Package 'CompRiskRel'

August 5, 2026

Type Package

Title Reliability and Competing Risks Analysis under Hybrid Censoring

Version 0.1.0

Description Generalized computational algorithms for competing risks analysis, stress-strength reliability modeling, optimal designs, and reliability acceptance sampling plans under various hybrid censoring schemes. Includes data generation routines, Maximum Likelihood Estimation (MLE) with seven optimization algorithms ('Newton-Raphson', 'BFGS', 'BFGSR', 'BHHH', 'SANN', 'CG', and 'Nelder-Mead'), Bayesian inference via Gibbs sampling and Metropolis-Hastings MCMC, Importance Sampling, and 'Lindley' asymptotic approximation. Visualization functions generate histograms, dot plots, and autocorrelation plots for model validation. Methodology and design principles are based on 'Balakrishnan', 'Cramer', and 'Kundu' (2023, ``Hybrid Censoring Know-How: Designs and Implementations'', Academic Press, ISBN:978-0-12-398387-9).

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

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 2026-08-05 06:30:11 UTC

Contents

`bayesian_sampling_plan_exp` *Bayesian Reliability Acceptance Sampling Plan for Exponential lifetimes (10.5.3)*

Description

Bayesian Reliability Acceptance Sampling Plan for Exponential lifetimes (10.5.3)

Usage

```
bayesian_sampling_plan_exp(
  n,
  T_star,
  r,
  prior_shape = 2,
  prior_rate = 1,
  c_accept = 1.2
)
```

Arguments

n	Sample size.
T_star	Termination time limit.
r	Failure count limit.
prior_shape	Shape parameter of Inverse-Gamma prior.
prior_rate	Rate parameter of Inverse-Gamma prior.
c_accept	Acceptance threshold constant.

Value

S3 object of class `sampling_plan` with posterior risk and decision criteria.

Examples

```
bayesian_sampling_plan_exp(
  n = 20, T_star = 1.0, r = 10,
  prior_shape = 2, prior_rate = 1, c_accept = 1.2
)
```

---

bayes_gibbs_censored *Bayesian Inference via Gibbs Sampling for Censored Data Models*

Description

Bayesian Inference via Gibbs Sampling for Censored Data Models

Usage

```
bayes_gibbs_censored(log_posterior, init_par, n_sim = 5000, burn_in = 1000)
```

Arguments

log_posterior	Function taking parameter vector and returning log posterior density.
init_par	Initial parameter vector.
n_sim	Total number of MCMC iterations.
burn_in	Number of initial iterations to discard as burn-in.

Value

S3 object of class `bayes_fit` with parameter chains, posterior means, and credible intervals.

Examples

```
log_post <- function(th) {
  if (th[1] <= 0) return(-Inf)
  dexp(th[1], rate = 1, log = TRUE) +
    sum(dexp(c(0.5, 1.2, 0.8), rate = th[1], log = TRUE))
}
bayes_gibbs_censored(
  log_post, init_par = c(1.0),
  n_sim = 1000, burn_in = 200
)
```

---

bayes_mh_censored *Bayesian Inference via Metropolis-Hastings MCMC Algorithm*

Description

Bayesian Inference via Metropolis-Hastings MCMC Algorithm

Usage

```
bayes_mh_censored(
  log_posterior,
  init_par,
  n_sim = 5000,
  burn_in = 1000,
  proposal_sd = 0.1
)
```

Arguments

`log_posterior`	Function evaluating log-posterior density.
`init_par`	Initial parameter values.
`n_sim`	Total MCMC sample size.
`burn_in`	Discarded burn-in samples.
`proposal_sd`	Proposal standard deviation vector.

Value

S3 object of class `bayes_fit`.

Examples

```
log_post <- function(th) {
  if (th[1] <= 0) return(-Inf)
  dexp(th[1], rate = 1, log = TRUE) +
    sum(dexp(c(0.5, 1.2, 0.8), rate = th[1], log = TRUE))
}
```

```
bayes_mh_censored(
  log_post, init_par = c(1.0),
  n_sim = 1000, burn_in = 200
)
```

`gen_competing_risks` *Generate Data for Competing Risks Analysis*

Description

Generate Data for Competing Risks Analysis

Usage

```
gen_competing_risks(
  pdf1,
  cdf1,
  pdf2,
  cdf2,
  lower = 0,
  upper = Inf,
  n,
  censoring_type = c("type1_hybrid", "type2_hybrid", "prog_hybrid"),
  r = NULL,
  T_star = NULL,
  R_plan = NULL,
  seed = NULL
)
```

Arguments

`pdf1`	Density function for Cause 1.
`cdf1`	Cumulative distribution function for Cause 1.
`pdf2`	Density function for Cause 2.
`cdf2`	Cumulative distribution function for Cause 2.
`lower`	Lower bound of support.
`upper`	Upper bound of support.
`n`	Total sample size.
`censoring_type`	Type of censoring ("type1_hybrid", "type2_hybrid", "prog_hybrid").
`r`	Number of failures required.
`T_star`	Time limit.
`R_plan`	Progressive plan (if applicable).
`seed`	Optional random seed.

Value

A list containing observed failure times, causes of failure (1 or 2), and censoring flags.

Examples

```
gen_competing_risks(
  pdf1 = function(x) dexp(x, rate = 1),
  cdf1 = function(x) pexp(x, rate = 1),
  pdf2 = function(x) dexp(x, rate = 1.5),
  cdf2 = function(x) pexp(x, rate = 1.5),
  lower = 0, upper = 10, n = 25,
  censoring_type = "type1_hybrid",
  r = 15, T_star = 1.0, seed = 123
)
```

---

gen_gen_prog_hybrid *Generate Data under Generalized Progressive Hybrid Censoring Scheme*

Description

Generate Data under Generalized Progressive Hybrid Censoring Scheme

Usage

```
gen_gen_prog_hybrid(
  pdf,
  cdf,
  lower = 0,
  upper = Inf,
  n,
  m,
  k,
  T_star,
  R_plan,
  seed = NULL
)
```

Arguments

pdf	Probability density function.
cdf	Cumulative distribution function.
lower	Lower bound of support.
upper	Upper bound of support.
n	Total sample size.
m	Target number of failures.

k	Threshold failure count.
T_star	Termination time limit.
R_plan	Progressive censoring plan vector of length m.
seed	Optional integer random seed for reproducibility.

Value

A list containing observed failure times, progressive removal plan, and censoring details.

Examples

```
gen_gen_prog_hybrid(
  pdf = function(x) dexp(x, rate = 1),
  cdf = function(x) pexp(x, rate = 1),
  lower = 0, upper = 10, n = 20, m = 10,
  k = 5, T_star = 1.2, R_plan = rep(1, 10),
  seed = 123
)
```

---

gen_stress_strength *Generate Data for Stress-Strength Reliability Models*

Description

Generate Data for Stress-Strength Reliability Models

Usage

```
gen_stress_strength(
  pdf_X,
  cdf_X,
  pdf_Y,
  cdf_Y,
  lower_X = 0,
  upper_X = Inf,
  lower_Y = 0,
  upper_Y = Inf,
  n_X,
  n_Y,
  censoring_type = c("type1_hybrid", "type2_hybrid", "prog_hybrid"),
  r_X = NULL,
  T_X = NULL,
  r_Y = NULL,
  T_Y = NULL,
  seed = NULL
)
```

Arguments

pdf_X	Density function for stress X.
cdf_X	Cumulative distribution function for stress X.
pdf_Y	Density function for strength Y.
cdf_Y	Cumulative distribution function for strength Y.
lower_X	Lower bound for X.
upper_X	Upper bound for X.
lower_Y	Lower bound for Y.
upper_Y	Upper bound for Y.
n_X	Sample size for stress X.
n_Y	Sample size for strength Y.
censoring_type	Censoring scheme.
r_X	Minimum required failures for X.
T_X	Time cutoff for X.
r_Y	Minimum required failures for Y.
T_Y	Time cutoff for Y.
seed	Optional random seed.

Value

A list containing generated stress dataset X and strength dataset Y.

Examples

```
gen_stress_strength(
  pdf_X = function(x) dexp(x, rate = 1.2),
  cdf_X = function(x) pexp(x, rate = 1.2),
  pdf_Y = function(y) dexp(y, rate = 0.8),
  cdf_Y = function(y) pexp(y, rate = 0.8),
  n_X = 20, n_Y = 20,
  censoring_type = "type1_hybrid",
  r_X = 12, T_X = 1.2, r_Y = 12, T_Y = 1.5, seed = 123
)
```

---

gen_type1_hybrid *Generate Data under Type-I Hybrid Censoring Scheme*

Description

Generate Data under Type-I Hybrid Censoring Scheme

Usage

```
gen_type1_hybrid(pdf, cdf, lower = 0, upper = Inf, n, r, T_star, seed = NULL)
```

Arguments

pdf	Probability density function.
cdf	Cumulative distribution function.
lower	Lower bound of the support of the distribution.
upper	Upper bound of the support of the distribution.
n	Total sample size.
r	Minimum required number of failures.
T_star	Fixed termination time.
seed	Optional integer random seed for reproducibility.

Value

A list containing:

observed_times	Vector of observed failure/censoring times.
censor_status	Binary indicator (1 for failure, 0 for censored at T_star).
n_failures	Number of observed failures.
termination_time	Effective termination time of the test.
scheme	Character string indicating censoring scheme.

Examples

```
gen_type1_hybrid(
  pdf = function(x) dexp(x, rate = 1),
  cdf = function(x) pexp(x, rate = 1),
  lower = 0, upper = 10, n = 20, r = 10,
  T_star = 1.5, seed = 123
)
```

---

gen_type2_hybrid *Generate Data under Type-II Hybrid Censoring Scheme*

Description

Generate Data under Type-II Hybrid Censoring Scheme

Usage

```
gen_type2_hybrid(pdf, cdf, lower = 0, upper = Inf, n, r, T_star, seed = NULL)
```

Arguments

pdf	Probability density function.
cdf	Cumulative distribution function.
lower	Lower bound of the support of the distribution.
upper	Upper bound of the support of the distribution.
n	Total sample size.
r	Minimum required number of failures.
T_star	Target observation time limit.
seed	Optional integer random seed for reproducibility.

Value

A list containing observed failure times, censor statuses, and termination details.

Examples

```
gen_type2_hybrid(
  pdf = function(x) dexp(x, rate = 1),
  cdf = function(x) pexp(x, rate = 1),
  lower = 0, upper = 10, n = 20, r = 10,
  T_star = 1.0, seed = 123
)
```

---

importance_sampling_censored

Importance Sampling for Posterior Estimation

Description

Importance Sampling for Posterior Estimation

Usage

```
importance_sampling_censored(
  log_target,
  proposal_pdf,
  rproposal,
  n_sim = 10000
)
```

Arguments

log_target	Function evaluating log target density (unnormalized log posterior).
proposal_pdf	Density function of proposal distribution.
rproposal	Random generation function for proposal distribution.
n_sim	Sample size of proposal draws.

Value

List with posterior mean estimates, normalized importance weights, and effective sample size.

Examples

```
importance_sampling_censored(
  log_target = function(th) dexp(th, rate = 2, log = TRUE),
  proposal_pdf = function(th) dexp(th, rate = 1),
  rproposal = function(n) rexp(n, rate = 1),
  n_sim = 1000
)
```

---

lindley_approx_censored

Lindley Asymptotic Approximation for Posterior Expectations

Description

Lindley Asymptotic Approximation for Posterior Expectations

Usage

```
lindley_approx_censored(log_lik, log_prior, init_par)
```

Arguments

`log_lik`	Log-likelihood function of parameter vector.
`log_prior`	Log-prior density function of parameter vector.
`init_par`	Maximum likelihood estimate or posterior mode.

Value

List with Lindley posterior estimates and standard errors.

Examples

```
lindley_approx_censored(
  log_lik = function(th) -sum((c(1.2, 0.8, 1.5) - th[1])^2),
  log_prior = function(th) dexp(th[1], rate = 1, log = TRUE),
  init_par = c(1.1)
)
```

`mle_competing_risks` *Maximum Likelihood Estimation for Competing Risks Analysis*

Description

Maximum Likelihood Estimation for Competing Risks Analysis

Usage

```
mle_competing_risks(data, pdf1, cdf1, pdf2, cdf2, init_par, method = "BFGS")
```

Arguments

`data`	Object of class `comp_risk_rel_data` generated by `gen_competing_risks`.
`pdf1`	Density function for Cause 1.
`cdf1`	CDF function for Cause 1.
`pdf2`	Density function for Cause 2.
`cdf2`	CDF function for Cause 2.
`init_par`	Vector of initial parameter values for both causes c(theta1, theta2).
`method`	Optimization method.

Value

S3 object of class `mle_fit`.

Examples

```
dat <- gen_competing_risks(
  pdf1 = function(x) dexp(x, rate = 1),
  cdf1 = function(x) pexp(x, rate = 1),
  pdf2 = function(x) dexp(x, rate = 1.5),
  cdf2 = function(x) pexp(x, rate = 1.5),
  lower = 0, upper = 10, n = 25,
  censoring_type = "type1_hybrid",
  r = 15, T_star = 1.0, seed = 123
)
mle_competing_risks(
  data = dat,
  pdf1 = function(x, th) dexp(x, rate = th[1]),
  cdf1 = function(x, th) pexp(x, rate = th[1]),
  pdf2 = function(x, th) dexp(x, rate = th[2]),
  cdf2 = function(x, th) pexp(x, rate = th[2]),
  init_par = c(0.8, 1.2), method = "BFGS"
)
```

`mle_gen_prog_hybrid` *Maximum Likelihood Estimation under Generalized Progressive Hybrid Censoring*

Description

Maximum Likelihood Estimation under Generalized Progressive Hybrid Censoring

Usage

```
mle_gen_prog_hybrid(
  data,
  pdf,
  cdf,
  init_par,
  R_plan = NULL,
  method = "BFGS",
  lower = -Inf,
  upper = Inf
)
```

Arguments

`data`	Object of class `comp_risk_rel_data`.
`pdf`	Density function.
`cdf`	CDF function.
`init_par`	Initial parameter values.
`R_plan`	Progressive removal vector.
`method`	Maximization algorithm.
`lower`	Lower parameter bound.
`upper`	Upper parameter bound.

Value

S3 object of class `mle_fit`.

Examples

```
dat <- gen_gen_prog_hybrid(
  pdf = function(x) dexp(x, rate = 1),
  cdf = function(x) pexp(x, rate = 1),
  lower = 0, upper = 10, n = 20, m = 10, k = 5,
  T_star = 1.2, R_plan = rep(1, 10), seed = 123
)
mle_gen_prog_hybrid(
  data = dat,
```

```
  pdf = function(x, theta) dexp(x, rate = theta[1]),
  cdf = function(x, theta) pexp(x, rate = theta[1]),
  init_par = c(0.8), R_plan = rep(1, 10), method = "BFGS"
)
```

`mle_type1_hybrid` *Maximum Likelihood Estimation under Type-I Hybrid Censoring*

Description

Maximum Likelihood Estimation under Type-I Hybrid Censoring

Usage

```
mle_type1_hybrid(
  data,
  pdf,
  cdf,
  init_par,
  method = "BFGS",
  lower = -Inf,
  upper = Inf
)
```

Arguments

`data`	Object of class `comp_risk_rel_data` or list with `observed_times`, `censor_status`, `termination_time`, and total sample size `n`.
`pdf`	Parametric probability density function of time x and vector parameter theta.
`cdf`	Parametric cumulative distribution function of time x and vector parameter theta.
`init_par`	Initial parameter vector for numerical optimization.
`method`	Maximization routine: "NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", or "NM".
`lower`	Lower bound for parameter vector.
`upper`	Upper bound for parameter vector.

Value

S3 object of class `mle_fit` with parameter estimates, log-likelihood, SE, and information criteria.

Examples

```
dat <- gen_type1_hybrid(
  pdf = function(x) dexp(x, rate = 1),
  cdf = function(x) pexp(x, rate = 1),
  lower = 0, upper = 10, n = 20, r = 10,
  T_star = 1.5, seed = 123
)
mle_type1_hybrid(
  data = dat,
  pdf = function(x, theta) dexp(x, rate = theta[1]),
  cdf = function(x, theta) pexp(x, rate = theta[1]),
  init_par = c(0.8), method = "BFGS"
)
```

---

mle_type2_hybrid *Maximum Likelihood Estimation under Type-II Hybrid Censoring*

Description

Maximum Likelihood Estimation under Type-II Hybrid Censoring

Usage

```
mle_type2_hybrid(
  data,
  pdf,
  cdf,
  init_par,
  method = "BFGS",
  lower = -Inf,
  upper = Inf
)
```

Arguments

data	Object of class comp_risk_rel_data or list.
pdf	Parametric density function.
cdf	Parametric CDF function.
init_par	Initial parameter values.
method	Maximization method ("NR", "BFGS", "BFGSR", "BHHH", "SANN", "CG", "NM").
lower	Lower parameter bounds.
upper	Upper parameter bounds.

Value

S3 object of class mle_fit.

Examples

```
dat <- gen_type2_hybrid(
  pdf = function(x) dexp(x, rate = 1),
  cdf = function(x) pexp(x, rate = 1),
  lower = 0, upper = 10, n = 20, r = 10,
  T_star = 1.0, seed = 123
)
mle_type2_hybrid(
  data = dat,
  pdf = function(x, theta) dexp(x, rate = theta[1]),
  cdf = function(x, theta) pexp(x, rate = theta[1]),
  init_par = c(0.8), method = "BFGS"
)
```

---

optimal_design_hybrid *Optimal Design Selection for Hybrid Censoring Schemes*

Description

Optimal Design Selection for Hybrid Censoring Schemes

Usage

```
optimal_design_hybrid(
  n,
  r_candidates,
  T_candidates,
  pdf,
  cdf,
  par,
  criterion = c("D-optimal", "A-optimal", "cost")
)
```

Arguments

n	Sample size.
r_candidates	Candidate failure count choices.
T_candidates	Candidate time limit choices.
pdf	Probability density function of lifetime distribution.
cdf	Cumulative distribution function of lifetime distribution.
par	Model parameters.
criterion	Optimization criterion: "D-optimal" (maximize determinant of Fisher Information), "A-optimal" (minimize trace of inverse Fisher Information), or "cost".

Value

List containing optimal choice of (r, T) and associated information measure.

Examples

```
optimal_design_hybrid(
  n = 20, r_candidates = c(5, 10, 15), T_candidates = c(0.5, 1.0, 1.5),
  pdf = function(x, th) dexp(x, rate = th[1]),
  cdf = function(x, th) pexp(x, rate = th[1]),
  par = c(1.0), criterion = "D-optimal"
)
```

---

plot.comp_risk_rel_data *Plot Method for Objects of Class comp_risk_rel_data*

Description

Plot Method for Objects of Class comp_risk_rel_data

Usage

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

Arguments

x	Object of class comp_risk_rel_data.
...	Additional graphical parameters.

Value

No return value, called for side effects.

---

plot_diagnostics *Diagnostic Plots for Censored Data Schemes*

Description

Produces 3 diagnostic plots (Histogram with PDF overlay, Dot plot, and ACF plot) to verify sample distributional properties and independence.

Usage

```
plot_diagnostics(data, pdf = NULL, main = "Censoring Diagnostic Plots")
```

Arguments

data	Object of class comp_risk_rel_data or a numeric vector of failure times.
pdf	Optional probability density function to overlay on histogram.
main	Overall plot title.

Value

No return value, called for side effects (renders diagnostic plots).

Examples

```
dat <- gen_type1_hybrid(
  pdf = function(x) dexp(x, rate = 1), cdf = function(x) pexp(x, rate = 1),
  lower = 0, upper = 10, n = 20, r = 10, T_star = 1.5, seed = 123
)
plot_diagnostics(dat, pdf = function(x) dexp(x, rate = 1))
```

---

print.bayes_fit *Print Method for Bayesian Fit Objects*

Description

Print Method for Bayesian Fit Objects

Usage

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

Arguments

x	Object of class bayes_fit.
...	Unused parameters.

Value

No return value, called for side effects.

---

print.mle_fit *Print Method for MLE Fit Objects*

Description

Print Method for MLE Fit Objects

Usage

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

Arguments

x	Object of class mle_fit.
...	Unused additional parameters.

Value

No return value, called for side effects.

---

print.sampling_plan *Print Method for Sampling Plan Objects*

Description

Print Method for Sampling Plan Objects

Usage

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

Arguments

x	Object of class sampling_plan.
...	Unused parameters.

Value

No return value, called for side effects.

---

sampling_plan_exponential *Reliability Acceptance Sampling Plan for Exponential Distribution (10.5.1)*

Description

Reliability Acceptance Sampling Plan for Exponential Distribution (10.5.1)

Usage

```
sampling_plan_exponential(
  n,
  T_star,
  r,
  alpha = 0.05,
  beta = 0.1,
  theta0,
  theta1
)
```

Arguments

n	Sample size placed on test.
T_star	Fixed test termination time.
r	Required failure limit.
alpha	Producer's risk.
beta	Consumer's risk.
theta0	Acceptable quality level (AQL) mean lifetime.
theta1	Rejectable quality level (RQL) mean lifetime.

Value

S3 object of class `sampling_plan` with acceptance constant c, operating characteristic (OC) values, and decision rule.

Examples

```
sampling_plan_exponential(
  n = 20, T_star = 1.0, r = 10,
  alpha = 0.05, beta = 0.10,
  theta0 = 2.0, theta1 = 0.8
)
```

---

sampling_plan_weibull *Reliability Acceptance Sampling Plan for Weibull Distribution (10.5.2)*

Description

Reliability Acceptance Sampling Plan for Weibull Distribution (10.5.2)

Usage

```
sampling_plan_weibull(
  n,
  T_star,
  r,
  beta_shape,
  alpha = 0.05,
  beta = 0.1,
  theta0,
  theta1
)
```

Arguments

n	Sample size.
T_star	Termination time.
r	Minimum failure limit.
beta_shape	Known shape parameter of Weibull distribution.
alpha	Producer's risk.
beta	Consumer's risk.
theta0	AQL scale parameter.
theta1	RQL scale parameter.

Value

S3 object of class sampling_plan.

Examples

```
sampling_plan_weibull(
  n = 25, T_star = 1.5, r = 12, beta_shape = 1.5,
  alpha = 0.05, beta = 0.10, theta0 = 3.0, theta1 = 1.0
)
```

---

stress_strength_rel *Stress-Strength Reliability Estimation R = P(X < Y)*

Description

Stress-Strength Reliability Estimation R = P(X < Y)

Usage

```
stress_strength_rel(
  data_X,
  data_Y,
  pdf_X,
  cdf_X,
  pdf_Y,
  cdf_Y,
  init_par_X,
  init_par_Y,
  method = "BFGS"
)
```

Arguments

data_X	Object of class comp_risk_rel_data for stress X.
data_Y	Object of class comp_risk_rel_data for strength Y.
pdf_X	Density function for X.
cdf_X	CDF function for X.
pdf_Y	Density function for Y.
cdf_Y	CDF function for Y.
init_par_X	Initial parameter value for X.
init_par_Y	Initial parameter value for Y.
method	Optimization method.

Value

S3 object containing parameter estimates, estimated reliability R = P(X < Y), standard error, and asymptotic confidence interval.

Examples

```
dat <- gen_stress_strength(
  pdf_X = function(x) dexp(x, rate = 1.2), cdf_X = function(x) pexp(x, rate = 1.2),
  pdf_Y = function(y) dexp(y, rate = 0.8), cdf_Y = function(y) pexp(y, rate = 0.8),
  lower_X = 0, upper_X = 10, lower_Y = 0, upper_Y = 10,
  n_X = 20, n_Y = 20, censoring_type = "type1_hybrid",
  r_X = 12, T_X = 1.2, r_Y = 12, T_Y = 1.5, seed = 123
)
stress_strength_rel(
  data_X = dat$data_X, data_Y = dat$data_Y,
 pdf_X = function(x, th) dexp(x, rate = th[1]), cdf_X = function(x, th) pexp(x, rate = th[1]),
 pdf_Y = function(y, th) dexp(y, rate = th[1]), cdf_Y = function(y, th) pexp(y, rate = th[1]),
  init_par_X = c(1.0), init_par_Y = c(0.7), method = "BFGS"
)
```

summary.bayes_fit *Summary Method for Bayesian Fit Objects*

Description

Summary Method for Bayesian Fit Objects

Usage

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

Arguments

object	Object of class bayes_fit.
...	Unused parameters.

Value

Summary table with posterior estimates and credible intervals.

summary.mle_fit *Summary Method for MLE Fit Objects*

Description

Summary Method for MLE Fit Objects

Usage

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

Arguments

object	Object of class mle_fit.
...	Unused additional parameters.

Value

A matrix summary of parameter estimates, standard errors, z-values, and p-values.

Index