Package 'gpcihybridIImcmc'

August 21, 2026

Type Package

Title Generalized Process Capability Indices for Hybrid Type-II Censored Data using MCMC

Version 0.1.0

Maintainer Shikhar Tyagi <shikhar1093tyagi@gmail.com>

Description Implements Bayesian Markov Chain Monte Carlo (MCMC) estimation using Metropolis-Hastings within Gibbs sampler for Generalized Process Capability Indices (GPCIs) under Hybrid Type-II censored lifetime data. Supports classical and generalized capability indices including Cpy, Cp, Cpk, Cpm, Cpmk, Spmk, CpTk, Cpc, CNp, CNpk, CNpm, CNpmk, CNpmc, and CNpmkc. Calculates posterior point estimates, bias, mean squared error (MSE), Bayes risk, Highest Posterior Density (HPD) credible intervals at 90%, 95%, and 99% levels, Heidelberger and Welch's MCMC convergence diagnostics, and coverage probabilities. Accommodates user-defined probability density/mass functions, cumulative distribution functions, and survival functions. Based on methods described in Childs et al. (2003) <doi:10.1080/0266476032000053637>, Kundu and Pradhan (2009) <doi:10.1016/j.spl.2008.09.006>, Saha and Dey (2019) <doi:10.1007/s41872-019-00081-4>, Alotaibi et al. (2022) <doi:10.1155/2022/3135264>, Dey et al. (2017) <doi:10.1080/03610918.2017.1280166>, and Wu

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.3.3

Imports coda, stats, graphics

Suggests gofPHCS, testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

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

Repository CRAN

Date/Publication 2026-08-21 13:00:14 UTC

Contents

`coef.gpci_mcmc` *Coef Method for gpci_mcmc Objects*

Description

Extracts posterior mean estimates of capability indices or distribution parameters.

Usage

```
## S3 method for class 'gpci_mcmc'
coef(object, what = c("indices", "parameters"), ...)
```

Arguments

object	An object of class "gpci_mcmc".
what	Character string: "indices" (default) or "parameters".
...	Additional arguments.

Value

Named numeric vector of posterior estimates.

---

compute_gpcis *Compute Generalized Process Capability Indices (GPCIs)*

Description

Evaluates classical and generalized Process Capability Indices (GPCIs) given process specification limits and a target distribution specification.

Usage

```
compute_gpcis(
  distribution,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = c("Cpy", "Cp", "Cpk", "Cpm", "Cpmk", "Spmk", "CNpmc"),
  mode = c("moments", "quantile"),
  u = 1,
  v = 1,
  C0 = 1,
  C1 = 0,
  C2 = 1,
  tolerance_t = USL - LSL,
  P0 = 0.9973002,
  LDL = LSL,
  UDL = USL
)
```

Arguments

distribution	A gpci_dist or gpci_dist_hybrid2mcmc object.
USL	Numeric Upper Specification Limit. Must be strictly greater than LSL.
LSL	Numeric Lower Specification Limit. Must be strictly less than USL.
target	Numeric Process Target value (defaults to midpoint (USL + LSL) / 2).
indices	Character vector of capability indices to compute. Choices include "Cpy", "Cp", "Cpk", "Cpu", "Cpl", "Cpm", "Cpmk", "Spmk", "CpTk", "Cpc", "CNp", "CNpk", "CNpm", "CNpmk", "CNpmc", "CNpmkc", "Cp_uv", "CNp_uv".

mode	Character string specifying computation mode: "moments" (default; uses mean and variance) or "quantile" (uses robust quantiles).
u	Non-negative numeric weight parameter u for generalized family. Defaults to 1.
v	Non-negative numeric weight parameter v for generalized family. Defaults to 1.
C0	Non-negative numeric coefficient for tolerance cost function. Defaults to 1.
C1	Non-negative numeric coefficient for tolerance cost function. Defaults to 0.
C2	Non-negative numeric coefficient for tolerance cost function. Defaults to 1.
tolerance_t	Positive numeric process tolerance t. Defaults to USL - LSL.
P0	Desirable conformance probability for Cpc and Cpy. Defaults to 0.9973002.
LDL	Lower Desired Limit for CpTk. Defaults to LSL.
UDL	Upper Desired Limit for CpTk. Defaults to USL.

Details

Supports moment-based and quantile-based capability indices including:

- C_{py}: Conformance yield capability index $(F(USL) - F(LSL))/P_0$.
- C_p, C_{pk}, C_{pu}, C_{pl}, C_{pm}, C_{pmk}: Classical capability indices.
- S_{pmk}: Conformance loss and target deviation index.
- C_{pTk}: Asymmetric target-based index.
- C_{pc}: Cost-ratio capability index.
- C_{Np}, C_{Npk}, C_{Npm}, C_{Npmk}: Quantile-based non-normal indices.
- C_{Npmc}, C_{Npmkc}: Tolerance cost-adjusted capability indices using $C_M(t) = C_0 + C_1 \exp(-C_2 t)$.
- $C_p(u, v)$, $C_{Np}(u, v)$: Generalized weighted family of capability indices.

Value

A named numeric vector of computed GPCI values.

Examples

```
dist <- dist_weibull(shape = 2, scale = 10)
compute_gpcis(dist, USL = 15, LSL = 5, target = 10)
```

`define_distribution_hybrid2mcmc` *Define a Lifetime Distribution Object for Hybrid Type-II Censored Data*

Description

Constructor to define a continuous or discrete lifetime distribution object for Bayesian MCMC capability analysis under Hybrid Type-II censored data.

Usage

```
define_distribution_hybrid2mcmc(
  name = "Custom",
  pdf = NULL,
  cdf = NULL,
  sf = NULL,
  surv = NULL,
  quantile = NULL,
  moments = NULL,
  param_names = NULL,
  default_params = list(),
  params = list(),
  support = c(0, Inf)
)

create_custom_dist(
  name = "Custom",
  pdf = NULL,
  cdf = NULL,
  sf = NULL,
  surv = NULL,
  quantile = NULL,
  moments = NULL,
  param_names = NULL,
  default_params = list(),
  params = list(),
  support = c(0, Inf)
)
```

Arguments

`name`	Character string naming the distribution (e.g. `"custom_model"`).
`pdf`	Function `function(x, ...)` computing probability density/mass. Defaults to `NULL`.
`cdf`	Function `function(q, ...)` computing cumulative probability. Defaults to `NULL`.

sf	Optional function function(q, ...) computing survival values ($S(t) = 1 - F(t)$). Defaults to NULL.
surv	Alias for sf. Defaults to NULL.
quantile	Optional function function(p, ...) computing quantiles. Defaults to NULL.
moments	Optional function function(...) returning a list with mean and var. Defaults to NULL.
param_names	Character vector of parameter names.
default_params	Named list or numeric vector of default parameter values.
params	Alias for default_params. Defaults to list().
support	Numeric vector of length 2 defining lower and upper bounds of support. Defaults to c(0, Inf).

Details

Users can define a distribution by specifying any of its Probability Density/Mass Function (PDF/PMF), Cumulative Distribution Function (CDF), Survival Function (SF), Quantile Function, or Moments. Missing functions are automatically and stably derived numerically.

Value

An S3 object of class "gpci_dist_hybrid2mcmc" and "gpci_dist" containing:

name	Character string naming the distribution.
pdf	Vectorized density/mass function.
cdf	Vectorized cumulative distribution function.
sf	Vectorized survival function.
surv	Alias for sf.
quantile	Vectorized quantile function.
moments	Function computing mean and variance.
param_names	Character vector of parameter names.
params	Named list of default parameter values.
support	Numeric vector of length 2 specifying support.

Examples

```
dist_custom <- define_distribution_hybrid2mcmc(
  name = "CustomExp",
  pdf = function(x, rate) dexp(x, rate = rate),
  cdf = function(q, rate) pexp(q, rate = rate),
  param_names = "rate",
  params = list(rate = 1)
)
print(dist_custom)
```

dist_exponential *Exponential Distribution Constructor*

Description

Exponential Distribution Constructor

Usage

```
dist_exponential(rate = 1)
```

Arguments

rate	Rate parameter (> 0).

Value

A gpci_dist_hybrid2mcmc object for Exponential distribution.

Examples

```
dist_e <- dist_exponential(rate = 0.5)
```

dist_gamma *Gamma Distribution Constructor*

Description

Gamma Distribution Constructor

Usage

```
dist_gamma(shape = 1, scale = 1)
```

Arguments

shape	Shape parameter (> 0).
scale	Scale parameter (> 0).

Value

A gpci_dist_hybrid2mcmc object for Gamma distribution.

Examples

```
dist_g <- dist_gamma(shape = 3, scale = 2)
```

dist_lindley *Lindley Distribution Constructor*

Description

Lindley Distribution Constructor

Usage

```
dist_lindley(theta = 1)
```

Arguments

theta	Parameter theta (> 0).

Value

A gpci_dist_hybrid2mcmc object for Lindley distribution.

Examples

```
dist_lind <- dist_lindley(theta = 1.5)
```

dist_logistic *Logistic Distribution Constructor*

Description

Logistic Distribution Constructor

Usage

```
dist_logistic(location = 0, scale = 1)
```

Arguments

location	Location parameter.
scale	Scale parameter (> 0).

Value

A gpci_dist_hybrid2mcmc object for Logistic distribution.

Examples

```
dist_l <- dist_logistic(location = 0, scale = 1)
```

dist_logistic_exponential *Logistic-Exponential Distribution Constructor*

Description

Logistic-Exponential Distribution Constructor

Usage

```
dist_logistic_exponential(alpha = 1, lambda = 1)
```

Arguments

alpha	Shape parameter alpha (> 0).
lambda	Scale parameter lambda (> 0).

Value

A gpci_dist_hybrid2mcmc object for Logistic-Exponential distribution.

Examples

```
dist_le <- dist_logistic_exponential(alpha = 1.2, lambda = 0.8)
```

dist_loglogistic *Log-Logistic Distribution Constructor*

Description

Log-Logistic Distribution Constructor

Usage

```
dist_loglogistic(shape = 1, scale = 1)
```

Arguments

shape	Shape parameter (> 0).
scale	Scale parameter (> 0).

Value

A gpci_dist_hybrid2mcmc object for Log-Logistic distribution.

Examples

```
dist_ll <- dist_loglogistic(shape = 2, scale = 3)
```

dist_lognormal *Lognormal Distribution Constructor*

Description

Lognormal Distribution Constructor

Usage

```
dist_lognormal(meanlog = 0, sdlog = 1)
```

Arguments

meanlog	Mean on the log scale.
sdlog	Standard deviation on the log scale (> 0).

Value

A gpci_dist_hybrid2mcmc object for Lognormal distribution.

Examples

```
dist_ln <- dist_lognormal(meanlog = 1, sdlog = 0.5)
```

dist_normal *Normal Distribution Constructor*

Description

Normal Distribution Constructor

Usage

```
dist_normal(mean = 0, sd = 1)
```

Arguments

mean	Mean parameter.
sd	Standard deviation parameter (> 0).

Value

A gpci_dist_hybrid2mcmc object for Normal distribution.

Examples

```
dist_n <- dist_normal(mean = 10, sd = 2)
```

dist_weibull *Weibull Distribution Constructor*

Description

Weibull Distribution Constructor

Usage

```
dist_weibull(shape = 1, scale = 1)
```

Arguments

shape	Shape parameter (> 0).
scale	Scale parameter (> 0).

Value

A gpci_dist_hybrid2mcmc object for Weibull distribution.

Examples

```
dist_w <- dist_weibull(shape = 2, scale = 5)
```

eval_log_prior *Safe Evaluation of Log-Prior Density*

Description

Evaluates the joint log-prior density for a vector or list of parameter values.

Usage

```
eval_log_prior(params, priors = NULL)
```

Arguments

params	Named numeric vector or list of distribution parameters.
priors	List of prior specifications or a custom log-prior function function(params). If NULL, a non-informative Gamma(0.001, 0.001) prior is assumed for positive parameters.

Value

Numeric scalar representing the joint log-prior density value. Returns -Inf for values outside parameter support.

Examples

```
eval_log_prior(c(shape = 2, scale = 5))
pr_spec <- list(rate = list(dist = "gamma", params = c(shape = 1, rate = 1)))
eval_log_prior(c(rate = 0.5), priors = pr_spec)
```

`gof_test_hybrid2` *Goodness-of-Fit Testing for Hybrid Type-II Censored Data using gofPHCS*

Description

Performs goodness-of-fit testing for Hybrid Type-II censored lifetime data using the `gofPHCS` package (`gofPHCS::gof_test`).

Usage

```
gof_test_hybrid2(
  fit = NULL,
  x = NULL,
  r = NULL,
  tc = NULL,
  n = NULL,
  distribution = NULL,
  statistic = "auto",
  p.method = c("auto", "asymptotic", "montecarlo"),
  nsim = 999,
  seed = NULL,
  conf.level = 0.95,
  ...
)

gof_test(
  fit = NULL,
  x = NULL,
  r = NULL,
  tc = NULL,
  n = NULL,
  distribution = NULL,
  statistic = "auto",
  p.method = c("auto", "asymptotic", "montecarlo"),
  nsim = 999,
  seed = NULL,
  conf.level = 0.95,
  ...
)
```

Arguments

fit	Optional gpci_mcmc object returned by gpci_hybrid2_mcmc. If supplied, x, r, tc, n, and distribution are extracted automatically. Defaults to NULL.
x	Numeric vector of observed failure times (required if fit is not supplied). Defaults to NULL.
r	Positive integer target number of failures (required if fit is not supplied). Defaults to NULL.
tc	Positive numeric fixed censoring time (required if fit is not supplied). Defaults to NULL.
n	Positive integer total sample size (required if fit is not supplied). Defaults to NULL.
distribution	A gpci_dist or gpci_dist_hybrid2mcmc distribution object (required if fit is not supplied). Defaults to NULL.
statistic	Character string specifying the test statistic (e.g. "auto", "AD", "CvM", "KS"). Defaults to "auto".
p.method	Character string specifying method for calculating p-values: "auto" (default), "asymptotic", or "montecarlo".
nsim	Positive integer scalar specifying number of Monte Carlo replicates. Defaults to 999.
seed	Optional integer seed for reproducibility. Defaults to NULL.
conf.level	Numeric confidence level for test. Defaults to 0.95.
...	Additional arguments passed to gofPHCS::gof_test.

Value

An S3 object of class "gpci_gof_hybrid2" containing:

fit	The original gpci_mcmc object, or NULL if raw data was passed.
cens_data	The gofPHCS censored data structure.
gof_result	The returned test result object from gofPHCS::gof_test.
distribution	The tested distribution object.

Examples

```
dist_exp <- dist_exponential(rate = 0.5)
x_obs <- c(0.2, 0.5, 0.8, 1.1)
gof_res <- gof_test_hybrid2(x = x_obs, r = 3, tc = 1.0, n = 10,
                            distribution = dist_exp, p.method = "montecarlo", nsim = 50)
print(gof_res)
```

gpci_hybrid2_mcmc *Main Bayesian MCMC Estimation of GPCIs under Hybrid Type-II Censored Data*

Description

Evaluates Generalized Process Capability Indices (GPCIs) under Hybrid Type-II censored lifetime data using Metropolis-Hastings within Gibbs sampler MCMC.

Usage

```
gpci_hybrid2_mcmc(
  x,
  r,
  tc,
  n,
  distribution = NULL,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  quantile = NULL,
  param_names = NULL,
  start = NULL,
  priors = NULL,
  length_chain = 10000,
  burn_in = 2000,
  thinning = 5,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = c("Cpy", "Cp", "Cpk", "Cpm", "Cpmk", "Spmk", "CNpmc"),
  mode = c("moments", "quantile"),
  u = 1,
  v = 1,
  C0 = 1,
  C1 = 0,
  C2 = 1,
  tolerance_t = USL - LSL,
  P0 = 0.9973002,
  true_param = NULL,
  true_gpci = NULL
)

gpcihybridIImcmc(
  x,
  r,
  tc,
```

```
  n,
  distribution = NULL,
  pdf = NULL,
  cdf = NULL,
  surv = NULL,
  quantile = NULL,
  param_names = NULL,
  start = NULL,
  priors = NULL,
  length_chain = 10000,
  burn_in = 2000,
  thinning = 5,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = c("Cpy", "Cp", "Cpk", "Cpm", "Cpmk", "Spmk", "CNpmc"),
  mode = c("moments", "quantile"),
  u = 1,
  v = 1,
  C0 = 1,
  C1 = 0,
  C2 = 1,
  tolerance_t = USL - LSL,
  P0 = 0.9973002,
  true_param = NULL,
  true_gpci = NULL
)
```

Arguments

x	Numeric vector of observed failure times. Missing values (NA, NaN) are not permitted.
r	Positive integer scalar specifying target number of failures.
tc	Positive numeric scalar specifying fixed censoring time.
n	Positive integer scalar specifying total sample size placed on test ($n \geq length(x)$ and $n \geq r$).
distribution	Optional gpci_dist or gpci_dist_hybrid2mcmc object. If NULL, pdf and cdf must be provided.
pdf	Function function(x, ...) computing probability density or mass.
cdf	Function function(q, ...) computing cumulative distribution.
surv	Optional function function(q, ...) computing survival values.
quantile	Optional function function(p, ...) computing quantiles.
param_names	Character vector of parameter names. Default is names(start).
start	Named numeric vector or list of initial parameter values.
priors	List of prior specifications or log-prior function. If NULL, non-informative priors are used.

length_chain	Positive integer scalar specifying total number of MCMC iterations. Default is 10000.
burn_in	Positive integer scalar specifying number of burn-in iterations. Default is 2000.
thinning	Positive integer scalar specifying thinning factor. Default is 5.
USL	Numeric Upper Specification Limit. Must satisfy USL > LSL.
LSL	Numeric Lower Specification Limit. Must satisfy LSL < USL.
target	Numeric Process Target value (default midpoint (USL + LSL) / 2).
indices	Character vector of GPCIs to compute. Choices include "Cpy", "Cp", "Cpk", "Cpu", "Cpl", "Cpm", "Cpmk", "Spmk", "CpTk", "Cpc", "CNp", "CNpk", "CNpm", "CNpmk", "CNpmc", "CNpmkc", "Cp_uv", "CNp_uv".
mode	Character string specifying mode of computation: "moments" (default) or "quantile".
u	Non-negative numeric weight parameter u for generalized family. Defaults to 1.
v	Non-negative numeric weight parameter v for generalized family. Defaults to 1.
C0	Non-negative numeric coefficient for tolerance cost function. Defaults to 1.
C1	Non-negative numeric coefficient for tolerance cost function. Defaults to 0.
C2	Non-negative numeric coefficient for tolerance cost function. Defaults to 1.
tolerance_t	Positive numeric process tolerance t (defaults to USL - LSL).
P0	Desirable conformance yield for Cpc and Cpy (default 0.9973002).
true_param	Optional named numeric vector of true parameter values for bias/MSE validation. If NULL, initial MLE point estimates are used.
true_gpci	Optional named numeric vector of true GPCI values for bias/MSE validation. If NULL, initial GPCI point estimates are used.

Details

Under Hybrid Type-II censoring, n units are placed on life test. The experiment terminates at $T^* = \max(x_r, T_c)$, where $r \leq n$ is the pre-fixed target number of failures and T_c is the pre-fixed censoring time.

The user can supply observed data (x, r, tc, n), custom functions for probability density/mass (pdf), cumulative distribution (cdf), and survival (surv), along with prior distributions, chain length, burn-in period, and thinning factor.

The algorithm first computes initial parameter estimates under Hybrid Type-II censoring, evaluates baseline GPCI point estimates, runs Metropolis-Hastings within Gibbs sampler MCMC, computes the thinned GPCI chain, and returns comprehensive posterior point estimates, bias, MSE, Bayes risk, HPD credible intervals at 90%, 95%, and 99% levels, Heidelberger and Welch's MCMC convergence diagnostics, and coverage probabilities.

Missing values (NA, NaN) in input vectors or invalid parameter evaluations are safely trapped and imputed using robust baseline statistics. Logical inputs are handled as standard boolean values (TRUE or FALSE).

Value

An object of class "gpci_mcmc" containing:

hybrid2_mle	Named numeric vector of initial maximum likelihood estimates under Hybrid Type-II censoring.
uncensored_mle	Alias for hybrid2_mle.
initial_gpcis	Named numeric vector of GPCI point estimates evaluated at initial MLEs.
param_chain	Numeric matrix of thinned post-burn-in parameter MCMC samples.
gpci_chain	Numeric matrix of thinned post-burn-in GPCI MCMC samples.
param_summary	Data frame containing parameter posterior statistics (Estimate, Bias, MSE, Bayes Risk, 90%, 95%, 99% HPD limits, Heidelberger-Welch test, Convergence Probability, Coverage Probability).
gpci_summary	Data frame containing GPCI posterior statistics (Estimate, Bias, MSE, Bayes Risk, 90%, 95%, 99% HPD limits, Heidelberger-Welch test, Convergence Probability, Coverage Probability).
accept_rates	Named numeric vector of acceptance rates for parameter updates.
data	List containing validated Hybrid Type-II censoring data components.
distribution	Target distribution object used for estimation.
USL, LSL, target	Process specification parameters.
indices	Character vector of evaluated indices.

Examples

```
# Example using custom user-supplied functions for Exponential lifetime data
my_pdf <- function(x, rate) dexp(x, rate = rate)
my_cdf <- function(q, rate) pexp(q, rate = rate)
my_surv <- function(q, rate) pexp(q, rate = rate, lower.tail = FALSE)

x_data <- c(0.5, 1.2, 2.1, 3.4)
fit <- gpci_hybrid2_mcmc(
  x = x_data, r = 3, tc = 2.5, n = 10,
  pdf = my_pdf, cdf = my_cdf, surv = my_surv,
  param_names = "rate", start = c(rate = 0.5),
  length_chain = 1000, burn_in = 200, thinning = 2,
  USL = 8, LSL = 0, target = 4
)
print(fit)
```

---

heidelberger_welch *Heidelberger and Welch's MCMC Convergence Diagnostic*

Description

Conducts the stationarity test and relative half-width test under Heidelberger and Welch's MCMC convergence diagnostic.

Usage

```
heidelberger_welch(x, alpha = 0.05, eps = 0.1)
```

Arguments

x	Numeric vector representing an MCMC sample chain.
alpha	Significance level for the test (default is 0.05).
eps	Target maximum ratio of half-width to sample mean (default is 0.1).

Value

A list containing:

stat	Cramér-von Mises stationarity test statistic.
pvalue	Estimated p-value for the stationarity test.
passed	Logical scalar: TRUE if the stationarity test passed, FALSE otherwise.
hw_stat	Half-width test ratio.
hw_passed	Logical scalar: TRUE if the half-width test passed, FALSE otherwise.
convergence_prob	Estimated convergence probability.

Examples

```
samples <- stats::rnorm(1000)
heidelberger_welch(samples)
```

---

hpd_interval *Highest Posterior Density (HPD) Interval Calculation*

Description

Computes the Highest Posterior Density (HPD) interval for MCMC posterior samples at specified confidence / credibility levels (e.g. 90%, 95%, 99%).

Usage

```
hpd_interval(x, prob = 0.95)
```

Arguments

x	Numeric vector of MCMC posterior samples. Missing and non-finite values are automatically removed.
prob	Credibility level ($1 - \alpha$). Default is 0.95.

Value

A named numeric vector of length 2 containing `lower` and `upper` HPD limits.

Examples

```
samples <- stats::rnorm(1000, mean = 5, sd = 1)
hpd_interval(samples, prob = 0.95)
```

`mcmc_hybrid_ty2` *Parameter Estimation and MCMC Sampling under Hybrid Type-II Censoring*

Description

Fits initial parameter estimates and generates MCMC posterior chains using Metropolis-Hastings within Gibbs sampler for Hybrid Type-II censored lifetime data.

Usage

```
mcmc_hybrid_ty2(
  x,
  r,
  tc,
  n,
  distribution,
  start = NULL,
  priors = NULL,
  length_chain = 10000,
  burn_in = 2000,
  thinning = 5,
  proposal_sd = NULL
)
```

Arguments

`x`	Numeric vector of observed failure times. Missing and non-finite values are not permitted.
`r`	Positive integer scalar specifying the target number of failures ($r \leq n$).
`tc`	Positive numeric scalar specifying the fixed censoring time ($T_c > 0$).
`n`	Positive integer scalar specifying total initial sample size placed on test ($n \geq length(x)$ and $n \geq r$).
`distribution`	A `gpci_dist` or `gpci_dist_hybrid2mcmc` distribution object.
`start`	Named numeric vector or list of initial parameter values. If `NULL`, uses `distribution$params`.
`priors`	List of prior specifications or a custom log-prior function. If `NULL`, non-informative priors are used.

length_chain	Positive integer scalar specifying total MCMC iterations. Default is 10000.
burn_in	Positive integer scalar specifying burn-in iterations to discard. Default is 2000.
thinning	Positive integer scalar specifying thinning factor. Default is 5.
proposal_sd	Optional numeric vector or list of proposal standard deviations for M-H updates.

Details

Under Hybrid Type-II censoring, a total of n items are placed on test. The test terminates at time $T^* = \max(x_r, T_c)$, where r is the pre-fixed target number of failures and T_c is the pre-fixed censoring time. The likelihood function is:

$$L(\theta|x, r, T_c, n) = \left[\prod_{i=1}^{d} f(x_i;\theta)\right] [S(T^*;\theta)]^{n-d}$$

where $d \geq r$ is the number of observed failures up to time T^*.

Value

A list containing:

hybrid2_mle	Named numeric vector of initial maximum likelihood estimates under Hybrid Type-II censoring.
uncensored_mle	Alias for hybrid2_mle.
param_chain	Numeric matrix of thinned post-burn-in parameter MCMC samples.
raw_chain	Numeric matrix of all generated MCMC samples.
accept_rates	Named numeric vector of acceptance rates for each parameter.
data	Validated Hybrid Type-II censoring data summary list.
distribution	Target distribution object.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x_obs <- c(0.4, 0.9, 1.5, 2.3)
res <- mcmc_hybrid_ty2(x = x_obs, r = 3, tc = 2.0, n = 10, distribution = dist_exp,
                       length_chain = 500, burn_in = 100, thinning = 2)
```

---

plot.gpci_mcmc *Plot Method for gpci_mcmc Objects*

Description

Generates MCMC trace plots and posterior density histograms with Highest Posterior Density (HPD) intervals.

Usage

```
## S3 method for class 'gpci_mcmc'
plot(x, index = NULL, ...)
```

Arguments

x	An object of class "gpci_mcmc".
index	Character string specifying the capability index to plot (e.g. "Cpy" or "CNpmc"). Defaults to the first index in the chain.
...	Additional graphical arguments.

Value

Invisibly returns the input object x.

---

print.gpci_dist_hybrid2mcmc *Print Method for gpci_dist_hybrid2mcmc Objects*

Description

Print Method for gpci_dist_hybrid2mcmc Objects

Usage

```
## S3 method for class 'gpci_dist_hybrid2mcmc'
print(x, ...)
```

Arguments

x	An object of class gpci_dist_hybrid2mcmc.
...	Additional arguments.

Value

The invisible object x.

print.gpci_gof_hybrid2 *Print Method for gpci_gof_hybrid2 Objects*

Description

Print Method for gpci_gof_hybrid2 Objects

Usage

```
## S3 method for class 'gpci_gof_hybrid2'
print(x, ...)
```

Arguments

x	An object of class gpci_gof_hybrid2.
...	Additional arguments.

Value

The invisible object x.

print.gpci_mcmc *Print Method for gpci_mcmc Objects*

Description

Prints a structured summary of initial point estimates and MCMC posterior capability analysis results.

Usage

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

Arguments

x	An object of class "gpci_mcmc".
...	Additional print arguments.

Value

Invisibly returns the input object x.

print.summary.gpci_mcmc *Print Method for summary.gpci_mcmc Objects*

Description

Print Method for summary.gpci_mcmc Objects

Usage

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

Arguments

x	An object of class "summary.gpci_mcmc".
...	Additional print arguments.

Value

Invisibly returns the input object x.

summarize_chain *Summarize MCMC Posterior Samples for Parameters and Capability Indices*

Description

Computes posterior mean point estimate, bias, mean squared error (MSE), Bayes risk, Highest Posterior Density (HPD) intervals at 90%, 95%, and 99% levels, Heidelberger-Welch convergence diagnostics, and empirical coverage probabilities.

Usage

```
summarize_chain(chain, true_val = NULL)
```

Arguments

chain	Numeric vector, matrix, or data frame of MCMC posterior draws.
true_val	Optional numeric scalar or named vector representing the true value or baseline estimate for bias and MSE evaluation.

Value

A data frame containing:

`Variable`	Name of the parameter or capability index.
`Estimate`	Posterior mean estimate.
`Bias`	Posterior bias ($Estimate - True\_Val$).
`MSE`	Posterior mean squared error.
`Bayes_Risk`	Bayes risk under squared error loss (posterior variance).
`HPD_90_Lower`, `HPD_90_Upper`	Lower and upper bounds of 90% HPD interval.
`HPD_95_Lower`, `HPD_95_Upper`	Lower and upper bounds of 95% HPD interval.
`HPD_99_Lower`, `HPD_99_Upper`	Lower and upper bounds of 99% HPD interval.
`HW_Stat`	Heidelberger-Welch stationarity test statistic.
`HW_PValue`	p-value for Heidelberger-Welch test.
`HW_Passed`	Logical status of Heidelberger-Welch stationarity test.
`Convergence_Prob`	Estimated convergence probability.
`Coverage_Prob`	Empirical coverage probability (1 if true value is within 95% HPD interval).

Examples

```
samples <- stats::rnorm(1000, mean = 1.5, sd = 0.2)
summarize_chain(samples, true_val = 1.5)
```

---

summary.gpci_dist_hybrid2mcmc *Summary Method for gpci_dist_hybrid2mcmc Objects*

Description

Summary Method for gpci_dist_hybrid2mcmc Objects

Usage

```
## S3 method for class 'gpci_dist_hybrid2mcmc'
summary(object, ...)
```

Arguments

`object`	An object of class `gpci_dist_hybrid2mcmc`.
`...`	Additional arguments.

Value

The invisible object `object`.

summary.gpci_gof_hybrid2 *Summary Method for gpci_gof_hybrid2 Objects*

Description

Summary Method for gpci_gof_hybrid2 Objects

Usage

```
## S3 method for class 'gpci_gof_hybrid2'
summary(object, ...)
```

Arguments

object	An object of class gpci_gof_hybrid2.
...	Additional arguments.

Value

The invisible object object.

summary.gpci_mcmc *Summary Method for gpci_mcmc Objects*

Description

Provides complete posterior statistical summary including HPD intervals and Heidelberger-Welch diagnostics.

Usage

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

Arguments

object	An object of class "gpci_mcmc".
...	Additional summary arguments.

Value

An S3 object of class "summary.gpci_mcmc" containing parameter and GPCI summary tables.

`validate_hybrid2_data` *Validate Hybrid Type-II Censored Data Inputs*

Description

Internal helper to validate inputs for Hybrid Type-II censored lifetime data.

Usage

```
validate_hybrid2_data(x, r, tc, n)
```

Arguments

`x`	Numeric vector of observed failure times. Missing values (`NA`, `NaN`) are not permitted.
`r`	Positive integer scalar specifying the target number of failures.
`tc`	Positive numeric scalar specifying the fixed censoring time.
`n`	Positive integer scalar specifying the total initial sample size placed on test. Must be greater than or equal to `length(x)` and `r`.

Value

A validated list containing:

`x`	Numeric vector of sorted observed failure times.
`r`	Target number of failures (integer).
`tc`	Fixed censoring time (numeric).
`n`	Total sample size on test (integer).
`d`	Number of observed failures (integer).
`t_star`	Effective test termination time `max(x[r], tc)`.

Examples

```
validate_hybrid2_data(x = c(0.5, 1.2, 2.1, 3.4), r = 3, tc = 2.5, n = 10)
```

Index