Package 'HiGarrote'

July 28, 2025

Type Package

Title Nonnegative Garrote Method Incorporating Hierarchical Relationships

Version 2.0.0

Date 2025-07-27

Author Wei-Yang Yu [aut, cre],
V. Roshan Joseph [aut]

Maintainer Wei-Yang Yu <wyu322@gatech.edu>

Description

An implementation of the nonnegative garrote method that incorporates hierarchical relationships among variables. The core function, HiGarrote(), offers an automated approach for analyzing experiments while respecting hierarchical structures among effects. For methodological details, refer to Yu and Joseph (2025) <doi:10.1080/00224065.2025.2513508>. This work is supported by U.S. National Science Foundation grant DMS-2310637.

License GPL (>= 2)

Imports Matrix, matrixcalc, MaxPro, nloptr, purrr, quadprog, Rcpp (>= 1.0.12), RcppArmadillo, rlist, scales, stringr

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.3.2

Encoding UTF-8

Depends R (>= 2.10)

LazyData true

NeedsCompilation yes

Repository CRAN

Date/Publication 2025-07-27 22:30:01 UTC

Contents

`blood_glucose` *Blood Glucose Experiment*

Description

Hamada and Wu (1992) analyzed an 18-run experiment designed to study blood glucose readings of a clinical testing device. The experiment contains one two-level factor and seven three-level quantitative factors, which are denoted by A through H.

Usage

```
data(blood_glucose)
```

Format

A data frame with 18 rows and 9 columns.

Source

Hamada, M. and Wu, C. F. J. (1992). Analysis of Designed Experiments with Complex Aliasing. *Journal of Quality Technology*, 24(3), 130-137. doi:10.1080/00224065.1992.11979383.

`cast_fatigue` *Cast Fatigue Experiment*

Description

Hunter et al. (1982) used a 12-run Plackett-Burman design to investigate the effects of seven two-level factors on the fatigue life of weld-repaired castings. The seven factors are denoted by capital letters A through G.

Usage

```
data(cast_fatigue)
```

Format

A data frame with 12 rows and 8 columns.

Source

Hunter, G. B., Hodi, F. S., and Eagar, T. W. (1982). High Cycle Fatigue of Weld Repaired Cast Ti-6AI-4V. *Metallurgical Transactions A*, 13(9), 1589-1594. doi:10.1007/BF02644799.

`epoxy` *Epoxy Experiment*

Description

Lin (1993) employed a half-fraction of a 28-run PB design to investigate an experiment aimed at developingan epoxy adhesive system. The original design matrix contains 14 runs and 24 factors. Since factors 13 and 16 were assigned to the same columns in the original design matrix, only factor 13 is listed here.

Usage

```
data(epoxy)
```

Format

A data frame with 14 rows and 24 columns.

Source

Lin, D. K. J. (1993). A New Class of Supersaturated Designs. *Technometrics*, 35(1), 28-31. doi:10.2307/1269286.

`HiGarrote` *An Automatic Method for the Analysis of Experiments using Hierarchical Garrote*

Description

'HiGarrote()' provides an automatic method for analyzing experimental data. This function applies the nonnegative garrote method to select important effects while preserving their hierarchical structures. It first estimates regression parameters using generalized ridge regression, where the ridge parameters are derived from a Gaussian process prior placed on the input-output relationship. Subsequently, the initial estimates will be used in the nonnegative garrote for effects selection.

Usage

```
HiGarrote(
  D,
  y,
  heredity = "weak",
  quali_id = NULL,
  quanti_id = NULL,
  quali_sum_idx = NULL,
  user_def_coding = NULL,
  user_def_coding_idx = NULL,
  model_type = 1
)
```

Arguments

`D` An $n \times p$ data frame for the unreplicated design matrix, where n is the run size and p is the number of factors.

`y` A vector for the responses corresponding to `D`. For replicated experiments, `y` should be an $n \times r$ matrix, where r is the number of replicates.

`heredity` Specifies the heredity principles to be used. Supported options are `"weak"` and `"strong"`. The default is `"weak"`.

`quali_id` A vector indexing qualitative factors. Qualitative factors are coded using Helmert coding. Different coding systems are allowed by specifying `quali_sum_idx`, `user_def_coding`, `user_def_coding_idx`.

`quanti_id` A vector indexing quantitative factors. Quantitative factors are coded using orthogonal polynomial coding.

`quali_sum_idx` Optional. Indicating which qualitative factors should use sum coding (`contr.sum()`).

`user_def_coding`

Optional. A list of user-defined orthogonal coding systems. Each element must be an orthogonal contrast matrix.

`user_def_coding_idx`

Optional. A list of indices specifying which qualitative factors should use the corresponding coding systems provided in `user_def_coding`.

`model_type` Integer indicating the type of model to construct.

model_type = 1 The model matrix includes all the main effects of qualitative factors, the first two main effects (linear and quadratic) of all the quantitative factors, and all the two-factor interactions generated by those main effects.

model_type = 2 The model matrix includes all the main effects of qualitative factors, the linear effects of all the quantitative factors, all the two-factor interactions generated by those main effects, and the quadratic effects of all the quantitative factors.

model_type = 3 The model matrix includes all the main effects of qualitative factors and the linear effects of all the quantitative factors.

The default is `model_type = 1`.

Value

The function returns a list with:

`nng_estimate` A vector for the nonnegative garrote estimates of the identified effects.

`U` A model matrix of `D`.

`pred_info` A list containing information needed for future predictions.

References

Yu, W. Y. and Joseph, V. R. (2025). Automated Analysis of Experiments using Hierarchical Garrote. *Journal of Quality Technology*, 1-15. doi:10.1080/00224065.2025.2513508.

Examples

```
# Cast fatigue experiment
data(cast_fatigue)
X <- cast_fatigue[,1:7]
y <- cast_fatigue[,8]
fit_Hi <- HiGarrote::HiGarrote(X, y)
fit_Hi$nng_estimate

# Blood glucose experiment
data(blood_glucose)
X <- blood_glucose[,1:8]
y <- blood_glucose[,9]
fit_Hi <- HiGarrote::HiGarrote(X, y, quanti_id = 2:8)
fit_Hi$nng_estimate


# Router bit experiment --- Use default Helmert coding
data(router_bit)
X <- router_bit[, 1:9]
y <- router_bit[,10]
fit_Hi <- HiGarrote::HiGarrote(X, y, quali_id = c(4,5))
fit_Hi$nng_estimate

# Router bit experiment --- Use sum coding
fit_Hi <- HiGarrote::HiGarrote(X, y, quali_id = c(4,5), quali_sum_idx = c(4,5))
fit_Hi$nng_estimate

# Router bit experiment --- Use user-defined coding system for qualitative factors
fit_Hi <- HiGarrote::HiGarrote(X, y, quali_id = c(4,5),
 user_def_coding = list(matrix(c(-1,-1,1,1,1,-1,-1,1,-1,1,-1,1), ncol = 3)),
 user_def_coding_idx = list(c(4,5)))
fit_Hi$nng_estimate

# Resin experiment --- Use model_type = 2
data(resin)
X <- resin[,1:9]
y <- log(resin$Impurity)
fit_Hi <- HiGarrote::HiGarrote(X, y, quanti_id = c(1:9), model_type = 2)
```

```
fit_Hi$nng_estimate

# Epoxy experiment --- Use model_type = 3
data(epoxy)
X <- epoxy[,1:23]
y <- epoxy[,24]
fit_Hi <- HiGarrote::HiGarrote(X, y, model_type = 3)
fit_Hi$nng_estimate

# Experiments with replicates
# Generate simulated data
data(cast_fatigue)
X <- cast_fatigue[,1:7]
U <- data.frame(model.matrix(~.^2, X)[,-1])
error <- matrix(rnorm(24), ncol = 2) # two replicates for each run
y <- 20*U$A + 10*U$A.B + 5*U$A.C + error
fit_Hi <- HiGarrote::HiGarrote(X, y)
fit_Hi$nng_estimate
```

nnGarrote *Nonnegative Garrote Method with Hierarchical Structures*

Description

'nnGarrote()' implements the nonnegative garrote method, as described in Yuan et al. (2009), for selecting important variables while preserving hierarchical structures. The method begins by obtaining the least squares estimates of the regression parameters under a linear model. These initial estimates are then used in the nonnegative garrote to perform variable selection. Note that this method is suitable only when the number of observations is much larger than the number of variables, ensuring that the least squares estimation remains reliable.

Usage

```
nnGarrote(X, y, heredity = "weak", model_type = 1)
```

Arguments

`X` An $n \times p$ input matrix, where n is the number of data and p is the number of variables.

`y` A vector for the responses.

`heredity` Specifies the heredity principles to be used. Supported options are `"weak"` and `"strong"`. The default is `"weak"`.

`model_type` Integer indicating the type of model to construct.

model_type = 1 The model matrix includes linear effects, two-factor interactions, and quadratic effects.

model_type = 2 The model matrix includes linear effects and two-factor interactions.

The default is `model_type = 1`.

Value

The function returns a list with:

`nng_estimate` A vector for the nonnegative garrote estimates of the identified variables.

`U` A scaled model matrix.

`pred_info` A list of information required for further prediction.

References

Yuan, M., Joseph, V. R., and Zou H. (2009). Structured Variable Selection and Estimation. *The Annals of Applied Statistics*, 3(4), 1738–1757. doi:10.1214/09AOAS254.

Examples

```
# Generate data
x1 <- runif(100)
x2 <- runif(100)
x3 <- runif(100)
error <- rnorm(100)
X <- data.frame(x1, x2, x3)
U <- model.matrix(~. + x1:x2 + x1:x3 + x2:x3 + I(x1^2) + I(x2^2) + I(x3^2) - 1, X)
U <- data.frame(scale(U))
colnames(U) <- c("x1", "x2", "x3", "x1:x1", "x2:x2", "x3:x3", "x1:x2", "x1:x3", "x2:x3")
y <- 3 + 3*U$x1 + 3*U$`x1:x1` + 3*U$`x1:x2`+ 3*U$`x1:x3` + error

# Fit nnGarrote
fit_nng <- HiGarrote::nnGarrote(X, y)
fit_nng$nng_estimate
```

`predict.HiGarrote` *Make Predictions from a "HiGarrote" Object*

Description

This function makes predictions from a linear model constructed using the important effects selected by `HiGarrote`.

Usage

```
## S3 method for class 'HiGarrote'
predict(object, new_D, ...)
```

Arguments

object	An HiGarrote object.
new_D	A new design matrix where predictions are to be made.
...	Additional arguments passed to ‘predict‘. Not used in this function.

Value

The function returns a list with:

new_U A model matrix of new_D.

prediction_nng Predictions for new_D. The coefficients of the predictive equation are based on nonnegative garrote estimates.

prediction_lm Predictions for new_D. The coefficients of the predictive equation are estimated via ordinary least squares.

Examples

```
# Cast fatigue experiment
data(cast_fatigue)
X <- cast_fatigue[1:10,1:7]
y <- cast_fatigue[1:10,8]
fit_Hi <- HiGarrote::HiGarrote(X, y)

# make predictions
new_D <- cast_fatigue[11:12,1:7]
pred_Hi <- predict(fit_Hi, new_D)
```

predict.nnGarrote *Make Predictions from a "nnGarrote" Object*

Description

This function makes predictions from a linear model constructed using the important effects selected by nnGarrote.

Usage

```
## S3 method for class 'nnGarrote'
predict(object, new_X, ...)
```

Arguments

object	An nnGarrote object.
new_X	A new input matrix where predictions are to be made.
...	Additional arguments passed to ‘predict‘. Not used in this function.

Value

The function returns a list with:

- `new_U` A model matrix of `new_X`.
- `prediction_nng` Predictions for `new_X`. The coefficients of the predictive equation are based on nonnegative garrote estimates.
- `prediction_lm` Predictions for `new_X`. The coefficients of the predictive equation are estimated via ordinary least squares.

Examples

```
# Generate data
x1 <- runif(100)
x2 <- runif(100)
x3 <- runif(100)
error <- rnorm(100)
X <- data.frame(x1, x2, x3)
U <- model.matrix(~. + x1:x2 + x1:x3 + x2:x3 + I(x1^2) + I(x2^2) + I(x3^2) - 1, X)
U <- data.frame(scale(U))
colnames(U) <- c("x1", "x2", "x3", "x1:x1", "x2:x2", "x3:x3", "x1:x2", "x1:x3", "x2:x3")
y <- 3 + 3*U$x1 + 3*U$`x1:x1` + 3*U$`x1:x2`+ 3*U$`x1:x3` + error

# training and testing set
train_idx <- sample(1:100, 80)
X_train <- X[train_idx,]
y_train <- y[train_idx]
X_test <- X[-train_idx,]
y_test <- y[-train_idx]

# fit nnGarrote
fit_nng <- HiGarrote::nnGarrote(X_train, y_train)

# predict
pred_nng <- predict(fit_nng, X_test)
```

`resin` *Resin Experiment*

Description

Jones and Lanzerath (2021) described a 21-run experiment aimed at creating a special resin for car vents designed to prevent moisture accumulation within a car's housing. The experiment is carried out using a definitive screening design (DSD) with nine continuous factors, labeled A through J, with the exclusion of I.

Usage

```
data(resin)
```

Format

A data frame with 21 rows and 12 columns.

Source

Jones, B. and Lanzerath, M. (2021). A Novel Application of a Definitive Screening Design: A Case Study. *Quality Engineering*, 33(3), 563-569. doi:10.1080/08982112.2021.1892758.

`router_bit` *Router Bit Experiment*

Description

Phadke (1986) described a 32-run experiment aimed at increasing the lifespan of router bits used in a routing process to cut printed wiring boards from a panel. The experiment contains seven two-level factors and two four-level qualitative factors which are denoted by A through J with the exclusion of I.

Usage

```
data(router_bit)
```

Format

A data frame with 32 rows and 10 columns.

Source

Phadke, M. S. (1986). Design Optimization Case Studies. *AT&T Technical Journal*, 65(2), 51-68. doi:10.1002/j.15387305.1986.tb00293.x.

Index