Package ‘varian’

July 22, 2025

Type Package

Version 0.2.2

Title Variability Analysis in R

Description Uses a Bayesian model to
estimate the variability in a repeated
measure outcome and use that as an outcome or a predictor
in a second stage model.

Date 2016-2-28

Author Joshua F. Wiley [aut, cre],
Elkhart Group Limited [cph]

Maintainer Joshua F. Wiley `<josh@elkhartgroup.com>`

URL `https://github.com/ElkhartGroup/varian`

BugReports `https://github.com/ElkhartGroup/varian/issues`

Depends R (>= 3.1.1), rstan (>= 2.7.0), ggplot2

Imports stats, MASS, Formula, grid, gridExtra

Suggests testthat

LazyLoad yes

License MIT + file LICENSE

NeedsCompilation no

Repository CRAN

Date/Publication 2016-02-28 12:57:15

Contents

empirical_pvalue 2
gamma_params 3
parallel_stan 3
param_summary 5
pval_smartformat 5
res_gamma 6

simulate_gvm 7
stan_inits 8
Variability_Measures 9
varian 10
vmp_plot 12
vm_diagnostics 13
vm_stan 14

Index **16**

`empirical_pvalue` *Calculates an empirical p-value based on the data*

Description

This function takes a vector of statistics and calculates the empirical p-value, that is, how many fall on the other side of zero. It calculates a two-tailed p-value.

Usage

```
empirical_pvalue(x, na.rm = TRUE)
```

Arguments

`x`	a data vector to operate on
`na.rm`	Logical whether to remove NA values. Defaults to `TRUE`

Value

a named vector with the number of values falling at or below zero, above zero, and the empirical p-value.

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
empirical_pvalue(rnorm(100))
```

`gamma_params` *Estimate the parameters for a Gamma distribution*

Description

This is a simple function to estimate what the parameters for a Gamma distribution would be from a data vector. It is used internally to generate start values.

Usage

```
gamma_params(x)
```

Arguments

`x`	a data vector to operate on

Value

a list of the shape (alpha) and rate (beta) parameters and the mean and variance

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

`parallel_stan` *Wrapper for the stan function to parallelize chains*

Description

This funcntion takes Stan model code, compiles the Stan model, and then runs multiple chains in parallel.

Usage

```
parallel_stan(model_code, standata, totaliter, warmup, thin = 1, chains, cl,
  cores, seeds, modelfit, verbose = FALSE, pars = NA, sample_file = NA,
  diagnostic_file = NA, init = "random", ...)
```

Arguments

`model_code`	A character string of Stan code
`standata`	A data list suitable for Stan for the model given
`totaliter`	The total number of iterations for inference. Note that the total number of iterations is automatically distributed across chains.
`warmup`	How many warmup iterations should be used? Note that every chain will use the same number of warmups and these will be *added on top of the total iterations* for each chain.
`thin`	The thin used, default to 1 indicating that all samples be saved.
`chains`	The number of independent chains to run.
`cl`	(optional) The name of a cluster to use to run the chains. If not specified, the function will make a new cluster.
`cores`	(optional) If the `cl` argument is not used, this specifies the number of cores to make on the new cluster. If both `cl` and `cores` are missing, defaults to the minimum of the number of chains specified or the number of cores available on the machine.
`seeds`	(optional) A vector of random seeds the same length as the number of independent chains being run, to make results replicable. If missing, random seeds will be generated and stored for reference in the output.
`modelfit`	(optional) A compiled Stan model, if available, saves compiling `model_code`.
`verbose`	A logical whether to print verbose output (defaults to `FALSE`)
`pars`	Parameter names from Stan to store
`sample_file`	The sample file for Stan
`diagnostic_file`	The diagnostic file for Stan
`init`	A character string (“random”) or a named list of starting values.
`...`	Additional arguments, not currently used.

Value

a named list with three elements, the `results`, compiled Stan `model`, and the random `seeds`

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
# Make me!
```

`param_summary` *Calculates summaries for a parameter*

Description

This function takes a vector of statistics and calculates several summaries: mean, median, 95 the empirical p-value, that is, how many fall on the other side of zero.

Usage

```
param_summary(x, digits = 2, pretty = FALSE, ..., na.rm = TRUE)
```

Arguments

`x`	a data vector to operate on
`digits`	Number of digits to round to for printing
`pretty`	Logical value whether prettified values should be returned. Defaults to `FALSE`.
`na.rm`	Logical whether to remove NA values. Defaults to `TRUE`
`...`	Additional arguments passed to `pval_smartformat` to control p-value printing.

Value

.

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
param_summary(rnorm(100))
param_summary(rnorm(100), pretty = TRUE)
```

`pval_smartformat` *nice formatting for p-values*

Description

nice formatting for p-values

Usage

```
pval_smartformat(p, d = 3, sd = 5)
```

Arguments

`p`	a numeric pvalue
`d`	the digits less than which should be displayed as less than
`sd`	scientific digits for round

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
varian:::pval_smartformat(c(1, .15346, .085463, .05673, .04837, .015353462,
  .0089, .00164, .0006589, .0000000053326), 3, 5)
```

`res_gamma` *Estimates the parameters of a Gamma distribution from SDs*

Description

This function calcualtes the parameters of a Gamma distribution from the residuals from an individuals' own mean. That is, the distribution of (standard) deviations from individuals' own mean are calculated and then an estimate of the parameters of a Gamma distribution are calculated.

Usage

```
res_gamma(x, ID)
```

Arguments

`x`	A data vector to operate on
`ID`	an ID variable of the same length as `x`

Value

a list of the shape (alpha) and rate (beta) parameters and the mean and variance

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
set.seed(1234)
y <- rgamma(100, 3, 2)
x <- rnorm(100 * 10, mean = 0, sd = rep(y, each = 10))
ID <- rep(1:100, each = 10)
res_gamma(x, ID)
```

`simulate_gvm` *Simulate a Gamma Variability Model*

Description

This function facilitates simulation of a Gamma Variability Model and allows the number of units and repeated measures to be varied as well as the degree of variability.

Usage

```
simulate_gvm(n, k, mu, mu.sigma, sigma.shape, sigma.rate, seed = 5346)
```

Arguments

`n`	The number of repeated measures on each unit
`k`	The number of units
`mu`	The grand mean of the variable
`mu.sigma`	The standard deviation of the random mean of the variable
`sigma.shape`	the shape (alpha) parameter of the Gamma distribution controlling the residual variability
`sigma.rate`	the rate (beta) parameter of the Gamma distribution controlling the residual variability
`seed`	the random seed, used to make simulations reproductible. Defaults to 5346 (arbitrarily).

Value

a list of the data, IDs, and the parameters used for the simulation

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
raw.sim <- simulate_gvm(12, 140, 0, 1, 4, .1, 94367)
sim.data <- with(raw.sim, {
  set.seed(265393)
  x2 <- MASS::mvrnorm(k, c(0, 0), matrix(c(1, .3, .3, 1), 2))
  y2 <- rnorm(k, cbind(Int = 1, x2) %*% matrix(c(3, .5, .7)) + sigma, sd = 3)
  data.frame(
    y = Data$y,
    y2 = y2[Data$ID2],
    x1 = x2[Data$ID2, 1],
    x2 = x2[Data$ID2, 2],
    ID = Data$ID2)
})
```

`stan_inits` *Calculate Initial Values for Stan VM Model*

Description

Internal function used to get rough starting values for a variability model in Stan. Uses inidivudal standard deviations, means, and linear regressions.

Usage

```
stan_inits(stan.data, design = c("V -> Y", "V -> M -> Y", "V", "X -> V",
  "X -> V -> Y", "X -> M -> V"), useU, ...)
```

Arguments

`stan.data`	A list containing the data to be passed to Stan
`design`	A character string indicating the type of model to be run. One of “V -> Y” for variability predicting an outcome, “V -> M -> Y” for mediation of variability on an outcome, “V” to take posterior samples of individual variability estimates alone.
`useU`	whether to include the random intercepts
`...`	Additional arguments (not currently used)

Value

A named list containing the initial values for Stan.

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
# make me!
```

`Variability_Measures` *Variability Measures*

Description

Variability Measures

`by_id` - Internal function to allow a simple statistic (e.g., SD) to be calculated individually by an ID variable and returned either as per ID (i.e., wide form) or for every observation of an ID (i.e., long form).

`sd_id` - Calculates the standard deviation of observations by `ID`.

`rmssd` - Calculates the root mean square of successive differences (RMSSD). Note that missing values are removed.

`rmssd_id` - Calculates the RMSSD by ID.

`rolling_diff` - Calculates the average rolling difference of the data. Within each window, the difference between the maximum and minimum value is computed and these are averaged across all windows. The equation is:

$$\frac{\sum_{t=1}^{N-k} max(x_t, \ldots, x_{t+k}) - min(x_t, \ldots, x_{t+k})}{N-k}$$

`rolling_diff_id` - Calculates the average rolling difference by ID

Usage

```
by_id(x, ID, fun, long = TRUE, ...)

sd_id(x, ID, long = TRUE)

rmssd(x)

rmssd_id(x, ID, long = TRUE)

rolling_diff(x, window = 4)

rolling_diff_id(x, ID, long = TRUE, window = 4)
```

Arguments

`x`	A data vector to operate on. Should be a numeric or integer vector, or coercible to such (e.g., logical).
`ID`	an ID variable indicating how to split up the `x` vector. Should be the same length as `x`.
`fun`	The function to calculate by ID
`long`	A logical indicating whether to return results in “long” form (the default) or wide (if `FALSE`).

`window`	An integer indicating the size of the rolling window. Must be at least the length of `x`.
`...`	Additional arguments passed on to `fun`

Value

`by_id` - A vector the same length as `x` if `long=TRUE`, or the length of unique `IDs` if `long=FALSE`.

`sd_id` - A vector of the standard deviations by ID

`rmssd` - The RMSSD for the data.

`rmssd_id` - A vector of the RMSSDs by ID

`rolling_diff` - The average of the rolling differences between maximum and minimum.

`rolling_diff_id` - A vector of the average rolling differences by ID

Note

These are a set of functions designed to calculate various measures of variability either on a single data vector, or calculate them by an ID.

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
sd_id(mtcars$mpg, mtcars$cyl, long=TRUE)
sd_id(mtcars$mpg, mtcars$cyl, long=FALSE)
rmssd(1:4)
rmssd(c(1, 3, 2, 4))
rmssd_id(mtcars$mpg, mtcars$cyl)
rmssd_id(mtcars$mpg, mtcars$cyl, long=FALSE)
rolling_diff(1:7, window = 4)
rolling_diff(c(1, 4, 3, 4, 5))
rolling_diff_id(mtcars$mpg, mtcars$cyl, window = 3)
```

`varian` *Variablity Analysis using a Bayesian Variability Model (VM)*

Description

This function uses a linear mixed effects model that assumes the level 1 residual variance varies by Level 2 units. That is rather than assuming a homogenous residual variance, it assumes the residual standard deviations come from a Gamma distribution. In the first stage of this model, each Level 2's residual standard deviation is estimated, and in the second stage, these standard deviations are used to predict another Level 2 outcome. The interface uses an intuitive formula interface, but the underlying model is implemented in Stan, with minimally informative priors for all parameters.

The Variability Analysis in R Package

Usage

```
varian(y.formula, v.formula, m.formula, data, design = c("V -> Y",
  "V -> M -> Y", "V", "X -> V", "X -> V -> Y", "X -> M -> V"), useU = TRUE,
  totaliter = 2000, warmup = 1000, chains = 1, inits = NULL, modelfit,
  opts = list(SD_Tol = 0.01, pars = NULL), ...)
```

Arguments

`y.formula`	A formula describing a model for the outcome. At present, this must be a continuous, normally distributed variable.
`v.formula`	A formula describing a model for the variability. Note this must end with `| ID`, where `ID` is the name of the ID variable in the dataset. At present, this must be a continuous, normally distributed variable.
`m.formula`	An optional formula decribing a model for a mediatior variable. At present, this must be a continuous normally distributed variable.
`data`	A long data frame containing an both the Level 2 and Level 1 outcomes, as well as all covariates and an ID variable.
`design`	A character string indicating the type of model to be run. One of “V -> Y” for variability predicting an outcome, “V -> M -> Y” for mediation of variability on an outcome, “V” to take posterior samples of individual variability estimates alone.
`useU`	A logical value whether the latent intercept estimated in Stage 1 should also be used as a predictor. Defaults to `TRUE`. Note if there is a mediator as well as main outcome, the latent intercepts will be used as a predictor for both.
`totaliter`	The total number of iterations to be used (not including the warmup iterations), these are distributed equally across multiple independent chains.
`warmup`	The number of warmup iterations. Each independent chain has the same number of warmup iterations, before it starts the iterations that will be used for inference.
`chains`	The number of independent chains to run (default to 1).
`inits`	Initial values passed on to `stan`. If `NULL`, the default, initial values are estimated means, standard deviations, and coefficients from a single level linear regression.
`modelfit`	A compiled Stan model (e.g., from a previous run).
`opts`	A list giving options. Currently only `SD_Tol` which controls the tolerance for how small a variables standard deviation may be without stopping estimation (this ensures that duplicate variables, or variables without any variability are included as predictors).
`...`	Additional arguments passed to `stan`.

Value

A named list containing the model `results`, the `model`, the `variable.names`, the `data`, the random `seeds`, and the initial function `.call`.

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
## Not run:
  sim.data <- with(simulate_gvm(4, 60, 0, 1, 3, 2, 94367), {
    set.seed(265393)
    x2 <- MASS::mvrnorm(k, c(0, 0), matrix(c(1, .3, .3, 1), 2))
    y2 <- rnorm(k, cbind(Int = 1, x2) %*% matrix(c(3, .5, .7)) + sigma, sd = 3)
    data.frame(
      y = Data$y,
      y2 = y2[Data$ID2],
      x1 = x2[Data$ID2, 1],
      x2 = x2[Data$ID2, 2],
      ID = Data$ID2)
  })
  m <- varian(y2 ~ x1 + x2, y ~ 1 | ID, data = sim.data, design = "V -> Y",
    totaliter = 10000, warmup = 1500, thin = 10, chains = 4, verbose=TRUE)

  # check diagnostics
  vm_diagnostics(m)

  sim.data2 <- with(simulate_gvm(21, 250, 0, 1, 3, 2, 94367), {
    set.seed(265393)
    x2 <- MASS::mvrnorm(k, c(0, 0), matrix(c(1, .3, .3, 1), 2))
    y2 <- rnorm(k, cbind(Int = 1, x2) %*% matrix(c(3, .5, .7)) + sigma, sd = 3)
    data.frame(
      y = Data$y,
      y2 = y2[Data$ID2],
      x1 = x2[Data$ID2, 1],
      x2 = x2[Data$ID2, 2],
      ID = Data$ID2)
  })
  # warning: may take several minutes
  m2 <- varian(y2 ~ x1 + x2, y ~ 1 | ID, data = sim.data2, design = "V -> Y",
    totaliter = 10000, warmup = 1500, thin = 10, chains = 4, verbose=TRUE)
  # check diagnostics
  vm_diagnostics(m2)

## End(Not run)
```

`vmp_plot` *Plot the posterior distributions of the focal parameters from a VM model*

Description

This function plots the univariate and bivariate (if applicable) distributions of the focal (alpha) parameters from a Variability Model where the variability is used as a predictor in a second-stage model. The latent variability estimates are referred to as “Sigma” and, if used, the latent intercepts are referred to as “U”.

Usage

```
vmp_plot(alpha, useU = TRUE, plot = TRUE, digits = 3, ...)
```

Arguments

`alpha`	Results from running `varian` and `extracting` the results.
`useU`	Logical indicating whether to plot the latent intercepts (defaults to `TRUE`). Must set to `FALSE` if not available.
`plot`	Logical whether to plot the results or just return the grob for the plots. Defaults to `TRUE`.
`digits`	Integer indicating how many digits should be used for displaying p-values
`...`	Additional arguments (not currently used)

Value

A list containing the `Combined` and the `Individual` plot objects.

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
# Using made up data because the real models take a long time to run
set.seed(1234) # make reproducible
vmp_plot(matrix(rnorm(1000), ncol = 2))
```

`vm_diagnostics` *Plot diagnostics from a VM model*

Description

This function plots a variety of diagnostics from a Variability Model. These include a histogram of the Rhat values (so-called percent scale reduction factors). An Rhat value of 1 indicates that no reduction in the variability of the estimates is possible from running the chain longer. Values below 1.10 or 1.05 are typically considered indicative of convergence, with higher values indicating the model did not converge and should be changed or run longer. A histogram of the effective sample size indicates for every parameter estimated how many effective posterior samples are available for inference. Low values may indicate high autocorrelation in the samples and may be a sign of failure to converge. The maximum possible will be the total iterations available. Histograms of the posterior medians for the latent variability and intercept estimates are also shown.

Usage

```
vm_diagnostics(object, plot = TRUE, ...)
```

Arguments

`object`	Results from running `varian`.
`plot`	Logical whether to plot the results or just return the grob for the plots. Defaults to `TRUE`.
`...`	Additional arguments not currently used

Value

A graphical object

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
# Make Me!
```

`vm_stan` *Create a Stan class VM object*

Description

Internal function to create and compile a Stan model.

Usage

```
vm_stan(design = c("V -> Y", "V -> M -> Y", "V", "X -> V", "X -> V -> Y",
  "X -> M -> V"), useU = TRUE, ...)
```

Arguments

`design`	A character string indicating the type of model to be run. One of "V -> Y" for variability predicting an outcome, "V -> M -> Y" for mediation of variability on an outcome, "V" to take posterior samples of individual variability estimates alone.
`useU`	A logical value whether the latent intercept estimated in Stage 1 should also be used as a predictor. Defaults to `TRUE`.
`...`	Additional arguments passed to `stan_model`.

Value

A compiled Stan model.

Author(s)

Joshua F. Wiley <josh@elkhartgroup.com>

Examples

```
# Make Me!
## Not run:
  test1 <- vm_stan("V -> Y", useU=TRUE)
  test2 <- vm_stan("V -> Y", useU=FALSE)
  test3 <- vm_stan("V -> M -> Y", useU=TRUE)
  test4 <- vm_stan("V -> M -> Y", useU=FALSE)
  test5 <- vm_stan("V")

## End(Not run)
```

Index

∗ **hplot**
    vm_diagnostics, 13
    vmp_plot, 12
∗ **models**
    stan_inits, 8
    varian, 10
    vm_stan, 14
∗ **utilities**
    empirical_pvalue, 2
    gamma_params, 3
    parallel_stan, 3
    param_summary, 5
    pval_smartformat, 5
    res_gamma, 6
    simulate_gvm, 7
    Variability_Measures, 9

by_id *(Variability_Measures)*, 9

empirical_pvalue, 2

gamma_params, 3

parallel_stan, 3
param_summary, 5
pval_smartformat, 5

res_gamma, 6
rmssd *(Variability_Measures)*, 9
rmssd_id *(Variability_Measures)*, 9
rolling_diff *(Variability_Measures)*, 9
rolling_diff_id *(Variability_Measures)*, 9

sd_id *(Variability_Measures)*, 9
simulate_gvm, 7
stan_inits, 8

Variability_Measures, 9
varian, 10
varian-package *(varian)*, 10
vm_diagnostics, 13
vm_stan, 14
vmp_plot, 12