# Package: austin (via r-universe)

September 8, 2024

**Type** Package

**Title** Do Things with Words

**Version** 0.5.0

**Date** 2020-12-23

**Description** Doing things with words currently means scaling documents on a presumed underlying dimension on the basis of word frequencies and heroic assumptions about language generation.

**URL** https://conjugateprior.github.io/austin

**BugReports** https://github.com/conjugateprior/austin/issues

**Depends** R (>= 3.1)

**Imports** dplyr, tibble, numDeriv, methods, Matrix, tokenizers, irlba

**License** file LICENSE

**LazyLoad** yes

**Suggests** knitr, rmarkdown, testthat

**VignetteBuilder** knitr

**Encoding** UTF-8

**RoxygenNote** 7.1.1

**Repository** https://conjugateprior.r-universe.dev

**RemoteUrl** https://github.com/conjugateprior/austin

**RemoteRef** HEAD

**RemoteSha** 42d6195bffb0ae36d430e186472e15738b7c4657

## Contents


as.docword . . . . . 3
as.wfm . . . . . 3
as.worddoc . . . . . 4
austin . . . . . 5
bootstrap.se . . . . . 5

classic.wordscores . . . 6
coef.classic.wordscores . . . 7
coef.wordfish . . . 8
daildata . . . 9
demanif . . . 9
demanif.econ . . . 10
demanif.foreign . . . 10
demanif.soc . . . 11
docs . . . 11
extractwords . . . 12
fitted.wordfish . . . 12
getdocs . . . 13
iebudget2009 . . . 14
initialize.urfish . . . 14
interestgroups . . . 15
is.wfm . . . 16
K2009 . . . 16
LB2002 . . . 17
LB2013 . . . 17
lbg . . . 18
LBG2003 . . . 18
LG2000 . . . 19
plot.classic.wordscores . . . 19
plot.coef.wordfish . . . 20
plot.wordfish . . . 20
predict.classic.wordscores . . . 21
predict.wordfish . . . 22
rescale . . . 23
sim.wordfish . . . 24
SP2008 . . . 25
SP2008_econ . . . 25
SP2008_for . . . 26
SP2008_soc . . . 26
summary.classic.wordscores . . . 27
summary.wordfish . . . 27
trim . . . 28
ukmanif . . . 29
wfm . . . 29
wfm2bmr . . . 30
wfm2lda . . . 31
wordfish . . . 32
wordmargin . . . 34
words . . . 34

**Index** **36**

## as.docword *Extract a Document by Word Matrix*

### Description

Extract a word count matrix with documents as rows and words as columns

### Usage

```
as.docword(wfm)
```

### Arguments

| | |
|---|---|
| wfm | an object of class wfm |

### Details

This is a helper function for wfm objects. Use it instead of manipulating wfm object themselves.

### Value

a document by word count matrix

### Author(s)

Will Lowe

### See Also

as.worddoc, wfm

## as.wfm *Coerce to a Word Frequency Matrix*

### Description

Constructs a wfm object from various other kinds of objects

### Usage

```
as.wfm(mat, word.margin = 1)
```

### Arguments

| | |
|---|---|
| mat | a matrix of counts |
| word.margin | which margin of mat represents the words |

## Value

an object of class wfm

## Author(s)

Will Lowe

## See Also

wfm

---

`as.worddoc` *Extract a Word by Document Matrix*

---

## Description

Extract a matrix of word counts with words as rows and documents as columns

## Usage

```
as.worddoc(wfm)
```

## Arguments

| | |
|---|---|
| `wfm` | an object of class wfm |

## Details

This is a helper function for wfm objects. Use it instead of manipulating wfm object themselves.

## Value

a word by document count matrix

## Author(s)

Will Lowe

## See Also

`as.docword`, `wfm`

## austin *austin: Do things with words*

### Description

Austin helps you see what people, usually politicians, do with words. Currently that means how positions on a presumed underlying policy scale are taken by manipulating word occurrence counts. The models implemented here try to try to recover those positions using only this information, plus some heroic assumptions about language generation, e.g. unidimensionality, conditional independence of words given ideal point and Poisson-distributed word counts.

### Details

The package currently implements Wordfish (Slapin and Proksch, 2008) and Wordscores (Laver, Benoit and Garry, 2003). See references for details.

## bootstrap.se *Compute Bootstrap Standard Errors*

### Description

Computes bootstrap standard errors for document positions from a fitted Wordfish model

### Usage

```
bootstrap.se(object, L = 50, verbose = FALSE, ...)
```

### Arguments

| | |
|---|---|
| object | a fitted Wordfish model |
| L | how many replications |
| verbose | Give progress updates |
| ... | Unused |

### Details

This function computes a parametric bootstrap by resampling counts from the fitted word counts, refitting the model, and storing the document positions. The standard deviations for each resampled document position are returned.

### Value

Standard errors for document positions

### Author(s)

Will Lowe

`classic.wordscores` *Old-Style Wordscores*

## Description

Construct a Wordscores model from reference document scores

## Usage

```
classic.wordscores(wfm, scores)
```

## Arguments

| | |
|---|---|
| `wfm` | object of class wfm |
| `scores` | reference document positions/scores |

## Details

This version of Wordscores is exactly as described in Laver et al. 2003 and is provided for historical interest and continued replicability of older analyses.

`scores` is a vector of document scores corresponding to the documents in the word frequency matrix `wfm`. The function computes wordscores and returns a model from which virgin text scores can be predicted.

## Value

An old-style Wordscores analysis.

## Author(s)

Will Lowe

## References

Laver, M. and Benoit, K. and Garry, J. (2003) 'Extracting policy positions from political texts using words as data' American Political Science Review. 97. pp.311-333

## See Also

`summary.classic.wordscores`

## Examples

```
data(lbg)
ref <- getdocs(lbg, 1:5)
ws <- classic.wordscores(ref, scores=seq(-1.5,1.5,by=0.75))
summary(ws)
vir <- getdocs(lbg, 'V1')
predict(ws, newdata=vir)
```

---

`coef.classic.wordscores` *Show Wordscores*

---

## Description

Lists wordscores from a fitted Wordscores model.

## Usage

```
## S3 method for class 'classic.wordscores'
coef(object, ...)
```

## Arguments

| | |
|---|---|
| `object` | a fitted Wordscores model |
| `...` | extra arguments, currently unused |

## Value

The wordscores

## Author(s)

Will Lowe

## See Also

`classic.wordscores`

`coef.wordfish` *Extract Word Parameters*

## Description

Extract word parameters beta and psi in an appropriate model parameterization

## Usage

```
## S3 method for class 'wordfish'
coef(object, form = c("poisson", "multinomial"), ...)
```

## Arguments

| | |
|---|---|
| `object` | an object of class wordfish |
| `form` | which parameterization of the model to return parameters for |
| `...` | extra arguments |

## Details

Slope parameters and intercepts are labelled beta and psi respectively. In multinomial form the coefficient names reflect the fact that the first-listed word is taken as the reference category. In poisson form, the coefficients are labeled by the words the correspond to.

Note that in both forms there will be beta and psi parameters, so make sure they are the ones you want.

## Value

A data.frame of word parameters from a wordfish model in one or other parameterization.

## Author(s)

Will Lowe

## See Also

`wordfish`

## dailydata *The 1991 Irish Confidence debate*

### Description

Irish Confidence Debate

### Details

This are word counts from the no-confidence motion debated in the Irish Dáil from 16-18 October 1991 over the future of the Fianna Fail-Progressive Democrat coalition. `dailydata` is a word frequency object.

### References

Laver, M. & Benoit, K.R. (2002). Locating TDs in Policy Spaces: Wordscoring Dáil Speeches. Irish Political Studies, 17(1), 59–73.

## demanif *German Party Manifesto Data*

### Description

A random sample of words and their frequency in German political party manifestos from 1990-2005.

### Details

demanif is word frequency matrix

### Source

Wordfish website (http://www.wordfish.org)

### References

J. Slapin and S.-O. Proksch (2008) 'A scaling model for estimating time-series party positions from texts' American Journal of Political Science 52(3), 705-722.

## demanif.econ *Economics sections of German Party Manifestos*

### Description

A word frequency matrix from the economic sections of German political party manifestos from 1990-2005.

### Details

demanif.econ is word frequency matrix

### Source

These data are courtesy of S.-O. Proksch.

### References

J. Slapin and S.-O. Proksch (2008) 'A scaling model for estimating time-series party positions from texts' American Journal of Political Science 52(3), 705-722.

## demanif.foreign *Foreign Policy Sections of German Party Manifestos*

### Description

A word frequency matrix from the foreign policy sections of German political party manifestos from 1990-2005.

### Details

demanif.foreign is word frequency matrix

### Source

These data courtesy of S.-O. Proksch.

### References

J. Slapin and S.-O. Proksch (2008) 'A scaling model for estimating time-series party positions from texts' American Journal of Political Science 52(3), 705-722.

## demanif.soc *Societal sections of German Party Manifestos*

### Description

A word frequency matrix from the societal sections of German political party manifestos from 1990-2005.

### Details

demanif.soc is word frequency matrix

### Source

These data courtesy are of S.-O. Proksch.

### References

J. Slapin and S.-O. Proksch (2008) 'A scaling model for estimating time-series party positions from texts' American Journal of Political Science 52(3), 705-722.

## docs *Extract Document Names*

### Description

Extracts the document names from a wfm object.

### Usage

```
docs(wfm)

docs(wfm) <- value
```

### Arguments

| | |
|---|---|
| wfm | an object of type wfm |
| value | replacement if assignment |

### Value

A list of document names.

### Author(s)

Will Lowe

## See Also

wfm

---

`extractwords` *Pull Words From a List*

---

## Description

Extract a list of matching words from another list of words

## Usage

```
extractwords(words, patternfile, pattern.type = c("glob", "re"))
```

## Arguments

| | |
|---|---|
| `words` | the words against which patters are matched |
| `patternfile` | file containing the patters to match, one per line |
| `pattern.type` | marks whether the patterns are 'globs' or full regular expressions |

## Value

A list of matching words.

## Author(s)

Will Lowe

---

`fitted.wordfish` *Get Fitted Values from a Wordfish Model*

---

## Description

Extracts the estimated word rates from a fitted Wordfish model

## Usage

```
## S3 method for class 'wordfish'
fitted(object, ...)
```

## Arguments

| | |
|---|---|
| `object` | a fitted Wordfish model |
| `...` | Unused |

### Value

Expected counts in the word frequency matrix

### Author(s)

Will Lowe

---

`getdocs` *Get Documents*

---

### Description

Gets particular documents from a wfm by name or index

### Usage

```
getdocs(wfm, which)
```

### Arguments

| | |
|---|---|
| `wfm` | a wfm object |
| `which` | names or indexes of documents |

### Details

getdocs is essentially a subset command that picks the correct margin for you.

### Value

A smaller wfm object containing only the desired documents with the same word margin setting as the original matrix.

### Author(s)

Will Lowe

### See Also

`as.wfm`, `as.docword`, `as.worddoc`, `docs`, `words`, `is.wfm`, `wordmargin`

## iebudget2009 *Irish Budget Debate Data 2009*

### Description

Irish budget debate 2009

Irish budget debate 2009

### Details

This are word counts from the 2009 Budget debate in Ireland.

This is a word frequency nmatrix. Loading this data also makes available `iebudget2009cov` which contains covariates for the speakers.

This are word counts from the 2009 Budget debate in Ireland.

This is a word frequency nmatrix. Loading this data also makes available `iebudget2009cov` which contains covariates for the speakers.

## initialize.urfish *initialize.urfish*

### Description

Get cheap starting values for a Wordfish model

### Usage

```
initialize.urfish(tY)
```

### Arguments

| | |
|---|---|
| `tY` | a document by word matrix of counts |

### Details

This function is only called by model fitting routines and does therefore not take a wfm classes. tY is assumed to be in document by term form.

In the poisson form of the model incidental parameters (alpha) are set to log(rowmeans/rowmeans[1]). intercept (psi) values are set to log(colmeans) These are subtracted from a the data matrix, which is logged and decomposed by SVD. Word slope (beta) and document position (theta) are estimated by rescaling SVD output.

## Value

List with elements:

| | |
|---|---|
| alpha | starting values of alpha parameters |
| psi | starting values of psi parameters |
| beta | starting values of beta parameters |
| theta | starting values for document positions |

## Author(s)

Will Lowe

## References

This is substantially the method used by Slapin and Proksch's original code.

---

# interestgroups *Interest Groups*

---

## Description

Interest Groups and the European Commission

## Details

Word counts from interest groups and a European Commission proposal to reduce CO2 emissions in 2007.

comm1 and comm2 are the Commission's proposal before and after the proposals of the interest groups.

## References

H. Kluever (2009) 'Measuring influence group influence using quantitative text analysis' European Union Politics 11:1.

## is.wfm *Checks for Word Frequency Matrix*

### Description

Checks whether an object is a Word Frequency Matrix

### Usage

```
is.wfm(x)
```

### Arguments

| | |
|---|---|
| x | a matrix of counts |

### Value

Whether the object can be used as a Word Frequency Matrix

### Author(s)

Will Lowe

### See Also

wfm

## K2009 *Interest Groups*

### Description

Interest Groups and the European Commission

### Details

Word counts from interest groups and a European Commission proposal to reduce CO2 emissions in 2007.

K2009 is a jl_df object.

### References

H. Kluever (2009) 'Measuring influence group influence using quantitative text analysis' European Union Politics 10(4) 535-549.

## LB2002 *The 1991 Irish Confidence debate*

### Description

Irish Confidence Debate (jl format)

### Details

This are word counts from the no-confidence motion debated in the Irish Dáil from 16-18 October 1991 over the future of the Fianna Fail-Progressive Democrat coalition.

`LB2003` is `jl_df` object.

### References

Laver, M. & Benoit, K.R. (2002). Locating TDs in Policy Spaces: Wordscoring Dáil Speeches. Irish Political Studies, 17(1), 59–73.

## LB2013 *Irish Budget Debate Data 2009*

### Description

Irish budget debate 2009

### Details

These are word counts from the 2009 Budget debate in Ireland.

`LB2013` is a `jl_df` object

### References

W. Lowe and K. Benoit (2013) 'Validating estimates of latent traits from textual data using human judgment as a benchmark' Political Analysis 21(3) 298-313.

## lbg *Example Data*

### Description

Example data from Laver Benoit and Garry (2003)

### Details

This is the example word count data from Laver, Benoit and Garry's (2000) article on Wordscores. Documents R1 to R5 are assumed to have known positions: -1.5, -0.75, 0, 0.75, 1.5. Document V1 is assumed unknown. The 'correct' position for V1 is presumed to be -0.45. `classic.wordscores` generates approximately -0.45.

To replicate the analysis in the paper, use the wordscores function either with identification fixing the first 5 document positions and leaving position of V1 to be predicted.

### References

Laver, Benoit and Garry (2003) 'Estimating policy positions from political text using words as data' American Political Science Review 97(2).

## LBG2003 *Example Data*

### Description

Example data from Laver Benoit and Garry (2003)

### Details

This is the example word count data from Laver, Benoit and Garry's (2000) article on Wordscores. Documents R1 to R5 are assumed to have known positions: -1.5, -0.75, 0, 0.75, 1.5. Document V1 is assumed unknown. The 'correct' position for V1 is presumably -0.45. `classic.wordscores` generates approximately -0.45.

To replicate the analysis in the paper, use the wordscores function either with identification fixing the first 5 document positions and leaving position of V1 to be predicted.

`LBG2003` is a `jl_df` object.

### References

M. Laver, K. Benoit and J. Garry (2003) 'Estimating policy positions from political text using words as data' American Political Science Review. 97(2) 311-331.

## LG2000 *UK Manifesto Data*

### Description

UK manifesto data from Laver et al.

### Details

This are word counts from the manifestos of the three main UK parties for the 1992 and 1997 elections.

`LG2000` is a `jl_df` object.

### References

M. Laver, K. Benoit and J. Garry (2003) 'Estimating policy positions from political text using words as data' American Political Science Review 97(2) 311-331.

## plot.classic.wordscores *Plot a Wordscores Model*

### Description

Plots Wordscores from a fitted Wordscores model

### Usage

```
## S3 method for class 'classic.wordscores'
plot(x, ...)
```

### Arguments

| | |
|---|---|
| x | a fitted Wordscores model |
| ... | other arguments, passed to the dotchart command |

### Value

A plot of the wordscores in increasing order.

### Author(s)

Will Lowe

### See Also

`classic.wordscores`

## plot.coef.wordfish *Plot the Word Parameters From a Wordfish Model*

### Description

Plots sorted beta and optionally also psi parameters from a Wordfish model

### Usage

```
## S3 method for class 'coef.wordfish'
plot(x, pch = 20, psi = TRUE, ...)
```

### Arguments

| | |
|---|---|
| x | a fitted Wordfish model |
| pch | Default is to use small dots to plot positions |
| psi | whether to plot word fixed effects |
| ... | Any extra graphics parameters to pass in |

### Value

A plot of sorted beta and optionally psi parameters.

### Author(s)

Will Lowe

### See Also

wordfish

## plot.wordfish *Plot a Wordfish Model*

### Description

Plots a fitted Wordfish model with confidence intervals

### Usage

```
## S3 method for class 'wordfish'
plot(x, truevals = NULL, level = 0.95, pch = 20, ...)
```

### Arguments

| | |
|---|---|
| x | a fitted Wordfish model |
| truevals | True document positions if known |
| level | Intended coverage of confidence intervals |
| pch | Default is to use small dots to plot positions |
| ... | Any extra graphics parameters to pass in |

### Value

A plot of sorted estimated document positions, with confidence intervals and true document positions, if these are available.

### Author(s)

Will Lowe

### See Also

wordfish

---

## predict.classic.wordscores

*Predict New Document Positions*

---

### Description

Predicts positions of new documents from a fitted Wordscores model

### Usage

```
## S3 method for class 'classic.wordscores'
predict(object, newdata = NULL, rescale = c("lbg", "none"), z = 0.95, ...)
```

### Arguments

| | |
|---|---|
| object | Fitted wordscores model |
| newdata | An object of class wfm in which to look for word counts to predict document ideal points. If omitted, the reference documents are used. |
| rescale | Rescale method for estimated positions. |
| z | Notional confidence interval coverage |
| ... | further arguments (quietly ignored) |

### Details

This is the method described in Laver et al. 2003, including rescaling for more than one virgin text. Confidence intervals are not provided if rescale is 'none'.

## Value

`predict.wordscores` produces a vector of predicted document positions and standard errors and confidence intervals.

## Author(s)

Will Lowe

## See Also

`classic.wordscores`

---

`predict.wordfish` *Predict Method for Wordfish*

---

## Description

Predicts positions of new documents using a fitted Wordfish model

## Usage

```
## S3 method for class 'wordfish'
predict(
  object,
  newdata = NULL,
  se.fit = FALSE,
  interval = c("none", "confidence"),
  level = 0.95,
  ...
)
```

## Arguments

| | |
|---|---|
| `object` | A fitted wordfish model |
| `newdata` | An optional data frame or object of class wfm in which to look for word counts to predict document ideal points which to predict. If omitted, the fitted values are used. |
| `se.fit` | A switch indicating if standard errors are required. |
| `interval` | Type of interval calculation |
| `level` | Tolerance/confidence level |
| `...` | further arguments passed to or from other methods. |

## Details

Standard errors for document positions are generated by numerically inverting the relevant Hessians from the profile likelihood of the multinomial form of the model.

### Value

`predict.wordfish` produces a vector of predictions or a matrix of predictions and bounds with column names 'fit' and 'se.fit', and with 'lwr', and 'upr' if 'interval' is also set.

### Author(s)

Will Lowe

### See Also

`wordfish`

---

## `rescale` *Rescale Estimated Document Positions*

---

### Description

Linearly rescales estimated document positions on the basis of two control points.

### Usage

```
rescale(object, ident = c(1, -1, 10, 1))
```

### Arguments

| | |
|---|---|
| `object` | fitted wordfish or wordscores object |
| `ident` | two documents indexes and and their desired new positions |

### Details

The rescaled positions set document with index ident[1] to position ident[2] and docuemnt with index ident[3] to position ident[4]. The fitted model passed as the first argument is not affected.

### Value

A data frame containing the rescaled document positions with standard errors if available.

### Author(s)

Will Lowe

`sim.wordfish` *Simulate data and parameters for a Wordfish model*

## Description

Simulates data and returns parameter values using Wordfish model assumptions: Counts are sampled under the assumption of independent Poisson draws with log expected means linearly related to a lattice of document positions.

## Usage

```
sim.wordfish(
  docs = 10,
  vocab = 20,
  doclen = 500,
  dist = c("spaced", "normal"),
  scaled = TRUE
)
```

## Arguments

| | |
|---|---|
| `docs` | How many 'documents' should be generated |
| `vocab` | How many 'word' types should be generated |
| `doclen` | A scalar 'document' length or vector of lengths |
| `dist` | the distribution of 'document' positions |
| `scaled` | whether the document positions should be mean 0, unit sd |

## Details

This function draws 'docs' document positions from a Normal distribution, or regularly spaced between 1/'docs' and 1.

'vocab'/2 word slopes are 1, the rest -1. All word intercepts are 0. 'doclen' words are then sampled from a multinomial with these parameters.

Document position (theta) is sorted in increasing size across the documents. If 'scaled' is true it is normalized to mean zero, unit standard deviation. This is most helpful when dist=normal.

## Value

| | |
|---|---|
| `Y` | A sample word-document matrix |
| `theta` | The 'document' positions |
| `doclen` | The 'document' lengths |
| `beta` | 'Word' intercepts |
| `psi` | 'Word' slopes |

### Author(s)

Will Lowe

---

`SP2008` *German Party Manifesto Data*

---

### Description

A random sample of words and their frequency in German political party manifestos from 1990-2005.

### Details

`SP2008` is a `jl_df` object.

### Source

Wordfish website (http://www.wordfish.org)

### References

J. Slapin and S.-O. Proksch (2008) 'A scaling model for estimating time-series party positions from texts' American Journal of Political Science 52(3), 705-722.

---

`SP2008_econ` *Economics sections of German Party Manifestos*

---

### Description

A word frequency matrix from the economic sections of German political party manifestos from 1990-2005.

### Details

`SP2008_econ` is a `jl_df` object

### Source

These data are courtesy of S.-O. Proksch.

### References

J. Slapin and S.-O. Proksch (2008) 'A scaling model for estimating time-series party positions from texts' American Journal of Political Science 52(3), 705-722.

## SP2008_for *Foreign Policy Sections of German Party Manifestos*

### Description

A word frequency matrix from the foreign policy sections of German political party manifestos from 1990-2005.

### Details

SP2008_for is a jl_df object

### Source

These data courtesy of S.-O. Proksch.

### References

J. Slapin and S.-O. Proksch (2008) 'A scaling model for estimating time-series party positions from texts' American Journal of Political Science 52(3), 705-722.

## SP2008_soc *Societal sections of German Party Manifestos*

### Description

A word frequency matrix from the societal sections of German political party manifestos from 1990-2005.

### Details

SP2008_soc is a jl_df object

### Source

These data courtesy of S.-O. Proksch.

### References

J. Slapin and S.-O. Proksch (2008) 'A scaling model for estimating time-series party positions from texts' American Journal of Political Science 52(3), 705-722.

## summary.classic.wordscores *Summarize an Classic Wordscores Model*

### Description

Summarises a Wordscores model

### Usage

```
## S3 method for class 'classic.wordscores'
summary(object, ...)
```

### Arguments

| | |
|---|---|
| object | a fitted wordscores model |
| ... | extra arguments (currently ignored) |

### Details

To see the wordscores, use coef.

### Value

A summary of information about the reference documents used to fit the model.

### Author(s)

Will Lowe

## summary.wordfish *Summarize a Wordfish Model*

### Description

Summarises estimated document positions from a fitted Wordfish model

### Usage

```
## S3 method for class 'wordfish'
summary(object, level = 0.95, ...)
```

### Arguments

| | |
|---|---|
| object | fitted wordfish model |
| level | confidence interval coverage |
| ... | extra arguments, e.g. level |

## Details

if 'level' is passed to the function, e.g. 0.95 for 95 percent confidence, this generates the appropriate width intervals.

## Value

A data.frame containing estimated document position with standard errors and confidence intervals.

## Author(s)

Will Lowe

## See Also

wordfish

---

# trim *Trim a Word Frequency Data*

---

## Description

Ejects low frequency observations and subsamples

## Usage

```
trim(wfm, min.count = 5, min.doc = 5, sample = NULL, verbose = TRUE)
```

## Arguments

| | |
|---|---|
| wfm | an object of class wfm, or a data matrix |
| min.count | the smallest permissible word count |
| min.doc | the fewest permissible documents a word can appear in |
| sample | how many words to randomly retain |
| verbose | whether to say what we did |

## Value

If sample is a number then this many words will be retained after min.doc and min.doc filters have been applied.

## Author(s)

Will Lowe

## See Also

wfm

## ukmanif *UK Manifesto Data*

### Description

UK manifesto data from Laver et al.

### Details

This are word counts from the manifestos of the three main UK parties for the 1992 and 1997 elections.

ukmanif is a word frequency object.

### References

Laver, Benoit and Garry (2003) 'Estimating policy positions from political text using words as data' American Political Science Review 97(2) 311-331.

## wfm *Word Frequency Matrix*

### Description

A word count matrix that know which margin holds the words.

### Usage

```
wfm(mat, word.margin = 1)
```

### Arguments

| | |
|---|---|
| mat | matrix of word counts or the name of a csv file of word counts |
| word.margin | which margin holds the words |

### Details

If mat is a filename it should name a comma separated value format with row labels in the first column and column labels in the first row. Which represents words and which documents is specified by word.margin, which defaults to words as rows.

A word frequency matrix is defined as any two dimensional matrix with non-empty row and column names and dimnames 'words' and 'docs' (in either order). The actual class of such an object is not important for the operation of the functions in this package, so wfm is essentially an interface. The function is.wfm is a (currently rather loose) check whether an object fulfils the interface contract.

For such objects the convenience accessor functions as.docword and as.worddoc can be used to to get counts whichever way up you need them.

words returns the words and docs returns the document titles. wordmargin reminds you which margin contains the words. Assigning wordmargin flips the dimension names.

To get extract particular documents by name or index, use getdocs.

as.wfm attempts to convert things to be word frequency matrices. This functionality is currently limited to objects on which as.matrix already works, and to TermDocument and DocumentTerm objects from the tm package.

## Value

A word frequency matrix from a suitable object, or read from a file if mat is character. Which margin is treated as representing words is set by word.margin.

## Author(s)

Will Lowe

## See Also

as.wfm, as.docword, as.worddoc, docs, words, is.wfm, wordmargin

## Examples

```
mat <- matrix(1:6, ncol = 2)
rownames(mat) <- c('W1','W2','W3')
colnames(mat) <- c('D1','D2')
m <- wfm(mat, word.margin = 1)
getdocs(as.docword(m), 'D2')
```

---

# wfm2bmr *Transform Word Frequency Matrix for BMR/BLR*

---

## Description

Transforms a wfm to the format used by BMR/BLR

## Usage

```
wfm2bmr(y, wfm, filename)
```

## Arguments

| | |
|---|---|
| y | integer dependent variable, may be NULL |
| wfm | a word frequency matrix |
| filename | Name of the file to save data to |

### Details

BMR is sparse matrix format similar to that used by SVMlight

Each line contains an optional dependent variable index and a sequence of indexes and feature value pairs divided by colons. Indexes refer to the words with non-zero counts in the original matrix, and the feature values are the counts.

### Value

A file containing the variables in in sparse matrix format.

### Author(s)

Will Lowe

### See Also

wfm

---

## wfm2lda *Transform Word Frequency Matrix for lda*

---

### Description

Transforms a wfm to the format used by the lda package

### Usage

```
wfm2lda(wfm, dir = NULL, names = c("mult.dat", "vocab.dat"))
```

### Arguments

| | |
|---|---|
| wfm | a word frequency matrix |
| dir | a file to dump the converted data |
| names | Names of the data and vocabulary file respectively |

### Details

See the documentation of lda package for the relevant object structures and file formats.

### Value

A list containing

| | |
|---|---|
| data | zero indexed word frequency information about a set of documents |
| vocab | a vocabulary list |

, unless dir is specified.

If dir is specified then the same information is dumped to 'vocab.dat' and 'mult.dat' in the dir folder.

## Author(s)

Will Lowe

## See Also

wfm

---

wordfish *Estimate a Wordfish Model*

---

## Description

Estimates a Wordfish model using Conditional Maximum Likelihood.

## Usage

```
wordfish(
  wfm,
  dir = c(1, length(docs(wfm))),
  control = list(tol = 1e-06, sigma = 3, startparams = NULL, conv.check = c("ll",
    "cor")),
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| wfm | a word frequency matrix |
| dir | set global identification by forcing theta[dir[1]] < theta[dir[2]] (defaults to first and last document) |
| control | list of estimation options |
| verbose | produce a running commentary |

## Details

Fits a Wordfish model with document ideal points constrained to mean zero and unit standard deviation.

The control list specifies options for the estimation process. conv.check is either 'll' which stops when the difference in log likelihood between iterations is less than tol, or 'cor' which stops when one minus the correlation between the thetas from the current and the previous iterations is less than tol. sigma is the standard deviation for the beta prior in poisson form. startparams is a list of starting values (theta, beta, psi and alpha) or a previously fitted Wordfish model for the same data. verbose generates a running commentary during estimation

The model has two equivalent forms: a poisson model with two sets of document and two sets of word parameters, and a multinomial with two sets of word parameters and document ideal points. The first form is used for estimation, the second is available for alternative summaries, prediction, and profile standard error calculations.

The model is regularized by assuming a prior on beta with mean zero and standard deviation sigma (in poisson form). If you don't want to regularize, set beta to a large number.

## Value

An object of class wordfish. This is a list containing:

| | |
|---|---|
| `dir` | global identification of the dimension |
| `theta` | document positions |
| `alpha` | document fixed effects |
| `beta` | word slope parameters |
| `psi` | word fixed effects |
| `docs` | names of the documents |
| `words` | names of words |
| `sigma` | regularization parameter for betas in poisson form |
| `ll` | final log likelihood |
| `se.theta` | standard errors for document position |
| `data` | the original data |

## Author(s)

Will Lowe

## References

Slapin and Proksch (2008) 'A Scaling Model for Estimating Time-Series Party Positions from Texts.' American Journal of Political Science 52(3):705-772.

## See Also

`plot.wordfish`, `summary.wordfish`, `coef.wordfish`, `fitted.wordfish`, `predict.wordfish`, `sim.wordfish`

## Examples

```
dd <- sim.wordfish()
wf <- wordfish(dd$Y)
summary(wf)
```

## wordmargin *Which margin holds the words*

### Description

Checks which margin (rows or columns) of a Word Frequency Matrix holds the words

### Usage

```
wordmargin(x)
```

### Arguments

| | |
|---|---|
| x | a word frequency matrix |

### Details

Changing the wordmargin by assignment just swaps the dimnames

### Value

1 if words are rows and 2 if words are columns.

### Author(s)

Will Lowe

### See Also

wfm

## words *Extract Words*

### Description

Extracts the words from a wfm object

### Usage

```
words(wfm)

words(wfm) <- value
```

## Arguments

| | |
|---|---|
| `wfm` | an object of type wfm |
| `value` | replacement if assignment |

## Value

A list of words.

## Author(s)

Will Lowe

## See Also

`wfm`, `docs`

# Index


∗ **datasets**
daildata, 9
demanif, 9
demanif.econ, 10
demanif.foreign, 10
demanif.soc, 11
iebudget2009, 14
interestgroups, 15
K2009, 16
LB2002, 17
LB2013, 17
lbg, 18
LBG2003, 18
LG2000, 19
SP2008, 25
SP2008_econ, 25
SP2008_for, 26
SP2008_soc, 26
ukmanif, 29

as.docword, 3, *4*, *13*, *29, 30*
as.wfm, 3, *13*, *30*
as.worddoc, *3*, 4, *13*, *29, 30*
austin, 5

bootstrap.se, 5

classic.wordscores, 6, *7*, *18, 19*, *22*
coef.classic.wordscores, 7
coef.wordfish, 8, *33*

daildata, 9
demanif, 9
demanif.econ, 10
demanif.foreign, 10
demanif.soc, 11
docs, 11, *13*, *30*, *35*
docs<- *(*docs*)*, 11

extractwords, 12

fitted.wordfish, 12, *33*

getdocs, 13, *30*

iebudget2009, 14
iebudget2009cov *(*iebudget2009*)*, 14
initialize.urfish, 14
interestgroups, 15
is.wfm, *13*, 16, *29, 30*

K2009, 16

LB2002, 17
LB2013, 17
lbg, 18
LBG2003, 18
LG2000, 19

plot.classic.wordscores, 19
plot.coef.wordfish, 20
plot.wordfish, 20, *33*
predict.classic.wordscores, 21
predict.wordfish, 22, *33*

rescale, 23

sim.wordfish, 24, *33*
SP2008, 25
SP2008_econ, 25
SP2008_for, 26
SP2008_soc, 26
summary.classic.wordscores, *6*, 27
summary.wordfish, 27, *33*

trim, 28

ukmanif, 29

wfm, *3, 4*, *12*, *16*, *28*, 29, *31, 32*, *34, 35*
wfm2bmr, 30
wfm2lda, 31

wordfish, *8*, *20, 21*, *23*, *28*, 32
wordmargin, *13*, *30*, 34
words, *13*, *30*, 34
words<- *(*words*)*, 34