Package ‘SPREDA’

November 25, 2018

Type Package
Title Statistical Package for Reliability Data Analysis
Version 1.1
Date 2018-11-25
Author Yili Hong, Yimeng Xie, and Zhibing Xu
Maintainer Yili Hong <yilihong@vt.edu>
Depends survival, nlme
Description The Statistical Package for REliability Data Analysis (SPREDA) implements recently-developed statistical methods for the analysis of reliability data. Modern technological developments, such as sensors and smart chips, allow us to dynamically track product/system usage as well as other environmental variables, such as temperature and humidity. We refer to these variables as dynamic covariates. The package contains functions for the analysis of time-to-event data with dynamic covariates and degradation data with dynamic covariates. The package also contains functions that can be used for analyzing time-to-event data with right censoring, and with left truncation and right censoring. Financial support from NSF and DuPont are acknowledged.
License GPL-2
NeedsCompilation no
Repository CRAN
Date/Publication 2018-11-25 17:50:03 UTC

R topics documented:

SPREDA-package ........................................ 2
ce.dat.prep ............................................ 3
cis ...................................................... 4
Coatingenv ............................................ 5
Coatingout ............................................. 5
deglmx .................................................. 6
i.spline.x ............................................. 8
kaplan.meier.location ................................ 9
Lifedata.MLE .......................................... 10
The Statistical Package for REliability Data Analysis (SPREDA) implements recently-developed statistical methods for the analysis of reliability data. Modern technological developments, such as sensors and smart chips, allow us to dynamically track product/system usage as well as other environmental variables, such as temperature and humidity. We refer to these variables as dynamic covariates. The package contains functions for the analysis of time-to-event data with dynamic covariates and degradation data with dynamic covariates. The package also contains functions that can be used for analyzing time-to-event data with right censoring, and with left truncation and right censoring. Financial support from NSF and DuPont are acknowledged.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>SPREDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.1</td>
</tr>
<tr>
<td>Date:</td>
<td>2018-11-25</td>
</tr>
<tr>
<td>License:</td>
<td>GPL-2</td>
</tr>
</tbody>
</table>

Contains functions that are useful for the analysis of reliability data.

Author(s)

Yili Hong, Yimeng Xie, and Zhibing Xu

Maintainer: Yili Hong <yilihong@vt.edu>
References

---

ce.dat.prep

Create an object for cumulative exposure

Description
Create a dataset with format that is suitable for analysis in the cumulative exposure model.

Usage
ce.dat.prep(xt.dat, failure.dat, ref_time = NULL)

Arguments
xt.dat 
"data.frame" format object. The first and second columns are sample id and time, respectively. Other columns are dynamic covariates.
failure.dat 
A "Surv" object. See Surv.
ref_time 
Reference time for each sample. If not specified, ref_time is a vector of 0's.

Value
A list of failure.dat, xt.obj and aux.inf.
**Description**

It is used to estimate the parameters which have restrictions on the domain based on mixed primal-dual bases algorithm.

**Usage**

```
cls(y, X)
```

**Arguments**

- `y`: A vector for responses.
- `X`: Matrix of covariates.

**Value**

- `y`: A vector for responses;
- `x`: Matrix of covariates;
- `betahat`: Estimates of parameters;

**Author(s)**

Yili Hong

**References**


**See Also**

`clme` and `deglmx`

**Examples**

```
y<-sort(rnorm(100, 10, 2))
x<-cbind(runif(100, 1, 3), sort(rnorm(100, 3, 1)))
res<-cls(y, x)
```
Coatingenv

Dynamic covariates for the coating data.

Description

A data frame with 36 observations and 9 variables.

Usage

data(Coatingenv)

Format

<table>
<thead>
<tr>
<th>Spec_num</th>
<th>Observation id.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>Observation time for each id.</td>
</tr>
<tr>
<td>UVB</td>
<td>a dynamic covariate.</td>
</tr>
<tr>
<td>UVA</td>
<td>a dynamic covariate.</td>
</tr>
<tr>
<td>VIS</td>
<td>a dynamic covariate.</td>
</tr>
<tr>
<td>TEMP</td>
<td>a dynamic covariate.</td>
</tr>
<tr>
<td>RH</td>
<td>a dynamic covariate.</td>
</tr>
<tr>
<td>UV</td>
<td>a dynamic covariate.</td>
</tr>
<tr>
<td>Group</td>
<td>Group for each id.</td>
</tr>
</tbody>
</table>

Source


Coatingout

Dynamic covariates for coating data

Description

A data frame with 36 observations and 4 variables.

Usage

data(Coatingout)
**Format**

- **SPEC_NUM**: Observation id.
- **TIME**: Observation time for each id.
- **DAMAGE_Y**: Response for each id.
- **GROUP**: Group for each id.

**Source**


---

**degglm**

*Functions for estimating parameters in the linear/nonlinear mixed models with dynamic covariates.*

**Description**

Functions for estimating parameters in the linear/nonlinear mixed models with dynamic covariates. Those dynamic covariates will have restricted-shape effects such as monotonic increasing, decreasing or quadratic shape.

**Usage**

```r
degglm(fixed, data, dyn.data, id, time, random = NULL, linear = TRUE, ytrend, splinetrend = NULL, splinetype = NULL, degree = NULL, knots = NULL, weights = NULL, subset = NULL, start, maxiter = 10, method = "BFGS", ...)
```

**Arguments**

- **fixed**: Formula with fixed effect.
- **data**: Data with id, time, response, and fixed covariates.
- **dyn.data**: Dynamic data with id, time, dynamic covariates.
- **id**: The name of the id which is characters or string.
- **time**: The name of time in the data or dyn.data which is characters or string.
- **random**: The formula for random parts which should condition on the id.
- **linear**: The index of linear or nonlinear.
- **ytrend**: If ytrend=1 indicates the increasing trend of the response, if ytrend=-1 indicates the decreasing trend of the response.
- **splatrend**: They are a vector of trends of dynamic covariate effects. Define 1 as increasing trend and -1 as decreasing trend.
- **splatnitype**: They are a vector of the spline basis type which can be chosen among "Ms", "Is", and "Cs".
\textbf{degImx}

\begin{tabular}{ll}
\textbf{degree} & The degree of the spline functions. \\
\textbf{knots} & The number of knots in the spline functions. \\
\textbf{weights} & Weights of the observation. \\
\textbf{subset} & Subset of the data. \\
\textbf{start} & The initial values for covariance and variance matrix. \\
\textbf{maxiter} & The maximum number of iteration in the optimization. \\
\textbf{method} & The method of \texttt{optim} function with "BFGS" as default. More details in \texttt{optim}. \\
\end{tabular}

\textbf{Value}

The returned outputs belong to class of "degImx". \texttt{list(type = type, fit = fit, dat = dat.obj, dyn.mat = cov.mat.tmp, ytrend = ytrend, dyncovnames = dyncovnames, dyn.data = dyn.data, beta.index = beta.index, call = mfun)}

\textbf{type} Type of the model either linear mixed or nonlinear mixed models.

\textbf{fit} The fitting results in the model including estimates, residuals, loglikelihood, and so on.

\textbf{dat} The modified data.

\textbf{dyn.mat} The spline basis functions.

\textbf{ytrend} The indication of response trend either increasing (1) or decreasing (-1).

\textbf{dyncovnames} Names of dynamic covariates in the model.

\textbf{dyn.data} The modified dynamic data.

\textbf{beta.index} Indications of parameters in the dynamic covariates.

\textbf{call} The call function in the model.

\textbf{Note}

For the nonlinear model, we currently only implement one specific nonlinear relationship.

\textbf{Author(s)}

Yili Hong

\textbf{References}

Examples

data(Coatingenv)
data(Coatingout)

fit=deglmx(DAMAGE_Y~UV+RH+TEMP, data=Coatingout, dyn.data=Coatingenv, id="SPEC_NUM", time="TIME", random=~TIME|SPEC_NUM, linear=TRUE, ytrend=-1, splinetrend=c(-1, -1, -1), splinetype=c("Is", "Cs", "Is"), degree=c(3, 3, 3), knots=c(4, 4, 4), weights=NULL, subset=NULL, start=c(0.017, 0.00013, -0.404, 0.0176), maxiter=10, method='BFGS')

fit=deglmx(DAMAGE_Y~G/(1+exp(-log(UV+RH+TEMP)/H)), data=Coatingout, dyn.data=Coatingenv, id="SPEC_NUM", time="TIME", random=~G+H|SPEC_NUM, linear=FALSE, ytrend=-1, splinetrend=c(1, 1, 1), splinetype=c("Is", "Cs", "Is"), degree=c(3, 3, 3), knots=c(4, 4, 4), weights=NULL, subset=NULL, start=c(0.1, 0.1, -0.5, 0.01), maxiter=4)

---

**i.spline.x**

**i_spline basis**

**Description**

Create the i_th splines basis for I_splines basis or C_splines basis

**Usage**

`i.spline.x(xx, tt, i, k, delta = 1e-04, Cs = F)`

**Arguments**

- `xx` values of of covariate
- `tt` values of knot sequence of the covariate
- `i` I_th splines basis
- `k` degree of the spline
- `delta` the length of the spans to split the range of covariate
- `Cs` indicator of whether the creation of C_splines basis is needed

**Details**

Creation of the i_th splines basis for I_splines basis or C_splines basis

**Value**

`i.spline.x` returns a vector of i_th splines basis for covariate values xx in knots i and degree k
kaplan.meier.location

Author(s)
Yili Hong

References

See Also
m.spline.x

Examples

iNsplineNxHxx=runifHRPIL tt=cHPL PL PL PNSL PNVL PNXL 1L 1L 1 IL i=SL k=SL
delta = 1eMPTL cs = falseI

iNsplineNxHxx=runifHRPIL tt=cHPL PL PL PNSL PNVL PNXL 1L 1L 1 IL i=SL k=SL
delta = 1e-04, Cs = TRUE

kaplan.meier.location  Kaplan-Meier Location

Description
Calculate the location of Kaplan-Meier estimator.

Usage
kaplan.meier.location(fit)

Arguments
fit  survfit object.

Value
xloc, yloc  x, y locations.

Author(s)
Yili Hong

Examples
require(survival)
fit <- survfit(Surv(time, status) ~ x, data = aml)
kaplan.meier.location(fit)
Lifedata.MLE  
Parametric Fitting for Lifetime Data

Description

Fit parametric models (based on log-location-scale family of distributions) to right censored and/or left truncated data, with (dynamic) covariates.

Usage

Lifedata.MLE(formula, data, xt.dat = NULL, dist, method = "BFGS", subset, truncation, na.action, weights, ref_time = NULL, starts = NULL, ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>formula</td>
<td>A formula object, which has a Surv object on the left of ~ and covariates on the right.</td>
</tr>
<tr>
<td>data</td>
<td>A data frame used to evaluate variables in the formula, subset and weights arguments.</td>
</tr>
<tr>
<td>xt.dat</td>
<td>A data frame contains dynamic covariates. The first and second columns must be sample id and time. If specified, cumulative exposure model will be used to fit the data.</td>
</tr>
<tr>
<td>dist</td>
<td>Distribution used in the model. Can be &quot;weibull&quot;, &quot;lognormal&quot;, &quot;loglogistic&quot;, &quot;frechet&quot;.</td>
</tr>
<tr>
<td>method</td>
<td>Default is &quot;BFGS&quot;. This function calls optim to do optimization. Other options can be found in optim.</td>
</tr>
<tr>
<td>subset</td>
<td>This is an optional argument that specifies the subset of observations to be used in the fitting procedure.</td>
</tr>
<tr>
<td>truncation</td>
<td>An optional data frame contains truncation time and indicators.</td>
</tr>
<tr>
<td>na.action</td>
<td>Indicates what to be done when data contain missing values.</td>
</tr>
<tr>
<td>weights</td>
<td>An optional vector of weights for each observation.</td>
</tr>
<tr>
<td>ref_time</td>
<td>An optional vector of reference time (start time) of dynamic covariates. If it is NULL, ref_time is a vector of 0’s.</td>
</tr>
<tr>
<td>starts</td>
<td>Initial values for parameters.</td>
</tr>
<tr>
<td>...</td>
<td>Further arguments can be passed to optim.</td>
</tr>
</tbody>
</table>

Details

The default method for choosing start values can sometimes failed, the user may need to try several start values to make the function work.
Value

This function returns an object of class "Lifedata.MLE". An "Lifedata.MLE" object is a list of

- `call`  Called function.
- `formula`  Formula argument in the input.
- `coef`  Vector of coefficients.
- `vov`  Variance-covariance matrix of `coef`.
- `min`  Min loglikelihood evaluated at `coef`.
- `surv`  Survival probabilities.
- `dat`  Dataset used to fit the model.

Author(s)

Yili Hong

References


See Also

`Surv`, `optim`, `summary.Lifedata.MLE`. See methods(class="Lifedata.MLE") for all the functions related to "Lifedata.MLE" object.

Examples

```r
### right censored data
data(shock)
res1=lifedata.MLE(Surv(Distance, Censor)-1, data=shock, dist="weibull")
summary(res1)

res2=lifedata.MLE(Surv(Distance, Censor)-1, data=shock, dist="lognormal")
summary(res2)

### left truncated right censoring data
data(testdata)
test=lifedata.MLE(Surv(truncation.age, age, failure)-1, data=testdata, dist="weibull",
method="Nelder-Mead", subset=(group=="MC_Old"))
summary(test)

### dynamix covariates
data(Prod2.xt.dat)
```
**Description**

Calculate MLE for samples from lifetime distribution, e.g., Weibull distribution. The observations can be complete or censored.

**Usage**

```r
lifetime.mle(dat, minusloglik, starts, method = "BFGS", hessian = TRUE,...)
```

**Arguments**

- `dat`: First column contains event times, second column contains event indicators.
- `minusloglik`: Loglikelihood.
- `starts`: Initial values for parameters.
- `method`: Default is "BFGS". This function call `optim` to do optimisation. Other options can be found in `optim`.
- `hessian`: A logical value. If TRUE, hessian matrix will be returned. Default is TRUE.
- `...`: Further arguments can be passed to `optim`.

**Value**

A list of:

- `call`: Called function.
- `coef`: The best minimizer found.
- `vcov`: Variance-covariance matrix of `coef`.
- `min`: Loglikelihood evaluated at `coef`.
- `dat`: Dataset used.
- `minusloglik`: Loglikelihood function.

**Author(s)**

Yili Hong
m.spline.x

See Also

optim

Examples

# censored samples from Weibull distribution
dat=cbind(c(1.1,2,3,6,4,5.3,7,7), c(1,1,1,1,0,0,0))
res=lifetime.mle(dat, minusloglik=minusloglik.sev, starts=c(0,1))
res$coef  # return \{u, log(\sigma)\}

m.spline.x  \hspace{1cm} M_splines basis

Description

Creation of M_splines basis

Usage

m.spline.x(x, tt, i, k)

Arguments

x  a value of the covariate

tt  values of knot sequence of the covariate

i  the i_th M_spline function

k  degree of the spline

Value

m.spline.x returns a value of M_spline basis value for x in i_th spline degree of k

Author(s)

Yili Hong

References


See Also

i.spline.x

Examples

m.spline.x(x=0.3, tt=c(0, 0, 0, 0.3, 0.6, 0.8, 1, 1, 1), i=2, k=3)
MIC.splines.basis

**Splines basis functions**

**Description**

Creation splines basis for M_splines, I_splines and C_splines

**Usage**

```
MIC.splines.basis(x, df = NULL, knots = NULL, boundary.knots = NULL,
                 type = "Ms", degree = 3, delta = 0.01, eq.alloc = F)
```

**Arguments**

- `x`: values of covariate
- `df`: number of splines needed which is equal to knots+degree
- `knots`: number of knots needed which does not include the number of knots at the beginning
- `boundary.knots`: the values of boundary knots, which are usually the minimum and maximum of covariate
- `type`: types of splines basis needed, which can be Ms Is or Cs
- `degree`: degree of the splines function
- `delta`: the length of the spans to split the range of covariate
- `eq.alloc`: indicators of whether the knots are equally allocated

**Value**

```
list(mat=mat,x=x, df=df, knots=knots, boundary.knots=boundary.knots, type=type, degree=degree, delta=delta)
```

- `mat`: it is a matrix of splines basis
- `x`: the input of covariate
- `df`: the input of number of splines needed
- `knots`: the input of number of knots needed not including the boundary
- `boundary.knots`: the values of boundary knots
- `type`: type of splines function which can be MS IS or Cs
- `degree`: degree of the splines functions
- `delta`: the length of the spans to split the range of covariate

**Author(s)**

Yili Hong
The Standard Largest Extreme Value Distribution

Description

The cdf, pdf, quantile function, and random number generation for the standard largest extreme value distribution.

Usage

plev(z)
dlev(z)
qlev(p)
rlev(n)

Arguments

z Vector of values where the cdf or pdf to be evaluated.
p Vector of probabilities where the quantile function to be evaluated.
n Number of random samples. If input is a vector, then the number generated is the length of it.

Value

plev returns cdf, dlev returns pdf, qlev returns quantiles, and rlev returns random samples.

Author(s)

Yili Hong

See Also

psev, dsev, qsev, rsev.
plotdeglmx

**Plot function for the class of "deglmx".**

### Description

Plots of dynamic covariates and fitting of the model in the class of "deglmx".

### Usage

```r
plotdeglmx(x, type)
```

### Arguments

- **x**: The fitting results of class "deglmx".
- **type**: If type=1, plot the spline effect plot. If type=2, plot the fitting plots. If type is missing, plot all of them.

### Author(s)

Yili Hong

### See Also

deglmx

### Examples

```r
# Load example data
data(Coatingenv)
data(Coatingout)

# Fit model
fit = deglmx(DAMAGE_Y~UV+RH+TEMP, data=Coatingout, dyn.data=Coatingenv,
             id="SPEC_NUM", time="TIME", random=-TIME|SPEC_NUM, linear=TRUE, ytrend=-1,
             splinetrend=c(-1, -1, -1), splinetype=c("1s", "Cs", "1s"), degree=c(3, 3, 3),
             knots=c(4, 4, 4), weights=NULL, subset=NULL, start=c(0.017, 0.0013, -0.404, 0.0176),
             maxiter=10, method='BFGS')

# Plot results
plotdeglmx(x=fit)
plotdeglmx(x=fit, type=1)
plotdeglmx(x=fit, type=1)
```
**Prod2.fai.dat**  
*Dataset of failure information of Product 2.*

**Description**
A data frame with 1800 observations and 3 variables.

**Usage**
```r
data(Prod2.fai.dat)
```

**Format**
- `failure.time`: a numeric vector
- `delta`: a numeric vector
- `ce`: a numeric vector

**Source**

**Prod2.xt.dat**  
*Dataset of covariate information of Produce 2.*

**Description**
A data frame with 80552 observations and 3 variables.

**Usage**
```r
data(Prod2.xt.dat)
```

**Format**
- `id`: a numeric vector of sample ids.
- `time`: a numeric vector of time.
- `x1`: a numeric vector of dynamic covariate x1.

**Source**
The Standard Smallest Extreme Value Distribution

Description
The cdf, pdf, quantile function, and random number generation for the standard smallest extreme value distribution.

Usage
psev(z)
dsev(z)
qsev(p)
rsev(n)

Arguments
z Vector of values where the cdf or pdf to be evaluated.
p Vector of probabilities where the quantile function to be evaluated.
n Number of random samples. If input is a vector, then the number generated is the length of it.

Value
psev returns cdf, dsev returns pdf, qsev returns quantiles, and rsev returns random samples.

Author(s)
Yili Hong

See Also
plev, dlev, qlev, rlev.

Examples
psev(c(2, 3))
dsev(c(2, 3))
qsev(0.1)
rsev(10)
**shock**

**Shock Absorber Failure Data**

**Description**

A data frame with 38 observations and 3 variables.

**Usage**

```r
data(shock)
```

**Format**

- **distance**: a numeric vector
- **mode**: a factor with levels `Censored`, `Mode1`, `Mode2`
- **censor**: a numeric vector

**Source**


---

**summary.Lifedata.MLE**

**Summaries of "Lifedata.MLE" Object**

**Description**

These functions summaries a "Lifedata.MLE" object.

**Usage**

```r
## S3 method for class 'Lifedata.MLE'
summary(object, ...)
```

**Arguments**

- **object**: A "Lifedata.MLE" object.
- **...**: Additional arguments.
Value

summary.Lifedata.MLE returns an object of class "summary.Lifedata.MLE", which is a list of

call Component from obj.
coef Vector of coefficients.
vcov Variance-covariance matrix of coef
coeffmat Matrix contains mean, sd, 95% lower CI and 95% upper CI of coefficients.
min Component from obj.
surv Component from obj.
dat Component from obj.
ori.coef coef component in obj.
ori.vcov vcov component in obj.

See Also

Lifedata.MLE

testdata Testdata

Description

This dataset involves left truncation and right censoring.

Usage

data(testdata)

Format

A data frame with 710 observations on the following 9 variables.

age a numeric vector
failure a numeric vector
manufacture.year a numeric vector
manufacturer a factor with levels MA MB MC MD ME Other
cooling a factor with levels FIFE NIFE NINE Unknown
insulation a factor with levels d55 d65
truncation a numeric vector
truncation.age a numeric vector
group a factor with levels MA_New MB_Old MC_ME.Other_New MC_Old MD_Old ME_Old Other_Old
Source

Index

*Topic C_splines basis
  i.spline.x, 8
*Topic I_splines basis
  i.spline.x, 8
*Topic package
  SPREDA-package, 2

ce.dat.prep, 3
clme, 4
clme (deg1mx), 6
cls, 4
Coating.env, 5
Coating.out, 5
coef.deg1mx.MLE (deg1mx), 6
coef.Lifedata.MLE (Lifedata.MLE), 10
coef.initial.ftn (deg1mx), 6

data.pre.fun (deg1mx), 6
deg1mx, 4, 6, 16
dlev, 18
dlev (plev), 15
dsev, 15
dsev (psev), 18

getCov (deg1mx), 6
getnames (deg1mx), 6
getRanName (deg1mx), 6
i.spline.x, 8, 13, 15

kaplan.meier.location, 9
Lifedata.MLE, 10, 20
lifetime.mle, 12
logLik.Lifedata.MLE (Lifedata.MLE), 10
m.spline.x, 9, 13, 15
match.dat.fun (deg1mx), 6
MIC.splines.basis, 14

minusloglik.ce.xt.logis (Lifedata.MLE), 10
minusloglik.ce.xt.norm (Lifedata.MLE), 10
minusloglik.ce.xt.sev (Lifedata.MLE), 10
minusloglik.lev.wts (Lifedata.MLE), 10
minusloglik.logis.wts (Lifedata.MLE), 10
minusloglik.normal.wts (Lifedata.MLE), 10
minusloglik.sev (lifetime.mle), 12
minusloglik.sew.wts (Lifedata.MLE), 10
minus.log.lik.nlme (deg1mx), 6
minus.loglik.lme (deg1mx), 6
mle.obj.to.fit.obj (deg1mx), 6

optim, 7, 10–13
optim.ftn.2 (deg1mx), 6
optim.step.2 (deg1mx), 6
optim.step.2.2 (deg1mx), 6

plev, 15, 18
plot.MICsplines (deg1mx), 6
plotdeg1mx, 16
print.deg1mx (deg1mx), 6
print.Lifedata.MLE (Lifedata.MLE), 10
print.summary.Lifedata.MLE (summary.Lifedata.MLE), 19
Prod2.fai.dat, 17
Prod2.xt.dat, 17
psev, 15, 18
Px (deg1mx), 6

qlev, 18
qlev (plev), 15
qsev, 15
qsev (psev), 18
rlev, 18
INDEX

rlev (plev), 15
rsev, 15
rsev (psev), 18

shock, 19
SPREDA (SPREDA-package), 2
SPREDA-package, 2
sqrt.mat (deglmx), 6
summary.Lifedata.MLE, 11, 19
Surv, 3, 11
survfit, 9

testdata, 20

xmat.obj.to.xmat (deglmx), 6
xmat.to.cumsum (deglmx), 6