Package ‘L1mstate’

October 15, 2021

Type Package
Title L1-Regularized Multi-State Models
Version 1.0.1
Date 2021-10-7
Author Xuan Dang
Maintainer Xuan Dang <xuandang11289@gmail.com>
Description Fitting the regularization path of the L1-regularized multi-state models since they can exploit sparsity structure of input. Different tuning regularization parameter methods are provided. The cumulative hazard rate estimation and the transition probability predictions can be made from the fitted models.
License GPL (>= 2)
Encoding UTF-8
Imports Rcpp (>= 1.0.1)
LinkingTo Rcpp, RcppEigen
Depends Matrix (>= 1.2-10), MASS, mstate, colorspace
RoxygenNote 6.0.1
NeedsCompilation yes
Repository CRAN
Date/Publication 2021-10-15 09:10:12 UTC

R topics documented:

L1mstate-package ......................................................... 2
coeffL1mstate ............................................................ 3
cumhazL1mstate .......................................................... 4
cv.L1mstateR .............................................................. 6
L1mstateR ................................................................. 8
plot.cumhazL1mstate ...................................................... 10
plot.L1mstateCoeff ....................................................... 12
plot.L1mstateCV ........................................................ 14
plot.probs.L1mstate ...................................................... 16
probs.L1mstate .......................................................... 18
L1mstate-package

Description

Fitting the regularization path of the L1-regularized multi-state models since they can exploit sparsity structure of input. Different tuning regularization parameter methods are provided. The cumulative hazard rate estimation and the transition probability predictions can be made from the fitted models.

Package Content

Index of help topics:

<table>
<thead>
<tr>
<th>Package Content</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1mstate-package</td>
<td>L1-Regularized Multi-State Models</td>
</tr>
<tr>
<td>coefL1mstate</td>
<td>Obtain the coefficients</td>
</tr>
<tr>
<td>cumhaz.L1mstate</td>
<td>Compute subject-specific transition hazard rates</td>
</tr>
<tr>
<td>cv.L1mstateR</td>
<td>Cross-validation for L1mstateR</td>
</tr>
<tr>
<td>L1mstateR</td>
<td>Fit multi-state models with lasso regularization</td>
</tr>
<tr>
<td>plot.cumhaz.L1mstate</td>
<td>Plot the estimated cumulative hazard rates of the multi-state model.</td>
</tr>
<tr>
<td>plot.L1mstateCV</td>
<td>Plot the cross-validation curve produced by cv.L1mstateR</td>
</tr>
<tr>
<td>plot.L1mstateCoef</td>
<td>Plots the coefficient paths of transitions produced by L1mstateR or cv.L1mstateR</td>
</tr>
<tr>
<td>plot.probs.L1mstate</td>
<td>Plot the transition probabilities</td>
</tr>
<tr>
<td>probs.L1mstate</td>
<td>Compute subject-specific or overall transition probabilities</td>
</tr>
</tbody>
</table>

Maintainer

Xuan Dang <xuandang11289@gmail.com>

Author(s)

Xuan Dang
**Description**

Obtain the optimal coefficients using the first cross-validation method or the penalized cross-validation method by setting \( s = \text{"lambda.min"} \) or \( \text{"lambda.pcvl"} \). Can also obtain the coefficient at any lambda value within the range of sequence.

**Usage**

\[
\text{coef}l1mstate(\text{object}, s=c(\text{"lambda.pcvl"}, \text{"lambda.min"}))
\]

**Arguments**

- **object**: fitted cv.l1mstateR object
- **s**: lambda value (numeric type) or cross-validation methods (character type)

**Details**

Return the coefficient values

**Value**

Return the coefficient values.

**Author(s)**

Xuan Dang <xuandang11289@gmail.com>

**Examples**

```r
set.seed(1001)
p <- 9
times <- 1:p
rho <- 0.5
H <- abs(outer(times, times, "-"))
C <- 1 * rho^H
C[cbind(1:p, 1:p)] <- C[cbind(1:p, 1:p)]
sigma <- matrix(C, p, p)
mu <- rep(0, p)

beta12 <- c(-.65, -.65, -.65, 0, 0, -.65, -.65, 0, 0)
beta13 <- c(-.65, -.65, 0, 0, 0, -.65, -.65, 0, 0)
beta23 <- c(0, -.65, -.65, 0, 0, -.65, -.65, 0, -.65)

N <- 200
x <- mvrnorm(n=N, mu, sigma)
col_names <- c(sprintf("X%d", seq(1, dim(x)[2])))
```
cumhaz.l1mstate

Compute subject-specific transition hazard rates

Description

This function computes subject-specific cumulative transition hazard rates for each of the possible transitions in the multi-state model.

Usage

cumhaz.l1mstate(object, longdt, newdata, cv.method = c("pcvl", "min"))

Arguments

- **object**: fitted cv.l1mstateR object.
- **longdt**: long-format data input
- **newdata**: a new data with the same long-format with the same covariate names as longdt.
- **cv.method**: the cross-validation method used to select the optimal result.
Value

time A list of all time points for each of the transitions in the multi-state model.

baseHaz A list of the estimated subject-specific baseline hazards for each of the transitions in the multi-state model.

var A list of the variance of the estimated subject-specific hazards for each of the transitions in the multi-state model.

cumbaseHaz A list of the estimated subject-specific cumulative hazards for each of the transitions in the multi-state model.

Haz A list of the estimated subject-specific hazards for each of the transitions in the multi-state model.

Author(s)

Xuan Dang <xuandang11289@gmail.com>

Examples

library(L1mstate)
set.seed(1001)

N <- 200

x1 <- rbinom(N, 1, .5)
x2 <- rbinom(N, 1, .5)
x3 <- rbinom(N, 1, .5)
x4 <- rbinom(N, 1, .5)
x5 <- rbinom(N, 1, .5)
x6 <- rbinom(N, 1, .5)
x <- data.frame(x1,x2,x3,x4,x5,x6)
col_names <- c(sprintf("X%d", seq(1,6)))
colnames(x) <- col_names

beta12 <- c(-.65,-.65,-.65,0,0,0)
beta13 <- c(-.65,0,0,0,-.65,0)
beta23 <- c(0,-.65,-.65,0,0,-.65)

N12 <- N-2*N%/%5
N13 <- N-N12
hx12 <- exp(as.matrix(x[1:N12,]) %*% beta12)
hy12 <- rexp(N12,hx12)
hy13 <- exp(as.matrix(x[(N12+1):N,]) %*% beta13)
hy13 <- rexp(N13,hy13)
hx23 <- exp(as.matrix(x[1:N12,]) %*% beta23)
hy23 <- rexp(N12,hx23)
t2 <- rep(0,N)
t2[(N12+1):N] <- hy13
s2 <- rep(0,N)
s2[1:N12] <- rep(1,N12)
t3 <- rep(0,N)
cv.l1mstateR <- 

t3[1:N12] <- ty12+ty23
t3[(N12+1):N] <- ty13

censor case for state 3

tcens3 <- rbinom(n=N, prob = 0.3, size = 1)
s3 <- 1-tcens3

# pre-process data into long-format using mssprep
dt <- data.frame(illt=t2, ills=s2, dt=t3, ds=s3, x)
tmat <- matrix(c(NA,NA,NA,1,NA,NA,2,3,NA), nrow = 3)
longdt <- mssprep(time=c(NA,"illt","dt"),status=c(NA,"ills","ds"),
keep = col_names, data=dt,trans=tmat)

out <- cv.l1mstateR(longdt, nlambda = 100, nfolds = 10)

ptsA <- longdt[which(longdt$X1==1 & longdt$X2==0 & longdt$X3==1 & longdt$X4==0
 & longdt$X5==1 & longdt$X6==1),]
## observed transitions (ground truth)

# predicted time = 0
events(ptsA)

# use models to predict the transition probabilities
L1MSTATE

ptA <- ptsA[which(ptsA$id == unique(ptsA$id)[1]),]
ptA <- ptA[,c(4,9:14)]
cumhazA <- cumhaz.l1mstate(object=out, longdt=longdt, newdata=ptA, cv.method="pcvl")

---

**cv.l1mstateR**

*Cross-validation for l1mstateR*

**Description**

Does k-fold cross-validation for l1mstateR

**Usage**

```r
cv.l1mstateR(longdt, lambda=NULL, nlambda=100, rlambda=NULL, nfolds=1, foldid=NULL, thresh=1e-7, maxit=1e+5)
```

**Arguments**

- **longdt**: input in long-format structure
- **lambda**: A user supplied lambda sequence. Typical usage is to have the program compute its own lambda sequence based on nlambda and rlambda. Supplying a value of lambda overrides this. **WARNING**: use with care. Avoid supplying a single value for lambda. Supply instead a decreasing sequence of lambda values. l1mstateR relies on its warms starts for speed, and its often faster to fit a whole path than compute a single fit.
- **nlambda**: The number of lambda values- default is 100.
cv.llmsTateR

**rlambda**
Smallest value for lambda, as a fraction of the maximum lambda, the (data derived) entry value (i.e. the smallest value for which all coefficients are zero). The default depends on the sample size relative to the number of covariates. If sample size > #covariates, the default is 0.001, close to zero. If sample size > #covariates, the default is 0.01.

**nfolds**
Number of folds - default is 10. Smallest value allowable is nfolds=3.

**foldid**
an optional vector of values between 1 and nfolds identifying what fold each observation is in.

**thresh**
Convergence threshold for one-step coordinate descent. Defaults value is 1E-7.

**maxit**
Maximum number of passes over the data for all lambda values; default is 1E+5.

**Value**

- **aBetaSTD** A list of coefficients in standardized form, each one corresponds to each lambda value.
- **aBetaO** A list of coefficients in original form, each one corresponds to each lambda value.
- **pBetaSTD** The coefficient in standardized form gives maximum log-likelihood value using the penalized cross-validation method.
- **pBetaO** The coefficient in original form gives maximum log-likelihood value using the penalized cross-validation method.
- **mBetaSTD** The coefficient in standardized form gives maximum log-likelihood value using the first cross-validation method.
- **mBetaO** The coefficient in original form gives maximum log-likelihood value using the first cross-validation method.
- **fit** A matrix includes lambda value, the mean cross-validation error.
- **fit** A matrix of lambda values and log-likelihood values
- **numcovs** Number of covariates
- **numtrans** Number of transitions

**Author(s)**

Xuan Dang <xuandang11289@gmail.com>

**Examples**

```r
set.seed(1001)
p <- 9
times <- 1:p
rho <- 0.5
H <- abs(outer(times, times, "-"))
C <- 1 * rho^H
C[cbind(1:p, 1:p)] <- C[cbind(1:p, 1:p)]
sigma <- matrix(C,p,p)
mu <- rep(0,p)
```
beta12 <- c(-.65, -.65, -.65, 0, 0, -.65, -.65, 0, 0)
beta13 <- c(-.65, -.65, 0, 0, 0, 0, -.65, 0, 0)
beta23 <- c(0, -.65, -.65, 0, 0, -.65, -.65, 0, -.65)

N <- 200
x <- mvrnorm(n=N, mu, sigma)
col_names <- c(sprintf("X%d", seq(1,dim(x)[2])))
colnames(x) <- col_names
N12 <- N-2*N%/%5
N13 <- N-N12
hx12 <- exp(x[1:N12,] %*% beta12)
ty12 <- rexp(N12,hx12)
hx13 <- exp(x[(N12+1):N,] %*% beta13)
ty13 <- rexp(N13,hx13)
hx23 <- exp(x[1:N12,] %*% beta23)
ty23 <- rexp(N12,hx23)
t2 <- rep(0,N)
t2[1:N12] <- ty12
t2[(N12+1):N] <- ty13
s2 <- rep(0,N)
s2[1:N12] <- rep(1,N12)
t3 <- rep(0,N)
t3[1:N12] <- ty12+ty23
t3[(N12+1):N] <- ty13
#censor case for state 3
tcens3 <- rbinom(n=N, prob = 0.3, size = 1)
s3 <- 1-tcens3

# pre-process data into long-format using msprep
dt <- data.frame(illt=t2, ills=s2, dt=t3, ds=s3, x)
tmat <- matrix(c(NA,NA,NA,1,NA,NA,2,3,NA), nrow = 3)
longdt <- msprep(time=c(NA,"illt","dt"),status=c(NA,"ills","ds"),
keep = col_names, data=dt,trans=tmat)
cv.l1fit <- cv.l1mstateR(longdt, nlambda = 100, nfolds = 10)

---

**l1mstateR**

**Fit multi-state models with lasso regularization**

**Description**

Fit a multi-state models via penalized partial likelihood. The regularization path is computed for the lasso at a path of values for the regularization parameter lambda. Can deal with right-censoring and left-truncated data.

**Usage**

l1mstateR(longdt, lambda=NULL, nlambda=100, rlambda=NULL, thresh=1e-7, maxit=1e+5)
Arguments

longdt input in long-format structure

lambda A user supplied lambda sequence. Typical usage is to have the program compute its own lambda sequence based on nlambdas and rlambdas. Supplying a value of lambda overrides this. WARNING: use with care. Avoid supplying a single value for lambda. Supply instead a decreasing sequence of lambda values. 11mstateR relies on its warms starts for speed, and it's often faster to fit a whole path than compute a single fit.

nlambdas The number of lambda values - default is 100.

rlambdas Smallest value for lambda, as a fraction of the maximum lambda, the (data derived) entry value (i.e. the smallest value for which all coefficients are zero). The default depends on the sample size relative to the number of covariates. If sample size > #covariates, the default is 0.001, close to zero. If sample size > #covariates, the default is 0.01.

thresh Convergence threshold for one-step coordinate descent. Defaults value is 1E-7.

maxit Maximum number of passes over the data for all lambda values; default is 1E+5.

Details

Please note that input has to be long-format structure.

Value

aBetaSTD A list of coefficients in standardized form, each one corresponds to each lambda value.

aBetaO A list of coefficients in original form, each one corresponds to each lambda value.

fit A matrix of lambda values and log-likelihood values

numcovs Number of covariates

numtrans Number of transitions

Author(s)

Xuan Dang <xuandang11289@gmail.com>

Examples

set.seed(1001)
p <- 9
times <- 1:p
rho <- 0.5
H <- abs(outer(times, times, "-"))
C <- 1 * rho^H
C[cbind(1:p, 1:p)] <- C[cbind(1:p, 1:p)]
sigma <- matrix(C, p, p)
mu <- rep(0, p)
beta12 <- c(-.65,-.65,-.65,0,0,-.65,-.65,0,0)
beta13 <- c(-.65,-.65,0,0,0,0,-.65,0,0)
beta23 <- c(0,-.65,-.65,0,0,-.65,-.65,0,-.65)

N <- 200
x <- mvrnorm(n=N, mu, sigma)
col_names <- c(sprintf("X%d", seq(1,dim(x)[2])))
colnames(x) <- col_names
N12 <- N-2*N%/%5
N13 <- N-N12
hx12 <- exp(x[1:N12,] %*% beta12)
ty12 <- rexp(N12,hx12)
hx13 <- exp(x[(N12+1):N,] %*% beta13)
ty13 <- rexp(N13,hx13)
hx23 <- exp(x[1:N12,] %*% beta23)
ty23 <- rexp(N12,hx23)
t2 <- rep(0,N)
t2[1:N12] <- ty12
t2[(N12+1):N] <- ty13
s2 <- rep(0,N)
s2[1:N12] <- rep(1,N12)
t3 <- rep(0,N)
t3[1:N12] <- ty12+ty23
t3[(N12+1):N] <- ty13
#censor case for state 3
tcens3 <- rbinom(n=N, prob = 0.3, size = 1)
s3 <- 1-tcens3

# pre-process data into long-format using msprep
dt <- data.frame(illt=t2, ills=s2, dt=t3, ds=s3, x)
tmat <- matrix(c(NA,NA,NA,1,NA,NA,2,3,NA), nrow = 3)
longdt <- msprep(time=c(NA,"illt","dt"),status=c(NA,"ills","ds"),
keep = col_names, data=dt,trans=tmat)

l1fit <- l1mstateR(longdt, nlambda = 100)

---

**plot.cumhaz.l1mstate**  
Plot the estimated cumulative hazard rates of the multi-state model.

**Description**
Plot the estimated cumulative hazard rates of the multi-state model.

**Usage**

```r
## S3 method for class 'cumhaz.l1mstate'
plot(x,type=c("single","separate"),cols,
  xlab="Years since transplant",ylab="Cumulative hazard",
  ylim,lwd=3,lty,legend,legend.pos,bty="o",...)
```
Arguments

x      fitted cumhaz.1mstate object

type   One of "single" or "separate"; in case of "single", all estimated cumulative hazards are drawn in a single plot, in case of "separate", separate plots are shown for the estimated cumulative hazards.

cols   A vector specifying colors for the different transitions

xlab   A title for the x-axis; default is "Years since transplant"

ylab   A title for the y-axis; default is "Cumulative hazard"

ylim   The y limits of the plot

lwd    The line width; default is 3

lty    The line type

legend The transition numbers; if missing, these will be taken from the transition matrix contained in cumhaz.1mstate object.

legend.pos The position of the legend; default is "topleft"

bty    The box type of the legend

...   Further arguments to plot

Details

A plot is produced, and nothing is returned.

Value

No return value

Author(s)

Xuan Dang <xuandang11289@gmail.com>

Examples

library(L1mstate)
set.seed(1001)

N <- 200

x1 <- rbinom(N, 1, .5)
x2 <- rbinom(N, 1, .5)
x3 <- rbinom(N, 1, .5)
x4 <- rbinom(N, 1, .5)
x5 <- rbinom(N, 1, .5)
x6 <- rbinom(N, 1, .5)
x <- data.frame(x1,x2,x3,x4,x5,x6)
col_names <- c(sprintf("X%d", seq(1,6)))
colnames(x) <- col_names

beta12 <- c(-.65,-.65,-.65,0,0,0)
beta13 <- c(-.65,0,0,0,-.65,0)
beta23 <- c(0,-.65,-.65,0,0,-.65)

N12 <- N-2*N%/%5
N13 <- N-N12
hx12 <- exp(as.matrix(x[1:N12,]) %*% beta12)
ty12 <- rexp(N12,hx12)
hx13 <- exp(as.matrix(x[(N12+1):N,]) %*% beta13)
ty13 <- rexp(N13,hx13)
hx23 <- exp(as.matrix(x[1:N12,]) %*% beta23)
ty23 <- rexp(N12,hx23)
t2 <- rep(0,N)
t2[1:N12] <- ty12
t2[(N12+1):N] <- ty13
s2 <- rep(0,N)
s2[1:N12] <- rep(1,N12)
t3 <- rep(0,N)
t3[1:N12] <- ty12+ty23
t3[(N12+1):N] <- ty13
# censor case for state 3
tcens3 <- rbinom(n=N, prob = 0.3, size = 1)
s3 <- 1-tcens3

# pre-process data into long-format using mprep
dt <- data.frame(illt=t2, ills=s2, dt=t3, ds=s3, x)
tmat <- matrix(c(NA,NA,NA,1,NA,NA,2,3,NA), nrow = 3)
longdt <- mprep(time=c(NA,"illt","dt"),status=c(NA,"ills","ds"),
keep = col_names, data=dt,trans=tmat)

out <- cv.l1mstateR(longdt, nlambda = 100, nfolds = 10)
ptsA <- longdt[which(longdt$X1==1 & longdt$X2==0 & longdt$X3==1 & longdt$X4==0 &
longdt$X5==1 & longdt$X6==1),]
## observed transitions (ground truth)
# predicted time = 0
events(ptsA)
# use models to predict the transition probabilities
# L1MSTATE
pta <- ptsA[which(ptsA$id == unique(ptsA$id)[1]),]
pta <- pta[,c(4,9:14)]
cumhazA <- cumhaz.l1mstate(object=out, longdt=longdt, newdata=pta, cv.method="pcvl")

plot.cumhaz.l1mstate(cumhazA, main = "L1MSTATE", type = "single", lwd=2,las=1,ylim = c(0,4))

legend("topleft",inset=.1,legend=c("1","2","3"),col=c("black","red","green"),
1ty=1,cex=0.8,title="Transitions",text.font=2,bg='white')

# Plots the coefficient paths of transitions produced by l1mstateR or cv.l1mstateR
Description

Plots the coefficient values as a function of the lambda values used.

Usage

## S3 method for class 'l1mstateCoef'
plot(x, trans = NULL,...)

Arguments

x fitted l1mstateR or cv.l1mstateR object.
trans choose the transition you want to plot the coefficient. It can be a single transition or a set of several transitions.
... further arguments to plot

Details

A plot is produced, and nothing is returned.

Value

No return value.

Author(s)

Xuan Dang <xuandang11289@gmail.com>

Examples

set.seed(1001)
p <- 9
times <- 1:p
rho <- 0.5
H <- abs(outer(times, times, "-"))
C <- 1 + rho^H
C[cbind(1:p, 1:p)] <- C[cbind(1:p, 1:p)]
sigma <- matrix(C,p,p)
mu <- rep(0,p)

beta12 <- c(-.65,-.65,-.65,0,0,-.65,-.65,0,0)
beta13 <- c(-.65,-.65,0,0,0,-.65,0,0)
beta23 <- c(0,-.65,-.65,0,0,-.65,-.65,0,-.65)

N <- 200
x <- mvrnorm(n=N, mu, sigma)
col_names <- c(sprintf("X%d", seq(1,dim(x)[2])))
colnames(x) <- col_names
N12 <- N-2*N%/%5
N13 <- N-N12
hx12 <- exp(x[1:N12,] %*% beta12)
ty12 <- rexp(N12, hx12)
hx13 <- exp(x[(N12+1):N,] %*% beta13)
ty13 <- rexp(N13, hx13)
hx23 <- exp(x[1:N12,] %*% beta23)
ty23 <- rexp(N12, hx23)
t2 <- rep(0, N)
t2[1:N12] <- ty12
t2[(N12+1):N] <- ty13
s2 <- rep(0, N)
s2[1:N12] <- rep(1, N12)
t3 <- rep(0, N)
t3[1:N12] <- ty12 + ty23
t3[(N12+1):N] <- ty13
# censor case for state 3
tcens3 <- rbinom(n=N, prob = 0.3, size = 1)
s3 <- 1 - tcens3

# pre-process data into long-format using msprep
dt <- data.frame(illt=t2, ills=s2, dt=t3, ds=s3, x)
tmat <- matrix(c(NA, NA, NA, 1, NA, NA, 2, 3, NA), nrow = 3)
longdt <- msprep(time=c(NA, "illt", "dt"), status=c(NA, "ills", "ds"),
keep = col_names, data=dt, trans=tmat)

l1fit <- l1mstateR(longdt, nlambda = 100)
plot.l1mstateCoef(l1fit, trans=1)

---

plot.l1mstateCV

plot the cross-validation curve produced by cv.l1mstateR

Description

Plots the cross-validation curve, and upper and lower standard deviation curves, as a function of the lambda values used.

Usage

## S3 method for class 'l1mstateCV'
plot(x, ...)

Arguments

x fitted cv.l1mstateR object

... Further arguments to plot

Details

A plot is produced, and nothing is returned.
Value
No return value.

Author(s)
Xuan Dang <xuandang11289@gmail.com>

Examples

```r
set.seed(1001)
p <- 9
times <- 1:p
rho <- 0.5
H <- abs(outer(times, times, "-"))
C <- I * rho^H
C[cbind(1:p, 1:p)] <- C[cbind(1:p, 1:p)]
sigma <- matrix(C, p, p)
mu <- rep(0, p)

beta12 <- c(-.65, -.65, -.65, 0, 0, -.65, -.65, 0, 0)
beta13 <- c(-.65, -.65, 0, 0, 0, 0, -.65, 0, 0)
beta23 <- c(0, -.65, -.65, 0, 0, -.65, -.65, 0, -.65)

N <- 200
x <- mvrnorm(n=N, mu, sigma)
col_names <- c(sprintf("X%d", seq(1, dim(x)[2])))
colnames(x) <- col_names
N12 <- N-2*N%/%5
N13 <- N-N12
hx12 <- exp(x[1:N12,] %*% beta12)
ty12 <- rexp(N12, hx12)
hx13 <- exp(x[(N12+1):N,] %*% beta13)
ty13 <- rexp(N13, hx13)
hx23 <- exp(x[1:N12,] %*% beta23)
ty23 <- rexp(N12, hx23)
t2 <- rep(0, N)
t2[1:N12] <- ty12
t2[(N12+1):N] <- ty13
s2 <- rep(0, N)
s2[1:N12] <- rep(1, N12)
t3 <- rep(0, N)
t3[1:N12] <- ty12+ty23
t3[(N12+1):N] <- ty13
# censor case for state 3
tcens3 <- rbinom(n=N, prob = 0.3, size = 1)
s3 <- 1-tcens3

# pre-process data into long-format using mprep
dt <- data.frame(illt=t2, ills=s2, dt=t3, ds=s3, x)
tmat <- matrix(c(NA, NA, NA, 1, NA, NA, 2, 3, NA), nrow = 3)
longdt <- mprep(time=c(NA, "illt", "dt"), status=c(NA, "ills", "ds"),
keep = col_names, data=dt, trans=tmat)
```
cv.l1fit <- cv.l1mstateR(longdt, nlambda = 100, nfolds = 10)
plot.l1mstateCV(cv.l1fit)

---

**plot.probs.l1mstate**  
*Plot the transition probabilities*

**Description**

Plot the transition probabilities produced by `probs.l1mstate`

**Usage**

```r
## S3 method for class 'probs.l1mstate'
plot(x, from, type = c("stacked", "filled", "single", "separate"),
     ord, cols, xlab = "Years since transplant", ylab = "Probability",
     xlim, ylim, lwd, lty, cex, legend, legend.pos, bty = "o", ...)
```

**Arguments**

- `x`: object produced from `probs.l1mstate`
- `from`: the starting state from which the probabilities are used to plot
- `type`: with "stacked" type, the transition probabilities are stacked and the distance between two adjacent curves indicates the probability; with "filled" type, it is the same but the space between adjacent curves are filled; with "single" type, the probabilities are shown as different curves in a single plot; with "separate" type, separate plots are shown for the estimated transition probabilities
- `ord`: A vector indicates the order of plotting in case "stacked" or "filled"
- `cols`: A vector specifying colors for the different transitions
- `xlab`: A title for the x-axis; default is "Years since transplant"
- `ylab`: A title for the y-axis; default is "Probability"
- `xlim`: The x limits of the plot; default is range of time
- `ylim`: The y limits of the plot
- `lwd`: The line width; default is 3
- `lty`: The line type
- `cex`: Character size
- `legend`: The transition numbers; if missing, these will be taken from the transition matrix contained in `cumhaz.l1mstate` object.
- `legend.pos`: The position of the legend; default is "topleft"
- `bty`: The box type of the legend
- `...`: Further arguments to `plot`
Details
A plot is produced, and nothing is returned.

Value
No return value.

Author(s)
Xuan Dang <xuandang11289@gmail.com>

Examples

library(L1mstate)
set.seed(1001)

N <- 200

x1 <- rbinom(N, 1, .5)
x2 <- rbinom(N, 1, .5)
x3 <- rbinom(N, 1, .5)
x4 <- rbinom(N, 1, .5)
x5 <- rbinom(N, 1, .5)
x6 <- rbinom(N, 1, .5)
x <- data.frame(x1,x2,x3,x4,x5,x6)
col_names <- c(sprintf("X%d", seq(1,6)))
colnames(x) <- col_names

beta12 <- c(-.65,-.65,-.65,0,0,0)
beta13 <- c(-.65,0,0,0,-.65,0)
beta23 <- c(0,-.65,-.65,0,0,-.65)

N12 <- N-2*N%/%5
N13 <- N-N12
hx12 <- exp(as.matrix(x[1:N12,]) %*% beta12)
ty12 <- rexp(N12,hx12)
hx13 <- exp(as.matrix(x[(N12+1):N,]) %*% beta13)
ty13 <- rexp(N13,hx13)
hx23 <- exp(as.matrix(x[1:N12,]) %*% beta23)
ty23 <- rexp(N12,hx23)
t2 <- rep(0,N)
t2[1:N12] <- ty12
t2[(N12+1):N] <- ty13
s2 <- rep(0,N)
s2[1:N12] <- rep(1,N12)
t3 <- rep(0,N)
t3[1:N12] <- ty12+ty23
t3[(N12+1):N] <- ty13
#censor case for state 3
tcens3 <- rbinom(n=N, prob = 0.3, size = 1)
s3 <- 1-tcens3
# pre-process data into long-format using msprep
dt <- data.frame(illt=t2, ills=s2, dt=t3, ds=s3, x)
tmat <- matrix(c(NA,NA,NA,1,NA,NA,2,3,NA), nrow = 3)
longdt <- msprep(time=c(NA,"illt","dt"),status=c(NA,"ills","ds"),
keep = col_names, data=dt,trans=tmat)

out <- cv.l1mstateR(longdt, nlambda = 100, nfolds = 10)

ptsA <- longdt[which(longdt$X1==1 & longdt$X2==0 & longdt$X3==1 & longdt$X4==0
 & longdt$X5==1 & longdt$X6==1),]
## observed transitions (ground truth)
# predicted time = 0
events(ptsA)
# use models to predict the transition probabilities
# L1MSTATE
pta <- ptsA[which(ptsA$id == unique(ptsA$id)[1]),]
pta <- pta[,c(4,9:14)]
cumhazA <- cumhaz.l1mstate(object=out, longdt=longdt, newdata=pta, cv.method="pcvl")
probA_0 <- probs.l1mstate(cumhazA, longdt = longdt, tmat, predt = 0, direction = "forward")
statecols <- heat_hcl(6, c = c(90, 10), l = c(20, 80), power = c(1/5, 2))[c(6, 5, 3, 4, 2, 1)]
ord <- c(1, 2, 3, 4, 5, 6)
plot.probs.l1mstate(probA_0, main = "L1MSTATE", from = 1, ord = ord,
las=1, cex=0, type = "filled", col = statecols[ord])

probs.l1mstate

---

### Description

Compute subject-specific or overall transition probabilities.

### Usage

```r
probs.l1mstate(object, longdt, tmat, predt, direction=c("forward","fixedhorizon"))
```

### Arguments

- **object**: fitted cumhaz.l1mstate object
- **longdt**: long-format data input
- **tmat**: a transition matrix for multi-state model
- **predt**: a prediction time
- **direction**: indicates whether prediction is forward or for a fixed horizon

### Value

A list with each element [[s]] containing a data frame with the estimated transition probabilities from state s. It also includes transition and tmat information for plotting purpose.
Examples

```r
library(L1mstate)
set.seed(1001)

N <- 200
x1 <- rbinom(N, 1, .5)
x2 <- rbinom(N, 1, .5)
x3 <- rbinom(N, 1, .5)
x4 <- rbinom(N, 1, .5)
x5 <- rbinom(N, 1, .5)
x6 <- rbinom(N, 1, .5)
x <- data.frame(x1,x2,x3,x4,x5,x6)
col_names <- c(sprintf("X%d", seq(1,6)))
colnames(x) <- col_names

beta12 <- c(-.65,-.65,-.65,0,0,0)
beta13 <- c(-.65,0,0,0,-.65,0)
beta23 <- c(0,-.65,-.65,0,0,-.65)

N12 <- N-2*N%/%5
N13 <- N-N12
hx12 <- exp(as.matrix(x[1:N12,]) %*% beta12)
ty12 <- rexp(N12,hx12)
hx13 <- exp(as.matrix(x[(N12+1):N,]) %*% beta13)
ty13 <- rexp(N13,hx13)
hx23 <- exp(as.matrix(x[1:N12,]) %*% beta23)
ty23 <- rexp(N12,hx23)
t2 <- rep(0,N)
t2[1:N12] <- ty12
t2[(N12+1):N] <- ty13
s2 <- rep(0,N)
s2[1:N12] <- rep(1,N12)
t3 <- rep(0,N)
t3[1:N12] <- ty12+ty23
t3[(N12+1):N] <- ty13
	#censor case for state 3
tcens3 <- rbinom(n=N, prob = 0.3, size = 1)
s3 <- 1-tcens3
	# pre-process data into long-format using msprep
dt <- data.frame(illt=t2, ills=s2, dt=t3, ds=s3, x)
tmat <- matrix(c(NA,NA,NA,1,NA,NA,2,3,NA), nrow = 3)
longdt <- msprep(time=c("illt","dt"),status=c("ills","ds"),
																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ptsA <- longdt[which(longdt$X1==1 & longdt$X2==0 & longdt$X3==1 & longdt$X4==0
    & longdt$X5==1 & longdt$X6==1),]
## observed transitions (ground truth)
# predicted time = 0
events(ptsA)
# use models to predict the transition probabilities
# L1MSTATE
ptA <- ptsA[which(ptsA$id == unique(ptsA$id)[1]),]
ptA <- ptA[,c(4,9:14)]
cumhazA <- cumhaz.l1mstate(object=out, longdt=longdt, newdata=ptA, cv.method="pcvl")
probA_0 <- probs.l1mstate(cumhazA, longdt = longdt, tmat, predt = 0, direction = "forward")
Index

* multi-state models
  coef.l1mstate, 3
  cumhaz.l1mstate, 4
  cv.l1mstateR, 6
  l1mstateR, 8
  plot.cumhaz.l1mstate, 10
  plot.l1mstateCoeff, 12
  plot.l1mstateCV, 14
  plot.probs.l1mstate, 16
  probs.l1mstate, 18

* package
  L1mstate-package, 2

* regularization
  coef.l1mstate, 3
  cumhaz.l1mstate, 4
  cv.l1mstateR, 6
  l1mstateR, 8
  plot.cumhaz.l1mstate, 10
  plot.l1mstateCoeff, 12
  plot.l1mstateCV, 14
  plot.probs.l1mstate, 16
  probs.l1mstate, 18

coef.l1mstate, 3
cumhaz.l1mstate, 4
cv.l1mstateR, 6

L1mstate (L1mstate-package), 2
L1mstate-package, 2
l1mstateR, 8

plot.cumhaz.l1mstate, 10
plot.l1mstateCoeff, 12
plot.l1mstateCV, 14
plot.probs.l1mstate, 16
probs.l1mstate, 18