Package `languageR`

February 20, 2015

**Type** Package

**Title** Data sets and functions with "Analyzing Linguistic Data: A practical introduction to statistics".

**Version** 1.4.1

**Date** 2013-12-9

**Author** R. H. Baayen

**Maintainer** R. H. Baayen <harald.baayen@uni-tuebingen.de>


**LazyData** yes

**License** GPL (>= 2)

**Depends** R(>= 3.0.2)

**Suggests** cluster, e1071, rms, Hmisc, MASS, rpart, lattice, zipfR, lme4, multcomp, lmerTest, optimx

**Imports** methods

**Repository** CRAN

**Date/Publication** 2013-12-12 01:27:42

**NeedsCompilation** no

**R topics documented:**

languageR-package .......................................................... 4
acf.fnc ................................................................. 6
affixProductivity .......................................................... 7
alice ................................................................. 10
aovlmer.fnc .......................................................... 10
auxiliaries ......................................................... 11
beginningReaders ...................................................... 12
collin.fnc .......................................................... 13
R topics documented:

- compare.richness.fnc
- corres-class
- corres.fnc
- corSup.fnc
- danish
- dative
- dativeSimplified
- degreesOrKnots.fnc
- durationsGe
- durationsOnt
- dutchSpeakersDist
- dutchSpeakersDistMeta
- english
- etymology
- faz
- finalDevoicing
- getKnots.fnc
- getMCMCintervals.fnc
- getPos.fnc
- getRange.fnc
- getRoot.fnc
- growth-class
- growth.fnc
- growth2vgc.fnc
- havelaar
- heid
- herdan.fnc
- imaging
- implementInteractions.fnc
- item.fnc
- items.quasif.fnc
- lags.fnc
- latinsquare
- lexdec
- lexicalMeasures
- lexicalMeasuresClasses
- lmerPlotInt.fnc
- make.reg.fnc
- makeDefaultMatrix.fnc
- makeSplineData.fnc
- moby
- mvrnormplot.fnc
- nesscg
- nessdemog
- nessw
- oldFrench
- oldFrenchMeta
- oz
R topics documented:

pairs.fnc .................................................. 71
parsePredName.fnc ...................................... 72
periphrasticDo ........................................... 73
phylogeny .................................................. 75
plot.corres ............................................... 79
plot.growth ............................................... 81
plotAll.fnc ............................................... 82
plotLMER.fnc ............................................ 83
plotlogistic.fit.fnc ..................................... 89
preparePredictor.fnc ................................... 90
primingHeid ............................................. 92
primingHeidPrevRT ..................................... 93
print.corres ............................................. 94
print.growth ............................................ 95
pvals.fnc ................................................ 96
quasif .................................................... 97
quasiF.fnc ............................................... 98
quasiFsim.fnc .......................................... 99
ratings ................................................... 100
regularity ............................................... 101
selfPacedReadingHeid .................................. 103
shadenormal.fnc ....................................... 105
show.growth ............................................ 106
shrinkage ............................................... 107
simulateLatinsquare.fnc .............................. 108
simulateQuasif.fnc ..................................... 109
simulateRegression.fnc ............................... 110
sizeRatings .............................................. 112
spanish .................................................. 113
spanishFunctionWords ................................. 114
spanishMeta ............................................. 115
spectrum.fnc ............................................ 116
splitplot ............................................... 117
subjects.latinsquare.fnc ............................ 118
subjects.quasif.fnc ................................... 119
summary.corres ....................................... 120
summary.growth ........................................ 121
tail.growth .............................................. 122
text2spc.fnc ............................................. 122
through ................................................ 123
transforming.fnc ....................................... 124
twente .................................................. 125
variationLijk ........................................... 125
ver ....................................................... 126
verbs .................................................... 127
warlpiri .................................................. 128
weightRatings .......................................... 129
writtenVariationLijk .................................. 130
Description


Details

Package: languageR
Type: Package
Version: 1.0
Date: 2007-01-15
License: GNU public license

The main function of this package is to make available the data sets discussed and analyzed in 'Analyzing Linguistic Data: A practical introduction to statistics using R', to appear with Cambridge University Press. The following packages should be installed, as ancillary functions in this package depend on them.

- **zipfR** for word frequency distributions
- **lme4** for mixed-effects models
- **coda** for Markov-Chain Monte Carlo estimation
- **lattice** for trellis graphics
- **Matrix** for mixed-effects modeling

The following packages need to be installed for working through specific examples.

- **rms** for regression modeling
- **rpart** for CART trees
- **e1071** for support vector machines
- **MASS** for many useful functions
- **ape** for phylogenetic clustering

The main convenience functions in this library are, by category:

**correspondence analysis** (extending code by Murtagh, 2005)

```
corres.fnc correspondence analysis
```
languageR-package

corSup.fnc supplementary data

**vocabulary richness** (supplementing current zipfR functionality)

`compare.richness.fnc` for two texts, compare richness
`growth.fnc` empirical vocabulary growth data for text
`growth2vgc` conversion to vgc object of zipfR
`spectrum.fnc` creates frequency spectrum
`text2spc.fnc` conversion to spc object of zipfR

**lmer functions** (p-values for mixed-effects models with lme4)

`pvals.fnc` p-values for table of coefficients including MCMC
`aovlmer.fnc` p-values for anova tables and/or MCMC p-value for specified factor

**simulation functions** (for comparing mixed models with traditional techniques including F1, F2, and F1+F2)

`simulateRegression.fnc` simulate simple regression design
`simulateQuasif.fnc` simulate data for Quasi-F ratios
`simulateLatinSquare.fnc` simulating simple Latin-square design

**miscellaneous** (convenience functions)

`pairsCor.fnc` scatterplot matrix with correlation tests
`collin.fnc` collinearity diagnostics
`pvals.fnc` p-values and MCMC confidence intervals for mixed models
`plot.logistic.fit.fnc` diagnostic visualization for logistic models
`xyLowess.fnc` trellis scatterplots with smoother
`mvnormPlot.fnc` scatterplot for bivariate standard normal random numbers with regression line

`lmerPlotInt.fnc` offers choice of four ways to visualize an interaction between two numeric predictors in an lmer model

**Author(s)**

R. H. Baayen
University of Alberta, Edmonton, Canada
<harald.baayen@gmail.com>
Maintainer: harald.baayen@gmail.com

**References**


**Examples**

```r
## Not run:
library(languageR)
data(package="languageR")

## End(Not run)```
acf.fnc  
Autocorrelation trellis graph

Description

This function creates a trellis plot with autocorrelation functions for by-subject sequential dependencies in response latencies.

Usage

acf.fnc(dat, group="Subject", time="Trial", x = "RT", plot=TRUE, ...)

Arguments

dat         A data frame with (minimally) a grouping factor, an index for successive trails/events, and a behavioral measure

group       A grouping factor such as Subject

time        A sequential time measure such as Trial number in the experimental list

x           The dependent variable, usually a chronometric measure such as RT

plot        If true, a trellis graph is produced, otherwise a data frame with the data on which the trellis graph is based is returned

...         other optional arguments, such as layout

Value

If plot=TRUE, a trellis graph, otherwise a data frame with as column names

Lag          Autocorrelation lag

Acf          Autocorrelation

Subject      The grouping factor, typically Subject

ci           The (approximate) 95% confidence interval.

Author(s)

R. H. Baayen

References


See Also

lags.fnc
Affix productivity, gauged by the P* productivity measure, for 27 English affixes in 44 texts.

Usage

data(affixProductivity)

Format

A data frame with 44 observations on the following 30 variables.

- semi  a numeric vector of P*-values
- anti  a numeric vector of P*-values
- ee    a numeric vector of P*-values
- ism   a numeric vector of P*-values
- ian   a numeric vector of P*-values
- ful   a numeric vector of P*-values
- y     a numeric vector of P*-values
- ness  a numeric vector of P*-values
- able  a numeric vector of P*-values
- ly    a numeric vector of P*-values
- unv   a numeric vector of P*-values
- una   a numeric vector of P*-values
- ize   a numeric vector of P*-values
- less  a numeric vector of P*-values
- erA   a numeric vector of P*-values
- erC   a numeric vector of P*-values
- ity   a numeric vector of P*-values
- super a numeric vector of P*-values
- est   a numeric vector of P*-values
- ment  a numeric vector of P*-values
ify a numeric vector of P*-values
re a numeric vector of P*-values
ation a numeric vector of P*-values
in. a numeric vector of P*-values
ex a numeric vector of P*-values
en a numeric vector of P*-values
be a numeric vector of P*-values
AuthorCodes a factor with levels
    BMo (Book of Mormon)
    CAS (Aesop’s fables, translation by Townsend)
    CBo (Baum, The Marvelous Land of Oz)
    CBp (Barrie, Peter Pan and Wendy)
    CBw (Baum, The Wonderful Wizard of Oz)
    CCa (Carroll, Alice’s Adventures in Wonderland)
    CCT (Carroll, Through the Looking Glass and what Alice Found There)
    CGr (Grimm Fairy Tales, translations)
    CKJ (Kipling, The Jungle Book)
    LAP (Austen, Pride and Prejudice)
    Lbp (Burroughs, A Princess of Mars)
    Lbw (Bronte, Wuthering Heights)
    LCl (Conrad, Lord Jim)
    Lcn (Conrad, Nigger of the Narcissus)
    LDb (Doyle, The Casebook of Sherlock Holmes)
    Ldc (Dickens, The Chimes: a Goblin Story)
    LDC (Dickens, A Christmas Carol)
    LDh (Doyle, The Hound of the Baskervilles)
    LDv (Doyle, The Valley of Fear)
    LJc (James, Confidence)
    LJc (James, The Europeans)
    LLC (London, The Call of the Wild)
    LLs (London, The Sea Wolf)
    LMA (Montgomery, Anne of Avonlea)
    LMP (Melville, Moby Dick)
    LMr (Morris, News from Nowhere)
    LMP (Milton, Paradise Lost)
    L0s (Orczy, The Scarlet Pimpernel)
    Lsd (Stoker, Dracula)
    Lss (Chu, More than a Chance Meeting (Startrek))
    LTa (Trollope, Ayala’s Angel)
    LTE (Trollope, The Eustace Diamonds)
affixProductivity

LTf (Trollope, Can you Forgive her?)
LTy (Twain, A Connecticut Yankee in King Arthur’s Court)
LWi (Wells, The Invisible Man)
LWt (Wells, The Time Machine)
LWW (Wells, The War of the Worlds)
OAF (The Federalist Papers)
OCh (Texts sampled from Congress Hearings)
OC1 (Texts sampled from Clinton’s Election Speeches)
ODo (Darwin, On the Origin of the Species)
OGa (Selected Texts from the Government Accounting Office)
OJe (James, Essays in Radical Empiricism)

Registers a factor with levels B (Biblical texts) C (Children’s books) L (Literary texts) O (other)
Birth a numeric vector for the author’s year of birth (where available)

Source

Most texts were obtained from the Gutenberg Project (http://www.gutenberg.org/wiki/Main_Page) and the Oxford Text Archive (http://ota.ahds.ac.uk/).

References


Examples

```r
## Not run:
data(affixProductivity)
affixes.pr = prcomp(affixProductivity[,1:ncol(affixProductivity)-3]],
center = TRUE, scale. = TRUE)
library(lattice)
trellis.device()
super.sym = trellis.par.get("superpose.symbol")
splom(data.frame(affixes.pr[,1:3]),
groups = affixProductivity$Registers,
panel = panel.superpose,
key = list(title = "texts in productivity space",
text = list(c("Religious", "Children", "Literary", "Other")),
points = list(pch = super.sym$pch[1:4], col = super.sym$col[1:4])))
## End(Not run)
```
alice

Alice’s Adventures in Wonderland

Description
The text of Lewis Carroll’s ‘Alice’s Adventures in Wonderland’, with punctuation marks removed.

Usage
data(alice)

Format
A character vector with 27269 words.

Source
The project Gutenberg at http://www.gutenberg.org/wiki/Main_Page

Examples
data(alice)
alice[1:5]

aovlmer.fnc

Compute p-values for factors in mixed models

Description
This function no longer works with recent versions of lme4. For p-values, see the anova() function in the lmerTest package.

Usage
aovlmer.fnc(object, ...)

Arguments
object An lmer or glmer model for a response variable fitted with lmer.
... Other optional arguments.

Value
A warning message.
**Description**

For 285 regular and irregular Dutch verbs, the auxiliary for the present and past perfect is listed together with the count of verbal synsets in WordNet. Regular and irregular verbs are matched in the mean for lemma frequency.

**Usage**

data(auxiliaries)

**Format**

A data frame with 285 observations on the following 4 variables.

- **Verb** a factor with 285 monomorphemic Dutch verbs.
- **Aux** a factor with as levels the auxiliaries hebben, zijn and zijnheb (for verbs allowing both auxiliaries).
- **Verbalsynsets** a numeric vector with the number of verbal synonym sets in WordNet in which the verb is listed.
- **Regularity** a factor with levels irregular and regular.
References


Examples

data(auxiliaries)
kruskal.test(auxiliaries$VerbalSynsets, auxiliaries$Aux)

beginningReaders

Visual lexical decision with beginning readers

Description

Visual lexical decision latencies for beginning readers (8 year-old Dutch children).

Usage

data(beginningReaders)

Format

A data frame with 7923 observations on the following 13 variables.

Word a factor for the words.

Subject a factor for the subjects.

LogRT a numeric vector with the log-transformed reaction time (in ms).

Trial a numeric vector coding the rank of the trial in the experimental list.

OrthLength a numeric vector coding the word’s length in letters.

LogFrequency a numeric vector with log-transformed frequency in Vermeer’s frequency dictionary of Dutch children’s texts.

LogFamilySize a numeric vector with the log-transformed morphological family size count (with family members judged to be unknown to young children removed).

ReadingScore a numeric vector with a score for reading proficiency.

ProportionOfErrors a numeric vector for the proportion of error responses for the word.

PC1 a numeric vector for the first principal component of a PCA orthogonalization of the preceding 4 reaction times

PC2 a numeric vector for the second principal component of a PCA orthogonalization of the preceding 4 reaction times

PC3 a numeric vector for the third principal component of a PCA orthogonalization of the preceding 4 reaction times

PC4 a numeric vector for the fourth principal component of a PCA orthogonalization of the preceding 4 reaction times
References


Examples

```r
## Not run:
data(beginningreaders)
require(lme4)
require(optimx)
require(lmerTest)

beginningReaders.lmer = lmer(LogRT ~ PC1 + PC2 + PC3 + ReadingScore + OrthLength + I(OrthLength^2) + LogFrequency + LogFamilySize + (1|Word) + (1|Subject) + (0+LogFrequency|Subject) + (0+OrthLength|Subject) + (0+PC1|Subject),
data = beginningReaders,
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")))
summary(beginningReaders.lmer)

## End(Not run)
```

collin.fnc  

*Calculate condition number with intercept included*

Description

Calculates the condition number with the intercept included, following Belsley, Kuh and Welsch (1980).

Usage

```r
collin.fnc(data, colvector)
```

Arguments

- **data**: A data frame.
- **colvector**: A vector with the column numbers in the data frame for which the collinearity is to be assessed. Only numeric predictors allowed.

Value

A list with components

- **svd**: Singular value decomposition
- **cindex**: Condition indices
- **cnumber**: The condition number
- **pi**: The phi matrix
Author(s)

F. J. Tweedie

References


See Also

kappa

Examples

```r
## Not run:
data(english)
collin.fnc(english[english$AgeSubj=="young",], 7:29)$cnumber

## End(Not run)
```

Description

Comparisons of lexical richness between two texts are carried out on the basis of the vocabulary size (number of types) and on the basis of the vocabulary growth rate. Variances of the number of types and of the number of hapax legomena required for the tests are estimated with the help of LNRE models.

Usage

```r
compare.richness.fnc(text1, text2, digits = 5)
```

Arguments

text1 First text in the comparison.
text2 Second text in the comparison.
digits Number of decimal digits required for the growth rate.
Details

The comparison for the vocabulary size is carried out with the test statistic

\[ Z = \frac{E[V_1] - E[V_2]}{\sqrt{\sigma(V_1)^2 + \sigma(V_2)^2}} \]

and the comparison of the growth rates with the test statistic

\[ Z = \frac{\frac{1}{N_1} E[V_1(1)] - \frac{1}{N_2} E[V_2]}{\sqrt{\frac{1}{N_1^2} \sigma(V_1(1))^2 + \frac{1}{N_2^2} \sigma(V_2(1))^2}} \]

where \( N \) denotes the sample size in tokens, \( V \) the vocabulary size, and \( V(1) \) the number of hapax legomena.

Value

A summary listing the Chi-Squared measure of goodness of fit for the LNRE models (available in the zipfR package) used to estimate variances, a table listing tokens, types, hapax legomena and the vocabulary growth rate, and two-tailed tests for differences in the vocabulary sizes and growth rates with Z-score and p-value.

Note

It is probably unwise to attempt to apply this function to texts comprising more than 500,000 words.

Author(s)

R. Harald Baayen Radboud University Nijmegen and Max Planck Institute for Psycholinguistics, Nijmegen, The Netherlands. baayen@mpi.nl

References


Examples

```r
## Not run:
data(alice, through, oz)
compare.richness.fnc(tolower(alice), tolower(through[1:length(alice)]))
compare.richness.fnc(tolower(alice), tolower(oz[1:25942]))
## End(Not run)
```
Description

A class for correspondence analysis

Objects from the Class

Objects can be created by calls of the form `new("corres", ...). Correspondence objects can be plotted, summarized, and printed.

Slots

data: Object of class "list"

Methods

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

Note

to be expanded

Author(s)

R. H. Baayen

References

Murtagh

See Also

See Also corres.fnc.

Examples

`showClass("corres")`
**corres.fnc**  

**Correspondence Analysis**

**Description**

Correspondence analysis for a contingency table.

**Usage**

`corres.fnc(xtab)`

**Arguments**

- `xtab`: A data frame cross-tabulating frequencies.

**Value**

A correspondence object with summary and plot methods. The summary method lists eigenvalue rates and coordinates, correlations and contributions for Factor 1 and Factor 2. By default, only the first six rows of the factor tables are shown. Full tables are obtained by specifying `header = FALSE` when calling `summary`. For information on higher dimensions, set the option `n` to the desired number (e.g., `n = 3`) within `summary`. See `plot.corres` for documentation of plot options.

**Author(s)**

Extension of the code in Murtagh (2005) by R. Harald Baayen

Radboud University Nijmegen & Max Planck Institute for Psycholinguistics

Nijmegen, The Netherlands

email: baayen@mpi.nl

**References**


**See Also**

See `corsup.fnc` for adding supplementary data to a correspondence plot, and `plot.corres` for plot options.

**Examples**

```r
## Not run:
data(oldFrench)
oldFrench.ca = corres.fnc(oldFrench)
oldFrench.ca
summary(oldFrench.ca, head = TRUE)
```
Supplementary rows or columns in correspondence analysis

Description

Corsup calculates supplementary rows or columns for correspondence analysis.

Usage

corsup.fnc(corres, bycol = TRUE, supp, plot = TRUE, font = 3, labels = "", cex = 1)

Arguments

corres     A correspondence object.
bycol      A logical value indicating whether supplementary columns (the default) or supplementary rows are required.
supp       Supplementary rows or columns from a data frame with the same structure as the data frame used for the corres object.
plot A logical value indicating whether supplementary rows or columns should be added to an already existing plot.

font An integer specifying the font to be used for plotting.

labels A character vector with row or column names to be used for plotting.

cex A real specifying the font size required for plotting.

Value

If plot = FALSE, a matrix with the supplementary coordinates. Otherwise, supplementary rows or columns are added to an already existing plot of a correspondence object.

Author(s)

Extension of the code in Murtagh (2005) by R. Harald Baayen
Radboud University Nijmegen & Max Planck Institute for Psycholinguistics
Nijmegen, The Netherlands
email: baayen@mpi.nl

References

F. Murtagh (2005) Correspondence Analysis and Data Coding with JAVA and R, Chapman & Hall/CRC, Boca Raton, FL.

See Also
corres.fnc

Examples

```r
## Not run:
data(oldFrench)
data(oldFrenchMeta)
prose = oldFrench[oldFrenchMeta$Genre=="prose" & !is_na(oldFrenchMeta$Year),]
proseinfo = oldFrenchMeta[oldFrenchMeta$Genre=="prose" & !is_na(oldFrenchMeta$Year),]
proseinfo$Period = as.factor(proseinfo$Year <= 1250)

prose.ca = corres.fnc(prose)
plot(prose.ca, addcol = FALSE, rcol = as.numeric(proseinfo$Period) + 1, rlabels = proseinfo$Year, rcex = 0.7)

proseSup = oldFrench[oldFrenchMeta$Genre == "prose" & !is_na(oldFrenchMeta$Year),]
corsup.fnc(prose.ca, bycol = FALSE, supp = proseSup, font = 2, cex = 0.8, labels = substr(rownames(proseSup), 1, 4))
```

## End(Not run)
Danish auditory lexical decision

Description
Auditory lexical decision latencies for Danish complex words.

Usage
data(danish)

Format
A data frame with 3326 observations on the following 16 variables.

Subject a random-effect factor coding participants in the experiment.
Word a random-effect factor coding the words for which auditory lexical decision responses were elicited.
Affix a random-effect factor coding the affixes in the words.
LogRT the dependent variable, log response latency.
PC1 first principal component orthogonalizing the four response latencies preceding the current trial in the experiment.
PC2 second principal component orthogonalizing the four response latencies preceding the current trial in the experiment.
PrevError factor with levels CORRECT and ERROR coding whether the preceding trial elicited a correct lexical decision.
Rank the trial number in the experiment.
Sex factor coding the sex of the participant, with levels F (female) and M (male).
LogWordFreq log-transformed word frequency.
LogAffixFreq log-transformed affix frequency.
ResidFamSize residualized morphological family size (taking out LogWordFreq and LogAffixFreq).
ResidSemRating residualized semantic rating (taking out morphological family size).
LogCUP log-transformed complex uniqueness point (CUP).
LogUP log-transformed uniqueness point (UP).
LogCUPtoEnd log of the distance (in msec) from the CUP to the end of the word.

References
Examples

```r
## Not run:
data(danish)
require(lme4)
require(lmerTest)
require(optimx)

### Mixed-effects regression with three random intercepts

danish.lmer = lmer(LogRT ~ PC1 + PC2 + PrevError + Rank +
    ResidSemRating + ResidFamSize + LogWordFreq*LogAffixFreq*Sex +
    poly(LogCUP, 2, raw=TRUE) + LogUP + LogCUPtoEnd +
    (1|Subject) + (1|Word) + (1|Affix), data = danish,
    control=lmerControl(optimizer="optimx", optCtrl=list(method="nlminb")))

danish.lmerA = lmer(LogRT ~ PC1 + PC2 + PrevError + Rank +
    ResidSemRating + ResidFamSize + LogWordFreq*LogAffixFreq*Sex +
    poly(LogCUP, 2, raw=TRUE) + LogUP + LogCUPtoEnd +
    (1|Subject) + (1|Word) + (1|Affix), data = danish,
    control=lmerControl(optimizer="optimx", optCtrl=list(method="nlminb")),
    subset=abs(scale(resid(danish.lmer)))<2.5)

summary(danish.lmerA)

## End(Not run)
```

---

**dative**

**Dative Alternation**

**Description**

Data describing the realization of the dative as NP or PP in the Switchboard corpus and the Treebank Wall Street Journal collection.

**Usage**

data(dative)

**Format**

A data frame with 3263 observations on the following 15 variables.

- **Speaker** a factor coding speaker; available only for the subset of spoken English.
- **Modality** a factor with levels spoken, written.
- **Verb** a factor with the verbs as levels.
- **SemanticClass** a factor with levels a (abstract: 'give it some thought'), c (communication: 'tell, give me your name'), f (future transfer of possession: 'owe, promise'), p (prevention of possession: 'cost, deny'), and t (transfer of possession: 'give an armband, send').
LengthOfRecipient a numeric vector coding the number of words comprising the recipient.

AnimacyOfRec a factor with levels animate and inanimate for the animacy of the recipient.

DefinOfRec a factor with levels definite and indefinite coding the definiteness of the recipient.

PronomOfRec a factor with levels nonpronominal and pronominal coding the pronominality of the recipient.

LengthOfTheme a numeric vector coding the number of words comprising the theme.

AnimacyOfTheme a factor with levels animate and inanimate coding the animacy of the theme.

DefinOfTheme a factor with levels definite and indefinite coding the definiteness of the theme.

PronomOfTheme a factor with levels nonpronominal and pronominal coding the pronominality of the theme.

RealizationOfRecipient a factor with levels NP and PP coding the realization of the dative.

AccessOfRec a factor with levels accessible, given, and new coding the accessibility of the recipient.

AccessOfTheme a factor with levels accessible, given, and new coding the accessibility of the theme.

References


Examples

```r
## Not run:
data(dative)

# analysis with CART tree

library(rpart)

# ---- initial tree

dative.rp = rpart(RealizationOfRecipient ~ .,
    data = dative[,,-c(1, 3)]) # exclude the columns with subjects, verbs
plot(dative.rp, compress = TRUE, branch = 1, margin = 0.1)
text(dative.rp, use.n = TRUE, pretty = 0)

# ---- pruning the initial tree

plotcp(dative.rp)
dative.rp1 = prune(dative.rp, cp = 0.041)
plot(dative.rp1, compress = TRUE, branch = 1, margin = 0.1)
text(dative.rp1, use.n = TRUE, pretty = 0)

# analysis with logistic regression
```
# ---- logistic regression with the rms package

library(rms)
dative.dd = datadist(dative)
options(datadist = 'dative.dd')
dative.lrm = lrm(RealizationOfRecipient ~
  AccessOfTheme + AccessOfRec + LengthOfRecipient + AnimacyOfRec +
  AnimacyOfTheme + PronomOfTheme + DefinOfTheme + LengthOfTheme +
  SemanticClass + Modality, data = dative)
anova(dative.lrm)
plot(Predict(dative.lrm))

# ---- mixed-effects logistic regression with the lme4 package

require(lme4)
require(lmerTest)
require(optimx)

dative.lmer = glmer(RealizationOfRecipient ~ AccessOfTheme +
  AccessOfRec + LengthOfRecipient + AnimacyOfRec +
  AnimacyOfTheme + PronomOfTheme + DefinOfTheme + LengthOfTheme +
  SemanticClass + Modality + (1|Verb),
  control=glmerControl(optimizer="optimx",optCtrl=list(method="nlminb")),
  data = dative, family = "binomial")

summary(dative.lmer)

# multiple comparisons for Accessibility of Theme
require(multcomp)
par(mar=c(5,8,3,1))
AcOfTheme.glht <- glht(dative.lmer, linfct = mcp(AccessOfTheme = "Tukey"))
plot(AcOfTheme.glht)
abline(v=0)
summary(AcOfTheme.glht)

## End(Not run)
Format

A data frame with 903 observations on the following 5 variables.

Verb a factor with the verbs as levels.
AnimacyOfRec a factor with levels animate and inanimate for the animacy of the recipient.
LengthOfTheme a numeric vector coding the number of words comprising the theme.
AnimacyOfTheme a factor with levels animate and inanimate coding the animacy of the theme.
RealizationOfRec a factor with levels NP and PP coding the realization of the dative.

References


Examples

```r
## Not run:
data(dative)

## End(Not run)
```

---

degreesOrKnots.fnc Extract degree of polynomial or knots for restricted cubic spline

Description

Extract degree of polynomial or knots for restricted cubic spline from the predictor name

Usage

degreesOrKnots.fnc(name)

Arguments

name name of predictor, e.g. poly(X, 2, raw = TRUE)

Details

attempts to find degrees or knots if present in input name

Value

Returns an integer for degrees or knots

Note

not intended for independent use
**durationsGe**

**Author(s)**
R. H. Baayen

**See Also**
See Also as `plotLMER.fnc`

**Examples**

```r
## Not run: not intended for independent use
```

---

**durationsGe**  
*Durational measurements on the Dutch prefix ge-*

**Description**

Durational measurements on the Dutch prefix *ge-* in the Spoken Dutch Corpus.

**Usage**

data(durationsGe)

**Format**

A data frame with 428 observations on the following 8 variables.

- **Word**  
a factor with the words as levels.
- **Frequency**  
a numeric vector with the word’s absolute frequency in the Spoken Dutch Corpus.
- **Speaker**  
a factor with the speakers as levels.
- **Sex**  
a factor with levels *female* and *male*, this information is missing for one speaker.
- **YearOfBirth**  
a numeric vector with years of birth.
- **DurationOfPrefix**  
a numeric vector with the duration of the prefix -ont in seconds.
- **SpeechRate**  
a numeric vector coding speech rate in number of syllables per second.
- **NumberSegmentsOnset**  
a numeric vector for the number of segments in the onset of the stem.

**References**

Durational measurements on the Dutch prefix ont-

Description

Durational measurements on the Dutch prefix ont- in the Spoken Dutch Corpus.

Usage

data(durationsOnt)

Format

A data frame with 102 observations on the following 11 variables.

- Word  a factor with the words as levels.
- Frequency  a numeric vector with the word’s logarithmically transformed frequency in the Spoken Dutch Corpus.
- Speaker  a factor with speakers as levels.
- Sex  a factor with levels female and male.
- YearOfBirth  a numeric vector coding year of birth of the speaker - 1900.
- DurationOfPrefix  a numeric vector for the duration of ont- in seconds
- DurationOfPrefixVowel  a numeric vector for the duration of the vowel in the prefix in seconds.
- DurationOfPrefixNasal  a numeric vector for the duration of the nasal in the prefix in seconds.
- DurationOfPrefixPlosive  a numeric vector for the duration of the plosive in the prefix in seconds.
- NumberOfSegmentsOnset  a numeric vector for the number of segments in the onset of the stem.
- PlosivePresent  a factor with levels no and yes for whether the plosive is realized in the signal.
- SpeechRate  a numeric vector coding speech rate in number of syllables per second.
References


Examples

data(durationsOnt)

    # #### modeling the duration of the prefix

prefix.lm = lm(DurationOfPrefix ~ (YearOfBirth + SpeechRate) * Frequency,
                data = durationsOnt)
summary(prefix.lm)

    # ---- model criticism

plot(prefix.lm)
outliers = c(36, 35, 17, 72)
prefix.lm = lm(DurationOfPrefix ~ (YearOfBirth + SpeechRate) * Frequency,
                data = durationsOnt[-outliers,])
summary(prefix.lm)

    # #### modeling the presence of the /t/

library(rms)
durationsOnt.dd = datadist(durationsOnt)
options(datadist = 'durationsOnt.dd')

plosive.lrm = lrm(PlosivePresent ~ SpeechRate + YearOfBirth,
                 data = durationsOnt, x = TRUE, y = TRUE)
plosive.lrm
validate(plosive.lrm, bw = TRUE, B = 200)

    # #### modeling the duration of the /n/

nasal.lm = lm(DurationPrefixNasal ~ PlosivePresent + Frequency +
                YearOfBirth, data = durationsOnt)
summary(nasal.lm)

    # ---- model criticism

plot(nasal.lm)
outliers = c(71, 28, 62, 33)
nasal.lm = lm(DurationPrefixNasal ~ PlosivePresent + Frequency +
                YearOfBirth, data = durationsOnt[-outliers,])
summary(nasal.lm)
**dutchSpeakersDist**

**Description**

A distance matrix for the conversations of 165 speakers in the Spoken Dutch Corpus. Metadata on the speakers are available in a separate dataset, `dutchSpeakersDistMeta`.

**Usage**

```r
data(dutchSpeakersDist)
```

**Format**

A data frame for a 165 by 165 matrix of between-speaker differences.

**Source**

http://lands.let.kun.nl/cgn/ data collected and analyzed in collaboration with Patrick Juola

**References**


**Examples**

```r
## Not run:
data(dutchSpeakersDist)
dutchSpeakersDist.d = as.dist(dutchSpeakersDist)
dutchSpeakersDist.mds = cmdscale(dutchSpeakersDist.d, k = 3)

data(dutchSpeakersDistMeta)
dat = data.frame(dutchSpeakersDist.mds, Sex = dutchSpeakersDistMeta$Sex, Year = dutchSpeakersDistMeta$AgeYear, EduLevel = dutchSpeakersDistMeta$EduLevel)
dat = dat[!is.na(dat$Year),]

par(mfrow=c(1,2))
plot(dat$Year, dat$X1, xlab="year of birth", ylab = "dimension 1", type = "p")
lines(lowess(dat$Year, dat$X1))
boxplot(dat$X3 ~ dat$Sex, ylab = "dimension 3")
par(mfrow=c(1,1))

cor.test(dat$X1, dat$Year, method="sp")
t.test(dat$X3 ~ dat$Sex)

## End(Not run)
```
**dutchSpeakersDistMeta**  
*Metadata for dutchSpeakersDist*

### Description
Meta-data for the cross-entropy based between-speaker distance matrix `dutchSpeakersDist`

### Usage
```r
data(dutchSpeakersDistMeta)
```

### Format
A data frame with 165 observations on the following 6 variables.

- **Speaker**  a factor with speakers as levels.
- **Sex**  a factor with levels `female` and `male`.
- **AgeYear**  a numeric vector with the speakers' year of birth.
- **AgeGroup**  a factor with levels `age18to24`, `age25to34`, `age35to44`, `age45to55`, and `age56up`.
- **ConversationType**  a factor with levels `femaleOnly`, `maleFemale`, `maleOnly`, and `unknown`.
- **EduLevel**  a factor with levels `eduUnknown`, `high`, `low mid`

### Source
http://lands.let.kun.nl/cgn/

### References

### Examples
```r
## Not run:
data(dutchSpeakersDistMeta)
## End(Not run)
```
Description

This data set gives mean visual lexical decision latencies and word naming latencies to 2284 monomorphemic English nouns and verbs, averaged for old and young subjects, with various predictor variables.

Usage

data(english)

Format

A data frame with 4568 observations on the following variables.

- rtlexdec: numeric vector of log RT in visual lexical decision.
- rtnaming: numeric vector of log RT in word naming.
- familiarity: numeric vector of subjective familiarity ratings.
- word: a factor with 2284 words.
- agesubject: a factor with levels young and old.
- wordcategory: a factor with levels N (noun) and V (verb).
- writtenfrequency: numeric vector with log frequency in the CELEX lexical database.
- writtenspokenfrequencyratio: numeric vector with the logged ratio of written frequency (CELEX) to spoken frequency (British National Corpus).
- familysize: numeric vector with log morphological family size.
- derivationalentropy: numeric vector with derivational entropy.
- inflectionalentropy: numeric vector with inflectional entropy.
- numbersimplexsynsets: numeric vector with the log-transformed count of synonym sets in WordNet in which the word is listed.
- numbercomplexsynsets: numeric vector with the log-transformed count of synonym sets in WordNet in which the word is listed as part of a compound.
- lengthinletters: numeric vector with length of the word in letters.
- ncount: numeric vector with orthographic neighborhood density, defined as the number of lemmas in CELEX with the same length (in letters) at Hamming distance 1.
- meanbigramfrequency: numeric vector with mean log bigram frequency.
- frequencyinitialdiphone: numeric vector with log frequency of initial diphone.
- conspelv: numeric vector with type count of orthographic neighbors.
- conspeln: numeric vector with token count of orthographic neighbors.
- conphonv: numeric vector with type count of phonological neighbors.
ConphonN numeric vector with token count of phonological neighbors.
ConfriendsV numeric vector with type counts of consistent words.
ConfriendsN numeric vector with token counts of consistent words.
ConffV numeric vector with type count of forward inconsistent words
ConffN numeric vector with token count of forward inconsistent words
ConfbV numeric vector with type count of backward inconsistent words
ConfbN numeric vector with token count of backward inconsistent words
NounFrequency numeric vector with the frequency of the word used as noun.
Verbfrequency numeric vector with the frequency of the word used as verb.
CV factor specifying whether the initial phoneme of the word is a consonant (C) or a vowel (V).
Obstruent factor specifying whether the initial phoneme of the word is a continuant (cont) or an obstruent (obst).
Frication factor specifying whether the initial phoneme has a burst (burst) or frication (frication) for consonant-initial words, and for vowel-initial words whether the vowel is long or short.
Voice factor indicating whether the initial phoneme is voiced or voiceless.
FrequencyInitialDiphoneWord numeric vector with the log-transformed frequency of the initial diphone given that it is syllable-initial.
FrequencyInitialDiphoneSyllable numeric vector with the log-transformed frequency of the initial diphone given that it is word initial.
CorrectLexdec numeric vector with the proportion of subjects that accepted the item as a word in lexical decision.

Source

References

Examples
## Not run:
data(english)

# ---- orthogonallize orthographic consistency measures
items = english[english$AgeSubject == "young",]
etymology

Etymological age and regularity in Dutch

Description

Estimated etymological age for regular and irregular monomorphemic Dutch verbs, together with other distributional predictors of regularity.
Usage
data(etymology)

Format
A data frame with 285 observations on the following 14 variables.

Verb a factor with the verbs as levels.

WrittenFrequency a numeric vector of logarithmically transformed frequencies in written Dutch (as available in the CELEX lexical database).

NcountStem a numeric vector for the number of orthographic neighbors.

MeanBigramFrequency a numeric vector for mean log bigram frequency.

InflectionalEntropy a numeric vector for Shannon’s entropy calculated for the word’s inflectional variants.

Auxiliary a factor with levels hebben, zijn and zijnheb for the verb’s auxiliary in the perfect tenses.

Regularity a factor with levels irregular and regular.

LengthInLetters a numeric vector of the word’s orthographic length.

Denominative a factor with levels den and n specifying whether a verb is derived from a noun according to the CELEX lexical database.

FamilySize a numeric vector for the number of types in the word’s morphological family.

EtymAge an ordered factor with levels Dutch, DutchGerman, WestGermanic, Germanic and IndoEuropean.

Valency a numeric vector for the verb’s valency, estimated by its number of argument structures.

NVRatio a numeric vector for the log-transformed ratio of the nominal and verbal frequencies of use.

WrittenSpokenRatio a numeric vector for the log-transformed ratio of the frequencies in written and spoken Dutch.

References


Examples
## Not run:
data(etymology)

# ---- EtymAge should be an ordered factor, set contrasts accordingly
etymology$EtymAge = ordered(etymology$EtymAge, levels = c("Dutch", "DutchGerman", "WestGermanic", "Germanic", "IndoEuropean"))
Description

Frequencies of references to previous years in issues of the Frankfurter Allgemeine Zeitung published in 1994.

Usage

data(faz)
Format

A data frame with 800 observations on the following 2 variables.

Year  a numeric vector coding years referenced in articles published in 1994.
Frequency  a numeric vector for the frequencies with which years are referenced.

References


Examples

```r
## Not run:
data(faz)
faz$Distance = 1:nrow(faz)

# ---- visualization
plot(log(faz$Distance), log(faz$Frequency + 1),
    xlab = "log Distance", ylab = "log Frequency")
abline(v = log(49), lty=1, col="red")  # 1945
abline(v = log(54), lty=1, col="red")  # 1940
abline(v = log(76), lty=2, col="blue")  # 1918
abline(v = log(80), lty=2, col="blue")  # 1914

# ---- breakpoint analysis

deviances = rep(0, nrow(faz)-1)
faz$LogFrequency = log(faz$Frequency + 1)
faz$LogDistance = log(faz$Distance)
for (pos in 1 : (nrow(faz)-1)) {  # be patient
  breakpoint = log(pos)
faz$ShiftedLogDistance = faz$LogDistance - breakpoint
faz$PastBreakPoint = as.factor(faz$ShiftedLogDistance > 0)
faz.both = lm(LogFrequency~ShiftedLogDistance:PastBreakPoint, data = faz)
deviances[pos] = deviance(faz.both)
}
breakpoint = log(which(deviances == min(deviances)))

# ---- refit and plot

faz$ShiftedLogDistance = faz$LogDistance - breakpoint
faz$PastBreakPoint = as.factor(faz$ShiftedLogDistance > 0)
faz.both = lm(LogFrequency ~ ShiftedLogDistance:PastBreakPoint, data = faz)

plot(faz$LogDistance, faz$LogFrequency,
     xlab = "log Distance", ylab = "log Frequency", col = "darkgrey")
lines(faz$LogDistance, fitted(faz.both))

## End(Not run)
```
Description

Phonological specifications for onset, nucleus and offset for 1697 Dutch monomorphemic words with a final obstruent. These final obstruents may exhibit a voicing alternation that is traditionally described as syllable-final devoicing: underlying /d/ in /hond/ becomes a /t/ when syllable-final ([hOnt]) and remains a /d/ otherwise ([hOn-den]).

Usage

data(finalDevoicing)

Format

A data frame with 1697 observations on the following 9 variables.

Word a factor with the words as levels.
Onset1Type a factor for the first consonant in the onset, with levels None, Obstruent and Sonorant.
Onset2Type a factor for the second consonant in the onset, with levels None, Obstruent and Sonorant.
VowelType a factor describing the vowel with levels iuy, long and short.
ConsonantType a factor for the first consonant in the offset, with levels None, Obstruent and Sonorant.
Obstruent a factor describing place and manner of articulation of the final obstruent, with levels F (/f,v/), P (/p,b/), S (/s,z/), T (/t,d/) and X (/x,g/).
Nsyll a numeric vector for the number of syllables in the word.
Stress a factor with levels A (antepenult), F (final) and P (penult).
Voice a factor with levels voiced and voiceless.

References


Examples

```R
## Not run:
data(finalDevoicing)
library(rpart)

# ---- CART tree

finalDevoicing.rp = rpart(Voice ~ ., data = finalDevoicing[, -1])
plotcp(finalDevoicing.rp)
```
getKnots.fnc

finalDevoicing.pruned = prune(finalDevoicing, cp = 0.02)
plot(finalDevoicing.pruned, margin = 0.1, compress = TRUE)
text(finalDevoicing.pruned, use.n = TRUE, pretty = 0, cex=0.8)

# ---- logistic regression

library(rms)

finalDevoicing.dd = datadist(finalDevoicing)
options(datadist='finalDevoicing.dd')

finalDevoicing.lrm = lrm(Voice ~ VowelType + ConsonantType + Obstruent + Nsyll + Stress + Onset1Type + Onset2Type, data = finalDevoicing)
anova(finalDevoicing.lrm)

# ---- model simplification

fastbw(finalDevoicing.lrm)

finalDevoicing.lrm = lrm(Voice ~ VowelType + ConsonantType + Obstruent + Nsyll, data = finalDevoicing, x = TRUE, y = TRUE)

plot(Predict(finalDevoicing.lrm))

# ---- model validation

validate(finalDevoicing.lrm, B = 200)

## End(Not run)

getKnots.fnc

Extracts knots from variable name

Description

Extracts knots for predictor specified simply as, e.g., X from column names of model@X or model@frame

Usage

getKnots.fnc(colnms, xlb)

Arguments

colnms columns of model@X

xlb simple predictor name

Details

not intended for independent use
Value

an integer (number of knots)

Note

not intended for independent use

Author(s)

R. H. Baayen

See Also

See Also as plotLMER.fnc

Examples

### Not run: not intended for independent use

```
getMCMCintervals.fnc  calculate HPD prediction intervals
```

Description

calculate HPD 95% prediction intervals

Usage

getMCMCintervals.fnc(fixf, mcmcmatrix, m)

Arguments

- **fixf** vector of fixed effects coefficients (fixef(model.lmer))
- **mcmcmatrix** MCMC matrix obtained with mcmcsamp or pvals.fnc
- **m** model matrix

Details

not intended for independent use

Value

A matrix with columns "lower" and "upper" and rows corresponding to the values of the predictor to be plotted on the X-axis.

Note

not intended for independent use
getPos.fnc

Author(s)
R. H. Baayen

References
languageR

See Also
See Also as plotLMER.fnc

Examples

```#
## Not run: not intended for independent use
```

Description
determines the position (in the X and Y vectors) for the adding of text to an interaction plot

Usage
getPos.fnc(vec, pos)

Arguments
vec vector of Y values
pos can be "beg", "mid", "end"

Details
not intended for independent use

Value
an integer specifying position in vector for X and Y values in plot

Note
not intended for independent use

Author(s)
R. H. Baayen
See Also

See Also as plotLMER.fnc

Examples

## Not run: not intended for independent use

---

getRange.fnc | Extracts range of predicted values from list of data frames

Description

Extracts range of predicted values from list of data frames

Usage

getRange.fnc(lst)

Arguments

- lst: a list with one or more data frames with column names `Y` and optionally `lower` and `upper`.

Details

not intended for independent use.

Value

- value: a two-element vector specifying the range of values in `Y`

Note

not intended for separate use.

Author(s)

R. H. Baayen

See Also

See Also as plotLMER.fnc

Examples

## Not run:

not intended for independent use

## End(Not run)
getRoot.fnc

extract simple name of predictor from expression with poly

Description

extract \( X \) from expressions such as \( \text{poly}(X, 3, \text{raw} = \text{TRUE}) \)

Usage

\[
\text{getRoot.fnc}(\text{xlabel})
\]

Arguments

\[
\text{xlabel} \quad \text{character string for predictor name}
\]

Details

not intended for independent use

Value

a character string (simple name of predictor)

Note

not intended for independent use

Author(s)

R. H. Baayen

See Also

See Also as \( \text{plotLMER.fnc} \)

Examples

\[
## \text{Not run: not intended for independent use}
\]
growth-class

Description

A class for the analysis of word frequency distributions

Objects from the Class

Objects can be created by calls of the form `new("growth", ...`). Growth objects can be plotted, summarized, and printed.

Slots

data: Object of class "list"

Methods

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

Note

to be expanded

Author(s)

R. H. Baayen

References

R. H. Baayen, 2007

See Also

See Also `growth.fnc`.

Examples

`showClass("corres")`
growth.fnc  

Calculate vocabulary growth curve and vocabulary richness measures

Description
This function calculates, for an increasing sequence of text sizes, the observed number of types, hapax legomena, dis legomena, tris legomena, and selected measures of lexical richness.

Usage
```
growth.fnc(text = alice, size = 646, nchunks = 40, chunks = 0)
```

Arguments
- **text**: A vector of strings representing a text.
- **size**: An integer giving the size of a text chunk when the text is to be split into a series of equally-sized text chunks.
- **nchunks**: An integer denoting the number of desired equally-sized text chunks.
- **chunks**: An integer vector denoting the token sizes for which growth measures are required. When chunks is specified, size and nchunks are ignored.

Value
A growth object with methods for plotting, printing. As running this function on large texts may take some time, a period is printed on the output device for each completed chunk to indicate progress. The data frame with the actual measures, which can be extracted with `object.name$data$data`, has the following columns.

- **Chunk**: a numeric vector with chunk numbers.
- **Tokens**: a numeric vector with the number of tokens up to and including the current chunk.
- **Types**: a numeric vector with the number of types up to and including the current chunk.
- **HapaxLegomena**: a numeric vector with the corresponding count of hapax legomena.
- **DisLegomena**: a numeric vector with the corresponding count of dis legomena.
- **TrisLegomena**: a numeric vector with the corresponding count of tris legomena.
- **Yule**: a numeric vector with Yule’s K.
- **Zipf**: a numeric vector with the slope of Zipf’s rank-frequency curve in the double-logarithmic plane.
- **TypeTokenRatio**: a numeric vector with the ratio of types to tokens.
- **Herdan**: a numeric vector with Herdan’s C.
- **Guiraud**: a numeric vector with Guiraud’s R.
- **Sichel**: a numeric vector with Sichel’s S.
- **Lognormal**: a numeric vector with mean log frequency.
Author(s)

R. H. Baayen

References


See Also

See Also `plot.growth`, and the zipfR package.

Examples

```r
# Not run:
data(alice)
alice.growth = growth.fnc(alice)
plot(alice.growth)

# End(Not run)
```

---

**growth2vgc.fnc**  
*Conversion of growth object into a vgc object*

Description

This function converts a growth object (as defined in the languageR package) to a vgc object (as defined in the zipfR package).

Usage

growth2vgc.fnc(growth)

Arguments

growth  
A growth object obtained with growth.fnc().

Value

A vgc object as defined in the zipfR library.

Author(s)

R. H. Baayen
havelaar

References

zipfR Website: <URL: http://purl.org/stefan.evert/zipfR/>

See Also

See also growth.fnc and the zipfR package.

Examples

```r
## Not run: 
library(zipfR)

data(alice)
alice.growth = growth.fnc(text = alice, size = 648, nchunks = 40)
alice.vgc = growth2vgc.fnc(alice.growth)
plot(alice.vgc)

## End(Not run)
```

havelaar  The determiner 'het' in the Dutch novel Max Havelaar

Description

The frequency of the determiner 'het' in the Dutch novel 'Max Havelaar' by Multatuli (Eduard Douwes Dekker), in 99 consecutive text fragments of 1000 tokens each.

Usage

data(havelaar)

Format

A data frame with 99 observations on the following 2 variables.

- **Chunk**  a numeric vector with the indices of the text fragments.
- **Frequency**  a numeric vector with the frequencies of the determiner 'het' in the text fragments.

Source

The text of Max Havelaar was obtained from the Project Gutenberg at at http://www.gutenberg.org/wiki/Main_Page
Examples

```r
## Not run:
data(havelaar)
n = 1000       # token size of text fragments
p = mean(havelaar$Frequency / n)  # relative frequencies

plot(qbinom(ppoints(99), n, p), sort(havelaar$Frequency),
     xlab = paste("quantiles of (", n, ",", round(p, 4), 
                      ")-binomial", sep=""), ylab = "frequencies")

lambda = mean(havelaar$Frequency)
ks.test(havelaar$Frequency, "puno", lambda)
ks.test(jitter(havelaar$Frequency), "puno", lambda)

## End(Not run)
```

### heid

**Lexical decision latencies for words ending in -heid**

Description

A simplified version of the primingHeid dataset.

Usage

```r
data(heid)
```

Format

A data frame with 832 observations on the following 4 variables.

- **subject**: a factor with subjects as levels.
- **word**: a factor with words as levels.
- **rt**: a numeric vector with logarithmically transformed reaction times in visual lexical decision.
- **basefrequency**: a numeric vector with the logarithmically transformed frequency of the base adjective of the word with the suffix -heid.

References

herdan.fnc

Examples

```r
## Not run:
data(heid)
heid = aggregate(heid$RT, list(heid$Word, heid$BaseFrequency), mean)
colnames(heid) = c("Word", "BaseFrequency", "MeanRT")

## End(Not run)
```

---

**Description**

This function calculates Herdan’s constant $C$.

**Usage**

```r
herdan.fnc(text, chunks)
```

**Arguments**

- `text`: A vector of strings representing a text.
- `chunks`: A vector of chunk sizes for which Herdan’s $C$ is required. Duplicate chunk sizes are not allowed, and the number of chunks should be at least 2.

**Value**

A list with components

- `growth`: A data frame with token and type counts for the requested chunk sizes.
- `C`: Herdan’s $C$.

**Author(s)**

R. H. Baayen

**References**


**See Also**

See Also `growth.fnc`.
**Examples**

```r
## Not run:
data(alice)
herdan.fnc(alice, cumsum(rep(floor(length(alice)/40), 40)))

## End(Not run)
```

### imaging

- **fMRI Filtered Signal and Priming Scores for Brain-Damaged Patients**

**Description**

Filtered fMRI signal at the most significant voxel and average priming scores for brain-damaged patients, in a study addressing the extent to which phonological and semantic processes recruit the same brain areas.

**Usage**

```r
data(imaging)
```

**Format**

A data frame with 35 observations on the following 3 variables.

- `condition`: a factor with levels `irregulars` (the morphological condition involving priming using inflected forms of irregular English verbs, e.g., 'began'-'begin') and `semantics` (priming with semantically related words such as 'card' and 'paper').
- `behavioral.score`: a numeric vector for the average priming scores.
- `filtered.signal`: a numeric vector for the intensity of the filtered fMRI signal at the most significant voxel.

**Details**

Location of data points reconstructed from the pixel map of Figure 2b of Tyler et al. 2005.

**Source**


**Examples**

```r
## Not run:
data(imaging)

imaging.lm = lm(FilteredSignal~BehavioralScore*Condition, data=imaging)
summary(imaging.lm)
```
implementInteractions.fnc

**implementInteractions.fnc**

implement interactions in the model matrix

**Description**
given a model matrix with main effects only, add interactions

**Usage**
implementInteractions.fnc(m)

**Arguments**
m a (model) matrix (rows observations, columns predictors)

**Details**
not intended for independent use

**Value**
an updated (model) matrix

**Note**
not intended for independent use
Author(s)
R. H. Baayen

See Also
plotLMER.fnc

Examples
## Not run: not intended for independent use

---

### item.fnc

*Function for by-item regression used by simulateRegression.fnc*

#### Description
This function carries out a by-item regression for the simulated data sets generated in simulate.regression.fnc. It is not designed to be used independently.

#### Usage

```
item.fnc(data)
```

#### Arguments

data

A data frame as produced by make.reg.fnc().

#### Value

A model fitted with lm().

#### Author(s)
R. H. Baayen

#### See Also

See Also simulateRegression.fnc and make.reg.fnc.

#### Examples

## Not run:
```
dat = make.reg.fnc()
dat.lm = item.fnc(dat)
summary(dat.lm)
```

## End(Not run)
By-item anova for simulated data for quasi-F analysis

Description

By-item anova for simulated data set as created within simulateQuasif.fnc. Not intended for independent use. Depends on the packages MASS, coda and lme4.

Usage

items.quasif.fnc(dat)

Arguments

dat

Simulated data set with Subjects, Item, and SOA treatment, as created within simulateQuasif.fnc, or the quasif dataset.

Value

A list with components

p

p-value of F-test for SOA.
data

the input data.
model

the fitted model.

Author(s)

R. H. Baayen

See Also

See also simulateQuasif.fnc.

Examples

## Not run:
data(quasif)
items.quasif.fnc(quasif)

## End(Not run)
lags.fnc

*Calculate vector at specified lag*

**Description**

This function calculates for a given dependent variable the value of that variable at lag timesteps earlier in the time series of an experiment.

**Usage**

```r
lags.fnc(dat, time="Trial", group = "Subject", depvar = "RT", lag=1)
```

**Arguments**

- `dat` A data frame with (minimally) a grouping factor, an time index for successive trails/events, and a behavioral measure
- `group` A grouping factor such as `subject`
- `time` A sequential time index measure such as `trial` number in an experimental list
- `depvar` The dependent variable, usually a chronometric measure such as `RT`
- `lag` The lag for which previous values are to be extracted

**Value**

A vector with the values of the dependent variable at the specified lag. The by-group mean is substituted for the first lag timestep(s), for which there is/are no preceding value(s) for the dependent variable.

**Author(s)**

R. H. Baayen

**See Also**

`acf.fnc`

**Examples**

```r
## Not run:
dfr = data.frame(Subject=c(rep("a", 5), rep("b", 5)),
                 Trial = c(rep(1:5,2)),
                 RT = rnorm(10, 500, 40))
dfr$prevRT = lag.fnc(dfr, time="Trial", group="Subject", depvar="RT")
dfr

## End(Not run)
```
Description

Simulated lexical decision latencies with SOA as treatment, using a Latin Square design with subjects and items, as available in Raaijmakers et al. (1999).

Usage

data(latinsquare)

Format

A data frame with 144 observations on the following 6 variables.

- **Group**: a factor with levels G1, G2 and G3, for groups of subjects.
- **Subject**: a factor with subjects labelled S1, ..., S12.
- **Word**: a factor with words labelled W1, ..., W12.
- **RT**: a numeric vector for reaction times.
- **SOA**: a factor with levels long, medium, and short.
- **List**: a factor with levels L1, L2, and L3 for lists of words.

Source


Examples

```r
## Not run:
data(latinsquare)
library(lme4)
latinsquare.with = simulateLatinsquare.fnc(latinsquare, nruns = 1000, with = TRUE)
latinsquare.without = simulateLatinsquare.fnc(latinsquare, nruns = 1000, with = FALSE)
latinsquare.with$alpha05
latinsquare.without$alpha05

## End(Not run)
```
Lexical decision latencies for 79 English nouns

Description

Lexical decision latencies elicited from 21 subjects for 79 English concrete nouns, with variables linked to subject or word.

Usage

data(lexdec)

Format

A data frame with 1659 observations on the following 28 variables.

- subject  a factor for the subjects.
- rt         a numeric vector for logarithmically transformed reaction times.
- trial      a numeric vector for the rank of the trial in the experimental list.
- sex        a factor with levels F (female) and M (male).
- nativelanguage  a factor with levels English and Other, distinguishing between native and non-native speakers of English.
- correct    a factor with levels correct and incorrect coding whether the word was correctly responded to as a word rather than a nonword.
- prevtype   a factor with levels nonword and word coding whether the item presented at the preceding trial was a word or a nonword.
- prevcorrect a factor with levels correct and incorrect coding whether the preceding item elicited a correct response.
- word       a factor with 79 words as levels.
- frequency  a numeric vector with logarithmically transformed lemma frequencies as available in the CELEX lexical database.
- familysize a numeric vector with the log-transformed count of a word’s morphological family members.
- synsetcount a numeric vector with the log-transformed count of synonym sets in WordNet in which the word is listed.
- length     a numeric vector for the word’s length in letters.
- class       a factor for the semantic category of the word’s referent, with levels animal and plant.
- freqsingular a numeric vector with the frequency in CELEX of the singular form.
- freqplural  a numeric vector with the frequency in CELEX of the plural form.
- deriventropy Shannon’s entropy calculated over the frequency distribution of a word’s family members.
- complex     a factor coding morphological complexity with levels complex and simplex.
rInf1  a numeric vector for the log of the ratio of the singular to the plural frequency.
meanRT  a numeric vector for the by-item mean reaction time averaged over subjects.
SubjFreq a numeric vector for the by-item mean subjective frequency estimate averaged over subjects.
meanSize  a numeric vector for the by-item mean size rating averaged over subjects.
meanWeight a numeric vector for the by-item mean weight rating averaged over subjects.
BNCw  a numeric vector with the logarithmically transformed frequency in the written part of the British National Corpus.
BNCc  a numeric vector with the logarithmically transformed frequency in the context-governed part of the British National Corpus.
BNCd  a numeric vector with the logarithmically transformed frequency in the demographic part of the British National Corpus.
BNCCRatio a numeric vector with the log of the ratio of the (absolute) frequencies in the context-governed and written parts of the British National Corpus, normalized for the differences in corpus size.
BNCdRatio a numeric vector with the log of the ratio of the (absolute) frequencies in the demographic and written parts of the British National Corpus, normalized for the differences in corpus size.

Source

Examples
```r
## Not run:
data(lexdec)
require(lme4)
require(lmerTest)
require(optimx)

lexdec.lmer = lmer(RT ~ 1 + Correct + Trial + PrevType * meanWeight +
Frequency + NativeLanguage * Length + (1|Subject) + (1|Word),
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")),
data = lexdec)
summary(lexdec.lmer)

# random slopes

lexdec.lmerA = lmer(RT ~ 1 + Correct + Trial + PrevType * meanWeight +
Frequency + NativeLanguage * Length + (Trial|Subject) + (1|Word),
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")),
data = lexdec)
anova(lexdec.lmer, lexdec.lmerA)

lexdec.lmerB = lmer(RT ~ 1 + Correct + Trial + PrevType * meanWeight +
Frequency + NativeLanguage * Length + (Trial|Subject) +
(Length|Subject) + (1|Word), data = lexdec,
```
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")))
anova(lexdec.lmerA, lexdec.lmerB)

# model criticism

qqnorm(resid(lexdec.lmerB))

lexdec.lmerC = lmer(RT ~ 1 + Correct + Trial + PrevType * meanWeight +
Frequency + NativeLanguage * Length +
(Trial|Subject) + (Length|Subject) + (1|Word),
data = lexdec[abs(scale(resid(lexdec.lmerB))))<2,],
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")))

qqnorm(resid(lexdec.lmerC))

# p values
summary(lexdec.lmerC)

## End(Not run)

---

**lexicalMeasures**

*Lexical measures for 2233 English monomorphemic words*

---

**Description**

Lexical distributional measures for 2233 English monomorphemic words. This dataset provides a subset of the data available in the dataset `english`.

**Usage**

data(lexicalMeasures)

**Format**

A data frame with 2233 observations on the following 24 variables.

- **Word** a factor with 2284 words.
- **Cels** numeric vector with log-transformed lemma frequency in the CELEX lexical database.
- **Fdif** numeric vector with the logged ratio of written frequency (CELEX) to spoken frequency (British National Corpus).
- **Vf** numeric vector with log morphological family size.
- **Dent** numeric vector with derivational entropy.
- **Ient** numeric vector with inflectional entropy.
- **Nsyl** numeric vector with the log-transformed count of synonym sets in WordNet in which the word is listed.
**lexicalMeasures**

Nsyc numeric vector with the log-transformed count of synonym sets in WordNet in which the word is listed as part of a compound.

Len numeric vector with length of the word in letters.

Ncou numeric vector with orthographic neighborhood density.

Bigr numeric vector with mean log bigram frequency.

InBi numeric vector with log frequency of initial diphone.

spelV numeric vector with type count of orthographic neighbors.

spelN numeric vector with token count of orthographic neighbors.

phonV numeric vector with type count of phonological neighbors.

phonN numeric vector with token count of phonological neighbors.

friendsV numeric vector with type counts of consistent words.

friendsN numeric vector with token counts of consistent words.

ffV numeric vector with type count of forward inconsistent words.

ffN numeric vector with token count of forward inconsistent words.

fbV numeric vector with type count of backward inconsistent words.

fbN numeric vector with token count of backward inconsistent words

ffNonzero a numeric vector with the count of forward inconsistent words with nonzero frequency.

NVRatio a numeric vector with the logarithmically transformed ratio of the noun and verb frequencies.

**References**


**Examples**

```R
## Not run:
data(lexicalMeasures)
data(lexicalMeasuresDist)

library(rms)
library(cluster)
plot(varclus(as.matrix(lexicalMeasures[, -1])))

lexicalMeasures.cor = cor(lexicalMeasures[, -1], method = "spearman")^2
lexicalMeasures.dist = dist(lexicalMeasures.cor)
pltree(diana(lexicalMeasures.dist))

data(lexicalMeasuresClasses)
x = data.frame(measure = rownames(lexicalMeasures.cor),
               cluster = cutree(diana(lexicalMeasures.dist), 5),
               class = lexicalMeasuresClasses$Class)
x = x[order(x$cluster), ]
x

## End(Not run)
```
lexicalMeasuresClasses

Classification of lexical measures

Description
A data frame labelling the lexical measures in the dataset lexicalMeasures as measures of form or meaning.

Usage
data(lexicalMeasuresClasses)

Format
A data frame with 23 observations on the following 3 variables.

Variable  a factor with as levels the measures:
- Bigr  Mean Bigram Frequency.
- CelS  CELEX Frequency.
- Dent  Derivational Entropy.
- fBN   Token Count of Backward Inconsistent Words.
- fBV   Type Count of Backward Inconsistent Words.
- Fdif  Ratio of Frequencies in Written and Spoken English.
- fFN   Token Count of Forward Inconsistent Words.
- fFNonZero Type Count of Forward Inconsistent Words with Nonzero Frequency.
- fFV   Type Count of Forward Inconsistent Words.
- friendsN Token Count of Consistent Words.
- friendsV Type Count of Consistent Words.
- Ient  Inflectional Entropy.
- IBi   Initial Bigram Frequency.
- Len   Length in Letters.
- Ncou  Orthographic Neighborhood Density.
- NsCy  Number of Complex Synsets.
- NsSyS Number of Simplex Synsets.
- NFratio Ratio of Noun and Verb Frequencies.
- phonN Token Count of Phonological Neighbors.
- phonV Type Count of Phonological Neighbors.
- spe1N  Token Count of Orthographic Neighbors.
- spe1V  Type Count of Orthographic Neighbors.
- Vf    Morphological Family Size.

Class  a factor with levels Form and Meaning.

Explanation  a factor with glosses for the variables.
References


Examples

```r
## Not run:
library(cluster)
data(lexicalMeasures)
data(lexicalMeasuresClasses)

lexicalMeasures.cor = cor(lexicalMeasures[,1], method = "spearman")^2
x = data.frame(measure = rownames(lexicalMeasures.cor),
    cluster = cutree(diana(dist(lexicalMeasures.cor)), 5),
    class = lexicalMeasuresClasses$Class)
x = x[order(x$cluster),]
x

## End(Not run)
```

---

**lmerPlotInt.fnc**

*Plot the interaction of two linear numeric predictors in a model fitted with lmer*

**Description**

Visualization of an interaction in a model fitted with lmer of two numeric predictors.

**Usage**

```r
lmerPlotInt.fnc(lmermodel, xname, yname, intxname,
     qntls = seq(0, 1, by = 0.1), view = 30,
     addStdError = FALSE, ndigits = 2, nlev = 30,
     which = "matplot", shadow = 0.5, colour = "lightblue",
     fun = NA, ylabel = NA, ...)
```

**Arguments**

- **lmermodel**: an lmer model object
- **xname**: name (character string) of first numeric predictor
- **yname**: name (character string) of second numeric predictor
- **intxname**: name (character string) of the interaction in the lmer summary
- **qntls**: vector of values to be shown for the second numeric predictor, defaults to deciles
- **view**: specifies the viewing parameter theta for the perspective plot
- **addStdError**: add noise with the standard deviation of the residual error in the lmer model to the plot
make.reg.fnc

Description

This convenience function creates a regression data set with subjects, items, and three numerical predictors, and optionally an effect of learning or fatigue. This function is called by simulateRegression.fnc, and is not intended for independent use.

ndigits number of digits to show for the second numeric predictor
nlev number of levels for the contour plot
which choices are "matplot" (default), "contour", "persp", "image", and "all", in which case a 2 by 2 panel is shown with all four plots
shadow the amount of shade for the perspective plot
colour the color used for the perspective plot, defaults to "lightblue"
fun for matplot displays, a function for transforming the predicted response
ylabel string, to be added to the Y-axis as y label
... other arguments

Value

A plot is shown on the graphics device.

Warning

This function should not be used to plot interactions when one of the predictors also has quadratic or higher terms in the model.

Author(s)

R. H. Baayen

Examples

## Not run:
require(lme4)
require(optimx)
lexdec.lmer = lmer(RT~BNCw+Frequency+(1|Subject)+(1|Word), data=lexdec,
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")))
lmerPlotInt.fnc(lexdec.lmer, "BNCw", "Frequency", "BNCw:Frequency",
which="matplot")

## End(Not run)
Usage

make.reg.fnc(nsubj = 10, nitem = 20, beta = c(400, 2, 6, 4),
learn = FALSE, learnRate = 10, stdevItem = 40, stdevSubj = 80,
stdevError = 50)

Arguments

nsubj Number of subjects (random effect) required.
nitem Number of items (random effect) required.
beta A numeric vector with four beta weights: one for the intercept and one for each
of three predictors.
learn A logical variable, if TRUE, a learning or fatigue effect will be implemented, as
specified by learnRate.
learnRate A number indicating learning (if negative) or fatigue (if positive).
stdevItem A number specifying the standard deviation of the Item random effect.
stdevSubj A number specifying the standard deviation of the Subject random effect.
stdevError A number specifying the standard deviation of the Residual Error.

Value

A data frame with intercept, predictors labelled X, Y and Z, Item, Subject, the simulated random
effects for Item and Subject, the residual errors, and the simulated RTs.

Author(s)

R. H. Baayen

See Also

simulateRegression.fnc

Examples

## Not run:
simdat = make.reg.fnc()
require(lme4)
require(lmerTest)
require(optmx)
simdat.lmer = lmer(RT ~ X + Y + Z + (1|Subject) + (1|Item),
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")),
data = simdat)
summary(simdat.lmer)
simdat = make.reg.fnc(learn = TRUE)
simdat.lmer = lmer(RT ~ X + Y + Z + Trial + (1|Subject) + (1|Item),
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")),
data = simdat)
summary(simdat.lmer)
makeDefaultMatrix.fnc  
Create model matrix with main effects only

Description

Creates a model matrix with main effects only

Usage

makeDefaultMatrix.fnc(model, n = 100, conditioningPred = ",
                       conditioningValue = NULL, control = NA)

Arguments

  model  A model fit with lmer
  n      integer specifying number of points to be plotted on X-axis
  conditioningPred
          name of predictor entering into interaction
  conditioningValue
          vector of values (numeric or factor level names) to be shown for interaction
  control
          a two-element list (predictor, value) specifying an additional predictor to be
          fixed to the given value in a partial effect plot. May be useful for hand-made
          plots for three-way interactions.

Details

  not intended for independent use

Value

  a (model) matrix

Note

  not intended for independent use

Author(s)

  R. H. Baayen

See Also

  See Also as plotLMER.fnc
Examples

## Not run: not intended for independent use

---

**Description**

creates a data set with $Y \sim 30 + \cos(X)$ for 10 subjects, to compare restricted cubic spline in `lmer` with the spline of `ols`.

**Usage**

```r
makeSplineData.fnc(intr=0)
```

**Arguments**

- `intr`: integer denoting type of data set: with 0 a data set with simple spline is made, with 1 a data set with a parallel interaction, and with 2 a data set with a crossed interaction.

**Details**

Requires `rms` package to be attached.

**Value**

A data frame with as values:

- `y`: $Y = 30 + \cos(X)$
- `X`: ranges from 2 to 8.28
- `Subject`: random-effects factor with 10 levels
- `Ranef`: subjects-specific changes to intercept
- `Error`: by-observation noise
- `Y`: the dependent variable, `y+Ranef+Error`

**Note**

intended for illustration only

**Author(s)**

R. H. Baayen

**See Also**

See Also as `plotLMER.fnc`
Examples

```r
# Not run:
require("rms")
require("optimx")
require("lmerTest")
dfr = makeSplineData.fnc()
table(dfr$Subject)
xlowess.fnc(Y ~ X | Subject, data = dfr)

dfr.lmer = lmer(Y ~ rcs(X, 5) + (1|Subject), data = dfr,
control=lmerControl(optimizer="optimx", optCtrl=list(method="nlminb")))
dfr$fittedLMER = as.vector(dfr.lmer@X %*% fixef(dfr.lmer))

dfr.dd = datadist(dfr)
options(datadist='dfr.dd')
dfr.ols = ols(Y-Subject+rcs(X), data=dfr, x=T, y=T)
dfr$fittedOLS = fitted(dfr.ols)

# we plot the lmer() fit in blue, the ols() fit in red (both adjusted for
# subject S1), and plot the underlying model in green
plot(dfr[dfr$Subject=="S1",]$X, dfr[dfr$Subject=="S1",]$fittedLMER +
ranef(dfr.lmer)[[1]]["S1",], type="l", col="blue",
ylim = range(dfr$y + ranef(dfr.lmer)[[1]]["S1",],
dfr[dfr$Subject == "S1",]$fittedLMER,
dfr[dfr$Subject == "S1",]$fittedOLS), xlab="X", ylab="Y")
lines(dfr[dfr$Subject=="S1",]$X, dfr[dfr$Subject=="S1",]$fittedOLS, col="red")
lines(dfr[dfr$Subject=="S1",]$X, dfr[dfr$Subject=="S1",]$y+ranef(dfr.lmer)[[1]]["S1",],
col="green")
legend(2,29,c("30*cos(x)", "lmer (S1)", "ols (S1)"), lty=rep(1,3),
col=c("green", "blue", "red"))

## End(Not run)
```

moby

**Moby Dick**

Description

The text of H. Melville’s ‘Moby Dick’, with punctuation marks removed.

Usage

`data(alice)`

Format

A character vector with 215994 words.
Source

The project Gutenberg at http://www.gutenberg.org/wiki/Main_Page

Examples

```r
## Not run:
data(moby)
moby[1:2]

## End(Not run)
```

---

### Description

This function produces a scatterplot for a bivariate standard normal distribution with least squares regression line.

### Usage

```r
mvrnormplot.fnc(r, n, limits)
```

### Arguments

- `r`: The correlation, defaults to 0.9.
- `n`: Number of simulated data points, defaults to 100.
- `limits`: Optional range for the axes

### Value

A scatterplot with ordinary least squares regression line is shown on the graphics device, with sample estimate of \( r \) added at the top of the plot.

### Author(s)

R. H. Baayen

### See Also

- `mvrnorm` (MASS package)

### Examples

```r
## Not run:
mvrnormplot.fnc(r=0.9, n=100)

## End(Not run)
```
nessdemog

Description

Frequency (m) and frequency of frequency (Vm) for string types with the suffix -ness in the context-governed subcorpus of the British National Corpus sampling spoken British English.

Usage

data(nessdemog)

Format

A data frame with 37 observations on the following 2 variables.

m  a numeric vector with word frequencies.
Vm  a numeric vector with the frequencies of word frequencies.

Source

The British National Corpus, see http://www.natcorp.ox.ac.uk/

Examples

```r
## Not run:
data(nessdemog)
library(zipfR)
esscg.spc = spc(m=nesscg$m, Vm = nesscg$Vm)
plot(nesscg.spc)

## End(Not run)
```
Format

A data frame with 15 observations on the following 2 variables.

\( m \) a numeric vector with word frequencies.

\( Vm \) a numeric vector with the frequencies of word frequencies.

Source

The British National Corpus, see http://www.natcorp.ox.ac.uk/

Examples

data(NESSw)
library(zipfR)
NESSw.spc = spc(m=NESSw$m, Vm = NESSw$Vm)
plot(NESSw.spc)


| nessw | Frequency spectrum for -ness in the written BNC |

Description

Frequency (m) and frequency of frequency (Vm) for string types with the suffix -ness in the sub-corpus of the British National Corpus sampling written British English.

Usage

data(NESSw)

Format

A data frame with 189 observations on the following 2 variables.

\( m \) a numeric vector with word frequencies.

\( Vm \) a numeric vector with the frequencies of word frequencies.

Source

The British National Corpus, see http://www.natcorp.ox.ac.uk/

Examples

```r
## Not run:
data(NESSw)
library(zipfR)
NESSw.spc = spc(m=NESSw$m, Vm = NESSw$Vm)
plot(NESSw.spc)
## End(Not run)
```
**oldFrench**  
*Frequencies of tag trigrams in Old French texts*

**Description**
Frequencies of 35 morphosyntactic tag trigrams in 343 Old French texts.

**Usage**
data(oldFrench)

**Format**
A data frame with the frequencies of 35 tag trigrams (columns) for 343 Old French texts (rows) in the Nouveau Corpus d’Amsterdam. See oldFrenchMeta for details on the texts (and manuscript versions).

**Source**
Data from Nouveau Corpus d’Amsterdam, [http://www.uni-stuttgart.de/lingrom/stein/corpus/](http://www.uni-stuttgart.de/lingrom/stein/corpus/).

**References**

**Examples**
data(oldFrench)  
data(oldFrenchMeta)
oldFrench.ca = corres.fnc(oldFrench)
plot(oldFrench.ca, rlabels = oldFrenchMeta$Genre,  
rcol = as.numeric(oldFrenchMeta$Genre), rcex = 0.5,  
extreme = 0.1, ccol = “blue”)

**oldFrenchMeta**  
*Meta data for the oldFrench data*

**Description**
Meta data for the oldFrench data, a matrix of frequencies for texts (rows) by tag trigrams (columns). The meta data provide information on the texts, manuscript variants, their authors, their region and approximate date of origin, their general topic, and their genre.
Usage

data(oldFrenchMeta)

Format

A data frame with 342 observations on the following 7 variables.

Textlabels a factor with texts coded as follows:

Abe  J. de Meun, Traduction de la premiere epitre de P. Abelard, 1–821
Hyl1  Anon, La vie de saint Hylaire
Art  J. de Meun, L’art de chevalerie
Bar  Anon, L’histoire de Barlaam et Josaphat
Cathy Anon, La vie de sainte Catherine d’Alexandrie
Hyl2  Anon, La vie de saint Hylaire
Ch1  Chretien de Troyes, Le Chevalier au lion
Ch2  Chretien de Troyes, Le chevalier au lion
Clari Robert de Clari, La conquete de Constantinople
Marie Rutebeuf, Sainte Marie l’Egyptienne
Fab4c Anon, Fabliau nr 4 ms C
Fab4e Anon, Fabliau nr 4 ms E
Fab4f Anon, Fabliau 4f
Faba Anon, Fabliaux nrs 1,2,4,23 et 29 du ms A
Fabb Anon, Fabliaux nrs 2 et 4 du ms B
Fabd COD Anon, Fabliaux nrs 2 et 4 du ms D
Hyl3  Anon, La vie de saint Hylaire
Jacobi Pierre de Beauvais, The Liber Sancti Jacobi
Louis J. de Joinville, La vie de saint Louis
Cathy1 Anon, La passion saynte Katherine
Lancelot Anon, Lancelot du Lac, p. 1.1–20.13
Merlin1 Merlin, Robert de Boron
Marga Anon, La vie de Sainte Marguerite de Wace
Martin Anon, Leben und Wunderthaten des heiligen Martin
Merlin2 Anon, Merlin, p.1–29 (ms. Huth)
RoseA J. de Meun, Le Roman de la Rose
Arthur Anon, La mort le roi Artu, par.1–35
NimAf Anon, charroi de Nimes, ms. A, fragment
NimB1 Anon, Le charroi de Nimes, ms B1
NimB2 Anon, Le charroi de Nimes, ms B2
Nouvel Jacquemart Gielee, Renart le Nouvel
Jehan Anon, La vie de saint Jehan Bouche d’Or
PerØ Chretien de Troyes, Perceval
perL Chretien de Troyes, Perceval
PerQ Chretien de Troyes, Perceval
PerS  Chretien de Troyes, Perceval
PerU  Chretien de Troyes, Perceval
Queste  Anon, La queste del saint Graal, p.1.1–41.17
Rob  Anon, Robert le Diable, v.1–808
RomB  Anon, Le Roman de Renart, br.VI, ms B
RomD  Anon, Roman de Renard, br.VI, ms D
RomL  Anon, Roman de Renard, br. VI, ms L
Rom0  Anon, Le Roman de Renart, br. VI, ms 0
RoseB  Guillaume de Lorris, Le roman de la rose
Sapi  Anon, Sermo de sapientia, dans: Li dialogue Gregoire lo pape
Troi  Anon, Le roman de Troie en prose, par.1–19
Conqueste  Josfroi de Vileharduyn, La conqueste de Costentinoble
YvA  Chretien de Troyes, Le chevalier au lion, v.1–1000
YvP  Chretien de Troyes, Le chevalier au lion, v.1–1000
YvS  Chretien de Troyes, Le chevalier au lion, v.1–1000
YvV  Chretien de Troyes, Le chevalier au lion, v 1–1000

Codes  a factor with manuscript variants, indicated by extensions to the text codes.
Author  a factor with levels Anon, ChretienDeTroyes, GuillaumeDeLorris, Joinville, Meun,
        NouvelRenart, PierreDeBeauvais, RobertDeBoron, RobertDeClari, RobertLeDiable, Rutebeuf,
        and Villeharduyn.
Topic  a factor with levels Knight, Other, and Saint.
Genre  a factor with levels poetry and prose.
Region  a factor with levels R1 (Picardie), R2 (Champenois), and R3 (Nievre-Allier).
Year  a numeric vector indicating approximate year of origin.

Source
Data from Nouveau Corpus d’Amsterdam, http://www.uni-stuttgart.de/lingrom/stein/corpus/.

References
heiligen in ruimte en tijd To appear in Onze Taal.

Examples
## Not run:
data(oldFrench)
data(oldFrenchMeta)
oldFrench.ca = corres.fnc(oldFrench)
plot(oldFrench.ca, rlabels = oldFrenchMeta$Genre,
     rcol = as.numeric(oldFrenchMeta$Genre), rcex = 0.5,
     extreme = 0.1, ccol = "blue")
## End(Not run)
ozone

Description

Usage
data(alice)

Format
A character vector with 39513 words.

Source
The project Gutenberg at http://www.gutenberg.org/wiki/Main_Page

Examples
```r
## Not run:
data(oz)
oz[1:5]

## End(Not run)
```

--

pairscor.fnc

Description
A matrix of scatterplots is produced with Pearson and Spearman correlations in the lower triangle. By default, smoothers are added to panels in the upper triangle, and histograms are added to the panels on the diagonal.

Usage
```r
pairscor.fnc(data, hist = TRUE, smooth = TRUE,
              cex.points = 1, col.points = "darkgrey")
```
parsePredName.fnc

Arguments

data a data frame or matrix with numeric vectors.
hist a logical indicating whether panels on the diagonal should contain a histogram.
smooth a logical indicating whether panels in the upper triangle should have a smoother added.
cex.points a number indicating the size of the points in the panels in the upper triangle, available only when smoothers are added.
col.points a number or string indicating the color of the points in the panels in the upper triangle, available only when smoothers are added.

Author(s)

R. Harald Baayen

See Also

See Also pairs and panel.smooth.

Examples

```r
## Not run:
data(lexicalMeasures)
pairs.cor.fnc(lexicalMeasures[,c("CelS", "Vf", "Ient", "NsyS", "Ncou")])

## End(Not run)
```

parsePredName.fnc parse character string specifying restricted cubic spline

Description

parse character string specifying restricted cubic spline into simple predictor name and number of knots

Usage

parsePredName.fnc(name)

Arguments

name character string for predictor, e.g. rcs(X, 3)

Details

not intended for independent use
Value

- a list with components
  - `basename`: character string denoting simple predictor name \( (X) \)
  - `knots`: integer specifying number of knots

Note

- not intended for independent use

Author(s)

- R. H. Baayen

See Also

- See Also as `plotLMER.fnc`

Examples

```r
## Not run: not intended for independent use
```

---

**Description**

The development of periphrastic *do* in English: Ellegard’s counts for the use of *do* across four sentence types in 11 consecutive time periods between 1390 and 1710.

**Usage**

```r
data(periphrasticDo)
```

**Format**

A data frame with 44 observations on the following 5 variables.

- `begin`: a numeric vector with beginnings of the time periods used by Ellegard.
- `end`: a numeric vector with ends of these time periods.
- `type`: a factor for sentence type, with levels `affdecl` (affirmative declarative), `affquest` (affirmative question), `negdecl` (negative declarative) and `negquest` (negative question).
- `do`: a numeric vector with the count of sentences with *do*.
- `other`: a numeric vector with the count of sentences without *do*.  

periphrasticDo The development of periphrastic do in English
Source


References


Examples

```r
## Not run:
data(periphrasticDo)

# add midpoints of time periods

periphrasticDo$year = periphrasticDo$begin +
  (periphrasticDo$end-periphrasticDo$begin)/2

# and ad an indicator variable distinguishing the first three time periods
# from the others

periphrasticDo$Indicator = rep(c(rep(0, 3), rep(1, 8)), 4)

# fit a logistic regression model

periphrasticDo(glm = glm(cbind(do, other) ~
  (year + I(year^2) + I(year^3)) * type + Indicator * type +
  Indicator * year, data = periphrasticDo, family = "binomial")

anova(periphrasticDo$glm, test = "F")

# visualization of data and model predictions

periphrasticDo$predict = predict(periphrasticDo$glm, type = "response")
par(mfrow=c(2, 2))
for (i in 1:nlevels(periphrasticDo$type)) {
  subset = periphrasticDo[periphrasticDo$type ==
    levels(periphrasticDo$type)[i], ]
  plot(subset$year,
    subset$do/(subset$do + subset$other),
    type = "p", ylab = "proportion", xlab = "year",
    ylim = c(0, 1), xlim = c(1400, 1700))
  mtext(levels(periphrasticDo$type)[i], line = 2)
  lines(subset$year, subset$predict, lty = 1)
}
par(mfrow=c(1, 1))

## End(Not run)
```
Phylogenetic relations between Papuan and Oceanic languages: 127 grammatical traits (absent/present) for 31 languages.

data(phylogeny)

A data frame with 31 observations on the following 127 variables.


Family a factor with levels Oceanic and Papuan.

Fric a numeric vector, 1: presence, 0: absence

PrenasalizedStops a numeric vector, 1: presence, 0: absence

PhonDistBetweenLandR a numeric vector, 1: presence, 0: absence

PhonVelarFricOrGlide a numeric vector, 1: presence, 0: absence

PhonVoicingContrAmongStops a numeric vector, 1: presence, 0: absence

PhonConsLength a numeric vector, 1: presence, 0: absence

PhonVowelLength a numeric vector, 1: presence, 0: absence

ContrPhonTypesForVowels a numeric vector, 1: presence, 0: absence

PhonStress a numeric vector, 1: presence, 0: absence

WordFinalConss a numeric vector, 1: presence, 0: absence

ConsClusters a numeric vector, 1: presence, 0: absence

DefOrSpecArt a numeric vector, 1: presence, 0: absence

IndefOrNonSpecArt a numeric vector, 1: presence, 0: absence

ArticleNounOrder a numeric vector, 1: presence, 0: absence

NounInitNps a numeric vector, 1: presence, 0: absence

InclExclDist a numeric vector, 1: presence, 0: absence

PronNum a numeric vector, 1: presence, 0: absence

PronRelationship a numeric vector, 1: presence, 0: absence

PronConflation a numeric vector, 1: presence, 0: absence

MoreThan2DegreesDistDem a numeric vector, 1: presence, 0: absence
NonSpkrAnchoredDem  a numeric vector, 1: presence, 0: absence
VerticalityDem  a numeric vector, 1: presence, 0: absence
ClassifiedDem  a numeric vector, 1: presence, 0: absence
NumDeterminedDecl  a numeric vector, 1: presence, 0: absence
GenderDeterminedDecl  a numeric vector, 1: presence, 0: absence
SuppletiveNouns  a numeric vector, 1: presence, 0: absence
SingMarkedNoun  a numeric vector, 1: presence, 0: absence
DualMarkedNoun  a numeric vector, 1: presence, 0: absence
PlMarkedNoun  a numeric vector, 1: presence, 0: absence
OtherNumMarkedNoun  a numeric vector, 1: presence, 0: absence
LimitedDistNumMarking  a numeric vector, 1: presence, 0: absence
NounClassesGenders  a numeric vector, 1: presence, 0: absence
ConcordBeyondNP  a numeric vector, 1: presence, 0: absence
NumeralClassifiers  a numeric vector, 1: presence, 0: absence
PossClassifiers  a numeric vector, 1: presence, 0: absence
PossClasses  a numeric vector, 1: presence, 0: absence
Inalienability  a numeric vector, 1: presence, 0: absence
MultiplePossConstr  a numeric vector, 1: presence, 0: absence
PrefixMarkedPoss  a numeric vector, 1: presence, 0: absence
SuffixMarkedPoss  a numeric vector, 1: presence, 0: absence
MarkedPossr  a numeric vector, 1: presence, 0: absence
MarkedPossessee  a numeric vector, 1: presence, 0: absence
PossPossdOrder  a numeric vector, 1: presence, 0: absence
DecimalNumerals  a numeric vector, 1: presence, 0: absence
QuinaryNumerals  a numeric vector, 1: presence, 0: absence
CollectiveNouns  a numeric vector, 1: presence, 0: absence
AdjVerbLexOverlap  a numeric vector, 1: presence, 0: absence
AdjAttributionPred  a numeric vector, 1: presence, 0: absence
CoreCaseMarking  a numeric vector, 1: presence, 0: absence
ObliqueCaseMarking  a numeric vector, 1: presence, 0: absence
Prepositions  a numeric vector, 1: presence, 0: absence
Postpositions  a numeric vector, 1: presence, 0: absence
TamPerson  a numeric vector, 1: presence, 0: absence
VerbPrefixesProclitics  a numeric vector, 1: presence, 0: absence
VerbSuffixesEnclitics  a numeric vector, 1: presence, 0: absence
PunctualContinuous  a numeric vector, 1: presence, 0: absence
RealisIrrealis  a numeric vector, 1: presence, 0: absence
phylogeny

SSuffix a numeric vector, 1: presence, 0: absence
SPrefix a numeric vector, 1: presence, 0: absence
ASuffix a numeric vector, 1: presence, 0: absence
APrefix a numeric vector, 1: presence, 0: absence
OSuffix a numeric vector, 1: presence, 0: absence
OPrefix a numeric vector, 1: presence, 0: absence
VerbVarTam a numeric vector, 1: presence, 0: absence
VerbVarVClass a numeric vector, 1: presence, 0: absence
VerbVarClauseType a numeric vector, 1: presence, 0: absence
VerbVarPerson a numeric vector, 1: presence, 0: absence
NumStemAlt a numeric vector, 1: presence, 0: absence
PersonStemAlt a numeric vector, 1: presence, 0: absence
SepVerbNumPerson a numeric vector, 1: presence, 0: absence
Portmanteau3Plus a numeric vector, 1: presence, 0: absence
DistributedCategory a numeric vector, 1: presence, 0: absence
NonCore a numeric vector, 1: presence, 0: absence
RecipientObj a numeric vector, 1: presence, 0: absence
X3PlacePreds a numeric vector, 1: presence, 0: absence
VerbNeg a numeric vector, 1: presence, 0: absence
VerbDirection a numeric vector, 1: presence, 0: absence
VerbSuppletion a numeric vector, 1: presence, 0: absence
ConjugationClasses a numeric vector, 1: presence, 0: absence
TransIntransAlt a numeric vector, 1: presence, 0: absence
TransitivizingMorph a numeric vector, 1: presence, 0: absence
IntranstivizingMorph a numeric vector, 1: presence, 0: absence
ReflexiveMorph a numeric vector, 1: presence, 0: absence
ReciprocalMorph a numeric vector, 1: presence, 0: absence
VerbClassifiers a numeric vector, 1: presence, 0: absence
Copula a numeric vector, 1: presence, 0: absence
NonVbPreds a numeric vector, 1: presence, 0: absence
SerialVerbConstr a numeric vector, 1: presence, 0: absence
Auxiliaries a numeric vector, 1: presence, 0: absence
VerbCompounds a numeric vector, 1: presence, 0: absence
VerbAdjunctConstr a numeric vector, 1: presence, 0: absence
VbIncorporation a numeric vector, 1: presence, 0: absence
ExistentialVerb a numeric vector, 1: presence, 0: absence
IrregularGive a numeric vector, 1: presence, 0: absence
ClosedClassOfVb a numeric vector, 1: presence, 0: absence
SvIntransClauses a numeric vector, 1: presence, 0: absence
VsIntransClauses a numeric vector, 1: presence, 0: absence
VInitTransClauses a numeric vector, 1: presence, 0: absence
VMedialTransClauses a numeric vector, 1: presence, 0: absence
VFinalTransClauses a numeric vector, 1: presence, 0: absence
FixedConstituentOrder a numeric vector, 1: presence, 0: absence
ClauseFinalNeg a numeric vector, 1: presence, 0: absence
ClauseInitNeg a numeric vector, 1: presence, 0: absence
ImpVsDeclNeg a numeric vector, 1: presence, 0: absence
VbAndNonVbPredIdentity a numeric vector, 1: presence, 0: absence
SOMorphInBasicConstr a numeric vector, 1: presence, 0: absence
SAMorphInBasicConstr a numeric vector, 1: presence, 0: absence
SOMorphInComplexConstr a numeric vector, 1: presence, 0: absence
SAMorphInComplexConstr a numeric vector, 1: presence, 0: absence
SynConflationOfSO a numeric vector, 1: presence, 0: absence
ControlledUncontrolled a numeric vector, 1: presence, 0: absence
ClauseChaining a numeric vector, 1: presence, 0: absence
SimultaneousSequential a numeric vector, 1: presence, 0: absence
SayInDesidConstr a numeric vector, 1: presence, 0: absence
RelativeClauses a numeric vector, 1: presence, 0: absence
PurpSubClauses a numeric vector, 1: presence, 0: absence
TemporalSubClauses a numeric vector, 1: presence, 0: absence
ComplementClauses a numeric vector, 1: presence, 0: absence
CausBySerialVerbConstr a numeric vector, 1: presence, 0: absence
CausByBoundAffClit a numeric vector, 1: presence, 0: absence
CausByConstrInvolvingSay a numeric vector, 1: presence, 0: absence
MorphTopicOrFocus a numeric vector, 1: presence, 0: absence
TailHeadLinkage a numeric vector, 1: presence, 0: absence
VerbRedup a numeric vector, 1: presence, 0: absence
NounRedup a numeric vector, 1: presence, 0: absence

Source

plot.corres

Examples

```r
# Not run:
data(phylogeny)
library(ape)

# joint analysis of both language families
phylogeny.dist = dist(phylogeny[,3:ncol(phylogeny)], method = "binary")
phylogeny.dist.tr = nj(phylogeny.dist)
families =
  as.character(phylogeny$Family[as.numeric(phylogeny.dist.tr$tip.label)])
languages =
  as.character(phylogeny$Language[as.numeric(phylogeny.dist.tr$tip.label)])
plot(phylogeny.dist.tr, type="u", font = as.numeric(as.factor(families)))

# analysis of subset of Papuan languages
papuan = phylogeny[phylogeny$Family == "Papuan",]
papuan$Language = as.factor(as.character(papuan$Language))
papuan.meta = papuan[,1:2]
papuan.mat = papuan[,3:ncol(papuan)]
papuan.meta$Geography = c("Bougainville", "Bismarck Archipelago", "Bougainville",
  "Bismarck Archipelago", "Bismarck Archipelago", "Central Solomons",
  "Bougainville", "Louisiade Archipelago", "Bougainville",
  "Bismarck Archipelago", "Bismarck Archipelago",
  "Bismarck Archipelago", "Central Solomons", "Central Solomons",
  "Central Solomons")
papuan.dist = dist(papuan.mat, method = "binary")
papuan.dist.tr = nj(papuan.dist)
fonts =
  as.character(papuan.meta$Geography[as.numeric(  
    papuan.dist.tr$tip.label)])
papuan.dist.tr$tip.label =
  as.character(papuan.meta$Language[as.numeric(  
    papuan.dist.tr$tip.label)])
plot(papuan.dist.tr, type = "u", font = as.numeric(as.factor(fonts)))

## End(Not run)
```

plot.corres  

Plot method for correspondence objects

Description

This function defines a plot method for correspondence objects.
Usage

```r
## S3 method for class 'corres'
plot(x, main = "", addcol = TRUE, extreme = 0, rcex = 1, rcol = 1,
     rlabels = "", stretch = 1.4, ccex = 1, ccol = 2, clabels = "", ...) 
```

Arguments

- `x`: A correspondence object as produced by `corres`.
- `main`: A string used for the main title of the plot.
- `addcol`: A logical, if true, columns are added to the plot.
- `extreme`: If nonzero, defines quantiles that define the extremes such that only data points exceeding these extremes are plotted.
- `rcex`: sets cex graphical parameter for rows.
- `rcol`: sets color for rows.
- `rlabels`: vector of row labels.
- `stretch`: a number defining the degree to which the columns (or rows) should be stretched out for visual presentation.
- `ccex`: sets cex graphical parameter for columns.
- `ccol`: sets color for columns.
- `clabels`: vector of column labels.
- `...`: other parameters to be passed through to plotting functions.

Value

A plot on the graphics device.

Author(s)

R. H. Baayen

See Also

See Also `corres.fnc`, `link{corsup.fnc}`.

Examples

```r
## Not run:
data(oldFrench)
data(oldFrenchMeta)

oldFrench.ca = corres.fnc(oldFrench)

plot(oldFrench.ca)

plot(oldFrench.ca, rlabels = oldFrenchMeta$Genre,
     rcol = as.numeric(oldFrenchMeta$Genre), rcex = 0.5,
     extreme = 0.1, ccol = "blue")
```
Description

This function defines the plot method for growth objects.

Usage

```r
## S3 method for class 'growth'
plot(x, w = "all", ...)
```

Arguments

- `x`: A growth object.
- `w`: A character string denoting the name of a specific variable to be plotted.
- `...`: other parameters to be passed through to plotting functions.

Value

A plot shown on the graphics device.

Author(s)

R. H. Baayen

See Also

See Also `growth.fnc`.

Examples

```r
## Not run:
data(alice)
alice.growth = growth.fnc(alice)
plot(alice.growth)
plot(alice.growth, w = "Yule")
## End(Not run)
```
plotAll.fnc

create plot or plots for list with data frames for plot or subplots

Description

given a list with one or more data frames with values for a plot (or subplot), create the actual plots

Usage

plotAll.fnc(reslist, sameYrange = TRUE, ylabel, xlabel = NA, intrName = NA,
pos = "end", ylimit = NA, addlines = FALSE, cexsize = 0.6, conditioningVals = NA,
conditioningColors = 1, conditioningLines = 1, lineColor = 1, addToExistingPlot = FALSE, ...)

Arguments

reslist list with as components either a data frame or a list with data frames, the data frames specify X and Y coordinates and HPD intervals
sameYrange logical, if TRUE, the ylim for each panel will be chosen to accomodate the range of values across all panels in the plot
ylabel label to be used for the vertical axis
xlabel label to be used for the horizontal axis; this option is active only when a single predictor is plotted
intrName label for the interaction predictor, if present
pos location of legend values for interaction
ylimit if specified, overrides sameYrange for ylim
addlines if TRUE, adds line between levels of same factor(s)
cexsize character expansion size for information in the plot for interactions, default is 0.6
conditioningVals vector of names of the levels of the conditioning factor in the interaction (the factor with different lines in the plot)
conditioningColors vector of names of the colors to be used for the levels of the conditioning factor in the interaction (the factor with different lines in the plot)
conditioningLines vector of names of the line types to be used for the levels of the conditioning factor in the interaction (the factor with different lines in the plot), by default solid lines
lineColor name of color to be used for the lines in the plot
addToExistingPlot if TRUE, the current plot is added to an already existing plot
...

further graphical parameters to be passed down, none are currently implemented
plotLMER.fnc

Details

Note that `reslist` may contain as elements lists of data frames, these then specify the separate points or lines to be plotted for a given interaction.

Value

A plot is produced on the graphics device.

Note

not intended for independent use

Author(s)

R. H. Baayen

See Also

See Also as `plotLMER.fnc`

Examples

## Not run: not intended for independent use

---

plotLMER.fnc  

Description

Plot partial effects of a (generalized) linear mixed-effects model fit with `lmer`. For gaussian models, 95% highest posterior density credible intervals can be added.

Usage

```r
plotLMER.fnc(model, xlabel = NA, xlabs = NA, ylabel = NA, ylimit = NA, ilabel = NA, fun = NA, pred = NA, control = NA, ranefs = NA, n = 100, intr = NA, lockylim = TRUE, addlines = FALSE, withList = FALSE, cexsize = 0.5, linecolor = 1, addToExistingPlot = FALSE, verbose = TRUE, ...)
```

Arguments

- `model`: a LMM or GLMM model object of class `lmerMod`
- `xlabel`: label for X-axis (if other than the variable name in the original model formula)
- `xlabs`: character vector with labels for X-axes in multipanel plot (if other than the variable names in the original model formula); if used, `xlabel` should not be specified

```r
```
ylabel: label for Y-axis (if other than the variable name of the dependent variable in the original model formula)

ylimit: range for vertical axis; if not specified, this range will be chosen such that all data points across all subplots, including HPD intervals, will be accommodated

ilabel: label for the interaction shown in the lower right-hand margin of the plot, overriding the original variable name in the model formula

fun: a function to be applied for transforming the dependent variable, if NA, no transformation is applied; for models with family = "binomial", fun is set to plogis by default; this can be disabled by setting fun=function(x) return(x).

pred: character string with name of predictor; if specified, a single plot will produced for the partial effect of this specific predictor

control: a two-element list list(predictor, val) specifying a predictor the value of which has to be set to val in the partial effect plot(s); the predictor name should be exactly as specified in names(model@fixef). It is up to the user to make sure that name and value make sense, the code here hands full 'control' to the user.

ranefs: a four-element list Group, Level, Predictor, specifying a random-effect Group (e.g. Subject), a level (e.g., S10) and a value (e.g., LogFrequency) for which partial effects have to be calibrated.

n: integer denoting number of points for the plot, chosen at equally spaced intervals across the empirical range of the predictor variable

intr: a list specifying an interaction to be graphed; obligatory arguments are (1) the name of the interaction variable, followed by (2) a vector of values for that variable, followed by (3) the position for interaction labels ("beg", "mid", or "end", or 'NA' if no labels are desired), optionally followed by (4) a list with as first element a vector of colors and as second element a vector of line types. The number of elements in both vectors should match the number of values specified under (2) for the interaction predictor.

lockylim: logical specifying whether all subplots should have the same range of values for the vertical axis; if TRUE, this range will be chosen to accomodate all fitted values including HPD intervals for all predictors across all plots

addlines: if TRUE, adds line(s) between levels of same factor(s)

withList: logical, if TRUE, a list will be output with all data frames for the subplots

cexsize: character expansion size (cex) for additional information in the plot for interactions

linecolor: color of lines in the plot, by default set to 1 (black)

addToExistingPlot: default FALSE, if set to TRUE, plot will be added to previous plot, but only if pred is specified

verbose: if TRUE (default), effect sizes and default transformations are reported

...: further graphical parameters to be passed down; warning: col, pch, lty and cex will often generate an error as they are internally already fully specified for specialized subplots
Details

When no predictor is specified, a series of plots is produced for the partial effects of each predictor. The graphs are shown for the reference level for factors and are adjusted for the median value for the other numerical predictors in the model. Interactions are not shown. The user should set up the appropriate number of subplots on the graphics device before running plotLMER.fnc().

Instead of showing all predictors jointly, plotLMER.fnc() can also be used to plot the partial effect of a specific predictor. When a specific predictor is specified (with `pred = ...`), a single plot is produced for that predictor. In this case, the `intr` argument can be used to specify a single second predictor that enters into an interaction with the selected main predictor.

Polynomials have to be fitted with `poly(..., degree, raw=TRUE)` and restricted cubic splines with `rcs()` from the `rms` package.

Value

A plot is produced on the graphical device.

Note

This code needs much more work, including (i) extension to `poly` with `raw=FALSE`, and (ii) general clean-up of the code.

Author(s)

R. H. Baayen

References

The ‘danish’ dataset in the example section is contributed by Laura Winther-Balling, see Winther-Balling, L. and Baayen, R. H., Morphological effects in auditory word recognition: Evidence from Danish, Language and Cognitive Processes, in press.

See Also

See also other utilities in languageR for facilitating work with `lmer`

Examples

```r
## Not run:

# we will stay as close to the older optimizer of lme4 as possible - # this requires the optimx package and using the control option of lmer()
require(optimx)

# fitting a cosine with a spline (simulated data)
require("rms", quietly=TRUE, character=TRUE)
```
require("lme4", quietly=TRUE, character=TRUE)
dfr = makeSplineData.fnc()
table(dfr$Subject)
xylowess.fnc(Y ~ X | Subject, data = dfr)
# the smoother doesn't recognize the cosine function implemented in makeSplineData.fnc()
dev.off()

dfr.lmer = lmer(Y ~ rcs(X, 5) + (1|Subject), data = dfr,
  control = lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")))
plotLMER.fnc(dfr.lmer)

# comparison with ols from Design package
dfr.lm = lm(Y~Subject+rcs(X), data=dfr, x=T, y=T)
dfr$fittedOLS = fitted(dfr.lm)
dfr$fittedLMER = as.vector(dfr.lmer@pp$x \%*% fixef(dfr.lmer))
# we plot the lmer() fit in blue, the ols() fit in red (both adjusted for
# subject S1), and plot the underlying model in green
plot(dfr[dfr$Subject=="S1",]$X,
  dfr[dfr$Subject=="S1",]$fittedLMER + ranef(dfr.lmer)[[1]]"S1",
  col="blue", ylim = c(24,30), xlab="X", ylab="Y", type="n")
lines(dfr[dfr$Subject=="S1",]$X, dfr[dfr$Subject=="S1",]$fittedOLS, col="red")
lines(dfr[dfr$Subject=="S1",]$X, dfr[dfr$Subject=="S1",]$fittedLMER, col="blue")
lines(dfr[dfr$Subject=="S1",]$X, dfr[dfr$Subject=="S1",]$y+ ranef(dfr.lmer)[[1]]"S1",
  col="green")
legend(23,30,"cos(X)", "lmer (S1)", "ols (S1)", lty=rep(1,3),
  col=c("green", "blue", "red"))

# a model with a raw polynomial
bg.lmer = lmer(LogRT ~ PC1+PC2+PC3 + ReadingScore +
  poly(OrthLength, 2, raw=TRUE) + LogFrequency + LogFamilySize +
  (1|Word) + (1|Subject)+(O+OrthLength|Subject) +
  (O+LogFrequency|Subject), data = beginningReaders,
  control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")))
pars = par()
pars(mfrow=c(3,3), mar=c(5,5,1,1))
plotLMER.fnc(bg.lmer, fun=exp, ylabel = "RT (ms)")

# a model with an interaction involving numeric predictors

danish.lmer = lmer(LogRT ~ PC1 + PC2 + PrevError + Rank +
  ResidSemRating + ResidFamSize + LogWordFreq+LogAffixFreq+Sex +
  poly(LogCUP, 2, raw=TRUE) + LogUP + LogCUPtoEnd +
  (1|Subject) + (1|Word) + (1|Affix), data = danish,
```r
control=lmerControl(optimizer="optimx", optCtrl=list(method="nlminb"))
danish.lmerA = lmer(LogRT ~ PC1 + PC2 + PreError + Rank +
            ResidSemRating + ResidFamSize + LogWordFreq*LogAffixFreq*Sex +
            poly(LogCUP, 2, raw=TRUE) + LogUP + LogCUPtoEnd +
            (1|Subject) + (1|Word) + (1|Affix), data = danish,
            subset=abs(scale(resid(danish.lmer)))<2.5,
            control=lmerControl(optimizer="optimx", optCtrl=list(method="nlminb"))

# plot for reference level of Sex
plotLMER.fnc(danish.lmerA, pred = "LogAffixFreq",
              intr=list("LogWordFreq", round(quantile(danish$LogWordFreq),3), "beg",
                            list(c("red", "green", "blue", "yellow", "purple"), rep(1,5)),
                            ylim=c(6.5,7.0))

# this model has a significant three-way interaction
# for visualization, we can either relevel Sex and refit,
# or make use of the control option. First releveling:

danish$Sex=relevel(danish$Sex, "F")
danish.lmerF = lmer(LogRT ~ PC1 + PC2 + PreError + Rank +
            ResidSemRating + ResidFamSize + LogWordFreq*LogAffixFreq*Sex +
            poly(LogCUP, 2, raw=TRUE) + LogUP + LogCUPtoEnd +
            (1|Subject) + (1|Word) + (1|Affix), data = danish,
            control=lmerControl(optimizer="optimx", optCtrl=list(method="nlminb"))

danish$Sex=relevel(danish$Sex, "M")
danish.lmerM = lmer(LogRT ~ PC1 + PC2 + PreError + Rank +
            ResidSemRating + ResidFamSize + LogWordFreq*LogAffixFreq*Sex +
            poly(LogCUP, 2, raw=TRUE) + LogUP + LogCUPtoEnd +
            (1|Subject) + (1|Word) + (1|Affix), data = danish,
            control=lmerControl(optimizer="optimx", optCtrl=list(method="nlminb"))

# Next preparing for using the control option:
#
# names(fixef(danish.lmer))[[10]] # SexM
# unique(danish.lmer@pp$X[,10]) # 1 0

par(mfrow=c(2,2))

plotLMER.fnc(danish.lmer, pred="LogWordFreq", ylim=c(6.5,7.0),
              intr=list("LogAffixFreq", round(quantile(danish$LogAffixFreq),2), "end"),
              control=list("SexM", 0))
mtext("females", line=1.5, cex=0.9)

plotLMER.fnc(danish.lmerF, pred="LogWordFreq", ylim=c(6.5,7.0),
              intr=list("LogAffixFreq", round(quantile(danish$LogAffixFreq),2), "end"),
              control=list("SexM", 1))
mtext("males", line=1.5, cex=0.9)

plotLMER.fnc(danish.lmerF, pred="LogWordFreq", ylim=c(6.5,7.0),
              intr=list("LogAffixFreq", round(quantile(danish$LogAffixFreq),2), "end"))
mtext("females", line=1.5, cex=0.9)

plotLMER.fnc(danish.lmerM, pred="LogWordFreq", ylim=c(6.5, 7.0),
              intr=list("LogAffixFreq", round(quantile(danish$LogAffixFreq),2), "end"))
```
intr=list("LogAffixFreq", round(quantile(danish$LogAffixFreq),2), "end");
mtext("males", line=1.5, cex=0.9)

par(mfrow=c(1,1))

# calculating effect sizes, defined as max - min
# effect size for a covariate

dfr = plotLMER.fnc(danish.lmerA, pred = "LogCUP", withList=TRUE)
max(dfr$LogCUP$Y)-min(dfr$LogCUP$Y)

# effect size for a factor

dfr = plotLMER.fnc(danish.lmerA, pred = "PrevError", withList=TRUE)
max(dfr$PrevError$Y)-min(dfr$PrevError$Y)

# effect sizes for the quantiles in an interaction plot

dfr = plotLMER.fnc(danish.lmerA, pred = "LogAffixFreq",
    withList=TRUE,
intr=list("LogWordFreq", round(quantile(danish$LogWordFreq),3), "beg"),
unlist(lapply(dfr$LogAffixFreq, FUN=function(X)return(max(X$Y)-min(X$Y))))

# plotting an interaction between two factors

danish$WordFreqFac = danish$LogWordFreq > median(danish$LogWordFreq)
danish.lmer2 = lmer(LogRT ~ WordFreqFac*Sex +
    (1|Subject) + (1|Word) + (1|Affix), data = danish,
    control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")))

plotLMER.fnc(danish.lmer2, pred = "Sex",
intr=list("WordFreqFac", c("TRUE", "FALSE"), "end",
    list(c("red", "blue"), rep(1,2)),
ylimit=c(6.7,6.9), cexsize=1.0, addlines=TRUE)

# a generalized linear mixed-effects model

dative.lmer = glmer(RealizationOfRecipient ~
    AccessOfTheme + AccessOfRec + LengthOfRecipient + AnimacyOfRec +
    AnimacyOfTheme + PronomOfTheme + DefinOfTheme + LengthOfTheme +
    SemanticClass + Modality + (1|Verb),
data = dative, family = "binomial",
control=glmerControl(optimizer="optimx",optCtrl=list(method="nlminb"))

plotlogistic.fit.fnc

Plot for goodness of fit of logistic regression

Description

This function plots observed proportions against mean predicted probabilities. For a good fit, points should be approximately on a straight line.

Usage

plotlogistic.fit.fnc(x, data, method, where, scalesize, ...)

Arguments

- **x**: A logistic regression model fitted with `lmer` or `lrm`.
- **data**: The data set to which the model was fitted.
- **method**: Either "cut", in which case the vector of cut-off points supplied by the "where" argument will be used to partition the fitted probabilities, or "shingle", in which a shingle (using `equal.count` and its defaults) will be used.
- **where**: A vector of cut-off points for partitioning the vector of fitted probabilities, by default `seq(0, 1, by=0.1)`.
- **scalesize**: A positive real <= 1. If not NA (the default), the circles representing data points in the graph are scaled to reflect the number of data points in the underlying data set. The scalesize parameter specifies how large the largest circle will be compared to 1 inch. For counts with large outliers, small values of scalesize are better. See example below.
- **...**: other parameters to be passed through to plotting functions.

Value

A plot is produced on the graphics device. The R-squared value shown above the plot represents the correlation between the X and Y values in the plot. It does NOT represent the R-squared of the `lrm` or `lmer` model.
preparePredictor.fnc

determine X and Y values for a given (sub)plot

Description

this function figures out the X and Y values for a given (sub)plot, including upper and lower 95% HPD intervals

Author(s)

R. H. Baayen

Examples

```r
## Not run:
data(dative)
require(lme4)
require(rms)
require(lmerTest)
require(optimx)

dative.lrm = lrm(RealizationOfRecipient ~ AccessOfTheme + AccessOfRec + LengthOfRecipient + AnimacyOfRec + AnimacyOfTheme + PronomOfTheme + DefinOfTheme + LengthOfTheme + SemanticClass + Modality, data = dative)

dative.glmm = glmer(RealizationOfRecipient ~ AccessOfTheme + AccessOfRec + LengthOfRecipient + AnimacyOfRec + AnimacyOfTheme + PronomOfTheme + DefinOfTheme + LengthOfTheme + SemanticClass + Modality + (1|Verb), control=glmerControl(optimizer="optimx",optCtrl=list(method="nlminb")), data = dative, family = "binomial")

par(mfrow=c(2,2))
plotlogistic.fit.fnc (dative.lrm, dative)
mttext("lrm", 3, 3)
plotlogistic.fit.fnc (dative.glmm, dative)
mttext("lmer", 3, 3)
plotlogistic.fit.fnc (dative.lrm, dative, scalesize=0.2)
mttext("lrm", 3, 3)
plotlogistic.fit.fnc (dative.glmm, dative, method="shingle")
mttext("lmer", 3, 3)
par(mfrow=c(1,1))

## End(Not run)
```
Usage
preparePredictor.fnc(pred, model, m, ylabel, fun, val, xlabel, ranefs, ...)

Arguments
pred character string denoting predictor to be plotted on horizontal axis
model model fit by lmer
m matrix as produced by makeDefaultMatrix.fnc
ylabel label for vertical axis (if other than name of dependent variable
fun character string denoting transformation function for dependent variable, currently only "plogis" or "exp"
val value of interacting variable
xlabel label for horizontal axis
ranefs a three-element list Group, Level, Predictor, specifying a random-effect Group (e.g. Subject), a level (e.g., S10) and a value (e.g., LogFrequency) for which partial effects have to be calibrated; implemented only for mcmcMat=NA.
... further graphical parameters, currently not implemented

Value
A data frame with values to be plotted, with columns
X values of predictor
Y fitted values
Type logical for whether predictor is factor
Interaction logical for whether predictor is interacting predictor
Levels for factors, the factor level names (only present for factors)

Note
not intended for independent use

Author(s)
R. H. Baayen

See Also
See Also as plotLMER.fnc

Examples
## Not run: Not intended for independent use.
Primed lexical decision latencies for neologisms ending in \textit{-heid}.

Usage

`data(primingHeid)`

Format

A data frame with 832 observations on the following 13 variables.

- \texttt{subject} a factor with subjects as levels.
- \texttt{word} a factor with words as levels.
- \texttt{trial} a numeric vector for the rank of the trial in its experimental list.
- \texttt{rt} a numeric vector with log-transformed lexical decision latencies.
- \texttt{condition} a factor coding the priming treatment, with levels \texttt{baseheid} (prime is the base word) and \texttt{heid} (the prime is the neologism).
- \texttt{rating} a numeric vector for subjective frequency estimates.
- \texttt{frequency} a numeric vector for log-transformed frequencies of the whole word.
- \texttt{basefrequency} a numeric vector for the log-transformed frequencies of the base word.
- \texttt{lengthinletters} a numeric vector coding orthographic length in letters.
- \texttt{familysize} a numeric vector for the log-transformed count of the word’s morphological family.
- \texttt{numberofsynsets} a numeric vector for the number of synonym sets in WordNet in which the base is listed.
- \texttt{responsetoprime} a factor with levels \texttt{correct} and \texttt{incorrect} for the response to the prime.
- \texttt{rttoprime} a numeric vector for the log-transformed reaction time to the prime.

References


Examples

```r
## Not run: data(primingHeid)

require(lme4)
require(lmerTest)
require(optimx)
```
**primingHeidPrevRT**

Primed lexical decision latencies for neologisms ending in -heid

---

### Description

Primed lexical decision latencies for Dutch neologisms ending in the suffix -heid, with information on RTs to preceding trials added to the data already in primingHeid.

### Usage

```r
data(primingHeidPrevRT)
```

### Format

A data frame with 832 observations on the following 17 variables.

- **subject**: a factor with subjects as levels.
- **word**: a factor with words as levels.
- **trial**: a numeric vector for the rank of the trial in its experimental list.
- **rt**: a numeric vector with log-transformed lexical decision latencies.
- **condition**: a factor coding the priming treatment, with levels baseheid (prime is the base word) and heid (the prime is the neologism).
- **rating**: a numeric vector for subjective frequency estimates.
- **frequency**: a numeric vector for log-transformed frequencies of the whole word.
- **basefrequency**: a numeric vector for the log-transformed frequencies of the base word.
- **lengthinletters**: a numeric vector coding orthographic length in letters.
- **familysize**: a numeric vector for the log-transformed count of the word’s morphological family.
- **numberofsynsets**: a numeric vector for the number of synonym sets in WordNet in which the base is listed.
ResponseToPrime  a factor with levels correct and incorrect for the response to the prime.
RTToPrime   a numeric vector for the log-transformed reaction time to the prime.
RTmin1     a numeric vector for reaction time in ms to the item preceding the target.
RTmin2     a numeric vector for reaction time in ms to the item preceding the target by two trials.
RTmin3     a numeric vector for reaction time in ms to the item preceding the target by three trials.
RTmin4     a numeric vector for reaction time in ms to the item preceding the target by four trials.

References

Examples
```r
## Not run:
data(primingHeidPrevRT)
require(lme4)
require(optimx)
require(lmerTest)

primingHeid.lmer = lmer(RT ~ RTtoPrime * ResponseToPrime + Condition +
log(RTmin1) + (1|Subject) + (1|Word), data = primingHeidPrevRT,
control=lmerControl(optimizer="optimx", optCtrl=list(method="nlminb")))
summary(primingHeid.lmer)

## End(Not run)
```

print.corres  

*Print method for correspondence object*

**Description**
Prints eigenvalues and eigenvalue rates for a correspondence object.

**Usage**
```
## S3 method for class 'corres'
print(x, ...)
```

**Arguments**
x A correspondence object.
...
other parameters to be passed through to plotting functions.

**Value**
Report of eigenvalues and eigenvalue rates.
Author(s)

R. H. Baayen

See Also

See also `corres.fnc`.

Examples

```r
## Not run:
data(oldFrench)
oldFrench.ca = corres.fnc(oldFrench)
oldFrench.ca

## End(Not run)
```

print.growth  

*Print method for growth objects.*

Description

Print method for growth objects.

Usage

```r
## S3 method for class 'growth'
print(x, ...)
```

Arguments

x  
A growth object, as produced by `growth.fnc`.

...  
other parameters to be passed through to plotting functions.

Value

The data frame with chunk sizes and associated vocabulary statistics is printed. To access the data frame that is being shown, use `<my.growth.object>@data$data`.

Author(s)

R. H. Baayen

See Also

See also `growth.fnc`.  

Examples

```r
## Not run:
data(alice)
alice.growth = growth.fnc(alice)
alice.growth
# for accessing the printed data frame:
alice.growth@data$data[1:4,]

## End(Not run)
```

---

**pvals.fnc**

*Compute p-values and MCMC confidence intervals for mixed models*

**Description**

This function used to calculate p-values and HPD intervals for the parameters of models fitted with `lmer`.

As MCMC is no longer supported by lme4, this function is now obsolete and does no longer produce any output, other than a warning.

See the lme4 function `pvalues()` for alternatives.

**Usage**

```r
pvals.fnc(object, ...)
```

**Arguments**

- `object` a `LMM` or `GLMM` model object of class `lmerMod`
- `...` Optional arguments that can be passed down.

**Value**

A warning.

**Author(s)**

R. H. Baayen

**See Also**

`pvalues`
### Examples

```r
## Not run:
data(primingHeid)
library(lme4)

# remove extreme outliers
primingHeid = primingHeid[primingHeid$RT < 7.1,]

# fit mixed-effects model

# we will stay as close to the older optimizer of lme4 as possible -
# this requires the optimx package and using the control option of lmer()
require(optimx)
require(lmerTest)

primingHeid.lmer = lmer(RT ~ RTtoPrime * ResponseToPrime +
                       Condition + (1|Subject) + (1|Word), data = primingHeid,
                       control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")))
summary(primingHeid.lmer)
amova(primingHeid.lmer)

## End(Not run)
```

### Description

Simulated lexical decision latencies with SOA as treatment, traditionally requiring an analysis using quasi-F ratios, as available in Raaijmakers et al. (1999).

### Usage

```r
data(quasif)
```

### Format

A data frame with 64 observations on the following 4 variables.

- **Subject** a factor coding subjects.
- **RT** a numeric vector for simulated reaction times in lexical decision.
- **Item** a factor coding items.
- **SOA** a factor coding SOA treatment with levels long and short.
Source

Examples
```r
## Not run:
data(quasif)
items.quasif.fnc(quasif)
## End(Not run)
```

quasif.fnc  Quasi-F test

Description
The textbook Quasi-F test for a design with subjects, items, and a single factorial predictor. Included for educational purposes for this specific design only.

Usage
```r
quasiF.fnc(ms1, ms2, ms3, ms4, df1, df2, df3, df4)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ms1</td>
<td>Mean squares Factor</td>
</tr>
<tr>
<td>ms2</td>
<td>Mean squares Item:Subject</td>
</tr>
<tr>
<td>ms3</td>
<td>Mean squares Factor:Subject</td>
</tr>
<tr>
<td>ms4</td>
<td>Mean squares Item</td>
</tr>
<tr>
<td>df1</td>
<td>Degrees of freedom Factor</td>
</tr>
<tr>
<td>df2</td>
<td>Degrees of freedom Item:Subject</td>
</tr>
<tr>
<td>df3</td>
<td>Degrees of freedom Factor:Subject</td>
</tr>
<tr>
<td>df4</td>
<td>Degrees of freedom Item</td>
</tr>
</tbody>
</table>

Value
A list with components

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Quasi-F value.</td>
</tr>
<tr>
<td>df1</td>
<td>degrees of freedom numerator.</td>
</tr>
<tr>
<td>df2</td>
<td>degrees of freedom denominator.</td>
</tr>
<tr>
<td>p</td>
<td>p-value.</td>
</tr>
</tbody>
</table>
Author(s)

R. H. Baayen

See Also

See Also as quasiFsim.fnc.

Examples

data(quasif)

quasif.lm = lm(RT ~ SOA + Item + Subject +
SOA:Subject + Item:Subject, data = quasif)
quasif.aov = anova(quasif.lm)
quasif.fnc(quasif.aov["SOA","Mean Sq"],
quasif.aov["Item:Subject", "Mean Sq"],
quasif.aov["SOA:Subject", "Mean Sq"],
quasif.aov["Item", "Mean Sq"],
quasif.aov["SOA","Df"],
quasif.aov["Item:Subject", "Df"],
quasif.aov["SOA:Subject", "Df"],
quasif.aov["Item", "Df"])

# much simpler is
quasiFsim.fnc(quasif)$quasiF

quasiFsim.fnc Quasi-F test for specific simple design

Description

This function carries out a Quasi-F test for data with columns labelled SOA, Subject, Item. This function is called by simulate.quasif.fnc, and is not intended for general use.

Usage

quasiFsim.fnc(dat)

Arguments

dat A data frame with RT (or RTsim), SOA, Subject and Item as predictors.
ratings

Value
A list with components

\( p \)
  The p-value of the quasi-F test.

\( \text{data} \)
  The input data.

\( \text{model} \)
  The linear model fitted to the data.

\( \text{qF} \)
  A list with F, df1, df2 and p-value of quasi-F test.

Author(s)
R. H. Baayen

See Also
See Also \texttt{quasiF.fnc}.

Examples

\begin{verbatim}
data(quasif)
quasiFsim.fnc(quasif)$quasiF
\end{verbatim}

---

\textbf{ratings}  
\textit{Ratings for 81 English nouns}

Description
Subjective frequency ratings, ratings of estimated weight, and ratings of estimated size, averaged over subjects, for 81 concrete English nouns.

Usage

data(ratings)

Format
A data frame with 81 observations on the following 14 variables.

\texttt{Word}  a factor with words as levels.
\texttt{Frequency}  a numeric vector of logarithmically transformed frequencies
\texttt{FamilySize}  a numeric vector of logarithmically transformed morphological family sizes.
\texttt{SynsetCount}  a numeric vector with logarithmically transformed counts of the number of synonym sets in WordNet in which the word is listed.
\texttt{Length}  a numeric vector for the length of the word in letters.
\texttt{Class}  a factor with levels \texttt{animal} and \texttt{plant}.
\texttt{FreqSingular}  a numeric vector for the frequency of the word in the singular.
FreqPlural a numeric vector with the frequency of the word in the plural.
DerivEntropy a numeric vector with the derivational entropies of the words.
Complex a factor coding morphological complexity with levels complex and simplex.
RInfl a numeric vector coding the log of ratio of singular to plural frequencies.
MeanWeightRating a numeric vector for the estimated weight of the word’s referent, averaged over subjects.
MeanSizeRating a numeric vector for the estimated size of the word’s referent, averaged over subjects.
MeanFamiliarity a numeric vector with subjective frequency estimates, averaged over subjects.

Source

Data collected together with Jen Hay at the University of Canterbury, Christchurch, New Zealand, 2004.

Examples

```r
## Not run:
data(ratings)
ratings.lm = lm(meanSizeRating ~ meanFamiliarity * Class + I(meanFamiliarity^2), data = ratings)
ratings$fitted = fitted(ratings.lm)
plot(ratings$meanFamiliarity, ratings$meanSizeRating, xlab = "mean familiarity", ylab = "mean size rating", type = "n")
text(ratings$meanFamiliarity, ratings$meanSizeRating, substr(as.character(ratings$Class), 1, 1), col = 'darkgrey')
plants = ratings[ratings$Class == "plant",]
animals = ratings[ratings$Class == "animal",]
plants = plants[order(plants$MeanFamiliarity),]
animals = animals[order(animals$MeanFamiliarity),]
lines(plants$MeanFamiliarity, plants$fitted)
lines(animals$MeanFamiliarity, animals$fitted)
## End(Not run)
```

regularity

Regular and irregular Dutch verbs

Description

Regular and irregular Dutch verbs and selected lexical and distributional properties.
Usage

data(regularity)

Format

A data frame with 700 observations on the following 13 variables.

Verb a factor with the verbs as levels.

WrittenFrequency a numeric vector of logarithmically transformed frequencies in written Dutch (as available in the CELEX lexical database).

NcountStem a numeric vector for the number of orthographic neighbors.

VerbalSynsets a numeric vector for the number of verbal synsets in WordNet.

MeanBigramFrequency a numeric vector for mean log bigram frequency.

InflectionalEntropy a numeric vector for Shannon’s entropy calculated for the word’s inflectional variants.

Auxiliary a factor with levels hebben, zijn and zijnheb for the verb’s auxiliary in the perfect tenses.

Regularity a factor with levels irregular and regular.

LengthInLetters a numeric vector of the word’s orthographic length.

FamilySize a numeric vector for the number of types in the word’s morphological family.

Valency a numeric vector for the verb’s valency, estimated by its number of argument structures.

NVRatio a numeric vector for the log-transformed ratio of the nominal and verbal frequencies of use.

WrittenSpokenRatio a numeric vector for the log-transformed ratio of the frequencies in written and spoken Dutch.

References


Examples

## Not run:
data(regularity)

# predicting regularity with a logistic regression model

library(rms)
regularity.dd = datadist(regularity)
options(datadist = 'regularity.dd')
selfPacedReadingHeid

regularity.lrm = lrm(Regularity ~ WrittenFrequency +
rcs(FamilySize, 3) + NcountStem + InflectionalEntropy +
Auxiliary + Valency + NVratio + WrittenSpokenRatio,
data = regularity, x = TRUE, y = TRUE)

anova(regularity.lrm)

# ---- model validation

validate(regularity.lrm, bw = TRUE, B = 200)
pentrace(regularity.lrm, seq(0, 0.8, by = 0.05))
regularity.lrm.pen = update(regularity.lrm, penalty = 0.6)
regularity.lrm.pen

# ---- a plot of the partial effects

plot(Predict(regularity.lrm.pen))

# predicting regularity with a support vector machine

library(e1071)
regularity$AuxNum = as.numeric(regularity$Auxiliary)
regularity.svm = svm(regularity[, -c(1,8,10)], regularity$Regularity, cross=10)
summary(regularity.svm)

## End(Not run)

---

**selfPacedReadingHeid**  
*Self-paced reading latencies for Dutch neologisms*

**Description**

Self-paced reading latencies for Dutch neologisms ending in the suffix `-heid`.

**Usage**

data(selfPacedReadingHeid)

**Format**

A data frame with 1280 observations on the following 18 variables.

- **Subject**  a factor with subjects as levels.
- **Word**  a factor with words as levels.
- **RT**  a numeric vector with logarithmically transformed reading latencies.
- **RootFrequency**  a numeric vector for the logarithmically transformed frequency of the lowest-level base of the neologism (e.g., `lob` in `[[lob]+ig]+heid`).
- **Condition**  a factor for the priming conditions with levels baseheid (neologism is preceded 40 trials back by its base word) and heidheid (the neologism is preceded 40 trials back by itself).
Rating  a numeric vector for the word’s subjective frequency estimate.
Frequency  a numeric vector for the neologism’s frequency (all zero).
BaseFrequency  a numeric vector for the base adjective underlying the neologism (e.g., lobbig in {}
LengthInLetters  a numeric vector coding word length in letters.
FamilySize  a numeric vector for the logarithmically transformed count of a word’s morphological
NumberOfSynsets  a numeric vector for the count of synonym sets in WordNet in which the word
RT4WordsBack  a numeric vector for the log-transformed reading latencies four trials back.
RT3WordsBack  a numeric vector for the log-transformed reading latencies three trials back.
RT2WordsBack  a numeric vector for the log-transformed reading latencies two trials back.
RT1WordBack  a numeric vector for the log-transformed reading latencies one trial back.
RT1WordLater  a numeric vector for the log-transformed reading latencies one trial later.
RT2WordsLater  a numeric vector for the log-transformed reading latencies two trials later.
RTtoPrime  a numeric vector for the log-transformed reading latency for the prime.

References

Examples
## Not run:
data(selfPacedReadingHeid)

# data validation
plot(sort(selfPacedReadingHeid$RT))
selfPacedReadingHeid = selfPacedReadingHeid[selfPacedReadingHeid$RT > 5 &
selfPacedReadingHeid$RT < 7.2,]

# fitting a mixed-effects model
require(lme4)
require(lmerTest)
require(optimx)
x = selfPacedReadingHeid[,12:15]
x.pr = prcomp(x, center = TRUE, scale = TRUE)
selfPacedReadingHeid$PC1 = x.pr$x[,1]

selfPacedReadingHeid.lmer = lmer(RT ~ RTtoPrime + LengthInLetters +
PC1 * Condition + (1|Subject) + (1|Word),
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")),
data = selfPacedReadingHeid)
summary(selfPacedReadingHeid.lmer)
shadenormal.fnc

# model criticism
selfPacedReadingHeid.lmerA = lmer(RT ~ RTtoPrime + LengthInLetters +
PC1 * Condition + (1|Subject) + (1|Word),
control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb"),
data = selfPacedReadingHeid[abs(scale(resid(selfPacedReadingHeid.lmer))) < 2.5, ])

qqnorm(resid(selfPacedReadingHeid.lmerA))
summary(selfPacedReadingHeid.lmerA)

## End(Not run)

---

**shadenormal.fnc**

*Shade rejection region for normal probability density function*

**Description**

This function plots the standard normal probability density function and shades the rejection region.

**Usage**

```r
shadenormal.fnc(qnts = c(0.025, 0.975))
```

**Arguments**

- `qnts`  
  A numeric vector with the Z-scores of the boundaries of the lower and upper rejection regions.

**Value**

A plot on the graphics device.

Type `shadenormal.fnc` to see the code. The polygon() function used for the shaded areas takes a sequence of X and Y coordinates, connects the corresponding points, and fills the area(s) enclosed with a specified color. To understand the use of polygon(), one can best think of making a polygon with a set of pins, a thread, and a board. Outline the polygon by placing the pins on the board at the corners of the polygon. First fasten the thread to one of the pins, then connect the thread to the second pin, from there to the third pin, and so on, until the first pin has been reached. What polygon() requires as input is a vector of the X-coordinates of the pins, and a vector of their Y-coordinates. These coordinates should be in exactly the order in which the thread is to be connected from pin to pin.

For shading the left rejection area, we specify the vectors of X and Y coordinates, beginning at the leftmost point of the tail, proceeding to the right edge of the shaded area, then up, and finally to the left and down to the starting point, thereby closing the polygon. The X-coordinates are therefore specified from left to right, and then from right to left. The corresponding Y-coordinates are all the zeros necessary to get from $-3$ to $1.96$ (the default, qnorm(0.025)), and then the Y-coordinates of the density in reverse order to return to where we began.
show.growth

Author(s)

R. H. Baayen

Examples

```r
## Not run:
shadenormal.fnc()

## End(Not run)
```

---

**show.growth**  
Plot method for growth objects.

Description

A print method for growth objects created with growth.fnc.

Usage

```r
## S3 method for class 'growth'
show(x)
```

Arguments

- `x`  
  A growth object.

Value

Prints growth object. To access the data frame embedded in the growth object, use `<my_growth_object>@data$data`.

Author(s)

R. H. Baayen

See Also

See Also **growth.fnc**.

Examples

```r
## Not run:
data(alice)
alice.growth = growth.fnc(alice, chunks = c(5000, 10000, 15000))
alice.growth

## End(Not run)
```
**Description**

Simulated data set for illustrating shrinkage.

**Usage**

```r
data(shrinkage)
```

**Format**

A data frame with 200 observations on the following 6 variables.

- `intercept` a numeric vector for the intercept.
- `frequency` a numeric vector for word frequency.
- `subject` a factor for subjects with levels S1, S2, ..., S10.
- `error` a numeric vector for residuals.
- `ranef` a numeric vector for random effect.
- `RT` a numeric vector for simulated RTs.

**Examples**

```r
## Not run
data(shrinkage)

require(lme4)
require(lmerTest)
require(lmList)

shrinkage.lmer = lmer(RT ~ frequency + (1|subject),
  data = shrinkage,
  control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")))
shrinkage.lmList = lmList(RT ~ frequency | subject, data = shrinkage)

# and visualize the difference between random regression
# and mixed-effects regression

mixed = coef(shrinkage.lmer)[[1]]
random = coef(shrinkage.lmList)
subj = unique(shrinkage[,c("subject", "ranef")])
subj = subj[order(subj$subject),]
subj$random = random[,1]
subj$random = mixed[,1]
subj = subj[order(subj$random),]
subj$rank = 1:nrow(subj)
```
simulateLatinsquare.fnc

Simulate simple Latin Square data and compare models

Description

This function creates a user-specified number of simulated datasets with a Latin Square design, and compares mixed-effects models with the by-subject anova.

Usage

simulateLatinsquare.fnc(dat, with = TRUE, trial = 0, nruns = 100, nsub = NA, nitem = NA, ...)  

Arguments

dat     A data frame with the structure of the data set latinsquare.  
with    Logical, if TRUE, effect of SOA built into the data.  
trial   A number which, if nonzero, gives the magnitude of a learning or a fatigue effect.  
nruns   A number indicating the required number of simulation runs.  
nsub    A number for the number of subjects.  
nitem   A number for the number of items.  
...     other parameters to be passed through to plotting functions.

Value

A list with components

alpha05 Description of 'comp1'  
alpha01 proportion of runs in which predictors are significant at the 05 significance level.  
res     Data frame with simulation results.  
with    Logical, TRUE if SOA effect is built into the simulations.
simulateQuasif.fnc

Author(s)
R. H. Baayen

Examples

```r
## Not run:
data(latinsquare)
\dontrun{
  library(lme4)
  simulateLatinsquare.fnc(latinsquare, nruns=100)
}
## End(Not run)
```

---

**simulateQuasif.fnc**  
*Simulate data for quasi-F analysis and compare models*

**Description**

This function creates a user-specified number of simulated datasets, and compares mixed-effects models with quasi-F and F1 and F2 analyses. It should be run with the version of R and the version of languageR used by Baayen, Davidson & Bates (2008, JML), as mcmcsamp no longer supports models with random correlation parameters.

**Usage**

`simulateQuasif.fnc(dat, with = TRUE, nruns = 100, nsub = NA, nitem = NA, ...)`

**Arguments**

- `dat`  
  Data frame with a data set with as variables Subject, Item and SOA, as in the quasif data set.

- `with`  
  Logical, if TRUE, an effect of SOA is built into the simulation.

- `nruns`  
  Integer for the number of simulation runs.

- `nsub`  
  Integer denoting the number of subjects.

- `nitem`  
  Integer denoting the number of items.

- `...`  
  Other parameters to be passed through to plotting functions.

**Details**

Model parameters are estimated from the input data set.

For each completed simulation run, a dot is added to the R console.
**Value**

A list with components

- `alpha05`: Description of 'comp1'
- `alpha01`: proportion of runs in which predictors are significant at the 05 significance level.
- `res`: Data frame with simulation results.
- `with`: Logical, TRUE if SOA effect is built into the simulations.

**Author(s)**

R. H. Baayen

**See Also**

See also `subjects.quasif.fnc`.

**Examples**

```r
## Not run:
data(quasif)
library(lme4)

quasif.sim = simulateQuasif.fnc(quasif, nruns = 1000, with = TRUE)
quasif.sim$alpha05

## End(Not run)
```

---

**simulateRegression.fnc**

*Simulate regression data and compare models*

**Description**

This function creates a user-specified number of simulated regression datasets, and compares mixed-effects regression with random regression, by-subject regression, by-item regression, and by-subject plus by-item regression. Optionally, an effect of learning or fatigue can be incorporated.

**Usage**

```r
simulateRegression.fnc(beta = c(400, 2, 6, 4), nitem = 20, nsubj = 10,
stdevItem = 40, stdevSubj = 80, stdevError = 50, nruns = 100, learn = FALSE,
learnRate = 10, ...)
```
simulateRegression.fnc

Arguments

- **beta**: A numeric vector with beta weights for the intercept and three numeric predictors.
- **nitem**: A number specifying the number of items.
- **nsubj**: A number specifying the number of subjects.
- **stdevItem**: A number specifying the standard deviation of the Item random effect.
- **stdevSubj**: A number specifying the standard deviation of the Subject random effect.
- **stdevError**: A number specifying the standard deviation of the Residual Error.
- **nruns**: A number specifying the required number of simulated datasets.
- **learn**: A logical that if TRUE, allows an effect of learning or fatigue into the model.
- **learnRate**: A number specifying the learning rate (if negative) or the effect of fatigue (if positive).
- ... other parameters to be passed through to plotting functions.

Value

A list with components

- **alpha05**: proportion of runs in which predictors are significant at the 05 significance level.
- **alpha01**: proportion of runs in which predictors are significant at the 01 significance level.
- **ranef**: mean estimated random effects.

As this may take some time, the index of each completed run is shown on the output device.

Author(s)

R. H. Baayen

See Also

See Also `make.reg.fnc`.

Examples

```r
# Not run:
library(lme4)
simulateRegression.fnc(beta = c(400, 2, 6, 4), nruns = 5)

\dontrun(simulateRegression.fnc(beta = c(400, 2, 6, 0), nruns = 1000, learn = TRUE))

# End(Not run)```
sizeRatings

Size ratings for 81 English concrete nouns

Description

Subjective estimates of the size of the referents of 81 English concrete nouns, collected from 38 subjects.

Usage

data(sizeRatings)

Format

A data frame with 3078 observations on the following 7 variables.

rating a numeric vector with subjective estimates of the size of the word’s referent.

subject a factor with subjects as levels.

word a factor with words as levels.

class a factor with levels animal and plant.

naive a factor with levels naive and notNaive, coding whether the subject new about the purpose of the experiment.

language a factor with levels English and notEnglish coding whether the subject was a native speaker of English.

meanfamiliarity a numeric vector for the by-item mean familiarity ratings.

Details


Examples

```r
## Not run:
data(sizeRatings)
require(lme4)
require(lmerTest)
require(optimx)
sizeRatings.lmer = lmer(Rating ~ class * naive +
  meanfamiliarity * language + (1|subject) + (1|word),
  control=lmerControl(optimizer="optimx",optCtrl=list(method="nlminb")),
data = sizeRatings)
summary(sizeRatings.lmer)

## End(Not run)
```
Relative frequencies of tag trigrams is selected Spanish texts

Description

Relative frequencies of the 120 most frequent tag trigrams in 15 texts contributed by 3 authors.

Usage

data(spanish)

Format

A data frame with 120 observations on 15 variables documented in spanishMeta.

References


Examples

```r
## Not run:
data(spanish)
data(spanishMeta)

# principal components analysis
spanish.t = t(spanish)
spanish.pca = prcomp(spanish.t, center = TRUE, scale = TRUE)
spanish.x = data.frame(spanish.pca$x)
spanish.x = spanish.x[order(rownames(spanish.x)), ]

library(lattice)
splom(~spanish.x[, 1:3], groups = spanishMeta$Author)

# linear discriminant analysis

library(MASS)
spanish.pca.lda = lda(spanish.x[, 1:8], spanishMeta$Author)
plot(spanish.pca.lda)

# cross-validation

n = 8
spanish.t = spanish.t[order(rownames(spanish.t)), ]
predictedClasses = rep("", 15)
for (i in 1:15) {
  training = spanish.t[-i,]
```
spanishFunctionWords

Relative frequencies of function words in selected Spanish texts

Description

Relative frequencies of the 120 most frequent function words in 15 texts contributed by 3 authors.

Usage

data(spanishFunctionWords)

Format

A data frame with 120 observations on 15 variables documented in spanishMeta.

References


Examples

```r
## Not run:
data(spanishFunctionWords)
data(spanishMeta)

# principal components analysis

spanishFunctionWords.t = t(spanishFunctionWords)
spanishFunctionWords.t =
  spanishFunctionWords.t[order(rownames(spanishFunctionWords.t)), ]
spanishFunctionWords.pca =
  prcomp(spanishFunctionWords.t, center = TRUE, scale = TRUE)
```
sdevs = spanishFunctionWords.pca$sdev^2
n = sum(sdevs/sum(sdevs)> 0.05)

# linear discriminant analysis with cross-validation

library(MASS)

predictedClasses = rep("", 15)
for (i in 1:15) {
  training = spanishFunctionWords.t[-i,]
  trainingAuthor = spanishMeta[-i,]$Author
  training.pca = prcomp(training, center = TRUE, scale = TRUE)
  training.x = data.frame(training.pca$x)
  training.x = training.x[order(rownames(training.x)), ]
  training.lda = lda(training[, 1:n], trainingAuthor)
  cl = predict(training.lda, spanishFunctionWords.t[,1:n]$class[i]
  predictedClasses[i] = as.character(cl)
}
ncorrect = sum(predictedClasses==spanishMeta$Author)
sum(dbinom(ncorrect:15, 15, 1/3))

## End(Not run)

---

spanishMeta

Metadata for the spanish and spanishFunctionWords data sets

**Description**

By-text metadata for the spanish and spanishFunctionWords data sets.

**Usage**

data(spanishMeta)

**Format**

A data frame with 15 observations on the following 6 variables.

Author a factor with levels C, M, and V.
YearOfBirth a numeric vector with year of birth of the author.
TextName a factor with codes for the texts as levels (X14458g11 ... X14476g11).
PubDate a numeric vector with data of publication of the text.
Nwords a numeric vector with text sizes in tokens.
FullName a factor with author names: Cela, Mendoza and VargasLlosa.
References


Examples

```r
## Not run:
data(spanishMeta)

## End(Not run)
```

---

spectrum.fnc | Frequency spectrum from text vector

### Description

This function creates a frequency spectrum for a text in character vector form.

### Usage

```r
spectrum.fnc(text)
```

### Arguments

- `text` A character vector containing the words of a text.

### Value

A data frame with as column variables

- `frequency` Word frequencies.
- `freqOfFreq` The frequencies of the word frequencies.

### Author(s)

R. H. Baayen

### References


### See Also

See Also the zipfR package.
splitplot

Examples

```r
## Not run:
data(alice)
alice.spectrum = spectrum.fnc(alice)
head(alice.spectrum)
tail(alice.spectrum)

## End(Not run)
```

splitplot  Simulated data set with split plot design

Description

Simulated lexical decision latencies with priming as treatment and reaction time in lexical decision as dependent variable.

Usage

data(splitplot)

Format

A data frame with 800 observations on the following 11 variables.

- `items` A factor with levels `w1, w2, ..., w40`, coding 40 word items.
- `ritems` The by-word random adjustments to the intercept.
- `list` A factor with levels `lista` and `listb`. The priming effect is counterbalanced for subjects across these two lists, compare table(splitplot$list, splitplot$subjects).
- `rlist` The by-list random adjustments to the intercept.
- `priming` A treatment factor with levels `primed` and `unprimed`.
- `fpriming` The priming effect, -30 for the primed and 0 for the unprimed condition.
- `subjects` A factor with levels `s1, s2, ..., s20` coding 20 subjects.
- `rsubject` The by-subject random adjustments to the intercept.
- `error` The by-observation noise.
- `int` The intercept.
- `RT` The reaction time.

Source

subjects.latinsquare.fnc

By-subject analysis of simple Latin Square data sets

Description

This function is called by simulateLatinsquare.fnc for by-subject analysis of simulated Latin Square datasets. It is not intended for independent use.

Usage

subjects.latinsquare.fnc(dat)

Arguments

dat A data frame with variables RT or RTsim, SOA, Subject, Item, Group and List, as in the latinsquare data set.

Value

A list with components

p The p-value of the by-subject anova.
data The input dataset.
model The fitted model.
subjects.quasif.fnc

Author(s)
R. H. Baayen

See Also
See also simulatelatinsquare.fnc.

Examples

## Not run:
data(latinsquare)
subjects.latinsquare.fnc(latinsquare)$p

## End(Not run)

subjects.quasif.fnc  By-subject analysis of data sets requiring quasi-F ratios

Description
This function is called by simulateQuasif.fnc for by-subject analysis of simulated datasets traditionally requiring quasi-F ratios. It is not intended for independent use.

Usage
subjects.quasif.fnc(dat)

Arguments
dat A data frame with variables RT or RTsim, SOA, Subject and Item.

Value
A list with components
p p-value for by-subject F-test.
data Data set with aggregated subject means.
model Anova table of fitted model.

Author(s)
R. H. Baayen

See Also
See also simulateQuasif.fnc.
Summarize a correspondence object

Description

This function provides a concise summary of a correspondence object.

Usage

## S3 method for class 'corres'
summary(object, n = 2, returnList = FALSE, head = TRUE, ...)

Arguments

- `object`: A correspondence object as produced by `corres`.
- `n`: A number indicating number of dimensions to be summarized.
- `returnList`: Logical, if TRUE, a list is returned with as components the full information on each factor, instead of only the first 6 lines.
- `head`: Logical, if TRUE, first 6 rows of factor summaries are shown.
- `...`: Additional arguments passed on to summaries.

Value

A summary with eigenvalue rates, and coordinates, correlations, and contributions for the factors (by default, 2, unless n is set to a higher number).

Author(s)

R. H. Baayen

See Also

See also `corres.fnc`.
summary.growth

Examples

```r
## Not run:
data(oldFrench)
oldFrench.ca = corres.fnc(oldFrench)
oldFrench.ca
summary(oldFrench.ca)

## End(Not run)
```

---

**summary.growth**  
Summary method for growth objects

**Description**

Summary method for vocabulary growth objects created with `growth.fnc`.

**Usage**

```r
## S3 method for class 'growth'
summary(object, ...)
```

**Arguments**

- `object` A vocabulary growth object.
- `...` other parameters to be passed through to plotting functions.

**Value**

The growth object is printed. For access to the data frame inside the object, use `<my.growth.object>@data$data`.

**Author(s)**

R. H. Baayen

**See Also**

See also `growth.fnc`. 
tail.growth

Show last rows of growth object.

Description

Prints last rows of growth object.

Usage

### S3 method for class 'growth'

```r
tail(x, n = 6, ...)
```

Arguments

- `x` A growth object.
- `n` An integer specifying the number of lines to be shown.
- `...` other parameters to be passed through to plotting functions.

Author(s)

R. H. Baayen

See Also

See Also `growth.fnc`.

Examples

```r
## Not run:
data(alice)
alice.growth = growth.fnc(alice)
tail(alice.growth)

## End(Not run)
```

text2spc.fnc

Create a frequency spectrum from a text vector

Description

This functions takes a text in the form of a character vector as input, and outputs a frequency spectrum object as defined in the `zipfR` package.

Usage

```r
text2spc.fnc(text)
```
**through**

**Arguments**

- `text` A text in the form of a character vector.

**Value**

A `spc` spectrum object as defined in the `zipfR` package.

**Author(s)**

R. H. Baayen

**See Also**

See the documentation for `zipfR` for `spc` objects.

**Examples**

```r
## Not run:
library(zipfR)
data(alice)
alice.spc = text2spc.fnc(alice)
plot(alice.spc)

## End(Not run)
```

---

**through**

*Through the Looking Glass*

**Description**

The text of Lewis Carroll’s ‘Through the Looking Glass’, with punctuation marks removed.

**Usage**

```r
data(through)
```

**Format**

A character vector with 29560 words.

**Source**

The project Gutenberg at [http://www.gutenberg.org/wiki/Main_Page](http://www.gutenberg.org/wiki/Main_Page)
transforming.fnc

transform vector according to specified function

Description

Apply function fun to input vector y

Usage

transforming.fnc(y, fun)

Arguments

y numerical vector (for dependent variable)
fun a function, or NA (in which case no transformation is applied)

Details

exists only to make code more readable

Value

a numerical vector

Note

not intended for independent use

Author(s)

R. H. Baayen

See Also

See Also as plotLMER.fnc

Examples

## Not run: not intended for independent use
**Description**

Frequency (m) and frequency of frequency (Vm) for string types in the Twente News Corpus.

**Usage**

data(twente)

**Format**

A data frame with 4639 observations on the following 2 variables.

- `m` a numeric vector with word frequencies.
- `Vm` a numeric vector with the frequencies of word frequencies.

**Source**

Twente News Corpus.

**Examples**

```r
## Not run:
data(twente)
library(zipfr)
twente.spc = spc(m=twente$m, Vm = twente$Vm)
plot(twente.spc)

## End(Not run)
```

---

**variationLijk**

*Variation in spoken Dutch in the use of the suffix -lijk*

**Description**

This dataset documents variation in the use of the suffix -lijk, as realized in 32 words, in spoken Dutch across region (Flanders versus The Netherlands), sex (females versus males) and education (high versus mid).

**Usage**

data(variationLijk)
**Format**

A data frame with 32 observations on the following 8 variables.

- `nlfemaleHigh` a numeric vector with counts for Dutch females with a mid education level.
- `nlfemaleMid` a numeric vector counts for Dutch females with a high education level.
- `nlmaleHigh` a numeric vector counts for Dutch males with a high education level.
- `nlmaleMid` a numeric vector counts for Dutch males with a mid education level.
- `vlfemaleHigh` a numeric vector counts for Flemish females with a high education level.
- `vlfemaleMid` a numeric vector counts for Flemish females with a mid education level.
- `vlmaleHigh` a numeric vector counts for Flemish males with a high education level.
- `vlmaleMid` a numeric vector counts for Flemish males with a mid education level.

**References**


**Examples**

```r
## Not run:
data(variationLijk)
variationLijk.ca = corres.fnc(variationLijk)
plot(variationLijk.ca, rcex=0.7, ccol="black",
     rcol = rep("blue", nrow(variationLijk)))

## End(Not run)
```

---

**ver**                          

*The Dutch prefix ver-: semantic transparency and frequency*

**Description**

Semantic transparency (dichotomous) and frequency for 985 words with the Dutch prefix `ver-`.

**Usage**

data(ver)

**Format**

A data frame with 985 observations on the following 2 variables.

- `frequency` a numeric vector for the words’ frequency.
- `semanticclass` a factor with levels opaque and transparent coding semantic transparency.
References


Examples

```r
## Not run:
data(verbs)
ver$Frequency = log(ver$Frequency)

plot(density(ver$Frequency))

# plot separate densities for opaque and transparent words

ver.transp = ver[ver$SemanticClass == "transparent",]$Frequency
ver.opaque = ver[ver$SemanticClass == "opaque",]$Frequency

ver.transp.d = density(ver.transp)
ver.opaque.d = density(ver.opaque)
xlimit = range(ver.transp.d$x, ver.opaque.d$x)
ylim = range(ver.transp.d$y, ver.opaque.d$y)
plot(ver.transp.d, lty = 1, col = "black",
xlab = "frequency", ylab = "density",
 xlim = xlimit, ylim = ylimit, main = "")
lines(ver.opaque.d, col = "darkgrey")
legend(6,0.25, lty=rep(1,2), col=c("black", "darkgrey"),
legend=c("transparent", "opaque"))

# test whether the difference is significant
ks.test(jitter(ver.transp), jitter(ver.opaque))

## End(Not run)
```

### verbs

Dative Alternation - simplified data set

Description

A simplified version of the dative data set, used for expository purposes only.

Usage

```r
data(verbs)
```
Format

A data frame with 903 observations on the following 5 variables.

- RealizationOfRec: a factor with levels NP and PP.
- Verb: a factor with the verbs as levels.
- AnimacyOfRec: a factor with levels animate and inanimate.
- AnimacyOfTheme: a factor with levels animate and inanimate.
- LengthOfTheme: a numeric vector coding the length in words of the theme.

References


Examples

data(verbs)
head(verbs)
xtabs(~ RealizationOfRec + AnimacyOfRec, data = verbs)
barplot(xtabs(~ RealizationOfRec + AnimacyOfRec, data = verbs),beside=TRUE)

---

**warlpiri**

**Ergative case marking in Warlpiri**

Description

This data set documents the use of ergative case marking in the narratives of native speakers of Lajamanu Warlpiri (8 children, 13 adults) describing events in picture books.

Usage

data(warlpiri)

Format

A data frame with 347 observations on the following 9 variables.

- Speaker: a factor with speakers as levels.
- Sentence: a factor with sentence as levels.
- AgeGroup: a factor with levels adult and child.
- CaseMarking: a factor with levels ergative and other.
- WordOrder: a factor with levels subInitial (subject initial) and subNotInitial (subject not initial).
- AnimacyOfSubject: a factor with levels animate and inanimate.
- OvertnessOfObject: a factor with levels notOvert and overt.
- AnimacyOfObject: a factor with levels animate and inanimate.
- Text: a factor with levels texta, textb and textc.
weightRatings

References


Examples

```r
## Not run:
data(warlpiri)
require(lme4)
require(lmerTest)
require(optimizer)

warlpiri.lmer = glmer(CaseMarking ~ WordOrder * AgeGroup + AnimacyOfSubject + (1|Text) + (1|Speaker),
control = glmerControl(optimizer="optimx", optCtrl = list(method = "nlminb")),
family = "binomial", data = warlpiri)

summary(warlpiri.lmer)

## End(Not run)
```

weightRatings  Subjective estimates of the weight of the referents of 81 English nouns

Description

Subjective estimates on a seven-point scale of the weight of the referents of 81 English nouns.

Usage

data(weightRatings)

Format

A data frame with 1620 observations on the following 7 variables.

- Subject  a factor with subjects as levels.
- Rating  a numeric vector.
- Trial  a numeric vector with the weight ratings.
- Sex  a factor with levels F and M.
- Word  a factor with words as levels.
- Frequency  a numeric vector with log-transformed lemma frequencies as available in the CELEX lexical database.
- Class  a factor with levels animal and plant.

References

writtenVariationLijk

Variation in written Dutch in the use of the suffix -lijk

Description

This dataset documents variation in the use of the 80 most frequent words ending in the suffix -lijk in written Dutch.

Usage

data(writtenVariationLijk)

Format

A data frame with 560 observations on the following 5 variables.

- **Corpus** a factor with as levels the sampled newspapers: belang (Het Belang van Limburg), gazet (De Gazet van Antwerpen), laatnieu (Het Laatste Nieuws), limburg (De Limburger), nrc (NRC Handelsblad), stand (De Standaard), and tele (De Telegraaf).
- **Word** a factor with the 80 most frequent words ending in -lijk.
- **Count** a numeric vector with token counts in the CONDIV corpus.
- **Country** a factor with levels Flanders and Netherlands.
- **Register** a factor with levels National, Quality and Regional coding the type of newspaper.

References


Examples

```r
## Not run:
data(writtenVariationLijk)
require(lme4)
require(lmerTest)
require(lme4)`
writtenVariationlijk.lmer = glmer(Count ~ Country * Register + (1|Word),
control=glmerControl(optimizer="optimx",optCtrl=list(method="nlminb")),
data = writtenVariationlijk, family = "poisson")

writtenVariationlijk.lmerA = glmer(Count ~ Country * Register + (Country|Word),
control=glmerControl(optimizer="optimx",optCtrl=list(method="nlminb")),
data = writtenVariationlijk, family = "poisson")
anova(writtenVariationlijk.lmer, writtenVariationlijk.lmerA)
summary(writtenVariationlijk.lmerA)

## End(Not run)

---

**xylowess.fnc**

*Trellis scatterplot with smoothers*

### Description
Convenience function for trellis scatterplots with smoothers added.

### Usage

```r
xylowess.fnc(fmla, data,
span = 2/3, symbolcolor = "darkgrey",
linemcolor = "blue", xlabell = "", ylabell = "", ...)
```

### Arguments

- **fmla**: A formula.
- **data**: A dataframe.
- **span**: Span for the smoother.
- **symbolcolor**: Color for plot symbols.
- **linemcolor**: Color for smoother.
- **xlabell**: Label for horizontal axis.
- **ylabell**: Label for vertical axis.
- **...**: Arguments to be passed to methods.

### Value
A trellis scatterplot matrix with smoothers is shown on the graphics device.

### Author(s)
R. H. Baayen
See Also

See also xyplot.

Examples

```r
## Not run:
data(weightRatings)
xylowess.fnc(Rating ~ Frequency | Subject, data = weightRatings,
             xlab = "log Frequency", ylab = "Weight Rating")
## End(Not run)
```

---

### yule.fnc

**Yule’s characteristic constant K**

**Description**

This function calculates Yule’s characteristic constant K given a frequency spectrum.

**Usage**

```r
yule.fnc(spect)
```

**Arguments**

- `spect` A frequency spectrum as generated by `spectrum.fnc`.

**Value**

Yule’s characteristic constant K

**Author(s)**

R. H. Baayen

**References**


**See Also**

See also `spectrum.fnc` and `growth.fnc`. 
Examples

```r
## Not run:
data(alice)
yule.fnc(spectrum.fnc(alice))

## End(Not run)
```

**zipf.fnc**  
Zipf’s rank frequency distribution

**Description**

This function calculates Zipf’s rank-frequency distribution for a text vector, and optionally produces the rank-frequency plot.

**Usage**

```r
zipf.fnc(text, plot = FALSE)
```

**Arguments**

- `text` A character vector containing a text.
- `plot` Logical, if TRUE, a rank-frequency plot is shown on the graphics device.

**Value**

A data frame with variables

- `frequency` Word Frequencies, ordered from large to small.
- `freqOffreq` Frequencies of word frequencies.
- `rank` Zipf rank.

**Author(s)**

R. H. Baayen

**References**


**See Also**

See also `growth.fnc`. 
Examples

```r
## Not run:
data(alice)
alice.zipf = zipf.fnc(alice, plot = TRUE)
head(alice.zipf)

## End(Not run)
```
zinf.fnc, 133

+Topic multivariate
  corres.fnc, 17
 corsup.fnc, 18
  plot.corres, 79
  print.corres, 94
  summary.corres, 120

+Topic package
  languageR-package, 4

+Topic regression
  aovlmer.fnc, 10
  collin.fnc, 13
  degreesOrKnots.fnc, 24
  getKnots.fnc, 37
  getMCMCintervals.fnc, 38
  getPos.fnc, 39
  getRange.fnc, 40
  getRoot.fnc, 41
  implementInteractions.fnc, 49
  item.fnc, 50
  items.quasif.fnc, 51
  make.reg.fnc, 60
  makeDefaultMatrix.fnc, 62
  makeSplineData.fnc, 63
  parsePredName.fnc, 72
  plotAll.fnc, 82
  plotLMER.fnc, 83
  plotlogistic.fit.fnc, 89
  preparePredictor.fnc, 90
  pvals.fnc, 96
  quasif.fnc, 98
  quasifSim.fnc, 99
  simulateLatinSquare.fnc, 108
  simulateQuasif.fnc, 109
  simulateRegression.fnc, 110
  subjects.latinsquare.fnc, 118
  subjects.quasif.fnc, 119
  transforming.fnc, 124
  xlowess.fnc, 131

acf.fnc, 6, 52
affixProductivity, 7
alice, 10
aovlmer.fnc, 10
auxiliaries, 11

beginningReaders, 12

collin.fnc, 13

corres-class, 16
corres.fnc, 17, 19, 80, 95, 120
corsup.fnc, 17, 18
danish, 20
dative, 21
dativeSimplified, 23
degreesOrKnots.fnc, 24
durationsGe, 25
durationsOnt, 26
dutchSpeakersDist, 27
dutchSpeakersDistMeta, 29

danish, 20
dative, 21
dativeSimplified, 23
degreesOrKnots.fnc, 24
durationsGe, 25

dutchSpeakersDist, 27
dutchSpeakersDistMeta, 29

english, 30
etymology, 32

faz, 34
finalDevoicing, 36

growth-class, 42

growth.fnc, 43, 45, 47, 81, 95, 106, 121, 122, 132, 133

growth2vgc.fnc, 44

havelaar, 45
heid, 46
herdan.fnc, 47

imaging, 48

implementInteractions.fnc, 49
item.fnc, 50
items.quasif.fnc, 51

kappa, 14

lags.fnc, 6, 52
languageR (languageR-package), 4
languageR-package, 4
latinsquare, 53
lexdec, 54
lexicalMeasures, 56
lexicalMeasuresClasses, 58
lmerPlotInt.fnc, 59

make.reg.fnc, 50, 60, 111
INDEX

makeDefaultMatrix.fnc, 62
makeSplineData.fnc, 63
moby, 64
mvrnormplot.fnc, 65
nesscg, 66
nessdemog, 66
nessw, 67
oldFrench, 68
oldFrenchMeta, 68
oz, 71
pairs, 72
pairscor.fnc, 71
panel.smooth, 72
parsePredName.fnc, 72
periphrasticDo, 73
phylogeny, 75
plot.corres, 17, 79
plot.growth, 44, 81
plotAll.fnc, 82
plotMER.fnc, 25, 38–41, 50, 62, 63, 73, 83, 89, 116, 132
plotlogistic.fit.fnc, 89
preparePredictor.fnc, 90
primingHeid, 92
primingHeidPrevRT, 93
print.corres, 94
print.growth, 95
pvals.fnc, 96
quasif, 97
quasiF.fnc, 98, 100
quasiFsim.fnc, 99, 99
ratings, 100
regularity, 101
selfPacedReadingHeid, 103
shadenormal.fnc, 105
show.growth, 106
shrinkage, 107
simulateLatinSquare.fnc, 108, 119
simulateQuasif.fnc, 51, 109, 119
simulateRegression.fnc, 50, 61, 110
sizeRatings, 112
spanish, 113
spanishFunctionWords, 114
spanishMeta, 115
spectrum.fnc, 116, 132
splitplot, 117
subjects.latinSquare.fnc, 118
subjects.quasif.fnc, 110, 119
summary.corres, 120
summary.growth, 121
tail.growth, 122
text2spc.fnc, 122
through, 123
transforming.fnc, 124
twente, 125
variationLijk, 125
ver, 126
verbs, 127
warlpiri, 128
weightRatings, 129
writtenVariationLijk, 130
xlowess.fnc, 131
xyplot, 132
yule.fnc, 132
zipf.fnc, 133