

Package ‘`intsurv`’

October 13, 2022

Title Integrative Survival Modeling

Version 0.2.2

Date 2021-01-07

Description Contains implementations of integrative survival analysis routines, including regular Cox cure rate model proposed by Kuk and Chen (1992) <[doi:10.1093/biomet/79.3.531](https://doi.org/10.1093/biomet/79.3.531)> via an EM algorithm proposed by Sy and Taylor (2000) <[doi:10.1111/j.0006-341X.2000.00227.x](https://doi.org/10.1111/j.0006-341X.2000.00227.x)>, regularized Cox cure rate model with elastic net penalty following Masud et al. (2018) <[doi:10.1177/0962280216677748](https://doi.org/10.1177/0962280216677748)>, and Zou and Hastie (2005) <[doi:10.1111/j.1467-9868.2005.00503.x](https://doi.org/10.1111/j.1467-9868.2005.00503.x)>, and weighted concordance index for cure models proposed by Asano and Hirakawa (2017) <[doi:10.1080/10543406.2017.1293082](https://doi.org/10.1080/10543406.2017.1293082)>.

Depends R (>= 3.2.3)

Imports Rcpp (>= 0.12.0), methods, stats

Suggests tinytest

LinkingTo Rcpp, RcppArmadillo

License GPL (>= 3)

LazyData true

Collate 'RcppExports.R' 'class.R' 'Surv.R' 'assessment.R' 'bootSe.R' 'coef.R' 'cox_cure.R' 'cox_cure_net.R' 'iCoxph.R' 'intsurv.R' 'misc.R' 'prep_model.R' 'print.R' 'show.R' 'simData4cure.R' 'simData4iCoxph.R' 'summary.R'

URL <https://wwenjie.org/intsurv>,
<https://github.com/wenjie2wang/intsurv>

BugReports <https://github.com/wenjie2wang/intsurv/issues>

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation yes

Author Wenjie Wang [aut, cre] (<<https://orcid.org/0000-0003-0363-3180>>),
 Kun Chen [ctb] (<<https://orcid.org/0000-0003-3579-5467>>),
 Jun Yan [ctb] (<<https://orcid.org/0000-0003-4401-7296>>)

Maintainer Wenjie Wang <wang@wwenjie.org>

Repository CRAN

Date/Publication 2021-01-08 09:10:09 UTC

R topics documented:

intsurv-package	2
BIC.cox_cure	3
BIC.cox_cure_net	4
bootSe	5
cIndex	6
coef,iCoxph-method	7
coef.cox_cure	8
coef.cox_cure_net	9
cox_cure	10
cox_cure_net	15
iCoxph	22
iCoxph-class	24
iCoxph.control	25
iCoxph.start	26
show-method	27
simData4cure	28
simData4iCoxph	29
summary,iCoxph-method	31
summary.cox_cure	32
summary.iCoxph-class	32
Survi	33
Survi-class	33
Index	34

intsurv-package *Integrative Survival Modeling*

Description

The package **intsurv** provides implementations of

Details

- integrative Cox model with uncertain event times (Wang et al., 2020)
- Cox cure rate model with uncertain event status (Wang et al., 2020)

It also contains other survival analysis routines, including regular Cox cure rate model, regularized Cox cure rate model with elastic net penalty, and weighted concordance index.

References

Wang, W., Aseltine, R. H., Chen, K., & Yan, J. (2020). Integrative Survival Analysis with Uncertain Event Times in Application to A Suicide Risk Study. *Annals of Applied Statistics*, 14(1), 51–73.

Wang, W., Luo, C., Aseltine, R. H., Wang, F., Yan, J., & Chen, K. (2020). Suicide Risk Modeling with Uncertain Diagnostic Records. *arXiv preprint arXiv:2009.02597*.

BIC.cox_cure

Bayesian Information Criterion (BIC)

Description

Compute Bayesian information criterion (BIC) or Schwarz's Bayesian criterion (SBC) for possibly one or several objects.

Usage

```
## S3 method for class 'cox_cure'
BIC(object, ..., method = c("obs", "effective"))

## S3 method for class 'cox_cure_uncer'
BIC(object, ..., method = c("obs", "certain-event"))
```

Arguments

object	An object for a fitted model.
...	Other objects of the same class.
method	A character string specifying the method for computing the BIC values. Notice that this argument is placed after ... and thus must be specified as a named argument. The available options for <code>cox_cure</code> objects are "obs" for regular BIC based on the number of observations, and "effective" for using BIC based on the number of effective sample size for censored data (number of uncensored events) proposed by Volinsky and Raftery (2000). The available options for <code>cox_cure_uncer</code> objects are "obs" for regular BIC based on the number of observations, and "certain-event" for a variant of BIC based on the number of certain uncensored events. For objects of either class, the former method is used by default.

References

Volinsky, C. T., & Raftery, A. E. (2000). Bayesian information criterion for censored survival models. *Biometrics*, 56(1), 256–262.

Examples

```
## See examples of function 'cox_cure'.
```

BIC.cox_cure_net *Bayesian Information Criterion (BIC)*

Description

Compute Bayesian information criterion (BIC) or Schwarz's Bayesian criterion (SBC) from a fitted solution path.

Usage

```
## S3 method for class 'cox_cure_net'  
BIC(object, ..., method = c("obs", "effective"))  
  
## S3 method for class 'cox_cure_net_uncer'  
BIC(object, ..., method = c("obs", "certain-event"))
```

Arguments

object	An object for a fitted solution path.
...	Other arguments for future usage. A warning message will be thrown for any invalid argument.
method	A character string specifying the method for computing the BIC values. Notice that this argument is placed after ... and thus must be specified as a named argument. The available options for <code>cox_cure</code> objects are "obs" for regular BIC based on the number of observations, and "effective" for using BIC based on the number of effective sample size for censored data (number of uncensored events) proposed by Volinsky and Raftery (2000). The available options for <code>cox_cure_uncer</code> objects are "obs" for regular BIC based on the number of observations, and "certain-event" for a variant of BIC based on the number of certain uncensored events. For objects of either class, the former method is used by default.

References

Volinsky, C. T., & Raftery, A. E. (2000). Bayesian information criterion for censored survival models. *Biometrics*, 56(1), 256–262.

Examples

```
## See examples of function 'cox_cure_net'.
```

bootSe

*Standard Error Estimates through Bootstrap***Description**

For `iCoxph-class` object, add (or update) standard error (SE) estimates through bootstrap methods, or compute the coefficient estimates from the given number of bootstrap samples.

Usage

```
bootSe(
  object,
  B = 50,
  se = c("inter-quartile", "mad", "sd"),
  return_beta = FALSE,
  ...
)
```

Arguments

<code>object</code>	<code>iCoxph-class</code> object.
<code>B</code>	A positive integer specifying number of bootstrap samples used for SE estimates. A large number, such as 200, is often needed for a more reliable estimation in practice. If <code>B = 1</code> is specified, the function will return the covariate coefficient estimates instead of a <code>iCoxph-class</code> object.
<code>se</code>	A character value specifying the way computing SE from bootstrap samples. The default method is based on median absolute deviation and the second method is based on inter-quartile, both of which are based on normality of the bootstrap estimates and provides robust estimates for SE. The third method estimates SE by the standard deviation of the bootstrap estimates.
<code>return_beta</code>	A logical value. If <code>TRUE</code> , the function returns the covariate coefficient estimates from the given number of bootstrap samples, which allows users to split the most computationally intensive step into small pieces that can be computed in a parallel manner. The default value is <code>FALSE</code> .
<code>...</code>	Other arguments for future usage. A warning will be thrown if any invalid argument is specified.

Details

Three different methods are available for computing SE from bootstrap samples through argument `se`. Given the fact that the bootstrap method is computationally intensive, the function returns the coefficient estimates in a matrix from the given number of bootstrap samples when `return_beta = TRUE` is specified, which can be used in parallel computing or high performance computing (HPC) cluster. The SE estimates can be further computed based on estimates from bootstrap samples by users on their own. The `return_beta = TRUE` is implied, when `B = 1` is specified.

Value

`iCoxph-class` object or a numeric matrix that contains the covariate coefficient estimates from the given number of bootstrap samples in rows.

See Also

`iCoxph` for fitting integrative Cox model.

Examples

```
## See examples of function 'iCoxph'.
```

cIndex

Concordance Index

Description

Compute concordance index (C-index or C-statistic) that allows weights for right-censored survival data. For example, Asano and Hirakawa (2017) proposed cure status weighting for cure models, which reduces to Harrell's C-index if weights are all ones.

Usage

```
cIndex(time, event = NULL, risk_score, weight = NULL)
```

Arguments

<code>time</code>	A numeric vector for observed times
<code>event</code>	A numeric vector for event indicators. If it is NULL (by default) or NA, event will be treated all as ones and the function will compute concordance index for uncensored survival data.
<code>risk_score</code>	A numeric vector representing the risk scores of events.
<code>weight</code>	A optional numeric vector for weights. If it is NULL (by default) or NA, equal weights will be used.

Details

Let r_i , t_i , and δ_i denote the risk score, observed time, and event indicator of i -th subject. The pair of (t_i, δ_i) and (t_j, δ_j) , where $i < j$, are defined to be comparable if $t_i < t_j, \delta_i = 1$ or $t_i = t_j, \delta_i = 1, \delta_j = 0$. In the context of survival analysis, the risk scores of a comparable pair are said to be concordant with the survival outcomes if $r_i > r_j$. The C-index is defined as the proportion of the concordant pairs among the comparable pairs. For comparable pair satisfying $t_i < t_j, \delta_i = 1$, we count 0.5 in the numerator of the concordance index for tied risk scores ($r_i = r_j$).

Value

A named numeric vector that consists of

- `index`: the concordance index.
- `concordant`: the number of concordant pairs.
- `comparable`: the number of comparable pairs.
- `tied_tisk`: the number of comparable pairs having tied risks.

References

Asano, J., & Hirakawa, A. (2017). Assessing the prediction accuracy of a cure model for censored survival data with long-term survivors: Application to breast cancer data. *Journal of Biopharmaceutical Statistics*, 27(6), 918–932.

Harrell, F. E., Lee, K. L., & Mark, D. B. (1996). Multivariable prognostic models: Issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in medicine*, 15(4), 361–387.

Examples

```
## See examples of function 'cox_cure'.
```

coef, iCoxph-method	<i>Estimated Covariates Coefficients</i>
---------------------	--

Description

coef, iCoxph-method is an S4 class method that extracts covariate coefficient estimates from [iCoxph-class](#) object from function [iCoxph](#).

Usage

```
## S4 method for signature 'iCoxph'  
coef(object, ...)
```

Arguments

object	iCoxph-class object.
...	Other arguments for future usage.

Value

A named numeric vector.

See Also

[iCoxph](#) for fitting integrative Cox model; [summary, iCoxph-method](#) for summary of a fitted model.

Examples

```
## See examples of function iCoxph.
```

coef.cox_cure	<i>Estimated Covariate Coefficients</i>
---------------	---

Description

Extract the covariate coefficient estimates from a fitted Cox cure rate model with possible uncertain event status.

Usage

```
## S3 method for class 'cox_cure'
coef(object, part = c("both", "survival", "cure"), ...)

## S3 method for class 'cox_cure_uncer'
coef(object, part = c("both", "survival", "cure"), ...)
```

Arguments

object	Object representing a fitted model.
part	A character string specifying the coefficient estimates from a particular model part. The available options are "both" for the coefficient estimates from both model parts, "survival" for the coefficient estimates from the survival model part, and "cure" for the coefficient estimates from the cure rate model part. The default option is "all".
...	Other arguments for future usage.

Value

If part = "both", this function returns a list that consists of the following named elements

- surv: the coefficient estimates of survival model part.
- cure: the coefficient estimates of cure rate model part.

Otherwise, a named numeric vector representing the coefficient estimates of the specified model part will be returned.

coef.cox_cure_net *Estimated Covariate Coefficients*

Description

Extract the covariate coefficient estimates from a solution path of regularized Cox cure rate model.

Usage

```
## S3 method for class 'cox_cure_net'
coef(object, naive_en = FALSE, selection = c("bic1", "bic2", "all"), ...)

## S3 method for class 'cox_cure_net_uncer'
coef(object, naive_en = FALSE, selection = c("bic1", "bic2", "all"), ...)
```

Arguments

object	Object representing a fitted solution path.
naive_en	A logical value specifying whether to return naive elastic net estimates. If FALSE by default, the elastic net estimates will be returned instead of the naive elastic net estimates.
selection	A character string for specifying the criterion for selection of coefficient estimates. The available options are "bic1" for selecting coefficient estimates by regular BIC criterion based on the number of observations, "bic2" for a variant BIC criterion based on the effective sample size, and "all" for returning the whole solution path. See BIC.cox_cure_net for details of selection by BIC.
...	Other arguments for future usage.

Value

A list that consists of the following named elements:

- surv: the selected coefficient estimates of survival model part.
- cure: the selected coefficient estimates of cure rate model part.

Examples

```
## see examples of function `cox_cure_net`
```

`cox_cure`*Cox Cure Rate Model*

Description

For right-censored data, fit a regular Cox cure rate model (Kuk and Chen, 1992; Sy and Taylor, 2000) via an EM algorithm. For right-censored data with uncertain event status, fit the Cox cure model proposed by Wang et al. (2020).

Usage

```
cox_cure(  
  surv_formula,  
  cure_formula,  
  time,  
  event,  
  data,  
  subset,  
  contrasts = NULL,  
  bootstrap = 0,  
  firth = FALSE,  
  surv_start = NULL,  
  cure_start = NULL,  
  surv_offset = NULL,  
  cure_offset = NULL,  
  em_max_iter = 200,  
  em_rel_tol = 1e-05,  
  surv_max_iter = 30,  
  surv_rel_tol = 1e-05,  
  cure_max_iter = 30,  
  cure_rel_tol = 1e-05,  
  tail_completion = c("zero", "exp", "zero-tau"),  
  tail_tau = NULL,  
  pmin = 1e-05,  
  early_stop = TRUE,  
  verbose = FALSE,  
  ...  
)  
  
cox_cure.fit(  
  surv_x,  
  cure_x,  
  time,  
  event,  
  cure_intercept = TRUE,  
  bootstrap = 0,  
  firth = FALSE,
```

```

    surv_start = NULL,
    cure_start = NULL,
    surv_offset = NULL,
    cure_offset = NULL,
    surv_standardize = TRUE,
    cure_standardize = TRUE,
    em_max_iter = 200,
    em_rel_tol = 1e-05,
    surv_max_iter = 30,
    surv_rel_tol = 1e-05,
    cure_max_iter = 30,
    cure_rel_tol = 1e-05,
    tail_completion = c("zero", "exp", "zero-tau"),
    tail_tau = NULL,
    pmin = 1e-05,
    early_stop = TRUE,
    verbose = FALSE,
    ...
)

```

Arguments

surv_formula	A formula object starting with ~ for the model formula in survival model part. For Cox model, no intercept term is included even if an intercept is specified or implied in the model formula. A model formula with an intercept term only is not allowed.
cure_formula	A formula object starting with ~ for the model formula in incidence model part. For logistic model, an intercept term is included by default and can be excluded by adding + 0 or - 1 to the model formula.
time	A numeric vector for the observed survival times.
event	A numeric vector for the event indicators. NA's are allowed and represent uncertain event status.
data	An optional data frame, list, or environment that contains the covariates and response variables (time and event), included in the model. If they are not found in data, the variables are taken from the environment of the specified formula, usually the environment from which this function is called.
subset	An optional logical vector specifying a subset of observations to be used in the fitting process.
contrasts	An optional list, whose entries are values (numeric matrices or character strings naming functions) to be used as replacement values for the contrasts replacement function and whose names are the names of columns of data containing factors. See contrasts.arg of model.matrix.default for details.
bootstrap	An integer representing the number of bootstrap samples for estimating standard errors of the coefficient estimates. The bootstrap procedure will not run with bootstrap = 0 by default. If bootstrap > 0, the specified number of bootstrap samples will be used in estimating standard errors.

<code>firth</code>	A logical value indicating whether to use Firth's bias-reduction method (Firth, 1993) in the logistic model component. The default value is FALSE for fitting a regular logistic model. Notice that this argument is experimental and only available for regular Cox cure rate model currently.
<code>surv_start, cure_start</code>	An optional numeric vector representing the starting values for the survival model component or the incidence model component. If <code>surv_start = NULL</code> is specified, the starting values will be obtained from fitting a regular Cox to events only. Similarly, if <code>cure_start = NULL</code> is specified, the starting values will be obtained from fitting a regular logistic model to the non-missing event indicators.
<code>surv_offset, cure_offset</code>	An optional numeric vector representing the offset term in the survival model component or the incidence model component. The function will internally try to find values of the specified variable in the data first. Alternatively, one or more offset terms can be specified in the formula (by <code>stats::offset()</code>). If more than one offset terms are specified, their sum will be used.
<code>em_max_iter</code>	A positive integer specifying the maximum iteration number of the EM algorithm. The default value is 200.
<code>em_rel_tol</code>	A positive number specifying the tolerance that determines the convergence of the EM algorithm in terms of the convergence of the covariate coefficient estimates. The tolerance is compared with the relative change between estimates from two consecutive iterations, which is measured by ratio of the L1-norm of their difference to the sum of their L1-norm. The default value is 1e-5.
<code>surv_max_iter, cure_max_iter</code>	A positive integer specifying the maximum iteration number of the M-step routine related to the survival model component or the incidence model component. The default value is 200.
<code>surv_rel_tol, cure_rel_tol</code>	A positive number specifying the tolerance that determines the convergence of the M-step related to the survival model component or the incidence model component in terms of the convergence of the covariate coefficient estimates. The tolerance is compared with the relative change between estimates from two consecutive iterations, which is measured by ratio of the L1-norm of their difference to the sum of their L1-norm. The default value is 1e-5.
<code>tail_completion</code>	A character string specifying the tail completion method for conditional survival function. The available methods are "zero" for zero-tail completion after the largest event times (Sy and Taylor, 2000), "exp" for exponential-tail completion (Peng, 2003), and "zero-tau" for zero-tail completion after a specified <code>tail_tau</code> . The default method is the zero-tail completion proposed by Sy and Taylor (2000).
<code>tail_tau</code>	A numeric number specifying the time of zero-tail completion. It will be used only if <code>tail_completion = "zero-tau"</code> . A reasonable choice must be a time point between the largest event time and the largest survival time.
<code>pmin</code>	A numeric number specifying the minimum value of probabilities for sake of numerical stability. The default value is 1e-5.

early_stop	A logical value specifying whether to stop the iteration once the negative log-likelihood unexpectedly increases, which may suggest convergence on likelihood, or indicate numerical issues or implementation bugs. The default value is TRUE.
verbose	A logical value. If TRUE, a verbose information will be given along iterations for tracing the convergence. The default value is FALSE.
...	Other arguments for future usage. A warning will be thrown if any invalid argument is specified.
surv_x	A numeric matrix for the design matrix of the survival model component.
cure_x	A numeric matrix for the design matrix of the cure rate model component. The design matrix should exclude an intercept term unless we want to fit a model only including the intercept term. In that case, we need further set <code>cure_intercept = FALSE</code> to not standardize the intercept term.
cure_intercept	A logical value specifying whether to add an intercept term to the cure rate model component. If TRUE by default, an intercept term is included.
surv_standardize	A logical value specifying whether to standardize the covariates for the survival model part. If FALSE, the covariates will be standardized internally to have mean zero and standard deviation one.
cure_standardize	A logical value specifying whether to standardize the covariates for the cure rate model part. If TRUE (by default), the covariates will be standardized internally to have mean zero and standard deviation one.

Value

cox_cure object for regular Cox cure rate model or cox_cure_uncer object for Cox cure rate model with uncertain events.

References

- Firth, D. (1993). Bias reduction of maximum likelihood estimates. *Biometrika*, 80(1), 27–38.
- Kuk, A. Y. C., & Chen, C. (1992). A mixture model combining logistic regression with proportional hazards regression. *Biometrika*, 79(3), 531–541.
- Peng, Y. (2003). Estimating baseline distribution in proportional hazards cure models. *Computational Statistics & Data Analysis*, 42(1-2), 187–201.
- Sy, J. P., & Taylor, J. M. (2000). Estimation in a Cox proportional hazards cure model. *Biometrics*, 56(1), 227–236.
- Wang, W., Luo, C., Aseltine, R. H., Wang, F., Yan, J., & Chen, K. (2020). Suicide Risk Modeling with Uncertain Diagnostic Records. *arXiv preprint arXiv:2009.02597*.

See Also

[cox_cure_net](#) for regularized Cox cure rate model with elastic-net penalty.

Examples

```

library(intsurv)

### regular Cox cure rate model =====
## 1. simulate right-censored data with a cure fraction
set.seed(123)
n_obs <- 2e2
p <- 5
x_mat <- matrix(rnorm(n_obs * p), nrow = n_obs, ncol = p)
colnames(x_mat) <- paste0("x", seq_len(p))
cure_beta <- rep(0.5, p)
b0 <- -1
expit <- binomial()$linkinv
ncure_prob <- expit(as.numeric(b0 + x_mat %*% cure_beta))
is_cure <- 1 - rbinom(n_obs, size = 1, prob = ncure_prob)
surv_beta <- rep(0.5, p)
risk_score <- as.numeric(x_mat %*% surv_beta)
event_time <- rexp(n_obs, exp(as.numeric(x_mat %*% surv_beta)))
censor_time <- 10
event <- ifelse(event_time < censor_time & ! is_cure, 1, 0)
obs_time <- ifelse(event > 0, event_time, censor_time)

## model-fitting from given design matrices
fit1 <- cox_cure.fit(x_mat, x_mat, obs_time, event, bootstrap = 30)
summary(fit1)

## coefficient estimates from both model parts
coef(fit1)

## or a particular part
coef(fit1, "surv")
coef(fit1, "cure")

## weighted concordance index (C-index)
fit1$model$c_index
## which also can be computed as follows
cIndex(time = obs_time, event = event,
        risk_score = fit1$fitted$surv_xBeta,
        weight = ifelse(event > 0, 1, fit1$fitted$susceptible_prob))

## 2. create a toy dataset
toy_dat <- data.frame(time = obs_time, status = event)
toy_dat$group <- cut(abs(x_mat[, 1L]), breaks = c(0, 0.5, 1, 3, Inf),
                    labels = LETTERS[1:4])
toy_dat <- cbind(toy_dat, as.data.frame(x_mat[, -1L, drop = FALSE]))

## model-fitting from given model formula
fit2 <- cox_cure(~ x3 + x4 + group, ~ group + x3 + offset(x2),
               time = time, event = status, surv_offset = x2,
               data = toy_dat, subset = group != "D", bootstrap = 30)
summary(fit2)

```

```

## get BIC's
BIC(fit1)
BIC(fit2)
BIC(fit1, fit2)

### Cox cure rate model with uncertain event status =====
## simulate sample data
sim_dat <- simData4cure(nSubject = 200, max_censor = 10, lambda_censor = 0.1,
                      survMat = x_mat, cureMat = x_mat, b0 = 1)
table(sim_dat$case)
table(sim_dat$obs_event, useNA = "ifany")

## use formula
fit3 <- cox_cure(~ x1 + x2 + x3, ~ z1 + z2 + z3,
               time = obs_time, event = obs_event, data = sim_dat)
summary(fit3)

## use design matrix
fit4 <- cox_cure.fit(x_mat, x_mat, time = sim_dat$obs_time,
                   event = sim_dat$obs_event)
summary(fit4)

## get BIC's
BIC(fit3, fit4)

```

cox_cure_net

Regularized Cox Cure Rate Model

Description

For right-censored data, fit a regularized Cox cure rate model through elastic-net penalty following Masud et al. (2018), and Zou and Hastie (2005). For right-censored data with uncertain event status, fit the regularized Cox cure model proposed by Wang et al. (2020). Without regularization, the model reduces to the regular Cox cure rate model (Kuk and Chen, 1992; Sy and Taylor, 2000)

Usage

```

cox_cure_net(
  surv_formula,
  cure_formula,
  time,
  event,
  data,
  subset,
  contrasts = NULL,
  surv_lambda = NULL,
  surv_alpha = 1,
  surv_nlambda = 10,

```

```
surv_lambda_min_ratio = 0.1,  
surv_l1_penalty_factor = NULL,  
cure_lambda = NULL,  
cure_alpha = 1,  
cure_nlambda = 10,  
cure_lambda_min_ratio = 0.1,  
cure_l1_penalty_factor = NULL,  
cv_nfolds = 0,  
surv_start = NULL,  
cure_start = NULL,  
surv_offset = NULL,  
cure_offset = NULL,  
surv_standardize = TRUE,  
cure_standardize = TRUE,  
em_max_iter = 200,  
em_rel_tol = 1e-05,  
surv_max_iter = 10,  
surv_rel_tol = 1e-05,  
cure_max_iter = 10,  
cure_rel_tol = 1e-05,  
tail_completion = c("zero", "exp", "zero-tau"),  
tail_tau = NULL,  
pmin = 1e-05,  
early_stop = TRUE,  
verbose = FALSE,  
...  
)
```

```
cox_cure_net.fit(  
  surv_x,  
  cure_x,  
  time,  
  event,  
  cure_intercept = TRUE,  
  surv_lambda = NULL,  
  surv_alpha = 1,  
  surv_nlambda = 10,  
  surv_lambda_min_ratio = 0.1,  
  surv_l1_penalty_factor = NULL,  
  cure_lambda = NULL,  
  cure_alpha = 1,  
  cure_nlambda = 10,  
  cure_lambda_min_ratio = 0.1,  
  cure_l1_penalty_factor = NULL,  
  cv_nfolds = 0,  
  surv_start = NULL,  
  cure_start = NULL,  
  surv_offset = NULL,
```

```

    cure_offset = NULL,
    surv_standardize = TRUE,
    cure_standardize = TRUE,
    em_max_iter = 200,
    em_rel_tol = 1e-05,
    surv_max_iter = 10,
    surv_rel_tol = 1e-05,
    cure_max_iter = 10,
    cure_rel_tol = 1e-05,
    tail_completion = c("zero", "exp", "zero-tau"),
    tail_tau = NULL,
    pmin = 1e-05,
    early_stop = TRUE,
    verbose = FALSE,
    ...
)

```

Arguments

surv_formula	A formula object starting with ~ for the model formula in survival model part. For Cox model, no intercept term is included even if an intercept is specified or implied in the model formula. A model formula with an intercept term only is not allowed.
cure_formula	A formula object starting with ~ for the model formula in incidence model part. For logistic model, an intercept term is included by default and can be excluded by adding + 0 or - 1 to the model formula.
time	A numeric vector for the observed survival times.
event	A numeric vector for the event indicators. NA's are allowed and represent uncertain event status.
data	An optional data frame, list, or environment that contains the covariates and response variables (time and event), included in the model. If they are not found in data, the variables are taken from the environment of the specified formula, usually the environment from which this function is called.
subset	An optional logical vector specifying a subset of observations to be used in the fitting process.
contrasts	An optional list, whose entries are values (numeric matrices or character strings naming functions) to be used as replacement values for the contrasts replacement function and whose names are the names of columns of data containing factors. See contrasts.arg of model.matrix.default for details.
surv_lambda, cure_lambda	A numeric vector consists of nonnegative values representing the tuning parameter sequence for the survival model part or the incidence model part.
surv_alpha, cure_alpha	A number between 0 and 1 for tuning the elastic net penalty for the survival model part or the incidence model part. If it is one, the elastic penalty will reduce to the well-known lasso penalty. If it is zero, the ridge penalty will be used.

surv_nlambda, cure_nlambda	A positive number specifying the number of surv_lambda or cure_lambda if surv_lambda or cure_lambda is not specified, respectively. The default value is 10.
surv_lambda_min_ratio, cure_lambda_min_ratio	The ratio of the minimum surv_lambda (or cure_lambda) to the large enough surv_lambda (or cure_lambda) that produces all-zero estimates on log scale. The default value is 1e-1.
surv_l1_penalty_factor, cure_l1_penalty_factor	A numeric vector that consists of nonnegative penalty factors (or weights) on L1-norm for the coefficient estimate vector in the survival model part or the incidence model part. The penalty is applied to the coefficient estimate divided by the specified weights. The specified weights are re-scaled internally so that their summation equals the length of coefficients. If NULL is specified, the weights are all set to be one.
cv_nfolds	An non-negative integer specifying number of folds in cross-validation (CV). The default value is 0 and the CV procedure is not enabled.
surv_start	An optional numeric vector representing the starting values for the survival model component or the incidence model component. If surv_start = NULL is specified, the starting values will be obtained from fitting a regular Cox to events only. Similarly, if cure_start = NULL is specified, the starting values will be obtained from fitting a regular logistic model to the non-missing event indicators.
cure_start	An optional numeric vector representing the starting values for the survival model component or the incidence model component. If surv_start = NULL is specified, the starting values will be obtained from fitting a regular Cox to events only. Similarly, if cure_start = NULL is specified, the starting values will be obtained from fitting a regular logistic model to the non-missing event indicators.
surv_offset	An optional numeric vector representing the offset term in the survival model component or the incidence model component. The function will internally try to find values of the specified variable in the data first. Alternatively, one or more offset terms can be specified in the formula (by stats::offset()). If more than one offset terms are specified, their sum will be used.
cure_offset	An optional numeric vector representing the offset term in the survival model component or the incidence model component. The function will internally try to find values of the specified variable in the data first. Alternatively, one or more offset terms can be specified in the formula (by stats::offset()). If more than one offset terms are specified, their sum will be used.
surv_standardize, cure_standardize	A logical value specifying whether to standardize the covariates for the survival model part or the incidence model part. If FALSE, the covariates will be standardized internally to have mean zero and standard deviation one.
em_max_iter	A positive integer specifying the maximum iteration number of the EM algorithm. The default value is 200.

em_rel_tol	A positive number specifying the tolerance that determines the convergence of the EM algorithm in terms of the convergence of the covariate coefficient estimates. The tolerance is compared with the relative change between estimates from two consecutive iterations, which is measured by ratio of the L1-norm of their difference to the sum of their L1-norm. The default value is 1e-5.
surv_max_iter, cure_max_iter	A positive integer specifying the maximum iteration number of the M-step routine related to the survival model component or the incidence model component. The default value is 10 to encourage faster convergence.
surv_rel_tol	A positive number specifying the tolerance that determines the convergence of the M-step related to the survival model component or the incidence model component in terms of the convergence of the covariate coefficient estimates. The tolerance is compared with the relative change between estimates from two consecutive iterations, which is measured by ratio of the L1-norm of their difference to the sum of their L1-norm. The default value is 1e-5.
cure_rel_tol	A positive number specifying the tolerance that determines the convergence of the M-step related to the survival model component or the incidence model component in terms of the convergence of the covariate coefficient estimates. The tolerance is compared with the relative change between estimates from two consecutive iterations, which is measured by ratio of the L1-norm of their difference to the sum of their L1-norm. The default value is 1e-5.
tail_completion	A character string specifying the tail completion method for conditional survival function. The available methods are "zero" for zero-tail completion after the largest event times (Sy and Taylor, 2000), "exp" for exponential-tail completion (Peng, 2003), and "zero-tau" for zero-tail completion after a specified tail_tau. The default method is the zero-tail completion proposed by Sy and Taylor (2000).
tail_tau	A numeric number specifying the time of zero-tail completion. It will be used only if tail_completion = "zero-tau". A reasonable choice must be a time point between the largest event time and the largest survival time.
pmin	A numeric number specifying the minimum value of probabilities for sake of numerical stability. The default value is 1e-5.
early_stop	A logical value specifying whether to stop the iteration once the negative log-likelihood unexpectedly increases, which may suggest convergence on likelihood, or indicate numerical issues or implementation bugs. The default value is TRUE.
verbose	A logical value. If TRUE, a verbose information will be given along iterations for tracing the convergence. The default value is FALSE.
...	Other arguments for future usage. A warning will be thrown if any invalid argument is specified.
surv_x	A numeric matrix for the design matrix of the survival model component.
cure_x	A numeric matrix for the design matrix of the cure rate model component. The design matrix should exclude an intercept term unless we want to fit a model only including the intercept term. In that case, we need further set cure_intercept = FALSE to not standardize the intercept term.

cure_intercept A logical value specifying whether to add an intercept term to the cure rate model component. If TRUE by default, an intercept term is included.

Details

The model estimation procedure follows expectation maximization (EM) algorithm. Variable selection procedure through regularization by elastic net penalty is developed based on cyclic coordinate descent and majorization-minimization (MM) algorithm.

Value

cox_cure_net object for regular Cox cure rate model or cox_cure_net_uncer object for Cox cure rate model with uncertain events.

References

- Kuk, A. Y. C., & Chen, C. (1992). A mixture model combining logistic regression with proportional hazards regression. *Biometrika*, 79(3), 531–541.
- Masud, A., Tu, W., & Yu, Z. (2018). Variable selection for mixture and promotion time cure rate models. *Statistical methods in medical research*, 27(7), 2185–2199.
- Peng, Y. (2003). Estimating baseline distribution in proportional hazards cure models. *Computational Statistics & Data Analysis*, 42(1-2), 187–201.
- Sy, J. P., & Taylor, J. M. (2000). Estimation in a Cox proportional hazards cure model. *Biometrics*, 56(1), 227–236.
- Wang, W., Luo, C., Aseltine, R. H., Wang, F., Yan, J., & Chen, K. (2020). Suicide Risk Modeling with Uncertain Diagnostic Records. *arXiv preprint arXiv:2009.02597*.
- Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2), 301–320.

See Also

[cox_cure](#) for regular Cox cure rate model.

Examples

```
library(intsurv)

### regularized Cox cure rate model =====
## simulate a toy right-censored data with a cure fraction
set.seed(123)
n_obs <- 100
p <- 10
x_mat <- matrix(rnorm(n_obs * p), nrow = n_obs, ncol = p)
colnames(x_mat) <- paste0("x", seq_len(p))
surv_beta <- c(rep(0, p - 5), rep(1, 5))
cure_beta <- c(rep(1, 2), rep(0, p - 2))
dat <- simData4cure(nSubject = n_obs, lambda_censor = 0.01,
                  max_censor = 10, survMat = x_mat,
                  survCoef = surv_beta, cureCoef = cure_beta,
```

```

      b0 = 0.5, p1 = 1, p2 = 1, p3 = 1)

## model-fitting from given design matrices
fit1 <- cox_cure_net.fit(x_mat, x_mat, dat$obs_time, dat$obs_event,
                       surv_nlambda = 10, cure_nlambda = 10,
                       surv_alpha = 0.8, cure_alpha = 0.8)

## model-fitting from given model formula
fm <- paste(paste0("x", seq_len(p)), collapse = " + ")
surv_fm <- as.formula(sprintf("~ %s", fm))
cure_fm <- surv_fm
fit2 <- cox_cure_net(surv_fm, cure_fm, data = dat,
                    time = obs_time, event = obs_event,
                    surv_alpha = 0.5, cure_alpha = 0.5)

## summary of BIC's
BIC(fit1)
BIC(fit2)

## list of coefficient estimates based on BIC
coef(fit1)
coef(fit2)

### regularized Cox cure model with uncertain event status =====
## simulate a toy data
set.seed(123)
n_obs <- 100
p <- 10
x_mat <- matrix(rnorm(n_obs * p), nrow = n_obs, ncol = p)
colnames(x_mat) <- paste0("x", seq_len(p))
surv_beta <- c(rep(0, p - 5), rep(1, 5))
cure_beta <- c(rep(1, 2), rep(0, p - 2))
dat <- simData4cure(nSubject = n_obs, lambda_censor = 0.01,
                  max_censor = 10, survMat = x_mat,
                  survCoef = surv_beta, cureCoef = cure_beta,
                  b0 = 0.5, p1 = 0.95, p2 = 0.95, p3 = 0.95)

## model-fitting from given design matrices
fit1 <- cox_cure_net.fit(x_mat, x_mat, dat$obs_time, dat$obs_event,
                       surv_nlambda = 5, cure_nlambda = 5,
                       surv_alpha = 0.8, cure_alpha = 0.8)

## model-fitting from given model formula
fm <- paste(paste0("x", seq_len(p)), collapse = " + ")
surv_fm <- as.formula(sprintf("~ %s", fm))
cure_fm <- surv_fm
fit2 <- cox_cure_net(surv_fm, cure_fm, data = dat,
                    time = obs_time, event = obs_event,
                    surv_nlambda = 5, cure_nlambda = 5,
                    surv_alpha = 0.5, cure_alpha = 0.5)

## summary of BIC's

```

```

BIC(fit1)
BIC(fit2)

## list of coefficient estimates based on BIC
coef(fit1)
coef(fit2)

```

iCoxph

Integrative Cox Model for Uncertain Event Times

Description

Fit an integrative Cox model proposed by Wang et al. (2020) for right-censored survival data with uncertain event times due to imperfect data integration.

Usage

```

iCoxph(
  formula,
  data,
  subset,
  na.action,
  contrasts = NULL,
  start = iCoxph.start(),
  control = iCoxph.control(),
  ...
)

```

Arguments

formula	Survi object specifying the covariates and response variable in the model, such as <code>Survi(ID, time, event) ~ x1 + x2</code> .
data	An optional data frame, list, or environment that contains the covariates and response variables included in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , usually the environment from which this function is called.
subset	An optional logical vector specifying a subset of observations to be used in the fitting process.
na.action	An optional function that indicates what should the procedure do if the data contains NAs. The default is set by the <code>na.action</code> setting of <code>options</code> . The "factory-fresh" default is <code>na.omit</code> . Other possible values include <code>na.fail</code> , <code>na.exclude</code> , and <code>na.pass</code> . <code>help(na.fail)</code> for details.
contrasts	An optional list, whose entries are values (numeric matrices or character strings naming functions) to be used as replacement values for the contrasts replacement function and whose names are the names of columns of data containing factors. See <code>contrasts.arg</code> of <code>model.matrix.default</code> for details.

start	A list returned by function iCoxph.start specifying starting values of the parameters to be estimated in the model. Please refer to the arguments of iCoxph.start for the available parameters.
control	A list returned by function iCoxph.control specifying control parameters for the model estimation procedure. Please refer to the arguments of iCoxph.control for the available parameters.
...	Other arguments for future usage. A warning will be thrown if any invalid argument is specified.

Value

An [iCoxph-class](#) object, whose slots include

- call: Function call.
- formula: Formula used in the model fitting.
- data: A processed data frame used for model fitting.
- nObs: Number of observation.
- estimates:
 - beta: Coefficient estimates.
 - pi: Estimated parameters in prior multinomial distribution indicating the probabilities of corresponding record being true.
 - baseline: A data frame that contains estimated baseline hazard function with columns named time and h0.
- start: The initial guesses beta_mat and pi_mat specified for the parameters to be estimated, and the set of initial guess beta_start and pi_start that resulted in the largest objective function, i.e., the observed-data likelihood function.
- control: The control list specified for model fitting.
- na.action: The procedure specified to deal with missing values in the covariate.
- xlevels: A list that records the levels in each factor variable.
- contrasts: Contrasts specified and used for each factor variable.
- convergeCode: code returned by function [nlm](#), which is an integer indicating why the optimization process terminated. [help\(nlm\)](#) for details.
- logL: A numeric vector containing the observed-data log-likelihood over iterations.

References

Wang, W., Aseltine, R. H., Chen, K., & Yan, J. (2020). Integrative Survival Analysis with Uncertain Event Times in Application to A Suicide Risk Study. *Annals of Applied Statistics*, 14(1), 51–73.

See Also

[iCoxph.start](#) and [iCoxph.control](#), respectively, for starting and controlling iCoxph fitting; [summary](#), [iCoxph-method](#) for summary of fitted model; [coef](#), [iCoxph-method](#) for estimated covariate coefficients; [bootSe](#) for SE estimates from bootstrap methods.

Examples

```

library(intsurv)
## generate simulated survival data with uncertain records
set.seed(123)
simuDat <- simData4iCoxph(nSubject = 200)
## fit the integertive Cox model
fit <- iCoxph(Survi(ID, time, event) ~ x1 + x2 + x3 + x4,
             data = simuDat, start = iCoxph.start(methods = "nearest"),
             control = iCoxph.control(tol_beta = 1e-5))
## estimated covariate coefficients
coef(fit)
## get SE estimates by bootstrap
fit <- bootSe(fit, B = 30)
## summary of the fitted model
summary(fit)

```

iCoxph-class

An S4 Class to Represent a Fitted Integrative Cox Model

Description

The iCoxph class is an S4 class that represents a fitted model. Function [iCoxph](#) produces objects of this class. See “Slots” for details.

Slots

call Function call.
formula Model formula.
nObs A positive integer.
data A data frame.
estimates A list.
start A list.
control A list.
na.action A length-one character vector.
xlevels A list.
contrasts A list.
convergeCode A non-negative integer.
logL A numeric value.

See Also

[iCoxph](#) for details of slots.

iCoxph.control	<i>Auxiliary for Controlling iCoxph Fitting</i>
----------------	---

Description

Auxiliary function for `iCoxph` that enable users to specify the control parameters of the model estimation procedure. Internally, the arguments `cm_gradtol`, `cm_stepmax`, `cm_steptol`, and `cm_max_iter` are passed to function `nlm` as `gradtol`, `stepmax`, `steptol`, and `iterlim`, respectively.

Usage

```
iCoxph.control(
  tol_beta = 1e-06,
  tol_pi = 1e-08,
  cm_gradtol = 1e-06,
  cm_stepmax = 100,
  cm_steptol = 1e-06,
  cm_max_iter = 100,
  ecm_max_iter = 200,
  ...
)
```

Arguments

<code>tol_beta</code>	A positive value specifying the tolerance that concludes the convergence of the covariate coefficient estimates. The tolerance is compared with the relative change between the estimates from two consecutive iterations that is measured by ratio of the L2-norm of their difference to the sum of their L2-norm. The default value is $1e-6$.
<code>tol_pi</code>	A positive value specifying the tolerance that concludes the convergence of the probability estimates of uncertain records being true. The tolerance is compared with the relative change between the estimates from two consecutive iterations measured by ratio of L2-norm of their difference to the L2-norm of their sum. The default value is $1e-8$.
<code>cm_gradtol</code>	A positive scalar giving the tolerance at which the scaled gradient is considered close enough to zero to terminate CM steps. The default value is $1e-6$.
<code>cm_stepmax</code>	A positive scalar that gives the maximum allowable scaled step length in CM steps. The default value is $1e2$.
<code>cm_steptol</code>	A positive scalar providing the minimum allowable relative step length in CM step. The default value is $1e-6$.
<code>cm_max_iter</code>	A positive integer specifying the maximum number of iterations to be performed before each CM step is terminated. The default value is $1e2$.
<code>ecm_max_iter</code>	A positive integer specifying the maximum number of iterations to be performed before the ECM algorithm is terminated. The default value is $2e2$.
<code>...</code>	Other arguments for future usage. A warning will be thrown if any invalid argument is specified.

Value

A list of class `intsurv-iCoxph.control` containing all specified control parameters.

See Also

[iCoxph](#) for fitting integrative Cox model.

Examples

```
## See examples of function 'iCoxph'.
```

<code>iCoxph.start</code>	<i>Auxiliary for Starting iCoxph Fitting</i>
---------------------------	--

Description

Auxiliary function for [iCoxph](#) that enable users to specify the starting values of the model estimation procedure.

Usage

```
iCoxph.start(
  beta_vec = NULL,
  beta_mat = NULL,
  methods = c("nearest_hazard", "unit_hazard"),
  ...
)
```

Arguments

<code>beta_vec</code>	A numeric vector for starting values of coefficient estimates. The default values are the coefficient estimates from the regular Cox model only fitting on records without uncertainty. If censoring rate among subjects having unique certain records is extremely high (> 99 indicator and one predictor, the starting values will be reset to be all zeros.
<code>beta_mat</code>	A numeric matrix that consists of additional starting values of coefficient estimates in columns. The default value is <code>NULL</code> .
<code>methods</code>	A character vector specifying the initialization methods for probabilities of uncertain records being true. The available methods are "nearest_hazard" for initializing baseline hazard by nearest (left) neighbor, and "unit_hazard" for initializing unit baseline hazard. Partial matching on method names is supported for ease of typing. By default, both methods are used. See Wang et al. (2020) for details of the initialization methods.
<code>...</code>	Other arguments for future usage. A warning will be thrown if any invalid argument is specified.

Value

A list of class `intsurv-iCoxph.start` containing all specified starting values of the parameters to be estimated from the model.

See Also

[iCoxph](#) for fitting integrative Cox model.

Examples

```
## See examples of function 'iCoxph'.
```

show-method

Show Methods

Description

S4 class methods that display or summarize certain objects.

Usage

```
## S4 method for signature 'iCoxph'  
show(object)
```

```
## S4 method for signature 'summary.iCoxph'  
show(object)
```

Arguments

`object` An object used to dispatch a method.

Details

- For [iCoxph-class](#) object, it prints out a brief summary of the fitted model.
- For [summary.iCoxph-class](#) object, it prints out summary of a fitted model.

Value

object input (invisibly).

simData4cure

*Simulate Data from Cox Cure Model with Uncertain Event Status***Description**

Simulate Data from Cox Cure Model with Uncertain Event Status

Usage

```
simData4cure(
  nSubject = 1000,
  shape = 2,
  scale = 0.1,
  lambda_censor = 1,
  max_censor = Inf,
  p1 = 0.9,
  p2 = 0.9,
  p3 = 0.9,
  survMat,
  cureMat = survMat,
  b0 = stats::binomial()$linkfun(0.7),
  survCoef = rep(1, ncol(survMat)),
  cureCoef = rep(1, ncol(cureMat)),
  ...
)
```

Arguments

nSubject	A positive integer specifying number of subjects.
shape	A positive number specifying the shape parameter of the distribution of the event times.
scale	A positive number specifying the scale parameter of the distribution of the event times.
lambda_censor	A positive number specifying the rate parameter of the exponential distribution for generating censoring times.
max_censor	A positive number specifying the largest censoring time.
p1	A number between 0 and 1 specifying the probability of simulating events with observed event indicators given the simulated event times.
p2	A number between 0 and 1 specifying the probability of simulating susceptible censoring times with observed event status given the simulated susceptible censoring times.
p3	A number between 0 and 1 specifying the probability of simulating cured censoring times with observed event status given the simulated cured censoring times.
survMat	A numeric matrix representing the design matrix of the survival model part.

cureMat	A numeric matrix representing the design matrix excluding intercept of the cure rate model part.
b0	A number representing the intercept term for the cure rate model part.
survCoef	A numeric vector for the covariate coefficients of the survival model part.
cureCoef	A numeric vector for the covariate coefficients of the cure model part.
...	Other arguments not used currently.

Value

A data.frame with the following columns:

- obs_time: Observed event/survival times.
- obs_event: Observed event status.
- event_time: Underlying true event times.
- censor_time: underlying true censoring times.
- oracle_event: underlying true event indicators.
- oracle_cure: underlying true cure indicators.
- case: underlying true case labels.

References

Wang, W., Luo, C., Aseltine, R. H., Wang, F., Yan, J., & Chen, K. (2020). Suicide Risk Modeling with Uncertain Diagnostic Records. *arXiv preprint arXiv:2009.02597*.

Examples

```
## see examples of function cox_cure
```

simData4iCoxph	<i>Simulated Survival Data with Uncertain Records</i>
----------------	---

Description

Generate survival data with uncertain records. An integrative Cox model can be fitted for the simulated data by function [iCoxph](#).

Usage

```
simData4iCoxph(
  nSubject = 1000,
  beta0Vec,
  xMat,
  maxNum = 2,
  nRecordProb = c(0.9, 0.1),
  matchCensor = 0.1,
```

```

matchEvent = 0.1,
censorMin = 0.5,
censorMax = 12.5,
lambda = 0.005,
rho = 0.7,
fakeLambda1 = lambda * exp(-3),
fakeRho1 = rho,
fakeLambda2 = lambda * exp(3),
fakeRho2 = rho,
mixture = 0.5,
randomMiss = TRUE,
eventOnly = FALSE,
...
)

```

Arguments

nSubject	Number of subjects.
beta0Vec	Time-invariant covariate coefficients.
xMat	Design matrix. By default, three continuous variables following standard normal distribution and one binary variable following Bernoulli distribution with equal probability are used.
maxNum	Maximum number of uncertain records.
nRecordProb	Probability of the number of uncertain records.
matchCensor	The matching rate for subjects actually having censoring times.
matchEvent	The matching rate for subjects actually having event times.
censorMin	The lower boundary of the uniform distribution for generating censoring time.
censorMax	The upper boundary of the uniform distribution for generating censoring time.
lambda	A positive number, scale parameter in baseline rate function for true event times.
rho	A positive number, shape parameter in baseline rate function for true event times.
fakeLambda1	A positive number, scale parameter in baseline rate function for fake event times from one distribution.
fakeRho1	A positive number, shape parameter in baseline rate function for fake event times from one distribution.
fakeLambda2	A positive number, scale parameter in baseline rate function for fake event times from another distribution.
fakeRho2	A positive number, shape parameter in baseline rate function for fake event times from another distribution.
mixture	The mixture weights, i.e., the probabilities (summing up to one) of fake event times coming from different mixture components.
randomMiss	A logical value specifying whether the labels of the true records are missing completely at random (MCAR) or missing not at random (MNAR). The default value is TRUE for MCAR.

eventOnly	A logical value specifying whether the uncertain records only include possible events. The default value is FALSE, which considers the censoring cases as the possible truth in addition to event records.
...	Other arguments for future usage.

Details

The event times are simulated from a Weibull proportional hazard model of given shape and baseline scale. The censoring times follow uniform distribution of specified boundaries.

Value

A data frame with the following columns,

- ID: subject ID
- time: observed event times
- event: event indicators
- isTure: latent labels indicating the true records

and the corresponding covariates.

Examples

```
## See examples of function iCoxph
```

summary,iCoxph-method *Summary of a Fitted Model*

Description

For `iCoxph` object, the function returns a `summary.iCoxph` object whose slots include

- call: Function call of model fitting.
- coefMat: Estimated covariate coefficients.
- logL: Log-likelihood under observed data.

Usage

```
## S4 method for signature 'iCoxph'
summary(object, showCall = TRUE, ...)
```

Arguments

object	<code>iCoxph-class</code> object.
showCall	A logic value with default TRUE, indicating whether show method prints out the original call information of <code>iCoxph</code> . Set FALSE for a more concise printout.
...	Other arguments for future usage.

Value

summary.iCoxph class object.

See Also

[iCoxph](#) for model fitting; [coef, iCoxph-method](#) for coefficient estimates.

Examples

```
## See examples of function iCoxph
```

summary.cox_cure	<i>Summary of a Fitted Model</i>
------------------	----------------------------------

Description

Summarize a fitted Cox cure rate model with possible uncertain event status.

Usage

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

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

Arguments

object	Object representing a fitted model.
...	Other arguments for future usage. A warning will be thrown if any invalid argument is specified.

summary.iCoxph-class	<i>An S4 Class to Represent Summary of a fitted integrative Cox model</i>
----------------------	---

Description

The `summary.intsurv` class is an S4 class that represents a summarized model. Function [summary, iCoxph-method](#) produces objects of this class. See “Slots” for details.

Slots

call	Function call.
coefMat	A matrix.
logL	A numeric value.

See Also

[summary](#), [iCoxph-method](#) for meaning of slots.

 Survival

Formula Response for Survival Data With Uncertain Event Times

Description

Survival returns an S4 class that represents formula response for survival data with uncertain records due to imperfect data integration. The last letter 'i' in Survival represents 'integration'.

Usage

```
Survival(ID, time, event, check = TRUE, ...)
```

Arguments

ID	Identifier of each subject.
time	Event times (whether certain or uncertain) or censoring times.
event	The status indicator, 0 = censored, 1 = event.
check	A logical value specifying whether to perform check on input data.
...	Other arguments for future usage. A warning will be thrown if any invalid argument is specified.

Value

Survival object. See [Survival-class](#) for details.

Examples

```
## See examples of function 'iCoxph'
```

 Survival-class

An S4 Class Representing Formula Response

Description

An S4 Class Representing Formula Response

Slots

.Data A numeric matrix object.
ID Identifier of each subject.

Index

BIC.cox_cure, [3](#)
BIC.cox_cure_net, [4](#), [9](#)
BIC.cox_cure_net_uncer
 (BIC.cox_cure_net), [4](#)
BIC.cox_cure_uncer (BIC.cox_cure), [3](#)
bootSe, [5](#), [23](#)

cIndex, [6](#)
coef, iCoxph-method, [7](#)
coef.cox_cure, [8](#)
coef.cox_cure_net, [9](#)
coef.cox_cure_net_uncer
 (coef.cox_cure_net), [9](#)
coef.cox_cure_uncer (coef.cox_cure), [8](#)
coef.iCoxph (coef, iCoxph-method), [7](#)
cox_cure, [10](#), [20](#)
cox_cure_net, [13](#), [15](#)

iCoxph, [6](#), [7](#), [22](#), [24–27](#), [29](#), [31](#), [32](#)
iCoxph-class, [24](#)
iCoxph.control, [23](#), [25](#)
iCoxph.start, [23](#), [26](#)
intsurv-package, [2](#)

model.matrix.default, [11](#), [17](#), [22](#)

na.exclude, [22](#)
na.fail, [22](#)
na.omit, [22](#)
na.pass, [22](#)
nlm, [23](#), [25](#)

options, [22](#)

show, iCoxph-method (show-method), [27](#)
show, summary.iCoxph-method
 (show-method), [27](#)
show-method, [27](#)
simData4cure, [28](#)
simData4iCoxph, [29](#)
summary, iCoxph-method, [31](#)
summary.cox_cure, [32](#)
summary.cox_cure_uncer
 (summary.cox_cure), [32](#)
summary.iCoxph, [31](#)
summary.iCoxph (summary, iCoxph-method),
 [31](#)
summary.iCoxph-class, [32](#)
Surv, [33](#)
Surv-class, [33](#)