Package ‘tram’

September 4, 2020

Title Transformation Models
Version 0.5-1
Date 2020-09-03
Description Formula-based user-interfaces to specific transformation models implemented in package ‘mlt’. Available models include Cox models, some parametric survival models (Weibull, etc.), models for ordered categorical variables, normal and non-normal (Box-Cox type) linear models, and continuous outcome logistic regression (Lohse et al., 2017, <DOI:10.12688/f1000research.12934.1>). The underlying theory is described in Hothorn et al. (2018) <DOI:10.1111/sjos.12291>. An extension to transformation models for clustered data is provided (Hothorn, 2019, <arxiv:1910.09219>). Multivariate conditional transformation models (<arxiv:1906.03151>) can be fitted as well.

Depends R (>= 3.5.0), mlt (>= 1.2-0)
Imports Formula, multcomp, variables (>= 1.0-2), basefun (>= 1.0-5), sandwich, stats, survival, graphics, Matrix, methods
Suggests MASS, TH.data, trtf (>= 0.3-3), mlbench, knitr, quantreg, colorspace, ATR, lme4, merDeriv, SparseGrid, alabama, numDeriv, gridExtra, lattice, latticeExtra, HSAUR3, mvtnorm, ordinalCont, coxme, glmmsr, mlt.docreg, ordinal, asht

VignetteBuilder knitr

URL http://ctm.R-forge.R-project.org
Encoding UTF-8
License GPL-2
NeedsCompilation yes
Author Torsten Hothorn [aut, cre] (<https://orcid.org/0000-0001-8301-0471>),
Luisa Barbanti [aut],
Brian Ripley [ctb],
Bill Venables [ctb],
Douglas M. Bates [ctb],
Nadja Klein [ctb]

Maintainer Torsten Hothorn <Torsten.Hothorn@R-project.org>
Repository CRAN
Date/Publication 2020-09-04 05:10:02 UTC
R topics documented:

Aareg ............................................................... 2
BoxCox ............................................................. 4
Colr ................................................................. 5
Coxph ............................................................... 7
Lehmann ............................................................ 8
Lm ................................................................. 10
mmlt ............................................................... 11
mtram .............................................................. 13
perm_test .......................................................... 15
Polr ................................................................. 17
score_test .......................................................... 18
Survreg ............................................................ 19
tram ............................................................... 21
tram-methods ....................................................... 24

Index 27

Aareg Aalen Additive Hazards Model

Description

Aalen model with fully parameterised hazard function

Usage

Aareg(formula, data, subset, weights, offset, cluster, na.action = na.omit, ...)

Arguments

formula an object of class "formula": a symbolic description of the model structure to be fitted. The details of model specification are given under tram and in the package vignette.

data an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula).

subset an optional vector specifying a subset of observations to be used in the fitting process.

weights an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If present, the weighted log-likelihood is maximised.

offset this can be used to specify an _a priori_ known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.

cluster optional factor with a cluster ID employed for computing clustered covariances.
Aareg

na.action a function which indicates what should happen when the data contain NAs. The
default is set to na.omit.

... additional arguments to tram.

Details

This function allows simultaneous estimation of the cumulative hazard parameterised by a Bern-
stein polynomial. The model is typically fitted with time-varying coefficients, all types of random
 censoring and truncation are allowed.
The responses is bounded (bounds = c(0,Inf)) when specified as a Surv object. Otherwise,
bounds can be specified via ....

Value

An object of class Aareg, with corresponding coef, vcov, logLik, estfun, summary, print, plot
and predict methods.

References

Torsten Hothorn, Lisa Moest, Peter Buehlmann (2018), Most Likely Transformations, Scandinavian

Examples

data("GBSG2", package = "TH.data")
library("survival")
GBSG2$time <- as.numeric(GBSG2$time)
GBSG2$y <- with(GBSG2, Surv(time, cens))

### Cox proportional hazards model
m1 <- Coxph(y ~ horTh, data = GBSG2, support = c(1, 1500))
logLik(m1)

### Aalen additive hazards model with time-varying effects
m2 <- Aareg(y | horTh ~ 1, data = GBSG2, support = c(1, 1500))
logLik(m2)

### compare the hazard functions
nd <- data.frame(horTh = unique(GBSG2$horTh))
col <- 1:2
lty <- 1:2
plot(as.mlt(m1), newdata = nd, type = "hazard",
col = col, lty = lty[1], xlab = "time")
plot(as.mlt(m2), newdata = nd, type = "hazard",
col = col, lty = 2, add = TRUE)
legend("topright", col = rep(col, each = 2),
lty = rep(1:2), bty = "n",
legend = paste(rep("Cox", "Aalen"), 2),
rep(c("Cox", "Aalen"), 2)))
**BoxCox**

*(Similar to) Box-Cox Models*

**Description**

Non-normal linear regression inspired by Box-Cox models

**Usage**

BoxCox(formula, data, subset, weights, offset, cluster, na.action = na.omit, ...)

**Arguments**

- **formula**: an object of class "formula": a symbolic description of the model structure to be fitted. The details of model specification are given under tram and in the package vignette.
- **data**: an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula).
- **subset**: an optional vector specifying a subset of observations to be used in the fitting process.
- **weights**: an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If present, the weighted log-likelihood is maximised.
- **offset**: this can be used to specify an _a priori_ known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.
- **cluster**: optional factor with a cluster ID employed for computing clustered covariances.
- **na.action**: a function which indicates what should happen when the data contain NAs. The default is set to na.omit.
- **...**: additional arguments to tram.

**Details**

A normal model for transformed responses, where the transformation is estimated from the data simultaneously with the regression coefficients. This is similar to a Box-Cox transformation, but the technical details differ. Examples can be found in the package vignette.

The model is defined with a negative shift term. Large values of the linear predictor correspond to large values of the conditional expectation response (but this relationship is potentially nonlinear).

**Value**

An object of class BoxCox, with corresponding coef, vcov, logLik, estfun, summary, print, plot and predict methods.
References


Examples

```r
data("BostonHousing2", package = "mlbench")

lm(cmedv ~ crim + zn + indus + chas + nox + rm + age + dis +
   rad + tax + ptratio + b + lstat, data = BostonHousing2)

BoxCox(cmedv ~ chas + crim + zn + indus + nox +
   rm + age + dis + rad + tax + ptratio + b + lstat,
   data = BostonHousing2)
```

---

**Colr**

*Continuous Outcome Logistic Regression*

Description

A proportional-odds model for continuous variables

Usage

`Colr(formula, data, subset, weights, offset, cluster, na.action = na.omit, ...)`

`PI(logOR, prob)`

Arguments

- `formula`: an object of class "formula": a symbolic description of the model structure to be fitted. The details of model specification are given under `tram` and in the package vignette.
- `data`: an optional data frame, list or environment (or object coercible by `as.data.frame`) containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`.
- `subset`: an optional vector specifying a subset of observations to be used in the fitting process.
- `weights`: an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If present, the weighted log-likelihood is maximised.
- `offset`: this can be used to specify an _a priori_ known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.
- `cluster`: optional factor with a cluster ID employed for computing clustered covariances.
na.action: a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset.

logOR: a log-odds ratio to be converted to a probabilistic index.

prob: a probabilistic index to be converted to a log-odds ratio.

...: additional arguments to tram.

Details
Simultaneous estimation of all possible binary logistic models obtained by dichotomisation of a continuous response. The regression coefficients can be constant allowing for an interpretation as log-odds ratios.

The model is defined with a positive shift term, thus \( \exp(\text{coef}(\)) \) is the multiplicative change of the odds ratio (conditional odds of treatment or for a one unit increase in a numeric variable divided by conditional odds of reference). Large values of the linear predictor correspond to small values of the conditional expectation response (but this relationship is nonlinear).

Value
An object of class Colr, with corresponding coef, vcov, logLik, estfun, summary, print, plot and predict methods.

References


Examples
```r
data("BostonHousing2", package = "mlbench")

lm(cmedv ~ crim + zn + indus + chas + nox + rm + age + dis +
  rad + tax + ptratio + b + lstat, data = BostonHousing2)

Colr(cmedv ~ chas + crim + zn + indus + nox +
  rm + age + dis + rad + tax + ptratio + b + lstat,
  data = BostonHousing2)
```
Cox Proportional Hazards Model

Description

Cox model with fully parameterised baseline hazard function

Usage

Coxph(formula, data, subset, weights, offset, cluster, na.action = na.omit, ...)

Arguments

- formula: an object of class "formula": a symbolic description of the model structure to be fitted. The details of model specification are given under tram and in the package vignette.
- data: an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula).
- subset: an optional vector specifying a subset of observations to be used in the fitting process.
- weights: an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If present, the weighted log-likelihood is maximised.
- offset: this can be used to specify an _a priori_ known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.
- cluster: optional factor with a cluster ID employed for computing clustered covariances.
- na.action: a function which indicates what should happen when the data contain NAs. The default is set to na.omit.
- ...: additional arguments to tram.

Details

The original implementation of Cox models via the partial likelihood, treating the baseline hazard function as a nuisance parameter, is available in coxph. This function allows simultaneous estimation of the log-hazard ratios and the log-cumulative baseline hazard, the latter parameterised by a Bernstein polynomial. The model can be fitted under stratification (time-varying coefficients), all types of random censoring and truncation. An early reference to this parameterisation is McLain and Ghosh (2013).

The responses is bounded (bounds = c(0,Inf)) when specified as a Surv object. Otherwise, bounds can be specified via . . .

Parameters are log-hazard ratios comparing treatment (or a one unit increase in a numeric variable) with a reference.
Lehmann

**Value**

An object of class `Coxph`, with corresponding `coef`, `vcov`, `logLik`, `estfun`, `summary`, `print`, `plot` and `predict` methods.

**References**


**Examples**

```r
data("GBSG2", package = "TH.data")
library("survival")
(m1 <- coxph(Surv(time, cens) ~ horTh, data = GBSG2))
(m2 <- Coxph(Surv(time, cens) ~ horTh, data = GBSG2))

### Wald intervals
confint(m1)
confint(m2)
### profile likelihood interval
confint(profile(m2))
### score interval
confint(score_test(m2))
### permutation score interval
confint(perm_test(m2))
```

**Description**

Non-normal linear regression for Lehmann-alternatives

**Usage**

`Lehmann(formula, data, subset, weights, offset, cluster, na.action = na.omit, ...)`
Arguments

formula: an object of class "formula": a symbolic description of the model structure to be fitted. The details of model specification are given under tram and in the package vignette.

data: an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula).

subset: an optional vector specifying a subset of observations to be used in the fitting process.

weights: an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If present, the weighted log-likelihood is maximised.

offset: this can be used to specify an _a priori_ known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.

cluster: optional factor with a cluster ID employed for computing clustered covariances.

na.action: a function which indicates what should happen when the data contain NAs. The default is set to na.omit.

...: additional arguments to tram.

Details

This transformation model uses the cumulative distribution function for the standard Gumbel maximum extreme value distribution to map the shifted transformation function into probabilities. The exponential of the shift parameter can be interpreted as a Lehmann-alternative or reverse time hazard ratio.

Value

An object of class Lehmann, with corresponding coef, vcov, logLik, estfun, summary, print, plot and predict methods.

References


Examples

data("BostonHousing2", package = "mlbench")

lm(cmedv ~ crim + zn + indus + chas + nox + rm + age + dis + rad + tax + ptratio + b + lstat, data = BostonHousing2)

Lehmann(cmedv ~ chas + crim + zn + indus + nox +
Normal Linear Model

Description

Normal linear model with benefits

Usage

Lm(formula, data, subset, weights, offset, cluster, na.action = na.omit, ...)

Arguments

- **formula**: an object of class "formula": a symbolic description of the model structure to be fitted. The details of model specification are given under `tram` and in the package vignette.
- **data**: an optional data frame, list or environment (or object coercible by `as.data.frame`) containing the variables in the model. If not found in data, the variables are taken from `environment(formula)`.
- **subset**: an optional vector specifying a subset of observations to be used in the fitting process.
- **weights**: an optional vector of weights to be used in the fitting process. Should be `NULL` or a numeric vector. If present, the weighted log-likelihood is maximised.
- **offset**: this can be used to specify an _a priori_ known component to be included in the linear predictor during fitting. This should be `NULL` or a numeric vector of length equal to the number of cases.
- **cluster**: optional factor with a cluster ID employed for computing clustered covariances.
- **na.action**: a function which indicates what should happen when the data contain NAs. The default is set to `na.omit`.
- **...**: additional arguments to `tram`.

Details

A normal linear model with simultaneous estimation of regression coefficients and scale parameter(s). This function also allows for stratum-specific intercepts and variances as well as censoring and truncation in the response.

Note that the scale of the parameters is different from what is reported by `lm`; the discrepancies are explained in the package vignette.

The model is defined with a negative shift term. Large values of the linear predictor correspond to large values of the conditional expectation response.
Value

An object of class `Lm`, with corresponding `coef`, `vcov`, `logLik`, `estfun`, `summary`, `print`, `plot` and `predict` methods.

References


Examples

data("BostonHousing2", package = "mlbench")

\[
\text{lm(cmedv} \sim \text{crim + zn + indus + chas + nox + rm + age + dis + rad + tax + pptratio + b + lstat, data = BostonHousing2)}
\]

\[
\text{Lm(cmedv} \sim \text{chas + crim + zn + indus + nox + rm + age + dis + rad + tax + pptratio + b + lstat, data = BostonHousing2)}
\]

### mmlt

*Multivariate Conditional Transformation Models*

Description

A proof-of-concept implementation of multivariate conditional transformation models

Usage

\[
\text{mmlt(..., formula} \sim 1, \text{data}, \text{theta} = \text{NULL, control.outer} = \text{list(trace} = 1, \text{scale} = 1)\]

Arguments

- `...` marginal transformation models, one for each response
- `formula` a model formula describing a model for the dependency structure via the lambda parameters. The default is set to `~ 1` for constant lambdas.
- `data` a data.frame
- `theta` an optional vector of starting values
- `control.outer` a list controlling `auglag`
- `scale` logical; parameters are not scaled prior to optimisation by default
The function implements multivariate conditional transformation models as described by Klein et al (2019). The response is assumed absolutely continuous at the moment, discrete versions will be added later.

Below is a simple example for an unconditional bivariate distribution. See `demo("undernutrition",package = "tram")` for a conditional three-variate example.

An object of class `mmlt` with `coef` and `predict` methods.


```r
### fit unconditional bivariate distribution of speed and distance to stop
## fit unconditional marginal transformation models
m_speed <- BoxCox(speed ~ 1, data = cars, support = ss <- c(4, 25),
                  add = c(-5, 5))
m_dist <- BoxCox(dist ~ 1, data = cars, support = sd <- c(0, 120),
                  add = c(-5, 5))

## fit multivariate unconditional transformation model
m_speed_dist <- mmlt(m_speed, m_dist, formula = ~ 1, data = cars)

## lambda defining the Cholesky of the precision matrix,
## with standard error
coef(m_speed_dist, newdata = cars[1,], type = "Lambda")
sqrt(vcov(m_speed_dist)["dist.sped.(Intercept)",
                        "dist.sped.(Intercept)"])

## linear correlation, ie Pearson correlation of speed and dist after
## transformation to bivariate normality
(r <- coef(m_speed_dist, newdata = cars[1,], type = "Corr"))

## Spearman’s rho (rank correlation), can be computed easily
## for Gaussian copula as
(r_s <- 6 * asin(r / 2) / pi)

## evaluate joint and marginal densities (needs to be more user-friendly)
nd <- expand.grid(c(nd_s <- mkgrid(m_speed, 100), nd_d <- mkgrid(m_dist, 100)))
nd$sds <- predict(m_speed_dist, newdata = nd, marginal = 1L)
nd$sps <- predict(m_speed_dist, newdata = nd, marginal = 1L,
                  deriv = c("speed" = 1))
nd$hd <- predict(m_speed_dist, newdata = nd, marginal = 2L)
```
nd$hpd <- predict(m_speed_dist, newdata = nd, marginal = 2L,
    deriv = c("dist" = 1))

## joint density
nd$d <- with(nd,
    dnorm(hs) *
    dnorm(coef(m_speed_dist)["dist.sped.(Intercept)"] * hs + hd) *
    hps * hpd)

## compute marginal densities
nd_s <- as.data.frame(nd_s)
nd_s$d <- predict(m_speed_dist, newdata = nd_s, type = "density")
nd_d <- as.data.frame(nd_d)
nd_d$d <- predict(m_speed_dist, newdata = nd_d, marginal = 2L,
    type = "density")

## plot bivariate and marginal distribution
col1 <- rgb(.1, .1, .1, .9)
col2 <- rgb(.1, .1, .1, .5)
w <- c(.8, .2)
layout(matrix(c(2, 1, 4, 3), nrow = 2), width = w, height = rev(w))
par(mai = c(1, 1, 0, 0) * par("mai"))
sp <- unique(nd$speed)
di <- unique(nd$dist)
d <- matrix(nd$d, nrow = length(sp))
contour(sp, di, d, xlab = "Speed (in mph)", ylab = "Distance (in ft)", xlim = ss, ylim = sd,
    col = col1)
points(cars$speed, cars$dist, pch = 19, col = col2)
mai <- par("mai")
par(mai = c(0, 1, 0, 1) * mai)
plot(d ~ speed, data = nd_s, xlim = ss, type = "n", axes = FALSE,
    xlab = "", ylab = "")
polygon(nd_s$speed, nd_s$d, col = col2, border = FALSE)
par(mai = c(1, 0, 1, 0) * mai)
plot(dist ~ d, data = nd_d, ylim = sd, type = "n", axes = FALSE,
    xlab = "", ylab = "")
polygon(nd_d$d, nd_d$dist, col = col2, border = FALSE)

### NOTE: marginal densities are NOT normal, nor is the joint
distribution. The non-normal shape comes from the data-driven
transformation of both variables to joint normality in this model.
Usage

```r
mtram(object, formula, data, standardise = FALSE,
      grd = SparseGrid::createSparseGrid(type = "KPU",
                                          dimension = length(rt$cnms[[1]]), k = 10),
      Hessian = FALSE, ...)
```

Arguments

- **object**: A `tram` object.
- **formula**: A formula specifying the random effects.
- **data**: A data frame.
- **standardise**: Two types of models can be estimated: M1 (with `standardise = FALSE`) corresponds to a marginal distribution function without direct interpretation of the fixed effects, M2 (with `standardise = TRUE`) allows a marginal interpretation of scaled fixed effects as log-odds or log-hazard ratios (depending on `object`). See Hothorn (2019).
- **grd**: A sparse grid used for numerical integration to get the likelihood.
- **Hessian**: A logical, if `TRUE`, the hessian is computed and returned.
- **...**: Additional argument.

Details

A Gaussian copula with a correlation structure obtained from a random intercept or random intercept / random slope model (that is, clustered or longitudinal data can by modelled only) is used to capture the correlations whereas the marginal distributions are described by a transformation model. The methodology is described in Hothorn (2019) and examples are given in the `mtram` package vignette.

This is a proof-of-concept implementation and still highly experimental. Only `coef()` and `logLik()` methods are available at the moment.

Value

An object of class `tram` with `coef()` and `logLik()` methods.

References

Perm_test

Permutation Transformation Tests

Description

P-values for a parameter in a linear transformation model and corresponding confidence intervals obtained from by the permutation principle.

Usage

perm_test(object, ...)
## S3 method for class 'tram'
perm_test(object, parm = names(coef(object)),
    statistic = c("Score", "Likelihood", "Wald"),
    alternative = c("two.sided", "less", "greater"),
    nullvalue = 0, confint = TRUE, level = .95,
    Taylor = FALSE, block_permutation = TRUE, maxsteps = 25, ...)

Arguments

object an object of class tram
parm a vector of names of parameters to be tested. These parameters must be present in object.
statistic a character string specifying the statistic to be permuted. The default Score is the classical permutation test for the residuals of a model excluding the parameter parm. Only available for nullvalue = 0, confidence intervals are not available. Permuting the likelihood or the model coefficients under the nullvalue is highly experimental as are the corresponding confidence intervals.
alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less".
nullvalue a number specifying an optional parameter used to form the null hypothesis.
confint a logical indicating whether a confidence interval should be computed. Score confidence intervals are computed by default. A 1st order Taylor approximation to the Score statistic is used with Taylor = TRUE (in case numerical inversion of the score statistic fails, Wald-type confidence intervals relying from this approximation are returned). For the remaining likelihood and Wald statistics, confidence intervals are highly experimental (and probably not worth looking at).
level the confidence level.
bias_permutation a logical indicating whether stratifying variables shall be interpreted as blocks defining admissible permutations.
Taylor a logical requesting the use of a 1st order Taylor approximation when inverting the score statistic.
maxsteps number of function evaluations when inverting the score statistic for computing confidence intervals.

... additional arguments to independence_test.

Details

Permutation test for one single parameters in the linear predictor of object is computed. This parameters must be present in object. This is somewhat experimental and not recommended for serious practical use (yet!).

Value

An object of class htest or a list thereof. See Coxph for an example.

Examples

```r
## Tritiated Water Diffusion Across Human Chorioamnion
## Hollander and Wolfe (1999, p. 110, Tab. 4.1)
diffusion <- data.frame(
  pd = c(0.80, 0.83, 1.89, 1.04, 1.45, 1.38, 1.91, 1.64, 0.73, 1.46,
     1.15, 0.88, 0.90, 0.74, 1.21),
  age = factor(rep(c("At term", "12-26 Weeks"), c(10, 5)))
)

### plot the two quantile functions
boxplot(pd ~ age, data = diffusion)

### the Wilcoxon rank sum test, with a confidence interval
### for a median shift
wilcox.test(pd ~ age, data = diffusion, conf.int = TRUE, exact = TRUE)

### a corresponding parametric transformation model with a log-odds ratio
### difference parameter, ie a difference on the log-odds scale
md <- Colr(pd ~ age, data = diffusion)

### assess model fit by plotting estimated distribution fcts
agef <- sort(unique(diffusion$age))
col <- c("black", "darkred")
plot(as.mlt(md), newdata = data.frame(age = agef),
     type = "distribution", col = col)
legend("bottomright", col = col, lty = 1, legend = levels(agef),
       bty = "n", pch = 19)

### compare with ECDFs: not too bad (but not good, either)
npfit <- with(diffusion, tapply(pd, age, ecdf))
lines(npfit[[1]], col = col[1])
lines(npfit[[2]], col = col[2])

### Wald confidence interval
confint(md)

### Likelihood confidence interval
```
confint(profile(md))

### Score confidence interval
confint(score_test(md))
confint(score_test(md, Taylor = TRUE))

### exact permutation score test
(pt <- perm_test(md, confint = TRUE, distribution = "exact"))
(pt <- perm_test(md, confint = TRUE, distribution = "exact",
                 Taylor = TRUE))

### compare with probabilistic indices obtained from asht::wmwTest
if (require("asht", warn.conflicts = FALSE)) {
  print(wt2 <- wmwTest(pd ~ I(relevel(age, "At term")),
                       data = diffusion, method = "exact.ce"))
  ### as log-odds ratios
  print(PI(prob = wt2$conf.int))
  print(PI(prob = wt2$estimate))
}

---

### Polr

**Ordered Categorical Regression**

**Description**

Some regression models for ordered categorical responses

**Usage**

Polr(formula, data, subset, weights, offset, cluster, na.action = na.omit,
     method = c("logistic", "probit", "loglog", "cloglog"), ...)

**Arguments**

- **formula**: an object of class "formula": a symbolic description of the model structure to be fitted. The details of model specification are given under `tram` and in the package vignette.
- **data**: an optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in data, the variables are taken from `environment(formula)`.
- **subset**: an optional vector specifying a subset of observations to be used in the fitting process.
- **weights**: an optional vector of weights to be used in the fitting process. Should be `NULL` or a numeric vector. If present, the weighted log-likelihood is maximised.
- **offset**: this can be used to specify an _a priori_ known component to be included in the linear predictor during fitting. This should be `NULL` or a numeric vector of length equal to the number of cases.
cluster optional factor with a cluster ID employed for computing clustered covariances.

na.action a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset.

method a character describing the link function.

... additional arguments to tram.

Details

Models for ordered categorical responses reusing the interface of polr. Allows for stratification, censoring and truncation.

The model is defined with a negative shift term, thus \( \exp(\text{coef}()) \) is the multiplicative change of the odds ratio (conditional odds for reference divided by conditional odds of treatment or for a one unit increase in a numeric variable). Large values of the linear predictor correspond to large values of the conditional expectation response (but this relationship is nonlinear).

Value

An object of class Polr, with corresponding coef, vcov, logLik, estfun, summary, print, plot and predict methods.

References


Examples

```r
data("wine", package = "ordinal")

library("MASS")
polr(rating ~ temp + contact, data = wine)

Polr(rating ~ temp + contact, data = wine)
```

```

score_test

Transformation Score Tests and Confidence Intervals

Description

P-values and confidence intervals for parameters in linear transformation models obtained from by the score test principle
```
Usage

score_test(object, ...)  
## S3 method for class 'tram'

score_test(object, parm = names(coef(object)),
    alternative = c("two.sided", "less", "greater"), nullvalue = 0,
    confint = TRUE, level = .95, Taylor = FALSE, maxsteps = 25, ...)

Arguments

object an object of class tram
parm a vector of names of parameters to be tested. These parameters must be present in object.
alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less".
nullvalue a number specifying an optional parameter used to form the null hypothesis.
confint a logical indicating whether a confidence interval should be computed. Score confidence intervals are computed by default. A 1st order Taylor approximation to the Score statistic is used with Taylor = TRUE (in case numerical inversion of the score statistic fails, Wald confidence intervals relying from this approximation are returned).
level the confidence level.
Taylor a logical requesting the use of a 1st order Taylor approximation when inverting the score statistic.
maxsteps number of function evaluations when inverting the score statistic for computing confidence intervals.
... additional arguments, currently ignored.

Details

Score tests and confidence intervals for the parameters in the linear predictor of object are computed. These parameters must be present in object.

Value

An object of class htest or a list thereof. See Coxph for an example. A corresponding permutation test for parameters in a transformation models is available in perm_test.

Survreg  Parametric Survival Models

Description

Weibull, log-normal, log-logistic and other parametric models (not exclusively) for survival analysis
Usage

Survreg(formula, data, subset, weights, offset, cluster, na.action = na.omit,
  dist = c("weibull", "logistic", "gaussian", "exponential", "rayleigh",
          "loggaussian", "lognormal", "loglogistic"), scale = 0, ...)

Arguments

formula an object of class "formula": a symbolic description of the model structure to be fitted. The details of model specification are given under tram and in the package vignette.
data an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula).
subset an optional vector specifying a subset of observations to be used in the fitting process.
weights an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If present, the weighted log-likelihood is maximised.
offset this can be used to specify an _a priori_ known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.
cluster optional factor with a cluster ID employed for computing clustered covariances.
na.action a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset.
dist character defining the conditional distribution of the (not necessarily positive) response, current choices include Weibull, logistic, normal, exponential, Rayleigh, log-normal (same as log-gaussian), or log-logistic.
scale a fixed value for the scale parameter(s).
... additional arguments to tram.

Details

Parametric survival models reusing the interface of survreg. The parameterisation is, however, a little different, see the package vignette.

The model is defined with a negative shift term. Large values of the linear predictor correspond to large values of the conditional expectation response (but this relationship is nonlinear). Parameters are log-hazard ratios comparing a reference with treatment (or a one unit increase in a numeric variable).

Value

An object of class Survreg, with corresponding coef, vcov, logLik, estfun, summary, print, plot and predict methods.
References

Examples

```r
data("GBSG2", package = "TH.data")
library("survival")
survreg(Surv(time, cens) ~ horTh, data = GBSG2)
Survreg(Surv(time, cens) ~ horTh, data = GBSG2)
```

---

**tram**  
*Stratified Linear Transformation Models*

**Description**
Likelihood-inference for stratified linear transformation models

**Usage**

```r
tram(formula, data, subset, weights, offset, cluster, na.action = na.omit, 
distribution = c("Normal", "Logistic", "MinExtrVal", "MaxExtrVal", "Exponential"), 
thtransformation = c("discrete", "linear", "logarithmic", "smooth"), 
LRtest = TRUE, prob = c(0.1, 0.9), support = NULL, 
bounds = NULL, add = c(0, 0), order = 6, 
negative = TRUE, scale = TRUE, extrapolate = FALSE, 
log_first = FALSE, model_only = FALSE, constraints = NULL, ...)
tram_data(formula, data, subset, weights, offset, cluster, na.action = na.omit)
```

**Arguments**

- **formula**: an object of class "formula": a symbolic description of the model structure to be fitted. The details of model specification are given under Details and in the package vignette.
- **data**: an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula).
- **subset**: an optional vector specifying a subset of observations to be used in the fitting process.
- **weights**: an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If present, the weighted log-likelihood is maximised.
- **distribution**: the distribution to be used. The available options are:"Normal", "Logistic", "MinExtrVal", "MaxExtrVal", and "Exponential".
- **transformation**: the type of transformation to be used. The available options are:"discrete", "linear", "logarithmic", and "smooth".
- **LRtest**: a logical indicating whether a likelihood ratio test should be performed.
- **prob**: a numeric vector specifying the probability levels for two-sided confidence intervals.
- **support**: the support of the distribution.
- **bounds**: the bounds of the distribution.
- **add**: the additive constant.
- **order**: the order of the model.
- **negative**: a logical indicating whether to use the negative log-likelihood.
- **scale**: a logical indicating whether to use the scale parameter.
- **extrapolate**: a logical indicating whether to extrapolate the model.
- **log_first**: a logical indicating whether to take the logarithm of the first variable.
- **model_only**: a logical indicating whether to return only the model.
- **constraints**: a list of constraints.

---

This document provides the necessary information for understanding and utilizing the `tram` package, particularly focusing on the `tram` function which is used for fitting stratified linear transformation models. The examples illustrate how to use the function with the `data` and `survreg` packages, while the `tram` function detailed in the description and usage sections offer insights into its application and functionality.
offset
	his can be used to specify an _a priori_ known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.

cluster

optional factor with a cluster ID employed for computing clustered covariances.

na.action

a function which indicates what should happen when the data contain NAs. The default is set to na.omit.

distribution

character specifying how the transformation function is mapped into probabilities. Available choices include the cumulative distribution functions of the standard normal, the standard logistic and the standard minimum extreme value distribution.

transformation

character specifying the complexity of the response-transformation. For discrete responses, one parameter is assigned to each level (except the last one), for continuous responses linear, log-linear and smooth (parameterised as a Bernstein polynomial) function are implemented.

LRtest

logical specifying if a likelihood-ratio test for the null of all coefficients in the linear predictor being zero shall be performed.

prob

two probabilities giving quantiles of the response defining the support of a smooth Bernstein polynomial (if transformation = "smooth").

support

a vector of two elements; the support of a smooth Bernstein polynomial (if transformation = "smooth").

bounds

an interval defining the bounds of a real sample space.

add

add these values to the support before generating a grid via mkgrid.

order

integer >= 1 defining the order of the Bernstein polynomial (if transformation = "smooth").

negative

logical defining the sign of the linear predictor.

scale

logical defining if variables in the linear predictor shall be scaled. Scaling is internally used for model estimation, rescaled coefficients are reported in model output.

extrapolate

logical defining the behaviour of the Bernstein transformation function outside support. The default FALSE is to extrapolate linearly without requiring the second derivative of the transformation function to be zero at support. If TRUE, this additional constraint is respected.

log_first

logical; if TRUE, a Bernstein polynomial is defined on the log-scale.

model_only

logical, if TRUE the unfitted model is returned.

constraints

additional constraints on regression coefficients in the linear predictor of the form lhs %*% coef(object) >= rhs, where lhs and rhs can be specified as a character (as in glht) or by a matrix lhs (assuming rhs = 0), or as a list containing the two elements lhs and rhs.

... additional arguments.

Details

The model formula is of the form y | s ~ x where y is an at least ordered response variable, s are the variables defining strata and x defines the linear predictor. y ~ x defines a model without strata.
(but response-varying intercept function) and \( y \mid s \sim \theta \) sets up response-varying coefficients for all variables in \( s \).

The two functions `tram` and `tram_data` are not intended to be called directly by users. Instead, functions `Coxph` (Cox proportional hazards models), `Survreg` (parametric survival models), `Polr` (models for ordered categorical responses), `Lm` (normal linear models), `BoxCox` (non-normal linear models) or `Colr` (continuous outcome logistic regression) allow direct access to the corresponding models.

The model class and the specific models implemented in `tram` are explained in the package vignette of package `tram`. The underlying theory of most likely transformations is presented in Hothorn et al. (2018), computational and modelling aspects in more complex situations are discussed by Hothorn (2018).

**Value**

An object of class `tram` inheriting from `mlt`.

**References**


Torsten Hothorn (2018), Most Likely Transformations: The mlt Package, *Journal of Statistical Software*, forthcoming. URL: [https://cran.r-project.org/package=mlt.docreg](https://cran.r-project.org/package=mlt.docreg)

**Examples**

```r
data("BostonHousing2", package = "mlbench")

### unconstrained regression coefficients
### BoxCox calls tram internally
m1 <- BoxCox(cmedv ~ chas + crim + zn + indus + nox +
             rm + age + dis + rad + tax + ptratio + b + lstat,
             data = BostonHousing2)

### now with two constraints on regression coefficients
m2 <- BoxCox(cmedv ~ chas + crim + zn + indus + nox +
             rm + age + dis + rad + tax + ptratio + b + lstat,
             data = BostonHousing2,
             constraints = c("crim >= 0", "chas1 + rm >= 1.5"))
coef(m1)
coef(m2)

K <- matrix(0, nrow = 2, ncol = length(coef(m2)))
colnames(K) <- names(coef(m2))
K[1, "crim"] <- 1
K[2, c("chas1", "rm")]<- 1
m3 <- BoxCox(cmedv ~ chas + crim + zn + indus + nox +
             rm + age + dis + rad + tax + ptratio + b + lstat,
             data = BostonHousing2,
             constraints = list(K, c(0, 1.5)))
```
Methods for Stratified Linear Transformation Models

Description

Methods for objects inheriting from class tram

Usage

```r
## S3 method for class 'tram'
as.mlt(object)
## S3 method for class 'tram'
model.frame(formula, ...)
## S3 method for class 'tram'
model.matrix(object, data = object$data, with_basline = FALSE, ...)
## S3 method for class 'tram'
coef(object, with_basline = FALSE, ...)
## S3 method for class 'Lm'
coef(object, as.lm = FALSE, ...)
## S3 method for class 'Survreg'
coef(object, as.survreg = FALSE, ...)
## S3 method for class 'tram'
vcov(object, with_basline = FALSE, complete = FALSE, ...)
## S3 method for class 'tram'
logLik(object, parm = coef(as.mlt(object), fixed = FALSE), ...)
## S3 method for class 'tram'
estfun(object, parm = coef(as.mlt(object), fixed = FALSE), ...)
## S3 method for class 'tram'
predict(object, newdata = model.frame(object),
    type = c("lp", "trafo", "distribution", "survivor", "density",
      "logdensity", "hazard", "loghazard", "cumhazard", "quantile"),
    ...)
## S3 method for class 'tram'
plot(x, newdata = model.frame(x),
     which = c("QQ-PIT", "baseline only", "distribution"),
     confidence = c("none", "interval", "band"), level = 0.95,
     K = 50, cheat = K, col = "black", fill = "lightgrey", lwd = 1, ...)
```

Arguments

- `object`, `formula`, `x`
  - A fitted stratified linear transformation model inheriting from class tram.
- `data`
  - An optional data frame.
with_baseline logical, if TRUE all model parameters are returned, otherwise parameters describing the baseline transformation are ignored.

as.lm logical, return parameters in the \texttt{lm} parameterisation if TRUE.

as.survreg logical, return parameters in the \texttt{survreg} parameterisation in TRUE.

parm model parameters, including baseline parameters.

complete currently ignored

newdata an optional data frame of new observations.

type type of prediction, current options include linear predictors ("lp", of x variables in the formula \(y \mid s \sim x\)), transformation functions ("trafo") or distribution functions on the scale of the cdf ("distribution"), survivor function, density function, log-density function, hazard function, log-hazard function, cumulative hazard function or quantile function.

which type of plot, either a QQ plot of the probability-integral transformed observations ("QQ-PIT"), of the baseline transformation of the whole distribution.

confidence type of uncertainty assessment.

level confidence level.

K number of grid points in the response, see \texttt{plot.ctm}.

cheat reduced number of grid points for the computation of confidence bands, see \texttt{confband}.

col line color.

fill fill color.

lwd line width.

... additional arguments to the underlying methods for class \texttt{mlt}, see \texttt{mlt-methods}.

Details

\texttt{coef} can be used to get (and set) model parameters, \texttt{logLik} evaluates the log-likelihood (also for parameters other than the maximum likelihood estimate); \texttt{vcov} returns the estimated variance-covariance matrix (possibly taking \texttt{cluster} into account) and \texttt{estfun} gives the score contribution by each observation. \texttt{predict} and \texttt{plot} can be used to inspect the model on different scales.

References


See Also

\texttt{mlt-methods.plot.ctm}
Examples

data("BostonHousing2", package = "mlbench")

### fit non-normal Box-Cox type linear model with two
### baseline functions (for houses near and off Charles River)
BC_BH_2 <- BoxCox(cmedv | 0 + chas ~ crim + zn + indus + nox +
                   rm + age + dis + rad + tax + ptratio + b + lstat,
                   data = BostonHousing2)

logLik(BC_BH_2)

### classical likelihood inference
summary(BC_BH_2)

### coefficients of the linear predictor
coef(BC_BH_2)

### plot linear predictor (mean of _transformed_ response)
### vs. observed values
plot(predict(BC_BH_2, type = "lp"), BostonHousing2$cmedv)

### all coefficients
coef(BC_BH_2, with_baseline = TRUE)

### compute predicted median along with 10% and 90% quantile for the first
### observations
predict(BC_BH_2, newdata = BostonHousing2[1:3,], type = "quantile",
        prob = c(.1, .5, .9))

### plot the predicted density for these observations
plot(BC_BH_2, newdata = BostonHousing2[1:3, -1],
     which = "distribution", type = "density", K = 1000)

### evaluate the two baseline transformations, with confidence intervals
nd <- model.frame(BC_BH_2)[1:2, -1]
nd$chas <- factor(c("0", "1"))

library("colorspace")
col <- diverge_hcl(2, h = c(246, 40), c = 96, l = c(65, 90))
fill <- diverge_hcl(2, h = c(246, 40), c = 96, l = c(65, 90), alpha = .3)
plot(BC_BH_2, which = "baseline only", newdata = nd, col = col,
     confidence = "interval", fill = fill, lwd = 2,
     xlab = "Median Value", ylab = expression(h[Y]))
legend("bottomright", lty = 1, col = col,
       title = "Near Charles River", legend = c("no", "yes"), bty = "n")
Index

* models
  Aareg, 2
  BoxCox, 4
  Colr, 5
  Coxph, 7
  Lehmann, 8
  Lm, 10
  mmlt, 11
  mtram, 13
  Polr, 17
  Survreg, 19
  tram, 21

* regression
  Aareg, 2
  BoxCox, 4
  Colr, 5
  Coxph, 7
  Lehmann, 8
  Lm, 10
  Polr, 17
  Survreg, 19
  tram, 21

* smooth
  Aareg, 2
  BoxCox, 4
  Colr, 5
  Coxph, 7
  Lehmann, 8
  tram, 21

* survival
  Aareg, 2
  Coxph, 7
  Survreg, 19
  tram, 21

Aareg, 2
as.mlt.tram (tram-methods), 24
auglag, 11

BoxCox, 4, 23

coef.Lm (tram-methods), 24
coeff.Survreg (tram-methods), 24
coeff.tram (tram-methods), 24
Colr, 5, 23
cnb band, 25
Coxph, 7, 16, 19, 23
coxph, 7
estfun.tram (tram-methods), 24
glht, 22
independence_test, 16
Lehmann, 8
Lm, 10, 23
lm, 10, 25
logLik.tram (tram-methods), 24
mkgrid, 22
mmlt, 11
model.frame.tram (tram-methods), 24
model.matrix.tram (tram-methods), 24
mtram, 13
perm_test, 15, 19
PI (Colr), 5
plot.ctm, 25
plot.tram (tram-methods), 24
Polr, 17, 23
polr, 18
predict.tram (tram-methods), 24
score_test, 18
Survreg, 19, 23
survreg, 20, 25
tram, 2–7, 9, 10, 15, 17–20, 21
tram-methods, 24
tram_data (tram), 21
vcov.tram (tram-methods), 24