Package ‘OrdFacReg’

October 12, 2022

Type  Package
Title  Least Squares, Logistic, and Cox-Regression with Ordered Predicators
Version  1.0.6
Date  2015-07-03
Author  Kaspar Rufibach
Maintainer  Kaspar Rufibach <kaspar.rufibach@gmail.com>
Depends  survival, eha, MASS
Imports  stats
Description  In biomedical studies, researchers are often interested in assessing the association between one or more ordinal explanatory variables and an outcome variable, at the same time adjusting for covariates of any type. The outcome variable may be continuous, binary, or represent censored survival times. In the absence of a precise knowledge of the response function, using monotonicity constraints on the ordinal variables improves efficiency in estimating parameters, especially when sample sizes are small. This package implements an active set algorithm that efficiently computes such estimators.
License  GPL (>= 2)
LazyLoad  yes
URL  http://www.kasparrufibach.ch
NeedsCompilation  no
Repository  CRAN
Date/Publication  2015-07-04 15:27:12

R topics documented:

OrdFacReg-package .............................................. 2
internal ...................................................... 3
ordFacReg ..................................................... 4
ordFacRegCox ................................................. 8
prepareData .................................................. 11

Index ......................................................... 13
Description
In biomedical studies, researchers are often interested in assessing the association between one or more ordinal explanatory variables and an outcome variable, at the same time adjusting for covariates of any type. The outcome variable may be continuous, binary, or represent censored survival times. In the absence of a precise knowledge of the response function, using monotonicity constraints on the ordinal variables improves efficiency in estimating parameters, especially when sample sizes are small. This package implements an active set algorithm that efficiently computes such estimators.

Details

Package: OrdFacReg
Type: Package
Version: 1.0.6
Date: 2015-07-03
License: GPL (>=2)
LazyLoad: yes

Use this package to get estimates in least squares, logistic, or Cox-regression where coefficients corresponding to dummy variables of ordered factors are estimated to be in non-decreasing order and at least 0. The package offers an active set algorithm implemented in the functions ordFacReg for least squares and logistic regression and ordFacRegCox for Cox-regression.

Author(s)
Kaspar Rufibach (maintainer)
<kaspar.rufibach@gmail.com>
http://www.kasparrufibach.ch

References

See Also
Examples are given in the help files of the functions ordFacReg and ordFacRegCox.
Description

Internal functions for ordered factor regression functions.

Details

These functions are not intended to be called by users directly.

- **Abeta** Function $A(\beta)$ in Rufibach (2010) that collects the indices of the inequalities violated by $\beta$.
- **constraintMats** Function that computes the matrices $B$ (collects the basis vectors given in Theorem 3.1 of Duembgen et al. (2007)) and $V$ (collects the vectors $v_i$ that make up the cone $K$ in Section 3.1 of Duembgen et al. (2007)).
- **coxDeriv** Computes gradient of (pseudo-)log-likelihood function in Cox-regression.
- **coxLoglik** Computes value of (pseudo-)log-likelihood function in Cox-regression.
- **coxSubspace** Computes maximizer on subspace, denoted by $\tilde{\psi}(A)$ in Table 1 of Duembgen et al. (2007).
- **dummy** Generates a matrix of dummy variables corresponding to the levels of the inputed factor. The dummy variable corresponding to the lowest level of the factor is omitted.
- **expandBeta** After computation of $\beta$ on subspace “blow up” this vector again to original dimension.
- **indexDummy** Computes column numbers of the dummy variables of the ordered factor(s).
- **lmLSE** Computes value of least squares criterion and least squares estimate.
- **lmSS** Computes value of least squares criterion and its gradient.
- **logRegDeriv** Gradient of log-likelihood function in logistic regression.
- **logRegLoglik** Computes value of log-likelihood function in logistic regression.
- **logRegSubspace** Computes maximizer on subspace, denoted by $\tilde{\psi}(A)$ in Table 1 of Duembgen et al. (2007).
- **LSEsubspace** Computes maximizer on subspace, denoted by $\tilde{\psi}(A)$ in Table 1 of Duembgen et al. (2007).
- **maxStep** Computes maximal permissible steplength, denoted by $t$ in Table 1 in Duembgen et al. (2007).
- **phi_jl** Function $\phi$ in Rufibach (2010) that maps the original indices $(i, j)$ to the inequality index $i$.
- **setminus** Remove elements in vector $B$ from vector $A$.
- **shrinkBeta** Collapses $\beta$ according to the active constraints specified by the set $A$. 
Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@gmail.com>
http://www.kasparrufibach.ch

References

densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available

Rufibach, K. (2010). An Active Set Algorithm to Estimate Parameters in Generalized Linear Models

See Also

All these functions are used by the ordered factor computation functions ordFacReg and ordFacRegCox.

---

**ordFacReg**

*Compute least squares or logistic regression for ordered predictors*

**Description**

This function computes estimates in least squares or logistic regression where coefficients corre-
sponding to dummy variables of ordered factors are estimated to be in non-decreasing order and at
least 0. An active set algorithm as described in Duembgen et al. (2007) is used.

**Usage**

```r
ordFacReg(D, Z, fact, ordfact, ordering = NA, type = c("LS", "logreg"),
intercept = TRUE, display = 0, eps = 0)
```

**Arguments**

- `D` Response vector, either in $R^n$ (least squares) or in $\{0, 1\}^n$ (logistic).
- `Z` Matrix of predictors. Factors are coded with levels from 1 to $j$.
- `fact` Specify columns in $Z$ that correspond to unordered factors.
- `ordfact` Specify columns in $Z$ that correspond to ordered factors.
- `ordering` Vector of the same length as `ordfact`. Specifies ordering of ordered factors: "i" means
  that the coefficients of the corresponding ordered factor are estimated in non-decreasing order and "d"
  means non-increasing order. See the examples below for details.
- `type` Specify type of response variable.
- `intercept` If TRUE, an intercept (= column of all 1’s) is added to the design matrix.
- `display` If display == 1 progress of the algorithm is output.
- `eps` Quantity to which the criterion in the Basic Procedure 2 in Duembgen et al. (2007) is compared.
Details

For a detailed description of the problem and the algorithm we refer to Rufibach (2010).

Value

L Value of the criterion function at the maximum.
beta Computed regression coefficients.
A Set A of active constraints.
design.matrix Design matrix that was generated.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@gmail.com>
http://www.kasparrufibach.ch

References


See Also

ordFacRegCox computes estimates for Cox-regression.

Examples

```r
## To illustrate least squares estimation, we generate the same data
## that was used in Rufibach (2010), Table 1.
## --------------------------------------------------------
## initialization
## --------------------------------------------------------
n <- 200
Z <- NULL
intercept <- FALSE

## quantitative variables
## --------------------------------------------------------
n.q <- 3
set.seed(14012009)
if (n.q > 0){for (i in 1:n.q){Z <- cbind(Z, rnorm(n, mean = 1, sd = 2))}}
```
ordFacReg

# --------------------------------------------------------
# unordered factors
# --------------------------------------------------------
un.levels <- 3
for (i in 1:length(un.levels)){
  Z <- cbind(Z, sample(rep(1:un.levels[i],
                      each = ceiling(n / un.levels)))[1:n])
}
fact <- n.q + 1:length(un.levels)

# --------------------------------------------------------
# ordered factors
# --------------------------------------------------------
levels <- 8
for (i in 1:length(un.levels)){
  Z <- cbind(Z, sample(rep(1:levels[i],
                      each = ceiling(n / levels)))[1:n])
  ordfact <- n.q + length(un.levels) + 1:length(levels)
}

# --------------------------------------------------------
# generate data matrices
# --------------------------------------------------------
Y <- prepareData(Z, fact, ordfact, ordering = NA, intercept)$Y

# --------------------------------------------------------
# generate response
# --------------------------------------------------------
D <- apply(Y * matrix(c(rep(c(2, -3, 0), each = n), rep(c(1, 1), each = n),
                      rep(c(0, 2, 2, 5, 5), each = n)), ncol = ncol(Y)), 1, sum) +
    rnorm(n, mean = 0, sd = 4)

# --------------------------------------------------------
# compute estimates
# --------------------------------------------------------
res1 <- lmLSE(D, Y)
res2 <- ordFacReg(D, Z, fact, ordfact, ordering = "i", type = "LS", intercept,
                  display = 1, eps = 0)
b1 <- res1$beta
g1 <- lmSS(b1, D, Y)$dL
b2 <- res2$beta
g2 <- lmSS(b2, D, Y)$dL
Ls <- c(lmSS(b1, D, Y)$L, lmSS(b2, D, Y)$L)
names(Ls) <- c("LSE", "ordFact")
disp <- cbind(1:length(b1), round(cbind(b1, g1, cumsum(g1)), 4),
               round(cbind(b2, g2, cumsum(g2)), 4))

# --------------------------------------------------------
# display results
# --------------------------------------------------------
disp
Ls

# --------------------------------------------------------
# Artificial data is used to illustrate logistic regression.
# --------------------------------------------------------
# initialization
## quantitative variables
## unordered factors
## ordered factors
## generate response
## generate design matrix
## compute estimates

```r
set.seed(1977)
n <- 500
Z <- NULL
intercept <- FALSE

n.q <- 2
if (n.q > 0){for (i in 1:n.q){Z <- cbind(Z, rnorm(n, rgamma(2, 2, 1)))}}

un.levels <- c(8, 2)
for (i in 1:length(un.levels)){Z <- cbind(Z, sample(round(runif(n, 0,
un.levels[i] - 1)) + 1))}
fact <- n.q + 1:length(un.levels)

levels <- c(2, 4, 10)
for (i in 1:length(levels)){Z <- cbind(Z, sample(round(runif(n, 0,
levels[i] - 1)) + 1))}
ordfact <- n.q + length(un.levels) + 1:length(levels)

D <- sample(c(rep(0, n/2), rep(1, n/2)))

Y <- prepareData(Z, fact, ordfact, ordering = NA, intercept)$Y

res1 <- matrix(glm.fit(Y, D, family = binomial(link = logit))$coefficients, ncol = 1)
res2 <- ordFacReg(D, Z, fact, ordfact, ordering = NA, type = "logreg",
intercept = intercept, display = 1, eps = 0)
b1 <- res1
g1 <- logRegDeriv(b1, D, Y)$dL
b2 <- res2$beta
g2 <- logRegDeriv(b2, D, Y)$dL
Ls <- unlist(c(logRegLoglik(res1, D, Y), res2$L))
names(Ls) <- c("MLE", "ordFact")
```

ordFacRegCox

Compute Cox-regression for ordered predictors

Description

This function computes estimates in Cox-regression where coefficients corresponding to dummy variables of ordered factors are estimated to be in non-decreasing order and at least 0. An active set algorithm as described in Duembgen et al. (2007) is used.

Usage

ordFacRegCox(ttf, tf, Z, fact, ordfact, ordering = NA, intercept = TRUE, display = 0, eps = 0)

Arguments

ttf Survival times.
tf Censoring indicator (1 = event, 0 = censored).
Z Matrix of predictors. Factors are coded with levels from 1 to j.
fact Specify columns in Z that correspond to unordered factors.
ordfact Specify columns in Z that correspond to ordered factors.
ordFacRegCox

ordering Vector of the same length as ordfact. Specifies ordering of ordered factors: "i" means that the coefficients of the corresponding ordered factor are estimated in non-decreasing order and "d" means non-increasing order. See the examples in ordFacReg for details.

intercept If TRUE, an intercept (= column of all 1’s) is added to the design matrix.

display If display == 1 progress of the algorithm is output.

eps Quantity to which the criterion in the Basic Procedure 2 in Duembgen et al. (2007) is compared.

Details

For a detailed description of the problem and the algorithm we refer to Rufibach (2010).

Value

L Value of the criterion function at the maximum.

beta Computed regression coefficients.

A Set A of active constraints.

design.matrix Design matrix that was generated.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@gmail.com>
http://www.kasparrufibach.ch

References


See Also

ordFacReg computes estimates for least squares and logistic regression.

Examples

```r
## ==============================================================
## Artificial data is used to illustrate Cox-regression.
## ==============================================================
## --------------------------------------------------------
## initialization
## --------------------------------------------------------
```
set.seed(1977)
n <- 500
Z <- NULL
intercept <- FALSE

## --------------------------------------------------------
## quantitative variables
## --------------------------------------------------------
n.q <- 2
if (n.q > 0){for (i in 1:n.q){Z <- cbind(Z, rnorm(n, rgamma(2, 2, 1)))}}

## --------------------------------------------------------
## unordered factors
## --------------------------------------------------------
un.levels <- c(8, 2)
for (i in 1:length(un.levels)){Z <- cbind(Z, sample(round(runif(n, 0, un.levels[i] - 1)) + 1))}

fact <- n.q + 1:length(un.levels)

## --------------------------------------------------------
## ordered factors
## --------------------------------------------------------
levels <- c(4, 5, 10)
for (i in 1:length(levels)){Z <- cbind(Z, sample(round(runif(n, 0, levels[i] - 1)) + 1))}

ordfact <- n.q + length(un.levels) + 1:length(levels)

## --------------------------------------------------------
## generate response
## --------------------------------------------------------
ttf <- rexp(n)
tf <- round(runif(n))

## --------------------------------------------------------
## generate design matrix
## --------------------------------------------------------
Y <- prepareData(Z, fact, ordfact, ordering = NA, intercept)

## --------------------------------------------------------
## compute estimates
## --------------------------------------------------------
res1 <- eha::coxreg.fit(Y, Surv(ttf, tf), max.survs = length(tf), strats = rep(1, length(tf))$coefficients
res2 <- ordFacRegCox(ttf, tf, Z, fact, ordfact, ordering = NA, intercept = intercept, display = 1, eps = 0)

b1 <- matrix(res1, ncol = 1)
g1 <- coxDeriv(b1, ttf, tf, Y)$dL
b2 <- res2$beta
g2 <- coxDeriv(b2, ttf, tf, Y)$dL

Ls <- c(coxLoglik(b1, ttf, tf, Y)$L, res2$L)

names(Ls) <- c("MLE", "ordFact")
disp <- cbind(1:length(b1), round(cbind(b1, g1, cumsum(g1)), 4), 
round(cbind(b2, g2, cumsum(g2)), 4))
## display results

Prepare input data to be used in active set algorithm

### Description

This function takes a matrix consisting of quantitative variables, unordered, and ordered factors and generates the corresponding matrix of dummy variables, and some further quantities that are used by the active set algorithm in `ordFacReg` and `ordFacRegCox`.

### Usage

```r
prepareData(Z, fact = NA, ordfact, ordering = NA, intercept = TRUE)
```

### Arguments

- **Z**: Matrix with quantitative variables in the first \( c \) columns, unordered factors in the next columns, and finally unordered factors. The latter two need to have levels from 1 to \( j \).
- **fact**: Specify columns in \( Z \) that correspond to unordered factors.
- **ordfact**: Specify columns in \( Z \) that correspond to ordered factors.
- **ordering**: Vector of the same length as `ordfact`. Specifies ordering of ordered factors: "i" means that the coefficients of the corresponding ordered factor are estimated in non-decreasing order and "d" means non-increasing order. See the examples in `ordFacReg` for details.
- **intercept**: If `TRUE`, an intercept (= column of all 1’s) is added to the design matrix.

### Value

Quantities that are used by the active set algorithm. The names of the objects roughly correspond to those in Rufibach (2010).

### Author(s)

Kaspar Rufibach (maintainer)

<kaspar.rufibach@gmail.com>

[http://www.kasparrufibach.ch](http://www.kasparrufibach.ch)

### References

See Also

This function is used by the ordered factor computation functions \texttt{ordFacReg} and \texttt{ordFacRegCox}.
Index

* ordered explanatory variable
  OrdFacReg-package, 2

* ordered factor
  OrdFacReg-package, 2

* ordered predictor
  OrdFacReg-package, 2

* regression
  internal, 3
  ordFacReg, 4
  OrdFacReg-package, 2
  ordFacRegCox, 8

Abeta, 3
Abeta (internal), 3

constraintMats, 3
constraintMats (internal), 3
coxDeriv, 3
coxDeriv (internal), 3
coxLoglik, 3
coxLoglik (internal), 3
coxSubspace, 3
coxSubspace (internal), 3

dummy, 3
dummy (internal), 3
expandBeta, 3
expandBeta (internal), 3

indexDummy, 3
indexDummy (internal), 3
internal, 3

lmLSE, 3
lmLSE (internal), 3
lmSS, 3
lmSS (internal), 3
logRegDeriv, 3
logRegDeriv (internal), 3
logRegLoglik, 3

logRegLoglik (internal), 3
logRegMLE (internal), 3
logRegSubspace, 3
logRegSubspace (internal), 3
LSEsubspace, 3
LSEsubspace (internal), 3

maxStep, 3
maxStep (internal), 3

OrdFacReg (OrdFacReg-package), 2
ordFacReg, 2, 4, 4, 9, 11, 12
OrdFacReg-package, 2
ordFacRegCox, 2, 4, 5, 8, 11, 12

phi_jl, 3
phi_jl (internal), 3
prepareData, 11

setminus, 3
setminus (internal), 3
shrinkBeta, 3
shrinkBeta (internal), 3