

# Package ‘SherlockHolmes’

January 20, 2025

**Version** 1.0.1

**Date** 2023-03-28

**Title** Building a Concordance of Terms in a Series of Texts

**Maintainer** Barry Zeeberg <barryz2013@gmail.com>

**Author** Barry Zeeberg [aut, cre]

**Depends** R (>= 4.2.0)

**LazyData** true

**Imports** qpdf, stringr, dptest, tableHTML, plotrix, zoo, stargazer,  
utils, graphics, grDevices, stats, textBoxPlacement,  
plot.matrix, devtools

**Description** Compute the frequency distribution of a search term in a series of texts. For example, Arthur Conan Doyle wrote a total of 60 Sherlock Holmes stories, comprised of 54 short stories and 4 longer novels. I wanted to test my own subjective impression that, in many of the stories, Sherlock Holmes' popularity was used as bait to induce the reader to read a story that is essentially not primarily a Sherlock Holmes story. I used the term ``Holmes" as a search pattern, since Watson would frequently address him by name, or use his name to describe something that he was doing. My hypothesis is that the frequency distribution of the search pattern ``Holmes" is a good proxy for the degree to which a story is or is not truly a Sherlock Holmes story. The results are presented in a manuscript that is available as a vignette and online at <<https://barryzee.github.io/Concordance/index.html>>.

**License** GPL (>= 2)

**Encoding** UTF-8

**VignetteBuilder** knitr

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**RoxygenNote** 7.2.3

**Config/testthat/edition** 3

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2023-03-28 14:40:06 UTC

## Contents

chronology	2
coChronology	3
concordance	4
contingency	5
csp	5
csw	5
distributions	6
freqHist	6
freqs	7
frequency	7
grabFunctionParameters	8
inside	8
lengths	9
mergeTables	9
outside	10
patterns	10
plot_dpseg2	11
readTitles	12
retrieveLmStats	12
rolling	13
segments	14
segs	14
Sherlock	15
startLine	16
starts	17
strSplitTab	17
texts	17
texts.vec	18
titles	18
titles.vec	18
<b>Index</b>	<b>19</b>

---

chronology

*chronology*

---

### Description

frequencies plotted in order of date (if the titles are given in order of date)

### Usage

```
chronology(titles.vec, patterns, starts, freqs, chronDir, overlay = FALSE)
```

**Arguments**

titles.vec	character vector containing the titles of the stories
patterns	vector of character string query patterns
starts	integer vector of starting positions
freqs	return value of frequency()
chronDir	character string full path name for output directory
overlay	Boolean if TRUE overlay the chronolgy for multiple search patterns

**Value**

returns no value, but has side effect generating graph

**Examples**

```
freqDir<-tempdir()
chronDir<-sprintf("%s/chronology", freqDir)
dir.create(chronDir)
dir.create(sprintf("%s/plots", chronDir))
dir.create(sprintf("%s/archive", chronDir))
print(chronDir)
chr<-chronology(titles.vec, c("Holmes", "Watson"), starts, freqs, chronDir)
```

---

coChronology

*coChronology*


---

**Description**

graphical indicator of search patterns within stories

**Usage**

```
coChronology(titles.vec, patterns, starts, freqs, chronDir)
```

**Arguments**

titles.vec	character vector containing the titles of the stories
patterns	vector of character string query patterns
starts	integer vector of starting positions
freqs	return value of frequency()
chronDir	character string full path name for output directory

**Value**

returns an integer matrix whose rows are search patterns and columns are stories, value of 1 indicates the presence of the corresponding search pattern in the corresponding story

**Examples**

```
freqDir<-tempdir()
chronDir<-sprintf("%s/chronology",freqDir)
dir.create(chronDir)
dir.create(sprintf("%s/plots",chronDir))
dir.create(sprintf("%s/archive",chronDir))
print(chronDir)
coch<-coChronology(titles.vec,c("Holmes","Watson"),starts,freqs,chronDir)
```

---

concordance

*concordance*

---

**Description**

retrieve words that are close to occurrences of pattern

**Usage**

```
concordance(freqs, titles.vec, texts.vec, starts, window, odir)
```

**Arguments**

freqs	return value of frequency()
titles.vec	character vector containing the titles of the stories
texts.vec	character vector of entire text
starts	integer vector of starting positions
window	integer number of lines to take before and after the pattern match
odir	character string containing the full path name for the output directory

**Value**

returns no value but has side effect of generating graphs

**Examples**

```
con<-concordance(freqs,titles.vec[3],texts.vec,starts>window=2,odir=tempdir())
```

---

contingency	<i>contingency</i>
-------------	--------------------

---

**Description**

compute chisq value for a 2 x 2 contingency table

**Usage**

```
contingency(inside, outside)
```

**Arguments**

inside	numeric vector of raw counts
outside	numeric vector of raw counts

**Value**

numeric vector of `chisq.test()` p.values

**Examples**

```
con<-contingency(inside=c(4,5),outside=c(20,7))
```

---

csp	<i>Sherlock data sets</i>
-----	---------------------------

---

**Description**

Sherlock data sets

**Usage**

```
data(csp)
```

---

csw	<i>Sherlock data sets</i>
-----	---------------------------

---

**Description**

Sherlock data sets

**Usage**

```
data(csw)
```

---

distributions	<i>distributions</i>
---------------	----------------------

---

**Description**

compute distribution of ratio of number of occurrences of query string divided by total number of words

**Usage**

```
distributions(freqs, titles.vec, minl, P, odir)
```

**Arguments**

freqs	return value of frequency()
titles.vec	character vector containing the titles of the stories
minl	is an integer param passed to dpseg::dpseg
P	is a numeric param passed to dpseg::dpseg
odir	character string containing the full path name for the output directory

**Value**

returns no value but has side effect of generating graphs

**Examples**

```
dis<-distributions(freqs,titles.vec[1],minl=100,P=0.00001,tempdir())
```

---

freqHist	<i>freqHist</i>
----------	-----------------

---

**Description**

histogram of frequencies

**Usage**

```
freqHist(patterns, starts, titles.vec, freqs, histDir)
```

**Arguments**

patterns	vector of character string query patterns
starts	integer vector of starting positions
titles.vec	character vector containing the titles of the stories
freqs	return value of frequency()
histDir	character string full path name for output directory

**Value**

returns no value, but has side effect generating histogram

**Examples**

```
fh<-freqHist(patterns,starts,titles.vec,freqs,histDir=tempdir())
```

---

freqs	<i>Sherlock data sets</i>
-------	---------------------------

---

**Description**

Sherlock data sets

**Usage**

```
data(freqs)
```

---

frequency	<i>frequency</i>
-----------	------------------

---

**Description**

compute ratio of number of occurrences of query string divided by total number of words

**Usage**

```
frequency(texts.vec, starts, patterns)
```

**Arguments**

texts.vec	character vector of entire text
starts	integer vector of starting positions
patterns	vector of character string query patterns

**Value**

- a list whose components are sub-lists  
 # indexed by the titles of the stories
- start integer starting line in text
  - end integer ending line in text
  - wPerLine integer words perline
  - wordSum integer sum of wPerLine
  - patterns a sub-list
    - integer pPerLine integer patterns per line
    - patSum integer total of pPerLine
    - fraction numeric ratio of patSum/wordSum

**Examples**

```
fr<-frequency(texts.vec,starts,patterns)
```

---

```
grabFunctionParameters
      grabFunctionParameters
```

---

**Description**

retrieve capture all of the parameter names and values passed in

**Usage**

```
grabFunctionParameters()
```

**Details**

copied and pasted from <https://stackoverflow.com/questions/66329835/using-r-how-to-get-all-parameters-passed-into-a-function-with-their-values>

**Value**

a list whose components are the symbolic names of the function parameters, and their values.

---

```
inside      Sherlock data sets
```

---

**Description**

Sherlock data sets

**Usage**

```
data(inside)
```



---

lengths	<i>lengths</i>
---------	----------------

---

**Description**

frequencies plotted in order of story length

**Usage**

```
lengths(titles.vec, patterns, starts, freqs, lengthDir)
```

**Arguments**

titles.vec	character vector containing the titles of the stories
patterns	vector of character string query patterns
starts	integer vector of starting positions
freqs	return value of frequency()
lengthDir	character string full path name for output directory

**Value**

returns no value, but has side effect generating graph

**Examples**

```
freqDir<-tempdir()
lengthDir<-sprintf("%s/length",freqDir)
dir.create(lengthDir)
print(lengthDir)
dir.create(sprintf("%s/plots",lengthDir))
dir.create(sprintf("%s/archive",lengthDir))
le<-lengths(titles.vec,patterns,starts,freqs,lengthDir)
```

---

mergeTables	<i>mergeTables</i>
-------------	--------------------

---

**Description**

merge (inner join) the results in 2 tables generated from 2 vectors

**Usage**

```
mergeTables(tv, tw, cnv, cnw)
```

**Arguments**

tv	first table
tw	second table
cnv	character name for column coming from v
cnw	character name for column coming from w

**Value**

numeric matrix generated from merging tables from v and w

**Examples**

```
mt<-mergeTables(inside,outside,"in","out")[1:10,]
```

---

outside	<i>Sherlock data sets</i>
---------	---------------------------

---

**Description**

Sherlock data sets

**Usage**

```
data(outside)
```

---

patterns	<i>Sherlock data sets</i>
----------	---------------------------

---

**Description**

Sherlock data sets

**Usage**

```
data(patterns)
```

---

plot\_dpseg2

*plot\_dpseg2*


---

### Description

Alternative plot procedure for dpseg, special function provided personally by dpseg curator. I made a few custom tweaks Including option to overlay multiple plots

### Usage

```
plot_dpseg2(
  x,
  delog = FALSE,
  col,
  main,
  xlab,
  ylab,
  res = 10,
  vlines,
  overlay,
  textX,
  textY,
  textLabel,
  ylim
)
```

### Arguments

x	dpseg object to plot
delog	Boolean use log scale if TRUE
col	color
main	character title of graph
xlab	character label for x axis
ylab	character label for y axis
res	numeric resolution
vlines	Boolean if FALSE suppress vertical lines in graph
overlay	Boolean if TRUE this plot is an overlay of previous plot
textX	numeric x position for text box
textY	numeric y position for text box
textLabel	character string to label the points in the graph
ylim	numeric vector ylim for plot

**Value**

returns no value but has side effect of producing a graph

**Examples**

```
pdp<-plot_dpseg2(segs,overlay=FALSE,xlab="xaxis",
  ylab="yaxis",vlines=FALSE,textX=2000,textY=20,
  textLabel="label",ylim=c(0,60))
```

---

readTitles

*readTitles*

---

**Description**

read and edit titles to remove blank lines and white space

**Usage**

```
readTitles(titles)
```

**Arguments**

`titles` is a character string containing the full path name for a text file containing the titles of the stories in the same order that they appear in the text file

**Value**

a character vector of titles

**Examples**

```
titles<-system.file("extdata/contents3.txt",package="SherlockHolmes")
rt<-readTitles(titles)
```

---

retrieveLmStats

*retrieveLmStats*

---

**Description**

This function retrieves intercept, slope, r.squared, and adj.r.squared from `lm()`

**Usage**

```
retrieveLmStats(x, y)
```

**Arguments**

x                    is second argument to lm()  
 y                    is first argument to lm()

**Value**

returns a list containing the return value of lm, intercept, slope, r.squared, and adj.r.squared

**Examples**

```
retr<-retrieveLmStats(1:10,runif(10,0,1))
```

---

rolling	<i>rolling</i>
---------	----------------

---

**Description**

compute rolling average of ratio of number of occurrences of query string divided by total number of words

**Usage**

```
rolling(freqs, titles.vec, windowPct = 0.1, odir, verbose)
```

**Arguments**

freqs                return value of frequency()  
 titles.vec          character vector containing the titles of the stories  
 windowPct          a numeric control size of plot window  
 odir                character string containing the full path name for the output directory  
 verbose             Boolean if TRUE print informative or diagnostic messages to console

**Value**

returns noo value, but has side effect of generating graphs

**Examples**

```
rol<-rolling(freqs,titles.vec,windowPct=0.10,odir=tempdir(),verbose=FALSE)
```

---

segments	<i>segments</i>
----------	-----------------

---

**Description**

reformat seqs\$segments as a legend to insert into segment plot

**Usage**

```
segments(segs)
```

**Arguments**

segs            return value of `dpseg::dpseg()`

**Value**

reformatted matrix suitable for printing

**Examples**

```
seg<-segments(segs)
```

---

segs	<i>Sherlock data sets</i>
------	---------------------------

---

**Description**

Sherlock data sets

**Usage**

```
data(segs)
```

---

 Sherlock

*Sherlock*


---

### Description

This function is the driver that organizes the computation of concordances in Sherlock Holmes stories

### Usage

```
Sherlock(
  titles = "NONE",
  texts,
  patterns,
  toupper,
  odir,
  concord = FALSE,
  minl = 100,
  P = 1e-05,
  verbose = FALSE
)
```

### Arguments

titles	is a character string containing the full path name for a text file containing the titles of the stories in the same order that they appear in the texts file. If titles=="NONE", treat the entire book as one story.
texts	is a character string containing the full path name for a text file containing the full texts of all of the stories
patterns	is a vector containing the search patterns
toupper	is a Boolean TRUE if the titles should be converted to upper case
odir	is a character string containing the full path name of the output directory
concord	Boolean if TRUE invoke concordance()
minl	is an integer param passed to dpseg::dpseg
P	is a numeric param passed to dpseg::dpseg
verbose	Boolean if TRUE print informative or diagnostic messages to console

### Value

returns no value but has side effect of driving the concordance computations

## Examples

```
titles<-system.file("extdata/contents3.txt",package="SherlockHolmes")
texts<-system.file("extdata/processed_download3.txt",package="SherlockHolmes")
SH<-Sherlock(titles=titles,texts=texts,patterns=patterns[1],
  toupper=TRUE,odir=tempdir(),concord=FALSE,minl=100,P=0.00001,
  verbose=FALSE)
```

---

startLine

*startLine*

---

## Description

where does each story start?

## Usage

```
startLine(titles.vec, texts.vec, toupper)
```

## Arguments

titles.vec	is a character string containing the full path name for a text file containing the titles of the stories in the same order that they appear in the texts file
texts.vec	is a character string containing the full path name for a text file containing the full texts of all of the stories
toupper	is a Boolean TRUE if the titles should be converted to upper case

## Details

each title in titles.vec must appear on a single line in titles.vec and texts.vec - a title cannot be split across multiple lines. each title must only appear one time within titles.vec and texts.vec

## Value

an integer vector of the starting lines of each story

## Examples

```
s1<-startLine(titles.vec,texts.vec,toupper=TRUE)
```



---

starts	<i>Sherlock data sets</i>
--------	---------------------------

---

**Description**

Sherlock data sets

**Usage**

data(starts)

---

strSplitTab	<i>strSplitTab</i>
-------------	--------------------

---

**Description**

use strsplit to parse words from text t, delete the empty string from the result, and compile into a sorted table of word frequencies

**Usage**

strSplitTab(t)

**Arguments**

t                      vector of character strings representing lines of the original text

**Value**

a sorted table of raw word counts

**Examples**

```
sst<-strSplitTab(texts.vec)
```

---

texts	<i>Sherlock data sets</i>
-------	---------------------------

---

**Description**

Sherlock data sets

**Usage**

data(texts)

---

texts.vec	<i>Sherlock data sets</i>
-----------	---------------------------

---

**Description**

Sherlock data sets

**Usage**

```
data(texts.vec)
```

---

titles	<i>Sherlock data sets</i>
--------	---------------------------

---

**Description**

Sherlock data sets

**Usage**

```
data(titles)
```

---

titles.vec	<i>Sherlock data sets</i>
------------	---------------------------

---

**Description**

Sherlock data sets

**Usage**

```
data(titles.vec)
```

# Index

chronology, 2  
coChronology, 3  
concordance, 4  
contingency, 5  
csp, 5  
csw, 5

distributions, 6

freqHist, 6  
freqs, 7  
frequency, 7

grabFunctionParameters, 8

inside, 8

lengths, 9

mergeTables, 9

outside, 10

patterns, 10  
plot\_dpseg2, 11

readTitles, 12  
retrieveLmStats, 12  
rolling, 13

segments, 14  
segs, 14  
Sherlock, 15  
startLine, 16  
starts, 17  
strSplitTab, 17

texts, 17  
texts.vec, 18  
titles, 18  
titles.vec, 18