Package ‘corpora’

February 19, 2015

Type Package
Title Statistics and data sets for corpus frequency data
Version 0.4-3
Depends R (>= 2.10.0)
Date 2012-04-04
Author Stefan Evert [http://purl.org/stefan.evert]
Maintainer Stefan Evert <evert@linglit.tu-darmstadt.de>
Description Utility functions and data sets for the statistical
analysis of corpus frequency data, used in the SIGIL statistics
course.
License GPL-3
URL http://SIGIL.R-Forge.R-Project.org/
LazyData yes
Encoding UTF-8
Repository CRAN
Date/Publication 2012-04-04 13:26:36
NeedsCompilation no

R topics documented:
corpora-package .................................................. 2
binom.pval ....................................................... 3
BNCbiber .......................................................... 4
BNCcomparison .................................................... 6
BNCdomains ...................................................... 7
BNCInChargeOf ................................................... 8
BNCmeta .......................................................... 9
chisq .............................................................. 10
chisq.pval ....................................................... 12
cont.table ....................................................... 13
fisher.pval ....................................................... 14
Description

The *corpora* provides some convenience functions and example data sets for statistical inference from corpus frequency data.

It is a companion package for the SIGIL course (*Statistical Inference - a Gentle Introduction for Linguists and similar creatures*) developed by Marco Baroni and Stefan Evert.

Details

**ToDo:** overview of functions and data sets in package

Author(s)

Stefan Evert <<stefan.evert@uos.de>>

References

The official homepage of both the SIGIL course and the corpora package is [http://SIGIL.R-Forge.R-Project.org/](http://SIGIL.R-Forge.R-Project.org/).

See Also

**ToDo:** entry points into corpora documentation

Examples

```r
## TODO: basic usage examples?
```
binom.pval

P-values of the binomial test for frequency counts (corpora)

Description

This function computes the p-value of a binomial test for frequency counts. In the two-sided case, a fast approximation is used that may be inaccurate for small samples.

Usage

```r
binom.pval(k, n, p = 0.5,
    alternative = c("two.sided", "less", "greater"))
```

Arguments

- `k`: frequency of a type in the corpus (or an integer vector of frequencies)
- `n`: number of tokens in the corpus, i.e. sample size (or an integer vector specifying the sizes of different samples)
- `p`: null hypothesis, giving the assumed proportion of this type in the population (or a vector of proportions for different types and/or different populations)
- `alternative`: a character string specifying the alternative hypothesis; must be one of `"two.sided"` (default), `"less"` or `"greater"`

Details

When `alternative` is `"two.sided`, a fast approximation of the two-sided p-value is used (multiplying the appropriate single-sided tail probability by two), which may be inaccurate for small samples. Unlike the exact algorithm of `binom.test`, this implementation can be applied to large frequencies and samples without a serious impact on performance.

Value

The p-value of a binomial test applied to the given data (or a vector of p-values).

Author(s)

Stefan Evert

See Also

`z.score.pval`, `prop.cint`
BNCbiber  

Biber’s (1988) register features for the British National Corpus

Description

This data set contains a table of the relative frequencies (per 1000 words) of 65 linguistic features (Biber 1988, 1995) for each text document in the British National Corpus (Aston & Burnard 1998). Biber (1988) introduced these features for the purpose of a multidimensional register analysis. Variables in the data set are numbered according to Biber’s list (see e.g. Biber 1995, 95f).

Feature frequencies were automatically extracted from the British National Corpus using query patterns based on part-of-speech tags (Gasthaus 2007). Note that features 60 and 65 had to be omitted because they cannot be identified with sufficient accuracy by the automatic methods. For further information on the extraction methodology, see Gasthaus (2007, 20-21). The original data set and the Python scripts used for feature extraction are available from http://cogsci.uni-osnabrueck.de/~CL/research/gasthaus2007/.

Usage

data(BNCbiber)

Format

A data set with 4048 rows and 66 columns, specifying document ID followed by the relative frequencies (per 1000 words) of 65 linguistic features. Documents are listed in the same order as the metadata in BNCmeta, so the two data frames can be merged directly with cbind.

```
id
f_01_past_tense
f_02_perfect_aspect
f_03_present_tense
f_04_place_adverbials
f_05_time_adverbials
f_06_first_person_pronomouns
f_07_second_person_pronomouns
f_08_third_person_pronomouns
f_09_pronom_it
f_10_demonstrative_pronomouns
f_11_indefinite_pronomouns
f_12_proverb_do
f_13_wh_question
f_14_nominalization
```

A. Tense and aspect markers

- Past tense
- Perfect aspect
- Present tense

B. Place and time adverbials

- Place adverbials (e.g., above, beside, outdoors)
- Time adverbials (e.g., early, instantly, soon)

C. Pronouns and pro-verbs

- First-person pronouns
- Second-person pronouns
- Third-person personal pronouns (excluding it)
- Pronoun it
- Demonstrative pronouns (that, this, these, those as pronouns)
- Indefinite pronouns (e.g., anybody, nothing, someone)
- Pro-verb do

D. Questions

- Direct wh-questions

E. Nominal forms

- Nominalizations (ending in -tion, -ment, -ness, -ity)
Gerunds (participial forms functioning as nouns)
Total other nouns

**F. Passives**
Agentless passives
by-passives

**G. Stative forms**
be as main verb
Existential *there*

**H. Subordination features**
*that* verb complements (e.g., I said that he went.)
*wh*-clauses (e.g., I believed what he told me.)

Present participial adverbial clauses (e.g., Stuffing his mouth with cookies, Joe ran out the door.)
Past participial adverbial clauses (e.g., Built in a single week, the house would stand for fifty years.)
Past participial postnominal (reduced relative) clauses (e.g., the solution produced by this process)

**I. Prepositional phrases, adjectives and adverbs**
Total prepositional phrases
Attributive adjectives (e.g., the big horse)
Predicative adjectives (e.g., The horse is big.)
Total adverbs

**J. Lexical specificity**
Type-token ratio (including punctuation)
Average word length (across tokens, excluding punctuation)

**K. Lexical classes**
Conjuncts (e.g., consequently, furthermore, however)
Downtoners (e.g., barely, nearly, slightly)
Hedges (e.g., at about, something like, almost)
Amplifiers (e.g., absolutely, extremely, perfectly)
Emphatics (e.g., a lot, for sure, really)
Discourse particles (e.g., sentence-initial well, now, anyway)

**L. Modals**
Possibility modals (can, may, might, could)
Necessity modals (ought, should, must)
Predictive modals (will, would, shall)

**M. Specialized verb classes**
**f_55.verb.public**  
Public verbs (e.g., assert, declare, mention)

**f_56.verb.private**  
Private verbs (e.g., assume, believe, doubt, know)

**f_57.verb.suasive**  
Suasive verbs (e.g., command, insist, propose)

**f_58.verb.seem**  
seem and appear

**N. Reduced forms and dispreferred structures**

**f_59.contractions**  
Contractions

**n/a**  
Subordinator that deletion (e.g., I think [that] he went.)

**f_61.stranded.preposition**  
Stranded prepositions (e.g., the candidate that I was thinking of)

**f_62.split.infinitive**  
Split infinitives (e.g., He wants to convincingly prove that . . .)

**f_63.split.auxiliary**  
Split auxiliaries (e.g., They were apparently shown to . . .)

**O. Co-ordination**

**f_64.phrasal.coordination**  
Phrasal co-ordination (N and N; Adj and Adj; V and V; Adv and Adv)

**n/a**  
Independent clause co-ordination (clause-initial and)

**P. Negation**

**f_66.neg.synthetic**  
Synthetic negation (e.g., No answer is good enough for Jones.)

**f_67.neg.analytic**  
Analytic negation (e.g., That’s not likely.)

**Author(s)**


**References**

See also the BNC homepage at [http://www.natcorp.ox.ac.uk/](http://www.natcorp.ox.ac.uk/).


**See Also**

-BNCmeta-

---

**BNCcomparison**  
*Comparison of written and spoken frequencies (BNC)*

**Description**

This data set compares the frequencies of 60 selected nouns in the written and spoken parts of the British National Corpus, World Edition (BNC). Nouns were chosen from three frequency bands, namely the 20 most frequent nouns in the corpus, 20 nouns with approximately 1000 occurrences, and 20 nouns with approximately 100 occurrences.

See Aston & Burnard (1998) for more information about the BNC, or go to [http://www.natcorp.ox.ac.uk/](http://www.natcorp.ox.ac.uk/).
**BNCdomains**

Usage

data(BNCcomparison)

Format

A data set with 61 rows and the following columns:

- **noun**: lemmatised noun (aka stem form)
- **written**: frequency in the written part of the BNC
- **spoken**: frequency in the spoken part of the BNC

Details

In addition to the 60 nouns, the data set contains a column labelled OTHER, which represents the total frequency of all other nouns in the BNC. This value is needed in order to calculate the sample sizes of the written and spoken part for frequency comparison tests.

Author(s)

Stefan Evert ([http://purl.org/stefan.evert](http://purl.org/stefan.evert))

References


---

**BNCdomains** Distribution of domains in the British National Corpus (BNC)

Description

This data set gives the number of documents and tokens in each of the 18 domains represented in the British National Corpus, World Edition (BNC). See Aston & Burnard (1998) for more information about the BNC and the domain classification, or go to [http://www.natcorp.ox.ac.uk/](http://www.natcorp.ox.ac.uk/).

Usage

data(BNCdomains)

Format

A data set with 19 rows and the following columns:

- **domain**: name of the respective domain in the BNC
- **documents**: number of documents from this domain
- **tokens**: total number of tokens in all documents from this domain
Details

For one document in the BNC, the domain classification is missing. This document is represented by the code Unlabeled in the data set.

Author(s)

Marco Baroni (<baroni@sslmit.unibo.it>)

References


Description

This data set lists collocations (in the sense of Sinclair 1991) of the phrase *in charge of* found in the British National Corpus, World Edition (BNC). A span size of 3 and a frequency threshold of 5 were used, i.e. all words that occur at least five times within a distance of three tokens from the key phrase *in charge of* are listed as collocates. Note that collocations were not allowed to cross sentence boundaries.

See Aston & Burnard (1998) for more information about the BNC, or go to http://www.natcorp.ox.ac.uk/.

Usage

data(BNCInChargeOf)

Format

A data set with 250 rows and the following columns:

collocate: a collocate of the key phrase *in charge of* (word form)

f.in: occurrences of the collocate within a distance of 3 tokens from the key phrase, i.e. inside the span

N.in: total number of tokens inside the span

f.out: occurrences of the collocate outside the span

N.out: total number of tokens outside the span

Details

Punctuation, numbers and any words containing non-alphabetic characters (except for ~) were not considered as potential collocates. Likewise, the number of tokens inside / outside the span given in the columns N.in and N.out only includes simple alphabetic word forms.
BNCmeta

Author(s)
Stefan Evert (http://purl.org/stefan.evert)

References

Description
This data set provides complete metadata for all 4048 texts of the British National Corpus (XML edition). See Aston & Burnard (1998) for more information about the BNC, or go to http://www.natcorp.ox.ac.uk/.
The data have automatically been extracted from the original BNC source files. Some transformations were applied so that all attribute names and their values are given in a human-readable form. The Perl scripts used in the extraction procedure are available from http://cwb.sourceforge.net/download.php#import.

Usage
data(BNCmeta)

Format
A data set with 4048 rows and the columns listed below. Unless specified otherwise, columns are coded as factors.

id: BNC document ID; character vector
title: Title of the document; character vector
n_words: Number of words in the document; integer vector
n_tokens: Total number of tokens (including punctuation and deleted material); integer vector
n_w: Number of w-units (words); integer vector
n_c: Number of c-units (punctuation); integer vector
n_s: Number of s-units (sentences); integer vector
publication_date: Publication date
text_type: Text type
context: Spoken context
respondent_age: Age-group of respondent
respondent_class: Social class of respondent (NRS social grades)
respondent_sex: Sex of respondent
interaction_type: Interaction type
region: Region
author_age: Author age-group
author_domicile: Domicile of author
author_sex: Sex of author
author_type: Author type
audience_age: Audience age
domain: Written domain
difficulty: Written difficulty
medium: Written medium
publication_place: Publication place
sampling_type: Sampling type
circulation: Estimated circulation size
audience_sex: Audience sex
availability: Availability
mode: Text mode (written/spoken)
derived_type: Text class
genre: David Lee’s genre classification

Author(s)
Stefan Evert (http://purl.org/stefan.evert)

References

---

chisq

Pearson’s chi-squared statistic for frequency comparisons (corpora)

Description
This function computes Pearson’s chi-squared statistic (often written as $X^2$) for frequency comparison data, with or without Yates’ continuity correction. The implementation is based on the formula given by Evert (2004, 82).

Usage

chisq(k1, n1, k2, n2, correct = TRUE, one.sided=FALSE)
**chisq**

**Arguments**

- `k1`: frequency of a type in the first corpus (or an integer vector of type frequencies)
- `n1`: the sample size of the first corpus (or an integer vector specifying the sizes of different samples)
- `k2`: frequency of the type in the second corpus (or an integer vector of type frequencies, in parallel to `k1`)
- `n2`: the sample size of the second corpus (or an integer vector specifying the sizes of different samples, in parallel to `n1`)
- `correct`: if TRUE, apply Yates’ continuity correction (default)
- `one.sided`: if TRUE, compute the signed square root of $X^2$ as a statistic for a one-sided test (see details below; the default value is FALSE)

**Details**

The $X^2$ values returned by this function are identical to those computed by `chisq.test`. Unlike the latter, `chisq` accepts vector arguments so that a large number of frequency comparisons can be carried out with a single function call.

The one-sided test statistic (for `one.sided=TRUE`) is the signed square root of $X^2$. It is positive for $k_1/n_1 > k_2/n_2$ and negative for $k_1/n_1 < k_2/n_2$. Note that this statistic has a standard normal distribution rather than a chi-squared distribution under the null hypothesis of equal proportions.

**Value**

The chi-squared statistic $X^2$ corresponding to the specified data (or a vector of $X^2$ values). This statistic has a chi-squared distribution with $df = 1$ under the null hypothesis of equal proportions.

**Author(s)**

Stefan Evert

**References**


**See Also**

`chisq.pval`, `chisq.test`, `cont.table`
Description

This function computes the p-value of Pearson's chi-squared test for the comparison of corpus frequency counts (under the null hypothesis of equal population proportions). It is based on the chi-squared statistic $X^2$ implemented by the chisq function.

Usage

chisq.pval(k1, n1, k2, n2, correct = TRUE,
           alternative = c("two.sided", "less", "greater"))

Arguments

k1 frequency of a type in the first corpus (or an integer vector of type frequencies)

n1 the sample size of the first corpus (or an integer vector specifying the sizes of different samples)

k2 frequency of the type in the second corpus (or an integer vector of type frequencies, in parallel to k1)

n2 the sample size of the second corpus (or an integer vector specifying the sizes of different samples, in parallel to n1)

correct if TRUE, apply Yates' continuity correction (default)

alternative a character string specifying the alternative hypothesis; must be one of two.sided (default), less or greater

Details

The p-values returned by this functions are identical to those computed by chisq.test (two-sided only) and prop.test (one-sided and two-sided) for two-by-two contingency tables.

Value

The p-value of Pearson’s chi-squared test applied to the given data (or a vector of p-values).

Author(s)

Stefan Evert

See Also

chisq.fisher.pval, chisq.test, prop.test
Build contingency tables for frequency comparison (corpora)

Description

This is a convenience function which constructs 2x2 contingency tables needed for frequency comparisons with `chisq.test`, `fisher.test` and similar functions.

Usage

```r
cont.table(k1, n1, k2, n2, as.list=NA)
```

Arguments

- `k1`: frequency of a type in the first corpus, a numeric scalar or vector
- `n1`: the size of the first corpus (sample size), a numeric scalar or vector
- `k2`: frequency of the type in the second corpus, a numeric scalar or vector
- `n2`: the size of the second corpus (sample size), a numeric scalar or vector
- `as.list`: whether multiple contingency tables can be constructed and are returned as a list (see "Details" below)

Details

If all four arguments `k1` `n1` `k2` `n2` are scalars (vectors of length 1), `cont.table` constructs a single contingency table, i.e. a 2x2 matrix. If at least one argument has length > 1, shorter vectors are replicated as necessary, and a list of 2x2 contingency tables is constructed.

With `as.list=TRUE`, the return value is always a list, even if it contains just a single contingency table. With `as.list=FALSE`, only scalar arguments are accepted and the return value is guaranteed to be a 2x2 matrix.

Value

A numeric matrix containing a two-by-two contingency table for the specified frequency comparison, or a list of such matrices (see "Details").

Author(s)

Stefan Evert

See Also

`chisq.test`, `fisher.test`
Description
This function computes the p-value of Fisher’s exact test (Fisher 1934) for the comparison of corpus frequency counts (under the null hypothesis of equal population proportions). In the two-sided case, a fast approximation is used that may be inaccurate for small samples.

Usage

```r
fisher.pval(k1, n1, k2, n2, alternative = c("two.sided", "less", "greater"))
```

Arguments

- `k1`: frequency of a type in the first corpus (or an integer vector of type frequencies)
- `n1`: the sample size of the first corpus (or an integer vector specifying the sizes of different samples)
- `k2`: frequency of the type in the second corpus (or an integer vector of type frequencies, in parallel to `k1`)
- `n2`: the sample size of the second corpus (or an integer vector specifying the sizes of different samples, in parallel to `n1`)
- `alternative`: a character string specifying the alternative hypothesis; must be one of `two.sided` (default), `less` or `greater`

Details
When `alternative` is `two.sided`, a fast approximation of the two-sided p-value is used (multiplying the appropriate single-sided tail probability by two), which may be inaccurate for small samples. Unlike the exact algorithm of `fisher.test`, this implementation is memory-efficient and can be applied to large samples and/or large frequency counts.

For one-sided tests, the p-values returned by this functions are identical to those computed by `fisher.test` on two-by-two contingency tables.

Value
The p-value of Fisher’s exact test applied to the given data (or a vector of p-values).

Author(s)
Stefan Evert
References

See Also
fisher.test, chisq.pval

## prop.cint

*Confidence interval for proportion based on frequency counts (corpora)*

### Description
This function computes a confidence interval for a population proportion from the corresponding frequency count in a corpus. The confidence interval can be based on a binomial test or on a z-score test (with or without continuity correction).

### Usage

```r
prop.cint(k, n, method = c("binomial", "z.score"), correct = TRUE,
          conf.level = 0.95, alternative = c("two.sided", "less", "greater"))
```

### Arguments

- **k**: frequency of a type in the corpus (or an integer vector of frequencies)
- **n**: number of tokens in the corpus, i.e. sample size (or an integer vector specifying the sizes of different samples)
- **method**: a character string specifying whether the confidence interval is based on the binomial test (binomial) or the z-score test (z.score)
- **correct**: if TRUE, apply Yates’ continuity correction for the z-score test (default)
- **conf.level**: the desired confidence level (defaults to 95%)
- **alternative**: a character string specifying the alternative hypothesis, yielding a two-sided (two.sided, default), lower one-sided (less) or upper one-sided (greater) confidence interval

### Details
The confidence intervals computed by this function correspond to those returned by `binom.test` and `prop.test`, respectively. However, `prop.cint` accepts vector arguments, allowing many confidence intervals to be computed with a single function call. In addition, it uses a fast approximation of the two-sided binomial test that can safely be applied to large samples.
The confidence interval for a z-score test is computed by solving the z-score equation

\[ \frac{k - np}{\sqrt{np(1 - p)}} = \alpha \]

for \( p \), where \( \alpha \) is the \( z \)-value corresponding to the chosen confidence level (e.g. ±1.96 for a two-sided test with 95% confidence). This leads to the quadratic equation

\[ p^2(n + \alpha^2) + p(-2k - \alpha^2) + \frac{k^2}{n} = 0 \]

whose two solutions correspond to the lower and upper boundary of the confidence interval.

When Yates’ continuity correction is applied, the value \( k \) in the numerator of the \( z \)-score equation has to be replaced by \( k^* \), with \( k^* = k - 1/2 \) for the lower boundary of the confidence interval (where \( k > np \)) and \( k^* = k + 1/2 \) for the upper boundary of the confidence interval (where \( k < np \)). In each case, the corresponding solution of the quadratic equation has to be chosen (i.e., the solution with \( k > np \) for the lower boundary and vice versa).

**Value**

A data frame with two columns, labelled `lower` for the lower boundary and `upper` for the upper boundary of the confidence interval. The number of rows is determined by the length of the longest input vector (\( k, n \) and \( \text{conf.level} \)).

**Author(s)**

Stefan Evert

**See Also**

`z.score.pval`, `prop.test`, `binom.pval`, `binom.test`

---

**Description**

This function takes a random sample of rows from a data frame, in analogy to the built-in function `sample` (which sadly does not accept a data frame).

**Usage**

```r
sample.df(df, size, replace=FALSE, sort=FALSE, prob=NULL)
```
Arguments

df  a data frame to be sampled from
size positive integer giving the number of rows to choose
replace Should sampling be with replacement?
sort Should rows in sample be sorted in original order?
prob a vector of probability weights for obtaining the elements of the vector being sampled

Details

Internally, rows are selected with the function `sample.int`. See its manual page for details on the arguments (except for `sort`) and implementation.

Value

A data frame containing the sampled rows of `df`, either their original order (`sort=TRUE`) or shuffled randomly (`sort=FALSE`).

Author(s)

Stefan Evert

---

**simulated.census**  
__Simulated census data for examples and illustrations (corpora)__

Description

This function generates a large simulated census data set with body measurements (height, weight, shoe size) for male and female inhabitants of a highly fictitious country.

The generated data set is usually named `fakecensus` (see code examples below) and is used for various exercises and illustrations in the SIGIL course.

Usage

```
simulated.census(N=502202, p.male=0.55, seed.rng=42)
```

Arguments

N population size, i.e. number of inhabitants of the fictitious country
p.male proportion of males in the country
seed.rng seed for the random number generator, so data sets with the same parameters (N, p.male, etc.) are reproducible
Details

The default population size corresponds to the estimated populace of Luxembourg on 1 January 2010 (according to http://en.wikipedia.org/wiki/Luxembourg).

Further parameters of the simulation (standard deviation, correlations, non-linearity) will be exposed as function arguments in future releases.

Value

A data set with \( N \) rows corresponding to inhabitants and the following columns:

- **height**: body height in cm
- **height**: body weight in kg
- **shoe.size**: shoe size in Paris points (Continental European scale)
- **sex**: sex, either \( m \) or \( f \)

Author(s)

Stefan Evert (http://purl.org/stefan.evert)

Examples

```r
FakeCensus <- simulated.census()
summary(FakeCensus)
```

Description

This function generates type and token counts, token-type ratios (TTR) and average word length for simulated articles from the English Wikipedia. Simulation parameters are based on data from the Wackypedia corpus.

The generated data set is usually named WackypediaStats (see code examples below) and is used for various exercises and illustrations in the SIGIL course.

Usage

```r
simulated.wikipedia(N=1429649, length=c(100,1000), seed.rng=42)
```
Arguments

- **N**: population size, i.e. total number of Wikipedia articles
- **length**: a numeric vector of length 2, specifying the typical range of Wikipedia article lengths
- **seed.rng**: seed for the random number generator, so data sets with the same parameters (N and length) are reproducible

Details

The default population size corresponds to the subset of the Wackypedia corpus from which the simulation parameters were obtained. This excludes all articles with extreme type-token statistics (very short, very long, extremely long words, etc.).

Article lengths are sampled from a lognormal distribution which is scaled so that the central 95% of the values fall into the range specified by the length argument.

The simulated data are surprising close to the original Wackypedia statistics.

Value

A data set with N rows corresponding to Wikipedia articles and the following columns:

- **tokens**: number of word tokens in the article
- **types**: number of distinct word types in the article
- **ttr**: token-type ratio (TTR) for the article
- **avglen**: average word length in characters (averaged across tokens)

Author(s)

Stefan Evert (http://purl.org/stefan.evert)

References

The Wackypedia corpus can be obtained from http://wacky.sslmit.unibo.it/doku.php?id=corpora.

Examples

```r
WackypediaStats <- simulated.wikipedia()
summary(WackypediaStats)
```
Description

This data set contains a small corpus (8043 tokens) of short stories from the collection Very Short Stories (VSS, see http://www.schtepfd.de/pages/stories.html). The text was automatically segmented (tokenised) and annotated with part-of-speech tags (from the Penn tagset) and lemmas (base forms), using the IMS TreeTagger (Schmid 1994).

Usage
data(VSS)

Format

A data set with 8043 rows corresponding to tokens and the following columns:

- **word**: the word form (or surface form) of the token
- **pos**: the part-of-speech tag of the token (using the Penn tagset)
- **word**: the lemma (or base form) of the token

Details

The Penn tagset defines the following part-of-speech tags:

- **CC**: Coordinating conjunction
- **CD**: Cardinal number
- **DT**:Determiner
- **EX**: Existential there
- **FW**: Foreign word
- **IN**: Preposition or subordinating conjunction
- **JJ**: Adjective
- **JJR**: Adjective, comparative
- **JJS**: Adjective, superlative
- **LS**: List item marker
- **MD**: Modal
- **NN**: Noun, singular or mass
- **NNS**: Noun, plural
- **NP**: Proper noun, singular
- **NPS**: Proper noun, plural
- **PDT**: Predeterminer
- **POS**: Possessive ending
- **PP**: Personal pronoun
- **PP$**: Possessive pronoun
- **RB**: Adverb
- **RBR**: Adverb, comparative
**z.score**

<table>
<thead>
<tr>
<th>Code</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBS</td>
<td>Adverb, superlative</td>
</tr>
<tr>
<td>RP</td>
<td>Particle</td>
</tr>
<tr>
<td>SYM</td>
<td>Symbol</td>
</tr>
<tr>
<td>TO</td>
<td><em>to</em></td>
</tr>
<tr>
<td>UH</td>
<td>Interjection</td>
</tr>
<tr>
<td>VB</td>
<td>Verb, base form</td>
</tr>
<tr>
<td>VBD</td>
<td>Verb, past tense</td>
</tr>
<tr>
<td>VBG</td>
<td>Verb, gerund or present participle</td>
</tr>
<tr>
<td>VBN</td>
<td>Verb, past participle</td>
</tr>
<tr>
<td>VBP</td>
<td>Verb, non-3rd person singular present</td>
</tr>
<tr>
<td>VBZ</td>
<td>Verb, 3rd person singular present</td>
</tr>
<tr>
<td>WDT</td>
<td>Wh-determiner</td>
</tr>
<tr>
<td>WP</td>
<td>Wh-pronoun</td>
</tr>
<tr>
<td>WP$</td>
<td>Possessive wh-pronoun</td>
</tr>
<tr>
<td>WRB</td>
<td>Wh-adverb</td>
</tr>
</tbody>
</table>

**Author(s)**

Stefan Evert ([http://purl.org/stefan.evert](http://purl.org/stefan.evert))

**References**


---

**z.score**

*The z-score statistic for frequency counts (corpora)*

**Description**

This function computes a z-score statistic for frequency counts, based on a normal approximation to the correct binomial distribution under the random sampling model.

**Usage**

```
z.score(k, n = 0.5, p = 0.5, correct = TRUE)
```

**Arguments**

- `k`: frequency of a type in the corpus (or an integer vector of frequencies)
- `n`: number of tokens in the corpus, i.e. sample size (or an integer vector specifying the sizes of different samples)
- `p`: null hypothesis, giving the assumed proportion of this type in the population (or a vector of proportions for different types and/or different populations)
- `correct`: if TRUE, apply Yates’ continuity correction (default)
Details

The \( z \) statistic is given by

\[
z := \frac{k - np}{\sqrt{np(1 - p)}}
\]

When Yates’ continuity correction is enabled, the absolute value of the numerator \( d := k - np \) is reduced by \( 1/2 \), but clamped to a non-negative value.

Value

The \( z \)-score corresponding to the specified data (or a vector of \( z \)-scores).

Author(s)

Stefan Evert

See Also

\( zNscoreNpval \)

---

### \( zscore.pval \)

\( P \)-values of the \( z \)-score test for frequency counts (corpora)

Description

This function computes the \( P \)-value of a \( z \)-score test for frequency counts, based on the \( z \)-score statistic implemented by \( z.score \).

Usage

\[
z.score.pval(k, n, p = 0.5, correct = TRUE, alternative = c("two.sided", "less", "greater"))
\]

Arguments

- \( k \) frequency of a type in the corpus (or an integer vector of frequencies)
- \( n \) number of tokens in the corpus, i.e. sample size (or an integer vector specifying the sizes of different samples)
- \( p \) null hypothesis, giving the assumed proportion of this type in the population (or a vector of proportions for different types and/or different populations)
- \( correct \) if \( TRUE \), apply Yates’ continuity correction (default)
- \( alternative \) a character string specifying the alternative hypothesis; must be one of \( \text{two.sided} \) (default), \( \text{less} \) or \( \text{greater} \)
z.score.pval

Value
The p-value of a z-score test applied to the given data (or a vector of p-values).

Author(s)
Stefan Evert

See Also
z.score.binom.pval, prop.cint
Index

*Topic **array**
  cont.table, 13

*Topic **datasets**
  BNCbiber, 4
  BNCcomparison, 6
  BNCdomains, 7
  BNCInChargeOf, 8
  BNCmeta, 9
  simulated.census, 17
  simulated.wikipedia, 18
  VSS, 20

*Topic **htest**
  binom.pval, 3
  chisq, 10
  chisq.pval, 12
  cont.table, 13
  fisher.pval, 14
  prop.cint, 15
  z.score, 21
  z.score.pval, 22

*Topic **manip**
  sample.df, 16

*Topic **package**
  corpora-package, 2

  binom.pval, 3, 16, 23
  binom.test, 3, 15, 16
  BNCbiber, 4
  BNCcomparison, 6
  BNCdomains, 7
  BNCInChargeOf, 8
  BNCmeta, 4, 6, 9

  chisq, 10, 12
  chisq.pval, 11, 12, 15
  chisq.test, 11–13
  cont.table, 11–13
  corpora (corpora-package), 2
  corpora-package, 2

FakeCensus (simulated.census), 17
fisher.pval, 12, 14
fisher.test, 13–15
prop.cint, 3, 15, 23
prop.test, 12, 15, 16
sample.df, 16
sample.int, 17
simulated.census, 17
simulated.wikipedia, 18
VSS, 20
WackypediaStats (simulated.wikipedia), 18
z.score, 21, 22, 23
z.score.pval, 3, 16, 22, 22