Package: quid (via r-universe)

September 8, 2024

Type Package

Title Bayesian Mixed Models for Qualitative Individual Differences

Version 0.0.1

Depends R (>= 3.2.0)

Maintainer Lukas Klima <lukas.klima@live.at>

Description Test whether equality and order constraints hold for all individuals simultaneously by comparing Bayesian mixed models through Bayes factors. A tutorial style vignette and a quickstart guide are available, via vignette(``manual", ``quid"), and vignette(``quickstart", ``quid") respectively. See Haaf and Rouder (2017) <doi:10.1037/met0000156>; Haaf, Klaassen and Rouder (2019) <doi:10.31234/osf.io/a4xu9>; and Rouder & Haaf (2021) <doi:10.5334/joc.131>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports BayesFactor, MCMCpack, checkmate, ellipsis, janitor, purrr, rlang, stringr, methods, magrittr, dplyr, ggplot2, tibble, tidyr, tidyselect, Matrix

Suggests rmarkdown, knitr, testthat

VignetteBuilder knitr

Repository https://lukasklima.r-universe.dev

RemoteUrl https://github.com/lukasklima/quid

RemoteRef HEAD

RemoteSha ab03d9e43ce40e2837e7cfbfbbe1d1a6aef9abcb

Contents

`BFBayesFactorConstraint-class`

S4 class for representing Bayes factor model comparisons and the Bayes factor for user-defined constraints

---

Description

`BFBayesFactorConstraint` is a S4 class that represents the Bayes factors of multiple model comparisons and the Bayes factor of all individual effects adhering to user-defined constraints. Furthermore, it also has slots defined for representing the design matrices of the constraints model, and for representing prior and posterior estimates.

Slots

`generalTestObj` an object of class `BFBayesFactor` representing the model comparisons. All S4 methods as defined in `BFBayesFactor-class` can be used.

`constraints` an object of class `BFConstraint-class`.

`individualEffects` a list containing a named vector for each level of the effect defined in the constraints. Effects are comprised of the common effect plus the individual deviation from it.

`posteriorMean` a named numeric giving the mean of the common effect.

`posteriorSD` a named numeric giving the standard deviation of individual effects.

`totalThetas` a list containing a `data.frame` for each level of the effect defined in the constraints. Rows are sampling iterations and columns comprised of individual estimates plus the common effect estimates.

`mcmcFull` an object of class `BFmcmc`, containing the MCMC samples from the posterior.

`designIndeces` a list giving the column indeces of the mcmcFull that were used to calculate the estimates of interest.

`observedEffects` a list containing a named vector for each level of the effect defined in the constraints. Values are the mean observed effect for each individual.

BFConstraint-class *S4 class for representing constraints*

Description

BFConstraint is a S4 class that represents the input and output of user-defined constraints in Bayesian Mixed Model analysis.

Slots

priorProbability a numeric giving the prior probability of all individual effects being as defined in the constraints.

posteriorProbability a numeric giving the posterior probability of all individual effects being as defined in the constraints.

bayesFactor a numeric giving the Bayes Factor in support of the defined constraints.

constraints a data frame containing the specified constraints.

cleanConstraints A data frame containing the specified constraints as they are presented by constraintBF.

calculateDifferences *Calculate differences between conditions specified in constraints*

Description

Calculates the differences between the conditions specified in the constraints of a BFBayesFactorConstraint object for each individual.

Usage

```
calculateDifferences(x, effect = c("estimate", "observed"))
```

Arguments

x	an object of class BFBayesFactorConstraint-class.
effect	the effect differences to be calculated. effect = "estimate" computes the effect differences from the model estimates; effect = "observed" computes the observed effect differences.

Value

calculateDifferences returns an object of class tbl_df, with columns for ID, type of effect, specified constraint, and estimates.

Examples

```
## Not run:
data(stroop)

resStroop <- constraintBF(rtS ~ ID*cond,
                          data = stroop,
                          whichRandom = "ID",
                          ID = "ID",
                          whichConstraint = c(cond = "2 > 1"),
                          rscaleEffects = c("ID" = 1, "cond" = 1/6, "ID:cond" = 1/10))

calculateDifferences(resStroop, effect = "estimate")
calculateDifferences(resStroop, effect = "observed")

## End(Not run)
```

`constraintBF` *Function to compute Bayes factors for ordinal constraints*

Description

This function uses Bayesian mixed models to estimate individual effect sizes and to test theoretical order constraints.

Usage

```
constraintBF(
  formula,
  data,
  whichRandom,
  ID,
  whichConstraint,
  rscaleEffects,
  iterationsPosterior = 10000,
  iterationsPrior = iterationsPosterior * 10,
  burnin = 1000,
  ...
)
```

Arguments

`formula`	a formula containing the full model.
`data`	a `data.frame` containing the data with all variables defined in the formula.
`whichRandom`	a character vector specifying which factors are random.
`ID`	a character vector of length one specifying which variable holds the subject ID.

whichConstraint
: a named character vector specifying the constraints placed on certain factors; see Details.

rscaleEffects
: a named vector of prior settings for individual factors. Values are scales, names are factor names; see Details.

iterationsPosterior
: the number of iterations to sample from the posterior of the full model.

iterationsPrior
: the number of iterations to sample from the prior of the full model.

burnin
: the number of initial iterations to discard from posterior sampling.

...
: further arguments to be passed to generalTestBF.

Details

This function provides a way of testing whether theoretical constraints on certain effects hold for all subjects. The backend is provided by the generalTestBF function from the BayesFactor-package. The input formula is the full model to be tested. It usually contains an interaction term between the subject ID and the effect for which constraints are tested (e.g. ID:condition). The ID variable is to be specified in ID and is usually a random factor to be specified in whichRandom.

Order constraints on effects should be specified in whichConstraint, as a named character vector. Each constraint in the vector can take 2 levels of the effect. They are of the form: "effect name" = "condition A" < "condition B". In order to impute more than 2 levels, the same effect name has to be entered with different conditions as the value. For instance, for testing whether conditions A < B < C, the input should be: "effect name" = "condition A" < "condition B", "effect name" = "condition B" < "condition C". At this point, constraints can only be tested for the same effect.

Priors have to be specified for all factors in whichConstraint, for ID, and for the interaction between the two. A Detailed description of the models, priors and methods is given in the documentation of anovaBF and more extensively in Rouder et al. (2012).

Value

An object of class BFBayesFactorConstraint-class.

References

Rouder, J. N., Morey, R. D., Speckman, P. L., Province, J. M., (2012) Default Bayes Factors for ANOVA Designs. Journal of Mathematical Psychology. 56. p. 356-374.

Examples

```
data(stroop)

resStroop <- constraintBF(rtS ~ ID*cond,
                          data = stroop,
                          whichRandom = "ID",
                          ID = "ID",
                          whichConstraint = c(cond = "2 > 1"),
                          rscaleEffects = c("ID" = 1, "cond" = 1/6, "ID:cond" = 1/10))
```

`ld5` *The dataset from Rouder et al. (2005) on the lexical task.*

Description

This is the cleaned dataset from Rouder et al. (2005) on the lexical task. Participants were presented with number stimuli ranging from 2 to 4 and from 6 to 8 and had to indicate whether the stimuli number is bigger or smaller than 5. The data cleaning that was performed was: removing two participants who gave up, excluding trials with too fast and too slow responses, excluding trials with wrong responses, excluding the first 20 trials of the first block, excluding the first trial in every block.

Usage

```
ld5
```

Format

A `data.frame` of 17031 rows and 9 columns

sub Factor with 52 levels giving the subject ID: 0-29, 35-42, 44-56, 58

block Numeric values giving the block: 1-5

trial Numeric values giving the number of the trial within a block

stim Numeric values giving the type of stimulus: 0 = 2, 1 = 3, 2 = 4, 3 = 6, 4 = 7, 5 = 8

resp Numeric values giving the participant's response: 1 = bigger, 2 = smaller

rt Numeric values giving the response times in milliseconds

error Numeric values indicating whether participant gave a wrong answer (all 0)

side Factor with 2 levels indicating whether stimulus is below (1) or above (2) 5

distance Factor with 3 levels indicating how far the stimulus number is away from 5

Source

Rouder, J. N., Lu, J., Speckman, P., Sun, D., & Jiang, Y. (2005). A hierarchical model for estimating response time distributions. Psychonomic Bulletin & Review, 12(2), 195-223., retrieved from https://raw.githubusercontent.com/PerceptionCognitionLab/data0/master/lexDec-dist5/ld5.all

plotEffects *Plot a BFBayesFactorConstraint object*

Description

Plots observed vs estimated individual effects for each level of the effect for which constraints are defined.

Usage

```
plotEffects(x, .raw = FALSE)
```

Arguments

x	an object of class BFBayesFactorConstraint-class.
.raw	if FALSE, outputs the plot. If TRUE returns the data.frame that is used for making the plot.

Value

A ggplot object if .raw = FALSE. A data.frame otherwise, with columns for ID, type of effect, specified constraint, estimates, and ordering of estimate from smallest to largest.

Examples

```
data(stroop)

resStroop <- constraintBF(rtS ~ ID*cond,
                          data = stroop,
                          whichRandom = "ID",
                          ID = "ID",
                          whichConstraint = c(cond = "2 > 1"),
                          rscaleEffects = c("ID" = 1, "cond" = 1/6, "ID:cond" = 1/10))

plotEffects(resStroop)
```

quid *Bayesian Mixed Models for Qualitative Individual Differences*

Description

Test whether equality and order constraints hold for all individuals simultaneously by comparing Bayesian mixed models through Bayes factors.

Author(s)

Maintainer: Lukas Klima <lukas.klima@live.at>

Authors:

- Julia Haaf <j.m.haaf@uva.nl> [degree supervisor, thesis advisor]

`stroop` *Stroop Dataset from Von Bastian et al. (2015)*

Description

The dataset from on Von Bastian et al. (2015) on the Stroop task.

Usage

```
stroop
```

Format

A `data.frame` of 11245 rows and 7 columns

ID Factor giving the participant ID: 1-121

congruency Factor giving one of the two conditions: congruent and incongruent

rtMS Numeric values of the response times in milliseconds

accuracy Numeric values giving the accuracy of the response, here only 1 = correct

cond Factor giving the condition: 1 = congruent, 2 = incongruent

trial Numeric values indicating the trial number of the participant

rtS Numeric values giving the response time in seconds

Details

In a Stroop task, average response times are compared across conditions. Participants get exposed to word stimuli and are instructed to indicate the font colour of the word. The word itself is also a colour word and is either **congruent** with the font colour or **incongruent**. For example, the stimulus word "red" in blue font would constitute the incongruent condition, whereas the word "red" in red font would constitute the congruent condition.

Source

Von Bastian, C. C., Souza, A. S., & Gade, M. (2016). No evidence for bilingual cognitive advantages: A test of four hypotheses. Journal of Experimental Psychology: General, 145(2), 246., retrieved from https://raw.githubusercontent.com/PerceptionCognitionLab/data0/master/contexteffects/FlankerStroopSimon/LEF_stroop.csv

Index