet)

# First, we read in a *single* text file, where each line is a document.
# This is the format that our regular expression program in python
# created: but there are lots of different ways to do this.

# Depending on the term you searched for, the file location here will be different.
# You can just put your cursor after the slash following "texts" and type tab: you'll
# get a list of all the files in the directory, including the one you created.

your_file = "/texts/[Nn]eurasthenia.txt"

# Here we read it in, one paragraph at a time.
paragraphs = scan(your_file,what="raw",sep="n")

# Paste this in to the first four paragraphs.
paragraphs[1:4]

# Now we create a mallet instance: this is a little program that stores our texts in
# a file for parsing, along with a list of English stopwords.

# If your texts are in another language, you may need to find a different language's
# stopword list.

# YOU MUST HAVE PUT THE
set.seed(1)

mallet.instances = mallet.import(id.array = as.character(1:length(paragraphs)),
                                 text.array = paragraphs,
                                 stoplist.file = "/texts/english_stopwords.txt")

# Here we set up the model and get ready to run.
# Incant, incant, incant.

topic.model = MalletLDA(num.topics=15)
topic.model$loadDocuments(mallet.instances)
topic.model$setAlphaOptimization(20, 50)
topic.model$model$setRandomSeed(as.integer(1))

# And finally we run it. These numbers say to run it for 200 steps, and then take another 20
# to fine-tune the results.
```

```
topic.model$train(200)
topic.model$maximize(20)

word.freqs = mallet.word.freqs(topic.model)
head(word.freqs)

doc.topics = mallet.doc.topics(topic.model, smoothed=TRUE, normalized=TRUE)
topic.words = mallet.topic.words(topic.model, smoothed=TRUE, normalized=TRUE)

# What are the top words in topic 1?

mallet.top.words(topic.model, word.weights = topic.words[6,], num.top.words = 10)

## For me, this is a topic about gastric pain and digestion. (Topic models aren't completely deterministi

## We can then make a sorted list, for example, and read the paragraphs that are most about digestions.

## You just change the number "6" below to check any other topic.


sorted = paragraphs[order(doc.topics[,6],decreasing = T)]
sorted[1:3]
```