Package ‘plavaan’

December 30, 2025

Title Penalized Estimation for Latent Variable Models with 'lavaan'

Version 0.0.1

Description Extends the popular 'lavaan' package by adding penalized estimation capabilities. It supports penalty on individual parameters as well as the difference between parameters.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Imports lavaan, numDeriv

URL https://marklhc.github.io/plavaan/, https://github.com/marklhc/plavaan

BugReports https://github.com/marklhc/plavaan/issues

NeedsCompilation no

Author Hok Chio (Mark) Lai [aut, cre] (ORCID: <https://orcid.org/0000-0002-9196-7406>)

Maintainer Hok Chio (Mark) Lai <marklhc@gmail.com>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2025-12-30 19:00:02 UTC

Contents

composite_pair_loss 2
loss 3
penalized_est 4

Index **7**

composite_pair_loss *Composite Pairwise Loss Function*

Description

Computes the total loss across all pairwise combinations of rows in a matrix.

Usage

```
composite_pair_loss(x, fun, trans = identity, rescale = "df", ...)
```

Arguments

`x`	A numeric vector, matrix, or data frame. If not a matrix, it will be coerced to one after applying the transformation function.
`fun`	A function to compute the loss for each pairwise difference. The package supports the alignment loss (`alf`) and the approximate L0 penalty (`l0a`), but users can provide custom functions as well.
`trans`	A transformation function to apply to `x` before computing pairwise differences. Default is `identity` (no transformation).
`rescale`	Either `"df"` (default) to rescale the total loss by the degrees of freedom (number of rows - 1), or a numeric value (likely between 0 and 1) to multiply the total loss by.
`...`	Additional arguments passed to the loss function `fun`.

Details

The function works by:

1. Applying the transformation function `trans` to the input `x`
2. Converting the result to a matrix
3. Generating all possible pairwise combinations of row indices
4. Computing the difference between each pair of rows
5. Applying the loss function `fun` to each difference
6. Summing all the individual losses

Value

A numeric scalar representing the sum of losses across all pairwise combinations of rows.

Examples

```
# Example with a simple matrix
x <- matrix(runif(12), nrow = 4)
composite_pair_loss(x, fun = alf)

# Example with log transformation and L2 loss
composite_pair_loss(x, fun = function(x) x^2, trans = log)
```

`loss` *Loss functions*

Description

For small eps this provides a smooth, numerically stable approximation of |x|^(1/2) (i.e. the square root of the absolute value). The function is vectorized over x.

Usage

```
alf(x, eps = 0.001)

l0a(x, eps = 0.01)
```

Arguments

`x`	Numeric vector. Input values to transform.
`eps`	Positive numeric scalar (default .001 for `alf()` and .01 for `l0a()`). Small regularization constant to avoid non-differentiability and division-by-zero issues.

Details

The ALF, (x^2 + eps)^(1/4), is useful when a smooth surrogate for sqrt(|x|) is required (for optimization or regularization) while maintaining numerical stability near x = 0.

L0a, x^2/(x^2 + eps), is an approximation of the L0 penalty.

Value

Numeric vector of the same length as x.

Examples

```
alf(0)
alf(c(-4, -1, 0, 1, 4))
alf(0.5, eps = 1e-6)
l0a(0)
l0a(c(0, 1e-3, 0.1, 1))
l0a(c(-2, 0, 2), eps = 1e-4)
```

penalized_est — *Penalized Parameter Estimation for Longitudinal CFA Models*

Description

Performs penalized estimation on a lavaan model object by optimizing a penalized objective function. The function extracts the objective function from a lavaan model, applies a penalty function to specified parameters or pairwise differences of parameters, and returns an updated model with the optimized parameter estimates.

Usage

```
penalized_est(
  x,
  w,
  pen_par_id = NULL,
  pen_diff_id = NULL,
  pen_fn = "l0a",
  pen_gr = NULL,
  se = "none",
  opt_control = list()
)
```

Arguments

`x` A fitted lavaan model object from which estimation components will be extracted.

`w` Numeric scalar. Penalty weight (multiplier) applied to the penalty terms.

`pen_par_id` Integer vector of parameter IDs to apply the penalty function directly to, in the same order as returned by `lavaan::coef()` and by `lavaan::partable()`, with only the free elements.

`pen_diff_id` List of matrices containing parameter IDs. For each matrix, the penalty is applied to the pairwise differences of parameters in the same column indicated by the IDs. For matrices with names starting with "loading", the log transformation is applied before computing differences.

`pen_fn` A character string (`"l0a"` or `"alf"`) or a function that computes the penalty. Default is `"l0a"`.

`pen_gr` A function that computes the gradient of the penalty function. If `pen_fn` is `"l0a"` or `"alf"`, this is automatically set.

`se` Character string specifying the type of standard errors to compute. Options are `"none"` (default; no standard errors) or `"robust.huber.white"` (robust sandwich estimator using numerical Hessian and first-order information, which is the same as used in the `"mlr"` estimator).

`opt_control` A list of control parameters passed to `stats::nlminb()`. Default includes `eval.max = 2e4`, `iter.max = 1e4`, and `abs.tol = 1e-20`.

Details

The function uses `nlminb()` to minimize a penalized objective function that combines the standard lavaan objective function with a penalty term. Only the parameter estimates and the log-likelihood should be interpreted. The returned object was not "fitted" (`do.fit = FALSE`) to avoid users interpreting the standard errors, which are generally not valid with penalized estimation. The degrees of freedom may also be inaccurate. If the optimization does not converge (convergence code != 0), a warning is issued.

Value

A lavaan model object updated with the penalized parameter estimates. The returned object includes an attribute `opt_info` containing the optimization information returned by `nlminb()`.

Warning

The returned object is not fitted using standard ML. Standard errors reported by `summary()` or `parameterEstimates()` will be missing unless `se = "robust.huber.white"` was specified. Even then, they are based on an experimental sandwich approximation and should be interpreted with caution.

See Also

`lavaan`, `nlminb`

Examples

```
library(lavaan)

# Define a longitudinal factor model with PoliticalDemocracy data
model <- "
  dem60 =~ y1 + y2 + y3 + y4
  dem65 =~ y5 + y6 + y7 + y8
  dem60 ~~ dem65
  dem60 ~~ 1 * dem60
  dem65 ~~ NA * dem65
  dem60 ~ 0
  dem65 ~ NA * 1
  y1 ~~ y5
  y2 ~~ y6
  y3 ~~ y7
  y4 ~~ y8
"

# Fit the model without constraints first to get parameter table
fit_un <- cfa(model, data = PoliticalDemocracy, std.lv = TRUE,
              meanstructure = TRUE, do.fit = FALSE)

# Get parameter IDs
pt <- parTable(fit_un)
# Loadings
load_60 <- pt$free[pt$op == "=~" & pt$lhs == "dem60"]
```

```
load_65 <- pt$free[pt$op == "=~" & pt$lhs == "dem65"]
# Intercepts
int_60 <- pt$free[pt$op == "~1" & pt$lhs %in% c("y1", "y2", "y3", "y4")]
int_65 <- pt$free[pt$op == "~1" & pt$lhs %in% c("y5", "y6", "y7", "y8")]

# Apply penalized estimation to penalize differences in loadings and intercepts
pen_fit <- penalized_est(
    x = fit_un,
    w = 0.03,
    pen_diff_id = list(
        loadings = rbind(load_60, load_65),
        intercepts = rbind(int_60, int_65)
    ),
    pen_fn = "l0a"
)

# Compare parameter estimates
summary(pen_fit)
```

Index

alf *(*loss*)*, 3

composite_pair_loss, 2

l0a *(*loss*)*, 3
lavaan, *5*
lavaan::partable(), *4*
loss, 3

nlminb, *5*

penalized_est, 4

stats::nlminb(), *4*