

# Package ‘corpora’

April 4, 2012

**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](mailto: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

## 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
prop.cint . . . . .	15
sample.df . . . . .	16
simulated.census . . . . .	17
simulated.wikipedia . . . . .	18
VSS . . . . .	20
z.score . . . . .	21
z.score.pval . . . . .	22
<b>Index</b>	<b>24</b>

---

corpora-package	<i>corpora: statistical inference from corpus frequency data</i>
-----------------	--

---

## 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/>.

## See Also

**ToDo:** entry points into corpora documentation

## Examples

## 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**

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

**Arguments**

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

**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](#)

**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`.

<code>id</code>	BNC document ID (character vector)
	<b>A. Tense and aspect markers</b>
<code>f_01_past_tense</code>	Past tense
<code>f_02_perfect_aspect</code>	Perfect aspect
<code>f_03_present_tense</code>	Present tense
	<b>B. Place and time adverbials</b>
<code>f_04_place_adverbials</code>	Place adverbials (e.g., <i>above, beside, outdoors</i> )
<code>f_05_time_adverbials</code>	Time adverbials (e.g., <i>early, instantly, soon</i> )
	<b>C. Pronouns and pro-verbs</b>
<code>f_06_first_person_pronouns</code>	First-person pronouns
<code>f_07_second_person_pronouns</code>	Second-person pronouns
<code>f_08_third_person_pronouns</code>	Third-person personal pronouns (excluding <i>it</i> )
<code>f_09_pronoun_it</code>	Pronoun <i>it</i>
<code>f_10_demonstrative_pronoun</code>	Demonstrative pronouns ( <i>that, this, these, those</i> as pronouns)
<code>f_11_indefinite_pronoun</code>	Indefinite pronouns (e.g., <i>anybody, nothing, someone</i> )
<code>f_12_proverb_do</code>	Pro-verb <i>do</i>
	<b>D. Questions</b>
<code>f_13_wh_question</code>	Direct <i>wh</i> -questions
	<b>E. Nominal forms</b>
<code>f_14_nominalization</code>	Nominalizations (ending in <i>-tion, -ment, -ness, -ity</i> )

f_15_gerunds	Gerunds (participial forms functioning as nouns)
f_16_other_nouns	Total other nouns
	<b>F. Passives</b>
f_17_agentless_passives	Agentless passives
f_18_by_passives	<i>by</i> -passives
	<b>G. Stative forms</b>
f_19_be_main_verb	<i>be</i> as main verb
f_20_existential_there	Existential <i>there</i>
	<b>H. Subordination features</b>
f_21_that_verb_comp	<i>that</i> verb complements (e.g., <i>I said that he went.</i> )
f_22_that_adj_comp	<i>that</i> adjective complements (e.g., <i>I'm glad that you like it.</i> )
f_23_wh_clause	<i>wh</i> -clauses (e.g., <i>I believed what he told me.</i> )
f_24_infinitives	Infinitives
f_25_present_participle	Present participial adverbial clauses (e.g., <i>Stuffing his mouth with cookies, Joe ran out the door.</i> )
f_26_past_participle	Past participial adverbial clauses (e.g., <i>Built in a single week, the house would stand for years.</i> )
f_27_past_participle_whiz	Past participial postnominal (reduced relative) clauses (e.g., <i>the solution produced by this method</i> )
f_28_present_participle_whiz	Present participial postnominal (reduced relative) clauses (e.g., <i>the event causing this death</i> )
f_29_that_subj	<i>that</i> relative clauses on subject position (e.g., <i>the dog that bit me</i> )
f_30_that_obj	<i>that</i> relative clauses on object position (e.g., <i>the dog that I saw</i> )
f_31_wh_subj	<i>wh</i> relatives on subject position (e.g., <i>the man who likes popcorn</i> )
f_32_wh_obj	<i>wh</i> relatives on object position (e.g., <i>the man who Sally likes</i> )
f_33_pied_piping	Pied-piping relative clauses (e.g., <i>the manner in which he was told</i> )
f_34_sentence_relatives	Sentence relatives (e.g., <i>Bob likes fried mangoes, which is the most disgusting thing I've ever eaten</i> )
f_35_because	Causative adverbial subordinator ( <i>because</i> )
f_36_though	Concessive adverbial subordinators ( <i>although, though</i> )
f_37_if	Conditional adverbial subordinators ( <i>if, unless</i> )
f_38_other_adv_sub	Other adverbial subordinators (e.g., <i>since, while, whereas</i> )
	<b>I. Prepositional phrases, adjectives and adverbs</b>
f_39_prepositions	Total prepositional phrases
f_40_adj_attr	Attributive adjectives (e.g., <i>the big horse</i> )
f_41_adj_pred	Predicative adjectives (e.g., <i>The horse is big.</i> )
f_42_adverbs	Total adverbs
	<b>J. Lexical specificity</b>
f_43_type_token	Type-token ratio (including punctuation)
f_44_mean_word_length	Average word length (across tokens, excluding punctuation)
	<b>K. Lexical classes</b>
f_45_conjuncts	Conjuncts (e.g., <i>consequently, furthermore, however</i> )
f_46_downtoners	Downtoners (e.g., <i>barely, nearly, slightly</i> )
f_47_hedges	Hedges (e.g., <i>at about, something like, almost</i> )
f_48_amplifiers	Amplifiers (e.g., <i>absolutely, extremely, perfectly</i> )
f_49_emphatics	Emphatics (e.g., <i>a lot, for sure, really</i> )
f_50_discourse_particles	Discourse particles (e.g., sentence-initial <i>well, now, anyway</i> )
f_51_demonstratives	Demonstratives
	<b>L. Modals</b>
f_52_modal_possibility	Possibility modals ( <i>can, may, might, could</i> )
f_53_modal_necessity	Necessity modals ( <i>ought, should, must</i> )
f_54_modal_predictive	Predictive modals ( <i>will, would, shall</i> )
	<b>M. Specialized verb classes</b>

f_55_verb_public	Public verbs (e.g., <i>assert, declare, mention</i> )
f_56_verb_private	Private verbs (e.g., <i>assume, believe, doubt, know</i> )
f_57_verb_suasive	Suasive verbs (e.g., <i>command, insist, propose</i> )
f_58_verb_seem	<i>seem</i> and <i>appear</i>
	<b>N. Reduced forms and dispreferred structures</b>
f_59_contractions	Contractions
n/a	Subordinator <i>that</i> deletion (e.g., <i>I think [that] he went.</i> )
f_61_stranded_preposition	Stranded prepositions (e.g., <i>the candidate that I was thinking of</i> )
f_62_split_infinitive	Split infinitives (e.g., <i>He wants to convincingly prove that ...</i> )
f_63_split_auxiliary	Split auxiliaries (e.g., <i>They were apparently shown to ...</i> )
	<b>O. Co-ordination</b>
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 <i>and</i> )
	<b>P. Negation</b>
f_66_neg_synthetic	Synthetic negation (e.g., <i>No answer is good enough for Jones.</i> )
f_67_neg_analytic	Analytic negation (e.g., <i>That's not likely.</i> )

**Author(s)**

Stefan Evert (<http://purl.org/stefan.evert>); data extracted by Jan Gasthaus (2007).

**References**

- Aston, Guy and Burnard, Lou (1998). *The BNC Handbook*. Edinburgh University Press, Edinburgh. See also the BNC homepage at <http://www.natcorp.ox.ac.uk/>.
- Biber, Douglas (1988). *Variations Across Speech and Writing*. Cambridge University Press, Cambridge.
- Biber, Douglas (1995). *Dimensions of Register Variation: A cross-linguistic comparison*. Cambridge University Press, Cambridge.
- Gasthaus, Jan (2007). *Prototype-Based Relevance Learning for Genre Classification*. B.Sc.\ thesis, Institute of Cognitive Science, University of Osnabrück. Data sets and software available from <http://cogsci.uni-osnabrueck.de/~CL/research/gasthaus2007/>.

**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/>.

**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>)

**References**

Aston, Guy and Burnard, Lou (1998). *The BNC Handbook*. Edinburgh University Press, Edinburgh. See also the BNC homepage at <http://www.natcorp.ox.ac.uk/>.

---

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/>.

**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**

Aston, Guy and Burnard, Lou (1998). *The BNC Handbook*. Edinburgh University Press, Edinburgh. See also the BNC homepage at <http://www.natcorp.ox.ac.uk/>.

---

BNCInChargeOf

*Collocations of the phrase "in charge of" (BNC)*

---

**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.

**Author(s)**

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

**References**

Aston, Guy and Burnard, Lou (1998). *The BNC Handbook*. Edinburgh University Press, Edinburgh. See also the BNC homepage at <http://www.natcorp.ox.ac.uk/>.

Sinclair, John (1991). *Corpus, Concordance, Collocation*. Oxford University Press, Oxford.

---

BNCmeta

*Metadata for the British National Corpus (XML edition)*

---

**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**

Aston, Guy and Burnard, Lou (1998). *The BNC Handbook*. Edinburgh University Press, Edinburgh.  
See also the BNC homepage at <http://www.natcorp.ox.ac.uk/>.

---

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)
```

**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 <i>signed square root</i> 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**

Evert, Stefan (2004). *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. Ph.D. thesis, Institut für maschinelle Sprachverarbeitung, University of Stuttgart. Published in 2005, URN urn:nbn:de:bsz:93-opus-23714. Available from <http://www.collocations.de/phd.html>.

**See Also**

[chisq.pval](#), [chisq.test](#), [cont.table](#)

---

chisq.pval	<i>P-values of Pearson's chi-squared test for frequency comparisons (corpora)</i>
------------	---

---

### 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 <code>two.sided</code> (default), <code>less</code> or <code>greater</code>

### 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`

---

cont.table	<i>Build contingency tables for frequency comparison (corpora)</i>
------------	--

---

### Description

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

### Usage

```
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](#)

---

`fisher.pval`*P-values of Fisher's exact test for frequency comparisons (corpora)*

---

**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**

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

**Arguments**

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

**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**

Fisher, R. A. (1934). *Statistical Methods for Research Workers*. Oliver & Boyd, Edinburgh, 2nd edition (1st edition 1925, 14th edition 1970).

**See Also**

[fisher.test](#), [chisq.pval](#)

---

prop.cint	<i>Confidence interval for proportion based on frequency counts (corpora)</i>
-----------	---

---

**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**

```
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.  $\pm 1.96$  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 `conf.level`).

### Author(s)

Stefan Evert

### See Also

[z.score.pval](#), [prop.test](#), [binom.pval](#), [binom.test](#)

---

sample.df

*Random samples from data frames (corpora)*

---

### 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

```
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	<i>Simulated census data for examples and illustrations (corpora)</i>
------------------	---

---

**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

weight: 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**

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

---

simulated.wikipedia    *Simulated type and token counts for Wikipedia articles (corpora)*

---

**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**

```
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**

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

VSS

*A small corpus of very short stories with linguistic annotations***Description**

This data set contains a small corpus (8043 tokens) of short stories from the collection *Very Short Stories* (VSS, see <http://www.schtep.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 <i>there</i>
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

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

**Author(s)**

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

**References**

Schmid, Helmut (1994). Probabilistic part-of-speech tagging using decision trees. In: *Proceedings of the International Conference on New Methods in Language Processing (NeMLaP)*, pages 44-49.

---

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, 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**

[z.score.pval](#)

---

z.score.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 two.sided (default), less or greater

**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