Package ‘BANOVA’

January 15, 2020

Type Package
Title Hierarchical Bayesian ANOVA Models
Version 1.1.6
Date 2020-01-12
Author Chen Dong, Michel Wedel
SystemRequirements JAGS-4.3.0, C++11
Maintainer Chen Dong <cdong@math.umd.edu>
Depends R (>= 3.2.3)
Imports rjags(>= 3-13), runjags (>= 1.2.1-0), coda (>= 0.16-1), rstan(>= 2.15.1), methods
Suggests knitr, rmarkdown
VignetteBuilder knitr

Description It covers several Bayesian Analysis of Variance (BANOVA) models used in analysis of experimental designs in which both within- and between- subjects factors are manipulated. They can be applied to data that are common in the behavioral and social sciences. The package includes: Hierarchical Bayes ANOVA models with normal response, t response, Binomial (Bernoulli) response, Poisson response, ordered multinomial response and multinomial response variables. All models accommodate unobserved heterogeneity by including a normal distribution of the parameters across individuals. Outputs of the package include tables of sums of squares, effect sizes and p-values, and tables of predictions, which are easily interpretable for behavioral and social researchers. The floodlight analysis and mediation analysis based on these models are also provided. BANOVA uses 'Stan' and 'JAGS' as the computational platform.

License GPL (>= 2)
NeedsCompilation yes
Repository CRAN
Date/Publication 2020-01-15 22:40:02 UTC

R topics documented:

BANOVA-package .......................... 2
BANOVA-package

BANOVA: Hierarchical Bayesian ANOVA Models

Description

This package includes several hierarchical Bayes Analysis of Variance models. These models are suited for the analysis of experimental designs in which both within- and between- subjects factors are manipulated, and account for a wide variety of distributions of the dependent variable. Floodlight analysis and mediation analysis based on these models are also provided. The package uses 'Stan' and 'JAGS' as the computational platform.

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>BANOVA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>1.1.6</td>
</tr>
<tr>
<td>Date</td>
<td>2020-01-12</td>
</tr>
<tr>
<td>License</td>
<td>GPL (&gt;= 2)</td>
</tr>
</tbody>
</table>
Model:
\[ E(y_i) = g^{-1}\left(\eta_i\right) \]
where \( \eta_i = \sum_{p=0}^{P} \sum_{j=1}^{J_p} \sum_{s=1}^{S} X_{i,j,p,s} \beta_{p,j,s} \), \( s_i \) is the subject id of data response \( i \). Missing values (NAs) of \( y_i \) are allowed. The within-subjects factors and their interactions are indexed by \( p(p = 1, 2, \ldots, P) \). Each index \( p \) represents a batch of \( J_p \) coefficients: \( \beta_{p,j,s} \), \( j = 1, \ldots, J_p; s = 1, \ldots, S \) indexes subjects. Note that if the subject-level covariate is continuous, \( J_p = 1 \), so that ANCOVA models are also accommodated (relaxing their "constant slope" assumption).

The population-level model allows for heterogeneity among subjects, because the subject-level coefficients \( \beta_{p,j,s} \) are assumed to follow a multivariate normal distribution. The between-subjects factors and their interactions are indexed by \( q, (q = 1, 2, \ldots, Q) \), \( q = 0 \) denotes the constant term. The population-level ANOVA can be written as:
\[ \beta_{p,j,s} = \sum_{q=0}^{Q} \theta_{p,q,j,k} + \delta_{p,j,s} \]

The population-level ANCOVA model can be expressed as a linear model with a design matrix \( Z \) that contains all between-subjects factors and their interactions and a constant term:
\[ \beta_{p,j,s} = \sum_{k=1}^{Q} Z_{s,k} \theta_{p,j,k} + \delta_{p,j,s} \]
where \( Z_{s,k} \) is an element of \( Z \), a \( S \times Q \) matrix of covariates. \( \theta_{p,j,k} \) is a hyperparameter which captures the effects of between-subjects factor \( q \) on the parameter \( \beta_{p,j,s} \) of within-subjects factor \( p \). The error \( \delta_{p,j,s} \) is assumed to be normal: \( \delta_{p,j,s} \sim N(0, \sigma_p^{-2}) \). Proper, but diffuse priors are assumed: \( \theta_{p,j,k} \sim N(0, \gamma) \), and \( \sigma_p^{-2} \sim Gamma(a, b) \), where \( \gamma, a, b \) are hyper-parameters. The default setting is \( \gamma = 10^{-4}, a = 1, b = 1 \).

Note that missing values of independent variables are currently not allowed in the package.

Author(s)
Chen Dong; Michel Wedel
Maintainer: Chen Dong <cdong@math.umd.edu>

References
BAnova

Function to print the table of effect sizes

Description

The analysis of variance is performed at level 1 (for the single level model) and level 2 equation of the Bayesian ANOVA see BANOVA-package. This makes it possible to capture the effects of level-1 or level-2 variables on the heterogeneity distribution of subjects, and compute sums of squares and effect sizes.

Usage

BAnova(x)

Arguments

x the object from BANOVAre.

Details

Measures of effect size in regression are measures of the degree of association between an effect (e.g., a main effect, an interaction, a linear contrast) and the dependent variable. They can be considered as the correlation between a categorical factor(effect) and the dependent variable. They are usually interpreted as the proportion of variance in the dependent variable that is attributable to each effect. In the package, partial Eta squared is calculated and displayed. It is defined as follows,

\[ \eta^2 = \frac{(SS_{\text{effect}})}{(SS_{\text{effect}} + SS_{\text{error}})} \]

Where: SS_effect= the sums of squares for the effect of interest
SS_error= the sums of squares for the error in the regression.

This equation is evaluated at each draw of the parameters, which allows for the calculation of not only the posterior mean, but also the credible interval of the effect size.

References


Examples

data(goalstudy)
res1 <- BANOVA.Normal(bid~1, ~progress*prodvar, goalstudy, goalstudy$id,
burnin = 1000, sample = 1000, thin = 2)
BAnova(res1)

library(rstan)
# or use BANOVA.run based on 'Stan'
res2 <- BANOVA.run(bid~1, ~progress*prodvar, model_name = 'Normal',
data = goalstudy, id = 'id', iter = 1000, chains = 2)
BAnova(res2)

BANOVA.Bernoulli

Estimation of BANOVA with a Bernoulli dependent variable

Description

BANOVA.Bernoulli implements a Bayesian ANOVA for binary dependent variable, using a logit link and a normal heterogeneity distribution.

Usage

BANOVA.Bernoulli(l1_formula = "NA", l2_formula = "NA", data,
id, l2_hyper = c(1, 1, 0.0001), burnin = 5000, sample = 2000, thin = 10,
adapt = 0, conv_speedup = F, jags = runjags.getOption('jagspath'))

## S3 method for class 'BANOVA.Bernoulli'
summary(object, ...)
## S3 method for class 'BANOVA.Bernoulli'
predict(object, newdata = NULL, ...)
## S3 method for class 'BANOVA.Bernoulli'
print(x, ...)

Arguments

  l1_formula    formula for level 1 e.g. 'Y~X1+X2'
  l2_formula    formula for level 2 e.g. '-Z1+Z2', response variable must not be included
  data          a data.frame in long format including all features in level 1 and level 2(covariates and categorical factors) and responses
  id            subject ID of each response unit
  l2_hyper      level 2 hyperparameters, c(a, b, γ), default c(1,1,0.0001)
  burnin        the number of burn in draws in the MCMC algorithm, default 5000
  sample        target samples in the MCMC algorithm after thinning, default 2000
thin  the number of samples in the MCMC algorithm that needs to be thinned, default 10
adapt  the number of adaptive iterations, default 0 (see run.jags)
conv_speedup  whether to speedup convergence, default F
jags  the system call or path for activating 'JAGS'. Default calls findjags() to attempt to locate 'JAGS' on your system
object  object of class BANOVA.Bern (returned by BANOVA.Bern)
ewdata  test data, either a matrix, vector or a data.frame. It must have the same format with the original data (the same number of features and the same data classes)
x  object of class BANOVA.Bern (returned by BANOVA.Bern)
...  additional arguments, currently ignored

Details

Level 1 model:
\[ y_i \sim \text{Binomial}(1, p_i), p_i = \logit^{-1}(\eta_i) \]
where \( \eta_i = \sum_{p=0}^{P} \sum_{j=1}^{J_p} X_{i,j}^{p} \beta_{j,s_i}^{p} \), \( s_i \) is the subject id of data record \( i \). see BANOVA-package

Value

BANOVA.Bernoulli returns an object of class "BANOVA.Bernoulli". The returned object is a list containing:

anova.table  table of effect sizes  BAnova
coef.tables   table of estimated coefficients
pvalue.table  table of p-values  table.pvalues
dMatrice      design matrices at level 1 and level 2
samples_l2_param  posterior samples of level 2 parameters
data          original data.frame
mf1           model.frame of level 1
mf2           model.frame of level 2
JAGSmodel     'JAGS' model

Examples

data(bernlogtime)

# model with the dependent variable : response
res <- BANOVA.Bernoulli(response~typical, ~blur + color, bernlogtime, bernlogtime$subject, burnin = 5000, sample = 2000, thin = 10)
summary(res)
# or use BANOVA.run
res0 <- BANOVA.run(response~typical, ~blur + color, data = bernlogtime, model_name = 'Bernoulli', id = 'subject', iter = 100, thin = 1, chains = 2)
summary(res0)
BANOVA.Binomial

Estimation of BANOVA with a Binomial dependent variable

Description

BANOVA.Binomial implements a Hierarchical Bayesian ANOVA for a binomial response variable using a logit link and a normal heterogeneity distribution.

Usage

BANOVA.Binomial(l1_formula = "NA", l2_formula = "NA", data, id, num_trials, l2_hyper = c(1, 1, 0.0001), burnin = 5000, sample = 2000, thin = 10, adapt = 0, conv_speedup = F, jags = runjags.getOption('jagpath'))

Arguments

- `l1_formula`: formula for level 1 e.g. 'Y~X1+X2'
- `l2_formula`: formula for level 2 e.g. '~Z1+Z2', response variable must not be included
- `data`: a data.frame in long format including all features in level 1 and level 2(covariates and categorical factors) and responses
- `id`: subject ID of each response unit
- `num_trials`: the number of trials of each observation (=1, if it is bernoulli), the type is forced to be 'integer'
- `l2_hyper`: level 2 hyperparameters, c(a, b, γ), default c(1,1,0.0001)
- `burnin`: the number of burn in draws in the MCMC algorithm, default 5000
- `sample`: target samples in the MCMC algorithm after thinning, default 2000
- `thin`: the number of samples in the MCMC algorithm that needs to be thinned, default 10
- `adapt`: the number of adaptive iterations, default 0 (see run.jags)
- `conv_speedup`: whether to speedup convergence, default F
- `jags`: the system call or path for activating 'JAGS'. Default calls findjags() to attempt to locate 'JAGS' on your system
- `object`: object of class BANOVA.Bin (returned by BANOVA.Bin)
- `newdata`: test data, either a matrix, vector or a data frame. It must have the same format with the original data (the same column number)
- `x`: object of class BANOVA.Bin (returned by BANOVA.Bin)
- `...`: additional arguments, currently ignored
Details

Level 1 model:
\[ y_i \sim \text{Binomial}(\text{ntrials}, p_i), \quad p_i = \logit^{-1}(\eta_i) \]
where ntrials is the binomial total for each record i, \( \eta_i = \sum_{p=0}^{J_p} \sum_{j=1}^{J_s} X_{i,j}^p \beta_{j,s} \), s is the subject id of response i. see BANOVA-package

Value

BANOVA.Binomial returns an object of class "BANOVA.Bin". The returned object is a list containing:

- `anova.table` table of effect sizes
- `coef.tables` table of estimated coefficients
- `pvalue.table` table of p-values
- `dMatrice` design matrices at level 1 and level 2
- `samples.l2.param` posterior samples of level 2 parameters
- `data` original data.frame
- `mf1` model.frame of level 1
- `mf2` model.frame of level 2
- `JAGSmodel` 'JAGS' model

Examples

data(colorad)

# mean center Blur for effect coding
colorad$blur <- colorad$blur - mean(colorad$blur)
res <- BANOVA.Binomial(y~typic, ~color*blur, colorad, colorad$id, as.integer(16),
  burnin = 5000, sample = 2000, thin = 10)
summary(res)

# or use BANOVA.run
res0 <- BANOVA.run(y~typic, ~color*blur, data = colorad, model_name = 'Binomial',
  id = 'id', num_trials = as.integer(16), iter = 100, thin = 1, chains = 2)
summary(res0)
table.predictions(res0)
# only in-model variables(except numeric variables) will be used
predict(res0, c(1, 0, 8, 2, 1, 0.03400759))
**BANOVA.build**  
*Build BANOVA models*

**Description**

BANOVA.build builds (compiles) BANOVA models.

**Usage**

BANOVA.build(BANOVA_model)

**Arguments**

- **BANOVA_model** an object of class "BANOVA.model"

**Value**

BANOVA.build returns an object of class "BANOVA.build". The returned object is a list containing:

- **stanmodel** the compiled 'Stan' model
- **model_name** the model name
- **single_level** if the model is a single level model

**Examples**

```r
model <- BANOVA.model('Poisson', single_level = FALSE)
Poisson_model <- BANOVA.build(model)
# visualize the model
cat(model$model_code)
# modify the model code and rebuild
# be careful to change any parameters
model$model_code <-"'
  data {
    int<lower=0> N;
    int<lower=0> J;
    int<lower=0> M;
    int<lower=0> K;
    matrix[N, J] X;
    matrix[M, K] Z;
    int<lower=0> id[N];
    int y[N];
  }

  parameters {
    matrix[J, M] beta1;
    matrix[K, J] beta2;
    vector<lower=0>[J] tau_beta1Sq;
  }
"
```
BANOVA.floodlight

Floodlight analysis based on BANOVA models

Description

BANOVA.floodlight conducts floodlight analysis based on various BANOVA models.

Usage

BANOVA.floodlight(sol, var_numeric, var_factor, flood_values = list())
## S3 method for class 'BANOVA.floodlight'
print(x, ...)
Details

A floodlight analysis (Spiller et al. 2013; Johnson and Neyman 1936) based on BANOVA models is conducted, which identifies regions of the numeric variable for which differences between the levels of the factor are significant. The endpoints of the 95% credible interval of the numeric variable provide the Johnson-Neyman points; for values outside of that interval there is 'strong' evidence that there is a difference between the levels of the factor.

Value

BANOVA.floodlight returns an object of class "BANOVA.floodlight". The returned object is a list containing:

- sol: table of the floodlight analysis including the 95% credible interval
- num_range: range of the numeric variable

References


Examples

data(condstudy)

library(rstan)
# use BANOVA.run
model <- BANOVA.model('Normal')
stanmodel <- BANOVA.build(model)
res <- BANOVA.run(att~cond+pict, ~type, fit = stanmodel, data = condstudy, id = 'id', iter = 500, thin = 1, chains = 2)
BANOVA.floodlight(res, var_factor = 'type', var_numeric = 'pict')

data(colorad)
colorad$num <- rnorm(nrow(colorad))
res1 <- BANOVA.run(y ~ typic*num, ~ color*blur, fit = stanmodel_bin, data = colorad, id = 'id', num_trials = as.integer(16))
# floodlight analysis at the +sd(standard deviation) level
BANOVA.floodlight(res1, var_factor = 'typic', var_numeric = 'blur', list(num = sd(colorad$num)))
Description

BANOVA.mediation conducts mediation/moderated mediation analysis based on various BANOVA models.

Usage

BANOVA.mediation(sol_1, sol_2, xvar, mediator, individual = F)
## S3 method for class 'BANOVA.mediation'
print(x, ...)

Arguments

sol_1       a BANOVA.* model based on an outcome variable, a causal variable, a mediator and possible moderators
sol_2       a BANOVA.Normal model for the mediator which includes the causal variable and moderators
xvar        the causal variable
mediator    the mediator variable
individual  whether to output individual level effects
x           a BANOVA.mediation object
...         additional arguments, currently ignored

Details

A mediation or moderated mediation analysis (Baron and Kenny 1986; Zao, Lynch and Chen 2010; Zhang, Wedel and Pieters 2008) based on BANOVA models is conducted, in which posterior distributions of the direct effect and indirect effect are calculated based on posterior samples. Means and 95% credible intervals are reported.

Value

BANOVA.mediation returns an object of class "BANOVA.mediation". The returned object is a list containing:

dir_effects tables of the direct effect
individual_direct the table of the direct effect at the individual level if individual = T and the causal variable is a within-subject variable
m1_effects tables of the effect of the mediator on the outcome
m2_effects tables of the effect of the causal variable on the mediator
indir_effects tables of the indirect effect
individual_indirect
  the table of the indirect effect at the individual level if individual = T and the
  mediator is a within-subject variable

xvar
  the name of the causal variable

mediator
  the name of the mediator

individual
  the value of the argument individual

References


Examples

data(condstudy)

library(rstan)
# use BANOVA.run based on 'Stan'
model <- BANOVA.model('Normal')
stanmodel <- BANOVA.build(model)
out2 <- BANOVA.run(att~cond+pict, ~type, fit = stanmodel, data = condstudy,
  id = 'id', iter = 500, thin = 1, chains = 2)
out3 <- BANOVA.run(pict~cond, ~type, fit = stanmodel, data = condstudy,
  id = 'id', iter = 500, thin = 1, chains = 2)
# (moderated) mediation
sol <- BANOVA.mediation(out2, out3, xvar='cond', mediator='pict')
print(sol)
print(sol$dir_effects)

BANOVA.model

Extract BANOVA models

Description

BANOVA.model extracts BANOVA models from the package.
Usage

BANOVA.model(model_name, single_level = F)

Arguments

model_name a character string in c('Normal', 'T', 'Bernoulli', 'Binomial', 'Poisson', 'ord-Multinomial', 'Multinomial')

single_level if the model is a single level model, default False

Value

BANOVA.model returns an object of class "BANOVA.model". The returned object is a list containing:

model_code the model code of the extracted model

model_name the model name

single_level if the model is a single level model

Examples

model <- BANOVA.model('Poisson', single_level = FALSE)
cat(model$model_code)

BANOVA.Multinomial Estimation of BANOVA with a Multinomial dependent variable

Description

BANOVA.Multinomial implements a Hierarchical Bayesian ANOVA for multinomial response variable using a logit link and a normal heterogeneity distribution.

Usage

BANOVA.Multinomial(l1_formula = "NA", l2_formula = "NA",
    dataX, dataZ, y, id, l2_hyper = c(1, 1, 0.0001), burnin = 5000, sample = 2000,
    thin = 10, adapt = 0, conv_speedup = F, jags = runjags.getOption('jagpath'))

## S3 method for class 'BANOVA.Multinomial'
summary(object, ...)

## S3 method for class 'BANOVA.Multinomial'
predict(object, Xsamples = NULL, Zsamples = NULL,...)

## S3 method for class 'BANOVA.Multinomial'
print(x, ...)
Arguments

l1_formula  formula for level 1 e.g. ‘~X1+X2’, response variable must not be included
l2_formula  formula for level 2 e.g. ‘~Z1+Z2’, response variable must not be included
dataX         a list of data frames (each corresponds to the choice set of each observation) that includes all covariates and factors
dataZ         a data frame (long format) that includes all level 2 covariates and factors
y             choice responses, 1, 2, 3...
id            subject id
l2_hyper      level 2 hyperparameters, c(a, b, γ), default c(1, 1, 0.0001)
burnin       the number of burn in draws in the MCMC algorithm, default 5000
sample        target samples in the MCMC algorithm after thinning, default 2000
thin          the number of samples in the MCMC algorithm that needs to be thinned, default 10
adapt         the number of adaptive iterations, default 0 (see run.jags)
conv_speedup  whether to speedup convergence, default F
jags           the system call or path for activating 'JAGS'. Default calls findjags() to attempt to locate 'JAGS' on your system
object        object of class BANOVA.Multinomial (returned by BANOVA.Multinomial)
Xsamples       new data samples in level one, must be a list (the same format with the training data), numeric variables must be mean centered.
Zsamples       new data samples in level two (the same format with the training data), numeric variables must be mean centered.
x             object of class BANOVA.Multinomial (returned by BANOVA.Multinomial)
...            additional arguments, currently ignored

Details

Level 1 model:
\[ P(y_i = \ell) = \frac{e^{\eta_{i\ell}}}{\sum_{\ell=1}^{L} e^{\eta_{i\ell}}} \]
where \( \eta_{i\ell} = \sum_{p=0}^{P} \sum_{j=1}^{J} X_{i,j}^{k,p} \beta_{j,s_i} \), \( s_i \) is the subject id of response \( i \), see BANOVA-package. \( X_{i,j}^{k,p} \) is the design matrix corresponding to each class \( \ell (\ell = 1, \ldots, L) \) of \( y_i \). The first level of the response is the base level, thus the intercept corresponding to this level will not be included.

Value

BANOVA.Multinomial returns an object of class "BANOVA.Multinomial". The returned object is a list containing:

anova.table  table of effect sizes Banova
coeff.tables  table of estimated coefficients
pvalue.table  table of p-values table.pvalues
dMatrice      design matrices at level 1 and level 2
samples_l2_param
posterior samples of level 2 parameters
dataX
original dataX
dataZ
original dataZ
mf1
model.frame of level 1
mf2
model.frame of level 2
n_categories
the number of categories of the response
JAGSmodel
'JAGS' model

Examples

# see 'choicedata'
data(choicedata)
# generate dataX(convert the within-subject variables to a list)
dataX <- list()
for (i in 1:nrow(choicedata)){
  logP <- as.numeric(log(choicedata[i,3:8]))
  # all numeric variables must be mean centered
  dataX[[i]] <- as.data.frame(logP) - mean(logP)
}
dataZ <- choicedata[,9:13]
res <- BANOVA.Multinomial(~ logP, ~ college, dataX, dataZ, 
  choicedata$choice, choicedata$hhid, burnin = 5000, sample = 2000, thin = 10)
# or use BANOVA.run based on 'Stan'
res <- BANOVA.run(~ logP, ~ college, dataX = dataX, dataZ = dataZ, 
  model_name = 'Multinomial', y_value = choicedata$choice, 
  id = 'hhid', iter = 100, thin = 1, chains = 2)

BANOVA.Normal
Estimation of BANOVA with a normally distributed dependent variable

Description

BANOVA.Normal implements a Hierarchical Bayesian ANOVA for linear models with normal response and a normal heterogeneity distribution.

Usage

BANOVA.Normal(l1_formula = "NA", l2_formula = "NA", data, 
id, l1_hyper = c(1, 1), l2_hyper = c(1, 1, 0.0001), burnin = 5000, 
sample = 2000, thin = 10, adapt = 0, conv_speedup = F, 
jags = runjagsgetOption('jagpath'))
## S3 method for class 'BANOVA.Normal'
summary(object, ...)
# S3 method for class 'BANOVA.Normal'
predict(object, newdata = NULL, ...)

## S3 method for class 'BANOVA.Normal'
print(x, ...)

Arguments

- **l1_formula**: formula for level 1 e.g. 'Y~X1+X2'
- **l2_formula**: formula for level 2 e.g. '~Z1+Z2', response variable must not be included, if missing, the single level model will be generated
- **data**: a data.frame in long format including all features in level 1 and level 2 (covariates and categorical factors) and responses
- **id**: subject ID of each response unit
- **l1_hyper**: level 1 hyperparameters, \( c(\alpha, \beta) \) for two-level models and \( c(\alpha, \beta, \sigma) \) for single level models, default \( c(1,1) \)
- **l2_hyper**: level 2 hyperparameters, \( c(a, b, \gamma) \), default \( c(1,1,0.0001) \)
- **burnin**: the number of burn in draws in the MCMC algorithm, default 5000
- **sample**: target samples in the MCMC algorithm after thinning, default 2000
- **thin**: the number of samples in the MCMC algorithm that needs to be thinned, default 10
- **adapt**: the number of adaptive iterations, default 0 (see run.jags)
- **conv.speedup**: whether to speedup convergence, default F
- **jags**: the system call or path for activating 'JAGS'. Default calls findjags() to attempt to locate 'JAGS' on your system
- **object**: object of class BANOVA.Normal (returned by BANOVA.Normal)
- **newdata**: test data, either a matrix, vector or a data frame. It must have the same format with the original data (the same column number)
- **x**: object of class BANOVA.Normal (returned by BANOVA.Normal)
- **...**: additional arguments, currently ignored

Details

Level 1 model:
\[ y_i \sim Normal(\eta_i, \sigma^{-2}) \]
where \( \eta_i = \sum_{p=0}^{P} \sum_{j=1}^{J_p} X_{i,j}^p \beta_{j,s_i}^p \), \( s_i \) is the subject id of response \( i \), \( \sigma^{-2} \sim Gamma(\alpha, \beta) \). see BANOVA-package

Value

BANOVA.Normal returns an object of class "BANOVA.Normal". The returned object is a list containing:

- **anova.table**: table of effect sizes BAnova
- **coef.tables**: table of estimated coefficients
Examples

# Use the ipadstudy data set
data(ipadstudy)
# mean center covariates
ipadstudy$age <- ipadstudy$age - mean(ipadstudy$age)
ipadstudy$owner <- ipadstudy$owner - mean(ipadstudy$owner)
res <- BANOVA.Normal(attitude~1, ~owner + age + gender + selfbrand*conspic, ipadstudy,
        id = 'id', burnin = 1000, sample = 1000, thin = 1)
summary(res)

# or use BANOVA.run based on 'Stan'
res <- BANOVA.run(attitude~1, ~owner + age + gender + selfbrand*conspic,
data = ipadstudy, model_name = 'Normal', id = 'id',
 iter = 100, thin = 1, chains = 2)

BANOVA.ordMultinomial  Estimation of BANOVA with a ordered Multinomial response variable

Description

BANOVA.ordMultinomial implements a Hierarchical Bayesian ANOVA for ordered multinomial responses, with a normal heterogeneity distribution.

Usage

BANOVA.ordMultinomial(l1_formula = "NA",
l2_formula = "NA", data, id, l1_hyper = c(0.0001, 100),
l2_hyper = c(1, 1, 0.0001, 100), burnin = 5000,
sample = 2000, thin = 10, adapt = 0, conv_speedup = F,
jags = runjags.getOption('jagspath'))
## S3 method for class 'BANOVA.ordMultinomial'
summary(object, ...)
## S3 method for class 'BANOVA.ordMultinomial'
predict(object, newdata = NULL, ...)
## S3 method for class 'BANOVA.ordMultinomial'
print(x, ...)
Arguments

- **l1_formula**: formula for level 1 e.g. 'Y~X1+X2'
- **l2_formula**: formula for level 2 e.g. '~Z1+Z2', response variable must not be included, if missing, the single level model will be generated
- **data**: a data frame
- **id**: subject ID of each response unit
- **l1_hyper**: level 1 hyperparameters for single level models, default c(0.0001,100)
- **l2_hyper**: level 2 hyperparameters, c(a, b, γ, d), default c(1,1,0.0001,100)
- **burnin**: the number of burn in draws in the MCMC algorithm, default 5000
- **sample**: target samples in the MCMC algorithm after thinning, default 2000
- **thin**: the number of samples in the MCMC algorithm that needs to be thinned, default 10
- **adapt**: the number of adaptive iterations, default 0 (see run.jags)
- **conv_speedup**: whether to speedup convergence, default F
- **jags**: the system call or path for activating 'JAGS'. Default calls findjags() to attempt to locate 'JAGS' on your system
- **object**: object of class BANOVA.ordMultinomial (returned by BANOVA.ordMultinomial)
- **newdata**: test data, either a matrix, vector or a data frame. It must have the same format with the original data (the same column number)
- **x**: object of class BANOVA.ordMultinomial (returned by BANOVA.ordMultinomial)
- **...**: additional arguments, currently ignored

Details

Level 1 model:

\[ y_i = 1, \text{ if } l_i < 0 \]
\[ y_i = 2, \text{ if } 0 < l_i < c_2 \]
... 
\[ y_i = \ell, \text{ if } c_{\ell-1} < l_i < \infty \]

\[ l_i = \eta_i + \epsilon_i \] where \( \epsilon_i \sim \text{logistic} (0,1) \), \( c_\ell, (\ell = 2,...,L - 1) \) are cut points, \( c_\ell \sim N(0, \bar{\sigma}_\ell^2) \), and \( \bar{\sigma}_\ell^2 \sim \text{Uniform}(0,d) \), with \( d \) a hyper-parameter.

\[ \eta_i = \sum_{p=0}^{P} \sum_{j=1}^{J_p} X_{i,j}^p \beta_{j,s_i}^p, s_i \text{ is the subject id of response } i. \] see BANOVA-package

Value

BANOVA.ordMultinomial returns an object of class "BANOVA.ordMultinomial". The returned object is a list containing:

- **anova.table**: table of effect sizes BAnova
- **coef.tables**: table of estimated coefficients
- **pvalue.table**: table of p-values table.pvalues
- **dMatrice**: design matrices at level 1 and level 2
samples_l2_param
posterior samples of level 2 parameters
samples_cutp_param
posterior samples of cutpoints
data
original data.frame
mf1
model.frame of level 1
mf2
model.frame of level 2
JAGSmodel
'JAGS' model

Examples

data(goalstudy)
res <- BANOVA.ordMultinomial (perceivedsim~1, ~progress*prodvar, goalstudy,
goalstudy$id, burnin = 5000, sample = 2000, thin = 10)
summary(res)
# or use BANOVA.run based on 'Stan'
res <- BANOVA.run(perceivedsim~1, ~progress*prodvar, data = goalstudy,
model_name = 'ordMultinomial', id = 'id', iter = 100, thin = 1, chains = 2)

BANOVA.Poisson
Estimation of BANOVA with Poisson dependent variables

Description

BANOVA.Poisson implements a Hierarchical Bayesian ANOVA for models with a count-data response variable and normal heterogeneity distribution.

Usage

BANOVA.Poisson(l1_formula = "NA", l2_formula = "NA",
data, id, l2_hyper = c(1, 1, 0.0001), burnin = 5000, sample = 2000, thin = 10,
adapt = 0, conv_speedup = F, jags = runjags.getOption('jagspath'))
## S3 method for class 'BANOVA.Poisson'
summary(object, ...)
## S3 method for class 'BANOVA.Poisson'
predict(object, newdata = NULL,...)
## S3 method for class 'BANOVA.Poisson'
print(x, ...)
BANOVA.Poisson

Arguments

11_formula  formula for level 1 e.g. ‘Y~X1+X2’
12_formula  formula for level 2 e.g. ‘~Z1+Z2’, response variable must not be included, if missing, the single level model will be generated
data         a data.frame in long format including all features in level 1 and level 2 (covariates and categorical factors) and responses
id           subject ID of each response unit
12_hyper     level 2 hyperparameters, c(a, b, γ), default c(1,1,0.0001)
burnin       the number of burn in draws in the MCMC algorithm, default 5000
sample       target samples in the MCMC algorithm after thinning, default 2000
thin         the number of samples in the MCMC algorithm that needs to be thinned, default 10
adapt        the number of adaptive iterations, default 0 (see run.jags)
cov_speedup  whether to speedup convergence, default F
jags          the system call or path for activating 'JAGS'. Default calls findjags() to attempt to locate 'JAGS' on your system
object       object of class BANOVA.Poisson (returned by BANOVA.Poisson)
newdata      test data, either a matrix, vector or a data frame. It must have the same format with the original data (the same column number)
x            object of class BANOVA.Poisson (returned by BANOVA.Poisson)
...           additional arguments, currently ignored

Details

Level 1 model:
y_i ∼ Poisson(λ_i), λ_i = exp(η_i + ϵ_i)
where η_i = ∑_{p=0}^{P} ∑_{j=1}^{J} X_{i,j} β_p j, s_i is the subject id of response i, see BANOVA-package. ϵ_i is a dispersion term.

Value

BANOVA.Poisson returns an object of class "BANOVA.Poisson". The returned object is a list containing:

anova.table  table of effect sizes Banova
coef.tables   table of estimated coefficients
pvalue.table  table of p-values table.pvalues
dMatrice      design matrices at level 1 and level 2
samples_l2_param posterior samples of level 2 parameters
samples_l2_sigma_param posterior samples of level 2 standard deviations
data          original data.frame
mf1  model.frame of level 1
mf2  model.frame of level 2
JAGSmodel 'JAGS' model

Examples

# use the bpndata dataset
data(bpndata)
# within-subjects model using the dependent variable : PIC_FIX
res1 <- BANOVA.Poisson(PIC_FIX ~ AD_ID + PIC_SIZE + PAGE_NUM + PAGE_POS, ~1, bpndata, bpndata$RESPONDENT_ID, burnin = 500, sample = 200, thin = 5)
summary(res1)

# use the goalstudy dataset
data(goalstudy)
goalstudy$bid <- as.integer(goalstudy$bid + 0.5)
res2 <- BANOVA.Poisson(bid~1, ~progress*prodvar, goalstudy, goalstudy$id, burnin = 5000, sample = 2000, thin = 10)
summary(res2)

# or use the BANOVA.run based on 'Stan'
res3 <- BANOVA.run(bid~1, ~progress*prodvar, data = goalstudy, model_name = 'Poisson', id = 'id', iter = 100, thin = 1, chains = 2)

BANOVA.run

Estimation of BANOVA models

Description

BANOVA.run implements Hierarchical Bayesian ANOVA models using 'Stan'

Usage

BANOVA.run(l1_formula = "NA", l2_formula = "NA", fit = NULL, model_name = 'NA', dataX = NULL, dataZ = NULL, data = NULL, y_value = NULL, id, iter = 2000, num_trials = 1, contrast = NULL, ...)
## S3 method for class 'BANOVA'
summary(object, ...)
## S3 method for class 'BANOVA'
predict(object, newdata = NULL, Xsamples = NULL, Zsamples = NULL, ...)
## S3 method for class 'BANOVA'
print(x, ...)
Arguments

- `l1_formula`: formula for level 1 e.g. `Y~X1+X2`
- `l2_formula`: formula for level 2 e.g. `~Z1+Z2`, response variable must not be included. If NULL, the single level model is used
- `fit`: a fitted BANOVA models, an object of class "BANOVA.build", default NULL which needs compilation
- `model_name`: a character string in c('Normal', 'T', 'Bernoulli', 'Binomial', 'Poisson', 'ord-Multinomial', 'Multinomial')
- `dataX`: a list of data frames(each corresponds to the choice set of each observation) that includes all covariates and factors, for the Multinomial model only, default NULL
- `dataZ`: a data frame(long format) that includes all level 2 covariates and factors, for the Multinomial model only, default NULL
- `data`: a data.frame in long format including all features in level 1 and level 2(covariates and categorical factors) and responses, default NULL
- `id`: subject ID (string) of each response unit
- `y_value`: choice responses, 1,2,3..., for the Multinomial model only, default NULL
- `iter`: target samples in the 'Stan' algorithm after thinning, default 2000
- `num_trials`: the number of trials of each observation(=1, if it is bernoulli), the type is forced to be 'integer', for the Binomial model only, default 0
- `contrast`: a list of contrasts for planned comparisons, default: effect coding (NULL value)
- `object`: an object of class BANOVA (returned by BANOVA.run)
- `x`: an object of class BANOVA (returned by BANOVA.run)
- `newdata`: test data, either a matrix, vector or a data frame. It must have the same format with the original data (the same column number)
- `Xsamples`: a list of sample data frames(each corresponds to the choice set of each observation) that includes all covariates and factors, for the Multinomial model only, default NULL
- `Zsamples`: a data frame(long format) that includes all level 2 covariates and factors, for the Multinomial model only, default NULL
- `...`: additional arguments, for BANOVA.run, it can include standard 'Stan' arguments, e.g. warmup, thin, chains, etc., see sampling for more details, for other functions, ignored currently

Value

BANOVA.run returns an object of class "BANOVA". The returned object is a list containing:

- `anova.table`: table of effect sizes BAnova
- `coef.tables`: table of estimated coefficients
- `pvalue.table`: table of p-values table.pvalues
- `dMatrice`: design matrices at level 1 and level 2
samples_l1_param
   posterior samples of level 1 parameters
samples_l2_param
   posterior samples of level 2 parameters
samples_l2_sigma_param
   posterior samples of level 2 standard deviations
samples_cutp_param
   posterior samples of cutpoints
data
   original data.frame
mf1
   model.frame of level 1
mf2
   model.frame of level 2
model_code
   'Stan' code
single_level
   if this is a single level model
stan_fit
   fitted samples
model_name
   the name of the model
contrast
   contrasts for planned comparisons
new_id
   id values coded in 1,2,3,...
old_id
   original id values

Examples

# Use the ipadstudy data set
data(ipadstudy)

library(rstan)
# build the BANOVA model first so that it can be reused
model <- BANOVA.model('Normal', single_level = FALSE)
banova_model <- BANOVA.build(model)
res_1 <- BANOVA.run(attitude~1, ~owner + age + gender + selfbrand*conspic,
fit = banova_model, data = ipadstudy, id = 'id', iter = 2000,
thin = 5, chains = 2)
summary(res_1)
res_2 <- BANOVA.run(attitude~1, -selfbrand*conspic,
fit = banova_model, data = ipadstudy, id = 'id', iter = 2000,
thin = 5, chains = 2)

# or call the function directly without specifying the fit argument
# but it needs compilation
res_1 <- BANOVA.run(attitude~1, ~owner + age + gender + selfbrand*conspic,
model_name = 'Normal', data = ipadstudy, id = 'id', iter = 2000,
thin = 5, chains = 2)
**BANOVA.T**

*Estimation of BANOVA with T-distribution of the dependent variable*

---

**Description**

BANOVA.T implements a Hierarchical Bayesian ANOVA for linear models with T-distributed response.

**Usage**

```r
BANOVA.T(l1_formula = "NA", l2_formula = "NA", data, id, l1_hyper = c(1, 1, 1),
   l2_hyper = c(1, 1, 0.0001), burnin = 5000, sample = 2000, thin = 10,
   adapt = 0, conv_speedup = F, jags = runjags.getOption('jagpath'))
```

```r
## S3 method for class 'BANOVA.T'
summary(object, ...)
## S3 method for class 'BANOVA.T'
predict(object, newdata = NULL, ...)
## S3 method for class 'BANOVA.T'
print(x, ...)
```

**Arguments**

- `l1_formula` formula for level 1 e.g. 'Y~X1+X2'
- `l2_formula` formula for level 2 e.g. '~Z1+Z2', response variable must not be included, if missing, the single level model will be generated
- `data` a data.frame in long format including all features in level 1 and level 2 (covariates and categorical factors) and responses
- `id` subject ID of each response unit
- `l1_hyper` level 1 hyperparameters, c(α, β, λ) for two-level models and c(α, β, λ, σ_y) for single level models, default c(1,1,1)
- `l2_hyper` level 2 hyperparameters, c(a, b, γ), default c(1,1,0.0001)
- `burnin` the number of burn in draws in the MCMC algorithm, default 5000
- `sample` target samples in the MCMC algorithm after thinning, default 2000
- `thin` the number of samples in the MCMC algorithm that needs to be thinned, default 10
- `adapt` the number of adaptive iterations, default 0 (see run.jags)
- `conv_speedup` whether to speedup convergence, default F
- `jags` the system call or path for activating 'JAGS'. Default calls findjags() to attempt to locate 'JAGS' on your system
- `object` object of class BANOVA.T (returned by BANOVA.T)
- `newdata` test data, either a matrix, vector or a data frame. It must have the same format with the original data (the same column number)
- `x` object of class BANOVA.T (returned by BANOVA.T)
- `...` additional arguments, currently ignored
Details

Level 1 model:
\[ y_i \sim t(\nu, \eta_i, \sigma^{-2}) \]
where \( \eta_i = \sum_{p=0}^{P} \sum_{j=1}^{J_p} X_{i,j}^p \beta_p^j, s_i \) is the subject id of response \( i \), see BANOVA-package. The hyper parameters: \( \nu \) is the degree of freedom, \( \nu \sim \text{Poisson}(\lambda) \) and \( \sigma \) is the scale parameter, \( \sigma^{-2} \sim \text{Gamma}(\alpha, \beta) \).

Value

BANOVA.T returns an object of class "BANOVA.T". The returned object is a list containing:

- anova.table: table of effect sizes
- coef.tables: table of estimated coefficients
- pvalue.table: table of p-values
- dMatrice: design matrices at level 1 and level 2
- samples_l2_param: posterior samples of level 2 parameters
- data: original data.frame
- mf1: model.frame of level 1
- mf2: model.frame of level 2
- JAGSmodel: 'JAGS' model

Examples

# Use the ipadstudy data set
data(ipadstudy)
res <- BANOVA.T(attitude~1, ~owner + age + gender + selfbrand*conspic, ipadstudy, ipadstudy$id, burnin = 5000, sample = 2000, thin = 10)
summary(res)

# or use BANOVA.run based on 'Stan'
res19 <- BANOVA.run(attitude~1, ~owner + age + gender + selfbrand*conspic, data = ipadstudy, model_name = 'T', id = 'id', iter = 100, thin = 1, chains = 2)

bernllogtime

Data for analysis of effects of typicality, blur and color on gist perception of ads

Description

Data from a mixed design experiment, where respondents were exposed to 32 ads, for 100 millisec. The ads were either typical or atypical (typical: 1 or 2). Respondents were exposed to ads that were either in full color or black-and-white (color: 1 or 2), and at different levels of blur (1=normal, 5 = very high blur). These are between-subjects factors. The dependent variables are the response 0/1, and the response time. Typicality is a within-subjects variable.
Usage

data(bernlogtime)

Format

This R object contains within-subject variable: $typical$ is a factor with 2 levels "0" (typical ads) and "1" (atypical ads); between-subjects variables: $blur$ is a factor with two levels (1=normal,5 = very high blur). $color$ denotes a factor with 2 levels "1" (full color) and "2" (grayscale). $subject$ is the ID of subjects. $response$ denotes if the ad is correctly identified. $logtime$ is the response time.

$bernlogtime$: 'data.frame': 3072 obs. of 6 variables:
  ...$ subject : int 5 5 5 5 5 5 5 5 5 5 ...
  ...$ typical : Factor w/ 2 levels "1","2": 1 2 1 1 1 2 2 2 2 1 ...
  ...$ blur : Factor w/ 2 levels "1","5": 1 1 1 1 1 1 1 1 1 1 ...
  ...$ color : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 2 ...
  ...$ response: int 1 1 1 1 1 1 1 1 1 1 ...
  ...$ logtime : num 0.977 1.73 1.784 1 1.149 ...

References


Examples

data(bernlogtime)

# model using the dependent variable : log of the response time(logtime)
res1 <- BANOVA.Normal(logtime~typical, ~blur + color, bernlogtime,
  bernlogtime$subject, burnin = 1000, sample = 1000, thin = 1)
summary(res1)
table.predictions(res1)

# model using the dependent variable : response
res2 <- BANOVA.Bernoulli(response~typical, ~blur + color, bernlogtime,
  bernlogtime$subject, burnin = 1000, sample = 1000, thin = 1)
summary(res2)
table.predictions(res2)
Description

Data were collected in an experimental study in which 88 participants freely paged through a magazine at home or in a waiting room. While flipping through pages at their own pace, participants’ eye-movements were recorded with infra-red corneal reflection eye-tracking methodology. In a subsequent memory task, participants were asked to identify the target brand in the ad as soon as possible by touching the correct brand name on the screen. Accuracy (accurate=1, inaccurate =0) of brand memory and response time were recorded for each ad and participant.

Usage

data(bpndata)

Format

This R object contains 3080 observations in the data (35 ads x 88 participants). The goal is to examine the effects of several ad design variables on both eye movements and memory. The variables include:

1. RESPONDENT_ID: ID number of a respondent;
2. AD_ID: ID number of an ad;
3. PAGE_NUM: page number in the magazine where an ad appears (1,2,3,...);
4. PAGE_POS: the right-side vs. left-side position on a page, 1 = right, 0 = left;
5. PIC_FIX: fixation count of the pictorial element (0, 1, 2, 3,...);
6. PIC_SIZE: surface size of the pictorial element, in inches2;
7. RECALL_ACCU: whether a respondent accurately recalls the brand name, 1= yes, 0 = no;
8. RECALL_TIME: the time it takes a respondent to answer the brand recall question, in seconds.

References


Examples

data(bpndata)
# within-subjects model using the dependent variable : PIC_FIX
library(rstan)
choicedata <- BANOVA.model('Poisson')
stanmodel <- BANOVA.build(stanmodel)
res0 <- BANOVA.run(PIC_FIX ~ PIC_SIZE + PAGE_NUM + PAGE_POS, ~1, fit = stanmodel, data = bpndata, id = 'RESPONDENT_ID', iter = 200, thin = 1, chains = 2)
res0
# or
res1 <- BANOVA.Poisson(PIC_FIX ~ PIC_SIZE + PAGE_NUM + PAGE_POS, ~1, bpndata, bpndata$RESPONDENT_ID, burnin = 1000, sample = 1000, thin = 1)
res1

# within-subjects model using the dependent variable: RECALL_ACCU
model_bern <- BANOVA.model('Bernoulli')
stanmodel_bern <- BANOVA.build(model_bern)
res2 <- BANOVA.run(RECALL_ACCU ~ RECALL_TIME + PAGE_NUM + PAGE_POS, ~1, fit = stanmodel_bern, data = bpndata, id = 'RESPONDENT_ID', iter = 200, thin = 1, chains = 2)
res2
# or
res3 <- BANOVA.Bernoulli(RECALL_ACCU ~ RECALL_TIME + PAGE_NUM + PAGE_POS, ~1, bpndata, bpndata$RESPONDENT_ID, burnin = 1000, sample = 1000, thin = 1)
res3

choicedata

Household Panel Data on Margarine Purchases

Description

Panel data on purchases of margarine by 204 households. Demographic variables are included.

Usage

data(choicedata)

Format

This is an R object that contains within-subjects variables and between-subjects variables:

Within-subject variables:

...$ choicePrice: 'data.frame': 1500 obs. of 13 variables:
  ...$ hhid : int 2100016 2100016 2100016 2100016 ...
  ...$ choice : int 1 1 1 1 1 1 1 1 1 1

...$ PPK\_Stk : num 0.66 0.63 0.29 0.62 0.5 0.58 0.29 ...
  ...$ PBB\_Stk : num 0.67 0.67 0.5 0.61 0.58 0.45 0.51 ...
  ...$ PFl\_Stk : num 1.09 1.09 1.09 1.09 1.09 1.09 1.09 ...
  ...$ PHse\_Stk: num 0.57 0.57 0.57 0.57 0.57 0.57 0.57...
choice is a multinomial indicator of one of the 6 brands (in order listed under format). All prices are in $.

Source


Examples

data(choicedata)
# generate dataX(convert the within-subjects variables to a list)
dataX <- list()
for (i in 1:nrow(choicedata)){
  logP <- as.numeric(log(choicedata[i,3:8]))
  # Note: Before the model initialization, all numeric variables(covariates)
  # must be mean centered
  dataX[[i]] <- as.data.frame(logP) - mean(logP)
}
dataZ <- choicedata[,9:13]
res <- BANOVA.Multinomial(~ logP, ~ college, dataX, dataZ, choicedata$choice, choicedata$hhid, burnin = 1000, sample = 1000, thin = 1)
summary(res)
predict(res, dataX[1:4], dataZ[1:4])
Description

Data from an experiment in which one hundred and sixteen subjects (53 men; mean age 23, ranging from 21 to 28) were randomly assigned to one condition of a 5 (blur: normal, low, medium, high, very high) x 2 (color: full color, grayscale) between-participants, x 2 (image: typical ads, atypical ads) within-participants, mixed design. Participants were exposed to 40 images, 32 full-page ads and 8 editorial pages. There were 8 ads per product category, with 4 typical and 4 atypical ones, the categories being car, financial services, food, and skincare. Subjects were asked to identify each image being flashed for 100msec. as being an ad or not. The total number of correct ad identifications, for typical and atypical ads, are used as a dependent variable.

Usage

data(colorad)

Format

This R object contains within-subject variable `typic` which is a factor with 2 levels "0" (typical ads) and "1" (atypical ads); between-subjects variables: `blur` which is a numerical variable denotes 5 different levels of blur (which must be mean centered), `blurfac` is a categorical data corresponding to the levels of `blur`, `color` which is a factor with 2 levels "0" (full color) and "1" (grayscale), `id` is the ID of subjects. `y` is the number of correct identifications of the 16 ads of each subject for each level of `typic`.

> data(colorad)

References


Examples

data(colorad)
library(rstan)
# Build the model
model_bin <- BANOVA.model('Binomial')
stanmodel_bin <- BANOVA.build(model_bin)
out0 = BANOVA.run(y ~ typic, ~ color+blurfac, fit = stanmodel_bin,
data = colorad, id = 'id', num_trials = as.integer(16),
iters = 10000, thin = 10, chains = 1)
summary(out0)
# planned comparison
out0_contra = BANOVA.run(y ~ typic, ~ color+blurfac, fit = stanmodel_bin,
data = colorad, id = 'id', num_trials = as.integer(16),
iters = 10000, thin = 10, chains = 1,contrast = list(typic = c(-1,1),
blurfac = c(1,1,0,-1,-1)))
summary(out0_contra)
# floodlight analysis
set.seed(200)
out0_f = BANOVA.run(y ~ typic, ~ color+blur, fit = stanmodel_bin,
data = colorad, id = 'id', num_trials = as.integer(16),
iters = 10000, thin = 10, chains = 1)
BANOVA.floodlight(out0_f, var_factor = 'typic', var_numeric = 'blur')

# Data for gist perception of advertising, study 2

Description

Data from an experiment in which One hundred and forty eight subjects (71 men; age ranging from 21 to 28) were randomly assigned to one condition of a 2 (blur: normal, very high) x 2 (color: full color, grayscale, inverted) between-participants design. Participants were exposed to 25 ads for five brands in each of five categories. Ads were selected to be typical for the category, using the same procedure as in colorad. The product categories used were cars, financial services, food, skincare and fragrance. Images were flashed for 100 msec. and subjects were asked to identify whether the image was an ad or not, and if they identified it correctly as an ad, they were asked to indicate which category (out of five) was advertised. The total number of correct ad identifications and category identifications are used as dependent variables.

Usage

data(colorad2)

Format

This R object contains between-subjects variables: \$B is a factor corresponding to the levels of blur (normal = 0, very high = 1), \$C1 and \$C2 are dummy variables denote ‘grayscale’ and ‘inverted’ levels of color. \$C is the original factor denote the color with 3 levels. \$ID is the ID of subjects. \$Y1 is the number of correct identifications of the 25 ads of each subject. \$Y2 is the number of correct identifications of the category, given the number of correct ad identifications.

\$ colorad2: ‘data.frame’: 148 obs. of 7 variables:
  ... \$ ID : int 1 2 3 4 5 6 7 8 9 10 ...
Data for the study of how brand attitudes were influenced by showing brands together with pleasant pictures.
Description

The study investigated how brand attitudes were influenced by showing brands together with pleasant pictures. Attitude change via conditioning can result from either a direct transfer of affect from the picture to the brand, or from an indirect association of the brand and the picture in memory. In Sweldens’ et al. (2010) experiment 1, indirect conditioning was implemented by presenting a brand repeatedly with the same picture, direct conditioning by presenting it simultaneously with different pictures. The pictures used were either neutral or positive. This study involved a mixed design, with a within-subject factor (cond = neutral, positive), and a between-subject factor (type = indirect, direct), as well as a within-subject mediator. Although the original mediation hypotheses are more intricate, here the mediation of the conditioning effect is investigated by measurements of attitudes towards the pictures that were shown with the brands (pict).

Usage

data(condstudy)

Format

This R object contains a between-subjects variable: type, which denotes a between-subject moderator. It has two levels, "indirect" and "direct". In the "indirect" condition the brands were shown with the same images, in the indirect condition the brands were shown with different images; Within-subject variables: cond, a within-subject factor with 2 levels: "pos", and "xneu", which indicates whether each brand was shown with a neutral (xneu) or positive (pos) emotional image. pict, a within-subject mediator variable measuring the valence (positive/negative) of the emotional image the respondent remembers the brand to have been shown with. att, a dependent variable which denotes the ratings of attitudes toward brands.

\$ id: 'data.frame': 888 obs. of 5 variables:
  ...\$ att: num 2.94 2.44 3.44 1.67 1.67 ...
  ...\$ cond: Factor w/ 2 levels 'pos','xneu': 1 1 1 2 2 2 2 2 2 2 ...
  ...\$ pict: int 6 7 6 2 4 5 3 2 5 ...

References


Examples

data(condstudy)

library(rstan)
model <- BANOVA.model("Normal")
stanmodel <- BANOVA.build(model)
out2 <- BANOVA.run(att~cond+pict, ~type, fit = stanmodel, data = condstudy)
conv.diag

conv.diag(out2)
summary(out2)
table.predictions(out2)
BANOVA.floodlight(out2, var_factor = 'type', var_numeric = 'pict')
cat(out2$model_code)

out3 <- BANOVA.run(pict~cond, ~type, fit = stanmodel, data = condstudy,
   id = 'id', iter = 500, thin = 1, chains = 2)
conv.diag(out3)
summary(out3)
BANOVA.mediation(out2, out3, xvar='cond', mediator='pict')

---

**conv.diag**

*Function to display the convergence diagnostics*

**Description**

The Geweke diagnostic and the Heidelberg and Welch diagnostic are reported. These two convergence diagnostics are calculated based on only a single MCMC chain. Both diagnostics require a single chain and may be applied with any MCMC method. The functions `geweke.diag`, `heidel.diag` in `coda` package is used to compute this diagnostic.

Geweke’s convergence diagnostic is calculated by taking the difference between the means from the first \( n_A \) iterations and the last \( n_B \) iterations. If the ratios \( n_A/n \) and \( n_B/n \) are fixed and \( n_A + n_B < n \), then by the central limit theorem, the distribution of this diagnostic approaches a standard normal as \( n \) tends to infinity. In our package, \( n_A = .2 \times n \) and \( n_B = .5 \times n \).

The Heidelberg and Welch diagnostic is based on a test statistic to accept or reject the null hypothesis that the Markov chain is from a stationary distribution. The present package reports the stationary test. The convergence test uses the Cramer-von Mises statistic to test for stationary. The test is successively applied on the chain. If the null hypothesis is rejected, the first 10% of the iterations are discarded and the stationarity test repeated. If the stationary test fails again, an additional 10% of the iterations are discarded and the test repeated again. The process continues until 50% of the iterations have been discarded and the test still rejects. In our package, \( \epsilon = 0.1, pvalue = 0.05 \) are used as parameters of the function `heidel.diag`.

**Usage**

`conv.diag(x)`

**Arguments**

x  
the object from BANOVA.*

**Value**

`conv.diag` returns a list of two diagnostics:

- `sol_geweke`  
The Geweke diagnostic
- `sol_heidel`  
The Heidelberg and Welch diagnostic
References


Examples

data(goalstudy)
library(rstan)
res1 <- BANOVA.run(bid~1, ~progress*prodvar, model_name = "Normal", data = goalstudy, id = 'id', iter = 100, thin = 1)
conv.diag(res1)
# might need pairs() to confirm the convergence

goalstudy

Data for the study of the impact of the variety among means on motivation to pursue a goal

Description

The study investigated how the perceived variety (high vs. low) among products, as means to a subjects’ goal, affects their motivation to pursue that goal. The hypothesis was that only when progress toward a goal is low, product variety increases motivation to pursue the goal. In the study, one hundred and five subjects were randomly assigned to conditions in a 2 (goal progress: low vs. high) by 2 (variety among means: low vs. high) between-subjects design. The final goal was a “fitness goal”, and the products used were protein bars; variety was manipulated by asking subjects to think about how the products were similar (low) or different (high); goal progress was primed by asking subjects questions regarding the frequency of their recent workouts on low (0,1,...,5 or more) versus high (5 or less, 6,7,..., 10) frequency scales. Subjects were asked questions regarding the similarity of protein bars, and the bid they were willing to make for the bars, used as dependent variables in the study.

Usage

data(goalstudy)
Format

This R object contains between-subjects variables: progress, which denotes the progress toward a goal (1: low, 2: high); prodvar, which denotes the amount of variety within the means to goal attainment (1: low, 2: high); perceivedsim, which is a seven-point scale dependent variable measuring the perceived similarity of the set of products (1 = not at all similar, 7 = very similar); and bid which denotes the amount that subjects would be willing to pay for the products.

\$ goalstudy: 'data.frame': 105 obs. of 5 variables:
  ...\$ id: int 1 2 3 4 5 6 7 8 9 10 ...
  ...\$ perceivedsim: int 5 7 2 5 5 5 4 5 7 ...
  ...\$ progress: Factor w/ 2 levels "1","2": 1 1 1 1 2 1 2 1 2 1 ...
  ...\$ prodvar: Factor w/ 2 levels "1","2": 2 1 2 1 1 1 2 1 1 ...
  ...\$ bid: num 5 0 1 15 3 10 5 4.5 3 0.75 ...

References


Examples

data(goalstudy)
res1 <- BANOVA.Normal(bid~1, ~progress*prodvar, goalstudy, goalstudy$id,
                      burnin = 1000, sample = 1000, thin = 1)

library(rstan)
# use BANOVA.run based on 'Stan'
res1 <- BANOVA.run(bid~1, ~progress*prodvar, model_name = "Normal",
data = goalstudy, id = 'id', iter = 1000, thin = 1, chains = 2)
conv.diag(res1)
# using pairs, it shows that the chains are not converged
# the level 1 and level 2 variances are not identified since the data
# has only one observation per subject, it is better to use single level models
pairs(res1, pars = c("beta1[1,1]", "beta2[1,1]", "tau_ySq", "tau_beta1Sq"))

# single level model
res1 <- BANOVA.run(bid~progress*prodvar, model_name = "Normal",
data = goalstudy, id = 'id', iter = 1000, thin = 1, chains = 2)
BAnova(res1)
table.pvalues(res1)
trace.plot(res1)
table.predictions(res1)

res1_t <- BANOVA.T(bid~1, ~progress*prodvar, goalstudy, goalstudy$id,
burnin = 1000, sample = 1000, thin = 1)
res2 <- BANOVA.ordMultinomial (perceivedsim~1, ~progress*prodvar, goalstudy,
goalstudy$id, burnin = 1000, sample = 1000, thin = 1)
summary(res2)
table.predictions(res2)
# print 'JAGS' code
Data for the study of relation between Conspicuous, Brand Usage, Self-Brand Connection and attitudes toward the brand

Description

The study is a between-subjects experiment which has factor (conspicuousness: low vs. high) and one measured variable (self-brand connection). The goal is to show that conspicuous brand use negatively affects attitudes toward the user and the brand only for observers low in self-brand connection. One hundred fifty-four participants were exposed to a video manipulating conspicuous brand usage. Participants completed the study by answering several questions which are used to measure the dependent (attitude) and independent (self-brand connection) variables in the model.

Usage

data(ipadstudy)

Format

This R object contains between-subjects variables: $owner is an indicator variable. If the subject owns iPad or iPhone, then owner = 1. It is equal to 0 otherwise. $age denotes the age of subjects. $gender denotes the gender of subjects. gender = 1 if the subject is a female, 0 otherwise. $conspic is an indicator variable related to conspicuousness. conspic = 1 if conspicuousness is high. $self-brand denotes the self-brand connection for Apple. $id is the id of subjects. $attitude denotes the attitudes towards the brand which is the continuous dependent variable. $apple_dl is a seven-point scale variable which denotes the attitudes (dislike = 1,..., like = 7)

$ ipadstudy: 'data.frame': 154 obs. of 9 variables:
  ..$ id : int 1 2 3 4 5 6 7 8 9 10 ...
  ..$ attitude : num 3.533 5.67 5.33 6 ...
  ..$ owner : num 0 0 0 1 1 0 1 0 1 0 ...
  ..$ age : int 19 33 25 41 38 37 46 41 55 ...
  ..$ gender : num 0 0 1 0 1 1 0 1 1 1 ...
  ..$ conspic : num 0 1 0 1 1 0 0 1 0 1 ...
  ..$ selfbrand : num -2.304 1.696 -0.161 -0.447 0.267 ...
  ..$ apple_dl : int 3 6 5 6 4 7 7 5 5 ...

References

Examples

```r
library(rstan)
data(ipadstudy)
res_1 <- BANOVA.run(attitude~1, ~owner + age + gender + selfbrand*conspic, model_name = 'Normal', data = ipadstudy, id = 'id', iter = 1000, thin = 1, chains = 2)
pairs(res_1, pars = c("beta2[1,1]","beta2[2,1]"))
```

### Description

A `pairs` method that is customized for MCMC output.

### Usage

```r
## S3 method for class 'BANOVA'
pairs(x, ...)
```

### Arguments

- `x`: an object of class "BANOVA"
- `...`: Further arguments to be passed to `pairs.stanfit`

### Details

For a detailed description see `pairs.stanfit`
table.predictions  
*Function to print the table of means*

**Description**

Output of this function is a table of means for the categorical predictors (and their interactions) at either within- or between-subjects level. Statistics of interest such as credible intervals and standard deviations of the means are also computed. Means of numeric variables and their interactions will not be computed.

**Usage**

```r
table.predictions(x)
```

**Arguments**

- `x` the object from BANOVA.*

**Examples**

```r
data(goalstudy)
res <- BANOVA.Normal(bid~1, ~progress*prodvar, goalstudy, goalstudy$id,
  burnin = 1000, sample = 1000, thin = 1)

library(rstan)
# or use BANOVA.run based on 'Stan'
res <- BANOVA.run(bid~progress*prodvar, model_name = "Normal",
  data = goalstudy, id = 'id', iter = 1000, thin = 1, chains = 2)

table.predictions(res)
```

---

table.pvalues  
*Function to print the table of p-values*

**Description**

Computes the Bayesian p-values for the test concerning all coefficients/parameters:

For $p = 1, \ldots, P$

- $H_0: \theta_{j,k}^p = 0$
- $H_1: \theta_{j,k}^p \neq 0$

The two-sided P-value for the sample outcome is obtained by first finding the one sided P-value, $\min(P(\theta_{j,k}^p < 0), P(\theta_{j,k}^p > 0))$ which can be estimated from posterior samples. For example,
$P(\theta_{j,k}^p > 0) = \frac{n_+}{n}$, where $n_+$ is the number of posterior samples that are greater than 0, $n$ is the target sample size. The two sided P-value is $P_0(\theta_{j,k}^p) = 2 \cdot \min(P(\theta_{j,k}^p < 0), P(\theta_{j,k}^p > 0))$.

If there are $\theta_{j,k_1}^p, \theta_{j,k_2}^p, ..., \theta_{j,k_J}^p$ representing J levels of a multi-level variable, we use a single P-value to represent the significance of all levels. The two alternatives are:

$H_0 : \theta_{j,k_1}^p = \theta_{j,k_2}^p = \cdots = \theta_{j,k_J}^p = 0$

$H_1 : \text{some } \theta_{j,k_j}^p \neq 0$

Let $\theta_{j,k_{min}}^p$ and $\theta_{j,k_{max}}^p$ denote the coefficients with the smallest and largest posterior mean. Then the overall P-value is defined as

$\min(P_0(\theta_{j,k_{min}}^p), P_0(\theta_{j,k_{max}}^p))$.

Usage

`table.pvalues(x)`

Arguments

- `x` the object from BANOVA.*

Source

It borrows the idea of Sheffe F-test for multiple testing: the F-stat for testing the contrast with maximal difference from zero. Thank Dr. P. Lenk of the University of Michigan for this suggestion.

Examples

```r
data(goalstudy)
res1 <- BANOVA.Normal(bid~1, ~progress*prodvar, goalstudy, goalstudy$id, 
burnin = 1000, sample = 1000, thin = 2)

library(rstan)
# or use BANOVA.run
res1 <- BANOVA.run(bid~1, ~progress*prodvar, model_name = "Normal", 
data = goalstudy, id = 'id', iter = 1000, thin = 1, chains = 2)

table.pvalues(res1)
```

trace.plot

Function to plot the trace of parameters

Description

Function to plot the trace of all coefficients/parameters. The plots can be saved as a pdf file.

Usage

`trace.plot(x, save = FALSE)`
Arguments

x the object from BANOVA.*

save whether to save the trace plot as a pdf file, the default is FALSE

Examples

data(goalstudy)
res1 <- BANOVA.Normal(bid~1, ~progress*prodvar, goalstudy, goalstudy$id,
burnin = 1000, sample = 1000, thin = 2)

library(rstan)
# or use BANOVA.run
res1 <- BANOVA.run(bid~1, ~progress*prodvar, model_name = "Normal",
data = goalstudy, id = 'id', iter = 1000, thin = 1, chains = 2)

trace.plot(res1)
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>BANOVA (BANOVA-package), 2</td>
</tr>
<tr>
<td>BAnova, 4, 6, 8, 15, 17, 19, 21, 23, 26</td>
</tr>
<tr>
<td>BANOVA-package, 2</td>
</tr>
<tr>
<td>BANOVA.Bernoulli, 5</td>
</tr>
<tr>
<td>BANOVA.Binomial, 7</td>
</tr>
<tr>
<td>BANOVA.build, 9</td>
</tr>
<tr>
<td>BANOVA.floodlight, 10</td>
</tr>
<tr>
<td>BANOVA.mediation, 12</td>
</tr>
<tr>
<td>BANOVA.model, 13</td>
</tr>
<tr>
<td>BANOVA.Multinomial, 14</td>
</tr>
<tr>
<td>BANOVA.Normal, 16</td>
</tr>
<tr>
<td>BANOVA.ordMultinomial, 18</td>
</tr>
<tr>
<td>BANOVA.Poisson, 20</td>
</tr>
<tr>
<td>BANOVA.run, 22</td>
</tr>
<tr>
<td>BANOVA.T, 25</td>
</tr>
<tr>
<td>bernlogtime, 26</td>
</tr>
<tr>
<td>bpndata, 27</td>
</tr>
<tr>
<td>chosendata, 29</td>
</tr>
<tr>
<td>colorad, 31, 32</td>
</tr>
<tr>
<td>colorad2, 32</td>
</tr>
<tr>
<td>condstudy, 33</td>
</tr>
<tr>
<td>conv.diag, 35</td>
</tr>
<tr>
<td>geweke.diag, 35</td>
</tr>
<tr>
<td>goalstudy, 36</td>
</tr>
<tr>
<td>heidel.diag, 35</td>
</tr>
<tr>
<td>ipadstudy, 38</td>
</tr>
<tr>
<td>pairs, 39</td>
</tr>
<tr>
<td>pairs.BANOVA, 39</td>
</tr>
<tr>
<td>pairs.stanfit, 39</td>
</tr>
<tr>
<td>predict.BANOVA (BANOVA.run), 22</td>
</tr>
<tr>
<td>predict.BANOVA.Bernoulli (BANOVA.Bernoulli), 5</td>
</tr>
<tr>
<td>predict.BANOVA.Binomial (BANOVA.Binomial), 7</td>
</tr>
<tr>
<td>predict.BANOVA.Multinomial (BANOVA.Multinomial), 14</td>
</tr>
<tr>
<td>predict.BANOVA.Normal (BANOVA.Normal), 16</td>
</tr>
<tr>
<td>predict.BANOVA.ordMultinomial (BANOVA.ordMultinomial), 18</td>
</tr>
<tr>
<td>predict.BANOVA.Poisson (BANOVA.Poisson), 20</td>
</tr>
<tr>
<td>predict.BANOVA.T (BANOVA.T), 25</td>
</tr>
<tr>
<td>print.BANOVA (BANOVA.run), 22</td>
</tr>
<tr>
<td>print.BANOVA.Bernoulli (BANOVA.Bernoulli), 5</td>
</tr>
<tr>
<td>print.BANOVA.Binomial (BANOVA.Binomial), 7</td>
</tr>
<tr>
<td>print.BANOVA.floodlight (BANOVA.floodlight), 10</td>
</tr>
<tr>
<td>print.BANOVA.mediation (BANOVA.mediation), 12</td>
</tr>
<tr>
<td>print.BANOVA.Multinomial (BANOVA.Multinomial), 14</td>
</tr>
<tr>
<td>print.BANOVA.Normal (BANOVA.Normal), 16</td>
</tr>
<tr>
<td>print.BANOVA.ordMultinomial (BANOVA.ordMultinomial), 18</td>
</tr>
<tr>
<td>print.BANOVA.Poisson (BANOVA.Poisson), 20</td>
</tr>
<tr>
<td>print.BANOVA.T (BANOVA.T), 25</td>
</tr>
<tr>
<td>run.jags, 6, 7, 15, 17, 19, 21, 25</td>
</tr>
<tr>
<td>sampling, 23</td>
</tr>
<tr>
<td>summary.BANOVA (BANOVA.run), 22</td>
</tr>
<tr>
<td>summary.BANOVA.Bernoulli (BANOVA.Bernoulli), 5</td>
</tr>
<tr>
<td>summary.BANOVA.Binomial (BANOVA.Binomial), 7</td>
</tr>
<tr>
<td>summary.BANOVA.Multinomial (BANOVA.Multinomial), 14</td>
</tr>
<tr>
<td>summary.BANOVA.Normal (BANOVA.Normal), 16</td>
</tr>
<tr>
<td>summary.BANOVA.ordMultinomial (BANOVA.ordMultinomial), 18</td>
</tr>
</tbody>
</table>
summary.BANOVA.Poisson (BANOVA.Poisson), 20
summary.BANOVA.T (BANOVA.T), 25

table.predictions, 40
table.pvalues, 6, 8, 15, 18, 19, 21, 23, 26, 40
trace.plot, 41