Package 'StressCensoR'

August 5, 2026

Type Package

Title Generalized Stress-Strength Reliability Estimation Under Censoring Schemes

Version 0.1.0

Description Generalized framework for data generation, Maximum Likelihood Estimation, and Bayesian estimation of stress-strength reliability R = P(Y < X) for arbitrary continuous distributions under censoring schemes based on Chapter 9 of 'Balakrishnan', 'Cramer', and 'Kundu' (2023) <ISBN:978-0-12-398387-9>. Users provide probability density functions, cumulative distribution functions, survival functions, support bounds, parameter ranges, and sample sizes. Implements data generation under Type-I, Type-II, progressive Type-II, Type-I hybrid, Type-II hybrid, generalized hybrid, progressive hybrid, joint, block random, middle, and truncation censoring schemes, accompanied by diagnostic histograms, dot plots, and autocorrelation plots. Maximum Likelihood Estimation supports optimization routines including 'Newton-Raphson', 'Broyden'-'Fletcher'-'Goldfarb'-'Shanno' ('BFGS'), 'BFGS' in R ('BFGSR'), 'Berndt'-'Hall'-'Hall'-'Hausman' ('BHHH'), Simulated Annealing ('SANN'), Conjugate Gradients ('CG'), and 'Nelder'-'Mead' ('NM'), returning summaries ('AIC', 'coef', 'logLik', 'nIter', 'stdEr', summary, 'vcov'). Bayesian estimation of stress-strength reliability R = P(Y < X) is performed via Gibbs sampling, Metropolis-Hastings algorithm, Importance Sampling, and 'Lindley' approximation (1980). Methods and censoring schemes are described in 'Balakrishnan', 'Cramer', and 'Kundu' (2023, ISBN:978-0-12-398387-9), 'Lindley' (1980) <doi:10.1111/j.2517-6161.1980.tb01102.x>, 'Geweke' (1989) <doi:10.2307/2290062>, 'Metropolis' (1953) <doi:10.1063/1.1699114>, 'Hastings' (1970) <doi:10.1093/biomet/57.1.97>, 'Geman' and 'Geman' (1984) <doi:10.1109/TPAMI.1984.4767596>, 'Kundu' and 'Gupta' (2005) <doi:10.1016/j.jspi.2004.09.006>, 'Kundu' and 'Gupta' (2006) <doi:10.1016/j.csda.2005.02.007>, 'Berndt', 'Hall', 'Hall', and 'Hausman' (1974) <doi:10.3386/t0003>, 'Fletcher' (1987, ISBN:978-0-471-91547-8), and 'Nelder' and 'Mead' (1965) <doi:10.1093/comjnl/7.4.308>.

License GPL-3

Encoding UTF-8

Language en-US

Depends R (>= 4.0.0)

Imports graphics, stats

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation no

Author Shikhar Tyagi [aut, cre] (ORCID:
<https://orcid.org/0000-0003-1606-0844>),
Arvind Pandey [aut],
Bhupendra Singh [aut],
Vrijesh Tripathi [aut]

Maintainer Shikhar Tyagi <shikhar1093tyagi@gmail.com>

Repository CRAN

Date/Publication 2026-08-05 06:40:23 UTC

Contents

`bayes_stress_strength` *Master Function for Bayesian Stress-Strength Reliability Estimation*

Description

Fits Bayesian stress-strength reliability R = P(Y < X) using Gibbs sampling, Metropolis-Hastings, Importance Sampling, or Lindley Approximation.

Usage

```
bayes_stress_strength(
  data_x,
  data_y,
  pdf_x,
  cdf_y,
  method = "gibbs",
  start_x = 1,
  start_y = 1,
  ...
)
```

Arguments

`data_x`	Stress data vector X.
`data_y`	Strength data vector Y.
`pdf_x`	PDF function for X.
`cdf_y`	CDF function for Y.
`method`	Bayesian estimation method: "gibbs", "mh", "is", or "lindley".
`start_x`	Initial parameter guess for X.
`start_y`	Initial parameter guess for Y.
`...`	Additional control arguments passed to specific algorithms.

Value

List of Bayesian estimation results.

`coef.mle_censored` *S3 Methods for mle_censored Objects*

Description

S3 Methods for mle_censored Objects

Usage

```
## S3 method for class 'mle_censored'
coef(object, ...)

## S3 method for class 'mle_censored'
vcov(object, ...)

## S3 method for class 'mle_censored'
logLik(object, ...)

## S3 method for class 'mle_censored'
AIC(object, ..., k = 2)

## S3 method for class 'mle_censored'
stdEr(object, ...)

## S3 method for class 'mle_censored'
nIter(object, ...)

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

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

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

Arguments

`object`	An object of class `mle_censored`.
`...`	Additional arguments.
`k`	Penalty per parameter for AIC calculation (default is 2).
`x`	An object of class `mle_censored` or `summary.mle_censored`.

Value

Extracted method values (coefficients, vcov matrix, logLik, AIC, stdEr, nIter, summary, or printed output).

`compute_stress_strength_R` *Compute Stress-Strength Reliability R = P(Y < X)*

Description

Computes R = P(Y < X) given PDF/CDF functions and parameter vectors.

Usage

```
compute_stress_strength_R(
  pdf_x,
  cdf_y,
  param_x,
  param_y,
  lower = 0,
  upper = Inf
)
```

Arguments

`pdf_x`	PDF function for X (stress), takes (x, param_x).
`cdf_y`	CDF function for Y (strength), takes (y, param_y).
`param_x`	Parameter vector for X.
`param_y`	Parameter vector for Y.
`lower`	Lower support bound.
`upper`	Upper support bound.

Value

Numeric scalar value of R in [0, 1].

example_aluminum_coupons *Solved Example 11.3: Aluminum Coupons Lifetime Data Under Hybrid Censoring*

Description

Solves Example 11.3 from Chapter 9 of Balakrishnan, Cramer, and Kundu (2023) using Type-I and Type-II Hybrid Censoring schemes (Birnbaum & Saunders 1958 dataset).

Usage

```
example_aluminum_coupons()
```

Value

A list containing dataset, hybrid censoring limits, MLEs, and Bayesian estimates.

Examples

```
res <- example_aluminum_coupons()
print(res$mle_hybrid1)
print(res$mle_hybrid2)
```

example_insulating_fluid *Solved Example 11.2: Insulating Fluid Breakdown Data Under Progressive Censoring*

Description

Solves Example 11.2 from Chapter 9 of Balakrishnan, Cramer, and Kundu (2023) using progressive Type-II censoring estimation (MLE and Bayesian methods).

Usage

```
example_insulating_fluid()
```

Value

A list containing dataset, MLE estimates, and Bayesian stress-strength reliability estimates.

Examples

```
res <- example_insulating_fluid()
print(res$mle_fit)
print(res$bayes_fit)
```

example_stress_strength *Solved Example: Stress-Strength Reliability Under Hybrid Censoring*

Description

Solves Stress-Strength Reliability R = P(Y < X) for stress X and strength Y under Chapter 9 hybrid censoring schemes using MLE and Bayesian methods.

Usage

```
example_stress_strength()
```

Value

A list containing stress/strength datasets, MLE, Gibbs, M-H, IS, and Lindley estimates.

Examples

```
res <- example_stress_strength()
print(res$R_mle)
print(res$R_lindley)
```

gibbs_stress_strength *Bayesian Stress-Strength Reliability Estimation via Gibbs Sampling*

Description

Bayesian Stress-Strength Reliability Estimation via Gibbs Sampling

Usage

```
gibbs_stress_strength(
  data_x,
  data_y,
  pdf_x,
  cdf_y,
  prior_x = NULL,
  prior_y = NULL,
  start_x = 1,
  start_y = 1,
  iter = 5000,
  burnin = 1000
)
```

Arguments

data_x	Stress data vector X.
data_y	Strength data vector Y.
pdf_x	PDF function for X.
cdf_y	CDF function for Y.
prior_x	Prior density for parameter of X.
prior_y	Prior density for parameter of Y.
start_x	Initial parameter guess for X.
start_y	Initial parameter guess for Y.
iter	Number of MCMC iterations.
burnin	Number of burn-in iterations.

Value

A list containing posterior samples of parameters and R, and point estimates.

---

is_stress_strength *Bayesian Stress-Strength Reliability Estimation via Importance Sampling*

Description

Bayesian Stress-Strength Reliability Estimation via Importance Sampling

Usage

```
is_stress_strength(
  data_x,
  data_y,
  pdf_x,
  cdf_y,
  proposal_sampler = NULL,
  proposal_density = NULL,
  n_samples = 2000
)
```

Arguments

data_x	Stress data vector X.
data_y	Strength data vector Y.
pdf_x	PDF function for X.
cdf_y	CDF function for Y.

`proposal_sampler` Function generating proposal parameter draws.

`proposal_density` Function computing proposal log-density.

`n_samples` Number of importance samples.

Value

A list containing importance weights, R estimate, and standard error.

`lindley_stress_strength` *Bayesian Stress-Strength Reliability Estimation via Lindley's Approximation (1980)*

Description

Bayesian Stress-Strength Reliability Estimation via Lindley's Approximation (1980)

Usage

```
lindley_stress_strength(data_x, data_y, pdf_x, cdf_y, start_x = 1, start_y = 1)
```

Arguments

`data_x` Stress data vector X.

`data_y` Strength data vector Y.

`pdf_x` PDF function for X.

`cdf_y` CDF function for Y.

`start_x` Initial parameter guess for X.

`start_y` Initial parameter guess for Y.

Value

A list containing Lindley approximation estimate of R.

`mh_stress_strength` *Bayesian Stress-Strength Reliability Estimation via Metropolis-Hastings Algorithm*

Description

Bayesian Stress-Strength Reliability Estimation via Metropolis-Hastings Algorithm

Usage

```
mh_stress_strength(
  data_x,
  data_y,
  pdf_x,
  cdf_y,
  prior_x = NULL,
  prior_y = NULL,
  start_x = 1,
  start_y = 1,
  iter = 5000,
  burnin = 1000
)
```

Arguments

`data_x`	Stress data vector X.
`data_y`	Strength data vector Y.
`pdf_x`	PDF function for X.
`cdf_y`	CDF function for Y.
`prior_x`	Prior density for parameter of X.
`prior_y`	Prior density for parameter of Y.
`start_x`	Initial parameter guess for X.
`start_y`	Initial parameter guess for Y.
`iter`	Number of MCMC iterations.
`burnin`	Number of burn-in iterations.

Value

A list containing posterior samples of parameters and R, and point estimates.

`mle_censored` *Generalized Maximum Likelihood Estimation Under Censoring Schemes*

Description

Fits parameter estimates and calculates stress-strength reliability R = P(Y < X) under user-specified censoring schemes and optimization algorithms.

Usage

```
mle_censored(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  scheme = "Type-I",
  method = "NR",
  data_y = NULL,
  pdf_y = NULL,
  cdf_y = NULL,
  start_y = NULL,
  ...
)
```

Arguments

`data`	Observed data vector or data frame.
`pdf`	PDF function f(x, param).
`cdf`	CDF function F(x, param).
`surv`	Survival function S(x, param).
`start`	Initial parameter vector.
`scheme`	Censoring scheme name.
`method`	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
`data_y`	Optional observed strength data Y.
`pdf_y`	Optional PDF function for Y.
`cdf_y`	Optional CDF function for Y.
`start_y`	Optional initial parameter vector for Y.
`...`	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

mle_genhybrid *MLE Under Generalized Hybrid Censoring*

Description

MLE Under Generalized Hybrid Censoring

Usage

```
mle_genhybrid(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

mle_hybrid1 *MLE Under Type-I Hybrid Censoring*

Description

MLE Under Type-I Hybrid Censoring

Usage

```
mle_hybrid1(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

`mle_hybrid2` *MLE Under Type-II Hybrid Censoring*

Description

MLE Under Type-II Hybrid Censoring

Usage

```
mle_hybrid2(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

`mle_interval` *MLE Under Interval Censoring*

Description

MLE Under Interval Censoring

Usage

```
mle_interval(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

`mle_joint` *MLE Under Joint Censoring*

Description

MLE Under Joint Censoring

Usage

```
mle_joint(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

`data`	Observed data vector or data frame.
`pdf`	PDF function f(x, param).
`cdf`	CDF function F(x, param).
`surv`	Survival function S(x, param).
`start`	Initial parameter vector.
`method`	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
`...`	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

mle_middle *MLE Under Middle Censoring*

Description

MLE Under Middle Censoring

Usage

```
mle_middle(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

mle_prog2 *MLE Under Progressive Type-II Censoring*

Description

MLE Under Progressive Type-II Censoring

Usage

```
mle_prog2(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

`mle_proghybrid` *MLE Under Progressive Hybrid Censoring*

Description

MLE Under Progressive Hybrid Censoring

Usage

```
mle_proghybrid(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

`mle_random` *MLE Under Random Censoring*

Description

MLE Under Random Censoring

Usage

```
mle_random(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

`mle_trunc` *MLE Under Truncated Distribution*

Description

MLE Under Truncated Distribution

Usage

```
mle_trunc(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

`data`	Observed data vector or data frame.
`pdf`	PDF function f(x, param).
`cdf`	CDF function F(x, param).
`surv`	Survival function S(x, param).
`start`	Initial parameter vector.
`method`	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
`...`	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

mle_type1 *MLE Under Type-I Censoring*

Description

MLE Under Type-I Censoring

Usage

```
mle_type1(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

mle_type2 *MLE Under Type-II Censoring*

Description

MLE Under Type-II Censoring

Usage

```
mle_type2(
  data,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start = NULL,
  method = "NR",
  ...
)
```

Arguments

data	Observed data vector or data frame.
pdf	PDF function f(x, param).
cdf	CDF function F(x, param).
surv	Survival function S(x, param).
start	Initial parameter vector.
method	Maximisation method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
...	Further control parameters passed to optimization.

Value

An object of class `mle_censored`.

---

nIter *Number of Iterations Generic Function*

Description

Number of Iterations Generic Function

Usage

```
nIter(object, ...)
```

Arguments

object	Model fit object.
...	Additional arguments.

Value

Number of optimization iterations.

plot.rcensored *Plot Method for Diagnostic Verification of Censored Samples*

Description

Produces diagnostic plots including Histogram with PDF overlay, Dot Plot, and Autocorrelation (ACF) plot for generated samples.

Usage

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

Arguments

x	An object of class rcensored.
...	Additional graphical parameters.

Value

Invisibly returns x.

rcensor_genhybrid *Data Generation Under Generalized Hybrid Censoring*

Description

Data Generation Under Generalized Hybrid Censoring

Usage

```
rcensor_genhybrid(
  n,
  k,
  r,
  T0,
  cdf = NULL,
  pdf = NULL,
  surv = NULL,
  param = NULL,
  lower = 0,
  upper = Inf,
  seed = NULL
)
```

Arguments

n	Total sample size.
k	Minimum failure count.
r	Maximum failure count.
T0	Time threshold.
cdf	CDF function.
pdf	PDF function.
surv	Survival function.
param	Parameter vector.
lower	Lower support limit.
upper	Upper support limit.
seed	Optional random seed for reproducibility.

Value

An object of class rcensored.

---

rcensor_hybrid1 *Data Generation Under Type-I Hybrid Censoring*

Description

Data Generation Under Type-I Hybrid Censoring

Usage

```
rcensor_hybrid1(
  n,
  r,
  T0,
  cdf = NULL,
  pdf = NULL,
  surv = NULL,
  param = NULL,
  lower = 0,
  upper = Inf,
  seed = NULL
)
```

Arguments

n	Total sample size.
r	Failure count threshold.
T0	Time threshold.
cdf	CDF function.
pdf	PDF function.
surv	Survival function.
param	Parameter vector.
lower	Lower support limit.
upper	Upper support limit.
seed	Optional random seed for reproducibility.

Value

An object of class rcensored.

---

rcensor_hybrid2 *Data Generation Under Type-II Hybrid Censoring*

Description

Data Generation Under Type-II Hybrid Censoring

Usage

```
rcensor_hybrid2(
  n,
  r,
  T0,
  cdf = NULL,
  pdf = NULL,
  surv = NULL,
  param = NULL,
  lower = 0,
  upper = Inf,
  seed = NULL
)
```

Arguments

n	Total sample size.
r	Failure count threshold.
T0	Time threshold.
cdf	CDF function.
pdf	PDF function.
surv	Survival function.
param	Parameter vector.
lower	Lower support limit.
upper	Upper support limit.
seed	Optional random seed for reproducibility.

Value

An object of class rcensored.

---

rcensor_interval *Data Generation Under Interval Censoring*

Description

Data Generation Under Interval Censoring

Usage

```
rcensor_interval(n, cdf = NULL, pdf = NULL, param = NULL, seed = NULL)
```

Arguments

n	Sample size.
cdf	CDF function.
pdf	PDF function.
param	Parameter vector.
seed	Optional random seed.

Value

An object of class rcensored.

`rcensor_joint` *Data Generation Under Joint Censoring Scheme*

Description

Data Generation Under Joint Censoring Scheme

Usage

```
rcensor_joint(
  n1,
  n2,
  r,
  cdf_x = NULL,
  cdf_y = NULL,
  param_x = NULL,
  param_y = NULL,
  seed = NULL
)
```

Arguments

`n1`	Sample size of group X.
`n2`	Sample size of group Y.
`r`	Total combined failure count.
`cdf_x`	CDF function for X.
`cdf_y`	CDF function for Y.
`param_x`	Parameter vector for X.
`param_y`	Parameter vector for Y.
`seed`	Optional random seed.

Value

An object of class `rcensored`.

rcensor_middle *Data Generation Under Middle Censoring*

Description

Data Generation Under Middle Censoring

Usage

```
rcensor_middle(n, cdf = NULL, pdf = NULL, param = NULL, seed = NULL)
```

Arguments

n	Sample size.
cdf	CDF function.
pdf	PDF function.
param	Parameter vector.
seed	Optional random seed.

Value

An object of class rcensored.

rcensor_prog2 *Data Generation Under Progressive Type-II Censoring*

Description

Data Generation Under Progressive Type-II Censoring

Usage

```
rcensor_prog2(
  n,
  r,
  R,
  cdf = NULL,
  pdf = NULL,
  surv = NULL,
  param = NULL,
  lower = 0,
  upper = Inf,
  seed = NULL
)
```

Arguments

n	Total sample size.
r	Number of failures.
R	Removal scheme vector of length r.
cdf	CDF function.
pdf	PDF function.
surv	Survival function.
param	Parameter vector.
lower	Lower support limit.
upper	Upper support limit.
seed	Optional random seed for reproducibility.

Value

An object of class rcensored.

---

rcensor_proghybrid *Data Generation Under Progressive Hybrid Censoring*

Description

Data Generation Under Progressive Hybrid Censoring

Usage

```
rcensor_proghybrid(
  n,
  r,
  R,
  T0,
  cdf = NULL,
  pdf = NULL,
  surv = NULL,
  param = NULL,
  lower = 0,
  upper = Inf,
  seed = NULL
)
```

Arguments

n	Total sample size.
r	Failure count threshold.
R	Removal scheme vector.
T0	Time threshold.
cdf	CDF function.
pdf	PDF function.
surv	Survival function.
param	Parameter vector.
lower	Lower support limit.
upper	Upper support limit.
seed	Optional random seed for reproducibility.

Value

An object of class rcensored.

---

rcensor_random *Data Generation Under Random Censoring*

Description

Data Generation Under Random Censoring

Usage

```
rcensor_random(
  n,
  cdf = NULL,
  pdf = NULL,
  param = NULL,
  cen_cdf = NULL,
  cen_param = NULL,
  seed = NULL
)
```

Arguments

n	Sample size.
cdf	CDF function.
pdf	PDF function.
param	Parameter vector.
cen_cdf	Censoring distribution CDF function.
cen_param	Censoring distribution parameters.
seed	Optional random seed.

Value

An object of class `rcensored`.

`rcensor_trunc` *Data Generation Under Truncated Distribution*

Description

Data Generation Under Truncated Distribution

Usage

```
rcensor_trunc(
  n,
  cdf = NULL,
  pdf = NULL,
  param = NULL,
  left_trunc = -Inf,
  right_trunc = Inf,
  seed = NULL
)
```

Arguments

`n`	Sample size.
`cdf`	CDF function.
`pdf`	PDF function.
`param`	Parameter vector.
`left_trunc`	Left truncation point.
`right_trunc`	Right truncation point.
`seed`	Optional random seed.

Value

An object of class `rcensored`.

`rcensor_type1` *Data Generation Under Type-I Censoring*

Description

Data Generation Under Type-I Censoring

Usage

```
rcensor_type1(
  n,
  T0,
  cdf = NULL,
  pdf = NULL,
  surv = NULL,
  param = NULL,
  lower = 0,
  upper = Inf,
  seed = NULL
)
```

Arguments

`n`	Sample size.
`T0`	Censoring time point.
`cdf`	CDF function.
`pdf`	PDF function.
`surv`	Survival function.
`param`	Parameter vector.
`lower`	Lower support limit.
`upper`	Upper support limit.
`seed`	Optional random seed for reproducibility.

Value

An object of class `rcensored`.

`rcensor_type2` *Data Generation Under Type-II Censoring*

Description

Data Generation Under Type-II Censoring

Usage

```
rcensor_type2(
  n,
  r,
  cdf = NULL,
  pdf = NULL,
  surv = NULL,
  param = NULL,
  lower = 0,
  upper = Inf,
  seed = NULL
)
```

Arguments

`n`	Total sample size.
`r`	Number of failures to observe.
`cdf`	CDF function.
`pdf`	PDF function.
`surv`	Survival function.
`param`	Parameter vector.
`lower`	Lower support limit.
`upper`	Upper support limit.
`seed`	Optional random seed for reproducibility.

Value

An object of class `rcensored`.

stdEr *Standard Error Generic Function*

Description

Standard Error Generic Function

Usage

```
stdEr(object, ...)
```

Arguments

object	Model fit object.
...	Additional arguments.

Value

Vector of standard errors.

Index