Package 'BSW'

May 6, 2026

Type Package

Title Fitting a Log-Binomial Model Using the Bekhit–Schöpe–Wagenpfeil (BSW) Algorithm

Version 0.1.2

Date 2025-10-06

Description
Implements a modified Newton-type algorithm (BSW algorithm) for solving the maximum likelihood estimation problem in fitting a log-binomial model under linear inequality constraints.

License GPL (>= 3)

Encoding UTF-8

URL https://github.com/UdS-MF-IMBEI/BSW

BugReports https://github.com/UdS-MF-IMBEI/BSW/issues

VignetteBuilder knitr

Depends R (>= 4.0)

Imports Matrix, matrixStats, quadprog, methods, stats, boot, checkmate

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

RoxygenNote 7.3.2

Config/testthat/edition 3

NeedsCompilation no

Author Adam Bekhit [aut],
Jakob Schöpe [aut],
Thomas Wolf [aut] (ORCID: <https://orcid.org/0009-0004-9532-1487>),
Julius Johannes Weise [aut, cre] (ORCID:
<https://orcid.org/0000-0002-0908-5579>),
Stefan Wagenpfeil [aut] (ORCID:
<https://orcid.org/0000-0002-4558-4041>)

Maintainer Julius Johannes Weise <imbei@med-imbei.uni-saarland.de>

Repository CRAN

Date/Publication 2025-10-06 10:50:02 UTC

Contents

`bootbsw` *Estimating bootstrap statistics of* `bsw()`

Description

`bootbsw()` applies nonparametric bootstrapping to an object of class `"bsw"` and computes bias-corrected accelerated confidence intervals (BCa) for the estimated Relative Risk.

Usage

```
bootbsw(object, ci_level = 0.95, R = 1000L, maxit = NULL, conswitch = NULL)
```

Arguments

`object`	An object of the class `"bsw"`.
`ci_level`	A value between 0 and 1 indicating the confidence interval. Provides bias-corrected accelerated bootstrap confidence intervals of the original estimated model parameters of `bsw()`.
`R`	A positive integer greater than or equal to 1000 giving the number of bootstrap replicates.
`maxit`	A positive integer giving the maximum number of iterations in the `bsw()` algorithm. If `NULL` (the default), the value stored in the `bsw` object is passed internally to `bootbsw()`.
`conswitch`	Specifies how the constraint matrix is constructed: **1 (default)** Generates all possible combinations of minimum and maximum values for the predictors (excluding the intercept), resulting in 2^{m-1} constraints. This formulation constrains model predictions within the observed data range, making it suitable for both risk factor identification and prediction (prognosis). **0** Uses the raw design matrix `x` as the constraint matrix, resulting in n constraints. This is primarily suitable for identifying risk factors, but not for prediction tasks, as predictions are not bounded to realistic ranges.

If NULL (the default), the value stored in the bsw object is passed internally to bootbsw().

Value

An object of class "bsw_boot", which is a list containing:

Call_bsw The original call to the bsw() function used to fit the model.

Successful_Bootstraps The number of bootstrap replicates that were completed successfully.

message A character string with a status message indicating how many bootstrap samples succeeded.

Coefficients A matrix with the original estimated model parameters (Orig. Est.), the mean of the bootstrap estimates (Boot. Est.), the standard error of the bootstrap estimates (Boot. SE), the difference between the bootstrap mean and the original estimate (bias), the Risk Difference (equal to the estimate; RD), and the bias-corrected accelerated confidence intervals at the specified level.

Bootstrap_Object An object of class "boot" (from the **boot** package) containing the full bootstrap output, including replicates and metadata. This can be used for further analyses or plotting.

Author(s)

Julius Johannes Weise, Thomas Wolf, Stefan Wagenpfeil

Examples

```
set.seed(123)
x <- rnorm(100, 50, 10)
y <- rbinom(100, 1, exp(-4 + x * 0.04))
fit <- bsw(formula = y ~ x, data = data.frame(y = y, x = x))
result <- bootbsw(fit, ci_level = 0.90)
print(result)
```

---

bsw *Fitting a log-binomial model using the Bekhit-Schöpe-Wagenpfeil (BSW) algorithm*

Description

bsw() fits a log-binomial model using a modified Newton-type algorithm (BSW algorithm) for solving the maximum likelihood estimation problem under linear inequality constraints.

Usage

```
bsw(formula, data, maxit = 200L, conswitch = 1)
```

Arguments

formula	An object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted.
data	A data frame containing the variables in the model.
maxit	A positive integer giving the maximum number of iterations.
conswitch	Specifies how the constraint matrix is constructed:

1 (default) Generates all possible combinations of minimum and maximum values for the predictors (excluding the intercept), resulting in 2^{m-1} constraints. This formulation constrains model predictions within the observed data range, making it suitable for both risk factor identification and prediction (prognosis).

0 Uses the raw design matrix x as the constraint matrix, resulting in n constraints. This is primarily suitable for identifying risk factors, but not for prediction tasks, as predictions are not bounded to realistic ranges.

Value

An object of S4 class "bsw" containing the following slots:

call	An object of class "call".
formula	An object of class "formula".
coefficients	A numeric vector containing the estimated model parameters.
iter	A positive integer indicating the number of iterations.
converged	A logical constant that indicates whether the model has converged.
y	A numerical vector containing the dependent variable of the model.
x	The model matrix.
data	A data frame containing the variables in the model.

Author(s)

Adam Bekhit, Jakob Schöpe

References

Wagenpfeil S (1996) Dynamische Modelle zur Ereignisanalyse. Herbert Utz Verlag Wissenschaft, Munich, Germany

Wagenpfeil S (1991) Implementierung eines SQP-Verfahrens mit dem Algorithmus von Ritter und Best. Diplomarbeit, TUM, Munich, Germany

Examples

```
set.seed(123)
x <- rnorm(100, 50, 10)
y <- rbinom(100, 1, exp(-4 + x * 0.04))
fit <- bsw(formula = y ~ x, conswitch = 1, data = data.frame(y = y, x = x))
summary(fit)
```

bsw-class *S4 Class* "bsw"

Description

S4 Class "bsw"

Slots

`call` An object of class "call".

`formula` An object of class "formula".

`coefficients` A numeric vector containing the estimated model parameters.

`iter` A positive integer indicating the number of iterations.

`converged` A logical constant that indicates whether the model has converged.

`y` A numeric vector containing the dependent variable of the model.

`x` The model matrix.

`data` A data frame containing the variables in the model.

Author(s)

Adam Bekhit, Jakob Schöpe

coef,bsw-method *Extracting the estimated model parameters of* bsw()

Description

For objects of class "bsw", coef() extracts the estimated model parameters of bsw().

Usage

```
## S4 method for signature 'bsw'
coef(object)
```

Arguments

object	An object of class "bsw".

Value

A numeric vector containing the estimated model parameters.

Author(s)

Adam Bekhit, Jakob Schöpe

`confint,bsw-method` *Estimating confidence intervals of the estimated model parameters of* `bsw()`

Description

For objects of class `"bsw"`, `confint()` estimates confidence intervals of the estimated model parameters of `bsw()`.

Usage

```
## S4 method for signature 'bsw'
confint(object, parm, level = 0.95, method = "wald", R = 1000L)
```

Arguments

`object`	An object of class `"bsw"`.
`parm`	A specification of which model parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all model parameters are considered.
`level`	A numeric value that indicates the level of confidence.
`method`	A character giving the estimation method of the confidence intervals (`"bca"` or `"wald"`).
`R`	A positive integer giving the number of bootstrap replicates.

Details

`confint` provides Wald (default) and bias-corrected accelerated bootstrap confidence intervals of the estimated model parameters of `bsw()`.

Value

A matrix with columns giving the lower and upper confidence limits of each estimated model parameter.

Author(s)

Adam Bekhit, Jakob Schöpe

constr *Setting the linear inequality constraints for* bsw()

Description

constr() sets the linear inequality constraints for bsw().

Usage

```
constr(x, version = 1)
```

Arguments

x	A model matrix.
version	switch for constraints

Value

A matrix containing the linear inequality constraints for bsw().

Author(s)

Adam Bekhit, Jakob Schöpe

gradF *Deriving the first derivatives of the log likelihood function of the log-binomial model in* bsw()

Description

gradF() derives the first derivatives of the log likelihood function of the log-binomial model.

Usage

```
gradF(theta, y, x)
```

Arguments

theta	A numeric vector containing the initial values of the model parameters.
y	A numeric vector containing the dependent variable of the model.
x	The model matrix.

Value

A numeric vector containing the first derivatives of the log likelihood function of the log-binomial model.

Author(s)

Adam Bekhit, Jakob Schöpe

`hess` *Deriving the second partial derivatives of the log likelihood function of the log-binomial model in* `bsw()` *(Hessian matrix)*

Description

`hess()` derives the second partial derivatives of the log likelihood function of the log-binomial model.

Usage

```
hess(theta, y, x)
```

Arguments

`theta`	A numeric vector containing the initial values of the model parameters.
`y`	A numeric vector containing the dependent variable of the model.
`x`	The model matrix.

Value

A numeric matrix containing the second partial derivatives of the log likelihood function of the log-binomial model (Hessian matrix).

Author(s)

Adam Bekhit, Jakob Schöpe

`summary,bsw-method` *Summarizing the estimated model parameters of* `bsw()`

Description

For objects of class `"bsw"`, `summary()` summarizes the estimated model parameters of `bsw()`.

Usage

```
## S4 method for signature 'bsw'
summary(object)
```

Arguments

`object`	An object of class `"bsw"`.

Value

A list containing the following elements:

coefficients	A numeric vector containing the estimated model parameters.
std.err	A numeric vector containing the estimated standard errors of the model parameters.
z.value	A numeric vector containing the estimated z test statistic of the model parameters.
p.value	A numeric vector containing the estimated p values of the model parameters.

Author(s)

Adam Bekhit, Jakob Schöpe

---

variable_selection_bsw

Variable Selection (Forward or Backward) for models of BSW()

Description

Performs forward or backward variable selection based on Wald test p-values for models estimated using bsw(). In each step, a new model is fitted using bsw(), and variables are added or removed based on the significance level defined by alpha.

Usage

```
variable_selection_bsw(model, selection = c("backward", "forward"), alpha = 0.157,
                           print_models = FALSE, maxit = NULL, conswitch = NULL)
```

Arguments

model	A model object from bsw() with full data and formula.
selection	Character string, either "backward" or "forward". Determines the direction of model selection. If not specified, backward elimination is performed by default.
alpha	P-value threshold for variable inclusion (forward) or exclusion (backward). Defaults to 0.157, as recommended by Heinze, G., Wallisch, C., & Dunkler, D. (2018).
print_models	Logical; whether to print each model during selection. Defaults to FALSE.
maxit	Maximum number of iterations in the bsw() algorithm. If NULL, defaults to 200L or value from original model call.
conswitch	Specifies how the constraint matrix is constructed: **1 (default)** Generates all possible combinations of minimum and maximum values for the predictors (excluding the intercept), resulting in 2^{m-1} constraints. This formulation constrains model predictions within the observed data range, making it suitable for both risk factor identification and prediction (prognosis).

0 Uses the raw design matrix `x` as the constraint matrix, resulting in n constraints. This is primarily suitable for identifying risk factors, but not for prediction tasks, as predictions are not bounded to realistic ranges.

Value

An object of class `"bsw_selection"`, which is a list containing:

final_model An object of class `bsw` representing the final model selected through the variable selection process.

model_list A list of intermediate `bsw` model objects fitted during each step of the selection.

skipped_models A named list of models that failed to converge and were skipped during the selection. Each entry includes the attempted formula.

final_formula The final model formula used in the last step.

EPV Estimated events-per-variable (EPV) of the final model, used as a diagnostic for model stability.

warnings Optional warning messages about convergence issues or model stability (e.g., low EPV or skipped variables).

Author(s)

Julius Johannes Weise, Thomas Wolf, Stefan Wagenpfeil

References

Heinze, G., Wallisch, C., & Dunkler, D. (2018). Variable selection – A review and recommendations for the practicing statistician. Biometrical Journal, 60(3), 431–449.

Examples

```
set.seed(123)
x1 <- rnorm(500, 50, 10)
x2 <- rnorm(500, 30, 5)
x3 <- rnorm(500, 40, 8)
x4 <- rnorm(500, 60, 12)
logit <- (-4 + x1 * 0.04 + x3 * 0.04)
p <- 1 / (1 + exp(-logit))
y <- rbinom(500, 1, p)
df <- data.frame(y, x1, x2, x3, x4)
fit <- bsw(formula = y ~ x1 + x2 + x3 + x4, data = df)
result <- variable_selection_bsw(fit, selection = "forward", alpha = 0.1)
print(result)
```

Index