

Package ‘mets’

July 2, 2014

Type Package

Title Analysis of Multivariate Event Times

Version 0.2.8

Date 2014-05-07

Author Klaus K. Holst and Thomas Scheike

Maintainer Klaus K. Holst <kkho@biostat.ku.dk>

Description Implementation of various statistical models for multivariate event history data. Including multivariate cumulative incidence models, and bivariate random effects probit models (Liability models). Also contains two-stage binomial modelling that can do pairwise odds-ratio dependence modelling based marginal logistic regression models. This is an alternative to the alternating logistic regression approach (ALR).

License GPL (>= 2)

LazyLoad yes

URL <http://lava.r-forge.r-project.org/>

Depends R (>= 2.15), timereg (>= 1.8.4), lava (>= 1.2.4)

Imports numDeriv, compiler, Rcpp, splines

Suggests lava.tobit (>= 0.4-7), prodlim, testthat, ucminf

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-05-13 14:57:03

R topics documented:

mets-package	3
aalenfrailty	3
back2timereg	4
bicomprisk	5
binomial.twostage	6
blocksample	8
bptwin	9
casewise	10
casewise.test	11
ClaytonOakes	13
concordance	14
cor.cif	15
Dbvn	19
dermalridges	20
dermalridgesMZ	20
easy.binomial.twostage	21
easy.twostage	24
fast.approx	26
fast.pattern	27
fast.reshape	27
Grandom.cif	29
ipw	33
lifetable.matrix	34
mena	35
migr	35
multcif	35
np	35
npc	36
phreg	36
plack.cif	37
printcasewisetest	38
prt	38
random.cif	39
simAalenFrailty	41
simClaytonOakes	42
simClaytonOakesWei	42
summary.cor	43
test.conc	44
twinbmi	45
twinlm	45
twinsim	47
twinstut	48
twostage	49

mets-package	<i>Analysis of Multivariate Events</i>
--------------	--

Description

Implementation of various statistical models for multivariate event history data. Including multivariate cumulative incidence models, and bivariate random effects probit models (Liability models)

Author(s)

Klaus K. Holst and Thomas Scheike

Examples

To appear

aalenfrailty	<i>Aalen frailty model</i>
--------------	----------------------------

Description

Additive hazards model with (gamma) frailty

Usage

```
aalenfrailty(time, status, X, id, theta, B = NULL, ...)
```

Arguments

time	Time variable
status	Status variable (0,1)
X	Covariate design matrix
id	cluster variable
theta	list of thetas (returns score evaluated here), or starting point for optimization (defaults to magic number 0.1)
B	(optional) Cumulative coefficients (update theta by fixing B)
...	Additional arguments to lower level functions

Details

Aalen frailty model

Value

Parameter estimates

Author(s)

Klaus K. Holst

Examples

```

dd <- simAalenFrailty(5000)
f <- ~1##+x
X <- model.matrix(f,dd) ## design matrix for non-parametric terms
system.time(out<-aalen(update(f,Surv(time,status)~.),dd,n.sim=0,robust=0))
dix <- which(dd$status==1)
t1 <- system.time(bb <- .Call("Bhat",as.integer(dd$status),
                             X,0.2,as.integer(dd$id),NULL,NULL,-1,
                             package="mets"))

spec <- 1
##plot(out,spec=spec)
## plot(dd$time[dix],bb$B2[,spec],col="red",type="s",
##       ylim=c(0,max(dd$time)*c(beta0,beta)[spec]))
## abline(a=0,b=c(beta0,beta)[spec])
##'

## Not run: thetas <- seq(0.1,2,length.out=10)
Us <- unlist(aalenfrailty(dd$time,dd$status,X,dd$id,as.list(thetas)))
plot(thetas,Us,type="l",ylim=c(-.5,1)); abline(h=0,lty=2); abline(v=theta,lty=2)
op <- aalenfrailty(dd$time,dd$status,X,dd$id)
op
## End(Not run)

```

back2timereg

Convert to timereg object

Description

convert to timereg object

Usage

back2timereg(obj)

Arguments

obj no use

Author(s)

Thomas Scheike

bicomprisk

*Estimation of concordance in bivariate competing risks data***Description**

Estimation of concordance in bivariate competing risks data

Usage

```
bicomprisk(formula, data, cause = c(1, 1), cens = 0, causes, indiv,
           strata = NULL, id, num, max.clust = 1000, marg = NULL,
           se.clusters = NULL, prodlim = FALSE, messages = TRUE, model,
           return.data = 0, uniform = 0, conservative = 1, resample.iid = 1, ...)
```

Arguments

formula	Formula with left-hand-side being a Hist object (see example below) and the left-hand-side specifying the covariate structure
data	Data frame
cause	Causes (default (1,1)) for which to estimate the bivariate cumulative incidence
cens	The censoring code
causes	causes
indiv	indiv
strata	Strata
id	Clustering variable
num	num
max.clust	max number of clusters in comp.risk call for iid decomposition, max.clust=NULL uses all clusters otherwise rougher grouping.
marg	marginal cumulative incidence to make stanard errors for same clusters for subsequent use in casewise.test()
se.clusters	to specify clusters for standard errors. Either a vector of cluster indices or a column name in data. Defaults to the id variable.
prodlim	prodlim to use prodlim estimator (Aalen-Johansen) rather than IPCW weighted estimator based on comp.risk function. These are equivalent in the case of no covariates.
messages	Control amount of output
model	Type of competing risk model (default is Fine-Gray model "fg", see comp.risk).
return.data	Should data be returned (skipping modeling)
uniform	to compute uniform standard errors for concordance estimates based on resampling.
conservative	for conservative standard errors, recommended for larger data-sets.
resample.iid	to return iid residual processes for further computations such as tests.
...	Additional arguments to lower level functions

Author(s)

Thomas Scheike, Klaus K. Holst

binomial.twostage *Fits Clayton-Oakes or bivariate Plackett (OR) models for binary data using marginals that are on logistic form. If clusters contain more than two times, the algorithm uses a compososite likelihood based on all pairwise bivariate models.*

Description

The pairwise pairwise odds ratio model provides an alternative to the alternating logistic regression (ALR).

Usage

```
binomial.twostage(margbin, data = sys.parent(), score.method = "nlminb",
  Nit = 60, detail = 0, clusters = NULL, silent = 1, weights = NULL,
  control = list(), theta = NULL, theta.des = NULL, var.link = 1,
  iid = 1, step = 0.5, notaylor = 1, model = "plackett",
  marginal.p = NULL, strata = NULL, max.clust = NULL,
  se.clusters = NULL, numDeriv = 0)
```

Arguments

margbin	Marginal binomial model
data	data frame
score.method	Scoring method
Nit	Number of iterations
detail	Detail
clusters	Cluster variable
silent	Debug information
weights	Weights for log-likelihood, can be used for each type of outcome in 2x2 tables.
control	Optimization arguments
theta	Starting values for variance components
theta.des	Variance component design
var.link	Link function for variance
iid	Calculate i.i.d. decomposition
step	Step size
notaylor	Taylor expansion
model	model
marginal.p	vector of marginal probabilities

strata	strata for fitting: considers only pairs where both are from same strata
max.clust	max clusters
se.clusters	clusters for iid decomposition for robust standard errors
numDeriv	uses Fisher scoring approx of second derivative if 0, otherwise numerical derivatives

Details

The reported standard errors are based on a cluster corrected score equations from the pairwise likelihoods assuming that the marginals are known. This gives correct standard errors in the case of the Plackett distribution (OR model for dependence), but incorrect standard errors for the Clayton-Oakes types model.

Author(s)

Thomas Scheike

References

Two-stage binomial modelling

Examples

```
data(twinstut)
twinstut0 <- subset(twinstut, tvparnr<2300000)
twinstut <- twinstut0
theta.des <- model.matrix( ~-1+factor(zyg),data=twinstut)
margbin <- glm(stutter~factor(sex)+age,data=twinstut,family=binomial())
bin <- binomial.twostage(margbin,data=twinstut,
  clusters=twinstut$tvparnr,theta.des=theta.des,detail=0,
  score.method="fisher.scoring")
summary(bin)

twinstut$cage <- scale(twinstut$age)
theta.des <- model.matrix( ~-1+factor(zyg)+cage,data=twinstut)
bina <- binomial.twostage(margbin,data=twinstut,
  clusters=twinstut$tvparnr,theta.des=theta.des,detail=0,
  score.method="fisher.scoring")
summary(bina)

theta.des <- model.matrix( ~-1+factor(zyg)+factor(zyg)*cage,data=twinstut)
bina <- binomial.twostage(margbin,data=twinstut,
  clusters=twinstut$tvparnr,theta.des=theta.des,detail=0,
  score.method="fisher.scoring")
summary(bina)

twinstut$binstut <- (twinstut$stutter=="yes")*1
## refers to zygosity of first subject in each pair : zyg1
## could also use zyg2 (since zyg2=zyg1 within twinpair's)
out <- easy.binomial.twostage(stutter~factor(sex)+age,data=twinstut,
  response="binstut",id="tvparnr",
```

```

theta.formula=~-1+factor(zyg1),
score.method="fisher.scoring")
summary(out)

## refers to zygosity of first subject in each pair : zyg1
## could also use zyg2 (since zyg2=zyg1 within twinpair's)
desfs<-function(x,num1="zyg1",num2="zyg2")
  c(x[num1]=="dz",x[num1]=="mz",x[num1]=="os")*1

out3 <- easy.binomial.twostage(binstut~factor(sex)+age,
  data=twinstut,response="binstut",id="tvparnr",
  score.method="fisher.scoring",theta.formula=desfs,desnames=c("mz","dz","os"))
summary(out3)

```

blocksample

Block sampling

Description

Sample blockwise from clustered data

Usage

```
blocksample(data, size, idvar = "id", replace = TRUE, ...)
```

Arguments

data	Data frame
idvar	Column defining the clusters
size	Size of samples
replace	Logical indicating whether to sample with replacement
...	additional arguments to lower level functions

Value

data.frame

Author(s)

Klaus K. Holst

Examples

```
d <- data.frame(x=rnorm(5), z=rnorm(5), id=c(4,10,10,5,5), v=rnorm(5))
(dd <- blocksample(d,size=20))
attributes(dd)$id

## Not run:
blocksample(data.table::data.table(d),1e6)

## End(Not run)
```

bptwin

*Liability model for twin data***Description**

Liability-threshold model for twin data

Usage

```
bptwin(formula, data, id, zyg, DZ, group = NULL, num = NULL,
weight = NULL, biweight = function(x) 1/min(x), strata = NULL,
messages = 1, control = list(trace = 0), type = "ace", eqmean = TRUE,
paironly = FALSE, samecens = TRUE, allmarg = samecens &
!is.null(weight), stderr = TRUE, robustvar = TRUE, p, indiv = FALSE,
constrain, bound = FALSE, varlink, ...)
```

Arguments

formula	Formula specifying effects of covariates on the response.
data	data.frame with one observation pr row. In addition a column with the zygosity (DZ or MZ given as a factor) of each individual much be specified as well as a twin id variable giving a unique pair of numbers/factors to each twin pair.
id	The name of the column in the dataset containing the twin-id variable.
zyg	The name of the column in the dataset containing the zygosity variable.
DZ	Character defining the level in the zyg variable corresponding to the dizygotic twins.
group	Optional. Variable name defining group for interaction analysis (e.g., gender)
num	Optional twin number variable
weight	Weight matrix if needed by the chosen estimator (IPCW)
biweight	Function defining the bivariate weight in each cluster
strata	Strata
messages	Control amount of messages shown
control	Control argument parsed on to the optimization routine. Starting values may be parsed as 'start'.

type	Character defining the type of analysis to be performed. Should be a subset of "acde" (additive genetic factors, common environmental factors, dominant genetic factors, unique environmental factors).
eqmean	Equal means (with type="cor")?
pairsonly	Include complete pairs only?
stderr	Should standard errors be calculated?
robustvar	If TRUE robust (sandwich) variance estimates of the variance are used
p	Parameter vector p in which to evaluate log-Likelihood and score function
indiv	If TRUE the score and log-Likelihood contribution of each twin-pair
constrain	Development argument
samecens	Same censoring
allmarg	Should all marginal terms be included
bound	Development argument
varlink	Link function for variance parameters
...	Additional arguments to lower level functions

Author(s)

Klaus K. Holst

See Also[twinlm](#), [twinlm.time](#), [twinlm.strata](#), [twinsim](#)**Examples**

```
data(twinstut)
b0 <- bptwin(stutter~sex,
             data=droplevels(subset(twinstut, zyg%in%c("mz", "dz"))),
             id="tvparnr", zyg="zyg", DZ="dz", type="ae")
summary(b0)
```

casewise

Estimates the casewise concordance based on Concordance and marginal estimate using prodlm but no testing

Description

.. content for description (no empty lines) ..

Usage

casewise(conc, marg, cause.marg)

Arguments

conc	Concordance
marg	Marginal estimate
cause.marg	specifies which cause that should be used for marginal cif based on prodlm

Author(s)

Thomas Scheike

Examples

```
library(prodlm)
data(prt);

### marginal cumulative incidence of prostate cancer##'
outm <- prodlm(Hist(time,status)~+1,data=prt)

times <- 60:100
cifmz <- predict(outm,cause=2,time=times,newdata=data.frame(zyg="MZ")) ## cause is 2 (second cause)
cifdz <- predict(outm,cause=2,time=times,newdata=data.frame(zyg="DZ"))

### concordance for MZ and DZ twins
cc <- bicomprisk(Hist(time,status)~strata(zyg)+id(id),data=prt,cause=c(2,2),prodlm=TRUE)
cdz <- cc$model$"DZ"
cmz <- cc$model$"MZ"

cdz <- casewise(cdz,outm,cause.marg=2)
cmz <- casewise(cmz,outm,cause.marg=2)

plot(cmz,ci=NULL,ylim=c(0,0.5),xlim=c(60,100),legend=TRUE,col=c(3,2,1))
par(new=TRUE)
plot(cdz,ci=NULL,ylim=c(0,0.5),xlim=c(60,100),legend=TRUE)
summary(cdz)
summary(cmz)
```

casewise.test	<i>Estimates the casewise concordance based on Concordance and marginal estimate using timereg and performs test for independence</i>
---------------	---

Description

Estimates the casewise concordance based on Concordance and marginal estimate using timereg and performs test for independence

Usage

```
casewise.test(conc, marg, test = "no-test", p = 0.01)
```

Arguments

conc	Concordance
marg	Marginal estimate
test	Type of test for independence assumption. "conc" makes test on concordance scale and "case" means a test on the casewise concordance
p	check that marginal probability is greater at some point than p

Details

Uses cluster based conservative standard errors for marginal

Author(s)

Thomas Scheike

Examples

```

data(prt);

prt <- prt[which(prt$id %in% sample(unique(prt$id),7500)),]
### marginal cumulative incidence of prostate cancer
times <- seq(60,100,by=2)
outm <- comp.risk(Hist(time,status)~+1,data=prt,cause=2,times=times)

cifmz <- predict(outm,X=1,uniform=0,resample.iid=1)
cifdz <- predict(outm,X=1,uniform=0,resample.iid=1)

### concordance for MZ and DZ twins
cc <- bicomprisk(Hist(time,status)~strata(zyg)+id(id),
                 data=prt,cause=c(2,2))
cdz <- cc$model$"DZ"
cmz <- cc$model$"MZ"

### To compute casewise cluster argument must be passed on,
### here with a max of 100 to limit comp-time
outm <-comp.risk(Hist(time,status)~+1,data=prt,
                 cause=2,times=times,max.clust=100)
cifmz <-predict(outm,X=1,uniform=0,resample.iid=1)
cc <-bicomprisk(Hist(time,status)~strata(zyg)+id(id),data=prt,
                 cause=c(2,2),se.clusters=outm$clusters)
cdz <- cc$model$"DZ"
cmz <- cc$model$"MZ"

cdz <- casewise.test(cdz,cifmz,test="case") ## test based on casewise
cmz <- casewise.test(cmz,cifmz,test="conc") ## based on concordance

plot(cmz,ylim=c(0,0.7),xlim=c(60,100))
par(new=TRUE)
plot(cdz,ylim=c(0,0.7),xlim=c(60,100))

```

ClaytonOakes

*Clayton-Oakes model with piece-wise constant hazards***Description**

Clayton-Oakes frailty model

Usage

```
ClaytonOakes(formula, data = parent.frame(), cluster, var.formula = ~1,
  cuts = NULL, type = "piecewise", start, control = list(),
  var.invlink = exp, ...)
```

Arguments

formula	formula specifying the marginal proportional (piecewise constant) hazard structure with the right-hand-side being a survival object (Surv) specifying the entry time (optional), the follow-up time, and event/censoring status at follow-up. The clustering can be specified using the special function cluster (see example below).
data	Data frame
cluster	Variable defining the clustering (if not given in the formula)
var.formula	Formula specifying the variance component structure (if not given via the cluster special function in the formula) using a linear model with log-link.
cuts	Cut points defining the piecewise constant hazard
type	when equal to two.stage, the Clayton-Oakes-Glidden estimator will be calculated via the timereg package
start	Optional starting values
control	Control parameters to the optimization routine
var.invlink	Inverse link function for variance structure model
...	Additional arguments

Author(s)

Klaus K. Holst

Examples

```
set.seed(1)
d <- subset(simClaytonOakes(500,4,2,1,stoptime=2,left=2),!truncated)
e <- ClaytonOakes(Surv(lefttime,time,status)~x1+cluster(~1,cluster),
  cuts=c(0,0.5,1,2),data=d)
e
```

```

d2 <- simClaytonOakes(500,4,2,1,stopTime=2,left=0)
d2$z <- rep(1,nrow(d2)); d2$z[d2$cluster%in%sample(d2$cluster,100)] <- 0
## Marginal=Cox Proportional Hazards model:
ts <- ClaytonOakes(Surv(time,status)~prop(x1)+cluster(~1,cluster),
                   data=d2,type="two.stage")
## Marginal=Aalens additive model:
ts2 <- ClaytonOakes(Surv(time,status)~x1+cluster(~1,cluster),
                    data=d2,type="two.stage")
## Marginal=Piecewise constant:
e2 <- ClaytonOakes(Surv(time,status)~x1+cluster(~1+factor(z),cluster),
                   cuts=c(0,0.5,1,2),data=d2)

e2
plot(ts)
plot(e2,add=TRUE)

e3 <- ClaytonOakes(Surv(time,status)~x1+cluster(~1,cluster),cuts=c(0,0.5,1,2),
                   data=d,var.invlink=identity)

e3

```

concordance

Concordance Computes concordance and probandwise concordance

Description

Concordance

Usage

```

concordance(object, cif1, cif2 = NULL, messages = TRUE, model = NULL,
            coefs = NULL, ...)

```

Arguments

object	Output from the cor.cif, rr.cif or or.cif function
cif1	Marginal cumulative incidence
cif2	Marginal cumulative incidence of other cause (cause2) if it is different from cause1
messages	To print messages
model	Specifies which model that is considered if object not given.
coefs	Specifies dependence parameters if object is not given.
...	Extra arguments, not used.

Author(s)

Thomas Scheike

cor.cif

Cross-odds-ratio, OR or RR risk regression for competing risks

Description

Fits a parametric model for the log-cross-odds-ratio for the predictive effect of for the cumulative incidence curves for T_1 experiencing cause i given that T_2 has experienced a cause k :

$$\log(COR(i|k)) = h(\theta, z_1, i, z_2, k, t) =_{default} \theta^T z =$$

with the log cross odds ratio being

$$COR(i|k) = \frac{O(T_1 \leq t, cause_1 = i | T_2 \leq t, cause_2 = k)}{O(T_1 \leq t, cause_1 = i)}$$

the conditional odds divided by the unconditional odds, with the odds being, respectively

$$O(T_1 \leq t, cause_1 = i | T_2 \leq t, cause_2 = k) = \frac{P_x(T_1 \leq t, cause_1 = i | T_2 \leq t, cause_2 = k)}{P_x((T_1 \leq t, cause_1 = i)^c | T_2 \leq t, cause_2 = k)}$$

and

$$O(T_1 \leq t, cause_1 = i) = \frac{P_x(T_1 \leq t, cause_1 = i)}{P_x((T_1 \leq t, cause_1 = i)^c)}.$$

Here B^c is the complement event of B , P_x is the distribution given covariates (x are subject specific and z are cluster specific covariates), and $h(\cdot)$ is a function that is the simple identity $\theta^T z$ by default.

Usage

```
cor.cif(cif, data, cause, times = NULL, cause1 = 1, cause2 = 1,
        cens.code = NULL, cens.model = "KM", Nit = 40, detail = 0,
        clusters = NULL, theta = NULL, theta.des = NULL, step = 1, sym = 0,
        weights = NULL, par.func = NULL, dpar.func = NULL, dimpar = NULL,
        score.method = "nlminb", same.cens = FALSE, censoring.probs = NULL,
        silent = 1, ...)
```

Arguments

cif	a model object from the comp.risk function with the marginal cumulative incidence of cause1, i.e., the event of interest, and whose odds the comparison is compared to the conditional odds given cause2
data	a data.frame with the variables.
cause	specifies the causes related to the death times, the value cens.code is the censoring value. When missing it comes from marginal cif.
times	time-vector that specifies the times used for the estimating equations for the cross-odds-ratio estimation.
cause1	specifies the cause considered.
cause2	specifies the cause that is conditioned on.

cens.code	specifies the code for the censoring if NULL then uses the one from the marginal cif model.
cens.model	specified which model to use for the ICPW, KM is Kaplan-Meier alternatively it may be "cox"
Nit	number of iterations for Newton-Raphson algorithm.
detail	if 0 no details are printed during iterations, if 1 details are given.
clusters	specifies the cluster structure.
theta	specifies starting values for the cross-odds-ratio parameters of the model.
theta.des	specifies a regression design for the cross-odds-ratio parameters.
step	specifies the step size for the Newton-Raphson algorithm.
sym	specifies if symmetry is used in the model.
weights	weights for estimating equations.
par.func	parfunc
dpar.func	dparfunc
dimpar	dimpar
score.method	"nlminb", can also use "fisher-scoring".
same.cens	if true then censoring within clusters are assumed to be the same variable, default is independent censoring.
censoring.probs	if cens.model is "user.weights" these probabilities are used for the bivariate censoring dist.
silent	1 to suppress output about convergence related issues.
...	Not used.

Details

The OR dependence measure is given by

$$OR(i, k) = \log\left(\frac{O(T_1 \leq t, cause_1 = i | T_2 \leq t, cause_2 = k)}{O(T_1 \leq t, cause_1 = i) | T_2 \leq t, cause_2 = k}\right)$$

This measure is numerically more stable than the COR measure, and is symmetric in i,k.

The RR dependence measure is given by

$$RR(i, k) = \log\left(\frac{P(T_1 \leq t, cause_1 = i, T_2 \leq t, cause_2 = k)}{P(T_1 \leq t, cause_1 = i)P(T_2 \leq t, cause_2 = k)}\right)$$

This measure is numerically more stable than the COR measure, and is symmetric in i,k.

The model is fitted under symmetry (sym=1), i.e., such that it is assumed that T_1 and T_2 can be interchanged and leads to the same cross-odd-ratio (i.e. $COR(i|k) = COR(k|i)$), as would be expected for twins or without symmetry as might be the case with mothers and daughters (sym=0).

$h()$ may be specified as an R-function of the parameters, see example below, but the default is that it is simply $\theta^T z$.

Value

returns an object of type 'cor'. With the following arguments:

theta	estimate of proportional odds parameters of model.
var.theta	variance for gamma.
hess	the derivative of the used score.
score	scores at final stage.
score	scores at final stage.
theta.iid	matrix of iid decomposition of parametric effects.

Author(s)

Thomas Scheike

References

Cross odds ratio Modelling of dependence for Multivariate Competing Risks Data, Scheike and Sun (2010), work in progress.

A Semiparametric Random Effects Model for Multivariate Competing Risks Data, Scheike, Zhang, Sun, Jensen (2010), Biometrika.

Examples

```
data(multcif);
multcif$cause[multcif$cause==0] <- 2
zyg <- rep(rbinom(200,1,0.5),each=2)
theta.des <- model.matrix(~-1+factor(zyg))

times=seq(0.05,1,by=0.05) # to speed up computations use only these time-points
add<-comp.risk(Hist(time,cause)~+1+cluster(id),data=multcif,cause=1,
              n.sim=0,times=times,model="fg",max.clust=NULL)
add2<-comp.risk(Hist(time,cause)~+1+cluster(id),data=multcif,cause=2,
              n.sim=0,times=times,model="fg",max.clust=NULL)

out1<-cor.cif(add,data=multcif,cause1=1,cause2=1)
summary(out1)

out2<-cor.cif(add,data=multcif,cause1=1,cause2=1,theta.des=theta.des)
summary(out2)

##out3<-cor.cif(add,data=multcif,cause1=1,cause2=2,cif2=add2)
##summary(out3)
#####
# investigating further models using parfunc and dparfunc
#####

set.seed(100)
prt<-simnordic(1000, cordz=2, cormz=5)
prt$status <-prt$cause
```

```

table(prt$status)

times <- seq(40,100,by=10)
cifmod <- comp.risk(Hist(time,cause)~+1+cluster(id),data=prt,
                   cause=1,n.sim=0,
                   times=times,conservative=1,max.clust=NULL,model="fg")
theta.des <- model.matrix(~-1+factor(zyg),data=prt)

parfunc <- function(par,t,pardes)
{
par <- pardes %*% c(par[1],par[2]) +
      pardes %*% c( par[3]*(t-60)/12,par[4]*(t-60)/12)
par
}
head(parfunc(c(0.1,1,0.1,1),50,theta.des))

dparfunc <- function(par,t,pardes)
{
dpar <- cbind(pardes, t(t(pardes) * c( (t-60)/12,(t-60)/12)) )
dpar
}
head(dparfunc(c(0.1,1,0.1,1),50,theta.des))

names(prt)
or1 <- or.cif(cifmod,data=prt,cause1=1,cause2=1,theta.des=theta.des,
             same.cens=TRUE,theta=c(0.6,1.1,0.1,0.1),
             par.func=parfunc,dpar.func=dparfunc,dimpar=4,
             score.method="fisher.scoring",detail=1)
summary(or1)

cor1 <- cor.cif(cifmod,data=prt,cause1=1,cause2=1,theta.des=theta.des,
               same.cens=TRUE,theta=c(0.5,1.0,0.1,0.1),
               par.func=parfunc,dpar.func=dparfunc,dimpar=4,
               control=list(trace=TRUE),detail=1)
summary(cor1)
##'
### piecewise contant OR model
gparfunc <- function(par,t,pardes)
{
cuts <- c(0,80,90,120)
grop <- diff(t<cuts)
paru <- (pardes[,1]==1) * sum(grop*par[1:3]) +
      (pardes[,2]==1) * sum(grop*par[4:6])
paru
}

dgparfunc <- function(par,t,pardes)
{
cuts <- c(0,80,90,120)
grop <- diff(t<cuts)
par1 <- matrix(c(grop),nrow(pardes),length(grop),byrow=TRUE)
parmz <- par1* (pardes[,1]==1)
pardz <- (pardes[,2]==1) * par1

```

```

dpar <- cbind( parmz, pardz)
dpar
}
head(dgparfunc(rep(0.1,6),50,theta.des))
head(gparfunc(rep(0.1,6),50,theta.des))

or1g <- or.cif(cifmod,data=prt,cause1=1,cause2=1,
              theta.des=theta.des, same.cens=TRUE,
              par.func=gparfunc,dpar.func=dgparfunc,
              dimpar=6,score.method="fisher.scoring",detail=1)
summary(or1g)
names(or1g)
head(or1g$theta.iid)

```

Dbvn

Derivatives of the bivariate normal cumulative distribution function

Description

Derivatives of the bivariate normal cumulative distribution function

Usage

```

Dbvn(p,design=function(p,...) {
  return(list(mu=cbind(p[1],p[1]),
             dmu=cbind(1,1),
             S=matrix(c(p[2],p[3],p[3],p[4]),ncol=2),
             dS=rbind(c(1,0,0,0),c(0,1,1,0),c(0,0,0,1)))  }),
     Y=cbind(0,0))

```

Arguments

p	Parameter vector
design	Design function with defines mean, derivative of mean, variance, and derivative of variance with respect to the parameter p
Y	column vector where the CDF is evaluated

Author(s)

Klaus K. Holst

dermalridges

Dermal ridges data (families)

Description

Data on dermal ridge counts in left and right hand in (nuclear) families

Format

Data on 50 families with ridge counts in left and right hand for moter, father and each child. Family id in 'family' and gender and child number in 'sex' and 'child'.

Source

Sarah B. Holt (1952). Genetics of dermal ridges: bilateral asymmetry in finger ridge-counts. *Annals of Eugenics* 17 (1), pp.211–231. DOI: 10.1111/j.1469-1809.1952.tb02513.x

Examples

```
data(dermalridges)
fast.reshape(dermalridges,id="family",varying=c("child.left","child.right","sex"))
```

dermalridgesMZ

Dermal ridges data (monozygotic twins)

Description

Data on dermal ridge counts in left and right hand in (nuclear) families

Format

Data on dermal ridge counts (left and right hand) in 18 monozygotic twin pairs.

Source

Sarah B. Holt (1952). Genetics of dermal ridges: bilateral asymmetry in finger ridge-counts. *Annals of Eugenics* 17 (1), pp.211–231. DOI: 10.1111/j.1469-1809.1952.tb02513.x

Examples

```
data(dermalridgesMZ)
fast.reshape(dermalridgesMZ,id="id",varying=c("left","right"))
```

easy.binomial.twostage

Fits two-stage binomial for describing dependence in binomial data using marginals that are on logistic form using the binomial.twostage function, but call is different and easier and the data manipulation is build into the function. Useful in particular for family design data.

Description

If clusters contain more than two times, the algorithm uses a composited likelihood based on the pairwise bivariate models.

Usage

```
easy.binomial.twostage(margbin = NULL, data = sys.parent(),
  score.method = "nlminb", response = "response", id = "id", Nit = 60,
  detail = 0, silent = 1, weights = NULL, control = list(),
  theta = NULL, theta.formula = NULL, desnames = NULL, deshelp = 0,
  var.link = 1, iid = 1, step = 0.5, model = "plackett",
  marginal.p = NULL, strata = NULL, max.clust = NULL,
  se.clusters = NULL)
```

Arguments

margbin	Marginal binomial model
data	data frame
response	name of response variable in data frame
id	name of cluster variable in data frame
score.method	Scoring method
Nit	Number of iterations
detail	Detail for more output for iterations
silent	Debug information
weights	Weights for log-likelihood, can be used for each type of outcome in 2x2 tables.
control	Optimization arguments
theta	Starting values for variance components
theta.formula	design for dependence, either formula or design function
desnames	names for dependence parameters
deshelp	if 1 then prints out some data sets that are used, on on which the design function operates
var.link	Link function for variance
iid	Calculate i.i.d. decomposition
step	Step size

model	model
marginal.p	vector of marginal probabilities
strata	strata for fitting
max.clust	max clusters
se.clusters	clusters for iid decomposition for robust standard errors

Details

The reported standard errors are based on the estimated information from the likelihood assuming that the marginals are known. This gives correct standard errors in the case of the plackett distribution (OR model for dependence), but incorrect for the clayton-oakes types model. The OR model is often known as the ALR model, but our fitting procedures gives correct standard errors and is quite a bit quicker.

Examples

```

data(twinstut)
twinstut0 <- subset(twinstut, tvparnr<230000)
twinstut <- twinstut0
theta.des <- model.matrix( ~-1+factor(zyg),data=twinstut0)
margbin <- glm(stutter~factor(sex)+age,data=twinstut0,family=binomial())
bin <- binomial.twostage(margbin,data=twinstut0,
  clusters=twinstut0$tvparnr,theta.des=theta.des,detail=0,
  score.method="fisher.scoring")
summary(bin)

twinstut0$cage <- scale(twinstut0$age)
theta.des <- model.matrix( ~-1+factor(zyg)+cage,data=twinstut0)
bina <- binomial.twostage(margbin,data=twinstut0,
  clusters=twinstut0$tvparnr,theta.des=theta.des,detail=0,
  score.method="fisher.scoring")
summary(bina)

theta.des <- model.matrix( ~-1+factor(zyg)+factor(zyg)*cage,data=twinstut0)
bina <- binomial.twostage(margbin,data=twinstut0,
  clusters=twinstut0$tvparnr,theta.des=theta.des,detail=0,
  score.method="fisher.scoring")
summary(bina)

twinstut0$binstut <- (twinstut0$stutter=="yes")*1
out <- easy.binomial.twostage(stutter~factor(sex)+age,data=twinstut0,
  response="binstut",id="tvparnr",
  theta.formula=~-1+factor(zyg1),
  score.method="fisher.scoring")
summary(out)

## refers to zygosity of first subject in each pair : zyg1
## could also use zyg2 (since zyg2=zyg1 within twinpair's)
desfs <- function(x,num1="zyg1",namesdes=c("mz","dz","os"))
  c(x[num1]=="mz",x[num1]=="dz",x[num1]=="os")*1

```



```

theta.formula=desfsi,desnames=c("pp","mother-child","father-child","cc"))
summary(udi)

##now looking to see if interactions with age or age influences marginal models
##converting factors to numeric to make all involved covariates numeric
##to use desfai2 rather than desfai that works on binfam

nbinfam <- binfam
nbinfam$num <- as.numeric(binfam$num)
head(nbinfam)

desfsai <- function(x,num1="num1",num2="num2")
{
  pp <- (x[num1]=="m")*(x[num2]=="f")*1
  ### av age for pp=1 i.e parent pairs
  agepp <- ((as.numeric(x["age1"])+as.numeric(x["age2"])))/2-30)*pp
  mc <- (x[num1]=="m")*(x[num2]=="b1" | x[num2]=="b2")*1
  fc <- (x[num1]=="f")*(x[num2]=="b1" | x[num2]=="b2")*1
  cc <- (x[num1]=="b1")*(x[num2]=="b1" | x[num2]=="b2")*1
  agecc <- ((as.numeric(x["age1"])+as.numeric(x["age2"])))/2-12)*cc
  c(pp,agepp,mc,fc,cc,agecc)
}

desfsai2 <- function(x,num1="num1",num2="num2")
{
  pp <- (x[num1]==1)*(x[num2]==2)*1
  agepp <- (((x["age1"]+x["age2"])/2-30)*pp) ### av age for pp=1 i.e parent pairs
  mc <- (x[num1]==1)*(x[num2]==3 | x[num2]==4)*1
  fc <- (x[num1]==2)*(x[num2]==3 | x[num2]==4)*1
  cc <- (x[num1]==3)*(x[num2]==3 | x[num2]==4)*1
  agecc <- ((x["age1"]+x["age2"])/2-12)*cc) ### av age for children
  c(pp,agepp,mc,fc,cc,agecc)
}

udxai2 <- easy.binomial.twostage(y~x+age,data=binfam,
  response="y",id="id",
  score.method="fisher.scoring",deshelp=0,detail=0,
  theta.formula=desfsai,
  desnames=c("pp","pp-age","mother-child","father-child","cc","cc-age"))
summary(udxai2)

```

easy.twostage

Fits two-stage model for describing dependence in survival data using marginals that are on cox or aalen form using the twostage function, but call is different and easier and the data manipulation build into the function. Useful in particular for family design data.

Description

If clusters contain more than two times, the algorithm uses a composite likelihood based on the pairwise bivariate models.

Usage

```
easy.twostage(margsurv = NULL, data = sys.parent(),
  score.method = "nlminb", status = "status", time = "time",
  entry = NULL, id = "id", Nit = 60, detail = 0, silent = 1,
  weights = NULL, control = list(), theta = NULL, theta.formula = NULL,
  desnames = NULL, deshelf = 0, var.link = 1, iid = 1, step = 0.5,
  model = "plackett", marginal.surv = NULL, strata = NULL,
  max.clust = NULL, se.clusters = NULL)
```

Arguments

margsurv	model
data	data frame
score.method	Scoring method
status	Status at exit time
time	Exit time
entry	Entry time
id	name of cluster variable in data frame
Nit	Number of iterations
detail	Detail for more output for iterations
silent	Debug information
weights	Weights for log-likelihood, can be used for each type of outcome in 2x2 tables.
control	Optimization arguments
theta	Starting values for variance components
theta.formula	design for dependence, either formula or design function
desnames	names for dependence parameters
deshelf	if 1 then prints out some data sets that are used, on on which the design function operates
var.link	Link function for variance (exp link)
iid	Calculate i.i.d. decomposition
step	Step size for newton-raphson
model	plackett or clayton-oakes model
marginal.surv	vector of marginal survival probabilities
strata	strata for fitting
max.clust	max clusters
se.clusters	clusters for iid decomposition for roubst standard errors

Details

The reported standard errors are based on the estimated information from the likelihood assuming that the marginals are known.

Examples

```
data(prt)
margp<- coxph(Surv(time,status==1)~factor(country),data=prt)
fitco<-twostage(margp,data=prt,clusters=prt$id)
summary(fitco)

des <- model.matrix(~-1+factor(zyg),data=prt);
fitco<-twostage(margp,data=prt,theta.des=des,clusters=prt$id)
summary(fitco)

dfam <- simSurvFam(1000)
dfam <- fast.reshape(dfam,var=c("x","time","status"))

desfs <- function(x,num1="num1",num2="num2")
{
pp <- (x[num1]=="m")*(x[num2]=="f")*1    ## mother-father
pc <- (x[num1]=="m" | x[num1]=="f")*(x[num2]=="b1" | x[num2]=="b2")*1 ## mother-child
cc <- (x[num1]=="b1")*(x[num2]=="b1" | x[num2]=="b2")*1          ## child-child
c(pp,pc,cc)
}

marg <- coxph(Surv(time,status)~factor(num),data=dfam)
out3 <- easy.twostage(marg,data=dfam,time="time",status="status",id="id",deshelp=0,
                    score.method="fisher.scoring",theta.formula=desfs,
                    desnames=c("parent-parent","parent-child","child-cild"))

summary(out3)
```

fast.approx

Fast approximation

Description

Fast approximation

Usage

```
fast.approx(time, new.time, equal = FALSE, ...)
```

Arguments

time	Original ordered time points
new.time	New time points
equal	If TRUE a list is returned with additional element
...	Optional additional arguments

Author(s)

Klaus K. Holst

Examples

```
id <- c(1,1,2,2,7,7,10,10)
fast.approx(unique(id),id)

t <- 0:6
n <- c(-1,0,0.1,0.9,1,1.1,1.2,6,6.5)
fast.approx(t,n,equal=TRUE)
```

fast.pattern	<i>Fast pattern</i>
--------------	---------------------

Description

Fast pattern

Usage

```
fast.pattern(x, y, ...)
```

Arguments

x	Matrix (binary) of patterns. Optionally if y is also passed as argument, then the pattern matrix is defined as the elements agreeing in the two matrices.
y	Optional matrix argument with same dimensions as x (see above)
...	Optional additional arguments

Author(s)

Klaus K. Holst

fast.reshape	<i>Fast reshape Simple reshape/tranpose of data</i>
--------------	---

Description

Fast reshape Simple reshape/tranpose of data

Usage

```
fast.reshape(data, varying, id, num, sep = "", keep, idname = "id",
  numname = "num", factors.keep = TRUE, idcombine = TRUE,
  labelnum = FALSE, ...)
```

Arguments

data	data.frame or matrix
id	id-variable. If omitted then reshape Wide->Long.
varying	Vector of prefix-names of the time varying variables. Optional for Long->Wide reshaping.
num	Optional number/time variable
sep	String separating prefix-name with number/time
keep	Vector of column names to keep
idname	Name of id-variable (Wide->Long)
numname	Name of number-variable (Wide->Long)
factors.keep	If false all factors are converted to integers
idcombine	If TRUE and id is vector of several variables, the unique id is combined from all the variables. Otherwise the first variable is only used as identifier.
labelnum	If TRUE varying variables in wide format (going from long->wide) are labeled 1,2,3,... otherwise use 'num' variable. In long-format (going from wide->long) varying variables matching 'varying' prefix are only selected if their postfix is a number.
...	Optional additional arguments

Author(s)

Thomas Scheike, Klaus K. Holst

Examples

```
library(lava)
m <- lvm(c(y1,y2,y3,y4)~x)
d <- sim(m,5)
fast.reshape(fast.reshape(d,"y"),id="id")

##### From wide-format
(dd <- fast.reshape(d,"y"))
## Same with explicit setting new id and number variable/column names
## and separator "" (default) and dropping x
fast.reshape(d,"y",idname="a",timevar="b",sep="",keep=c())
## Same with 'reshape' list-syntax
fast.reshape(d,list(c("y1","y2","y3","y4"))))

##### From long-format
fast.reshape(dd,id="id")
## Restrict set up within-cluster varying variables
fast.reshape(dd,"y",id="id")
fast.reshape(dd,"y",id="id",keep="x",sep=".")

#####
x <- data.frame(id=c(5,5,6,6,7),y=1:5,x=1:5,tv=c(1,2,2,1,2))
(xw <- fast.reshape(x,id="id"))
```

```

(xl <- fast.reshape(xw,c("y","x"),idname="id2",keep=c()))
(xl <- fast.reshape(xw,c("y","x","tv")))
(xw2 <- fast.reshape(xl,id="id",num="num"))
fast.reshape(xw2,c("y","x"),idname="id")

### more generally:
### varying=list(c("ym","yf","yb1","yb2"), c("zm","zf","zb1","zb2"))
### varying=list(c("ym","yf","yb1","yb2"))

##### Family cluster example
d <- mets::simBinFam(3)
d
dd <- fast.reshape(d,var="y")
dd
dd2 <- fast.reshape(d,varying=list(c("ym","yf","yb1","yb2")))
dd2
##'
##'

d <- sim(lvm(~y1+y2+ya),10)
(dd <- fast.reshape(d,"y"))
fast.reshape(d,"y",labelnum=TRUE)
fast.reshape(dd,id="id",num="num")
fast.reshape(dd,id="id",num="num",labelnum=TRUE)
fast.reshape(d,c(a="y"),labelnum=TRUE) ## New column name

##### Unbalanced data
m <- lvm(c(y1,y2,y3,y4)~ x+z1+z3+z5)
d <- sim(m,3)
d
fast.reshape(d,c("y","z"))

##### not-varying syntax:
fast.reshape(d,-c("x"))

##### Automatically define varying variables from trailing digits
fast.reshape(d)

##### Prostate cancer example
data(prt)
head(prtw <- fast.reshape(prt,"cancer",id="id"))
ftable(cancer1~cancer2,data=prtw)
rm(prtw)

```

Description

Fits a random effects model describing the dependence in the cumulative incidence curves for subjects within a cluster. Given the gamma distributed random effects it is assumed that the cumulative incidence curves are independent, and that the marginal cumulative incidence curves are on additive form

$$P(T \leq t, \text{cause} = 1|x, z) = P_1(t, x, z) = 1 - \exp(-x^T A(t) - tz^T \beta)$$

Usage

```
Grandom.cif(cif, data, cause, cif2 = NULL, times = NULL, cause1 = 1,
  cause2 = 1, cens.code = NULL, cens.model = "KM", Nit = 40,
  detail = 0, clusters = NULL, theta = NULL, theta.des = NULL,
  weights = NULL, step = 1, sym = 0, same.cens = FALSE,
  censoring.probs = NULL, silent = 1, exp.link = 0,
  score.method = "fisher.scoring", entry = NULL, estimator = 1,
  trunkp = 1, admin.cens = NULL, random.design = NULL, ...)
```

Arguments

<code>cif</code>	a model object from the <code>comp.risk</code> function with the marginal cumulative incidence of <code>cause2</code> , i.e., the event that is conditioned on, and whose odds the comparison is made with respect to
<code>data</code>	a <code>data.frame</code> with the variables.
<code>cause</code>	specifies the causes related to the death times, the value <code>cens.code</code> is the censoring value.
<code>cif2</code>	specificities model for <code>cause2</code> if different from <code>cause1</code> .
<code>times</code>	time points
<code>cause1</code>	cause of first coordinate.
<code>cause2</code>	cause of second coordinate.
<code>cens.code</code>	specifies the code for the censoring if <code>NULL</code> then uses the one from the marginal <code>cif</code> model.
<code>cens.model</code>	specified which model to use for the ICPW, KM is Kaplan-Meier alternatively it may be "cox"
<code>Nit</code>	number of iterations for Newton-Raphson algorithm.
<code>detail</code>	if 0 no details are printed during iterations, if 1 details are given.
<code>clusters</code>	specifies the cluster structure.
<code>theta</code>	specifies starting values for the cross-odds-ratio parameters of the model.
<code>theta.des</code>	specifies a regression design for the cross-odds-ratio parameters.
<code>step</code>	specifies the step size for the Newton-Raphson algorithm.
<code>sym</code>	1 for symmetry and 0 otherwise
<code>weights</code>	weights for score equations.
<code>same.cens</code>	if true then censoring within clusters are assumed to be the same variable, default is independent censoring.

censoring.probs	Censoring probabilities
silent	debug information
exp.link	if exp.link=1 then var is on log-scale.
score.method	default uses "nlminb" optimizer, alternatively, use the "fisher-scoring" algorithm.
entry	entry-age in case of delayed entry. Then two causes must be given.
estimator	estimator
trunkp	gives probability of survival for delayed entry, and related to entry-ages given above.
admin.cens	Administrative censoring
random.design	specifies a regression design of 0/1's for the random effects.
...	extra arguments.

Details

We allow a regression structure for the independent gamma distributed random effects and their variances that may depend on cluster covariates.

random.design specifies the random effects for each subject within a cluster. This is a matrix of 1's and 0's with dimension $n \times d$. With d random effects. For a cluster with two subjects, we let the random.design rows be v_1 and v_2 . Such that the random effects for subject 1 is

$$v_1^T(Z_1, \dots, Z_d)$$

, for d random effects. Each random effect has an associated parameter $(\lambda_1, \dots, \lambda_d)$. By construction subjects 1's random effect are Gamma distributed with mean $\lambda_1/v_1^T \lambda$ and variance $\lambda_1/(v_1^T \lambda)^2$. Note that the random effect $v_1^T(Z_1, \dots, Z_d)$ has mean 1 and variance $1/(v_1^T \lambda)$.

The parameters $(\lambda_1, \dots, \lambda_d)$ are related to the parameters of the model by a regression construction $pard$ ($d \times k$), that links the d λ parameters with the (k) underlying θ parameters

$$\lambda = pard\theta$$

Value

returns an object of type 'random.cif'. With the following arguments:

theta	estimate of parameters of model.
var.theta	variance for gamma.
hess	the derivative of the used score.
score	scores at final stage.
theta.iid	matrix of iid decomposition of parametric effects.

Author(s)

Thomas Scheike

References

A Semiparametric Random Effects Model for Multivariate Competing Risks Data, Scheike, Zhang, Sun, Jensen (2010), *Biometrika*.

Cross odds ratio Modelling of dependence for Multivariate Competing Risks Data, Scheike and Sun (2012), *Biostatistics*, to appear.

Scheike, Holst, Hjelmberg (2012), *LIDA*, to appear. Estimating heritability for cause specific hazards based on twin data

Examples

```

data(multcif)
multcif$cause[multcif$cause==0] <- 2
addm<-comp.risk(Hist(time,cause)~const(X)+cluster(id),data=multcif,
               cause=1,n.sim=0)

### making group indicator
g.des<-data.frame(group2=rep(rbinom(200,1,0.5),rep(2,200)))
theta.des <- model.matrix(~-1+factor(group2),g.des)

out1m<-random.cif(addm,data=multcif,cause1=1,cause2=1,Nit=15,detail=0,
                 theta=2,theta.des=theta.des,step=1.0)
summary(out1m)

## this model can also be formulated as a random effects model
## but with different parameters
out2m<-Grandom.cif(addm,data=multcif,cause1=1,cause2=1,Nit=10,detail=0,
                  random.design=theta.des,step=1.0)
summary(out2m)
1/out2m$theta
out1m$theta

#####
##### ACE modelling of twin data #####
#####
### assume that zygbn gives the zygosity of mono and dizygotic twins
### 0 for mono and 1 for dizygotic twins. We now formulate an AC model
zygbn <- g.des$group2 ## indicator of dizygotic twins

n <- nrow(multcif)
### random effects for each cluster
des.rv <- cbind(theta.des,(zygbn==1)*rep(c(1,0)),(zygbn==1)*rep(c(0,1)),1)
### design making parameters half the variance for dizygotic components
pardes <- rbind(c(1,0), c(0.5,0),c(0.5,0), c(0.5,0), c(0,1))

outacem <-Grandom.cif(addm,data=multcif,causeS=1,Nit=30,detail=0,
                    theta=c(-1.21,2.1),theta.des=pardes,step=1.0,random.design=des.rv)
summary(outacem)
### genetic variance is
exp(outacem$theta[1])/sum(exp(outacem$theta))^2

```

ipw *Inverse Probability of Censoring Weights*

Description

Calculates Inverse Probability of Censoring Weights (IPCW) and adds them to a data.frame

Usage

```
ipw(formula, data, cluster, samecens = FALSE, obsonly = TRUE,
    weightname = "w", cens.model = "aalen", pairs = FALSE, ...)
```

Arguments

formula	Formula specifying the censoring model
data	data frame
cluster	clustering variable
samecens	For clustered data, should same censoring be assumed (bivariate probability calculated as minimum of the marginal probabilities)
obsonly	Return data with uncensored observations only
weightname	Name of weight variable in the new data.frame
cens.model	Censoring model (default Aalen additive model)
pairs	For paired data (e.g. twins) only the complete pairs are returned (With pairs=TRUE)
...	Additional arguments to censoring model

Author(s)

Klaus K. Holst

Examples

```
data(prt)
prtw <- ipw(Surv(time,status==0)~country, data=prt[sample(nrow(prt),5000),],
            cluster="id",weightname="w")
plot(0,type="n",xlim=range(prtw$time),ylim=c(0,1),xlab="Age",ylab="Probability")
count <- 0
for (l in unique(prtw$country)) {
  count <- count+1
  prtw <- prtw[order(prtw$time),]
  with(subset(prtw,country==l),
       lines(time,w,col=count,lwd=2))
}
legend("topright",legend=unique(prtw$country),col=1:4,pch=-1,lty=1)
```

lifetable.matrix	<i>Life table</i>
------------------	-------------------

Description

Create simple life table

Usage

```
## S3 method for class 'matrix'
lifetable(x, strata = list(), breaks = c(),
          confint = FALSE, ...)

## S3 method for class 'formula'
lifetable(x, data=parent.frame(), breaks = c(),
          confint = FALSE, ...)
```

Arguments

x	time formula (Surv) or matrix/data.frame with columns time,status or entry,exit,status
strata	Strata
data	data.frame
breaks	Time intervals
confint	If TRUE 95% confidence limits are calculated
...	Additional arguments to lower level functions

Author(s)

Klaus K. Holst

Examples

```
library(timereg)
data(TRACE)

lifetable(Surv(time,status==9)~sex+I(cut(wmi,c(-Inf,1,1.5,Inf))),
          data=TRACE,breaks=c(0.2),confint=TRUE)

d <- with(TRACE,lifetable(Surv(time,status==9)~sex+vf,breaks=c(0.2,0.5)))
summary(glm(events ~ offset(log(atrisk))+interval*vf + sex*vf,
            data=d,poisson))
```

mena	<i>Menarche data set</i>
------	--------------------------

Description

Menarche data set

Source

Simulated data

migr	<i>Migraine data</i>
------	----------------------

Description

Migraine data

multcif	<i>Multivariate Cumulative Incidence Function example data set</i>
---------	--

Description

Multivariate Cumulative Incidence Function example data set

Source

Simulated data

np	<i>np data set</i>
----	--------------------

Description

np data set

Source

Simulated data

npc *For internal use*

Description

For internal use

Author(s)

Klaus K. Holst

phreg *Fast Cox PH regression*

Description

Fast Cox PH regression

Usage

```
phreg(formula, data, ...)
```

Arguments

formula	formula with 'Surv' outcome (see coxph)
data	data frame
...	Additional arguments to lower level funtions

Author(s)

Klaus K. Holst

Examples

```
simcox <- function(n=1000, seed=1, beta=c(1,1), entry=TRUE) {
  if (!is.null(seed))
    set.seed(seed)
  library(lava)
  m <- lvm()
  regression(m,T~X1+X2) <- beta
  distribution(m,~T+C) <- coxWeibull.lvm(scale=1/100)
  distribution(m,~entry) <- coxWeibull.lvm(scale=1/10)
  m <- eventTime(m,time~min(T,C=0),"status")
  d <- sim(m,n);
  if (!entry) d$entry <- 0
  else d <- subset(d, time>entry,select=-c(T,C))
}
```

```

    return(d)
  }

  n <- 1e3;
  d <- mets:::simCox(n); d$id <- seq(nrow(d)); d$group <- factor(rbinom(nrow(d),1,0.5))

  (m1 <- phreg(Surv(entry,time,status)~X1+X2,data=d))
  (m2 <- coxph(Surv(entry,time,status)~X1+X2+cluster(id),data=d))
  (coef(m3 <-cox.aalen(Surv(entry,time,status)~prop(X1)+prop(X2),data=d)))

  ## Not run:
  (m1b <- phreg(Surv(entry,time,status)~X1+X2+strata(group),data=d))
  (m2b <- coxph(Surv(entry,time,status)~X1+X2+cluster(id)+strata(group),data=d))
  (coef(m3b <-cox.aalen(Surv(entry,time,status)~-1+group+prop(X1)+prop(X2),data=d)))

  ## End(Not run)

  m <- phreg(Surv(entry,time,status)~X1*X2+strata(group)+cluster(id),data=d)
  m
  plot(m,ylim=c(0,1))

```

plack.cif

plack Computes concordance for or.cif based model, that is Plackett random effects model

Description

.. content for description (no empty lines) ..

Usage

```
plack.cif(cif1, cif2, object)
```

Arguments

cif1	Cumulative incidence of first argument.
cif2	Cumulative incidence of second argument.
object	or.cif object with dependence parameters.

Author(s)

Thomas Scheike

printcasewisetest *prints Concordance test*

Description

.. content for description (no empty lines) ..

Usage

```
printcasewisetest(x, digits = 3, ...)
```

Arguments

x	output from casewise.test
digits	number of digits
...	Additional arguments to lower level functions

Author(s)

Thomas Scheike

prt *Prostate data set*

Description

Prostate data set

Source

Simulated data

random.cif

*Random effects model for competing risks data***Description**

Fits a random effects model describing the dependence in the cumulative incidence curves for subjects within a cluster. Given the gamma distributed random effects it is assumed that the cumulative incidence curves are independent, and that the marginal cumulative incidence curves are on the form

$$P(T \leq t, \text{cause} = 1|x, z) = P_1(t, x, z) = 1 - \exp(-x^T A(t) \exp(z^T \beta))$$

We allow a regression structure for the random effects variances that may depend on cluster covariates.

Usage

```
random.cif(cif, data, cause, cif2 = NULL, cause1 = 1, cause2 = 1,
  cens.code = NULL, cens.model = "KM", Nit = 40, detail = 0,
  clusters = NULL, theta = NULL, theta.des = NULL, step = 1,
  same.cens = FALSE, exp.link = 0, score.method = "fisher.scoring",
  entry = NULL, trunkp = 1, ...)
```

Arguments

cif	a model object from the comp.risk function with the marginal cumulative incidence of cause2, i.e., the event that is conditioned on, and whose odds the comparison is made with respect to
data	a data.frame with the variables.
cause	specifies the causes related to the death times, the value cens.code is the censoring value.
cif2	specifies model for cause2 if different from cause1.
cause1	cause of first coordinate.
cause2	cause of second coordinate.
cens.code	specifies the code for the censoring if NULL then uses the one from the marginal cif model.
cens.model	specified which model to use for the ICPW, KM is Kaplan-Meier alternatively it may be "cox"
Nit	number of iterations for Newton-Raphson algorithm.
detail	if 0 no details are printed during iterations, if 1 details are given.
clusters	specifies the cluster structure.
theta	specifies starting values for the cross-odds-ratio parameters of the model.
theta.des	specifies a regression design for the cross-odds-ratio parameters.
step	specifies the step size for the Newton-Raphson algorithm.

same.cens	if true then censoring within clusters are assumed to be the same variable, default is independent censoring.
exp.link	if exp.link=1 then var is on log-scale.
score.method	default uses "nlminb" optimizer, alternatively, use the "fisher-scoring" algorithm.
entry	entry-age in case of delayed entry. Then two causes must be given.
trunkp	gives probability of survival for delayed entry, and related to entry-ages given above.
...	extra arguments.

Value

returns an object of type 'cor'. With the following arguments:

theta	estimate of proportional odds parameters of model.
var.theta	variance for gamma.
hess	the derivative of the used score.
score	scores at final stage.
score	scores at final stage.
theta.iid	matrix of iid decomposition of parametric effects.

Author(s)

Thomas Scheike

References

A Semiparametric Random Effects Model for Multivariate Competing Risks Data, Scheike, Zhang, Sun, Jensen (2010), *Biometrika*.

Cross odds ratio Modelling of dependence for Multivariate Competing Risks Data, Scheike and Sun (2012), work in progress.

Examples

```
data(multcif)

times <- seq(0.3,1,length=4)
add<-comp.risk(Hist(time,cause)~+1+cluster(id),data=multcif,cause=1,
              n.sim=0,times=times,max.clust=NULL)

out1<-random.cif(add,data=multcif,cause1=1,cause2=1)
summary(out1)

zyg <- rep(rbinom(200,1,0.5),each=2)
theta.des <- model.matrix(~-1+factor(zyg))
###theta.des<-model.matrix(~-1+factor(zyg),data=np)
out2<-random.cif(add,data=multcif,cause1=1,cause2=1,theta.des=theta.des)
summary(out2)
#####
```



```
##### 2 different causes
#####

## multcif$cause[multcif$cause==0] <- 2

## ###times<-sort(multcif$time[multcif$status %in% c(1,2)])
## add1<-comp.risk(Hist(time,status)~const(X)+cluster(id),data=multcif,cause=1,
##   multcif$cause,n.sim=0,times=times)
## add2<-comp.risk(Hist(time,status)~const(X)+cluster(id),data=multcif,cause=2,
##   multcif$cause,n.sim=0,times=times)

## out1<-random.cif(add1,data=multcif,cause1=1,cause2=2,cif2=add2)
## summary(out1) ## negative dependence

## out1g<-random.cif(add1,data=multcif,cause1=1,cause2=2,
##   cif2=add2,theta.des=theta.des)
## summary(out1g)
```

 simAalenFrailty

Simulate from the Aalen Frailty model

Description

Simulate observations from Aalen Frailty model with Gamma distributed frailty and constant intensity.

Usage

```
simAalenFrailty(n = 5000, theta = 0.3, K = 2, beta0 = 1.5, beta = 1,
  cens = 1.5, cuts = 0, ...)
```

Arguments

n	Number of observations in each cluster
theta	Dependence paramter (variance of frailty)
K	Number of clusters
beta0	Baseline (intercept)
beta	Effect (log hazard ratio) of covariate
cens	Censoring rate
cuts	time cuts
...	Additional arguments

Author(s)

Klaus K. Holst

simClaytonOakes *Simulate from the Clayton-Oakes frailty model*

Description

Simulate observations from the Clayton-Oakes copula model with piecewise constant marginals.

Usage

```
simClaytonOakes(K, n, eta, beta, stoptime, left = 0, pairleft = 0)
```

Arguments

K	Number of clusters
n	Number of observations in each cluster
eta	1/variance
beta	Effect (log hazard ratio) of covariate
stoptime	Stopping time
left	Left truncation
pairleft	pairwise (1) left truncation or individual (0)

Author(s)

Klaus K. Holst

simClaytonOakesWei *Simulate from the Clayton-Oakes frailty model*

Description

Simulate observations from the Clayton-Oakes copula model with Weibull type baseline and Cox marginals.

Usage

```
simClaytonOakesWei(K, n, eta, beta, stoptime, weiscale = 1, weishape = 2,
  left = 0, pairleft = 0)
```

Arguments

K	Number of clusters
n	Number of observations in each cluster
eta	1/variance
beta	Effect (log hazard ratio) of covariate
stoptime	Stopping time
weiscale	weibull scale parameter
weishape	weibull shape parameter
left	Left truncation
pairleft	pairwise (1) left truncation or individual (0)

Author(s)

Klaus K. Holst

summary.cor

Summary for dependence models for competing risks

Description

Computes concordance and probandwise concordance for dependence models for competing risks models of the type cor.cif, rr.cif or or.cif for the given cumulative incidences and the different dependence measures in the object.

Usage

```
## S3 method for class 'cor'
summary(object, marg.cif = NULL, marg.cif2 = NULL,
        digits = 3, ...)
```

Arguments

object	object from cor.cif rr.cif or or.cif for dependence between competing risks data for two causes.
marg.cif	a number that gives the cumulative incidence in one time point for which concordance and probandwise concordance are computed.
marg.cif2	the cumulative incidence for cause 2 for concordance and probandwise concordance are computed. Default is that it is the same as marg.cif.
digits	digits in output.
...	Additional arguments.

Value

prints summary for dependence model.

probandwise gives probandwise concordance that is, probability of cause 2 (related to cif2) given that cause 1 (related to cif1) has occurred.

concordance gives concordance that is, probability of cause 2 (related to cif2) and cause 1 (related to cif1).

cif1 cumulative incidence for cause1.

cif2 cumulative incidence for cause1.

Author(s)

Thomas Scheike

References

Cross odds ratio Modelling of dependence for Multivariate Competing Risks Data, Scheike and Sun (2012), Biostatistics to appear.

A Semiparametric Random Effects Model for Multivariate Competing Risks Data, Scheike, Zhang, Sun, Jensen (2010), Biometrika.

Examples

```
data(multcif) # simulated data
multcif$cause[multcif$cause==0] <- 2

times=seq(0.05,3,by=0.1) # to speed up computations use only these time-points
add<-comp.risk(Hist(time,cause)~const(X)+cluster(id),data=multcif,
               n.sim=0,times=times)
###
out1<-cor.cif(add,data=multcif,cause1=1,cause2=1,theta=log(2+1))
summary(out1)

pad <- predict(add,X=1,Z=0,se=0,uniform=0)$P1
summary(out1,marg.cif=pad)
```

test.conc

Concordance test Compares two concordance estimates

Description

.. content for description (no empty lines) ..

Usage

```
test.conc(conc1, conc2, same.cluster = FALSE)
```

Arguments

conc1	Concordance estimate of group 1
conc2	Concordance estimate of group 2
same.cluster	if FALSE then groups are independent, otherwise estimates are based on same data.

Author(s)

Thomas Scheike

twinbmi	<i>BMI data set</i>
---------	---------------------

Description

BMI data set

Format

Self-reported BMI-values on 11,411 subjects

tvparnr: twin id bmi: BMI (m/kg²) age: Age gender: (male/female) zyg: zygosity, MZ:=mz, DZ(same sex):=dz, DZ(opposite sex):=os

twinlm	<i>Classic twin model for quantitative traits</i>
--------	---

Description

Fits a classical twin model for quantitative traits.

Usage

```
twinlm(formula, data, id, zyg, DZ, group = NULL, group.equal = FALSE,
strata = NULL, weight = NULL, type = c("ace"), twinum = "twinnum",
binary = FALSE, keep = weight, estimator = "gaussian",
constrain = TRUE, control = list(), messages = 1, ...)
```

Arguments

formula	Formula specifying effects of covariates on the response
data	data.frame with one observation pr row. In addition a column with the zygosity (DZ or MZ given as a factor) of each individual much be specified as well as a twin id variable giving a unique pair of numbers/factors to each twin pair
id	The name of the column in the dataset containing the twin-id variable.
zyg	The name of the column in the dataset containing the zygosity variable
DZ	Character defining the level in the zyg variable corresponding to the dizygotic twins. If this argument is missing, the reference level (i.e. the first level) will be interpreted as the dizygotic twins
group	Optional. Variable name defining group for interaction analysis (e.g., gender)
group.equal	If TRUE marginals of groups are asummed to be the same
strata	Strata variable name
weight	Weight matrix if needed by the chosen estimator. For use with Inverse Probability Weights
type	Character defining the type of analysis to be performed. Should be a subset of "aced" (additive genetic factors, common environmental factors, unique environmental factors, dominant genetic factors).
twinnum	The name of the column in the dataset numbering the twins (1,2). If it does not exist in data it will automatically be created.
binary	If TRUE a liability model is fitted. Note that if the right-hand-side of the formula is a factor, character vector, og logical variable, then the liability model is automatically chosen (wrapper of the bptwin function).
keep	Vector of variables from data that are not specified in formula, to be added to data.frame of the SEM
estimator	Choice of estimator/model
constrain	Development argument
control	Control argument parsed on to the optimization routine
messages	Control amount of messages shown
...	Additional arguments parsed on to lower-level functions

Value

Returns an object of class twinlm.

Author(s)

Klaus K. Holst

See Also

[bptwin](#), [twinlm.time](#), [twinlm.strata](#), [twinsim](#)

Examples

```
## Simulate data
set.seed(1)
d <- twinsim(1000,b1=c(1,-1),b2=c(),acde=c(1,1,0,1))
## E(y|z1,z2) = z1 - z2. var(A) = var(C) = var(E) = 1

## E.g to fit the data to an ACE-model without any confounders we simply write
ace <- twinlm(y ~ 1, data=d, DZ="DZ", zyg="zyg", id="id")
ace
## An AE-model could be fitted as
ae <- twinlm(y ~ 1, data=d, DZ="DZ", zyg="zyg", id="id", type="ae")
## LRT:
lava::compare(ae,ace)
## AIC
AIC(ae)-AIC(ace)
## To adjust for the covariates we simply alter the formula statement
ace2 <- twinlm(y ~ x1+x2, data=d, DZ="DZ", zyg="zyg", id="id", type="ace")
## Summary/GOF
summary(ace2)
## An interaction could be analyzed as:
ace3 <- twinlm(y ~ x1+x2 + x1:I(x2<0), data=d, DZ="DZ", zyg="zyg", id="id", type="ace")
ace3
## Categorical variables are also supported
d2 <- transform(d,x2cat=cut(x2,3,labels=c("Low","Med","High")))
ace4 <- twinlm(y ~ x1+x2cat, data=d2, DZ="DZ", zyg="zyg", id="id", type="ace")
## plot the model structure
## Not run:
plot(ace4)

## End(Not run)
```

twinsim

Simulate twin data

Description

Simulate twin data from a linear normal ACE/ADE/AE model.

Usage

```
twinsim(nMZ = 100, nDZ = nMZ, b1 = c(), b2 = c(), mu = 0,
       acde = c(1, 1, 0, 1), randomslope = NULL, threshold = 0, cens = FALSE,
       wide = FALSE, ...)
```

Arguments

nMZ	Number of monozygotic twin pairs
nDZ	Number of dizygotic twin pairs

b1	Effect of covariates (labelled x1,x2,...) of type 1. One distinct covariate value for each twin/individual.
b2	Effect of covariates (labelled g1,g2,...) of type 2. One covariate value for each twin pair.
mu	Intercept parameter.
acde	Variance of random effects (in the order A,C,D,E)
randomslope	Logical indicating wether to include random slopes of the variance components w.r.t. x1,x2,...
threshold	Treshold used to define binary outcome y0
cens	Logical variable indicating whether to censor outcome
wide	Logical indicating if wide data format should be returned
...	Additional arguments parsed on to lower-level functions

Author(s)

Klaus K. Holst

See Also[twinlm](#)

twinstut

*Stutter data set***Description**

Based on nation-wide questionnaire answers from 33,317 Danish twins

Format

tvparnr: twin-pair id zyg: zygosity, MZ:=mz, DZ(same sex):=dz, DZ(opposite sex):=os stutter:
 stutter status (yes/no) age: age nr: number within twin-pair

twostage	<i>Fits Clayton-Oakes or bivariate Plackett models for bivariate survival data using marginals that are on Cox or additive form. If clusters contain more than two times, the algorithm uses a composite likelihood based on the pairwise bivariate models.</i>
----------	---

Description

The reported standard errors are based on the estimated information from the likelihood assuming that the marginals are known.

Usage

```
twostage(margsurv, data = sys.parent(), score.method = "nlminb", Nit = 60,
  detail = 0, clusters = NULL, silent = 1, weights = NULL,
  control = list(), theta = NULL, theta.des = NULL, var.link = 1,
  iid = 1, step = 0.5, notaylor = 0, model = "plackett",
  marginal.trunc = NULL, marginal.survival = NULL, marginal.status = NULL,
  strata = NULL, se.clusters = NULL, max.clust = NULL, numDeriv = 1)
```

Arguments

margsurv	Marginal model
data	data frame
score.method	Scoring method
Nit	Number of iterations
detail	Detail
clusters	Cluster variable
silent	Debug information
weights	Weights
control	Optimization arguments
theta	Starting values for variance components
theta.des	Variance component design
var.link	Link function for variance
iid	Calculate i.i.d. decomposition
step	Step size
notaylor	Taylor expansion
model	model
marginal.trunc	marginal left truncation probabilities
marginal.survival	optional vector of marginal survival probabilities

marginal.status related to marginal survival probabilities
 strata strata for fitting, see example
 se.clusters for clusters for se calculation with iid
 max.clust max se.clusters for se calculation with iid
 numDeriv to get numDeriv version of second derivative, otherwise uses sum of squared score

Author(s)

Thomas Scheike

References

Clayton-Oakes and Plackett bivariate survival distributions,

Examples

```

data(diabetes)

# Marginal Cox model with treat as covariate
margph <- coxph(Surv(time,status)~treat,data=diabetes)
### Clayton-Oakes, from timereg
fitco1<-two.stage(margph,data=diabetes,theta=1.0,detail=0,Nit=40,clusters=diabetes$id)
summary(fitco1)
### Plackett model
fitp<-twostage(margph,data=diabetes,theta=3.0,Nit=40,
               clusters=diabetes$id,var.link=1)
summary(fitp)
### Clayton-Oakes
fitco2<-twostage(margph,data=diabetes,theta=0.0,detail=0,
                 clusters=diabetes$id,var.link=1,model="clayton.oakes")
summary(fitco2)
fitco3<-twostage(margph,data=diabetes,theta=1.0,detail=0,
                 clusters=diabetes$id,var.link=0,model="clayton.oakes")
summary(fitco3)

### without covariates using Aalen for marginals
marg <- aalen(Surv(time,status)~+1,data=diabetes,n.sim=0,max.clust=NULL,robust=0)
fitpa<-twostage(marg,data=diabetes,theta=1.0,detail=0,Nit=40,
                clusters=diabetes$id,score.method="optimize")
summary(fitpa)

fitcoa<-twostage(marg,data=diabetes,theta=1.0,detail=0,Nit=40,clusters=diabetes$id,
                 var.link=1,model="clayton.oakes")
summary(fitcoa)

### Piecewise constant cross hazards ratio modelling
#####

d <- subset(simClaytonOakes(2000,2,0.5,0,stoptime=2,left=0),!truncated)

```

```

udp <- piecewise.twostage(c(0,0.5,2),data=d,score.method="optimize",
                        id="cluster",timevar="time",
                        status="status",model="clayton.oakes",silent=0)

summary(udp)

### Same model using the strata option, a bit slower
#####
## makes the survival pieces for different areas in the plane
##ud1=surv.boxarea(c(0,0),c(0.5,0.5),data=d,id="cluster",timevar="time",status="status")
##ud2=surv.boxarea(c(0,0.5),c(0.5,2),data=d,id="cluster",timevar="time",status="status")
##ud3=surv.boxarea(c(0.5,0),c(2,0.5),data=d,id="cluster",timevar="time",status="status")
##ud4=surv.boxarea(c(0.5,0.5),c(2,2),data=d,id="cluster",timevar="time",status="status")

## everything done in one call
ud <- piecewise.data(c(0,0.5,2),data=d,timevar="time",status="status",id="cluster")
ud$strata <- factor(ud$strata);
ud$intstrata <- factor(ud$intstrata)

## makes strata specific id variable to identify pairs within strata
## se's computed based on the id variable across strata "cluster"
ud$idstrata <- ud$id+(as.numeric(ud$strata)-1)*2000

marg2 <- aalen(Surv(boxtime,status)~-1+factor(num):factor(intstrata),
              data=ud,n.sim=0,robust=0)
tdes <- model.matrix(~-1+factor(strata),data=ud)
fitp2<-twostage(marg2,data=ud,se.clusters=ud$cluster,clusters=ud$idstrata,
              score.method="fisher.scoring",model="clayton.oakes",
              theta.des=tdes,step=0.5)

summary(fitp2)

### now fitting the model with symmetry, i.e. strata 2 and 3 same effect
ud$stratas <- ud$strata;
ud$stratas[ud$strata=="0.5-2,0-0.5"] <- "0-0.5,0.5-2"
tdes2 <- model.matrix(~-1+factor(stratas),data=ud)
fitp3<-twostage(marg2,data=ud,clusters=ud$idstrata,se.cluster=ud$cluster,
              score.method="fisher.scoring",model="clayton.oakes",
              theta.des=tdes2,step=0.5)

summary(fitp3)

### same model using strata option, a bit slower
fitp4<-twostage(marg2,data=ud,clusters=ud$cluster,se.cluster=ud$cluster,
              score.method="fisher.scoring",model="clayton.oakes",
              theta.des=tdes2,step=0.5,strata=ud$strata)

summary(fitp4)

```

Index

- *Topic **binomial**
 - binomial.twostage, 6
 - easy.binomial.twostage, 21
- *Topic **data**
 - dermalridges, 20
 - dermalridgesMZ, 20
 - mena, 35
 - migr, 35
 - multcif, 35
 - np, 35
 - prt, 38
 - twinbmi, 45
 - twinstut, 48
- *Topic **models**
 - blocksample, 8
 - twinlm, 45
 - twinsim, 47
- *Topic **package**
 - mets-package, 3
- *Topic **regression**
 - binomial.twostage, 6
 - easy.binomial.twostage, 21
 - twinlm, 45
 - twinsim, 47
- *Topic **survival**
 - cor.cif, 15
 - easy.twostage, 24
 - Grandom.cif, 29
 - random.cif, 39
 - summary.cor, 43
 - twostage, 49
- *Topic **twostage**
 - easy.twostage, 24
- *Topic **utilities**
 - blocksample, 8
 - npc, 36
- aalenfrailty, 3
- alpha2kendall (npc), 36
- alpha2spear (npc), 36
- back2timereg, 4
- bicomprisk, 5
- binomial.twostage, 6
- biprobbit (bptwin), 9
- blocksample, 8
- bptwin, 9, 46
- casewise, 10
- casewise.test, 11
- ClaytonOakes, 13
- cluster.index (npc), 36
- coefmat (npc), 36
- concordance, 14
- cor.cif, 15
- corsim.prostate (npc), 36
- Dbvn, 19
- dermalridges, 20
- dermalridgesMZ, 20
- easy.binomial.twostage, 21
- easy.twostage, 24
- familycluster.index (npc), 36
- fast.approx, 26
- fast.pattern, 27
- fast.reshape, 27
- faster.reshape (npc), 36
- Grandom.cif, 29
- grouptable (npc), 36
- ipw, 33
- lifetable (lifetable.matrix), 34
- lifetable.matrix, 34
- loglikMVN (npc), 36
- mena, 35
- mets-package, 3
- migr, 35

multcif, 35

nonparcuminc (npc), 36
np, 35
npc, 36

or.cif (cor.cif), 15

pbvn (npc), 36
phreg, 36
piecewise.data (npc), 36
piecewise.twostage (npc), 36
plack.cif, 37
plack.cif2 (plack.cif), 37
plotcr (npc), 36
pmvn (npc), 36
printcasewisetest, 38
prt, 38

random.cif, 39
rr.cif (cor.cif), 15

sim (npc), 36
simAalenFrailty, 41
simBinFam (npc), 36
simBinFam2 (npc), 36
simBinPlack (npc), 36
simClaytonOakes, 42
simClaytonOakesWei, 42
simCox (npc), 36
simnordic (npc), 36
simSurvFam (npc), 36
summary.cor, 43
surv.boxarea (npc), 36

test.conc, 44
twinbmi, 45
twinlm, 10, 45, 48
twinlm.strata, 10, 46
twinlm.time, 10, 46
twinlm.time (bptwin), 9
twinsim, 10, 46, 47
twinstut, 48
twostage, 49