

# Package ‘inpdfr’

October 24, 2018

**Type** Package

**Title** Analyse Text Documents Using Ecological Tools

**Version** 0.1.8

**Author** Rebaudo Francois (IRD, UMR EGCE, Univ.ParisSud-CNRS-IRD-  
Univ.ParisSaclay)

**Maintainer** Rebaudo Francois <francois.rebaudo@ird.fr>

**Description** A set of functions to analyse and compare texts, using classical  
text mining functions, as well as those from theoretical ecology.

**License** GPL-2

**LazyData** TRUE

**Imports** wordcloud (>= 2.5), RColorBrewer (>= 1.1-2), tm (>= 0.6-2),  
SnowballC (>= 0.5.1), ca (>= 0.58), cluster (>= 2.0.1),  
entropart (>= 1.4.1), metacom (>= 1.4.4), parallel (>= 3.1.3),  
stringi (>= 1.0-1), R.devices (>= 2.14.0)

**SystemRequirements** XPDF (<http://www.foolabs.com/xpdf/download.html>)

**Suggests** knitr, rmarkdown, testthat

**VignetteBuilder** knitr

**RoxygenNote** 6.1.0

**URL** <https://github.com/frareb/inpdfr/>

**BugReports** <https://github.com/frareb/inpdfr/issues>

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2018-10-24 09:50:03 UTC

## R topics documented:

doCA	2
doCluster	3
doKmeansClust	4

doMetacomEntropart . . . . .	6
doMetacomMetacom . . . . .	7
excludeStopWords . . . . .	8
exclusionList_FR . . . . .	9
exclusionList_SP . . . . .	9
exclusionList_UK . . . . .	10
getAllAnalysis . . . . .	10
getListFiles . . . . .	11
getMostFreqWord . . . . .	12
getMostFreqWordCor . . . . .	13
getPDF . . . . .	14
getStopWords . . . . .	15
getSummaryStatsBARPLOT . . . . .	15
getSummaryStatsHISTO . . . . .	16
getSummaryStatsOCCUR . . . . .	17
getTXT . . . . .	18
getwordOccuDF . . . . .	18
getXFreqWord . . . . .	19
IdentifyStructure . . . . .	20
inpdfr . . . . .	21
loremIpsum . . . . .	21
makeWordcloud . . . . .	22
mergeWordFreq . . . . .	23
postProcTxt . . . . .	24
preProcTxt . . . . .	25
quitSpaceFromChars . . . . .	25
truncNumWords . . . . .	26

<b>Index</b>	<b>27</b>
--------------	-----------

---

doCA	<i>Performs a correspondance analysis on the basis of the word-occurrence data.frame.</i>
------	---

---

## Description

Performs a correspondance analysis on the basis of the word-occurrence data.frame using `ca` function.

## Usage

```
doCA(wordF, getPlot = TRUE, mwidth = 800, mheight = 800,
      formatType = "png", ...)
```

**Arguments**

wordF	The data.frame containing word occurrences.
getPlot	If TRUE, save the <code>ca</code> plot in the RESULTS directory.
mwidth	The width of the plot in pixels.
mheight	The height of the plot in pixels.
formatType	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").
...	Additional arguments from the <code>ca</code> function.

**Value**

The results of the `ca` function.

**Examples**

```
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"), excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
doCA(wordF = wordOccuDF)
```

---

doCluster	<i>Performs a cluster analysis on the basis of the word-occurrence data.frame.</i>
-----------	--

---

**Description**

Performs a cluster analysis on the basis of the word-occurrence data.frame using `hclust` function.

**Usage**

```
doCluster(wordF, myMethod = "ward.D2", gp = FALSE, nbGp = 5,
  getPlot = TRUE, mwidth = 800, mheight = 800, formatType = "png",
  ...)
```

**Arguments**

wordF	The data.frame containing word occurrences.
myMethod	The method to compute distances, see <a href="#">dist</a> function.
gp	A logical to specify if groups should be made.
nbGp	An integer to specify the number of groups. Ignored if gp=FALSE.
getPlot	If TRUE, save the cluster plot in the RESULTS directory.
mwidth	The width of the plot in pixels.
mheight	The height of the plot in pixels.
formatType	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").
...	Additional arguments from the <a href="#">hclust</a> function.

**Value**

An object of class [hclust](#).

**Examples**

```
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"), excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
doCluster(wordF = wordOccuDF, myMethod = "ward.D2")
```

---

doKmeansClust

*Performs a k-means cluster analysis on the basis of the word-occurrence data.frame.*

---

**Description**

Performs a k-means cluster analysis on the basis of the word-occurrence data.frame using [kmeans](#) function.

## Usage

```
doKmeansClust(wordF, nbClust = 4, nbIter = 10,  
  algo = "Hartigan-Wong", getPlot = TRUE, mwidth = 800,  
  mheight = 800, formatType = "png", ...)
```

## Arguments

wordF	The data.frame containing word occurrences.
nbClust	The number of clusters.
nbIter	The number of iterations allowed.
algo	The algorithm used (see <a href="#">kmeans</a> ).
getPlot	If TRUE, save the k-means cluster plot in the RESULTS directory.
mwidth	The width of the plot in pixels.
mheight	The height of the plot in pixels.
formatType	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").
...	Additional arguments from the <a href="#">kmeans</a> function.

## Value

An object of class `kmeans` (see [kmeans](#)).

## Examples

```
data("loremIpsum")  
loremIpsum01 <- loremIpsum[1:100]  
loremIpsum02 <- loremIpsum[101:200]  
loremIpsum03 <- loremIpsum[201:300]  
loremIpsum04 <- loremIpsum[301:400]  
loremIpsum05 <- loremIpsum[401:500]  
subDir <- "RESULTS"  
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)  
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")  
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")  
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")  
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")  
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")  
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"), excludeSW = FALSE)  
file.remove(list.files(full.names = TRUE,  
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))  
doKmeansClust(wordF = wordOccuDF, nbClust = 2)
```

---

doMetacomEntropart      *Performs an analysis of ecological diversity and structure.*

---

### Description

Uses the [entropart-package](#) to analyse the word-occurrence data.frame, considering words as species and documents as communities.

### Usage

```
doMetacomEntropart(wordF, getPlot = c(TRUE, TRUE, TRUE, TRUE),
  getTextSink = c(TRUE, TRUE, TRUE, TRUE), mwidth = 800,
  mheight = 800, formatType = "png")
```

### Arguments

wordF	The data.frame containing word occurrences.
getPlot	A vector with four logical values. If <code>getPlot[1]==TRUE</code> , the <code>MetaCommunity</code> object is plotted and saved in the <code>RESULTS</code> directory. If <code>getPlot[2]==TRUE</code> , the <code>DivPart</code> analysis is plotted and saved in the <code>RESULTS</code> directory. If <code>getPlot[3]==TRUE</code> , the <code>DivEst</code> analysis is plotted and saved in the <code>RESULTS</code> directory. If <code>getPlot[4]==TRUE</code> , the <code>DivProfile</code> analysis is plotted and saved in the <code>RESULTS</code> directory.
getTextSink	A vector with four logical values. If <code>getTextSink[1]==TRUE</code> , the <code>MetaCommunity</code> object is saved in the <code>RESULTS</code> directory. If <code>getTextSink[2]==TRUE</code> , the <code>DivPart</code> analysis is saved in the <code>RESULTS</code> directory. If <code>getTextSink[3]==TRUE</code> , the <code>DivEst</code> analysis is saved in the <code>RESULTS</code> directory. If <code>getTextSink[4]==TRUE</code> , the <code>DivProfile</code> analysis is saved in the <code>RESULTS</code> directory.
mwidth	The width of the plot in pixels.
mheight	The height of the plot in pixels.
formatType	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").

### Value

A `MetaCommunity` object (see [entropart-package](#)).

### Examples

```
## Not run:
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:10]
loremIpsum02 <- loremIpsum[11:20]
loremIpsum03 <- loremIpsum[21:30]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
```

```
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"),
  excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
doMetacomEntropart(wordF = wordOccuDF)

## End(Not run)
```

---

doMetacomMetacom      *Performs a metacomunity analysis.*

---

## Description

Use the package [Metacomunity](#) to analyse the word-occurrence data.frame, considering words as species and documents as communities.

## Usage

```
doMetacomMetacom(wordF, numSim = 10, limit = "Inf", getPlot = TRUE,
  getTextSink = TRUE, mwidth = 800, mheight = 800,
  formatType = "png")
```

## Arguments

wordF	The data.frame containing word occurrences.
numSim	Number of simulated null matrices, see <a href="#">Metacomunity</a> .
limit	An integer to limit the number of words to use in the analysis.
getPlot	If TRUE, save the plot in the RESULTS directory.
getTextSink	If TRUE, save the console output in the RESULTS directory.
mwidth	The width of the plot in pixels.
mheight	The height of the plot in pixels.
formatType	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").

## Value

An object of class [Metacomunity](#).

## Examples

```
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
```

```

write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"),
  excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
doMetacomMetacom(wordF = wordOccuDF)

```

---

excludeStopWords	<i>Exclude StopWords form the word-occurrence data.frame.</i>
------------------	---

---

### Description

Exclude StopWords form the word occurrences data.frame. excludeStopWords uses parallel to perform parallel computation.

### Usage

```
excludeStopWords(wordF, lang = "English")
```

### Arguments

wordF	The data.frame containing word occurrences.
lang	The language used ("French", "English", "Spanish").

### Value

The word-occurrence data.frame.

### Examples

```

## Not run:
excludeStopWords(wordF = myDF, lang = "French")

## End(Not run)

```



---

`exclusionList_FR`      *Stop words in French.*

---

**Description**

A vector containing stop words in French.

**Usage**

`exclusionList_FR`

**Format**

A vector with 173 elements (character), with UTF-8 characters escaped using `stringi::stri_escape_unicode(exclusionList_FR)`.

**Source**

Adapted from <http://www.ranks.nl/stopwords/french>.

---

`exclusionList_SP`      *Stop words in Spanish.*

---

**Description**

A vector containing stop words in Spanish

**Usage**

`exclusionList_SP`

**Format**

A vector with 190 elements (character), with UTF-8 characters escaped using `stringi::stri_escape_unicode(exclusionList_SP)`.

**Source**

Adapted from <http://www.ranks.nl/stopwords/spanish>.

---

exclusionList_UK	<i>Stop words in English.</i>
------------------	-------------------------------

---

**Description**

A vector containing stop words in English.

**Usage**

```
exclusionList_UK
```

**Format**

A vector with 542 elements (character).

**Source**

Adapted from <http://www.ranks.nl/stopwords>.

---

getAllAnalysis	<i>A quick way to compute a set of analysis from the word-occurrence data.frame.</i>
----------------	--

---

**Description**

A quick way to compute a set of analysis from the word-occurrence data.frame.

**Usage**

```
getAllAnalysis(dataset, wcloud = TRUE, sumStats = TRUE, freqW = TRUE,
  corA = TRUE, clust = TRUE, metacom = TRUE)
```

**Arguments**

dataset	A single word-occurrence data.frame.
wcloud	A logical to for word cloud analysis.
sumStats	A logical to for summary statistics analysis.
freqW	A logical to for word frequency analysis.
corA	A logical to for correspondence analysis.
clust	A logical to for cluster analysis.
metacom	A logical to for metacommunity analysis.

**Value**

A set of analyses available from the `inpdfR` package.

## Examples

```
## Not run:
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"),
  excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
getAllAnalysis(dataset = wordOccuDF, wcloud = FALSE, sumStats = FALSE)

## End(Not run)
```

---

getListFiles

*List files in a specified directory sorted by extension.*

---

## Description

List files in a specified directory sorted by extension. The function takes into account .txt and .pdf files based on strsplit function.

## Usage

```
getListFiles(mywd)
```

## Arguments

mywd                    A string containing the working directory.

## Value

A list of length 2 with file names sorted by extension (pdf and txt).

## Examples

```
getListFiles(mywd = getwd())
```

---

getMostFreqWord	Returns most frequent words.
-----------------	------------------------------

---

## Description

Returns most frequent words and plots their frequencies per document.

## Usage

```
getMostFreqWord(wordF, numWords, getPlot = TRUE, mwidth = 1024,  
  mheight = 800, formatType = "png")
```

## Arguments

wordF	The data.frame containing word occurrences.
numWords	The number of words to be returned.
getPlot	If TRUE, save a scatter plot in the RESULTS directory.
mwidth	The width of the plot in pixels.
mheight	The height of the plot in pixels.
formatType	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").

## Value

The numWords most frequent words.

## Examples

```
data("loremIpsum")  
loremIpsum01 <- loremIpsum[1:100]  
loremIpsum02 <- loremIpsum[101:200]  
loremIpsum03 <- loremIpsum[201:300]  
loremIpsum04 <- loremIpsum[301:400]  
loremIpsum05 <- loremIpsum[401:500]  
subDir <- "RESULTS"  
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)  
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")  
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")  
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")  
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")  
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")  
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"),  
  excludeSW = FALSE)  
file.remove(list.files(full.names = TRUE,  
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))  
getMostFreqWord(wordF = wordOccuDF, numWords = 5)
```

---

getMostFreqWordCor      *Test for correlation between the most frequent words.*

---

### Description

Test for correlation between the most frequent words.

### Usage

```
getMostFreqWordCor(wordF, numWords, getPlot = c(TRUE, TRUE),
  getTextSink = TRUE, mwidth = 1024, mheight = 1024,
  formatType = "png")
```

### Arguments

wordF	The data.frame containing word occurrences.
numWords	The number of words to be returned.
getPlot	A vector with two logical values. If plots[1]==TRUE, an image of the correlation matrix is saved in the RESULTS directory. If plots[2]==TRUE, the image of the p-value matrix associated with the correlation is saved in the RESULTS directory.
getTextSink	If TRUE, save the correlation matrix and the associated p-values in a text file in the RESULTS directory.
mwidth	The width of the plot in pixels.
mheight	The height of the plot in pixels.
formatType	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").

### Value

A list with the correlation matrix and the p-value matrix.

### Examples

```
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"),
```

```

excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
getMostFreqWordCor(wordF = wordOccuDF, numWords = 5)

```

---

getPDF

*Extract text from PDF files and return a word-occurrence data.frame.*


---

### Description

getPDF returns a word-occurrence data.frame from PDF files. It needs XPDF in order to run (<http://www.foolabs.com/xpdf/download.html>) and uses parallel to perform parallel computation.

### Usage

```

getPDF(myPDFs, minword = 1, maxword = 20, minFreqWord = 1,
  pathToPdf totext = "")

```

### Arguments

myPDFs	A character vector containing PDF file names.
minword	An integer specifying the minimum number of letters per word into the returned data.frame.
maxword	An integer to specifying the maximum number of letters per word into the returned data.frame.
minFreqWord	An integer specifying the minimum word frequency into the returned data.frame.
pathToPdf totext	A character containing an alternative path to XPDF pdf totext function, see Details section.

### Details

getPDF uses XPDF pdf totext function to extract the content of PDF files into a TXT file. If pdf totext is not in the PATH, an alternative is to provide the full path of the program into the pathToPdf totext parameter.

### Value

A list of list with word-occurrence data.frame and file name.

### Examples

```

## Not run:
getPDF(myPDFs = "mypdf.pdf")

## End(Not run)

```

---

getStopWords	<i>Load a list of stopwords.</i>
--------------	----------------------------------

---

**Description**

getStopWords returns a list of stopwords.

**Usage**

```
getStopWords()
```

**Value**

A list of vectors with stopwords for French, English, and Spanish languages.

**Examples**

```
getStopWords()
```

---

getSummaryStatsBARPLOT
------------------------

---

*Perform a barplot with the number of unique words per document*

---

**Description**

Perform a barplot with the number of unique words per document using [barplot](#) function.

**Usage**

```
getSummaryStatsBARPLOT(wordF, getPlot = TRUE, mwidth = 480,  
  mheight = 480, formatType = "png", ...)
```

**Arguments**

wordF	The data.frame containing word occurrences.
getPlot	If TRUE, save the bar plot in the RESULTS directory.
mwidth	The width of the plot in pixels.
mheight	The height of the plot in pixels.
formatType	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").
...	Additional arguments from barplot function.

**Value**

The number of unique words per document.

**Examples**

```

data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"),
  excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
getSummaryStatsBARPLOT(wordF = wordOccuDF)

```

---

getSummaryStatsHISTO *Plot an histogram with the number of words excluding stop words*

---

**Description**

Plot a histogram with the number of words excluding stop words using `hist` function.

**Usage**

```

getSummaryStatsHISTO(wordF, mwidth = 800, mheight = 800,
  formatType = "png", ...)

```

**Arguments**

<code>wordF</code>	The data.frame containing word occurrences.
<code>mwidth</code>	The width of the plot in pixels.
<code>mheight</code>	The height of the plot in pixels.
<code>formatType</code>	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").
<code>...</code>	Additional arguments from <code>hist</code> function.

**Examples**

```

data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]

```



```

subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"),
  excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
getSummaryStatsHISTO(wordF = wordOccuDF)

```

---

getSummaryStatsOCCUR *Plot a scatter plot with the proportion of documents using similar words.*

---

### Description

Plot a scatter plot with the proportion of documents using similar words.

### Usage

```

getSummaryStatsOCCUR(wordF, getPlot = TRUE, mwidth = 800,
  mheight = 800, formatType = "png")

```

### Arguments

wordF	The data.frame containing word occurrences.
getPlot	If TRUE, save the scatter plot in the RESULTS directory.
mwidth	The width of the plot in pixels.
mheight	The height of the plot in pixels.
formatType	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").

### Value

A data.frame containing the proportion of documents and the number of similar words.

### Examples

```

## Not run:
getSummaryStatsOCCUR(wordF = myDF)

## End(Not run)

```

---

getTXT	<i>Extract text from TXT files and return a word-occurrence data.frame.</i>
--------	---

---

**Description**

Extract text from TXT files and return a word-occurrence data.frame.

**Usage**

```
getTXT(myTXTs)
```

**Arguments**

myTXTs            A character vector containing TXT file names (or complete path to these files).

**Value**

A list of list with word-occurrence data.frame and file name.

**Examples**

```
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuFreq <- getTXT(myTXTs = list.files(path = paste0(getwd(),
  "/RESULTS/"), pattern = "loremIpsum", full.names = TRUE))
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
```

---

getwordOccuDF	<i>A quick way to obtain the word-occurrence data.frame from a set of documents.</i>
---------------	--

---

**Description**

A quick way to obtain the word-occurrence data.frame from a set of documents.

**Usage**

```
getwordOccuDF(mywd, language = "English", excludeSW = TRUE)
```

**Arguments**

mywd                    A character variable containing the working directory.  
 language              The language used ("French", "English", "Spanish").  
 excludeSW             A logical to exclude stop words.

**Value**

A single word-occurrence data.frame.

**Examples**

```
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"),
  excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
```

---

getXFreqWord

*Returns most frequent words*

---

**Description**

Returns most frequent words

**Usage**

```
getXFreqWord(wordF, occuWords)
```

**Arguments**

wordF                    The data.frame containing word occurrences.  
 occuWords              The minimum number of occurrences for words to be returned.

**Value**

A vector with most frequent words.

**Examples**

```
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuDF <- getwordOccuDF(mywd = paste0(getwd(), "/RESULTS"),
  excludeSW = FALSE)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
getXFreqWord(wordF = wordOccuDF, occuWords = 5)
```

---

IdentifyStructure      *Copy of the identifyStructure function from Tad Dallas metacom package.*

---

**Description**

Identifies structure (or quasi-structure) and outputs a classification.

**Usage**

```
IdentifyStructure(metacom.obj)
```

**Arguments**

metacom.obj      The result of the ‘Metacomunity’ function, containing a list of 4 elements; the empirical matrix being tested, and results for coherence, turnover, and boundary clumping.

**Details**

Tad Dallas <tDallas@uga.edu> identifyStructure function no longer maintained in metacom package. see <https://github.com/taddallas/metacom>. This function was copy-pasted from version 1.4.4 of package metacom with minor modification (fix warning: the condition has length > 1 and only the first element will be used).

**Value**

Ouputs a classification of the metacommunity.

**Note**

Quasi structures, as well as 'random' and 'Gleasonian' structures, may not strictly be discernable through the EMS approach, as they rely on inferring a result from a non-significant test ('accepting the null'), which is typically a bad idea.

---

inpdfr

*inpdfr: A package to analyse PDF Files Using Ecological Tools.*

---

**Description**

The inpdfr package allows analysing and comparing PDF/TXT documents using both classical text mining tools and those from theoretical ecolgy. In the later, words are considered as species and documents as communities, therefore allowing analysis at the community and metacommunity levels. The inpdfr package provides three cathegories of functions: functions to extract and process text into a word-occurrence data.frame, functions to analyse the word-occurrence data.frame with standard and ecological tools, and functions to use inpdfr through a Gtk2 Graphical User Interface (GitHub version only).

---

loremIpsum

*Lorem Ipsum text.*

---

**Description**

A vector containing a Lorem Ipsum text for testing purposes.

**Usage**

```
loremIpsum
```

**Format**

A vector with 556 elements, each element corresponds to a line in the original text (character).

**Source**

<http://lipsum.com/>.

---

makeWordcloud	<i>Word cloud based on the word-occurrence data.frame.</i>
---------------	--

---

### Description

Plot a word cloud from the word-occurrence data.frame using `wordcloud` function.

### Usage

```
makeWordcloud(wordF, wcFormat = "png", wcminFreq = 3,  
              wcmxWords = Inf, wcRandOrder = FALSE,  
              wcCol = RColorBrewer::brewer.pal(8, "Dark2"), getPlot = c(TRUE,  
              TRUE), mwidth = 1000, mheight = 1000, formatType = "png")
```

### Arguments

<code>wordF</code>	The data.frame containing word occurrences.
<code>wcFormat</code>	Output format for the word cloud (deprecated, only "png").
<code>wcminFreq</code>	Minimum word frequency for words to be plotted (see <code>wordcloud</code> ).
<code>wcmxWords</code>	Maximum number of words to be plotted (see <code>wordcloud</code> ).
<code>wcRandOrder</code>	Plot words in random order (see <code>wordcloud</code> ).
<code>wcCol</code>	Color words (see <code>wordcloud</code> ).
<code>getPlot</code>	A vector with two logical values. If <code>plots[1]==TRUE</code> , a word cloud is made for each document. If <code>plots[2]==TRUE</code> , a word cloud is made for the combination of all documents.
<code>mwidth</code>	The width of the plot in pixels.
<code>mheight</code>	The height of the plot in pixels.
<code>formatType</code>	The format for the output file ("eps", "pdf", "png", "svg", "tiff", "jpeg", "bmp").

### Examples

```
## Not run:  
makeWordcloud(wordF = myDF)  
  
## End(Not run)
```

---

mergeWordFreq	<i>Merge word-occurrence data.frames into a single data.frame.</i>
---------------	--

---

## Description

Merge word-occurrence data.frames into a single data.frame.

## Usage

```
mergeWordFreq(wordF)
```

## Arguments

wordF            The data.frame containing word occurrences.

## Value

A single word-occurrence data.frame with each column corresponding to a text file.

## Examples

```
data("loremIpsum")
loremIpsum01 <- loremIpsum[1:100]
loremIpsum02 <- loremIpsum[101:200]
loremIpsum03 <- loremIpsum[201:300]
loremIpsum04 <- loremIpsum[301:400]
loremIpsum05 <- loremIpsum[401:500]
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum01, file = "RESULTS/loremIpsum01.txt")
write(x = loremIpsum02, file = "RESULTS/loremIpsum02.txt")
write(x = loremIpsum03, file = "RESULTS/loremIpsum03.txt")
write(x = loremIpsum04, file = "RESULTS/loremIpsum04.txt")
write(x = loremIpsum05, file = "RESULTS/loremIpsum05.txt")
wordOccuFreq <- getTXT(myTXTs = list.files(path = paste0(getwd(),
  "/RESULTS/"), pattern = "loremIpsum", full.names = TRUE))
wordOccuDF <- mergeWordFreq(wordF = wordOccuFreq)
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
```

---

postProcTxt	<i>Prossess vectors containing words into a data.frame of word occurrences.</i>
-------------	---

---

### Description

Prossess vectors containing words into a data.frame of word occurrences.

### Usage

```
postProcTxt(txt, minword = 1, maxword = 20, minFreqWord = 1)
```

### Arguments

txt	A vector containing text.
minword	An integer specifying the minimum number of letters per word into the returned data.frame.
maxword	An integer to specifying the maximum number of letters per word into the returned data.frame.
minFreqWord	An integer specifying the minimum word frequency into the returned data.frame.

### Value

A data.frame (freq = occurrences, stem = stem words, word = words), sorted by word occurrences.

### Examples

```
## Not run:
postProcTxt(txt = preProcTxt(filetxt = "loremIpsum.txt"))

## End(Not run)
data("loremIpsum")
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum, file = "RESULTS/loremIpsum.txt")
preProcTxt(filetxt = paste0(getwd(), "/RESULTS/loremIpsum.txt"))
postProcTxt(txt =
  preProcTxt(filetxt = paste0(getwd(), "/RESULTS/loremIpsum.txt")))
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
```



---

```
preProcTxt
```

*Extract text from txt files and pre-process content.*

---

**Description**

Extract text from txt files and pre-process content.

**Usage**

```
preProcTxt(filetxt, encodingIn = "UTF-8", encodingOut = "UTF-8")
```

**Arguments**

filetxt	A character containing the name of a txt file.
encodingIn	Encoding of the text file (default = "UTF-8").
encodingOut	Encoding of the text extracted (default = "UTF-8").

**Value**

A character vector with the content of the pre-process txt file (one element per line).

**Examples**

```
data("loremIpsum")
subDir <- "RESULTS"
dir.create(file.path(getwd(), subDir), showWarnings = FALSE)
write(x = loremIpsum, file = "RESULTS/loremIpsum.txt")
preProcTxt(filetxt = paste0(getwd(), "/RESULTS/loremIpsum.txt"))
file.remove(list.files(full.names = TRUE,
  path = paste0(getwd(), "/RESULTS"), pattern = "loremIpsum"))
```

---

```
quitSpaceFromChars
```

*Delete spaces in file names.*

---

**Description**

Delete spaces in file names located in the current working directory.

**Usage**

```
quitSpaceFromChars(vectxt)
```

**Arguments**

vectxt	A vector containing character entries corresponding to the names of files in the current working directory.
--------	---

**Value**

The function returns a logical for each file, with TRUE if the file has been found, and FALSE otherwise.

**Examples**

```
quitSpaceFromChars(c("my pdf.pdf", "my other pdf.pdf"))
```

---

truncNumWords	<i>Truncate the word-occurrence data.frame.</i>
---------------	---

---

**Description**

Truncate the word-occurrence data.frame.

**Usage**

```
truncNumWords(wordF, maxWords)
```

**Arguments**

wordF	The data.frame containing word occurrences.
maxWords	The maximum number of words in the data.frame.

**Value**

The data.frame containing word occurrences.

**Examples**

```
## Not run:  
truncNumWords(wordF = myWordOccurrenceDF, maxWords = 50)  
  
## End(Not run)
```

# Index

## \*Topic **datasets**

- exclusionList\_FR, 9
- exclusionList\_SP, 9
- exclusionList\_UK, 10
- loremIpsum, 21

barplot, 15

ca, 2, 3

dist, 4

doCA, 2

doCluster, 3

doKmeansClust, 4

doMetacomEntropart, 6

doMetacomMetacom, 7

excludeStopWords, 8

exclusionList\_FR, 9

exclusionList\_SP, 9

exclusionList\_UK, 10

getAllAnalysis, 10

getListFiles, 11

getMostFreqWord, 12

getMostFreqWordCor, 13

getPDF, 14

getStopWords, 15

getSummaryStatsBARPLOT, 15

getSummaryStatsHISTO, 16

getSummaryStatsOCCUR, 17

getTXT, 18

getwordOccuDF, 18

getXFreqWord, 19

hclust, 3, 4

hist, 16

IdentifyStructure, 20

inpdfr, 21

inpdfr-package (inpdfr), 21

kmeans, 4, 5

loremIpsum, 21

makeWordcloud, 22

mergeWordFreq, 23

Metacommunity, 7

postProcTxt, 24

preProcTxt, 25

quitSpaceFromChars, 25

truncNumWords, 26

wordcloud, 22