Package ‘changeLOS’

January 2, 2012

Version 2.0.9-2

Date 2008-10-16

Title Change in LOS

Author Matthias Wangler <mw@imbi.uni-freiburg.de>, Jan Beyersmann <jan@fdm.uni-freiburg.de>

Maintainer Arthur Allignol <arthur.allignol@fdm.uni-freiburg.de>

Depends R (>= 1.8.1), survival

Description Change in length of hospital stay (LOS) is frequently used to assess the impact and the costs of hospital-acquired complications. In order to compute the attributable change in LOS, it is crucial to account for the timing of events: A complication can only have an effect on LOS, once it has occured. These temporal dynamics can be adequately handled by multistate models; however, there is few software for such models available. We introduce an R-package ‘‘changeLOS’’ for computing change in LOS based on methods described in Schulgen and Schumacher (1996). We will illustrate the program on data from a prospective cohort study on hospital-acquired infections. Main features of the R-package ‘‘changeLOS’’ are R-methods to: (1) describe the multi-state model. (2) compute the Aalen-Johansen estimator for the matrix of transition probabilities P(u-, u) for all observed transition times u.(3) compute the Aalen-Johansen estimator for the matrix of transition probabilities P(s,t); the estimator is a finite matrix product of matrices P(u-,u) for every observed event time in the interval(s,t]. (4) visualize the temporal dynamics of the data, illustrated by transition probabilities. (5) compute and visualize change in LOS. (6) compute bootstrap variances for change in LOS.

Repository CRAN

Date/Publication 2009-03-26 11:29:35

R topics documented:

`aj` *Aalen-Johansen estimator*

Description

computes the Aalen-Johansen estimator for the matrix of transition probabilities $P(s,t)$. The entry (l,m) of the matrix denotes the estimated probability that state m has been reached by time t given state l has been occupied by time s.

Usage

```
aj(tr, s, t)
```

Arguments

`tr`	an object of 'trans'
`s`	begin of the time interval
`t`	end of the time interval

Details

The Aalen-Johansen estimator is considered in detail by Andersen et al. (1993). Usually, the process describing movements between states is considered to be Markovian; this assumption may be relaxed, cf. Datta and Satten (2001).

The Aalen-Johansen estimator is a finite matrix product, one matrix for every observed transition time in the time interval $(s, t]$. These matrices can be obtained from `trans`.

Value

An object of class 'aj'. The object is a list of:

`matrix`	the matrix of transition probabilities $P(s, t)$
`start`	the beginning s of the time interval $(s, t]$
`end`	the end t of the time interval $(s, t]$
`times`	the transition times in the interval $(s, t]$
`matrices`	array of estimators for $P(s, u)$ for all transition times u in $(s, t]$

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

References

P Andersen, O Borgan, R Gill, and N Keiding (1993). *Statistical models based on counting processes.* New York: Springer

S Datta, and G Satten (2001). Validity of the Aalen-Johansen estimators of stage occupation probabilities and Nelson-Aalen estimators of integrated transition hazards for non-Markov models. *Statistics and Probability Letters* 55 (4), 403–411.

Andersen and Keiding (2002). Multi-state models for event history analysis. *Statistical Methods in Medical Research* 11 (2), 91–115.

See Also

`trans`

Examples

```
data(los.data)
my.observ <- prepare.los.data(x=los.data)
my.model <- msmodel(c("0","1","2","3"),cens.name="cens")
my.trans <- trans(model=my.model,observ=my.observ)
my.aj <- aj(my.trans,s=0,t=80)
```

clos *Change in LOS*

Description

estimates the expected change in length of stay (LOS) associated with an intermediate event (IE). In order to account for the timing of events (an IE can only have an effect once it has occurred), a multi-state model is used. The data are stratified into 'cases' (the IE has occurred) and 'controls' (the IE has *not yet* occurred) on a daywise basis. ('Daywise' if time is in days.) The expected change in LOS is estimated for each day and a weighted average is computed.

Usage

```
clos(model, observ, aw=FALSE)
```

Arguments

`model`	an object of the class 'msmodel' which describes the multi-state model
`observ`	a data.frame of the form data.frame(id, from, to, time, oid) (see also `prepare.los.data`): **id** id (patient id, admision id) **from** the state from where a transition occurs **to** the state to which a transition occurs **time** the time a transition occurs **oid** the observation id
`aw`	locical, with (TRUE) or without(FALSE) alternative weighting

Details

`clos` is based on 'approach B' in Schulgen and Schumacher (1996), however with some modifications: We use a multi-state model with four states. All individuals start in an initial state. They may pass through the intermediate state. LOS is determined by reaching one of two competing, absorbing states. In clinical research these latter two states are typically 'discharge' and 'death', respectively. The IE is often interpreted as a complication. Backward transitions are not possible. Schulgen and Schumacher (1996) modelled a second intermediate state, motivated by their application. However, this model has some individuals 'drop out' in the sense that they become neither 'case' nor 'control'. In addition, `clos` computes an weighted average, where the weights are given by the weighting time distribution in the intial state. Schulgen and Schumacher (1996) used a conditional version of it, given one reaches the IE. In the case of no censoring, the former weights have every individual contribute to the weighting, whereas the latter has not. One can interpret the conditional weighting in Schulgen and Schumacher (1996) as assuming a patient's viewpoint who experiences the IE (and analogously for patients uninfected, with weights given one is directly discharged/dies without prior IE.) Finally, `clos` treats the daywise comparison as being zero if there are not both patients with and without IE on that day. See also Beyersmann et al. (2005) for a non-technical explanation of these methods.

See the examples for special features:

- bootstrapping
- use of the alternative weight
- distinguishing between patients discharged and patients deceased

Value

An object of class c('clos'). The object is a list of

`cLOS` change in LOS

`trans` an object of class 'trans'

`e.given.1` estimates E(LOS|X_s =intermediate event) for all observed transition times s, where X_s denotes the state by time s

`e.given.0` estimates E(LOS|X_s =initial state) for all observed transition times s, where X_s denotes the state by time s

`phi2` weighted average of the difference between phi2.case and phi2.control, this quantity can be interpreted as the contribution to the expected change in LOS at time s by patients infected at time s who eventually discharge

`phi2.case` estimates E(LOS **1**(X_LOS =discharge)|X_s =intermediate event), where **1** denotes the indicator function

`phi2.control` P(X_LOS =discharge|X_s =intermediate event) E(LOS|X_s =initial state)

`phi3` weighted average of the difference between phi3.case and phi3.control, this quantity can be interpreted as the contribution to the expected change in LOS at time s by patients infected at time s who eventually die

`phi3.case` estimates E(LOS **1**(X_LOS =death)|X_s =intermediate event), where **1** denotes the indicator function

`phi3.control` P(X_LOS =death|X_s =intermediate event) E(LOS|X_s =initial state)

`empty.1` event times with the group 'intermediate, but no terminal event yet' being empty

`empty.0` event times with the group 'no intermediate or terminal event yet' being empty

`weights` weights for the weighted average

`w.times` time points corresponding to the weights

`called` how the function was called

`patients` total number of observed patients

`patients.discharge` number of patients being discharged

`patients.death` number of patients who die

`patients.cens` number of patients being censored, i. e. for whom neither discharge or death was observed

`cases` number of patients who experienced the IE

`cases.discharge` number of patients who experienced the IE being discharged

`cases.death` number of patients who experienced the IE and died

`cases.cens` number of patients who experienced the IE and were censored

Value 'alternative weighting'

When aw=TRUE an object of class c('clos','closa'). The object is a list like above with the following extra items:

weights.1 weights corresponding to the conditional weighting time in the intial state given one reaches the IE.

weights.23 weights corresponding to the conditional weighting time in the intial state given one *does not* reach the IE.

given.1 change in LOS corresponding to the alternative weights 'weights.1'

given.23 change in LOS corresponding to the alternative weights 'weights.2'

Note

required packages: survival

Author(s)

Jan Beyersmann <jan@fdm.uni-freiburg.de> Matthias Wangler <mw@imbi.uni-freiburg.de>

References

G Schulgen and M Schumacher (1996). Estimation of prolongation of hospital stay attributable to nosocomial infections. *Lifetime Data Analysis* 2, 219-240.

J Beyersmann, P Gastmeier, H Grundmann, S Baerwolf, C Geffers, M Behnke, H Rueden, and M Schumacher (2006). Use of Multistate Models to Assess Prolongation of Intensive Care Unit Stay Due to Nosocomial Infection. *Infection Control and Hospital Epidemiology* 27, 493-499.

M Wrangler, J Beyersmann and M Schumacher (2006). changeLOS: An R-package for change in length of hospital stay based on the Aalen-Johansen estimator. *R News* 6(2), 31–35.

A reference on multi-state models:

P Andersen and N Keiding (2002). Multi-state models for event history analysis. *Statistical Methods in Medical Research* 11, 91-115

See Also

survival, exampleclos

Examples

```
## run clos
data(los.data)
my.observ <- prepare.los.data(x=los.data)
tra <- matrix(FALSE,4,4)
diag(tra) <- TRUE
tra[1,] <- TRUE
tra[2,3:4] <- TRUE
my.model <- msmodel(c("0","1","2","3"),tra,cens.name="cens")
los <- clos(model=my.model,observ=my.observ)
```

```
summary(los)
plot(los)

## expected change in LOS due to an IE
phi <- los$e.given.1 - los$e.given.0

## distinguishing between patients discharged
phi2 <- los$phi2.case - los$phi2.control
## and patients deceased
phi3 <- los$phi3.case - los$phi3.control

## we have phi = phi2 + phi3
all(round(phi, digits=10) == round(phi2+phi3, digits=10),na.rm=TRUE)


## compute bootstrap SE with function boot() from library boot

## first we need a statistic, which boot takes as an argument
"clos.for.bstrap" <- function(data, index, mod){
  my.observ <- prepare.los.data(x=data[index,])
  return(clos(model=mod,observ=my.observ)$cLOS)
}

## our estimate is
clos.for.bstrap(data=los.data, index=1:length(los.data[,1]), mod=my.model)

## now bootstrap
library(boot)
nb <- 20 ## nb <- 2000
los.bootstrap <- boot(los.data, clos.for.bstrap, nb, mod=my.model)
sqrt(var(los.bootstrap$t))


## compute change in LOS 'by hand'; also works for alternative weights

## first, get pure event times
my.evtimes <- sort(unique(my.observ$time[my.observ$to !="cens"]))
## compute daywise difference between expected LOS
my.diffs <- los$e.given.1 - los$e.given.0
## restrict to those days when a transition out of the initial state was observed
my.diffs <- my.diffs[is.element(my.evtimes, los$w.times)]
## compute weighted average, but don't sum up over
## days where one of the groups (no) IE (yet) experienced was empty.
sum((my.diffs * los$weights)[!(is.element(los$w.times, c(los$empty.0, los$empty.1)))])
```

---

exampleclos *example for clos*

Description

An example program estimating change in length of stay associated with an intermediate event.

Usage

```
exampleclos()
```

Value

An object of class c('clos'). The object is a list of

`cLOS`	change in LOS
`trans`	an object of class 'trans'
`e.given.1`	estimates E(LOS|X_s =intermediate event) for all observed transition times s, where X_s denotes the state by time s
`e.given.0`	estimates E(LOS|X_s =initial state) for all observed transition times s, where X_s denotes the state by time s
`phi2`	weighted average of the difference between phi2.case and phi2.control
`phi2.case`	estimates E(LOS 1(X_LOS =discharge)|X_s =intermediate event)
`phi2.control`	P(X_LOS =discharge|X_s =intermediate event) E(LOS|X_s =initial state)
`phi3`	weighted average of the difference between phi3.case and phi3.control
`phi3.case`	estimates E(LOS 1(X_LOS =death)|X_s =intermediate event)
`phi3.control`	P(X_LOS =death|X_s =intermediate event) E(LOS|X_s =initial state)
`empty.1`	event times with the group 'intermediate, but no terminal event yet' being empty
`empty.0`	event times with the group 'no intermediate or terminal event yet' being empty
`weights`	weights for the weighted average
`w.times`	time points corresponding to the weights
`called`	how the function was called
`patients`	total number of observed patients
`patients.discharge`	number of patients being discharged
`patients.death`	number of patients who die
`patients.cens`	number of patients being censored, i. e. for whom neither discharge or death was observed
`cases`	number of patients who experienced the IE
`cases.discharge`	number of patients who experienced the IE being discharged
`cases.death`	number of patients who experienced the IE and died
`cases.cens`	number of patients who experienced the IE and were censored

Note

required packages: survival

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

See Also

`clos`

`los.data` *Length of hospital stay*

Description

The `los.data` data frame has 756 rows, one row for each patient, and 7 columns.

Usage

```
data(los.data)
```

Format

A data frame with the following columns:

adm.id admision id of the patient

j.01 observed time for jump from 0 (initial state) to 1 (intermediate state)

j.02 observed time for jump from 0 to 2 (discharge)

j.03 observed time for jump from 0 to 3 (death)

j.12 observed time for jump from 1 to 2

j.13 observed time for jump from 1 to 3

cens censoring time (either in initial or intermediate state)

Examples

```
data(los.data)
```

msmodel *multi-state model*

Description

Makes a 'msmodel' - object to describe a 'multi-state model'

Usage

```
msmodel(state.names, tra, cens.name)
```

Arguments

state.names	character vector of the state names
tra	quadratic matrix of locigal values. TRUE/FALSE: transition is/is not possible
cens.name	character string, name of the censoring variable

Value

An object of the class 'msmodel'. The object is a list of

msmodel$tra	quadratic matrix of locigal values. TRUE/FALSE: transition is/is not possible
msmodel$states	numeric vector, the set of *names* of the states. The internal representation will be numbers 1, 2, 3, ...; if censoring occurrs, the highest number will be the censoring code
msmodel$state.names	character vector of the statenames
msmodel$transitions	matrix with two columns, 1.column: state 'from', 2.column: state 'to',the number of rows is the number of possibles transitions

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

Examples

```
my.model <- msmodel(c("0","1","2","3"),cens.name="cens")
```

plot.aj *Plot function for objects of class 'aj'*

Description

Draws a plot for an object of class 'aj' in the active graphics device: Transition probabilities are plotted as a function of time. The time origin is determined by the object of the class aj.

Usage

```
## S3 method for class 'aj'
plot(x,
        from,
        to,
        xlab=expression(paste(Time, " ", italic(t))),
        ylab= eval(substitute(expression(paste("Estimate of ",
                   P[{a}][{b}], "(", italic(s), ",",  italic(t), ")")),
                   list(a=from[1],b=to[1],s=x$start))),
        xlim = c(x$start,max(x$times)),
        ylim=c(0,1),
        lab=c(10,10,7),
        txt=eval(substitute(expression(paste(hat(P)[{a}][{b}],
                 "(",italic(s), ",", italic(t), ")")),
                 list(a=from[1],b=to[1],s=x$start))),
        x.txt=(xlim[2]+xlim[1])/2,
        y.txt=ylim[2]*0.9,
        col=1, ...)
```

Arguments

x	an object of the class 'aj'
from	a character vector naming the states 'from'
to	a character vector naming the states 'to'
xlab	a title for the x axis
ylab	a title for the y axis
xlim	the x limits (min,max) of the plot
ylim	the y limits (min,max) of the plot
lab	A numerical vector of the form 'c(x, y, len)' which modifies the way that axes are annotated. The values of 'x' and 'y' give the (approximate) number of tickmarks on the x and y axes and 'len' specifies the label size.
txt	one or more character strings or expressions specifying a text to be written.
x.txt	the x co-ordinates to be used to position the text
y.txt	the y co-ordinate to be used to position the text
col	the color of the line(s)
...	other graphical arguments

Value

A matrix with ncol = 1 + length(from):

`column 1:` vector of the timepoints, the x-cooridinates

`column 2 to column ncol:`
vector of the estimated transition probabilities, the y-coordinates

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

See Also

aj

Examples

```
data(los.data)
my.observ <- prepare.los.data(x=los.data)

my.model <- msmodel(c("0","1","2","3"),cens.name="cens")
my.trans <- trans(model=my.model,observ=my.observ)
my.aj <- aj(my.trans, s=0, t=80)
plot(my.aj,c("0","0","0","0"),c("0","1","2","3"))
```

---

plot.clos *Plot function for objects of class 'clos'*

Description

Draws a plot for an object of class 'clos' in the active graphics device: It illustrates the estimated expected change in LOS associated with an intermediate event.

Usage

```
## S3 method for class 'clos'
plot(x, opt=0,
          xlab=expression(paste(Time, " ", italic(t))),
          ylab.1="Expected LOS",ylab.2="Weights",
          xlim = c(0,max(x$trans$times[!is.na(x$e.given.1) |
                                       !is.na(x$e.given.0)])),
          xlim.2 = c(0,max(x$trans$times[!is.na(x$phi2.case) |
                                         !is.na(x$phi2.control) |
                                         !is.na(x$phi3.case) |
                                         !is.na(x$phi3.control)])),
          ylim.0=c(0,max(x$weights,na.rm=TRUE)),
          ylim.1=c(0,max(x$e.given.1,x$e.given.0,na.rm=TRUE)),
          ylim.2=c(0,max(x$x$phi2.case,x$phi2.control,
```

```
                x$phi3.case,x$phi3.control, na.rm=TRUE)),
col1=c(1,2),col2=c(1), lty1=c(1,1), lty2=c(1), lwd1=c(2,2), lwd2=c(2),
 lab.1=c(10,10,7), lab.2=c(10,3,7),
 lgd.1=expression(
    paste(Intermediate, " ", event, " ", by, " ", time, " ",italic(t)),
     paste(No, " ", intermediate, " ", event, " ", by, " ",
           time, " ",italic(t))),
 lgd.2=expression(
     paste(Case, " ", term, " ", by, " ",  time, " ",
           italic(t),", ",  patients, " ", discharged),
     paste(Control, " ", term, " ", by, " ", time, " ",
           italic(t),", ",  patients, " ", discharged)),
 lgd.3=expression(
     paste(Case, " ", term, " ", by, " ", time, " "
           ,italic(t),", ",  patients, " ", deceased),
     paste(Control, " ", term, " ", by, " ", time, " ",
           italic(t),", ",  patients, " ", deceased)),
 x.lgd=0,
 y.lgd.1=ylim.1[2]*0.9,
 y.lgd.2=ylim.2[2]*0.9,
 bty.lgd="n",
 cexlab=1,
 cexleg=1, ...)
```

Arguments

`x`	an object of class 'clos'
`opt`	numeric with the following valid values: **0** plots weights and expected LOS (the default) **1** plots expected LOS **2** plots weights **3** plots phi2 and phi3 **4** plots phi2 **5** plots phi3
`xlab`	a title for the x axis
`ylab.1`	a title for the y axis in the plot of the expected LOS
`ylab.2`	a title for the y axis in the plot of the weights
`xlim`	the x limits (min,max) of the plot of the expected LOS and of the weights
`xlim.2`	the x limits (min,max) of the plot of phi2 and phi3
`ylim.0`	the y limits of the plot of the weights
`ylim.1`	the y limits of the plot of the expected LOS
`ylim.2`	the y limits of the plot of phi2 and phi3
`col1`	the color of the lines in the plot of the expected LOS, phi2 and phi3
`col2`	the color of the line in the plot of the weights

`lty1`	the line type of the lines in the plot of the expected LOS, phi2 and phi3
`lty2`	the line type of the line in the plot of the weights
`lwd1`	the positive line width of the lines in the plot of the expected LOS, phi2 and phi3
`lwd2`	the positive line width of the line in the plot of the weights
`lab.1`	A numerical vector of the form 'c(x, y, len)' which modifies the way that axes are annotated. The values of 'x' and 'y' give the (approximate) number of tickmarks on the x and y axes and 'len' specifies the label size. Plot of the expected LOS, phi2 and phi3
`lab.2`	like lab.1, but for the plot of the weights
`lgd.1`	a vector of text values or an 'expression' to appear in the legend of the plot of the expected LOS
`lgd.2`	a vector of text values or an 'expression' to appear in the legend of the plot of phi2
`lgd.3`	a vector of text values or an 'expression' to appear in the legend of the plot of phi3
`x.lgd`	the x co-ordinate to be used to position the legend of the plot of the expected LOS, phi2 and phi3
`y.lgd.1`	the y co-ordinate to be used to position the legend of the the plot of the expected LOS
`y.lgd.2`	the y co-ordinate to be used to position the legend of the the plot of phi2 and phi3
`bty.lgd`	the type of box to be drawn around the legend. The allowed values are 'n' (the default) and 'o'.
`cexlab`	The magnification to be used for x and y labels relative to the current.
`cexleg`	character expansion factor used for the legend
`...`	other graphical arguments

Details

By default, two graphs are drawn. In a lower graph, the expected LOS given the intermediate event has (not yet) occurred by the time running on the x-axis is plotted. The difference between these two curves equals the change in LOS associated with the intermediate event *by the time* running on the x-axis. I. e. the difference between the two curves is the estimator of E(LOS|X_s =intermediate event) - E(LOS|X_s =initial state), where where X_s denotes the state by time s.

The estimated change in LOS is then computed as an weighted average over all these differences; the weights are illustrated in the upper plot. The weights derive from the estimated waiting time distribution in the initial state.

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

See Also

clos

Examples

```
data(los.data)
my.observ <- prepare.los.data(x = los.data)
tra <- matrix(FALSE, 4, 4)
diag(tra) <- TRUE
tra[1, ] <- TRUE
tra[2, 3:4] <- TRUE
my.model <- msmodel(c("0", "1", "2", "3"), tra, cens.name = "cens")
los <- clos(model = my.model, observ = my.observ)
plot(los, xlim=c(0,80), ylim.1=c(0,120))
## same plot with nicer y-axis in the lower graph
plot(los, xlim=c(0,80), ylim.1=c(0,120), lab.1=c(8,6,7))
```

---

plot.progdismodel *Plot function for objects of class 'progdismodel'*

Description

Draws three plots for an object of class 'progdismodel' in the active graphics devices or to the passed postscript files.

Usage

```
## S3 method for class 'progdismodel'
plot(x,
                  file1, file2, file3,
                  lwd=2, cex=1.2,
                  lty1=1, lty2=1, lty3=1,
                  color1=4, color2=1, color3=2, ... )
```

Arguments

`x`	an object of the class 'aj'
`file1`	ps file name for the first plot (mortality)
`file2`	ps file name for the second plot(attributable mortality)
`file3`	ps file name for the third plot (population attributable mortality)
`lwd`	the line widths
`cex`	character expansion factor
`lty1`	the line type of line 1 of the first plot
`lty2`	the line type of line 2 of the second plot
`lty3`	the line type of line 3 of the third plot
`color1`	the color of line 1 of the first plot
`color2`	the color of line 2 of the second plot
`color3`	the color of line 3 of the third plot
`...`	other graphical arguments

Details

The first plot shows $P(death, t)$, $P(death|riskfactorabsent, t)$ and $P(death|riskfactorpresent, t)$: mortality.

The second plot shows $P(death|riskfactorpresent, t) - P(death|riskfactorabsent, t)$: attributable mortality.

The third plot shows $(P(death, t) - P(death|riskfactorabsent, t))/P(death, t)$: population attributable mortality.

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

Examples

```
data(los.data)

p <- prepare.progdismodel(data=los.data)

pdm <- progdismodel(p$model, p$observ)

plot(pdm)
```

---

prepare.los.data *Prepare the data for clos*

Description

Prepare data as the ones generated by `read.los.data` to be passed to clos().

Usage

```
prepare.los.data(x)
```

Arguments

`x` data.frame of the form data.frame(id, j.01, j.02, j.03, j.12, j.13, cens):

- **id:** id (patient id, admision id)
- **j.01:** observed time for jump from 0 to 1
- **j.02:** observed time for jump from 0 to 2
- **j.03:** observed time for jump from 0 to 3
- **j.12:** observed time for jump from 1 to 2
- **j.13:** observed time for jump from 1 to 3
- **cens:** censoring time (either in initial or intermediate state)

Value

a data.frame of the form data.frame(id, from, to, time, oid):

`id:`	id (patient id, admision id)
`from:`	the state from where a transition occurs
`to:`	the state to which a transition occurs
`time:`	time of the transition
`oid:`	the observation id

Note

For bootstrap applications (see the examples to `clos`), where an individual may enter the new bootstrap data set several times, it is useful to have 'oid', which will tell several draws of the same individual apart.

Author(s)

Matthias Wangler `<mw@imbi.uni-freiburg.de>`

See Also

`clos`, `read.los.data`

Examples

```
data(los.data)
my.observ <- prepare.los.data(x=los.data)
```

`prepare.progdismodel` *Prepare the data and the model for progdismodel*

Description

Prepare the model and the data as the ones generated by `read.los.data` to be passed to progdismodel().

Usage

```
prepare.progdismodel(data)
```

Arguments

`data` data.frame of the form data.frame(id, j.01, j.02, j.03, j.12, j.13, cens):

- **id:** id (patient id, admision id)
- **j.01:** observed time for jump from 0 to 1
- **j.02:** observed time for jump from 0 to 2
- **j.03:** observed time for jump from 0 to 3
- **j.12:** observed time for jump from 1 to 2
- **j.13:** observed time for jump from 1 to 3
- **cens:** censoring time (either in initial or intermediate state)

Value

A list of:

`observ` a data.frame of the form data.frame(id, from, to, time, oid):

- **id:** id (patient id, admision id)
- **from:** the state from where a transition occurs
- **to:** the state to which a transition occurs
- **time:** time of the transition
- **oid:** the observation id

`model` an object of class 'msmodel', which describes the progressive disability model with two transient and four absorbing states.

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

See Also

prepare.los.data, read.los.data

Examples

```
data(los.data)
p <- prepare.progdismodel(data=los.data)
```

---

print.aj *Print method for objects of class 'aj'*

Description

Print method for objects of class `aj`.

Usage

```
## S3 method for class 'aj'
print(x, ...)
```

Arguments

x	an object of class 'aj'
...	other arguments

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

See Also

aj

Examples

```
data(los.data)
my.observ <- prepare.los.data(x=los.data)
my.model <- msmodel(c("0","1","2","3"),cens.name="cens")
my.trans <- trans(model=my.model,observ=my.observ)
my.aj <- aj(my.trans,s=0,t=80)
print(my.aj)
```

---

print.clos *Print method for objects of class 'clos'*

Description

Print method for objects of class clos.

Usage

```
## S3 method for class 'clos'
print(x, ...)
```

Arguments

x	An object of class clos.
...	other arguments

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

See Also

clos

Examples

```
data(los.data)
my.observ <- prepare.los.data(x=los.data)
trans <- matrix(FALSE,4,4)
diag(trans) <- TRUE
trans[1,] <- TRUE
trans[2,3:4] <- TRUE
my.model <- msmodel(c("0","1","2","3"),trans,cens.name="cens")
los <- clos(model=my.model,observ=my.observ)
print(los)
```

---

print.msmodel — *Print method for objects of class 'msmodel'*

Description

Print method for objects of class `msmodel`.

Usage

```
## S3 method for class 'msmodel'
print(x, ...)
```

Arguments

`x`	An object of class `msmodel`.
`...`	other arguments

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

Examples

```
my.model <- msmodel(c("0","1","2","3"),cens.name="cens")
print(my.model)
```

print.progdismodel — *Print method for objects of class 'progdismodel'*

Description

Print method for objects of class `progdismodel`.

Usage

```
## S3 method for class 'progdismodel'
print(x, ...)
```

Arguments

`x`	An object of class `progdismodel`.
`...`	other arguments

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

Examples

```
data(los.data)

p <- prepare.progdismodel(data=los.data)

pdm <- progdismodel(p$model, p$observ)

print(pdm)
```

print.trans — *Print method for objects of class 'trans'*

Description

Print method for objects of class `trans`.

Usage

```
## S3 method for class 'trans'
print(x, ...)
```

Arguments

`x`	An object of class `trans`.
`...`	other arguments

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

Examples

```
data(los.data)
my.observ <- prepare.los.data(x=los.data)
my.model <- msmodel(c("0","1","2","3"),cens.name="cens")
my.trans <- trans(model=my.model,observ=my.observ)
summary(my.trans)
```

---

progdismodel — *progressive disability model*

Description

Change in LOS and impact of an intermediate event on mortality can also be investigated in a so-called progressive disability model. This multi-state model can be described and the Aalen-Johansen estimator for transition probabilities can be computed. Further will be computed the mortality, attributable mortality and the population attributable mortality.

Usage

```
progdismodel(model,observ, max.time)
```

Arguments

`observ`	a data.frame of the form data.frame(id, from, to, time, oid): **id:** id (patient id, admision id) **from:** the state from where a transition occurs **to:** the state to which a transition occurs **time:** time of the transition **oid:** the observation id
`model`	an object of class 'msmodel', which describes the progressive disability model with two transient and four absorbing states.
`max.time`	the last time point of interest

Details

mortality: $P(death, t)$, $P(death|riskfactorabsent, t)$ and $P(death|riskfactorpresent, t)$.

attributable mortality: $P(death|riskfactorpresent, t) - P(death|riskfactorabsent, t)$.

population attributable mortality: $(P(death, t) - P(death|riskfactorabsent, t))/P(death, t)$.

Value

A list of

trans	an object of class trans
aj	an object of class aj
times.par	the time points for which the following values are computed.
PAR	population attributable mortality
AR	attributable mortality
death	$P(death, t)$
death.given.rfa	$P(death \vert riskfactorabsent, t)$
death.given.rfp	$P(death \vert riskfactorpresent, t)$

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

Examples

```
data(los.data)

p <- prepare.progdismodel(data=los.data)

pdm <- progdismodel(p$model, p$observ)
```

---

read.los.data *Read the data for clos*

Description

Read data fom a textfile, which contains one row per individual and one time variable per possible transition.

Usage

```
read.los.data(file, sep = ";", header = TRUE, row.names=NULL,
pos.id=1, pos.columns)
```

Arguments

`file`	name of the file to be read. If it does not contain an absolute path, the file name is relative to the current working directory, `getwd()`.
`sep`	the field separator character
`header`	a logical value indicating whether the file contains the names of the variables as its first line
`row.names`	a vector of row names. This can be a vector giving the actual row names, or a single number giving the column of the table which contains the row names, or character string giving the name of the table column containing the row names. If there is a header and the first row contains one fewer field than the number of columns, the first column in the input is used for the row names. Otherwise if 'row.names' is missing, the rows are numbered. Using 'row.names = NULL' forces row numbering.
`pos.id`	the position of the unique id (patient id, admision id)
`pos.columns`	the positions of the columns which are holding the observed times:

pos.columns[1 :] transition from initial state to intermediate state

pos.columns[2 :] transition from initial state to absorbing state (discharge)

pos.columns[3 :] transition from initial state to competing absorbing state (death)

pos.columns[4 :] transition from intermediate state to absorbing state

pos.columns[5 :] transition from intermediate state to competing absorbing state

pos.columns[6 :] censoring time (either in initial or intermediate state)

Details

The data textfile to be read must contain one time variable per possible transition and one row per individual. An additional variable contains censoring times. If a transition was *not* observed for an individual, the respective row of the textfile has entry `Inf` ('infinite') for that variable.

This data structure is well suited for bootstrapping with respect to the number of individuals. It is still concise for the four-state-model for change in LOS; note that this model does not allow for backward transitions. However, for a larger number of possible transitions, this data structure will be less desirable.

Value

a data frame of the form data.frame(id,j.01,j.02,j.03,j.12,j.13,cens)

`id`	id (patient id, admision id)
`j.01`	observed time for jump from 0 (initial state) to 1 (intermediate state)
`j.02`	observed time for jump from 0 to 2 (discharge)
`j.03`	observed time for jump from 0 to 3 (death)
`j.12`	observed time for jump from 1 to 2
`j.13`	observed time for jump from 1 to 3
`cens`	censoring time (either in initial or intermediate state)

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

See Also

clos, prepare.los.data

Examples

```
## Locate data file "los.data.csv" in sub-directory of package "changeLOS"
filename <- paste(searchpaths()[seq(along=search())[search()==
                  "package:changeLOS"]], "/data/los.data.csv", sep="")

los.data <- read.los.data(filename,pos.id=1,pos.columns=c(2,3,4,5,6,7))

## Results in the same data frame as: data(los.data)
```

---

summary.clos — *Summary method for objects of class 'clos'*

Description

Summary method for objects of class clos.

Usage

```
## S3 method for class 'clos'
summary(object, ...)
```

Arguments

object	An object of class clos.
...	other arguments

Value

A list is returned by the function summary.clos with the following elements:

cLOS	change in LOS
patients	total number of observed patients
patients.discharge	number of patients being discharged
patients.death	number of patients who die
patients.cens	number of patients being censored, i. e. for whom neither discharge or death was observed
cases	number of patients who experienced the intermediate event (IE)

`cases.discharge`
number of patients who experienced the IE being discharged

`cases.death` number of patients who experienced the IE and died

`cases.cens` number of patients who experienced the IE and were censored

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

See Also

`clos`

Examples

```
data(los.data)
my.observ <- prepare.los.data(x=los.data)
trans <- matrix(FALSE,4,4)
diag(trans) <- TRUE
trans[1,] <- TRUE
trans[2,3:4] <- TRUE
my.model <- msmodel(c("0","1","2","3"),trans,cens.name="cens")
los <- clos(model=my.model,observ=my.observ)
summary(los)
```

`summary.trans` *Summary method for objects of class 'trans'*

Description

Summary method for objects of class `trans`.

Usage

```
## S3 method for class 'trans'
summary(object, ...)
```

Arguments

`object` An object of class `trans`.

`...` other arguments

Value

A list is returned by the function `summary.trans` with the following elements:

`nrtransitions:` a matrix with

- **column 1** the state from where a transition occurs
- **column 2** the state to which a transition occurs
- **column 3** the number of transitions

`state.names:` vector with the names of the states

`times` the transition times

`nr.before` matrix with the number in each state just before the transition times

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

Examples

```
data(los.data)
my.observ <- prepare.los.data(x=los.data)
my.model <- msmodel(c("0","1","2","3"),cens.name="cens")
my.trans <- trans(model=my.model,observ=my.observ)
summary(my.trans)
```

`trans` *transition matrices*

Description

computes the Aalen-Johansen estimator for the matrix of transition probabilities $P(u-, u)$ for all observed transition times u. The entry (l, m) of the matrix denotes the estimated probability that state m has been reached by time u given state l has been occupied just before time u.

Usage

```
trans(model, observ)
```

Arguments

`model` an object of the class 'msmodel' which describes the multi-state model

`observ` a data.frame of the form data.frame(id, from, to, time, oid) (see also `prepare.los.data`):

- **id** id (patient id, admision id)
- **from** the state from where a transition occurs
- **to** the state to which a transition occurs
- **time** the time a transition occurs
- **oid** the observation id

Details

The estimator for $P(u-,u)$ is described by Andersen et al. (1993) at the bottom of p. 288. Non-diagonal entries (h,j) are given as the number of observed transitions from state$h$ to state j, divided by the number of individuals in state h just prior to time u. The diagonal elements are chosen such that the sum of each row equals 1.

The Aalen-Johansen estimator for $P(s,t)$ can then be computed as matrix product of all matrices $P(u-,u)$ for all transition times u in $(s,t]$, see aj.

Value

An object of the class 'trans'. The object is a list of:

`matrices`	array of matrices $P(u-,u)$ for every transition time u
`times`	the transition times
`nrtransitions`	a matrix with **column 1** the state from where a transition occurs **column 2** the state to which a transition occurs **column 3** the number of transitions
`state.names`	vector with the names of the states
`nr.before`	matrix with the number in each state just before the transition times

Author(s)

Matthias Wangler <mw@imbi.uni-freiburg.de>

References

P Andersen, O Borgan, R Gill, and N Keiding (1993). *Statistical models based on counting processes.* New York: Springer

S Datta, and G Satten (2001). Validity of the Aalen-Johansen estimators of stage occupation probabilities and Nelson-Aalen estimators of integrated transition hazards for non-Markov models. *Statistics and Probability Letters* 55 (4), 403–411.

Andersen and Keiding (2002). Multi-state models for event history analysis. *Statistical Methods in Medical Research* 11 (2), 91–115.

See Also

msmodel, clos, aj

Examples

```
data(los.data)
my.observ <- prepare.los.data(x=los.data)
my.model <- msmodel(c("0","1","2","3"),cens.name="cens")
my.trans <- trans(model=my.model,observ=my.observ)
my.matrices <- my.trans$matrices
my.times <- my.trans$times
```

Index