Package 'AdaptHyCensor'

August 4, 2026

Type Package

Title Generalized Inference and Data Generation for Adaptive Progressive Hybrid Censoring Schemes

Version 0.1.0

Description Comprehensive computational tools for data generation, statistical inference, and visual diagnostics under Adaptive Type-I and Adaptive Type-II Progressive Hybrid Censoring Schemes. Users can supply custom probability density functions (PDF), cumulative distribution functions (CDF), survival functions, parameter ranges, and progressive schemes for any univariate lifetime distribution. Parameter estimation methods include Maximum Likelihood Estimation (MLE) using multiple optimization algorithms (Newton-Raphson (NR), Broyden-Fletcher-Goldfarb-Shanno (BFGS), BFGS in R (BFGSR), Berndt-Hall-Hall-Hausman (BHHH), Simulated Annealing (SANN), Conjugate Gradients (CG), and Nelder-Mead (NM)), Bayesian estimation via Gibbs and Metropolis-Hastings (M-H) MCMC sampling, Importance Sampling (IS), and Lindley's approximation. Diagnostic tools provide histograms, dot plots, and autocorrelation function (ACF) plots for model validation. Methods are based on Balakrishnan, Cramer, and Kundu (2023, ISBN:978-0-12-398387-9), Ng, Kundu, and Chan (2009, IEEE Transactions on Reliability, 58, 634-642), Lin and Huang (2012, Journal of Statistical Computation and Simulation, 82, 1005-1018), Lindley (1980, Journal of the Royal Statistical Society, Series B, 42, 223-237), and Berndt, Hall, Hall, and Hausman (1974, Annals of Economic and Social Measurement, 3, 653-665).

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.0.0)

Imports stats, graphics

Suggests testthat (>= 3.0.0)

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 2026-08-04 14:10:45 UTC

Contents

`AIC.mle_adapt_fit` *Generic AIC Function for mle_adapt_fit*

Description

Generic AIC Function for mle_adapt_fit

Usage

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

Arguments

`object`	An object of class `mle_adapt_fit`.
`...`	Additional arguments.
`k`	Penalty parameter, default is 2.

Value

A numeric value representing AIC.

`bayes_adapt1_phcs` *Bayesian Estimation via Metropolis-Hastings MCMC under Adaptive Type-I PHCS*

Description

Performs Bayesian parameter estimation under Adaptive Type-I Progressive Hybrid Censoring using Metropolis-Hastings / Gibbs MCMC sampling.

Usage

```
bayes_adapt1_phcs(
  data,
  n,
  m,
  T_thresh,
  R,
  pdf,
  cdf = NULL,
  surf = NULL,
  prior = NULL,
  start,
  N_iter = 5000,
  burn_in = 1000,
  proposal_sd = NULL,
  lower = NULL,
  upper = NULL
)
```

Arguments

`data`	Numeric vector of observed failure times.
`n`	Integer, total initial sample size placed on test.
`m`	Integer, pre-fixed target number of failures.
`T_thresh`	Numeric, pre-fixed time threshold T > 0.
`R`	Numeric vector of length m, progressive censoring scheme.
`pdf`	Function, user-supplied probability density function pdf(x, par).
`cdf`	Function (optional), user-supplied cumulative distribution function cdf(x, par).
`surf`	Function (optional), user-supplied survival function surf(x, par).
`prior`	Function, user-supplied prior density function log_prior(par). Default is flat prior.
`start`	Numeric vector, initial parameter values.
`N_iter`	Integer, total number of MCMC iterations (default 5000).
`burn_in`	Integer, burn-in iterations to discard (default 1000).

proposal_sd	Numeric vector, standard deviations for random-walk proposal.
lower	Numeric vector (optional), lower parameter constraints.
upper	Numeric vector (optional), upper parameter constraints.

Value

A list containing posterior means, medians, modes, standard deviations, HPD credible intervals, and parameter chain samples.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
my_surf <- function(x, par) 1 - pexp(x, rate = par[1])
flat_prior <- function(par) ifelse(par[1] > 0, 0, -Inf)
R_plan <- c(rep(1, 8), rep(0, 4))
dat <- r_adapt1_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)$data
bayes_res <- bayes_adapt1_phcs(
  data = dat, n = 20, m = 12,
  T_thresh = 2.5, R = R_plan,
  pdf = my_pdf, cdf = my_cdf,
  surf = my_surf, prior = flat_prior,
  start = c(rate = 0.8), N_iter = 1000,
  burn_in = 200
)
print(bayes_res$summary)
```

---

bayes_adapt2_phcs *Bayesian Estimation via Metropolis-Hastings MCMC under Adaptive Type-II PHCS*

Description

Performs Bayesian parameter estimation under Adaptive Type-II Progressive Hybrid Censoring using Metropolis-Hastings / Gibbs MCMC sampling.

Usage

```
bayes_adapt2_phcs(
  data,
  n,
  m,
  T_thresh,
```

```
  R,
  pdf,
  cdf = NULL,
  surf = NULL,
  prior = NULL,
  start,
  N_iter = 5000,
  burn_in = 1000,
  proposal_sd = NULL,
  lower = NULL,
  upper = NULL
)
```

Arguments

data	Numeric vector of m observed failure times.
n	Integer, total initial sample size placed on test.
m	Integer, pre-fixed target number of failures.
T_thresh	Numeric, pre-fixed time threshold T > 0.
R	Numeric vector of length m, progressive censoring scheme.
pdf	Function, user-supplied probability density function pdf(x, par).
cdf	Function (optional), user-supplied cumulative distribution function cdf(x, par).
surf	Function (optional), user-supplied survival function surf(x, par).
prior	Function, user-supplied prior density function log_prior(par). Default is flat prior.
start	Numeric vector, initial parameter values.
N_iter	Integer, total number of MCMC iterations (default 5000).
burn_in	Integer, burn-in iterations to discard (default 1000).
proposal_sd	Numeric vector, standard deviations for random-walk proposal.
lower	Numeric vector (optional), lower parameter constraints.
upper	Numeric vector (optional), upper parameter constraints.

Value

A list containing posterior means, medians, modes, standard deviations, HPD credible intervals, and parameter chain samples.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
my_surf <- function(x, par) 1 - pexp(x, rate = par[1])
flat_prior <- function(par) ifelse(par[1] > 0, 0, -Inf)
R_plan <- c(rep(1, 8), rep(0, 4))
```

```
dat <- r_adapt2_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)$data
bayes_res <- bayes_adapt2_phcs(
  data = dat, n = 20, m = 12,
  T_thresh = 2.5, R = R_plan,
  pdf = my_pdf, cdf = my_cdf,
  surf = my_surf, prior = flat_prior,
  start = c(rate = 0.8), N_iter = 1000,
  burn_in = 200
)
print(bayes_res$summary)
```

`is_adapt1_phcs` *Importance Sampling Estimation under Adaptive Type-I PHCS*

Description

Estimates parameters and posterior expectations via Importance Sampling under Adaptive Type-I PHCS.

Usage

```
is_adapt1_phcs(
  data,
  n,
  m,
  T_thresh,
  R,
  pdf,
  cdf = NULL,
  surf = NULL,
  prior = NULL,
  start,
  proposal_cov = NULL,
  N_samples = 5000,
  lower = NULL,
  upper = NULL
)
```

Arguments

`data`	Numeric vector of observed failure times.
`n`	Integer, total initial sample size placed on test.
`m`	Integer, pre-fixed target number of failures.

T_thresh	Numeric, pre-fixed time threshold T > 0.
R	Numeric vector of length m, progressive censoring scheme.
pdf	Function, user-supplied probability density function pdf(x, par).
cdf	Function (optional), user-supplied cumulative distribution function cdf(x, par).
surf	Function (optional), user-supplied survival function surf(x, par).
prior	Function (optional), user-supplied prior density function log_prior(par).
start	Numeric vector, center for proposal distribution (e.g., MLE or initial guess).
proposal_cov	Matrix (optional), covariance matrix for Gaussian proposal distribution.
N_samples	Integer, number of importance samples (default 5000).
lower	Numeric vector (optional), lower parameter bounds.
upper	Numeric vector (optional), upper parameter bounds.

Value

A list containing estimated posterior mean, standard error, effective sample size (ESS), and importance weights.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
my_surf <- function(x, par) 1 - pexp(x, rate = par[1])
flat_prior <- function(par) ifelse(par[1] > 0, 0, -Inf)
R_plan <- c(rep(1, 8), rep(0, 4))
dat <- r_adapt1_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)$data
is_res <- is_adapt1_phcs(
  data = dat, n = 20, m = 12,
  T_thresh = 2.5, R = R_plan,
  pdf = my_pdf, cdf = my_cdf,
  surf = my_surf, prior = flat_prior,
  start = c(rate = 0.5), N_samples = 1000
)
print(is_res$summary)
```

---

is_adapt2_phcs *Importance Sampling Estimation under Adaptive Type-II PHCS*

Description

Estimates parameters and posterior expectations via Importance Sampling under Adaptive Type-II PHCS.

Usage

```
is_adapt2_phcs(
  data,
  n,
  m,
  T_thresh,
  R,
  pdf,
  cdf = NULL,
  surf = NULL,
  prior = NULL,
  start,
  proposal_cov = NULL,
  N_samples = 5000,
  lower = NULL,
  upper = NULL
)
```

Arguments

data	Numeric vector of m observed failure times.
n	Integer, total initial sample size placed on test.
m	Integer, pre-fixed target number of failures.
T_thresh	Numeric, pre-fixed time threshold T > 0.
R	Numeric vector of length m, progressive censoring scheme.
pdf	Function, user-supplied probability density function pdf(x, par).
cdf	Function (optional), user-supplied cumulative distribution function cdf(x, par).
surf	Function (optional), user-supplied survival function surf(x, par).
prior	Function (optional), user-supplied prior density function log_prior(par).
start	Numeric vector, center for proposal distribution (e.g., MLE or initial guess).
proposal_cov	Matrix (optional), covariance matrix for Gaussian proposal distribution.
N_samples	Integer, number of importance samples (default 5000).
lower	Numeric vector (optional), lower parameter bounds.
upper	Numeric vector (optional), upper parameter bounds.

Value

A list containing estimated posterior mean, standard error, effective sample size (ESS), and importance weights.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
```

```
my_surf <- function(x, par) 1 - pexp(x, rate = par[1])
flat_prior <- function(par) ifelse(par[1] > 0, 0, -Inf)
R_plan <- c(rep(1, 8), rep(0, 4))
dat <- r_adapt2_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)$data
is_res <- is_adapt2_phcs(
  data = dat, n = 20, m = 12,
  T_thresh = 2.5, R = R_plan,
  pdf = my_pdf, cdf = my_cdf,
  surf = my_surf, prior = flat_prior,
  start = c(rate = 0.5), N_samples = 1000
)
print(is_res$summary)
```

`lindley_adapt1_phcs` *Bayesian Estimation via Lindley's Approximation under Adaptive Type-I PHCS*

Description

Computes Bayes estimates of parameters using Lindley's expansion for Adaptive Type-I PHCS.

Usage

```
lindley_adapt1_phcs(
  data,
  n,
  m,
  T_thresh,
  R,
  pdf,
  cdf = NULL,
  surf = NULL,
  prior = NULL,
  start,
  lower = NULL,
  upper = NULL
)
```

Arguments

`data`	Numeric vector of observed failure times.
`n`	Integer, total initial sample size placed on test.
`m`	Integer, pre-fixed target number of failures.
`T_thresh`	Numeric, pre-fixed time threshold $T > 0$.

R	Numeric vector of length m, progressive censoring scheme.
pdf	Function, user-supplied probability density function pdf(x, par).
cdf	Function (optional), user-supplied cumulative distribution function cdf(x, par).
surf	Function (optional), user-supplied survival function surf(x, par).
prior	Function (optional), user-supplied prior density function log_prior(par). Default is flat prior.
start	Numeric vector, initial parameter guess for finding MLE.
lower	Numeric vector (optional), lower bounds for parameters.
upper	Numeric vector (optional), upper bounds for parameters.

Value

A list containing Lindley Bayes estimates, MLEs, and log-likelihood details.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
my_surf <- function(x, par) 1 - pexp(x, rate = par[1])
flat_prior <- function(par) ifelse(par[1] > 0, 0, -Inf)
R_plan <- c(rep(1, 8), rep(0, 4))
dat <- r_adapt1_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)$data
lindley_res <- lindley_adapt1_phcs(
  data = dat, n = 20, m = 12,
  T_thresh = 2.5, R = R_plan,
  pdf = my_pdf, cdf = my_cdf,
  surf = my_surf, prior = flat_prior,
  start = c(rate = 0.8)
)
print(lindley_res$estimates)
```

---

lindley_adapt2_phcs *Bayesian Estimation via Lindley's Approximation under Adaptive Type-II PHCS*

Description

Computes Bayes estimates of parameters using Lindley's expansion for Adaptive Type-II PHCS.

Usage

```
lindley_adapt2_phcs(
  data,
  n,
  m,
  T_thresh,
  R,
  pdf,
  cdf = NULL,
  surf = NULL,
  prior = NULL,
  start,
  lower = NULL,
  upper = NULL
)
```

Arguments

data	Numeric vector of m observed failure times.
n	Integer, total initial sample size placed on test.
m	Integer, pre-fixed target number of failures.
T_thresh	Numeric, pre-fixed time threshold T > 0.
R	Numeric vector of length m, progressive censoring scheme.
pdf	Function, user-supplied probability density function pdf(x, par).
cdf	Function (optional), user-supplied cumulative distribution function cdf(x, par).
surf	Function (optional), user-supplied survival function surf(x, par).
prior	Function (optional), user-supplied prior density function log_prior(par). Default is flat prior.
start	Numeric vector, initial parameter guess for finding MLE.
lower	Numeric vector (optional), lower bounds for parameters.
upper	Numeric vector (optional), upper bounds for parameters.

Value

A list containing Lindley Bayes estimates, MLEs, and log-likelihood details.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
my_surf <- function(x, par) 1 - pexp(x, rate = par[1])
flat_prior <- function(par) ifelse(par[1] > 0, 0, -Inf)
R_plan <- c(rep(1, 8), rep(0, 4))
dat <- r_adapt2_phcs(
  n = 20, m = 12, T_thresh = 2.5,
```

```
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)$data
lindley_res <- lindley_adapt2_phcs(
  data = dat, n = 20, m = 12,
  T_thresh = 2.5, R = R_plan,
  pdf = my_pdf, cdf = my_cdf,
  surf = my_surf, prior = flat_prior,
  start = c(rate = 0.8)
)
print(lindley_res$estimates)
```

`mle_adapt1_phcs` *Maximum Likelihood Estimation under Adaptive Type-I Progressive Hybrid Censoring*

Description

Computes Maximum Likelihood Estimates (MLE) under the Adaptive Type-I Progressive Hybrid Censoring Scheme using various optimization algorithms ("NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", "NM").

Usage

```
mle_adapt1_phcs(
  data,
  n,
  m,
  T_thresh,
  R,
  pdf,
  cdf = NULL,
  surf = NULL,
  start,
  method = "NR",
  lower = NULL,
  upper = NULL,
  max_iter = 500,
  ...
)

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

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

## S3 method for class 'summary.mle_adapt_fit'
```

```
print(x, ...)

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

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

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

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

Arguments

data	Numeric vector of observed failure times.
n	Integer, total initial sample size placed on test.
m	Integer, pre-fixed target number of failures.
T_thresh	Numeric, pre-fixed time threshold T > 0.
R	Numeric vector of length m, progressive censoring scheme.
pdf	Function, user-supplied probability density function pdf(x, par).
cdf	Function (optional), user-supplied cumulative distribution function cdf(x, par).
surf	Function (optional), user-supplied survival function surf(x, par).
start	Numeric vector, initial parameter guesses.
method	Optimization method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM" (case-insensitive, default "NR").
lower	Numeric vector (optional), lower bounds for parameters.
upper	Numeric vector (optional), upper bounds for parameters.
max_iter	Maximum iterations for optimization algorithm (default 500).
...	Additional arguments.
x	An object of class mle_adapt_fit.
object	An object of class mle_adapt_fit.

Value

An S3 object of class mle_adapt_fit containing estimates, standard errors, log-likelihood, Hessian, covariance matrix, and model info.

Methods return printed summary, coefficients, variance-covariance matrix, log-likelihood, standard errors, or AIC.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
my_surf <- function(x, par) 1 - pexp(x, rate = par[1])
R_plan <- c(rep(1, 8), rep(0, 4))
dat <- r_adapt1_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)$data
fit <- mle_adapt1_phcs(
  data = dat, n = 20, m = 12,
  T_thresh = 2.5, R = R_plan,
  pdf = my_pdf, cdf = my_cdf,
  surf = my_surf, start = c(rate = 1.0),
  method = "BFGS"
)
summary(fit)
```

`mle_adapt2_phcs` *Maximum Likelihood Estimation under Adaptive Type-II Progressive Hybrid Censoring*

Description

Computes Maximum Likelihood Estimates (MLE) under the Adaptive Type-II Progressive Hybrid Censoring Scheme using various optimization algorithms ("NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", "NM").

Usage

```
mle_adapt2_phcs(
  data,
  n,
  m,
  T_thresh,
  R,
  pdf,
  cdf = NULL,
  surf = NULL,
  start,
  method = "NR",
  lower = NULL,
  upper = NULL,
  max_iter = 500,
  ...
)
```

Arguments

data	Numeric vector of m observed failure times.
n	Integer, total initial sample size placed on test.
m	Integer, pre-fixed target number of failures.
T_thresh	Numeric, pre-fixed time threshold T > 0.
R	Numeric vector of length m, progressive censoring scheme.
pdf	Function, user-supplied probability density function pdf(x, par).
cdf	Function (optional), user-supplied cumulative distribution function cdf(x, par).
surf	Function (optional), user-supplied survival function surf(x, par).
start	Numeric vector, initial parameter guesses.
method	Optimization method: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM" (case-insensitive, default "NR").
lower	Numeric vector (optional), lower bounds for parameters.
upper	Numeric vector (optional), upper bounds for parameters.
max_iter	Maximum iterations for optimization algorithm (default 500).
...	Additional arguments passed to optimization routines.

Value

An S3 object of class `mle_adapt_fit` containing estimates, standard errors, log-likelihood, Hessian, covariance matrix, and model info.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
my_surf <- function(x, par) 1 - pexp(x, rate = par[1])
R_plan <- c(rep(1, 8), rep(0, 4))
dat <- r_adapt2_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)$data
fit <- mle_adapt2_phcs(
  data = dat, n = 20, m = 12,
  T_thresh = 2.5, R = R_plan,
  pdf = my_pdf, cdf = my_cdf,
  surf = my_surf, start = c(rate = 1.0),
  method = "BFGS"
)
summary(fit)
```

`plot_adapt1_phcs` *Diagnostic Plots for Adaptive Type-I Progressive Hybrid Censoring Scheme*

Description

Generates 3-panel diagnostic graphics (Histogram with density overlay, Dot plot, and ACF plot) under Adaptive Type-I PHCS.

Usage

```
plot_adapt1_phcs(
  input,
  T_thresh = NULL,
  pdf = NULL,
  par = NULL,
  main = "Adaptive Type-I PHCS Diagnostics",
  ...
)
```

Arguments

`input`	Numeric vector of failure times, dataset list, or fit object.
`T_thresh`	Time threshold T. If NULL, extracted from input list if available.
`pdf`	Function (optional), probability density function pdf(x, par) for density overlay.
`par`	Numeric vector (optional), parameters for PDF overlay.
`main`	Title prefix for plots.
`...`	Additional graphic parameters.

Value

Invisibly returns NULL. Displays diagnostic plots.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
R_plan <- c(rep(1, 8), rep(0, 4))
dat_gen <- r_adapt1_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)
plot_adapt1_phcs(
  dat_gen, pdf = my_pdf, par = 0.5
)
```

`plot_adapt2_phcs` *Diagnostic Plots for Adaptive Type-II Progressive Hybrid Censoring Scheme*

Description

Generates 3-panel diagnostic graphics (Histogram with density overlay, Dot plot, and ACF plot) under Adaptive Type-II PHCS.

Usage

```
plot_adapt2_phcs(
  input,
  T_thresh = NULL,
  pdf = NULL,
  par = NULL,
  main = "Adaptive Type-II PHCS Diagnostics",
  ...
)
```

Arguments

`input`	Numeric vector of failure times, dataset list, or fit object.
`T_thresh`	Time threshold T. If NULL, extracted from input list if available.
`pdf`	Function (optional), probability density function pdf(x, par) for density overlay.
`par`	Numeric vector (optional), parameters for PDF overlay.
`main`	Title prefix for plots.
`...`	Additional graphic parameters.

Value

Invisibly returns NULL. Displays diagnostic plots.

Examples

```
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
R_plan <- c(rep(1, 8), rep(0, 4))
dat_gen <- r_adapt2_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, par = 0.5
)
plot_adapt2_phcs(
  dat_gen, pdf = my_pdf, par = 0.5
)
```

`r_adapt1_phcs` *Random Data Generation under Adaptive Type-I Progressive Hybrid Censoring Scheme*

Description

Generates random censored lifetimes under the Adaptive Type-I Progressive Hybrid Censoring Scheme as described by Lin and Huang (2012).

Usage

```
r_adapt1_phcs(
  n,
  m,
  T_thresh,
  R,
  pdf = NULL,
  cdf = NULL,
  qdf = NULL,
  par = NULL,
  seed = NULL,
  lower = 1e-07,
  upper = 1e+05
)
```

Arguments

`n`	Integer, total number of units placed on test.
`m`	Integer, pre-fixed target number of failures (1 <= m <= n).
`T_thresh`	Numeric, pre-fixed time threshold parameter T > 0.
`R`	Numeric vector of length m, progressive censoring scheme satisfying sum(R) + m = n.
`pdf`	Function, user-supplied probability density function pdf(x, par).
`cdf`	Function, user-supplied cumulative distribution function cdf(x, par).
`qdf`	Function (optional), user-supplied quantile function qdf(p, par).
`par`	Numeric vector, true parameter values for the lifetime distribution.
`seed`	Optional integer, seed for random number generation.
`lower`	Numeric, lower bound for numerical root finding (default 1e-7).
`upper`	Numeric, upper bound for numerical root finding (default 1e5).

Value

A list containing:

`data`	Numeric vector of observed failure times up to test termination.
`censored_at_T`	Logical indicator or count of units censored at time T.
`T_thresh`	The time threshold T.
`n`	Total sample size.
`m`	Target number of failures.
`R_effective`	Effective progressive censoring plan applied.
`d`	Total number of failures observed.
`scheme`	String descriptor of the censoring scheme.

References

Lin, C. T., & Huang, Y. L. (2012). On adaptive Type-I progressive hybrid censored samples. *Journal of Statistical Computation and Simulation*, 82(7), 1005-1018.

Balakrishnan, N., Cramer, E., & Kundu, D. (2023). *Hybrid censoring know-how: Designs and implementations*. Academic Press.

Examples

```
# Example: Standard Exponential distribution
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
my_qdf <- function(p, par) qexp(p, rate = par[1])
R_plan <- c(rep(1, 8), rep(0, 4))
res <- r_adapt1_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, qdf = my_qdf, par = 0.5
)
print(res$data)
```

---

r_adapt2_phcs *Random Data Generation under Adaptive Type-II Progressive Hybrid Censoring Scheme*

Description

Generates random censored lifetimes under the Adaptive Type-II Progressive Hybrid Censoring Scheme as described by Ng, Kundu, and Chan (2009).

Usage

```
r_adapt2_phcs(
  n,
  m,
  T_thresh,
  R,
  pdf = NULL,
  cdf = NULL,
  qdf = NULL,
  par = NULL,
  seed = NULL,
  lower = 1e-07,
  upper = 1e+05
)
```

Arguments

n	Integer, total number of units placed on test.
m	Integer, pre-fixed target number of failures (1 <= m <= n).
T_thresh	Numeric, pre-fixed time threshold parameter T > 0.
R	Numeric vector of length m, progressive censoring scheme satisfying sum(R) + m = n.
pdf	Function, user-supplied probability density function pdf(x, par).
cdf	Function, user-supplied cumulative distribution function cdf(x, par).
qdf	Function (optional), user-supplied quantile function qdf(p, par).
par	Numeric vector, true parameter values for the lifetime distribution.
seed	Optional integer, seed for random number generation.
lower	Numeric, lower bound for numerical root finding (default 1e-7).
upper	Numeric, upper bound for numerical root finding (default 1e5).

Value

A list containing:

data	Numeric vector of m observed failure times.
T_thresh	The time threshold T.
n	Total sample size.
m	Target number of failures.
R_effective	Effective progressive censoring plan applied.
d	Number of failures observed at or before time T.
scheme	String descriptor of the censoring scheme.

References

Ng, H. K. T., Kundu, D., & Chan, P. S. (2009). Statistical inference for a progressive line-testing experiment with adaptive progressive Type-II censoring. *IEEE Transactions on Reliability*, 58(4), 634-642.

Balakrishnan, N., Cramer, E., & Kundu, D. (2023). *Hybrid censoring know-how: Designs and implementations*. Academic Press.

Examples

```
# Example: Standard Exponential distribution
set.seed(123)
my_cdf <- function(x, par) pexp(x, rate = par[1])
my_pdf <- function(x, par) dexp(x, rate = par[1])
my_qdf <- function(p, par) qexp(p, rate = par[1])
R_plan <- c(rep(1, 8), rep(0, 4))
res <- r_adapt2_phcs(
  n = 20, m = 12, T_thresh = 2.5,
  R = R_plan, pdf = my_pdf,
  cdf = my_cdf, qdf = my_qdf, par = 0.5
)
print(res$data)
```

---

stdEr *Generic Standard Error Function*

Description

Generic Standard Error Function

Usage

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

Arguments

`object`	An object of class `mle_adapt_fit`.
`...`	Additional arguments.

Value

A numeric vector of standard errors.

Index