Package: MleCensoR (via r-universe)

July 23, 2026

Type Package

Title Maximum Likelihood Estimation under Censoring Schemes

Version 0.1.0

Description Provides generalized functions to compute Maximum Likelihood Estimation (MLE) for any univariate distribution under various censoring and truncation schemes. Users supply the probability density function (PDF), cumulative distribution function (CDF), survival function, support bounds, and initial parameter values; the package constructs and maximizes the appropriate log-likelihood automatically. Supported schemes include right and left truncation, random, right, left, interval, and middle censoring, block random censoring, balanced joint progressive Type-II (BJPT-II), progressive first failure, joint Type-I, Type-I, Type-II, progressive Type-II, Type-II progressively hybrid, joint Type-II, hybrid, hybrid Type-I, doubly Type-II, Type-I hybrid, and hybrid Type-II censoring. Optimization methods include Newton-Raphson (NR), Broyden-Fletcher-Goldfarb-Shanno (BFGS), the BFGS algorithm implemented in R (BFGSR), Berndt-Hall-Hall-Hausman (BHHH), Simulated Annealing (SANN), Conjugate Gradients (CG), and Nelder-Mead (NM). Inference summaries provide the Akaike Information Criterion (AIC), estimated coefficients, log-likelihood, iteration count, standard errors, z-values, p-values, and the variance-covariance matrix. Methods are described in Nagar, Kumar, and Krishna (2026) <doi:10.59467/IJASS.2026.22.1>, Goel, Kumar, and Krishna (2026, ``Estimation in power Lindley distributions using balanced joint progressively Type-II censored data''), Wu and Kus (2009) <doi:10.1016/j.csda.2009.03.010>, Goel and Krishna (2026) <doi:10.1007/s13198-026-03208-w>, Balakrishnan and Aggarwala (2000, ISBN:978-1-4612-1334-5), Mondal and Kundu (2020) <doi:10.1080/03610926.2018.1554128>, Ding and Gui (2023) <doi:10.3390/math11092003>, Prajapati, Mitra, and Kundu (2019) <doi:10.1007/s13571-018-0167-0>, Yadav, Jaiswal, and Yadav (2026) <doi:10.1007/s11135-026-02647-8>, Iyer, Jammalamadaka,

and Kundu (2008) <doi:10.1016/j.jspi.2007.03.062>, Banerjee and Kundu (2008) <doi:10.1109/TR.2008.916890>, Kundu and Joarder (2006) <doi:10.1016/j.csda.2005.05.002>, Berndt, Hall, Hall, and Hausman (1974) ``Estimation and Inference in Nonlinear Structural Models'' <doi:10.3386/t0003>, Fletcher (1987, ``Practical Methods of Optimization'', ISBN:978-0-471-91547-8), Nelder and Mead (1965) <doi:10.1093/comjnl/7.4.308>, McKinnon (1999) ``Convergence of the Nelder-Mead simplex method to a non-stationary point'' <doi:10.1137/S1052623496303482>, Kirkpatrick, Gelatt, and Vecchi (1983) <doi:10.1126/science.220.4598.671>, Fletcher and Reeves (1964) <doi:10.1093/comjnl/7.2.149>, and Nocedal and Wright (2006, ``Numerical Optimization'', ISBN:978-0-387-30303-1).

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

Imports stats

Suggests testthat (>= 3.0.0)

NeedsCompilation no

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

Maintainer Shikhar Tyagi <shikhar1093tyagi@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-07-23 14:10:09 UTC

RemoteUrl https://github.com/cran/MleCensoR

RemoteRef HEAD

RemoteSha 0b9e7482a97fbb7be529bcbb0468d77ab2c2dc2a

Contents

`AIC.mle_fit` *Extract Akaike Information Criterion (AIC)*

Description

Extract Akaike Information Criterion (AIC)

Usage

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

Arguments

`object`	an object of class ‘mle_fit‘
`...`	further arguments
`k`	numeric, penalty per parameter (default is 2)

Value

numeric value of AIC

coef.mle_fit *Extract Parameter Estimates*

Description

Extract Parameter Estimates

Usage

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

Arguments

object	an object of class ‘mle_fit‘
...	further arguments (currently ignored)

Value

numeric vector of estimated parameters

logLik.mle_fit *Extract Log-Likelihood*

Description

Extract Log-Likelihood

Usage

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

Arguments

object	an object of class ‘mle_fit‘
...	further arguments (currently ignored)

Value

an object of class ‘logLik‘ with attributes "df" and "nobs"

mle_bjpt2 *MLE under Balanced Joint Progressive Type-II (BJPT-II) Censoring*

Description

MLE under Balanced Joint Progressive Type-II (BJPT-II) Censoring

Usage

```
mle_bjpt2(
  w,
  z,
  D_A,
  D_B,
  pdf_A = NULL,
  cdf_A = NULL,
  surv_A = NULL,
  pdf_B = NULL,
  cdf_B = NULL,
  surv_B = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

w	numeric vector of failure times (ordered).
z	numeric vector indicating group membership (1 for population A, 0 for population B).
D_A	numeric vector of removals from population A at each failure time.
D_B	numeric vector of removals from population B at each failure time.
pdf_A	density function of population A.
cdf_A	cumulative distribution function of population A.
surv_A	survival function of population A (optional).
pdf_B	density function of population B (optional, defaults to pdf_A).
cdf_B	cumulative distribution function of population B (optional, defaults to cdf_A).
surv_B	survival function of population B (optional).
start	numeric vector of initial parameter values.

method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_block_random` *MLE under Block Random Censoring*

Description

MLE under Block Random Censoring

Usage

```
mle_block_random(
  x,
  status,
  block,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

x	numeric vector of observed values.
status	numeric vector indicating censoring status (1 = failure, 0 = censored).
block	vector indicating block/group membership for each observation.
pdf	density function of the distribution, potentially accepting a ‘block‘ argument.

cdf	cumulative distribution function of the distribution, potentially accepting a ‘block‘ argument.
surv	survival function of the distribution (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_doubly_type2` *MLE under Doubly Type-II Censoring*

Description

MLE under Doubly Type-II Censoring

Usage

```
mle_doubly_type2(
  x,
  r,
  s,
  n,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

x	numeric vector of observed failure times (ordered).
r	integer, index of the first observed failure (1-based).
s	integer, index of the last observed failure (1-based).
n	integer, total initial units.
pdf	density function of the distribution.
cdf	cumulative distribution function of the distribution.
surv	survival function of the distribution (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

---

mle_hybrid *MLE under Hybrid Censoring (Type-I Hybrid)*

Description

MLE under Hybrid Censoring (Type-I Hybrid)

Usage

```
mle_hybrid(
  x,
  r,
  tc,
  n,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
```

```
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

x	numeric vector of observed failure times (ordered).
r	integer, target number of failures.
tc	numeric, fixed censoring time.
n	integer, total initial units.
pdf	density function of the distribution.
cdf	cumulative distribution function of the distribution.
surv	survival function of the distribution (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

---

mle_hybrid_type1 *MLE under Hybrid Type-I Censoring*

Description

Alias for mle_hybrid. The experiment terminates at $T^* = \min(x_{(r)}, T_c)$.

Usage

```
mle_hybrid_type1(
  x,
  r,
  tc,
  n,
  pdf = NULL,
```

```
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

x	numeric vector of observed failure times (ordered).
r	integer, target number of failures.
tc	numeric, fixed censoring time.
n	integer, total initial units.
pdf	density function of the distribution.
cdf	cumulative distribution function of the distribution.
surv	survival function of the distribution (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

---

mle_hybrid_type2 *MLE under Hybrid Type-II Censoring*

Description

MLE under Hybrid Type-II Censoring

Usage

```
mle_hybrid_type2(
  x,
  r,
  tc,
  n,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

x	numeric vector of observed failure times (ordered).
r	integer, target number of failures.
tc	numeric, fixed censoring time.
n	integer, total initial units.
pdf	density function of the distribution.
cdf	cumulative distribution function of the distribution.
surv	survival function of the distribution (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_interval` *MLE under Interval Censoring*

Description

MLE under Interval Censoring

Usage

```
mle_interval(
  l,
  r,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`l`	numeric vector of lower bounds of the censoring intervals.
`r`	numeric vector of upper bounds of the censoring intervals.
`pdf`	density function of the distribution.
`cdf`	cumulative distribution function of the distribution.
`surv`	survival function of the distribution (optional).
`start`	numeric vector of initial parameter values.
`method`	character string, optimization method.
`constraints`	list, optimization constraints.
`grad`	gradient function.
`hess`	Hessian function.
`param_range`	list, parameter ranges.
`logLik`	custom log-likelihood function.
`...`	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_joint_type1` *MLE under Joint Type-I Censoring*

Description

MLE under Joint Type-I Censoring

Usage

```
mle_joint_type1(
  x,
  z,
  tc,
  n_A,
  n_B,
  pdf_A = NULL,
  cdf_A = NULL,
  surv_A = NULL,
  pdf_B = NULL,
  cdf_B = NULL,
  surv_B = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`x`	numeric vector of observed failure times (<= tc) for both groups.
`z`	numeric vector indicating group membership (1 for population A, 0 for population B).
`tc`	numeric, fixed censoring time.
`n_A`	integer, total number of initial units in population A.
`n_B`	integer, total number of initial units in population B.
`pdf_A`	density function of population A.
`cdf_A`	cumulative distribution function of population A.
`surv_A`	survival function of population A (optional).
`pdf_B`	density function of population B (optional, defaults to pdf_A).
`cdf_B`	cumulative distribution function of population B (optional, defaults to cdf_A).

surv_B	survival function of population B (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_joint_type2` *MLE under Joint Type-II Censoring*

Description

MLE under Joint Type-II Censoring

Usage

```
mle_joint_type2(
  w,
  z,
  n_A,
  n_B,
  pdf_A = NULL,
  cdf_A = NULL,
  surv_A = NULL,
  pdf_B = NULL,
  cdf_B = NULL,
  surv_B = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

w	numeric vector of observed failure times (ordered).
z	numeric vector indicating group membership (1 for population A, 0 for population B).
n_A	integer, total initial units in population A.
n_B	integer, total initial units in population B.
pdf_A	density function of population A.
cdf_A	cumulative distribution function of population A.
surv_A	survival function of population A (optional).
pdf_B	density function of population B (optional, defaults to pdf_A).
cdf_B	cumulative distribution function of population B (optional, defaults to cdf_A).
surv_B	survival function of population B (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

---

mle_left *MLE under Left Censoring*

Description

MLE under Left Censoring

Usage

```
mle_left(
  x,
  status,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
```

```
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

x	numeric vector of observed values.
status	numeric vector indicating censoring status (1 = failure, 0 = censored).
pdf	density function of the distribution.
cdf	cumulative distribution function of the distribution.
surv	survival function of the distribution (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

---

mle_left_truncation *MLE under Left Truncation*

Description

MLE under Left Truncation

Usage

```
mle_left_truncation(
  x,
  tl,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
```

```
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

x	numeric vector of observed values.
tl	numeric vector or scalar of left truncation limits.
pdf	density function of the distribution.
cdf	cumulative distribution function of the distribution.
surv	survival function of the distribution (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

---

mle_middle *MLE under Middle Censoring*

Description

MLE under Middle Censoring

Usage

```
mle_middle(
  x,
  u,
  v,
  status,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

x	numeric vector of observed times (or NA if censored).
u	numeric vector of lower bounds of the censoring intervals.
v	numeric vector of upper bounds of the censoring intervals.
status	numeric vector indicating censoring status (1 = exact, 0 = censored).
pdf	density function of the distribution.
cdf	cumulative distribution function of the distribution.
surv	survival function of the distribution (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.
grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_progressive_first_failure` *MLE under Progressive First Failure Censoring*

Description

MLE under Progressive First Failure Censoring

Usage

```
mle_progressive_first_failure(
  x,
  r_removals,
  k_group_size,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`x`	numeric vector of observed first-failure times (ordered).
`r_removals`	numeric vector of group removals at each failure time.
`k_group_size`	integer, number of units per group.
`pdf`	density function of the distribution.
`cdf`	cumulative distribution function of the distribution.
`surv`	survival function of the distribution (optional).
`start`	numeric vector of initial parameter values.
`method`	character string, optimization method.
`constraints`	list, optimization constraints.
`grad`	gradient function.
`hess`	Hessian function.
`param_range`	list, parameter ranges.
`logLik`	custom log-likelihood function.
`...`	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_progressive_hybrid_type2` *MLE under Type-II Progressively Hybrid Censoring*

Description

Computes the MLE for any univariate distribution under the Type-II progressively hybrid censoring scheme (Kundu & Joarder, 2006). The experiment terminates at $T^* = \min(x_{(m)}, T_c)$, where m is the planned number of failures and T_c is a pre-specified time limit.

Usage

```
mle_progressive_hybrid_type2(
  x,
  r_removals,
  tc,
  n,
  m,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`x`	numeric vector of observed failure times (ordered), only failures observed before T^*.
`r_removals`	numeric vector of the *full* planned progressive removal scheme $(R_1, R_2, \ldots, R_m)$.
`tc`	numeric, pre-specified time limit T_c.
`n`	integer, total number of units initially placed on test.
`m`	integer, planned total number of failures (length of the full removal scheme).
`pdf`	density function of the distribution, accepting `(x, theta, ...)`.
`cdf`	cumulative distribution function of the distribution, accepting `(x, theta, ...)`.

`surv`	survival function of the distribution (optional). If not provided, computed as `1 - cdf`.
`start`	numeric vector of initial parameter values.
`method`	character string, optimization method.
`constraints`	list, optimization constraints.
`grad`	gradient function.
`hess`	Hessian function.
`param_range`	list, parameter ranges.
`logLik`	custom log-likelihood function.
`...`	further arguments passed to optimization or user-defined functions.

Details

The likelihood depends on whether the m-th failure occurs before or after T_c:

Case I ($D = m$, i.e., all m failures observed before T_c):

$$L(\theta) \propto \prod_{i=1}^{m} f(x_i;\theta) \cdot S(x_i;\theta)^{R_i}$$

Case II ($D = J < m$, i.e., only J failures observed before T_c):

$$L(\theta) \propto \left[\prod_{i=1}^{J} f(x_i;\theta) \cdot S(x_i;\theta)^{R_i}\right] \cdot S(T_c;\theta)^{R_J^*}$$

where $R_J^* = n - J - \sum_{i=1}^{J} R_i$.

Value

An object of class `mle_fit` containing the optimization results.

References

Kundu, D., & Joarder, A. (2006). Analysis of Type-II progressively hybrid censored data. *Computational Statistics & Data Analysis*, 50(10), 2509-2528.

Examples

```
# Exponential distribution under Type-II progressively hybrid censoring
pdf_exp <- function(x, theta) dexp(x, rate = theta[1])
cdf_exp <- function(x, theta) pexp(x, rate = theta[1])

# Suppose n = 20, m = 10, tc = 2.0
# 7 failures observed before tc: Case II
x <- c(0.12, 0.35, 0.62, 0.89, 1.15, 1.48, 1.82)
r_removals <- c(1, 0, 1, 0, 1, 0, 0, 1, 0, 0)  # full plan for m = 10

fit <- mle_progressive_hybrid_type2(
```

```
  x = x, r_removals = r_removals, tc = 2.0, n = 20, m = 10,
  pdf = pdf_exp, cdf = cdf_exp,
  start = c(rate = 1.0)
)
summary(fit)
```

`mle_progressive_type2` *MLE under Progressive Type-II Censoring*

Description

MLE under Progressive Type-II Censoring

Usage

```
mle_progressive_type2(
  x,
  r_removals,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`x`	numeric vector of observed failure times (ordered).
`r_removals`	numeric vector of removals at each failure time.
`pdf`	density function of the distribution.
`cdf`	cumulative distribution function of the distribution.
`surv`	survival function of the distribution (optional).
`start`	numeric vector of initial parameter values.
`method`	character string, optimization method.
`constraints`	list, optimization constraints.
`grad`	gradient function.
`hess`	Hessian function.
`param_range`	list, parameter ranges.
`logLik`	custom log-likelihood function.
`...`	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_random` *MLE under Random Censoring*

Description

MLE under Random Censoring

Usage

```
mle_random(
  x,
  status,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`x`	numeric vector of observed values.
`status`	numeric vector indicating censoring status (1 = failure, 0 = censored).
`pdf`	density function of the distribution.
`cdf`	cumulative distribution function of the distribution.
`surv`	survival function of the distribution (optional).
`start`	numeric vector of initial parameter values.
`method`	character string, optimization method.
`constraints`	list, optimization constraints.
`grad`	gradient function.
`hess`	Hessian function.
`param_range`	list, parameter ranges.
`logLik`	custom log-likelihood function.
`...`	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_right` *MLE under Right Censoring*

Description

MLE under Right Censoring

Usage

```
mle_right(
  x,
  status,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`x`	numeric vector of observed values.
`status`	numeric vector indicating censoring status (1 = failure, 0 = censored).
`pdf`	density function of the distribution.
`cdf`	cumulative distribution function of the distribution.
`surv`	survival function of the distribution (optional).
`start`	numeric vector of initial parameter values.
`method`	character string, optimization method.
`constraints`	list, optimization constraints.
`grad`	gradient function.
`hess`	Hessian function.
`param_range`	list, parameter ranges.
`logLik`	custom log-likelihood function.
`...`	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_right_truncation` *MLE under Right Truncation*

Description

MLE under Right Truncation

Usage

```
mle_right_truncation(
  x,
  tr,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`x`	numeric vector of observed values.
`tr`	numeric vector or scalar of right truncation limits.
`pdf`	density function of the distribution, accepting ‘(x, theta, ...)‘.
`cdf`	cumulative distribution function of the distribution, accepting ‘(x, theta, ...)‘.
`surv`	survival function of the distribution (optional).
`start`	numeric vector of initial parameter values.
`method`	character string, optimization method.
`constraints`	list, optimization constraints.
`grad`	gradient function.
`hess`	Hessian function.
`param_range`	list, parameter ranges.
`logLik`	custom log-likelihood function.
`...`	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_type1` *MLE under Type-I Censoring*

Description

MLE under Type-I Censoring

Usage

```
mle_type1(
  x,
  tc,
  status = NULL,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`x`	numeric vector of observed values.
`tc`	numeric, fixed censoring time.
`status`	numeric vector indicating censoring status (optional, determined as x < tc if NULL).
`pdf`	density function of the distribution.
`cdf`	cumulative distribution function of the distribution.
`surv`	survival function of the distribution (optional).
`start`	numeric vector of initial parameter values.
`method`	character string, optimization method.
`constraints`	list, optimization constraints.
`grad`	gradient function.
`hess`	Hessian function.
`param_range`	list, parameter ranges.
`logLik`	custom log-likelihood function.
`...`	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_type1_hybrid` *MLE under Type-I Hybrid Censoring*

Description

Alias for `mle_hybrid`. The experiment terminates at $T^* = \min(x_{(r)}, T_c)$.

Usage

```
mle_type1_hybrid(
  x,
  r,
  tc,
  n,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

`x`	numeric vector of observed failure times (ordered).
`r`	integer, target number of failures.
`tc`	numeric, fixed censoring time.
`n`	integer, total initial units.
`pdf`	density function of the distribution.
`cdf`	cumulative distribution function of the distribution.
`surv`	survival function of the distribution (optional).
`start`	numeric vector of initial parameter values.
`method`	character string, optimization method.
`constraints`	list, optimization constraints.
`grad`	gradient function.

hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`mle_type2` *MLE under Type-II Censoring*

Description

MLE under Type-II Censoring

Usage

```
mle_type2(
  x,
  n,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  start,
  method = NULL,
  constraints = NULL,
  grad = NULL,
  hess = NULL,
  param_range = NULL,
  logLik = NULL,
  ...
)
```

Arguments

x	numeric vector of observed failure times (ordered).
n	integer, total number of units in the experiment.
pdf	density function of the distribution.
cdf	cumulative distribution function of the distribution.
surv	survival function of the distribution (optional).
start	numeric vector of initial parameter values.
method	character string, optimization method.
constraints	list, optimization constraints.

grad	gradient function.
hess	Hessian function.
param_range	list, parameter ranges.
logLik	custom log-likelihood function.
...	further arguments passed to optimization or user-defined functions.

Value

An object of class ‘mle_fit‘ containing the optimization results.

`nIter` *Extract Number of Iterations*

Description

Extract Number of Iterations

Usage

```
nIter(x, ...)
```

Arguments

x	an object of class ‘mle_fit‘
...	further arguments (currently ignored)

Value

integer representing number of iterations

`print.mle_fit` *Print MLE Fit*

Description

Print MLE Fit

Usage

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

Arguments

x	an object of class ‘mle_fit‘
...	further arguments (currently ignored)

Value

The input object x is returned invisibly. Called for its side effect of printing estimated parameters and the log-likelihood value to the console.

---

print.summary.mle_fit *Print Summary Table*

Description

Print Summary Table

Usage

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

Arguments

x	an object of class ‘summary.mle_fit‘
...	further arguments (currently ignored)

Value

The input object x is returned invisibly. Called for its side effect of printing a formatted summary table to the console.

---

stdEr *Extract Standard Errors*

Description

Extract Standard Errors

Usage

```
stdEr(x, ...)
```

Arguments

x	an object of class ‘mle_fit‘
...	further arguments (currently ignored)

Value

numeric vector of standard errors

summary.mle_fit *Summarize MLE Fit*

Description

Summarize MLE Fit

Usage

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

Arguments

object	an object of class ‘mle_fit‘
...	further arguments (currently ignored)

Value

summary.mle_fit object

vcov.mle_fit *Extract Variance-Covariance Matrix*

Description

Extract Variance-Covariance Matrix

Usage

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

Arguments

object	an object of class ‘mle_fit‘
...	further arguments (currently ignored)

Value

variance-covariance matrix

Index