

# Package: pubmed.mineR (via r-universe)

May 26, 2026

**Type** Package

**Title** Text Mining of PubMed Abstracts

**Version** 1.0.21

**Date** 2024-09-06

**Maintainer** S. Ramachandran <ramuigib@gmail.com>

**Description** Text mining of PubMed Abstracts (text and XML) from  
<<https://pubmed.ncbi.nlm.nih.gov/>>.

**Depends** R (>= 3.5.0), methods

**Imports** RCurl, XML, boot, R2HTML, RJSONIO

**Collate** 'Abstracts-class.R' 'HGNC-class.R' 'Yearwise.R' 'Genewise.R'  
'combineabs.R' 'gene\_atomization.R' 'Find\_conclusion.R'  
'getabs.R' 'getabsT.R' 'gethgnc.R' 'ready.R' 'readabs.R'  
'removeabs.R' 'searchabsL.R' 'searchabsT.R' 'sendabs.R'  
'subabs.R' 'cleanabs.R' 'word\_atomizations.R' 'SentenceToken.R'  
'contextSearch.R' 'uniprotfun.R' 'local\_uniprotfun.R'  
'tdm\_for\_lsa.R' 'printabs.R' 'pubtator\_function.R'  
'cos\_sim\_calc.R' 'cos\_sim\_calc\_boot.R' 'wordscluster.R'  
'whichcluster.R' 'wordsclusterview.R' 'find\_intro\_conc\_html.R'  
'cluster\_words.R' 'get\_original\_term.R' 'get\_original\_term2.R'  
'input\_for\_find\_intro\_conc\_html.R' 'xmlreadabs.R'  
'xmlword\_atomizations.R' 'xmlgene\_atomizations.R'  
'pubtator\_result\_list\_to\_table.R' 'genes\_BWI.R' 'BWI.R'  
'currentabs\_fn.R' 'previousabs\_fn.R' 'altnamesfun.R'  
'subsetabs.R' 'prevsymbol\_fn.R' 'alias\_fn.R' 'get\_NMids.R'  
'get\_PMCIDS.R' 'get\_PMCtable.R' 'get\_Sequences.R'  
'Give\_Sentences\_PMC.R' 'head\_abbrev.R' 'names\_fn.R'  
'official\_fn.R' 'pmids\_to\_abstracts.R' 'get\_gene\_sentences.R'  
'Give\_Sentences.R' 'get\_MedlinePlus.R' 'co\_occurrence\_fn.R'  
'space\_quasher.R' 'readabsnew.R' 'word\_associations.R'  
'get\_DOIs.R' 'additional\_info.R' 'new\_xmlreadabs.R'  
'pubtator\_function\_JSON.R' 'xmlgene\_atomizations\_new.R'  
'co\_occurrence\_advance.R' 'pubtator3\_function.R'

**License** GPL-3

**LazyLoad** true  
**LazyData** true  
**NeedsCompilation** no  
**Author** Jyoti Rani [aut], S.Ramachandran [aut], Ab Rauf Shah [aut], S. Ramachandran [cre]  
**Config/pak/sysreqs** make libxml2-dev  
**Repository** <https://ramuigibnew.r-universe.dev>  
**Date/Publication** 2024-09-09 02:57:10 UTC  
**RemoteUrl** <https://github.com/cran/pubmed.mineR>  
**RemoteRef** HEAD  
**RemoteSha** fda200ef9d555102b83fcdcb260443f4549b8743

## Contents

Abstracts-class	4
additional_info	5
alias_fn	5
altnamesfun	6
BWI	7
cleanabs	8
cleanabs-methods	9
cluster_words	9
co_occurrence_advance	10
co_occurrence_fn	11
combineabs	12
combineabs-methods	12
common_words_new	13
contextSearch	13
contextSearch-methods	14
cos_sim_calc	14
cos_sim_calc_boot	15
currentabs_fn	16
Find_conclusion	17
find_intro_conc_html	18
gene_atomization	19
genes_BWI	20
GeneToEntrez	21
Genewise	21
Genewise-methods	22
get_DOIs	22
get_gene_sentences	23
get_MedlinePlus	24
get_NMids	24
get_original_term	25
get_original_term2	26

get_PMCIDS	27
get_PMCTable	27
get_Sequences	28
getabs	29
getabs-methods	29
getabsT	30
getabsT-methods	30
Give_Sentences	31
Give_Sentences_PMC	31
head_abbrev	32
HGNC-class	33
HGNC2UniprotID	34
HGNCdata	34
input_for_find_intro_conc_html	35
local_uniprotfun	36
names_fn	37
new_xmlreadabs	38
official_fn	39
pmids_to_abstracts	40
previousabs_fn	40
prevsymbol_fn	41
printabs	42
pubtator_function	43
pubtator_function_JSON	44
pubtator_result_list_to_table	45
pubtator3_function	46
readabs	47
readabsnew	47
ready	48
removeabs	49
removeabs-methods	49
searchabsL	50
searchabsL-methods	51
searchabsT	51
searchabsT-methods	52
sendabs	52
sendabs-methods	53
SentenceToken	54
space_quasher	54
subabs	55
subabs-methods	56
subsetabs	56
subsetabs-methods	57
tdm_for_lsa	57
uniprotfun	58
whichcluster	58
word_associations	59
word_atomizations	60

wordcluster . . . . .	61
wordclusterview . . . . .	62
xmlgene_atomizations . . . . .	63
xmlgene_atomizations_new . . . . .	64
xmlreadabs . . . . .	64
xmlword_atomizations . . . . .	65
Yearwise . . . . .	66
Yearwise-methods . . . . .	67
<b>Index</b>	<b>68</b>

---

Abstracts-class	<i>Class "Abstracts" Abstract Class</i>
-----------------	---

---

## Description

S4 Class with three slots Journal, Abstract, PMID to store abstracts from PubMed

## Objects from the Class

Objects can be created by calls of the form `new("Abstracts", ...)`.

## Slots

**Journal:** Object of class "character" to store Journals of the abstracts from PubMed

**Abstract:** Object of class "character" to store Abstracts from the PubMed

**PMID:** Object of class "numeric" to store PMIDs of abstracts from PubMed

## Methods

No methods defined with class "Abstracts" in the signature.

## Author(s)

S.Ramachandran, Ab Rauf Shah

## See Also

[searchabsL](#) [getabs](#) [contextSearch](#) [Genewise](#) [Yearwise](#) [combineabs](#) [subabs](#) [subsetabs](#) [readabs](#)

## Examples

```
showClass("Abstracts")
```

---

additional_info	<i>To extract sentences with multiple keywords from Abstracts</i>
-----------------	---

---

**Description**

additional\_info will help to extract the sentences containing multiple query term(s) from a large corpus of multiple abstracts.

**Usage**

```
additional_info(abs, pmid, keywords)
```

**Arguments**

abs	abs an S4 object of class Abstracts.
pmid	Vector of PMIDs from abstracts
keywords	Character Vector of Terms

**Value**

It will return a matrix object containing PMID, keywords and sentences

**Author(s)**

Surabhi Seth

**See Also**

[Give\\_Sentences](#)

**Examples**

```
## Not run: additional_info(abs = Abstract, pmid = "26564970", keywords = "text-mining" )
```

---

alias_fn	<i>To extract sentences containing Alias of the Human Genes from Pubmed abstracts.</i>
----------	--

---

**Description**

alias\_fn This function returns the sentences containing alias of gene and the user given terms from the Abstracts using HGNC gee data table. In this sense this function is a 2 Dimensional search.

**Usage**

```
alias_fn(genes, data, abs, filename, terms)
```

**Arguments**

genes	genes a table containing genes (official symbol,first column) with its frequency of occurrence (second column)could be an output of gene_atomization function and subsequently subsetting the table using for example the code genes_table = subset(t2diababs_genes, select = c("Gene_symbol","Freq")). Alternatively, a custome gene table can be supplied with two columns, the first one being the column for Gene symbols and the second one being the Frequency of occurrence. If Frequency of occurrence is not available then a dummy value of 1 can be set.
data	data is HGNC data table with all 49 features (columns) available from the web site <a href="https://www.genenames.org/">https://www.genenames.org/</a>
abs	abs an S4 object of class Abstracts.
filename	filename specifies the name of output file. Please note that the term alias will be suffixed to the given filename.
terms	terms query term(s) to be search in the abstracts, could be a vector of terms.

**Value**

An output file containing sentences with aliases of genes.For convenience both the official symbol and the corresponding alias are written in the output. The PMID of the corresponding Abstract containing the extracted sentence also appears just before the sentence. Note that multiple sentences from different abstracts are clubbed together under one gene alias that appears in those sentences.

**Author(s)**

S.Ramachandran

**See Also**

[prevsymbol\\_fn](#)

**Examples**

```
## Not run: alias_fn(genes,data,myabs,"nephro_",c("diabetic nephropathy","kidney disease"))
## genes output of gene_atomization()
```

---

altnamesfun

*To Get Alternative names of Genes*

---

**Description**

This function is used to retrieve the Alternative names of genes from UniProt using HGNC gene symbol.

**Usage**

```
altnamesfun(m)
```

**Arguments**

`m` is a character vector of HGNC official gene symbols.

**Value**

It returns a list of alternative names of given Gene symbols.

**Author(s)**

S.Ramachandran

**References**

UniProt Consortium. "The universal protein resource (UniProt)." Nucleic acids research 36.suppl 1 (2008): D190-D195. <http://www.uniprot.org/>

**See Also**

[uniprotfun](#), ~~~

**Examples**

```
## Not run: test = altnamesfun(c("ADIPOQ", "BDNF"))
## here "ADIPOQ" is the HGNC gene symbol for which alternative name(s) is required.
```

---

BWI

*To obtain the Buzz Word Index of terms from the Abstracts.*

---

**Description**

This function is used to obtain the Buzz word index value for the terms.

**Usage**

```
BWI(current, previous, n, N)
```

**Arguments**

`current` current an S4 object containing the Abstracts for the current year we require the BWI an output from `currentabs_fn()`

`previous` previous an S4 object containing the Abstracts for years previous to current year of study an output from `previousabs_fn()`.

`n` `n` is a character term for which Buzz Word Index is to be calculated.

`N` `N` is a character value specifying the theme from the large corpus.

**Value**

It returns a list containing BWI value for the given word.

**Author(s)**

S.Ramachandran

**References**

Jensen, Lars Juhl, Jasmin Saric, and Peer Bork. "Literature mining for the biologist: from information retrieval to biological discovery." *Nature reviews genetics* 7.2 (2006): 119-129.

**See Also**

[genes\\_BWI](#)

**Examples**

```
## Not run: result = BWI(mycurrentabs, mypreviousabs, "insulin", "inflammation")
## BWI for the term "insulin" and the theme is inflammation.
## Note that in the previous, years are starting one before the current year 2015;
## current is an S4 object containing the output from currentabs_fn()
## previous is an S4 object containing the output from previousabs_fn().
## 'n' and 'N' are query terms whose BWI is sought and the theme respectively
```

---

cleanabs

*To clean the result of searchabsL*

---

**Description**

It will remove the 'NONE' abstracts from the result of searchabsL.

**Usage**

```
cleanabs(object)
```

**Arguments**

object            an S4 object of class Abstracts.

**Value**

an S4 object of class Abstracts.

**Author(s)**

Jyoti Rani

**See Also**[searchabsL](#)**Examples**

```
## Not run: test1 = searchabsL(abs, include=c("term1", "term2"));
test2 = cleanabs(test1)
## End(Not run)
## here 'abs' is an S4 object of class Abstracts
## 'term1', 'term2' are the searchterms
## test1 is an S4 object containing abstracts for given terms
## and test2 is an S4 object of class Abstracts containing clean abstracts of searchabsL
```

---

cleanabs-methods	<i>Methods for Function cleanabs</i>
------------------	--------------------------------------

---

**Description**

To clean 'NONE' part of searchabsL output.

**Methods**

signature(object = "Abstracts") From an S4 object of class 'Abstracts' the cleanabs function is able to clean the output of searchabsL by removing the 'NONE' part of resultant abstracts.

---

cluster_words	<i>To Find the highest frequency of words within clusters</i>
---------------	---

---

**Description**

Function for finding the word (term) of highest frequency within clusters.

**Usage**

```
cluster_words(wordscluster, n)
```

**Arguments**

wordscluster	an R object containing the output of wordscluster()
n	a numeric vector containing cluster numbers

**Value**

a list containing cluster and its highest frequency word

**Author(s)**

S. Ramachandran

**See Also**

[wordscluster](#)

**Examples**

```
## Not run: test = cluster_words(wordscluster, 5)
## wordscluster is an R object of wordscluster
## 5 is number of cluster
## End(Not run)
```

---

co\_occurrence\_advance *Extracts multiple sentence with co-occurrence of two sets of terms*

---

**Description**

Extracts single or multiple sentences with co-occurrence of given terms

**Usage**

```
co_occurrence_advance(abstract, term1, term2, n)
```

**Arguments**

abstract	an S4 object of class Abstracts
term1	a character vector of terms
term2	a character vector of terms
n	A numeric value, which can be 0,1,2.

**Details**

Sentences with co-occurrence of two terms will be extracted along with the corresponding PMIDs. The output will be a data frame. In regard to the argument n, when the value is 0 then the co-occurrence is sought in the same sentence. When the value is 1, then the co-occurrence is sought in two consecutive sentences, namely, first term in the first sentence and second term in the next sentence. When the value is 2, then the co-occurrence is sought in two sentences separated by a sentence without either term1 or term2.

**Value**

It will return a data frame object containing PMID,sentences and the terms pairs.

**Author(s)**

Shashwat Badoni Surabhi Seth

**See Also**[co\\_occurrence\\_fn](#)**Examples**

```
## Not run: co_occurrence_advance(myabs,"resistance", c("genes","genetic"), 2
```

---

co_occurrence_fn	<i>Extracts sentences with co-occurrence of two sets of terms</i>
------------------	---

---

**Description**

co\_occurrence\_fn will automatically extract sentences with co-occurrence of two sets of terms.

**Usage**

```
co_occurrence_fn(terms1, abs, filename, terms2)
```

**Arguments**

terms1	a character vector of terms.
abs	an S4 object of class Abstracts
filename	a single character, filename
terms2	a character vector of terms.

**Details**

Sentences with co-occurrence of two terms will be extracted along with the corresponding PMIDs. The data will be written in a text file with the user given filename and the word co\_occurrence will be suffixed to it.

**Value**

A text file.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: co_occurrence_fn("resistance",myabs,"resistance_genetic",c("genes","genetic"))  
##
```

---

combineabs	<i>To combine the abstracts</i>
------------	---------------------------------

---

**Description**

combineabs will automatically combine two abstracts of two objects.

**Usage**

```
combineabs(object1, object2)
```

**Arguments**

object1	An S4 object of class Abstracts
object2	An S4 object of class Abstracts

**Details**

Two objects of class 'Abstracts' are combined to return non-redundant combined abstracts. It can be used sequentially to combine many objects of class 'Abstracts'. It will also write the number of combined abstracts into a text file named "data\_out.txt"

**Value**

An R object containing the combined abstracts, and a text file named "data\_out.txt" containing the number of abstracts combined together

**Author(s)**

S.Ramachandran, Jyoti Rani

**Examples**

```
## Not run: res1 = combineabs(x,y)
## here 'x', 'y' are the S4 objects of class 'Abstracts'.
```

---

combineabs-methods	<i>Abstracts Method to Combine Abstracts</i>
--------------------	--

---

**Description**

combineabs method to combine the abstracts. object1 and object2 are from Abstracts class.

**Methods**

```
signature(object1 = "Abstracts") An S4 object of class "Abstracts"
signature(object2 = "Abstracts") An S4 object of class "Abstracts"
```

---

common_words_new	<i>R Data containing words which frequently in text</i>
------------------	---

---

**Description**

This dataset is used to remove common words from the abstracts. This step is used for size reduction for further data mining.

**Usage**

```
data(common_words_new)
```

**Format**

The format is: chr "common\_words\_new"

**Details**

The dataset containing common words used to remove them from the text for size reduction.

**References**

[https://en.wikipedia.org/wiki/Most\\_common\\_words\\_in\\_English](https://en.wikipedia.org/wiki/Most_common_words_in_English)

**Examples**

```
data(common_words_new)
```

---

contextSearch	<i>For Context Search</i>
---------------	---------------------------

---

**Description**

contextSearch is a method to extract the sentences containing a given query term

**Usage**

```
contextSearch(object, y)
```

**Arguments**

object	An S4 object of Class Abstracts containing text abstracts
y	a character vector of term(s)

**Details**

It takes object of class Abstracts and query term(s) as arguments and returns a text and latex file of the sentences containing query term. The latex file can be further converted into PDF by using the system command in R i.e. `system("pdflatex filename.tex")`. pdflatex is a shell command in Linux to convert the latex file into PDF. In the pdf file the terms are written in bold face type to enable ease of reading

**Value**

`contextSearch()` will write two files one is a text file named "companion.txt", and other is a Latex file. If the single term is given in query then file name comes with the term name. If multiple terms are used then the file name will be "combined.tex"

**Author(s)**

Dr.S.Ramachandran, Jyoti Rani

**Examples**

```
## Not run: contextSearch(x, "diabetes")
## here 'x' is S4 object of class 'Abstracts', and query term is 'diabetes'.
```

---

`contextSearch-methods` *Method for Context Search*

---

**Description**

`contextSearch` will search the sentence for the given term(s).

**Methods**

`signature(object = "Abstracts")` The object from where it will search should be an S4 object of class Abstracts

---

`cos_sim_calc` *To calculate the cosine similarity between terms.*

---

**Description**

`cos_sim_calc` calculates the cosine measure of similarity between pairs of terms from a corpus.

**Usage**

```
cos_sim_calc(nummatrix)
```

**Arguments**

nummatrix      A numerical matrix for e.g. a Term Document matrix (output from tdm\_for\_lsa)

**Details**

The term document matrix is taken as input and cosine measures of similarity between all pairs of terms are calculated.

**Value**

A tab delimited text file containing the similarity values between all pairs of terms.

**Note**

This file can be input to cytoscape directly.

**Author(s)**

S. Ramachandran

**References**

[https://en.wikipedia.org/wiki/Cosine\\_similarity](https://en.wikipedia.org/wiki/Cosine_similarity)

**See Also**

[tdm\\_for\\_lsa](#)

**Examples**

```
## Not run: x = cos_sim_calc(nummatrix)
## here nummatrix is the 'Term Document Matrix' generated from tdm_for_lsa()
```

---

cos\_sim\_calc\_boot      *Cosine Similarity Calculation by Boot Strapping*

---

**Description**

cos\_sim\_calc\_boot allows boot strap analysis. This function should be used as argument for 'statistic' in the boot function of 'boot' package.

**Usage**

```
cos_sim_calc_boot(data, indices)
```

**Arguments**

data	Term Document Matrix generated from <code>tdm_for_lsa</code> function of this package. In this matrix, rows are terms and columns are abstracts.
indices	index of matrix.

**Details**

while calling this function we need to transpose the input `tdm` and can also set the number of replicates. `boot` package is required to call this function.

**Value**

It will return a matrix containing the cosine similarity of pairs of terms in the abstracts. This object is in same format as returned by the `'boot'` function of `'boot'` package.

**Author(s)**

Dr.S.Ramachandran

**See Also**

[tdm\\_for\\_lsa](#)

**Examples**

```
## Not run: test_boot = boot(data = t(nummatrix), statistic = cos_sim_calc_boot, R = 2)
## here 'nummatrix' is a Term Document Matrix, boot inbuilt function of boot package,
## R is number of replicates here it is 2. User can extend this number.
```

---

currentabs\_fn

*To Retrieve the Abstracts for year.*

---

**Description**

This function is used to extract the abstracts for year we want to study. Its output is used as input in other functions like `BWI()` and `genes_BWI()`

**Usage**

```
currentabs_fn(yr_to_include, theme, parentabs)
```

**Arguments**

yr_to_include	yr_to_include is the year for which we want to extract the Abstracts.
theme	theme is a character value specifying the themes for the Abstracts.
parentabs	parentabs an S4 object containing the Abstracts.

**Value**

It returns an S4 object containing the abstracts of the given year.

**Author(s)**

S.Ramachandran

**See Also**

[previousabs\\_fn](#)

**Examples**

```
## Not run: test = currentabs_fn("2015", "atherosclerosis", diabetesabs)
## here "2015" is the year for which, we wish to extract the abstracts on theme"Atherosclerosis"
## from the large corpus of diabetes i.e. diabetesabs.
```

---

Find_conclusion	<i>To find the conclusion from the abstract(s).</i>
-----------------	---

---

**Description**

This function is designed for the user convenience, so that user can get the conclusion from the abstract(s) without reading the whole abstract(s).

**Usage**

```
Find_conclusion(y)
```

**Arguments**

y                    An S4 object of class 'Abstract'.

**Value**

A list containing conclusions of given abstract(s)

**Author(s)**

S.Ramachandran, Jyoti Rani

**Examples**

```
## Not run: res1 = Find_conclusion(y)
## here 'y' is an S4 object of class Abstract.
```

---

find\_intro\_conc\_html *To find the introduction and conclusion from the abstracts.*

---

### Description

it helps to fetch the introduction and conclusion part from the abstracts.

### Usage

```
find_intro_conc_html(y, themes, all)
```

### Arguments

y	and S4 object of class Abstracts
themes	a character vector containing terms to be search in the abstracts
all	is logical, if true, will include title and author otherwise only abstracts will be considered.

### Details

find\_intro\_conc\_html provides an HTML file containing space separated introduction and conclusion part from the abstracts of given query term as well as gives a link directly to PubMed for the resulting PMID.

### Value

an HTML file.

### Author(s)

S.Ramachandran, Jyoti Rani

### See Also

[input\\_for\\_find\\_intro\\_conc\\_html](#)

### Examples

```
## Not run: test = find_intro_conc_html(abs, "diet", all=FALSE)
## here 'abs' is an S4 object of class Abstracts
## and 'diet' is a term to be search from the abstracts
## this function works for small size of corpus, say about 30-40 abstracts
```

---

gene_atomization	<i>To Extract Genes from the Abstracts</i>
------------------	--

---

**Description**

gene\_atomization will automatically fetch the genes (HGNC approved Symbol) from the text and report their frequencies. presently only HGNC approved symbols are used.

**Usage**

```
gene_atomization(m)
```

**Arguments**

m	An S4 object of class Abstracts
---	---------------------------------

**Details**

The function writes a text file with file name "data\_table.txt". The function gene\_atomization() is used to obtain the name of genes along with their frequencies of occurrence.

**Value**

A tab delimited table containing gene name and their frequencies of occurrence.

**Author(s)**

S.Ramachandran, Jyoti Rani

**Examples**

```
## Not run: gene_atomization(myabs)
## here myabs is an S4 object of class 'Abstracts' containing the abstracts
## uses older version of HGNC data (https://www.genenames.org/) by default.
## users may also use other functions such as official_fn and related
## family of functions for deeper data mining.
```

---

`genes_BWI`*Function to obtain the Buzz Word Index of Genes from the abstracts.*

---

**Description**

This function provides the Buzz word index for each gene. The theme is the context in which the gene is studied for e.g. atherosclerosis. Using this function user can identify abstracts with emphasis on a given gene.

**Usage**

```
genes_BWI(currentabs, previousabs, theme, genes)
```

**Arguments**

<code>currentabs</code>	<code>currentabs</code> an S4 object containing the Abstracts for the year we want to study. Output from <code>currentabs_fn()</code>
<code>previousabs</code>	<code>previousabs</code> an S4 object containing the Abstracts for years previous than our year of study. Output from <code>previousabs_fn()</code> .
<code>theme</code>	<code>theme</code> a character value to categorize our search. For e.g. 'Atherosclerosis' from 'diabetes' Abstracts.
<code>genes</code>	<code>genes</code> list of genes.

**Value**

It returns a dataframe containig Genes with their corresponding BWI values.

**Author(s)**

S.Ramachandran

**See Also**

[BWI](#)

**Examples**

```
## Not run: test = genes_BWI(currentabs, previousabs, theme, genes)
## currentabs is an S4 object contaning the Abstracts for the year we want to study.
## previousabs is an S4 object contaning the Abstracts for the years previous
## than our query year for e.g. before 2015
## theme is a character value specifying the search.
## genes is a character vector of gene symbols.
```

---

GeneToEntrez	<i>Data containing Entrez Ids</i>
--------------	-----------------------------------

---

**Description**

This dataset is used in DAVID\_info function of the package, and it contains the Entrez Ids for the respective genes and these Entrez Ids will be used to get information about human genes.

**Usage**

```
data(GeneToEntrez)
```

**Format**

The format is: chr "GeneToEntrez"

**Examples**

```
data(GeneToEntrez)
```

---

Genewise	<i>To Search the number of abstracts for Genes</i>
----------	--

---

**Description**

Genewise reports the number of abstracts for given gene(s) name(s)

**Usage**

```
Genewise(object, gene)
```

**Arguments**

object	An S4 object of class Abstracts
gene	a character input of gene name(HGNC approved symbol)

**Details**

This function will report the number of abstracts containing the query gene term(s) [HGNC approved symbols], and the result is saved in a text file "dataout.txt". Genewise() will report numbers of abstracts only. The abstracts themselves for corresponding gene names can be obtained using searchabsL() and searchabsT.

**Value**

Genewise will return an R object containing the abstracts for given gene, and a text file named "dataout.txt" containing the number of abstracts

**Author(s)**

S. Ramachandran, Jyoti Rani

**Examples**

```
## Not run: Genewise(x, "TLR4")
## here 'x' contains the S4 object of Abstracts.
```

---

Genewise-methods	<i>method to find the abstracts for the given gene.</i>
------------------	---

---

**Description**

Genewise The method Genewise will automatically report the numbers of abstracts for a given gene. It will write the result in the text file named "dataout.txt"

**Methods**

signature(object = "Abstracts") This method will search in an S4 object, containing abstracts. It will write a text file named "dataout.txt", containing the number of abstracts for the query gene terms

---

get_DOIs	<i>function for extracting Digital Object Identifier (DOIs) of papers</i>
----------	---

---

**Description**

get\_DOIs is used to extract DOIs of papers.

**Usage**

```
get_DOIs(abs)
```

**Arguments**

abs                    An S4 object of class Abstracts

**Details**

get\_DOIs allow users to get DOIs for individual papers.

**Value**

It returns a list object containing DOIs. This is useful for further extraction of papers

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: test = get_DOIs(vitiligoabs)
##
```

---

get\_gene\_sentences      *To extract the sentences for genes from the corpus.*

---

**Description**

get\_gene\_sentences is used to extract the exact sentence in which query gene is discussed.

**Usage**

```
get_gene_sentences(genes, abs, filename)
```

**Arguments**

genes                  genes a character vector containing the gene symbols.  
abs                     abs an S4 object of class Abstracts  
filename                filename specifies the output file name.

**Value**

an output file containing the sentences for given gene.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: get_gene_sentences("RBP4", abstracts, "RBP4_sentence.txt")
```

---

get\_MedlinePlus      *To Get MedLinePlus Summary*

---

**Description**

This function is to get the summary from MedLinePlus.

**Usage**

```
get_MedlinePlus(x)
```

**Arguments**

x                    x is a character input of terms: for examples 'malaria', 'pneumonia', 'chronic diseases'

**Value**

It returns a HTML file with name result\_Medline\_plus.html to be opened with any browser

**Author(s)**

S.Ramachandran

**References**

www.medlineplus.gov, Conuel T. Finding answers in a beauty shop. NIH MedlinePlus: the magazine [Internet]. 2012 Fall [cited 2013 Feb 9]; 7(3):24-26. Available from: <https://medlineplus.gov/magazine/issues/fall12/article26.html>

**Examples**

```
## Not run: get_MedlinePlus("malaria")
```

---

get\_NMids              *To extract NM ids from NCBI.*

---

**Description**

get\_NMids is to fetch the NM ids from the NCBI for corresponding gene/s to further fetch the sequence of that gene/s.

**Usage**

```
get_NMids(x)
```

**Arguments**

x                            x an R object containing Locus IDs for genes from NCBI2R package.

**Value**

It returns a list object containing corresponding NM id from NCBI.

**Author(s)**

S.Ramachandran

**References**

<http://www.ncbi.nlm.nih.gov/gene>

**See Also**

[get\\_Sequences](#)

**Examples**

```
## Not run: getNMids("5950")  
## 5950 is Locus id of RBP4 gene.
```

---

`get_original_term`        *To get the original terms from the corpus. deprecated*

---

**Description**

`get_original_term` is used to get the exact term as it is present in corpus. This function is not recommended anymore.

**Usage**

```
get_original_term(m, n)
```

**Arguments**

m                            an S4 object of class Abstracts containing the corpus.  
n                            a list object output from the function `cluster_words`

**Value**

a list object containing the terms.

**Author(s)**

S.Ramachandran, Jyoti Rani

**See Also**

[wordscluster](#)

**Examples**

```
## Not run: test = get_original_term(abs, words)
## here abs is an S4 object of class Abstracts
## words is the output object of cluster_words()
```

---

get\_original\_term2      *To get the original terms from the corpus.*

---

**Description**

get\_original\_term2 is used to get the exact term as it is present in corpus. It takes one term at a time. For multiple terms we can use lapply.

**Usage**

```
get_original_term2(x, y)
```

**Arguments**

x	x is a character value specifying the query term.
y	y is an S4 object containing abstracts.

**Value**

It returns a list object containing accurate term.

**Author(s)**

Jyoti Rani, S.Ramachandran.

**See Also**

[get\\_original\\_term](#)

**Examples**

```
## Not run: test = get_original_term("hba1c", diababs)
## here it will return accurate formation of hba1c i.e. HbA1c from diababs.
```

---

get_PMCIDS	<i>To extract the PMC Ids of the abstracts.</i>
------------	---

---

**Description**

get\_PMCIDS is used to fetch the PMC Ids of the abstracts from the corpus.

**Usage**

```
get_PMCIDS(abs)
```

**Arguments**

abs                    absan S4 object of class Abstracts.

**Value**

It returns a list containing PMC Ids.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: get_PMCIDS(abstracts)
```

---

get_PMCTable	<i>To fetch the given PMC article tables. Deprecated</i>
--------------	--

---

**Description**

get\_PMCTable is used to extract the full text article by giving query PMC Id. Deprecated.

**Usage**

```
get_PMCTable(url)
```

**Arguments**

url                    url is url of query PMC Id.

**Value**

It will return a full text article.

**Author(s)**

S.Ramachandran

**References**

<http://www.ncbi.nlm.nih.gov/pmc/>

**See Also**

[get\\_PMCIDS](#)

**Examples**

```
## Not run: get_PMtable("http://www.ncbi.nlm.nih.gov/pmc/?term=4039032")
```

---

get_Sequences	<i>To extract the Gene sequence from the NCBI.</i>
---------------	--

---

**Description**

get\_Sequences is used to fetch the sequences of genes using NM ids.

**Usage**

```
get_Sequences(x, filename)
```

**Arguments**

x	NM Id of the sequence.
filename	filename specifies the name of output file.

**Value**

It will return a text file containing sequence.

**Author(s)**

S.Ramachandran

**See Also**

[get\\_NMids, ~~~](#)

**Examples**

```
## Not run: get_Sequences("NM_012238.4", "SIRT1")
```

---

getabs	<i>To get Abstracts for a given term.</i>
--------	---

---

**Description**

getabs will automatically fetch the abstracts containing the query term. A base function of the package pubmed.mineR.

**Usage**

```
getabs(object, x, y)
```

**Arguments**

object	An S4 object of class Abstracts
x	A character string for the term
y	logical, if TRUE, search will be case sensitive

**Details**

getabs() is used to find and extract the abstracts for any given term, from the large a large corpus of abstracts. It uses regexpr based search strategy.

**Value**

An S4 object of class 'Abstracts', containing the result abstracts for the given term.

**Author(s)**

Dr.S.Ramachandran

**Examples**

```
## Not run: getabs(x, "term")  
## x is an S4 obeject of class abstracts containing the abstracts.
```

---

getabs-methods	getabs <i>To Get abstracts for a term</i>
----------------	---

---

**Description**

getabs will search for the abstracts of a given term. It is case sensitive.

**Methods**

signature(object = "Abstracts") This method takes three arguments, first 'object' containing data to be search, 'x', the term to be search, 'y' is logical if set "YES" will consider the case of text.

---

getabsT	<i>To get Abstracts for a given term.</i>
---------	---

---

**Description**

getabsT will automatically fetch the abstracts containing the query term.

**Usage**

```
getabsT(object, x, y)
```

**Arguments**

object	An S4 object of class Abstracts
x	A character string for the term
y	is logical, if set TRUE, search will be case sensitive.

**Details**

getabsT() is similar to getabs(), but it performs more specific search.

**Value**

An object of class 'Abstracts', containing the resulted abstracts for term.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: getabsT(diabdata, "term")
```

---

getabsT-methods	<i>To Get Abstracts</i>
-----------------	-------------------------

---

**Description**

getabsT will automatically return the abstracts of a term from the data.

**Methods**

signature(object = "Abstracts") getabsT will search for the abstracts of a term in the data, and will automatically write the number of abstracts into a text file named "dataout.txt".

---

Give\_Sentences      *To extract sentences from the Abstracts*

---

**Description**

Give\_Sentences will help to extract the sentence containing query term/s from the large corpus.

**Usage**

```
Give_Sentences(m, abs)
```

**Arguments**

m	m a character term.
abs	abs an S4 object of class Abstracts.

**Value**

It will return a list object containing sentences

**Author(s)**

S.Ramachandran

**See Also**

[Give\\_Sentences\\_PMC](#)

**Examples**

```
## Not run: Give_Sentences("diabetes", Abstracts)
```

---

Give\_Sentences\_PMC      *To fetch the sentence from the PMC full text article*

---

**Description**

Give\_Sentences\_PMC is used to extract the sentences from the full text article of given PMC id/s.

**Usage**

```
Give_Sentences_PMC(PMCID, term)
```

**Arguments**

PMCID	PMCID represents the PMC Id from where we want to extract the sentence.
term	term represents the term contained in a sentence.

**Value**

It will return a list object containing the sentences for query term from the given article.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: Give_Sentences_PMC(PMC4039032, "atherosclerosis")
```

---

head\_abbrev

*To extract the abbreviated term.*

---

**Description**

head\_abbrev is used to find expansion for which abbreviation is used. It will help to find the falsely matching abbreviations from the abstracts.

**Usage**

```
head_abbrev(limits, term, pmid, abs)
```

**Arguments**

limits	limits specifies the limit up to which expansion should be displayed. Default is 50
term	term is the query term (abbreviation)
pmid	pmid describes the PMID
abs	absan S4 object of class Abstracts.

**Value**

It will return a list.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: head_abbrev(50, "AR", "16893912", myabs)
```

---

HGNC-class

*HGNC Class for package.*

---

### **Description**

"HGNC"

### **Objects from the Class**

Objects can be created by calls of the form `new("HGNC", ...)`.

### **Slots**

HGNCID: Object of class "character"

ApprovedSymbol: Object of class "character"

ApprovedName: Object of class "character"

Status: Object of class "character"

PreviousSymbols: Object of class "character"

Aliases: Object of class "character"

Chromosome: Object of class "character"

AccessionNumbers: Object of class "character"

RefSeqIDs: Object of class "character"

### **Author(s)**

Dr.S.Ramachandran, Ab Rauf Shah

### **See Also**

[Abstracts](#)

### **Examples**

```
showClass("HGNC")
```

---

HGNC2UniprotID	<i>R Data containing HGNC2UniprotID data mapping.</i>
----------------	---

---

**Description**

This dataset contains HGNC2UniprotID from Uniprot and is used in uniprotfn() function of this package, to get the information of a gene from the Uniprot.

**Usage**

```
data(HGNC2UniprotID)
```

**Format**

The format is: chr "HGNC2UniprotID"

**Details**

The dataset contains HGNC2UniprotID

**References**

UniProt Consortium. "The universal protein resource (UniProt)." Nucleic acids research 36.suppl 1 (2008): D190-D195. <http://www.uniprot.org/>

**Examples**

```
data(HGNC2UniprotID)
```

---

HGNCdata	<i>R Data containing HGNC data.</i>
----------	-------------------------------------

---

**Description**

This dataset contains data from Human Gene Nomenclature Committee i.e HGNC ID, HGNC approved symbol, approved name, gene synonyms, chromosome no., accession numbers and RefSeq ids.

**Usage**

```
data(HGNCdata)
```

**Format**

The format is: chr "HGNCdata"

### **Details**

The dataset contains HGNCdata

### **References**

Povey, Sue, et al. "The HUGO gene nomenclature committee (HGNC)." *Human genetics* 109.6 (2001): 678-680. <http://www.genenames.org/>

### **Examples**

```
data(HGNCdata)
```

---

```
input_for_find_intro_conc_html  
    fetch the abstracts using E-utilities.
```

---

### **Description**

it helps in searching and fetching the abstracts from E-utilities using PMIDs.

### **Usage**

```
input_for_find_intro_conc_html(y, all)
```

### **Arguments**

<code>y</code>	an S4 object of class Abstracts
<code>all</code>	is logical if true, will include title and author also.

### **Details**

it takes an S4 object as input and uses its PMIDs to fetch the abstracts from E-utilities. The output will be used as input for `find_intro_conc_html` as it contains neat data i.e. abstracts only.

### **Value**

a list containing abstracts and PMID

### **Author(s)**

S.Ramachandran, Jyoti Rani

### **References**

[literature/http://eutils.ncbi.nlm.nih.gov/](http://eutils.ncbi.nlm.nih.gov/)

**See Also**

[find\\_intro\\_conc\\_html](#)

**Examples**

```
## Not run: test=input_for_find_intro_conc_html(abs)
## here 'abs' is an S4 object of class Abstracts.
```

---

local\_uniprotfun      *To Get Information from Uniprot.*

---

**Description**

It is an auxiliary function for altnamesfun.

**Usage**

```
local_uniprotfun(y)
```

**Arguments**

y                      y a character value containing HGNC Gene symbol

**Value**

It writes an output file named "x.txt" which will be used as input in altnamesfun().

**Author(s)**

S.Ramachandran, Jyoti Rani

**See Also**

[uniprotfun](#)

**Examples**

```
## Not run: local_uniprotfun("TLR4")
## here it will generate an output file named "x.txt" containing
## result for TLR4.
```

---

names_fn	<i>To extract the sentences in asbtracts containing gene names from HGNC.</i>
----------	---

---

**Description**

names\_fn matches the gene symbols to gene names and extract from HGNC.

**Usage**

```
names_fn(genes, data, abs, filename, terms)
```

**Arguments**

genes	genes is output of gene_atomization or a table containing HGNC gene symbols in first column with its frequency in second column.
data	data is HGNC data table with all 49 features (columns) available from the web site <a href="https://www.genenames.org/">https://www.genenames.org/</a>
abs	abs an S4 object of class Abstracts.
filename	filename specifies the name of output file.
terms	terms second query term to be searched in the same sentence (co-occurrence) of abstracts.

**Value**

It returns an output file containing genes with their corresponding gene names and sentences with co-occurrences if any.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run:  
names_fn(genes, data, diabetes_abs, "names", c("diabetic nephropathy", "DN"))  
  
## End(Not run)  
## genes output of gene_atomization()
```

---

new_xmlreadabs	<i>To read the abstracts from the PubMed saved in XML format.</i>
----------------	---

---

### Description

new\_xmlreadabs is modified form of xmlreadabs as it reads the abstracts downloaded or saved in XML format from PubMed. This function should be used for recent XML format from PubMed.

### Usage

```
new_xmlreadabs(file)
```

### Arguments

file            an XML file saved from PubMed.

### Value

an S4 object of class Abstracts containing journals, abstracts and PMID.

### Note

This function is useful with recent format of XML files from PubMed. The older xmlreadabs will not work with recent format.

### Author(s)

S.Ramachandran

### See Also

[readabsnew](#) [readabs](#)

### Examples

```
## Not run: xmlabs = new_xmlreadabs("easyPubMed_00001.txt")  
## here "easyPubMed_00001.txt" is an xml file from PubMed using package easyPubMed
```

---

official_fn	<i>To extract the sentences containing official gene symbol from abstracts.</i>
-------------	---

---

### Description

official\_fn is used to fetch the sentences containing official gene symbol from HGNC.

### Usage

```
official_fn(genes, abs, filename, terms)
```

### Arguments

genes	genes is output of gene_atomization, or a table containing HGNC gene symbols in first column with its frequency in second column.
abs	abs an S4 object of class Abstracts.
filename	filename specifies the name of output file.
terms	terms second query term to be searched in the same sentence (co-occurrence) of abstracts.

### Value

It will return a text file containing corresponding official gene symbol.

### Author(s)

S.Ramachandran

### Examples

```
## Not run:  
official_fn(genes, diabetes_abs, "genes", c("diabetic nephropathy", "DN"))  
  
## End(Not run)  
## genes output of gene_atomization()
```

---

pmids\_to\_abstracts      *To Find and match the PMIDs to the abstracts.*

---

**Description**

pmids\_to\_abstracts is used to extract the abstract/s of query PMID/s.

**Usage**

```
pmids_to_abstracts(x, abs)
```

**Arguments**

x	x a numeric vector containing PMIDs
abs	abs an S4 object of class Abstracts.

**Value**

It will return an S4 object of class abstracts containing abstracts for query PMIDs.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: pmids_to_abstracts(26878666,abs)
```

---

previousabs\_fn      *To Retrieve the Abstracts from the large corpus for given years.*

---

**Description**

This function is used to extract the abstracts from the large corpus excluding the years and under a given theme. Its output is used in other functions like BWI and genes\_BWI

**Usage**

```
previousabs_fn(yrs_to_exclude, theme, parentabs)
```

**Arguments**

yrs_to_exclude	yrs_to_exclude is abstracts for the list of years we want to exclude from the corpus
theme	theme is a character value specifying the themes for the Abstracts.
parentabs	parentabs an S4 object containing the Abstracts.

**Value**

It returns an S4 object containing the abstracts of the given year.

**Author(s)**

S.Ramachandran

**See Also**

[currentabs\\_fn](#)

**Examples**

```
## Not run: test = previousabs_fn(as.character(2015:2010), "atherosclerosis", diabetesabs
## here we will get the abstracts before 2010 for 'atherosclerosis'
## from the large corpus diabetesabs.
```

---

prevsymbol_fn	<i>To extract the sentences containing Previous symbols of HGNC genes.</i>
---------------	--

---

**Description**

prevsymbol\_fn will return the sentences containing previous symbols of the genes from the abstracts using HGNC data.

**Usage**

```
prevsymbol_fn(genes, data, abs, filename, terms)
```

**Arguments**

genes	genes is output of gene_atomization, or a table containing HGNC gene symbols in first column with its frequency in second column.
data	data is HGNC data table with all 49 features (columns) available from the web site <a href="https://www.genenames.org/">https://www.genenames.org/</a>
abs	abs an S4 object of class Abstracts.
filename	filename specify the name of output file
terms	terms second query term to be searched in the same sentence (co-occurrence) of abstracts.

**Value**

It returns a text file containing gene symbol with corresponding previous symbols.

**Author(s)**

S.Ramachandran

**See Also**

[names\\_fn](#), [official\\_fn](#)

**Examples**

```
## Not run:  
prevsymbol_fn(genes, data, diabetes_abs, "prevsym", c("diabetic nephropathy", "DN"))  
  
## End(Not run)
```

---

printabs	<i>To print the total number of abstracts in an S4 object of class Abstracts, its start and end</i>
----------	---

---

**Description**

It gives overview of the abstracts in an S4 object of class Abstracts.

**Usage**

```
printabs(object)
```

**Arguments**

object            An S4 object of class Abstracts.

**Value**

prints the total number of abstracts in an S4 object with additional information.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: printabs(myabs)  
## here myabs is an S4 object of class Abstracts.
```

---

pubtator\_function      *function for text annotation using PubTator.Deprecated.*

---

### Description

pubtator\_function is used to extract specific information from an abstract like Gene, chemical, and diseases etc. Deprecated.

### Usage

```
pubtator_function(x)
```

### Arguments

x                      numeric value PMID.

### Details

pubtator\_function allow users to get information about 'Gene', 'Chemical' and 'Disease' for given PMID. It uses online tool PubTator on R platform. It also removes redundancy from the output. It takes one PMID at once, for multiple PMIDs user can use lapply() function.

### Value

It returns a list object containing Gene, Chemical, Disease and PMID. The corresponding concept id numbers are joined by a '>' character. This is useful for further data mining

### Author(s)

S.Ramachandran, Jyoti Rani

### References

Wei CH et. al., PubTator: a Web-based text mining tool for assisting Biocuration, Nucleic acids research, 2013, 41 (W1): W518-W522. doi: 10.1093/nar/gkt44

Wei CH et. al., Accelerating literature curation with text-mining tools: a case study of using PubTator to curate genes in PubMed abstracts, Database (Oxford), bas041, 2012

Wei CH et. al., PubTator: A PubMed-like interactive curation system for document triage and literature curation, in Proceedings of BioCreative 2012 workshop, Washington DC, USA, 145-150, 2012

### Examples

```
## Not run: test = pubtator_function(17922911)
## here pubtator_function() will extract the information from this given pmid.
```

pubtator\_function\_JSON

*function for text annotation using PubTator*

---

### **Description**

pubtator\_function is used to extract specific information from an abstract like Gene, chemical, and diseases etc.

### **Usage**

```
pubtator_function_JSON(x)
```

### **Arguments**

x                      numeric value PMID.

### **Details**

pubtator\_function\_JSON allow users to get information about 'Gene', 'Chemical' and 'Disease' for given PMID. It uses online tool PubTator on R platform. It also removes redundancy from the output. It takes one PMID at once, for multiple PMIDs user can use lapply() function.

### **Value**

It returns a list object containing Gene, Chemical, Disease and PMID. The corresponding concept id numbers are joined by a '>' character. This is useful for further data mining

### **Author(s)**

S.Ramachandran, Jyoti Rani

### **References**

Wei CH et. al., PubTator: a Web-based text mining tool for assisting Biocuration, Nucleic acids research, 2013, 41 (W1): W518-W522. doi: 10.1093/nar/gkt44

Wei CH et. al., Accelerating literature curation with text-mining tools: a case study of using PubTator to curate genes in PubMed abstracts, Database (Oxford), bas041, 2012

Wei CH et. al., PubTator: A PubMed-like interactive curation system for document triage and literature curation, in Proceedings of BioCreative 2012 workshop, Washington DC, USA, 145-150, 2012

### **See Also**

pubtator\_function()

**Examples**

```
## Not run: test = pubtator_function_JSON(17922911)
## here pubtator_function_JSON() will extract the information from
## this given pmid.
```

---

pubtator\_result\_list\_to\_table

*Function to Convert Pubtator result from list into Table*

---

**Description**

This function is used to collect the outputs of pubtator\_function() after using lapply over multiple PMIDs. This function enables to convert it into table for easy reading and further analysis.

**Usage**

```
pubtator_result_list_to_table(x)
```

**Arguments**

x                    here x is list output of pubtator\_function().

**Value**

It returns table for pubtator\_function output.

**Author(s)**

S.Ramachandran, Jyoti Rani

**See Also**

[pubtator\\_function](#)

**Examples**

```
## Not run: test = pubtator_result_list_to_table(x)
##here x is the output of pubtator_function
```

---

pubtator3\_function     *function for text annotation using PubTator*

---

### Description

pubtator\_function is used to extract specific information from an abstract like Gene, chemical, and diseases etc.

### Usage

```
pubtator3_function(x)
```

### Arguments

x                      numeric value PMID.

### Details

pubtator\_function allow users to get information about 'Gene', 'Chemical' and 'Disease' for given PMID. It uses online tool PubTator on R platform. It also removes redundancy from the output. It takes one PMID at once, for multiple PMIDs user can use lapply() function.

### Value

It returns a list object containing Gene, Chemical, Disease and PMID. The corresponding concept id numbers are joined by a '>' character. This is useful for further data mining

### Author(s)

S.Ramachandran, Jyoti Rani

### References

Wei, Chih-Hsuan, et al. "PubTator 3.0: an AI-powered literature resource for unlocking biomedical knowledge." *Nucleic Acids Research* (2024): gkae235.

Wei CH et. al., PubTator: a Web-based text mining tool for assisting Biocuration, *Nucleic acids research*, 2013, 41 (W1): W518-W522. doi: 10.1093/nar/gkt44

Wei CH et. al., Accelerating literature curation with text-mining tools: a case study of using PubTator to curate genes in PubMed abstracts, *Database (Oxford)*, bas041, 2012

Wei CH et. al., PubTator: A PubMed-like interactive curation system for document triage and literature curation, in *Proceedings of BioCreative 2012 workshop*, Washington DC, USA, 145-150, 2012

### Examples

```
## Not run: test = pubtator3_function(17922911)
## here pubtator_function() will extract the information from this given pmid.
```

---

readabs	<i>To read Abstracts</i>
---------	--------------------------

---

**Description**

readabs will automatically read the abstracts from the pubmed file.

**Usage**

```
readabs(x)
```

**Arguments**

x                      Text file of PubMed abstracts. (Abstracts downloaded from PubMed)

**Details**

The saved file from a general pubmed search as text file is read via readabs().

**Value**

An S4 object of class "Abstracts", and a text file with tab delimited headers Journal, Abstract, PMID written with file name "newabs.txt".

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: readabs("pubmed_result.txt")  
##here pubmed_result.txt is the text file of abstracts saved from PubMed.
```

---

readabsnew	<i>To read Abstracts</i>
------------	--------------------------

---

**Description**

readabsnew will automatically read the abstracts from the pubmed text file.

**Usage**

```
readabsnew(x)
```

**Arguments**

x                      Text file of PubMed abstracts. (Abstracts downloaded from PubMed)

**Details**

The saved file from a general pubmed search as text file is read via readabsnew().

**Value**

An S4 object of class "Abstracts" and a text file with tab delimited headers Journal, Abstract, PMID written with file name "newabs.txt".

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: readabsnew("pubmed_result.txt")  
##here pubmed_result.txt is the text file of abstracts saved from PubMed.
```

---

ready

*To Initiate the Classes.*

---

**Description**

ready will initiate the classes necessary for other functions.

**Usage**

```
ready()
```

**Details**

This function is necessary to initiate the classes which are needed for the implementation of other functions.

**Value**

classes

**Author(s)**

S. Ramachandran

**Examples**

```
## Not run: ready()
```

---

removeabs	<i>To remove abstracts for the query term.</i>
-----------	--

---

**Description**

removeabs will remove the abstracts from a corpus for a given term.

**Usage**

```
removeabs(object, x, y)
```

**Arguments**

object	An S4 object of class Abstracts
x	A character value
y	is logical, if set 'TRUE' search will be case specific

**Details**

removeabs() finds the abstracts for the given term and remove them from the large set of abstracts. A text file of file name "dataout.txt" will be written containing the number of abstracts removed.

**Value**

An S4 object of class Abstracts and a text file named "dataout.txt"

**Author(s)**

S.Ramachandran, Jyoti Rani

**Examples**

```
## Not run: removeabs(myabs, "atherosclerosis", TRUE)
```

---

removeabs-methods	removeabs <i>To remove abstracts of a term from the data.</i>
-------------------	---

---

**Description**

removeabs This function will search for the abstracts containing the given term to remove them from the data.

**Methods**

signature(object = "Abstracts") This method depicts its function, it will remove the abstracts from the data, and the number of abstracts removed will be written the text file named "dataout.txt"

---

`searchabsL`*To Search the abstracts of term(s) in a combination mode.*

---

**Description**

`searchabsL` will search for abstracts for the given term(s). Multiple combinations are allowed.

**Usage**

```
searchabsL(object, yr, include, restrict, exclude)
```

**Arguments**

<code>object</code>	An S4 object of class Abstracts
<code>yr</code>	character vector specifies the year of search
<code>include</code>	character vector specifies the terms contained in the abstracts.
<code>restrict</code>	character vector specifies the term contained in the abstracts for which search should be restricted.
<code>exclude</code>	character vector specifies the terms contained in the abstracts for excluding these abstracts from the search results.

**Details**

In the arguments except for the object all other arguments have "NONE" as default. To export or write the result of `searchabsL()` we use `sendabs()` function.

**Value**

An object of class Abstracts satisfying the term combinations, In addition a text file named "out.txt" reporting the number of abstracts for given query term combinations.

**Author(s)**

S.Ramachandran

**See Also**

[searchabsT](#)

**Examples**

```
## Not run: searchabsL(myabs, include="term")
searchabsL(myabs, yr="2013")
searchabsL(myabs, restrict="term")
searchabsL(myabs, exclude="term")
searchabsL(myabs, include="term", exclude="term2")
## End(Not run)
## Here myabs is the object of class Abstracts containing data,
## "term" is the query term to be search.
```

---

searchabsL-methods      *Searching Abstracts*

---

### Description

searchabsL will automatically search the abstracts from the data for the given terms or their combination of several terms.

### Methods

signature(object = "Abstracts") searchabsL will search the abstracts for the given term or combinations of several terms. In this method the argument "include" uses the boolean operator 'OR' and is liberal whereas the 'restrict' and 'exclude' use the boolean operator 'AND' to specify additional filters. If the restriction to individual terms are desired then they can be individually searched and then the multiple abstracts can be combined using combineasb() function.

---

searchabsT      *To Search Abstracts*

---

### Description

searchabsTIt is similar to searchabsL() but performs more specific search. It performs case sensitive search.

### Usage

```
searchabsT(object, yr, include, restrict, exclude)
```

### Arguments

object	An S4 object of class Abstracts
yr	character vector specifies the year(s) of search.
include	character vector specifies the term(s) for which abstracts to be searched.
restrict	character vector specifies the term(s) contained in the abstracts for which search should be restricted.
exclude	character vector specifies the term(s) contained in the abstracts for excluding these abstracts from our search results.

### Details

In the arguments except the object all arguments have "NONE" as default. Use sendabs() function to write the results in a tab delimited text file.

**Value**

An object of class Abstracts meeting the term and the term combinations. A text file reporting the number of abstracts for the query terms and their combinations is also written with the filename "out.txt".

**Author(s)**

Dr.S.Ramachandran

**See Also**

[searchabsL](#)

**Examples**

```
## Not run: searchabsT(myabs,yr="2013")
searchabsT(myabs,include="term")
searchabsT(myabs,restrict="term")
searchabsT(myabs,exclude="term")
searchabsT(myabs,yr="2013", include="term")
## End(Not run)
## Here myabs is an S4 object of class Abstracts containing the abstracts to search,
## "term" is the query term to be search.
```

---

searchabsT-methods	searchabsT <i>Searching abstracts</i>
--------------------	---------------------------------------

---

**Description**

searchabsT will perform a specific search for the given term.

**Methods**

signature(object = "Abstracts") It is similar to the searchabsL method, but it is more specific than searchabsL, it is case sensitive, however searchabsL is not.

---

sendabs	<i>To send abstracts</i>
---------	--------------------------

---

**Description**

sendabs will send the abstracts into a tab delimited text file with the fields Journal, Abstract, and PMID.

**Usage**

```
sendabs(object, x)
```

**Arguments**

object            An S4 object of class 'Abstracts'  
x                 "filename.txt" to write the abstracts

**Details**

A general writing function for object of class 'Abstracts'

**Value**

A tab delimited text file with headers Journal, Abstract, PMID.

**Author(s)**

S.Ramachandran, Jyoti Rani

**Examples**

```
## Not run: sendabs(myabs,"myabs.txt")  
## here myabs is the S4 object of class 'Abstracts' and  
## 'abs.txt' is the file where abstracts will be written.
```

---

sendabs-methods            *To send the Data into a File*

---

**Description**

sendabs will write the data of an object of class 'Abstracts' into a tab delimited text file with header Journal, Abstract, and PMID

**Methods**

signature(object = "Abstracts") sendabs will send the data into a text file. It writes a tab delimited text file for PubMed abstracts containing Journal, Abstract, and PMID.

---

SentenceToken	<i>To Tokenize the sentences</i>
---------------	----------------------------------

---

**Description**

SentenceToken will tokenize abstracts into individual sentences.

**Usage**

```
SentenceToken(x)
```

**Arguments**

x is a character string; could be an output from paste

**Details**

This function is necessary for extracting sentences from abstracts, used by contextSearch function. The tokenization principle follows the overall strategy as described in contextSearch

**Value**

A character vector of sentences

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: SentenceToken(x)
```

---

space_quasher	<i>Removes extra spaces between words.</i>
---------------	--

---

**Description**

space\_quasher will automatically remove extra spaces between words. Therefore only one space between any pair of words will be left

**Usage**

```
space_quasher(x)
```

**Arguments**

x is a text with single or multiple sentences given within double quotes.

**Details**

The extra spaces between words in sentences is quashed to one via `space_quasher()`.

**Value**

Sentences(s) in which extra spaces between any pair of words are quashed to one.

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: space_quasher("I am a ghostbuster. I have the tools required to hunt ghosts")
##here pubmed_result.txt is the text file of abstracts saved from PubMed.
```

---

subabs	<i>To find sub-abstracts</i>
--------	------------------------------

---

**Description**

subabs will automatically extract the sub-abstracts from large set of abstracts.

**Usage**

```
subabs(object, start, end)
```

**Arguments**

object	An S4 object of class Abstracts
start	integer, specifies starting limit of the range to perform search
end	integer, specifies end limit of the range to perform search

**Details**

From a large number of asbstracts wish to extract a subset of abstracts into a separate object.

**Value**

An R object of class 'Abstracts' containing the extracted abstracts meeting a given range.

**Author(s)**

Jyoti Rani, S.Ramachandran

**Examples**

```
## Not run: subabs(myabs,1,5)
## Here 'myabs is an S4 object of class 'Abstracts',
## 1 and 5 are the start and end respectively.
```

subsetabs-methods

*Getting subabstracts*

---

**Description**

subsetabs subsetabs will extract the sub abstracts corresponding to a given range, from the whole data.

**Methods**

signature(object = "Abstracts") From an S4 object of class 'Abstracts' the subsetabs function is able to extract the abstracts corresponding to a given range.

---

subsetabs

*To make subsets of large corpus.*

---

**Description**

It is used to divide the large corpus into a given range.

**Usage**

```
subsetabs(object, indices)
```

**Arguments**

object            object is an S4 object containing Abstracts.  
indices           indices a numeric range (e.g. 1:10, c(1,5,7,9,10)).

**Value**

It returns an S4 object of extracted Abstracts.

**Author(s)**

S. Ramachandran.

**Examples**

```
## Not run: test = subsetabs(diabetesabs, 1:50)  
## here we want to extract the Abstracts ranges from 1 to 50  
## from the large corpus of diabetes.
```

---

subsetabs-methods	<i>To make subset of Abstracts.</i>
-------------------	-------------------------------------

---

**Description**

subsetabs is used to subset of Abstracts from the large corpus. Its output is used in other functions like currentabs\_fn and previousabs\_fn

**Methods**

signature(object = "Abstracts") subsetabs will divide the large corpus into subset.

---

tdm_for_lsa	<i>create Term Document Matrix for lsa analysis</i>
-------------	---

---

**Description**

lsa package take "Term Document Matrix" as input, so it is needed to create a 'tdm' for Abstracts and tdm\_for\_lsa do the same as it find out the frequency of given term in each abstract and each abstract is considered as separate document. It prepares term document matrix of terms in the 'abstracts' corpus

**Usage**

```
tdm_for_lsa(object, y)
```

**Arguments**

object	An S4 object of class 'Abstracts'
y	a character vector specifying the terms

**Value**

a Term Document Matrix (Numerical matrix) containing the raw frequencies of given terms in each abstract.

**Author(s)**

Jyoti Rani

**Examples**

```
## Not run: y = c("insulin", "inflammation", "obesity")
tdm_for_lsa(myabs,y)
## End(Not run)
```

---

uniprotfun

*To get information about gene from the UniProt.Deprecated.*

---

**Description**

uniprotfun will access the UniProt data for a given gene as per HGNC approved gene symbols. Deprecated.

**Usage**

```
uniprotfun(y)
```

**Arguments**

y                    a HGNC approved gene symbol as character

**Details**

This function retrieves data from the UniProt. At present uniprotfun() works with only HGNC approved gene symbols.

**Value**

A text file written with filename as the 'query' name suffixed with .txt

**Author(s)**

S.Ramachandran

**Examples**

```
## Not run: uniprotfun("SIRT1")
```

---

whichcluster

*To fetch the cluster for words*

---

**Description**

whichcluster is used to get the cluster in which a given word (term) occurs.

**Usage**

```
whichcluster(clusterobject, y)
```

**Arguments**

`clusterobject` an R object containing the clusters of words output by `wordscluster` function.  
`y` a character string of query terms.

**Value**

a list containing the number of cluster under which given term occurs.

**Author(s)**

S.Ramachandran

**See Also**

[wordscluster](#)

**Examples**

```
## Not run: test<-whichcluster(x, "diabetes")
## here x is an R object output form wordscluster function.
## and "diabetes" is the term for which cluster number is to be searched.
## End(Not run)
```

---

word_associations	<i>Extracts the words associated (to the left and to the right) with a given word</i>
-------------------	---

---

**Description**

`word_associations` will automatically extract associated words for a given word, namely the words immediately to the left and to the right. The given word is usually in the middle except for those cases, where the given word occurs either at the start or the end of the sentence.

**Usage**

```
word_associations(term, abs)
```

**Arguments**

`term` is a single word  
`abs` an S4 object of class Abstracts

**Details**

Certain words are qualified by authors in various ways. For example, physical therapy, gene therapy etc. This function is useful in extracting these qualified words in the form of available associated words. Useful for preparing terms to be given in `co_occurrence_fn()`. There could be other uses also.

**Value**

comp1            A list of all the word pairs in a given set of abstracts.

**Author(s)**

S. Ramachandran

**References**

Rani J, Shah AB, Ramachandran S. pubmed.mineR: an R package with text-mining algorithms to analyse PubMed abstracts. J Biosci. 2015 Oct;40(4):671-82. PubMed PMID: 26564970.

**See Also**

Give\_Sentences

**Examples**

```
## Not run: word_associations("therapy",myabs
##
```

---

word\_atomizations        *Atomization of words*

---

**Description**

word\_atomizations will automatically break the whole text into words and rank them according to their frequency of occurrence.

**Usage**

```
word_atomizations(m)
```

**Arguments**

m                    An S4 object of class Abstracts

**Details**

word\_atomizations() will break down the whole text into words after removing the extra white space, punctuation marks and very common english words.

**Value**

A text file containing words with their frequencies

**Author(s)**

S. Ramachandran, Jyoti Sharma

## Examples

```
## Not run: word_atomizations(myabs)
## here myabs is the object containing abstracts.
```

---

wordcluster

*To cluster the words*

---

## Description

wordcluster is used to cluster the words, using the levenshtein distance concept, which are coming together in combination with either 'prefixes' or 'suffixes' or other compound words. The first word, usually of lowest length, could be 'stemmed' word in many cases drastically so, is considered as representative for that cluster.

## Usage

```
wordcluster(lower, upper)
```

## Arguments

lower	lower limit for characters in word. Default = 5.
upper	upper limit of characters in word. Default = 30

## Details

This function is usefull for dampening the 'explotion' of words output from word\_atomizations. This step enables easy examination of the terms.

## Value

a list object of words clustered together and a text filenameed "resulttable.txt" with the columns cluster number, cluster size and representatives of clusters.

## Note

The function may run faster when the lower limits are reduced but 'risks' producing plenty of 'decoy' situations. Their frequencies are very rare. Decoy situations: Some 'words' with part identity to other smaller words will runaway with smaller words. This event creates an unfavorable situation whereby the generated 'clusters' of words become difficult to interpret. This situation can be minimized by increasing the lower limit of word length, however at the cost of lowering computational speed. An example is: the word hypercholesterolemia runsaway with the smaller word 'lester' which could be another name. In this instance increasing the lower limit will be more usefull. Words longer than 30 characters are usually names of chemical compnunds in IUPAC system of nomenclature.

## Author(s)

S.Ramachandran, Jyoti Rani

**See Also**

[whichcluster word\\_atomizations](#)

**Examples**

```
## Not run:  
test=wordycluster(5, 10)  
## here it will start making cluster of words of length with minimum of 5 characters  
## and maximum of 10 characters.  
  
## End(Not run)
```

---

wordyclusterview	<i>To view the words in cluster</i>
------------------	-------------------------------------

---

**Description**

wordyclusterview is used to view the words comes in cluster formed by wordycluster function.

**Usage**

```
wordyclusterview(words_cluster, all)
```

**Arguments**

`words_cluster` an R object containing output of wordycluster  
`all` is logical and default is FALSE, if set to TRUE includes those with one member word.

**Details**

The first 5 words and 5 words near the median and 5 words at the tail end are shown for clusters with more than 15 members. In case of cluster size less than 15, all the words are written in output.

**Value**

It returns a text file named word\_cluster\_view.txt

**Author(s)**

S.Ramachandran, Jyoti Rani

**See Also**

[wordycluster](#)

**Examples**

```
## Not run: test= wordsclusterview(cluster)
# here cluster is output from wordscluster
## End(Not run)
```

---

xmlgene\_atomizations *Gene atomization of xml abstracts.Deprecated.*

---

**Description**

xmlgene\_atomizations is used to fetch the list of genes from the xml abstracts.Deprecated.

**Usage**

```
xmlgene_atomizations(m)
```

**Arguments**

m an S4 object of class Abstracts, output from xmlreadabs.

**Value**

a list containing genes from the text with their frequency of occurrence.

**Author(s)**

S.Ramachandran, Jyoti Sharma

**See Also**

[xmlreadabs](#)

**Examples**

```
## Not run: test = xmlgene_atomizations(xmlabs)
## xmlabs is an S4 object of class Abstracts i.e. output of xmlreadabs
```

xmlgene\_atomizations\_new

*Gene atomization of xml abstracts.*

---

### **Description**

xmlgene\_atomizations\_new is used to fetch the list of genes from the xml abstracts

### **Usage**

```
xmlgene_atomizations_new(m)
```

### **Arguments**

m                    an S4 object of class Abstracts, output from xmlreadabs.

### **Value**

a list containing genes from the text with their frequency of occurrence.

### **Author(s)**

S.Ramachandran, Jyoti Sharma

### **See Also**

[new\\_xmlreadabs](#)

### **Examples**

```
## Not run: test = xmlgene_atomizations(xmlabs)
## xmlabs is an S4 object of class Abstracts i.e. output of xmlreadabs
```

---

xmlreadabs

*To read the abstracts from the PubMed saved in XML format.*

---

### **Description**

xmlreadabs is modified form of readabs as it reads the abstracts downloaded/saved in XML format from PubMed. This is helpful to give clean and better result after preprocessing i.e. word\_atomizations, wordscluster etc.

### **Usage**

```
xmlreadabs(file)
```

**Arguments**

file                    an XML file saved from PubMed.

**Value**

an S4 object of class Abstracts containing journals, abstracts and PMID.

**Author(s)**

S.Ramachandran

**See Also**

[readabs](#)

**Examples**

```
## Not run: xmlabs = xmlreadabs("pubmed_result.xml")
## here "pubmed_result.xml" is an xml format file downloaded from PubMed.
```

---

xmlword\_atomizations    *Word atomizations of abstracts from xml format.*

---

**Description**

xmlword\_atomizations is used to process the abstracts from PubMed in XML format.

**Usage**

```
xmlword_atomizations(m)
```

**Arguments**

m                      an S4 object of class Abstracts resulted from xmlreadabs.

**Value**

a list containing words from the text with their frequencies.

**Note**

xmlword\_atomizations cannot work on output of readabs.

**Author(s)**

S. Ramachandran

**See Also**[xmlreadabs](#)**Examples**

```
## Not run: test = xmlword_atomizations(xmlabs)
## here xmlabs is an S4 object i.e. output of xmlreadabs
```

---

Yearwise

*To Search abstracts Year wise*

---

**Description**

Yearwise reports the no. of abstracts in a year.

**Usage**

```
Yearwise(object, year)
```

**Arguments**

object	An S4 object of class Abstracts.
year	a character vector specifies the year.

**Details**

Yearwise() is useful to find the no. of abstracts for the given year.

**Value**

A text file containing the no. of abstracts for given Year(s)

**Author(s)**

Dr.S.Ramachandran

**Examples**

```
## Not run: Yearwise(myabs, "2011") or
Yearwise(myabs, c("2011", "2013", "2009"))
## End(Not run)
## Here myabs is the object containing PubMed abstracts.
```

---

Yearwise-methods

Yearwise *Year wise extraction of Abstracts*

---

**Description**

Yearwise will report the abstracts for given year(s).

**Methods**

signature(object = "Abstracts") This method "Yearwise" is written to fetch the abstracts yearly.

# Index

## \* **Functions**

sendabs, [52](#)

## \* **Function**

additional\_info, [5](#)  
cleanabs, [8](#)  
cluster\_words, [9](#)  
co\_occurrence\_fn, [11](#)  
combineabs, [12](#)  
contextSearch, [13](#)  
cos\_sim\_calc, [14](#)  
cos\_sim\_calc\_boot, [15](#)  
Find\_conclusion, [17](#)  
find\_intro\_conc\_html, [18](#)  
gene\_atomization, [19](#)  
Genewise, [21](#)  
get\_gene\_sentences, [23](#)  
get\_NMids, [24](#)  
get\_original\_term, [25](#)  
getabs, [29](#)  
getabsT, [30](#)  
Give\_Sentences, [31](#)  
input\_for\_find\_intro\_conc\_html, [35](#)  
new\_xmlreadabs, [38](#)  
printabs, [42](#)  
readabs, [47](#)  
readabsnew, [47](#)  
removeabs, [49](#)  
searchabsL, [50](#)  
searchabsT, [51](#)  
SentenceToken, [54](#)  
space\_quasher, [54](#)  
subabs, [55](#)  
uniprotfun, [58](#)  
whichcluster, [58](#)  
word\_associations, [59](#)  
wordscluster, [61](#)  
wordsclusterview, [62](#)  
xmlgene\_atomizations, [63](#)  
xmlgene\_atomizations\_new, [64](#)

xmlreadabs, [64](#)

xmlword\_atomizations, [65](#)

Yearwise, [66](#)

## \* **classes**

Abstracts-class, [4](#)

HGNC-class, [33](#)

## \* **currentabs\_fn**

currentabs\_fn, [16](#)

## \* **datasets**

common\_words\_new, [13](#)

GeneToEntrez, [21](#)

HGNC2UniprotID, [34](#)

HGNCdata, [34](#)

## \* **function**

alias\_fn, [5](#)

altnamesfun, [6](#)

BWI, [7](#)

genes\_BWI, [20](#)

get\_DOIs, [22](#)

get\_MedlinePlus, [24](#)

get\_PMCIDS, [27](#)

get\_PMCtable, [27](#)

get-Sequences, [28](#)

Give\_Sentences\_PMC, [31](#)

head\_abbrev, [32](#)

names\_fn, [37](#)

official\_fn, [39](#)

pmids\_to\_abstracts, [40](#)

prevsymbol\_fn, [41](#)

pubtator3\_function, [46](#)

pubtator\_function, [43](#)

ready, [48](#)

tdm\_for\_lsa, [57](#)

word\_atomizations, [60](#)

## \* **get\_original\_term2**

get\_original\_term2, [26](#)

## \* **local\_uniprot\_fun**

local\_uniprotfun, [36](#)

## \* **methods**

- cleanabs-methods, 9
- combineabs-methods, 12
- contextSearch-methods, 14
- Genewise-methods, 22
- getabs-methods, 29
- getabsT-methods, 30
- removeabs-methods, 49
- searchabsL-methods, 51
- searchabsT-methods, 52
- sendabs-methods, 53
- subabs-methods, 56
- subsetabs-methods, 57
- Yearwise-methods, 67
- \* **previousabs\_fn**
  - previousabs\_fn, 40
- \* **pubtator\_result\_list\_to\_table**
  - pubtator\_result\_list\_to\_table, 45
- \* **subsetabs**
  - subsetabs, 56
- Abstracts, 33
- Abstracts-class, 4
- additional\_info, 5
- alias\_fn, 5
- altnamesfun, 6
- BWI, 7, 20
- cleanabs, 8
- cleanabs, Abstracts-method
  - (cleanabs-methods), 9
- cleanabs-methods, 9
- cluster\_words, 9
- co\_occurrence\_advance, 10
- co\_occurrence\_fn, 11, 11
- combineabs, 4, 12
- combineabs, Abstracts-method
  - (combineabs-methods), 12
- combineabs-methods, 12
- common\_words\_new, 13
- contextSearch, 4, 13
- contextSearch, Abstracts-method
  - (contextSearch-methods), 14
- contextSearch-methods, 14
- cos\_sim\_calc, 14
- cos\_sim\_calc\_boot, 15
- currentabs\_fn, 16, 41
- Find\_conclusion, 17
- find\_intro\_conc\_html, 18, 36
- gene\_atomization, 19
- genes\_BWI, 8, 20
- GeneToEntrez, 21
- Genewise, 4, 21
- Genewise, Abstracts-method
  - (Genewise-methods), 22
- Genewise-methods, 22
- get\_DOIs, 22
- get\_gene\_sentences, 23
- get\_MedlinePlus, 24
- get\_NMids, 24, 28
- get\_original\_term, 25, 26
- get\_original\_term2, 26
- get\_PMCIDS, 27, 28
- get\_PMCTable, 27
- get\_Sequences, 25, 28
- getabs, 4, 29
- getabs, Abstracts-method
  - (getabs-methods), 29
- getabs-methods, 29
- getabsT, 30
- getabsT, Abstracts-method
  - (getabsT-methods), 30
- getabsT-methods, 30
- Give\_Sentences, 5, 31
- Give\_Sentences\_PMC, 31, 31
- head\_abbrev, 32
- HGNC-class, 33
- HGNC2UniprotID, 34
- HGNCdata, 34
- input\_for\_find\_intro\_conc\_html, 18, 35
- local\_uniprotfun, 36
- names\_fn, 37, 42
- new\_xmlreadabs, 38, 64
- official\_fn, 39, 42
- pmids\_to\_abstracts, 40
- previousabs\_fn, 17, 40
- prevsymbol\_fn, 6, 41
- printabs, 42
- pubtator3\_function, 46
- pubtator\_function, 43, 45
- pubtator\_function\_JSON, 44

- pubtator\_result\_list\_to\_table, [45](#)
- readabs, [4](#), [38](#), [47](#), [65](#)
- readabsnew, [38](#), [47](#)
- ready, [48](#)
- removeabs, [49](#)
- removeabs, Abstracts-method  
(removeabs-methods), [49](#)
- removeabs-methods, [49](#)
  
- searchabsL, [4](#), [9](#), [50](#), [52](#)
- searchabsL, Abstracts-method  
(searchabsL-methods), [51](#)
- searchabsL-methods, [51](#)
- searchabsT, [50](#), [51](#)
- searchabsT, Abstracts-method  
(searchabsT-methods), [52](#)
- searchabsT-methods, [52](#)
- sendabs, [52](#)
- sendabs, Abstracts-method  
(sendabs-methods), [53](#)
- sendabs-methods, [53](#)
- SentenceToken, [54](#)
- space\_quasher, [54](#)
- subabs, [4](#), [55](#)
- subabs, Abstracts-method  
(subabs-methods), [56](#)
- subabs-methods, [56](#)
- subsetabs, [4](#), [56](#)
- subsetabs, Abstracts-method  
(subsetabs-methods), [57](#)
- subsetabs-methods, [57](#)
  
- tdm\_for\_lsa, [15](#), [16](#), [57](#)
  
- uniprotfun, [7](#), [36](#), [58](#)
  
- whichcluster, [58](#), [62](#)
- word\_associations, [59](#)
- word\_atomizations, [60](#), [62](#)
- wordscluster, [10](#), [26](#), [59](#), [61](#), [62](#)
- wordsclusterview, [62](#)
  
- xmlgene\_atomizations, [63](#)
- xmlgene\_atomizations\_new, [64](#)
- xmlreadabs, [63](#), [64](#), [66](#)
- xmlword\_atomizations, [65](#)
  
- Yearwise, [4](#), [66](#)
- Yearwise, Abstracts-method  
(Yearwise-methods), [67](#)
- Yearwise-methods, [67](#)