

Package ‘intcox’

June 7, 2009

Title Iterated Convex Minorant Algorithm for interval censored event data

Version 0.9.2

Date 2009-06-05

Author Volkmar Henschel, Christiane Heiss, Ulrich Mansmann
<mansmann@ibe.med.uni-muenchen.de>

Description Implementation of ICM-Algorithm by Wei Pan, J. Comp. & Gr. Stat. 78: 109-120, 1999
Algorithm for the Cox proportional hazard model for interval censored data

Maintainer Volkmar Henschel<vhenschel@gmx.de>

License GPL (>= 2)

Depends R (>= 1.7.0), survival

Repository CRAN

Date/Publication 2009-06-07 13:08:31

R topics documented:

AA.data	1
intcox	2
intcox.example	4
Index	6

 AA.data

Shrinkage of aneurisms

Description

Data on the shrinkage of aneurisms associated with cerebral arteriovenous malformations (cAVM) after treatment. The time to a shrinkage of the aneurism to below 50% of the baseline volume was of interest. Several patients had multiple aneurisms.

Usage

```
data(AA.data)
```

Format

A data frame with 149 observations on the following 7 variables.

obs.t one random inspection time, current status

z censoring variable

mo the degree of cAMV occlusion by embolization (dichotomized at 50%)

lok the location of the aneurism, whether at the midline arteries or at other afferent cerebral arteries

gr The single aneurisms are not independent because aneurisms within a patient may shrink in the same way. Multiple aneurisms were observed per patient. This clustering of aneurisms is indicated by this grouping variable.

Source

H.~J. Meisel, U.~Mansmann, H.~Alvarez, G.~Rodesch, M.~Brock, and P.~Lasjaunias. Cerebral arteriovenous malformations and associated aneurysms: Analysis of 305 cases from a series of 662 patients. *Neurosurgery*, 46:793–802, 2000.

Examples

```
data(AA.data)
```

 intcox

Cox proportional hazards model for interval censored data

Description

Intcox fits the Cox proportional hazards model for interval censored data by the Iterative Convex Minorant Algorithm (ICM)

Usage

```
intcox(formula = formula(data), data = parent.frame(), subset, na.action, x = FALSE)
```

Arguments

<code>formula</code>	a formula object, with the response on the left of a <code>~</code> operator, and the terms on the right. The response must be a survival object of type <code>"interval2"</code> as returned by the <code>Surv</code> function.
<code>data</code>	a <code>data.frame</code> in which to interpret the variables named in the <code>formula</code> , or in the <code>subset</code> argument.
<code>subset</code>	expression saying that only a subset of the rows of the data should be used in the fit.
<code>na.action</code>	a missing-data filter function, applied to the <code>model.frame</code> , after any <code>subset</code> argument has been used. Default is <code>options()\$na.action</code> .
<code>x</code>	Return the design matrix in the model object?
<code>y</code>	Return the response in the model object?
<code>epsilon</code>	convergence treshold. Iteration will continue until the relative change in the log-likelihood is less then epsilon. Default is <code>.0001</code> .
<code>itermax</code>	maximum number of iteration
<code>no.warnings</code>	logical value indicating how to handle warnings. If <code>TRUE</code> , warnings will be displayed. Default is <code>FALSE</code> .

Details

With this package the Cox proportional hazards model can be applied for interval censored data. It tries to maximise the log-likelihood by a simultaneous improvement of the coefficients and the cumulative hazard function in the gradient direction weighted by the main diagonal elements of the negative Hessian matrix.

Value

an object of class `"coxph"`. See `coxph.object` for details. Not all features are realised. Additionally there are given

<code>lambda0</code>	estimated baseline hazard
<code>time.point</code>	corresponding time points for the steps
<code>likeli.vec</code>	vector of the estimated loglik of each step
<code>termination</code>	indicator for the reason of termination, 1 - algorithm converged 2 - no improvement of likelihood possible, the iteration number is shown 3 - algorithm did not converge - maximum number of iteration reached 4 - inside precondition(s) are not fulfilled at this iteration

Author(s)

Ch. Heiss, V. Henschel, U. Mansmann

References

Wei Pan, (1999), Extending the Iterative Convex Minorant Algorithm to the Cox Model for Interval-Censored Data, Journal of Computational & Graphical Statistics, vol. 8, pp. 109-120

See Also

[coxph](#), [Surv](#)

Examples

```
data(intcox.example)
intcox(Surv(left,right,type="interval2")~x.1+x.2+x.3+x.4,data=intcox.example)
```

intcox.example *Example set for interval censored data with four covariates*

Description

Weibull distributed survival times with four covariates interval censored by a fixed grid

Usage

```
data(intcox.example)
```

Format

A data frame with 200 observations on the following 6 variables.

left left interval ends

right right interval ends

x.1 binary covariate

x.2 binary covariate

x.3 uniformly on $[-1, 1]$ distributed covariate

x.4 standard normally distributed covariate

Details

The example dataset consists of 200 Weibull distributed random variables with $shape=0.75$, while $scale$ is derived from $(1/\lambda)^{(1/shape)}$ with $\lambda = \exp \beta_0 + \beta'X$ where $\beta = 0.5, -0.5, 0.5, 0.5'$ and design matrix X which is formed by the four covariates.

Furthermore, an interval was generated where the left interval end is zero and the right interval end is defined by maximum value ($T.max$) of the which is got by the 0.9-quantile of a Weibull r.v. with $shape=0.75$ and $scale=median(scale)$.

If the value of event time $\geq T.max$ then the event time is right censored ($left=T.max$ and $right=NA$). Otherwise the interval $[0, T.max]$ was randomly divided into subintervals ($grid=10$) in order to determine the corresponding interval ends for each event time which is not right censored.

The generating code is given below.

Examples

```
## Not run:
sim.weibull.intcox.rfc <-function (N=200,beta.0=0.1,beta.cov=c(0.5,-0.5,0.5,0.5),alpha=0.75,
{
  x.design<-cbind(rbinom(N,1,p.cov[1]),rbinom(N,1,p.cov[2]),runif(N,-1,1),rnorm(N,0,1))
  colnames(x.design)<-paste("x.",1:4,sep="")
  lambda<-exp(beta.0+x.design%*%matrix(beta.cov,ncol=1))
  scale<-(1/lambda)^(1/alpha)
  t.true<-rweibull(N,alpha,scale)
  T.max<-max(qweibull(0.9,alpha,median(scale)))
  t.left<-NULL
  t.right<-NULL
  for (i in 1:N) {
    tt<-unique(c(0,sort(runif(grid,0,T.max)),T.max))
    if (t.true[i]>=T.max) {
      x.left<-T.max
      x.right<-NA
    } else {
      x.left<-max(tt[t.true[i]>tt])
      x.right<-min(tt[t.true[i]<tt])
    }
    t.left<-c(t.left,x.left)
    t.right<-c(t.right,x.right)
  }
  return(data.frame(ID=1:N,left=t.left,right=t.right,x.design))
}
## End(Not run)

data(intcox.example)
```

Index

*Topic **datasets**

AA.data, 1

intcox.example, 4

*Topic **survival**

intcox, 2

AA.data, 1

coxph, 3

coxph.object, 3

intcox, 2

intcox.example, 4

intcox.hazard0.beg(*intcox*), 2

intcox.pavaC(*intcox*), 2

Surv, 3