Package 'INLAvaan'

January 28, 2026

Type Package

Title Approximate Bayesian Latent Variable Analysis

Version 0.2.3

Description Implements approximate Bayesian inference for Structural Equation Models (SEM) using a custom adaptation of the Integrated Nested Laplace Approximation as described in Rue et al. (2009) <doi:10.1111/j.1467-9868.2008.00700.x>. Provides a computationally efficient alternative to Markov Chain Monte Carlo (MCMC) for Bayesian estimation, allowing users to fit latent variable models using the 'lavaan' syntax.

License GPL (>= 3)

URL https://inlavaan.haziqj.ml/, https://github.com/haziqj/INLAvaan

BugReports https://github.com/haziqj/INLAvaan/issues

Imports blavaan, cli, cowplot, dplyr, ggplot2, lavaan, methods, modeest, mvtnorm, numDeriv, qrng, scales, statmod, stats, tidyr, ucminf, utils

Suggests future, knitr, lme4, psychotools, quarto, rmarkdown, rstan, sn, spacefillr, testthat (>= 3.0.0)

VignetteBuilder quarto

Config/Needs/website rmarkdown

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Author Haziq Jamil [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-3298-1010>), Håvard Rue [ctb] (ORCID: <https://orcid.org/0000-0002-0222-1881>, Statistical and computational methodology), Alvin Bong [ctb] (Initial site build)

Maintainer Haziq Jamil <haziq.jamil@gmail.com>

Repository CRAN

Date/Publication 2026-01-28 22:50:02 UTC

Contents

acfa . . . 2
agrowth . . . 4
asem . . . 7
as_fun_string . . . 10
compare . . . 10
dbeta_box . . . 12
dsnorm . . . 12
fitMeasures,INLAvaan-method . . . 13
fit_skew_normal . . . 14
fit_skew_normal_samp . . . 15
inlavaan . . . 16
INLAvaan-class . . . 19
is_same_function . . . 20
qsnorm_fast . . . 21
standardisedsolution . . . 22

Index **25**

`acfa` *Fit an Approximate Bayesian Confirmatory Factor Analysis Model*

Description

Fit an Approximate Bayesian Confirmatory Factor Analysis Model

Usage

```
acfa(
  model,
  data,
  dp = blavaan::dpriors(),
  marginal_method = c("skewnorm", "asymgaus", "marggaus", "sampling"),
  nsamp = 500,
  test = "standard",
  marginal_correction = c("shortcut", "hessian", "none"),
  sn_fit_logthresh = -6,
  sn_fit_temp = NA,
  control = list(),
  verbose = TRUE,
  debug = FALSE,
  add_priors = TRUE,
  optim_method = c("nlminb", "ucminf", "optim"),
  numerical_grad = FALSE,
  ...
)
```

Arguments

model	A description of the user-specified model. Typically, the model is described using the lavaan model syntax. See model.syntax for more information. Alternatively, a parameter table (eg. the output of the lavParTable() function) is also accepted.
data	An optional data frame containing the observed variables used in the model. If some variables are declared as ordered factors, lavaan will treat them as ordinal variables.
dp	Default prior distributions on different types of parameters, typically the result of a call to dpriors(). See the dpriors() help file for more information.
marginal_method	The method for approximating the marginal posterior distributions. Options include "skewnorm" (skew normal), "asymgaus" (two-piece asymmetric Gaussian), "marggaus" (marginalising the Laplace approximation), and "sampling" (sampling from the joint Laplace approximation).
nsamp	The number of samples to draw for all sampling-based approaches (including posterior sampling for model fit indices).
test	Character indicating whether to compute posterior fit indices. Defaults to "standard". Change to "none" to skip these computations.
marginal_correction	Which type of correction to use when fitting the skew normal or two-piece Gaussian marginals. "hessian" computes the full Hessian-based correction (slow), "shortcut" (default) computes only diagonals, and "none" (or FALSE) applies no correction.
sn_fit_logthresh	The log-threshold for fitting the skew normal. Points with log-posterior drop below this threshold (relative to the maximum) will be excluded from the fit. Defaults to -6.
sn_fit_temp	Temperature parameter for fitting the skew normal. If NA, the temperature will be included in the optimisation during the skew normal fit.
control	A list of control parameters for the optimiser.
verbose	Logical indicating whether to print progress messages.
debug	Logical indicating whether to return debug information.
add_priors	Logical indicating whether to include prior densities in the posterior computation.
optim_method	The optimisation method to use for finding the posterior mode. Options include "nlminb" (default), "ucminf", and "optim" (BFGS).
numerical_grad	Logical indicating whether to use numerical gradients for the optimisation.
...	Additional arguments to be passed to the lavaan::lavaan model fitting function.

Details

The acfa() function is a wrapper for the more general inlavaan() function, using the following default arguments:

- int.ov.free = TRUE
- int.lv.free = FALSE
- auto.fix.first = TRUE (unless std.lv = TRUE)
- auto.fix.single = TRUE
- auto.var = TRUE
- auto.cov.lv.x = TRUE
- auto.efa = TRUE
- auto.th = TRUE
- auto.delta = TRUE
- auto.cov.y = TRUE

For further information regarding these arguments, please refer to the lavaan::lavOptions() documentation.

Value

An S4 object of class INLAvaan which is a subclass of the lavaan::lavaan class.

See Also

Typically, users will interact with the specific latent variable model functions instead, including acfa(), asem(), and agrowth().

Examples

```
# The famous Holzinger and Swineford (1939) example
HS.model <- '
  visual  =~ x1 + x2 + x3
  textual =~ x4 + x5 + x6
  speed   =~ x7 + x8 + x9
'
utils::data("HolzingerSwineford1939", package = "lavaan")

# Fit a CFA model with standardised latent variables
fit <- acfa(HS.model, data = HolzingerSwineford1939, std.lv = TRUE, nsamp = 100)
summary(fit)
```

---

agrowth *Fit an Approximate Bayesian Growth Curve Model*

Description

Fit an Approximate Bayesian Growth Curve Model

Usage

```
agrowth(
  model,
  data,
  dp = blavaan::dpriors(),
  marginal_method = c("skewnorm", "asymgaus", "marggaus", "sampling"),
  nsamp = 500,
  test = "standard",
  marginal_correction = c("shortcut", "hessian", "none"),
  sn_fit_logthresh = -6,
  sn_fit_temp = NA,
  control = list(),
  verbose = TRUE,
  debug = FALSE,
  add_priors = TRUE,
  optim_method = c("nlminb", "ucminf", "optim"),
  numerical_grad = FALSE,
  ...
)
```

Arguments

model	A description of the user-specified model. Typically, the model is described using the lavaan model syntax. See model.syntax for more information. Alternatively, a parameter table (eg. the output of the lavParTable() function) is also accepted.
data	An optional data frame containing the observed variables used in the model. If some variables are declared as ordered factors, lavaan will treat them as ordinal variables.
dp	Default prior distributions on different types of parameters, typically the result of a call to dpriors(). See the dpriors() help file for more information.
marginal_method	The method for approximating the marginal posterior distributions. Options include "skewnorm" (skew normal), "asymgaus" (two-piece asymmetric Gaussian), "marggaus" (marginalising the Laplace approximation), and "sampling" (sampling from the joint Laplace approximation).
nsamp	The number of samples to draw for all sampling-based approaches (including posterior sampling for model fit indices).
test	Character indicating whether to compute posterior fit indices. Defaults to "standard". Change to "none" to skip these computations.
marginal_correction	Which type of correction to use when fitting the skew normal or two-piece Gaussian marginals. "hessian" computes the full Hessian-based correction (slow), "shortcut" (default) computes only diagonals, and "none" (or FALSE) applies no correction.

`sn_fit_logthresh`
: The log-threshold for fitting the skew normal. Points with log-posterior drop below this threshold (relative to the maximum) will be excluded from the fit. Defaults to `-6`.

`sn_fit_temp`
: Temperature parameter for fitting the skew normal. If `NA`, the temperature will be included in the optimisation during the skew normal fit.

`control`
: A list of control parameters for the optimiser.

`verbose`
: Logical indicating whether to print progress messages.

`debug`
: Logical indicating whether to return debug information.

`add_priors`
: Logical indicating whether to include prior densities in the posterior computation.

`optim_method`
: The optimisation method to use for finding the posterior mode. Options include `"nlminb"` (default), `"ucminf"`, and `"optim"` (BFGS).

`numerical_grad`
: Logical indicating whether to use numerical gradients for the optimisation.

`...`
: Additional arguments to be passed to the lavaan::lavaan model fitting function.

Details

The `asem()` function is a wrapper for the more general `inlavaan()` function, using the following default arguments:

- `meanstructure = TRUE`
- `int.ov.free = FALSE`
- `int.lv.free = TRUE`
- `auto.fix.first = TRUE` (unless `std.lv = TRUE`)
- `auto.fix.single = TRUE`
- `auto.var = TRUE`
- `auto.cov.lv.x = TRUE`
- `auto.efa = TRUE`
- `auto.th = TRUE`
- `auto.delta = TRUE`
- `auto.cov.y = TRUE`

Value

An S4 object of class `INLAvaan` which is a subclass of the lavaan::lavaan class.

See Also

Typically, users will interact with the specific latent variable model functions instead, including `acfa()`, `asem()`, and `agrowth()`.

Examples

```
# Linear growth model with a time-varying covariate
mod <- "
  # Intercept and slope with fixed coefficients
    i =~ 1*t1 + 1*t2 + 1*t3 + 1*t4
    s =~ 0*t1 + 1*t2 + 2*t3 + 3*t4

  # (Latent) regressions
    i ~ x1 + x2
    s ~ x1 + x2

  # Time-varying covariates
    t1 ~ c1
    t2 ~ c2
    t3 ~ c3
    t4 ~ c4
"
utils::data("Demo.growth", package = "lavaan")
str(Demo.growth)

fit <- agrowth(mod, data = Demo.growth, nsamp = 100)
summary(fit)
```

---

asem *Fit an Approximate Bayesian Structural Equation Model*

Description

Fit an Approximate Bayesian Structural Equation Model

Usage

```
asem(
  model,
  data,
  dp = blavaan::dpriors(),
  marginal_method = c("skewnorm", "asymgaus", "marggaus", "sampling"),
  nsamp = 500,
  test = "standard",
  marginal_correction = c("shortcut", "hessian", "none"),
  sn_fit_logthresh = -6,
  sn_fit_temp = NA,
  control = list(),
  verbose = TRUE,
  debug = FALSE,
  add_priors = TRUE,
  optim_method = c("nlminb", "ucminf", "optim"),
  numerical_grad = FALSE,
```

```
  ...
)
```

Arguments

model	A description of the user-specified model. Typically, the model is described using the lavaan model syntax. See model.syntax for more information. Alternatively, a parameter table (eg. the output of the lavParTable() function) is also accepted.
data	An optional data frame containing the observed variables used in the model. If some variables are declared as ordered factors, lavaan will treat them as ordinal variables.
dp	Default prior distributions on different types of parameters, typically the result of a call to dpriors(). See the dpriors() help file for more information.
marginal_method	The method for approximating the marginal posterior distributions. Options include "skewnorm" (skew normal), "asymgaus" (two-piece asymmetric Gaussian), "marggaus" (marginalising the Laplace approximation), and "sampling" (sampling from the joint Laplace approximation).
nsamp	The number of samples to draw for all sampling-based approaches (including posterior sampling for model fit indices).
test	Character indicating whether to compute posterior fit indices. Defaults to "standard". Change to "none" to skip these computations.
marginal_correction	Which type of correction to use when fitting the skew normal or two-piece Gaussian marginals. "hessian" computes the full Hessian-based correction (slow), "shortcut" (default) computes only diagonals, and "none" (or FALSE) applies no correction.
sn_fit_logthresh	The log-threshold for fitting the skew normal. Points with log-posterior drop below this threshold (relative to the maximum) will be excluded from the fit. Defaults to -6.
sn_fit_temp	Temperature parameter for fitting the skew normal. If NA, the temperature will be included in the optimisation during the skew normal fit.
control	A list of control parameters for the optimiser.
verbose	Logical indicating whether to print progress messages.
debug	Logical indicating whether to return debug information.
add_priors	Logical indicating whether to include prior densities in the posterior computation.
optim_method	The optimisation method to use for finding the posterior mode. Options include "nlminb" (default), "ucminf", and "optim" (BFGS).
numerical_grad	Logical indicating whether to use numerical gradients for the optimisation.
...	Additional arguments to be passed to the lavaan::lavaan model fitting function.

Details

The `asem()` function is a wrapper for the more general `inlavaan()` function, using the following default arguments:

- `int.ov.free` = TRUE
- `int.lv.free` = FALSE
- `auto.fix.first` = TRUE (unless `std.lv` = TRUE)
- `auto.fix.single` = TRUE
- `auto.var` = TRUE
- `auto.cov.lv.x` = TRUE
- `auto.efa` = TRUE
- `auto.th` = TRUE
- `auto.delta` = TRUE
- `auto.cov.y` = TRUE

For further information regarding these arguments, please refer to the `lavaan::lavOptions()` documentation.

Value

An S4 object of class `INLAvaan` which is a subclass of the lavaan::lavaan class.

See Also

Typically, users will interact with the specific latent variable model functions instead, including `acfa()`, `asem()`, and `agrowth()`.

Examples

```
# The industrialization and Political Democracy Example from Bollen (1989), page
# 332
model <- "
  # Latent variable definitions
     ind60 =~ x1 + x2 + x3
     dem60 =~ y1 + a*y2 + b*y3 + c*y4
     dem65 =~ y5 + a*y6 + b*y7 + c*y8

  # (Latent) regressions
    dem60 ~ ind60
    dem65 ~ ind60 + dem60

  # Residual correlations
    y1 ~~ y5
    y2 ~~ y4 + y6
    y3 ~~ y7
    y4 ~~ y8
    y6 ~~ y8
"
```

```
utils::data("PoliticalDemocracy", package = "lavaan")

fit <- asem(model, PoliticalDemocracy, test = "none")
summary(fit)
```

`as_fun_string` *Convert function to single string*

Description

Convert function to single string

Usage

```
as_fun_string(f)
```

Arguments

`f`	Function to convert.

Value

A single character vector representing the function.

Examples

```
f <- function(x) { x^2 + 1 }
as_fun_string(f)
```

`compare` *Compare Bayesian Models Fitted with INLAvaan*

Description

Compare Bayesian Models Fitted with INLAvaan

Usage

```
compare(x, y, ...)

## S4 method for signature 'INLAvaan'
compare(x, y, ...)
```

Arguments

`x, y, ...`	An object of class `INLAvaan` or `inlavaan_internal`.

Details

The function computes the log Bayes Factor (logBF) relative to the best fitting model (the one with the highest Marginal Log-Likelihood).

The output table sorts models by descending Marginal Log-Likelihood.

- **Marg.Loglik**: The approximated marginal log-likelihood.
- **DIC**: Deviance Information Criterion (if available).
- **pD**: Effective number of parameters (if available).
- **logBF**: The natural logarithm of the Bayes Factor relative to the best model.

Value

A data frame of class `compare.inlavaan_internal` containing model fit statistics.

References

https://lavaan.ugent.be/tutorial/groups.html

Examples

```
# Model comparison on multigroup analysis (measurement invariance)
HS.model <- "
  visual  =~ x1 + x2 + x3
  textual =~ x4 + x5 + x6
  speed   =~ x7 + x8 + x9
"
utils::data("HolzingerSwineford1939", package = "lavaan")

# Configural invariance
fit1 <- acfa(HS.model, data = HolzingerSwineford1939, group = "school")

# Weak invariance
fit2 <- acfa(
  HS.model,
  data = HolzingerSwineford1939,
  group = "school",
  group.equal = "loadings"
)

# Strong invariance
fit3 <- acfa(
  HS.model,
  data = HolzingerSwineford1939,
  group = "school",
  group.equal = c("intercepts", "loadings")
)

# Compare models
compare(fit1, fit2, fit3)
```

dbeta_box *Density of a Beta distribution on a bounded interval*

Description

Density of a Beta distribution on a bounded interval

Usage

```
dbeta_box(x, shape1, shape2, a, b, log = FALSE)
```

Arguments

x	A numeric vector of quantiles.
shape1, shape2	non-negative parameters of the Beta distribution.
a	The lower bound of the interval.
b	The upper bound of the interval.
log	Logical; if TRUE, probabilities p are given as log(p).

Value

A numeric vector of density values.

Examples

```
# Beta(2,5) on (0,100)
x <- seq(0, 100, length.out = 100)
y <- dbeta_box(x, shape1 = 2, shape2 = 5, a = 0, b = 100)
plot(x, y, type = "l", main = "Beta(2,5) on (0,100)")

# Beta(1,1) i.e. uniform on (-1, 1)
x <- seq(-1, 1, length.out = 100)
y <- dbeta_box(x, shape1 = 1, shape2 = 1, a = -1, b = 1)
plot(x, y, type = "l", main = "Beta(1,1) on (-1,1)")
```

dsnorm *The Skew Normal Distribution*

Description

Density for the skew normal distribution with location `xi`, scale `omega`, and shape `alpha`.

Usage

```
dsnorm(x, xi, omega, alpha, logC = 0, log = FALSE)
```

Arguments

x	Vector of quantiles.
xi	Location parameter.
omega	Scale parameter.
alpha	Shape parameter.
logC	Log-normalization constant.
log	Logical; if TRUE, returns the log density.

Value

A numeric vector of (log) density values.

References

https://en.wikipedia.org/wiki/Skew_normal_distribution

Examples

```
x <- seq(-2, 5, length.out = 100)
y <- dsnorm(x, xi = 0, omega = 1, alpha = 5)
plot(x, y, type = "l", main = "Skew Normal Density")
```

---

fitMeasures,INLAvaan-method

Fit Measures for a Latent Variable Model estimated using INLA

Description

Fit Measures for a Latent Variable Model estimated using INLA

Usage

```
## S4 method for signature 'INLAvaan'
fitMeasures(object, fit.measures = "all", baseline.model = NULL)

## S4 method for signature 'INLAvaan'
fitmeasures(object, fit.measures = "all", baseline.model = NULL)
```

Arguments

object	An object of class INLAvaan.
fit.measures	If "all", all fit measures available will be returned. If only a single or a few fit measures are specified by name, only those are computed and returned.
baseline.model	Not currently used, added for compatability with {lavaan}.

Value

A named numeric vector of fit measures.

`fit_skew_normal` *Fit a skew normal distribution to log-density evaluations*

Description

Fit a skew normal distribution to log-density evaluations

Usage

```
fit_skew_normal(x, y, threshold_log_drop = -6, temp = NA)
```

Arguments

`x`	A numeric vector of points where the density is evaluated.
`y`	A numeric vector of log-density evaluations at points `x`.
`threshold_log_drop`	A negative numeric value indicating the log-density drop threshold below which points are ignored in the fitting. Default is -6.
`temp`	A numeric value for the temperature parameter `k`. If NA (default), it is included in the optimisation.

Details

This skew normal fitting function uses a weighted least squares approach to fit the log-density evaluations provided in `y` at points `x`. The weights are set to be the density evaluations raised to the power of the temperature parameter `k`. This has somewhat an interpretation of finding the skew normal fit that minimises the Kullback-Leibler divergence from the true density to it.

In R-INLA, the C code implementation from which this was translated from can be found here.

Value

A list with fitted parameters:

- `xi`: location parameter
- `omega`: scale parameter
- `alpha`: shape parameter
- `logC`: log-normalization constant
- `k`: temperature parameter
- `rsq`: R-squared of the fit

Note that `logC` and `k` are not used when fitting from a sample.

Examples

```
library(sn)
library(tidyr)
library(ggplot2)

logdens <- function(x) dgamma(x, shape = 3, rate = 1, log = TRUE)

x_grid <- seq(0.1, 8, length.out = 21)
y_log <- sapply(x_grid, logdens)
y_log <- y_log - max(y_log) # normalise to have maximum at zero

res <- fit_skew_normal(x_grid, y_log, temp = 10)
unlist(res)

plot_df <-
  pivot_longer(
    tibble(
      x = seq(0.1, 8, length.out = 200),
      truth = exp(logdens(x)),
      approx = dsnorm(x, xi = res$xi, omega = res$omega, alpha = res$alpha)
    ),
    cols = c("truth", "approx"),
    names_to = "type",
    values_to = "density"
  )

ggplot() +
  # truth as filled area
  geom_area(
    data = subset(plot_df, type == "truth"),
    aes(x, density, fill = "Truth"),
    alpha = 0.38
  ) +
  # approx as blue line
  geom_line(
    data = subset(plot_df, type == "approx"),
    aes(x = x, y = density, col = "SN Approx."),
    linewidth = 1
  ) +
  scale_fill_manual(name = NULL, values = "#131516") +
  scale_colour_manual(name = NULL, values = "#00A6AA") +
  theme_minimal() +
  theme(legend.position = "top")
```

`fit_skew_normal_samp` *Fit a skew normal distribution to a sample*

Description

Fit a skew normal distribution to a sample

Usage

```
fit_skew_normal_samp(x)
```

Arguments

x	A numeric vector of sample data.

Details

Uses maximum likelihood estimation to fit a skew normal distribution to the provided numeric vector x.

Value

A list with fitted parameters:

- xi: location parameter
- omega: scale parameter
- alpha: shape parameter
- logC: log-normalization constant
- k: temperature parameter
- rsq: R-squared of the fit

Note that logC and k are not used when fitting from a sample.

Examples

```
x <- rnorm(100, mean = 5, sd = 1)
unlist(fit_skew_normal_samp(x))
```

---

inlavaan *Fit an Approximate Bayesian Latent Variable Model*

Description

This function fits a Bayesian latent variable model by approximating the posterior distributions of the model parameters using various methods, including skew normal, asymmetric Gaussian, marginal Gaussian, or sampling-based approaches. It leverages the lavaan package for model specification and estimation.

Usage

```
inlavaan(
  model,
  data,
  model.type = "sem",
  dp = blavaan::dpriors(),
  vb_correction = TRUE,
  marginal_method = c("skewnorm", "asymgaus", "marggaus", "sampling"),
  marginal_correction = c("shortcut", "hessian", "none"),
  nsamp = 500,
  test = "standard",
  sn_fit_logthresh = -6,
  sn_fit_temp = NA,
  control = list(),
  verbose = TRUE,
  debug = FALSE,
  add_priors = TRUE,
  optim_method = c("nlminb", "ucminf", "optim"),
  numerical_grad = FALSE,
  ...
)
```

Arguments

model	A description of the user-specified model. Typically, the model is described using the lavaan model syntax. See model.syntax for more information. Alternatively, a parameter table (eg. the output of the lavParTable() function) is also accepted.
data	An optional data frame containing the observed variables used in the model. If some variables are declared as ordered factors, lavaan will treat them as ordinal variables.
model.type	Set the model type: possible values are "cfa", "sem" or "growth". This may affect how starting values are computed, and may be used to alter the terminology used in the summary output, or the layout of path diagrams that are based on a fitted lavaan object.
dp	Default prior distributions on different types of parameters, typically the result of a call to dpriors(). See the dpriors() help file for more information.
vb_correction	Logical indicating whether to apply a variational Bayes correction for the posterior mean vector of estimates. Defaults to TRUE.
marginal_method	The method for approximating the marginal posterior distributions. Options include "skewnorm" (skew normal), "asymgaus" (two-piece asymmetric Gaussian), "marggaus" (marginalising the Laplace approximation), and "sampling" (sampling from the joint Laplace approximation).
marginal_correction	Which type of correction to use when fitting the skew normal or two-piece Gaussian marginals. "hessian" computes the full Hessian-based correction (slow),

	"shortcut" (default) computes only diagonals, and "none" (or FALSE) applies no correction.
nsamp	The number of samples to draw for all sampling-based approaches (including posterior sampling for model fit indices).
test	Character indicating whether to compute posterior fit indices. Defaults to "standard". Change to "none" to skip these computations.
sn_fit_logthresh	The log-threshold for fitting the skew normal. Points with log-posterior drop below this threshold (relative to the maximum) will be excluded from the fit. Defaults to -6.
sn_fit_temp	Temperature parameter for fitting the skew normal. If NA, the temperature will be included in the optimisation during the skew normal fit.
control	A list of control parameters for the optimiser.
verbose	Logical indicating whether to print progress messages.
debug	Logical indicating whether to return debug information.
add_priors	Logical indicating whether to include prior densities in the posterior computation.
optim_method	The optimisation method to use for finding the posterior mode. Options include "nlminb" (default), "ucminf", and "optim" (BFGS).
numerical_grad	Logical indicating whether to use numerical gradients for the optimisation.
...	Additional arguments to be passed to the lavaan::lavaan model fitting function.

Value

An S4 object of class INLAvaan which is a subclass of the lavaan::lavaan class.

See Also

Typically, users will interact with the specific latent variable model functions instead, including acfa(), asem(), and agrowth().

Examples

```
# The Holzinger and Swineford (1939) example
HS.model <- "
  visual  =~ x1 + x2 + x3
  textual =~ x4 + x5 + x6
  speed   =~ x7 + x8 + x9
"
utils::data("HolzingerSwineford1939", package = "lavaan")

fit <- inlavaan(
  HS.model,
  data = HolzingerSwineford1939,
  auto.var = TRUE,
  auto.fix.first = TRUE,
  auto.cov.lv.x = TRUE
```

```
)
summary(fit)
```

INLAvaan-class *Class For Representing a (Fitted) Latent Variable Model*

Description

This is a class that extends the lavaan::lavaan class. Several S4 methods are available.

Usage

```
## S4 method for signature 'INLAvaan,ANY'
plot(x, y, ...)

## S4 method for signature 'INLAvaan'
predict(object, nsamp = 1000, ...)

## S4 method for signature 'INLAvaan'
show(object)

## S4 method for signature 'INLAvaan'
coef(object)

## S4 method for signature 'INLAvaan'
summary(
  object,
  header = TRUE,
  fit.measures = TRUE,
  estimates = TRUE,
  standardized = FALSE,
  rsquare = FALSE,
  postmedian = FALSE,
  postmode = FALSE,
  priors = TRUE,
  nd = 3L,
  ...
)

## S4 method for signature 'INLAvaan'
vcov(object)
```

Arguments

x	An object of class INLAvaan.
y	Not used.
...	Not used.

object	An object of class INLAvaan.
nsamp	The number of samples to draw for all sampling-based approaches (including posterior sampling for model fit indices).
header	Logical; if TRUE, print model fit information header.
fit.measures	Logical; if TRUE, print fit measures (DIC and PPP).
estimates	Logical; if TRUE, print parameter estimates table.
standardized	Logical; if TRUE, include standardized estimates.
rsquare	Logical; if TRUE, include R-square values.
postmedian	Logical; if TRUE, include posterior median in estimates.
postmode	Logical; if TRUE, include posterior mode in estimates.
priors	Logical; if TRUE, include prior information in estimates.
nd	Integer; number of decimal places to print for numeric values.

Slots

`external` A list containing an `inlavaan_internal` object.

See Also

lavaan::lavaan

---

is_same_function *Helper function to check if two functions are the same*

Description

Helper function to check if two functions are the same

Usage

```
is_same_function(f, g)
```

Arguments

f, g	Functions to compare.

Value

Logical.

Examples

```
f1 <- function(x) { x^2 + 1 }
f2 <- function(x) { x^2 + 1 }
is_same_function(f1, f2)  # TRUE
```

`qsnorm_fast` *Fast Approximation of Skew-Normal Quantile Function*

Description

A fast approximation of skew-normal quantiles using the high-performance approximation algorithm from the INLA GMRFLib C source, and originally by Thomas Luu (see details for reference).

Usage

```
qsnorm_fast(p, xi = 0, omega = 1, alpha = 0)
```

Arguments

`p`	Vector of probabilities.
`xi`	Location parameter (numeric vector).
`omega`	Scale parameter (numeric vector).
`alpha`	Shape parameter (numeric vector).

Details

This function implements a high-performance approximation for the skew-normal quantile function based on the algorithm described by Luu (2016). The method uses a domain decomposition strategy to achieve high accuracy ($< 10^{-7}$ relative error) without iterative numerical inversion.

The domain is split into two regions:

- **Tail Regions:** For extreme probabilities where $|u|$ is large, the quantile is approximated using the Lambert W-function, $W(z)$, solving $z = \Phi(q)$ via asymptotic expansion:

$$q \approx \sqrt{2W\left(\frac{1}{2\pi(1-p)^2}\right)}$$

- **Central Region:** For the main body of the distribution, the function uses a high-order Taylor expansion of the inverse error function around a carefully selected expansion point x_0:

$$\Phi^{-1}(p) \approx \sum_{k=0}^{5} c_k (z - x_0)^k$$

This approach is significantly faster than standard numerical inversion (e.g., `uniroot`) while maintaining sufficient precision for most statistical applications.

Value

Vector of quantiles.

References

Luu, T. (2016). *Fast and accurate parallel computation of quantile functions for random number generation* #' (Doctoral thesis). UCL (University College London). https://discovery.ucl.ac.uk/1482128/

`standardisedsolution` *Standardised solution of a latent variable model*

Description

Standardised solution of a latent variable model

Usage

```
standardisedsolution(
  object,
  type = "std.all",
  se = TRUE,
  ci = TRUE,
  level = 0.95,
  postmedian = FALSE,
  postmode = FALSE,
  cov.std = TRUE,
  remove.eq = TRUE,
  remove.ineq = TRUE,
  remove.def = FALSE,
  nsamp = 250,
  ...
)

standardisedSolution(
  object,
  type = "std.all",
  se = TRUE,
  ci = TRUE,
  level = 0.95,
  postmedian = FALSE,
  postmode = FALSE,
  cov.std = TRUE,
  remove.eq = TRUE,
  remove.ineq = TRUE,
  remove.def = FALSE,
  nsamp = 250,
  ...
)
```

```
standardizedsolution(
  object,
  type = "std.all",
  se = TRUE,
  ci = TRUE,
  level = 0.95,
  postmedian = FALSE,
  postmode = FALSE,
  cov.std = TRUE,
  remove.eq = TRUE,
  remove.ineq = TRUE,
  remove.def = FALSE,
  nsamp = 250,
  ...
)

standardizedSolution(
  object,
  type = "std.all",
  se = TRUE,
  ci = TRUE,
  level = 0.95,
  postmedian = FALSE,
  postmode = FALSE,
  cov.std = TRUE,
  remove.eq = TRUE,
  remove.ineq = TRUE,
  remove.def = FALSE,
  nsamp = 250,
  ...
)
```

Arguments

`object`	An object of class INLAvaan.
`type`	If `"std.lv"`, the standardized estimates are on the variances of the (continuous) latent variables only. If `"std.all"`, the standardized estimates are based on both the variances of both (continuous) observed and latent variables. If `"std.nox"`, the standardized estimates are based on both the variances of both (continuous) observed and latent variables, but not the variances of exogenous covariates.
`se`	Logical. If TRUE, standard errors for the standardized parameters will be computed, together with a z-statistic and a p-value.
`ci`	If `TRUE`, simple symmetric confidence intervals are added to the output
`level`	The confidence level required.
`postmedian`	Logical; if TRUE, include posterior median in estimates.
`postmode`	Logical; if TRUE, include posterior mode in estimates.

cov.std	Logical. If TRUE, the (residual) observed covariances are scaled by the square root of the 'Theta' diagonal elements, and the (residual) latent covariances are scaled by the square root of the 'Psi' diagonal elements. If FALSE, the (residual) observed covariances are scaled by the square root of the diagonal elements of the observed model-implied covariance matrix (Sigma), and the (residual) latent covariances are scaled by the square root of diagonal elements of the model-implied covariance matrix of the latent variables.
remove.eq	Logical. If TRUE, filter the output by removing all rows containing equality constraints, if any.
remove.ineq	Logical. If TRUE, filter the output by removing all rows containing inequality constraints, if any.
remove.def	Logical. If TRUE, filter the ouitput by removing all rows containing parameter definitions, if any.
nsamp	The number of samples to draw from the approximate posterior distribution for the calculation of standardised estimates.
...	Additional arguments sent to lavaan::standardizedSolution().

Value

A data.frame containing standardised model parameters.

Examples

```
HS.model <- "
  visual  =~ x1 + x2 + x3
  textual =~ x4 + x5 + x6
  speed   =~ x7 + x8 + x9
"
utils::data("HolzingerSwineford1939", package = "lavaan")

# Fit a CFA model with standardised latent variables
fit <- acfa(HS.model, data = HolzingerSwineford1939, test = "none")
standardisedsolution(fit, nsamp = 100)
```

Index

acfa, 2
acfa(), *3*, *4*, *6*, *9*, *18*
agrowth, 4
agrowth(), *4*, *6*, *9*, *18*
as_fun_string, 10
asem, 7
asem(), *4*, *6*, *9*, *18*

coef,INLAvaan-method *(*INLAvaan-class*)*, 19
compare, 10
compare,INLAvaan-class *(*compare*)*, 10
compare,INLAvaan-method *(*compare*)*, 10

dbeta_box, 12
dsnorm, 12

fit_skew_normal, 14
fit_skew_normal_samp, 15
fitMeasures,INLAvaan-method, 13
fitmeasures,INLAvaan-method *(*fitMeasures,INLAvaan-method*)*, 13

INLAvaan, *13*, *19, 20*, *23*
inlavaan, 16
inlavaan(), *3*, *6*, *9*
INLAvaan-class, 19
is_same_function, 20

lavaan::lavaan, *3, 4*, *6*, *8, 9*, *18–20*
lavaan::lavOptions(), *4*, *9*

model.syntax, *3*, *5*, *8*, *17*

plot,INLAvaan,ANY-method *(*INLAvaan-class*)*, 19
predict,INLAvaan-method *(*INLAvaan-class*)*, 19

qsnorm_fast, 21

show,INLAvaan-method *(*INLAvaan-class*)*, 19
standardisedSolution *(*standardisedsolution*)*, 22
standardisedsolution, 22
standardizedSolution *(*standardisedsolution*)*, 22
standardizedsolution *(*standardisedsolution*)*, 22
summary,INLAvaan-method *(*INLAvaan-class*)*, 19

vcov,INLAvaan-method *(*INLAvaan-class*)*, 19