Package 'gpcihybridIILinApp'

August 21, 2026

Type Package

Title Lindley Approximation for Capability Indices under Hybrid Censoring

Version 0.1.0

Description Provides a comprehensive framework for estimating Generalized Process Capability Indices (GPCIs) under Hybrid Type-II censored lifetime data using Lindley's 3rd-order approximation method (Lindley, 1980 <doi:10.2307/2345271>). Supports user-supplied probability density/mass functions (PDF/PMF), cumulative distribution functions (CDF), survival functions (SF), and quantile functions. Computes Maximum Likelihood Estimates (MLE) using the 'MleCensoR' package (Childs et al., 2003 <doi:10.1007/BF02517803>; Balakrishnan & Kundu, 2013 <doi:10.1002/nav.21545>) and Bayesian posterior expectations for classical and non-normal capability indices, including Cpy (Maiti et al., 2010 <doi:10.1080/16843703.2010.11673233>), Spmk (Dey & Saha, 2019 <doi:10.1007/s41872-019-00081-4>), CpTk (Saha et al., 2019), Cpc (Saha et al., 2022 <doi:10.1080/02664763.2021.1971632>), CNpmc (Alotaibi et al., 2022 <doi:10.1155/2022/3135264>), CNpmkc (Saha et al., 2024 <doi:10.1142/S021853932450013X>), CNpk (Saha et al., 2018 <doi:10.1080/21681015.2018.1437793>), and Vannman's Cp(u,v) family. Generates posterior parameter and GPCI chains via sampling with burn-in and thinning, calculating Bias, Mean Squared Error (MSE), Bayes Risk (SEL and Linex), Highest Posterior Density (HPD) credible intervals at 90%, 95%, and 99% levels, and Heidelberger and Welch's MCMC Convergence Diagnostics (Heidelberger & Welch, 1983 <doi:10.1287/opre.31.6.1109>) with convergence probabilities. Evaluates parametric and non-parametric bootstrap confidence intervals (Percentile, Normal, Basic, BCp, BCa) at 90%, 95%, and 99% levels of significance. Integrates goodness-of-fit testing for Hybrid Type-II censored data via the 'gofPHCS' package.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.0.0)

Imports stats, graphics, numDeriv, MleCensoR, gofPHCS

Suggests testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

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

Maintainer Shikhar Tyagi <shikhar1093tyagi@gmail.com>

Repository CRAN

Date/Publication 2026-08-21 13:00:20 UTC

Contents

`bootstrap_hybrid2_gpci` *Parametric and Non-Parametric Bootstrap Confidence Intervals under Hybrid Type-II Censoring*

Description

Computes bootstrap confidence intervals for Generalized Process Capability Indices (GPCIs) under Hybrid Type-II censored lifetime data using parametric or non-parametric resampling. Supports Percentile, Normal, Basic, and BCp (Bias-Corrected Percentile) bootstrap methods at 90

Usage

```
bootstrap_hybrid2_gpci(
  x,
  r,
  tc,
  n,
  distribution = NULL,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  type = c("parametric", "nonparametric"),
  B = 500,
  conf_levels = c(0.9, 0.95, 0.99),
  u = 1,
  v = 1,
  ...
)

boot_ci_hybrid2(
  x,
  r,
  tc,
  n,
  distribution = NULL,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  type = c("parametric", "nonparametric"),
```

```
  B = 500,
  conf_levels = c(0.9, 0.95, 0.99),
  u = 1,
  v = 1,
  ...
)

bootstrap_gpci_hybrid2(
  x,
  r,
  tc,
  n,
  distribution = NULL,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  type = c("parametric", "nonparametric"),
  B = 500,
  conf_levels = c(0.9, 0.95, 0.99),
  u = 1,
  v = 1,
  ...
)
```

Arguments

x	Numeric vector of observed failure times.
r	Positive integer target number of failures.
tc	Positive numeric fixed censoring time.
n	Positive integer total initial sample size placed on test.
distribution	A gpci_dist_hybrid2 object. Defaults to dist_normal().
USL	Numeric Upper Specification Limit.
LSL	Numeric Lower Specification Limit.
target	Numeric process target value (default (USL + LSL) / 2).
indices	Character vector of GPCI names to evaluate.
type	Character string specifying bootstrap type: "parametric" (default) or "nonparametric".
B	Integer number of bootstrap replications. Default is 500.
conf_levels	Numeric vector of confidence levels (default c(0.90, 0.95, 0.99)).
u, v	Numeric weighting parameters for Vännman's Cp(u,v) family (default 1).
...	Additional arguments passed to capability_hybrid2.

Details

Under Hybrid Type-II censoring, bootstrap samples of initial size n are generated. For parametric bootstrap, n observations are simulated from the fitted distribution, sorted in ascending order $y_1 \leq y_2 \leq \cdots \leq y_n$, and the experiment termination time $T^* = \max(y_r, T_c)$ is applied to select observed failures $y_i \leq T^*$.

Value

An object of S3 class `"gpciBootstrapHybrid2"` containing a list with:

`orig_estimates`	Named numeric vector of original point estimates computed on input data.
`boot_matrix`	A data frame of evaluated GPCI values across all `B` bootstrap samples.
`bootstrap_results`	A list for each GPCI containing original estimate, standard error, bias, raw bootstrap draws, and confidence interval tables (Percentile, Normal, Basic, BCp) at 90%, 95%, and 99% confidence levels.
`type`	Character string indicating bootstrap type (`"parametric"` or `"nonparametric"`).
`B`	Integer number of bootstrap replications.
`specs`	List containing data parameters, distribution, and specification limits.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x_data <- c(0.2, 0.5, 0.8, 1.1)
boot_res <- bootstrap_hybrid2_gpci(
  x = x_data, r = 3, tc = 1.0, n = 10,
  distribution = dist_exp, USL = 3, LSL = 0,
  indices = c("Cpy", "Cp"), B = 5
)
print(boot_res)
```

`capability_hybrid2` *Compute Process Capability Indices under Hybrid Type-II Censored Data*

Description

Computes classical and generalized Process Capability Indices (GPCIs) under Hybrid Type-II censored lifetime data.

Usage

```
capability_hybrid2(
  x = NULL,
  r = NULL,
  tc = NULL,
  n = NULL,
```

```
  distribution,
  USL,
  LSL,
  target = (USL + LSL)/2,
 indices = c("Cpy", "Spmk", "CpTk", "Cpc", "CNpmc", "CNpmkc", "CNpk", "Cp", "Cpk",
    "Cpm", "Cpmk"),
  u = 1,
  v = 1,
  mode = c("moments", "quantile"),
  fit = TRUE,
  start = NULL,
  C0 = 1,
  C1 = 0,
  C2 = 1,
  tolerance_t = USL - LSL,
  P0 = 0.9973002,
  LDL = LSL,
  UDL = USL,
  ...
)

gpci_fit_hybrid2(
  x = NULL,
  r = NULL,
  tc = NULL,
  n = NULL,
  distribution,
  USL,
  LSL,
  target = (USL + LSL)/2,
 indices = c("Cpy", "Spmk", "CpTk", "Cpc", "CNpmc", "CNpmkc", "CNpk", "Cp", "Cpk",
    "Cpm", "Cpmk"),
  u = 1,
  v = 1,
  mode = c("moments", "quantile"),
  fit = TRUE,
  start = NULL,
  C0 = 1,
  C1 = 0,
  C2 = 1,
  tolerance_t = USL - LSL,
  P0 = 0.9973002,
  LDL = LSL,
  UDL = USL,
  ...
)

capability(
```

```
  x = NULL,
  r = NULL,
  tc = NULL,
  n = NULL,
  distribution,
  USL,
  LSL,
  target = (USL + LSL)/2,
 indices = c("Cpy", "Spmk", "CpTk", "Cpc", "CNpmc", "CNpmkc", "CNpk", "Cp", "Cpk",
    "Cpm", "Cpmk"),
  u = 1,
  v = 1,
  mode = c("moments", "quantile"),
  fit = TRUE,
  start = NULL,
  C0 = 1,
  C1 = 0,
  C2 = 1,
  tolerance_t = USL - LSL,
  P0 = 0.9973002,
  LDL = LSL,
  UDL = USL,
  ...
)
```

Arguments

x	Numeric vector of observed failure times. Can be NULL (default) if distribution parameters are already fixed or estimated.
r	Positive integer target number of failures. Can be NULL (default) if not fitting raw data.
tc	Positive numeric fixed censoring time. Can be NULL (default) if not fitting raw data.
n	Positive integer total sample size placed on test. Can be NULL (default) if not fitting raw data.
distribution	A gpci_dist_hybrid2 distribution object.
USL	Numeric Upper Specification Limit. Must satisfy USL > LSL.
LSL	Numeric Lower Specification Limit. Must satisfy LSL < USL.
target	Numeric process target. Defaults to (USL + LSL) / 2.
indices	Character vector of capability indices to compute. Choices include: "Cpy", "Spmk", "CpTk", "Cpc", "CNpmc", "CNpmkc", "CNpk", "Cp", "Cpk", "Cpu", "Cpl", "Cpm", "Cpmk", "Cp_uv", "Cp_q", "Cpk_q", "Cpu_q", "Cpl_q", "Cpm_q", "Cpmk_q", "CNp_uv".
u	Non-negative numeric weight parameter u for the generalized Cp(u, v) family. Defaults to 1.

v	Non-negative numeric weight parameter v for the generalized Cp(u, v) family. Defaults to 1.
mode	Character string specifying mode of computation: "moments" (default; uses mean and variance) or "quantile" (uses robust quantiles).
fit	Logical scalar. If TRUE (default) and x is supplied, distribution parameters are estimated under Hybrid Type-II censoring. If FALSE, parameters in distribution are used directly without fitting.
start	Optional named list or numeric vector of starting parameter values for MLE. Defaults to NULL.
C0	Non-negative numeric coefficient for the tolerance cost function in CNpmc and CNpmkc. Defaults to 1.
C1	Non-negative numeric coefficient for the tolerance cost function in CNpmc and CNpmkc. Defaults to 0.
C2	Non-negative numeric coefficient for the tolerance cost function in CNpmc and CNpmkc. Defaults to 1.
tolerance_t	Positive numeric process tolerance t for the tolerance cost function. Defaults to USL - LSL.
P0	Desirable process yield 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.
...	Additional arguments passed to fit_distribution_hybrid2.

Details

Evaluates classical and generalized process capability indices for arbitrary continuous process distributions under Hybrid Type-II censoring. Supported indices include:

- C_{py}: Yield-based capability index (Maiti et al., 2010).
- S_{pmk}: Generalized loss-based index (Dey & Saha, 2019).
- C_{pTk}: Asymmetric target-based index (Saha et al., 2019).
- C_{pc}: Loss-based capability index (Saha et al., 2022).
- C_{Npmc}: Tolerance-loss based index (Alotaibi et al., 2022).
- C_{Npmkc}: Generalized asymmetric index (Saha et al., 2024).
- C_{Npk}: Non-normal capability index (Saha et al., 2018).
- $C_p, C_{pk}, C_{pu}, C_{pl}, C_{pm}, C_{pmk}$: Classical process capability indices.
- $C_p(u, v), C_{Np}(u, v)$: Vännman's generalized family of capability indices.
- Quantile-based capability indices ($C_{p,q}, C_{pk,q}, C_{pu,q}, C_{pl,q}, C_{pm,q}, C_{pmk,q}$).

Value

An S3 object of class "gpcifit_hybrid2" containing:

x	Observed failure times.
r	Target number of failures.
tc	Fixed censoring time.
n	Total sample size.
distribution	Fitted or supplied gpci_dist_hybrid2 distribution object.
USL	Upper Specification Limit.
LSL	Lower Specification Limit.
target	Target value.
indices	Character vector of requested capability index names.
estimates	Named numeric vector of capability index point estimates.
p_hat	Expected process non-conformance proportion.
mode	Computation mode used ("moments" or "quantile").
options	List of calculation options and parameters used.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x <- c(0.2, 0.5, 0.8, 1.1)
res <- capability_hybrid2(
  x = x, r = 3, tc = 1.0, n = 10,
  distribution = dist_exp,
  USL = 3, LSL = 0, target = 1.5,
  indices = c("Cpy", "Spmk", "CpTk", "Cpc", "CNpmc", "CNpmkc", "Cp", "Cpk", "Cpm", "Cpmk")
)
print(res)
```

---

coef.gpcifit_hybrid2 *Coef Method for gpcifit_hybrid2*

Description

Extract point estimates of capability indices or fitted distribution parameters.

Usage

```
## S3 method for class 'gpcifit_hybrid2'
coef(object, what = c("indices", "parameters"), ...)
```

Arguments

object	An object of class gpcifit_hybrid2.
what	Character string: "indices" (default) or "parameters".
...	Additional arguments.

Value

Named numeric vector of indices or parameters.

`compute_theoretical_moments_hybrid2`
Compute Theoretical Moments for a Process Distribution

Description

Compute Theoretical Moments for a Process Distribution

Usage

```
compute_theoretical_moments_hybrid2(distribution)
```

Arguments

`distribution`	A `gpci_dist_hybrid2` distribution object.

Value

A list containing `mean`, `var`, `skewness`, and `kurtosis`.

Examples

```
dist_norm <- dist_normal(mean = 10, sd = 2)
compute_theoretical_moments_hybrid2(dist_norm)
```

`define_distribution_hybrid2`
Define a Process Distribution for Hybrid Type-II Censored Data

---

Description

Constructor to define a continuous or discrete probability distribution for process capability analysis under Hybrid Type-II censored data. The distribution can be defined by specifying any one (or more) of its Probability Density/Mass Function (PDF/PMF), Cumulative Distribution Function (CDF), Survival Function (SF), or Quantile Function. Missing functions are automatically and numerically derived using adaptive integration and root-finding algorithms.

Usage

```
define_distribution_hybrid2(
  name,
  pdf = NULL,
  cdf = NULL,
  sf = NULL,
  quantile = NULL,
  params = list(),
  support = c(-Inf, Inf)
)

gpci_dist_hybrid2(
  name,
  pdf = NULL,
  cdf = NULL,
  sf = NULL,
  quantile = NULL,
  params = list(),
  support = c(-Inf, Inf)
)

define_distribution(
  name,
  pdf = NULL,
  cdf = NULL,
  sf = NULL,
  quantile = NULL,
  params = list(),
  support = c(-Inf, Inf)
)
```

Arguments

`name`	Character string naming the distribution (e.g. `"custom_weibull"`).
`pdf`	Optional function representing the probability density/mass function (PDF/PMF). Must be of the form `function(x, ...)`. Defaults to `NULL`.
`cdf`	Optional function representing the cumulative distribution function (CDF). Must be of the form `function(x, ...)`. Defaults to `NULL`.
`sf`	Optional function representing the survival function (SF = 1 - CDF). Must be of the form `function(x, ...)`. Defaults to `NULL`.
`quantile`	Optional function representing the quantile function. Must be of the form `function(p, ...)`. Defaults to `NULL`.
`params`	Named list of numeric parameter values for the distribution. Defaults to `list()`.
`support`	Numeric vector of length 2 defining the lower and upper support bounds. Defaults to `c(-Inf, Inf)`.

Value

An S3 object of class "gpci_dist_hybrid2" containing:

name	Character string of the distribution name.
pdf	The probability density function.
cdf	The cumulative distribution function.
sf	The survival function.
quantile	The quantile function.
params	Named list of numeric parameter values.
param_names	Character vector of parameter names.
support	Numeric vector of length 2 specifying support limits.

Examples

```
# Define custom Weibull distribution using only the survival function
custom_weib <- define_distribution_hybrid2(
  name = "custom_weibull",
  sf = function(x, shape, scale) stats::pweibull(x, shape, scale, lower.tail = FALSE),
  params = list(shape = 2, scale = 10),
  support = c(0, Inf)
)
print(custom_weib)
```

---

dist_exponential *Exponential Distribution Constructor*

Description

Exponential Distribution Constructor

Usage

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

Arguments

rate	Rate parameter. Defaults to 1.

Value

A gpci_dist_hybrid2 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. Defaults to 1.
scale	Scale parameter. Defaults to 1.

Value

A gpci_dist_hybrid2 object for Gamma distribution.

Examples

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

dist_logistic *Logistic Distribution Constructor*

Description

Logistic Distribution Constructor

Usage

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

Arguments

location	Location parameter. Defaults to 0.
scale	Scale parameter. Defaults to 1.

Value

A gpci_dist_hybrid2 object for Logistic distribution.

Examples

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

dist_loglogistic *Log-Logistic Distribution Constructor*

Description

Log-Logistic Distribution Constructor

Usage

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

Arguments

shape	Shape parameter. Defaults to 1.
scale	Scale parameter. Defaults to 1.

Value

A gpci_dist_hybrid2 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 of the logarithm. Defaults to 0.
sdlog	Standard deviation of the logarithm. Defaults to 1.

Value

A gpci_dist_hybrid2 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. Defaults to 0.
sd	Standard deviation parameter. Defaults to 1.

Value

A gpci_dist_hybrid2 object for Normal distribution.

Examples

```
dist_norm <- 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. Defaults to 1.
scale	Scale parameter. Defaults to 1.

Value

A gpci_dist_hybrid2 object for Weibull distribution.

Examples

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

`fit_distribution_hybrid2` *Parameter Estimation for Hybrid Type-II Censored Data using MleCensoR*

Description

Fits process distribution parameters under Hybrid Type-II censored lifetime data using the `MleCensoR` package (`MleCensoR::mle_hybrid_type2`) with robust numerical optimization fallbacks.

Usage

```
fit_distribution_hybrid2(
  x,
  r,
  tc,
  n,
  distribution,
  start = NULL,
  method = NULL,
  lower = NULL,
  upper = NULL,
  ...
)

gpci_fit_hybrid2_dist(
  x,
  r,
  tc,
  n,
  distribution,
  start = NULL,
  method = NULL,
  lower = NULL,
  upper = NULL,
  ...
)

fit_hybrid2(
  x,
  r,
  tc,
  n,
  distribution,
  start = NULL,
  method = NULL,
  lower = NULL,
```

```
  upper = NULL,
  ...
)
```

Arguments

x	Numeric vector of observed failure times (sorted in ascending order).
r	Positive integer scalar specifying target number of failures.
tc	Positive numeric scalar specifying fixed censoring time.
n	Positive integer scalar specifying total initial sample size placed on test.
distribution	A gpci_dist_hybrid2 distribution object.
start	Optional named numeric vector or list of starting parameter values. If NULL, uses distribution$params.
method	Character string specifying optimization method (e.g. "L-BFGS-B", "BFGS", "Nelder-Mead").
lower	Optional numeric vector of lower bounds for parameters. Defaults to NULL.
upper	Optional numeric vector of upper bounds for parameters. Defaults to NULL.
...	Additional arguments passed to MleCensoR::mle_hybrid_type2 or stats::optim.

Details

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

$$\ell(\theta) = \sum_{i=1}^{d} \ln f(x_i;\theta) + (n-d)\ln S(T^*;\theta)$$

where $d = \text{length}(x)$ is the number of observed failures up to time T^*, and $S(t;\theta) = 1 - F(t;\theta)$ is the survival function.

Value

A fitted gpci_dist_hybrid2 distribution object with updated parameter estimates and attributes:

vcov	Asymptotic variance-covariance matrix of estimated parameters.
se	Numeric vector of standard errors for estimated parameters.
std_errs	Alias for se.
loglik	Scalar log-likelihood value at the optimum.
vdata	Validated Hybrid Type-II censoring data summary list.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x <- c(0.2, 0.5, 0.8, 1.1)
fit <- fit_distribution_hybrid2(x = x, r = 3, tc = 1.0, n = 10, distribution = dist_exp)
print(fit)
```

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 gpcihybridIILinApp or gpcifit_hybrid2 object. 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_hybrid2 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.

Details

Goodness-of-fit testing verifies whether the assumed continuous probability distribution fits the observed failure times under Hybrid Type-II censoring. The test utilizes the specialized framework of the 'gofPHCS' package.

Value

An S3 object of class "gpci_gof_hybrid2" containing:

fit	The original fit 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 gpci_dist_hybrid2 distribution object.

Examples

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

gpci_index_hybrid2 *Extract a Specific Capability Index*

Description

Extract a Specific Capability Index

Usage

```
gpci_index_hybrid2(fit, index)
```

Arguments

fit	A gpcifit_hybrid2 object.
index	Name of the capability index to retrieve or compute.

Value

Numeric scalar value of the index.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x <- c(0.2, 0.5, 0.8, 1.1)
fit <- capability_hybrid2(x = x, r = 3, tc = 1.0, n = 10, distribution = dist_exp, USL = 3, LSL = 0)
gpci_index_hybrid2(fit, "Cpy")
```

gpci_lindley_hybrid2 *High-Level User Interface Function for Hybrid Type-II Lindley Approximation GPCI Analysis*

Description

Main user-facing function allowing input of custom PDF, CDF, and Survival functions, prior distribution hyperparameters, chain length, burn-in, and thinning under Hybrid Type-II censoring.

Usage

```
gpci_lindley_hybrid2(
  x,
  r,
  tc,
  n,
  pdf = NULL,
  cdf = NULL,
  sf = NULL,
```

```
  quantile = NULL,
  prior = NULL,
  chain_length = 1000,
  burn_in = 200,
  thinning = 1,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  B = 500,
  ...
)

gpcihybridIILinApp(
  x,
  r,
  tc,
  n,
  pdf = NULL,
  cdf = NULL,
  sf = NULL,
  quantile = NULL,
  prior = NULL,
  chain_length = 1000,
  burn_in = 200,
  thinning = 1,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  B = 500,
  ...
)
```

Arguments

x	Numeric vector of observed failure times.
r	Positive integer target number of failures.
tc	Positive numeric fixed censoring time.
n	Positive integer total initial sample size placed on test.
pdf	Custom PDF function `function(x, ...)`.
cdf	Custom CDF function `function(x, ...)`.
sf	Custom Survival Function `function(x, ...)`.
quantile	Custom Quantile Function `function(p, ...)`.
prior	Prior distribution hyperparameters list or log-prior function.
chain_length	Desired MCMC chain length (default 1000).

burn_in	MCMC burn-in iterations (default 200).
thinning	Thinning interval (default 1).
USL	Upper Specification Limit.
LSL	Lower Specification Limit.
target	Process target value (default midpoint of USL and LSL).
indices	Character vector of GPCI names to evaluate.
B	Number of bootstrap replications (default 500).
...	Additional arguments passed to fit_distribution_hybrid2.

Value

An object of S3 class "gpcihybridIILinApp" containing MCMC parameter chains, GPCI chains, MLE point estimates, Lindley Bayes point estimates, bootstrap confidence intervals, and analysis specifications.

Examples

```
my_pdf <- function(x, rate = 1) stats::dexp(x, rate = rate)
my_cdf <- function(x, rate = 1) stats::pexp(x, rate = rate)
my_sf  <- function(x, rate = 1) stats::pexp(x, rate = rate, lower.tail = FALSE)

fit_res <- gpci_lindley_hybrid2(
  x = c(0.2, 0.5, 0.8, 1.1),
  r = 3, tc = 1.0, n = 10,
  pdf = my_pdf, cdf = my_cdf, sf = my_sf,
  chain_length = 20, burn_in = 5,
  USL = 3, LSL = 0,
  indices = c("Cpy", "Cp"),
  B = 5
)
print(fit_res)
```

---

heidelberger_welch *Heidelberger and Welch's MCMC Convergence Diagnostic*

Description

Conducts stationarity and relative half-width tests under Heidelberger and Welch's MCMC convergence diagnostic, returning test statistics, status, and convergence probability.

Usage

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

Arguments

x	Numeric vector representing an MCMC / simulation 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).

Details

Implements the two-stage diagnostic procedure of Heidelberger & Welch (1983). First, the Cramér-von Mises statistic is computed on cumulative sums of the series using batch-means spectral density variance estimation at frequency zero. If the test fails, successive 10 achieved. Second, a relative half-width test is performed to verify whether the half-width of the sample mean confidence interval is smaller than `eps * |mean|`.

Value

A list containing Heidelberger and Welch convergence diagnostic results:

stat	Cramér-von Mises test statistic for stationarity.
pvalue	Stationarity test p-value.
passed	Logical flag indicating whether stationarity test passed.
hw_stat	Half-width test statistic (ratio of half-width to sample mean).
hw_passed	Logical flag indicating whether half-width test passed.
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 sample draws at specified confidence / credibility levels (e.g., 90

Usage

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

Arguments

x	Numeric vector of sample draws. Missing and non-finite values are automatically removed.
prob	Credibility level (1 - significance level). Default is 0.95.

Value

A named numeric vector of length 2 with elements `"lower"` and `"upper"` specifying the Highest Posterior Density (HPD) interval bounds at the requested credibility level.

Examples

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

---

lindley_approx_hybrid2

Lindley Approximation Method for Parameter and GPCI Bayesian Estimation under Hybrid Type-II Censoring

---

Description

Computes Bayesian posterior expectations of model parameters and Generalized Process Capability Indices (GPCIs) using Lindley's 3rd-order Taylor series approximation for Hybrid Type-II censored lifetime data.

Usage

```
lindley_approx_hybrid2(
  x,
  r,
  tc,
  n,
  distribution,
  prior = NULL,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  u = 1,
  v = 1,
  C0 = 1,
  C1 = 0,
  C2 = 1,
  P0 = 0.9973002,
  ...
)
```

Arguments

`x`	Numeric vector of observed failure times.
`r`	Positive integer target number of failures.

`tc`	Positive numeric fixed censoring time.
`n`	Positive integer initial total sample size placed on test.
`distribution`	A `gpci_dist_hybrid2` object.
`prior`	Log-prior density function `function(params)` or a hyperparameter list. Default assumes non-informative flat or conjugate log-priors.
`USL`	Numeric Upper Specification Limit.
`LSL`	Numeric Lower Specification Limit.
`target`	Numeric process target value. Default is `(USL + LSL) / 2`.
`indices`	Character vector of GPCI names to evaluate. Default evaluates all available indices.
`u`	Parameter `u` for Vännman's `Cp(u,v)` index (default 1).
`v`	Parameter `v` for Vännman's `Cp(u,v)` index (default 1).
`C0`, `C1`, `C2`	Parameters for tolerance loss function (defaults: 1, 0, 1).
`P0`	Baseline process yield (default 0.9973002).
`...`	Additional arguments passed to `fit_distribution_hybrid2`.

Details

Under Hybrid Type-II censoring with initial sample size n, target failures r, and censoring time T_c, the experiment terminates at $T^* = \max(x_r, T_c)$. The likelihood is:

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

where $d = \text{length}(x) \geq r$ is the number of observed failures.

Lindley (1980) proposed an asymptotic expansion for the posterior expectation of an arbitrary function $u(\theta)$:

$$E[u(\theta) \mid \text{data}] \approx u(\hat{\theta}) + \frac{1}{2}\sum_{i,j}(u_{ij} + 2u_i\rho_j)\sigma_{ij} + \frac{1}{2}\sum_{i,j,k,l} L_{ijk}\sigma_{ij}\sigma_{kl}u_l$$

where $\hat{\theta}$ is the MLE under Hybrid Type-II censoring, σ_{ij} are elements of the asymptotic covariance matrix $\Sigma = [-\nabla^2\ell(\hat{\theta})]^{-1}$, $\rho_j = \partial \ln \pi(\theta)/\partial\theta_j$ is the log-prior gradient at $\hat{\theta}$, and $L_{ijk} = \partial^3\ell/\partial\theta_i\partial\theta_j\partial\theta_k$ are third-order log-likelihood derivatives evaluated at $\hat{\theta}$.

Value

An object of S3 class `"gpciLindleyHybrid2"` containing a list with:

`mle_params`	A named list of Maximum Likelihood Estimates (MLE) of parameters under Hybrid Type-II censoring.
`lindley_params`	A named list of Lindley Bayes point estimates of parameters.
`mle_gpci`	A named numeric vector of GPCI point estimates computed at parameter MLEs.
`lindley_gpci`	A named numeric vector of Lindley Bayes point estimates of evaluated GPCIs.

cov_matrix	Estimated parameter variance-covariance matrix derived from observed Hessian.
hessian	Observed Hessian matrix of negative log-likelihood.
specs	A list containing input data summary, distribution object, and specification limits.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x_data <- c(0.2, 0.5, 0.8, 1.1)
fit_lnd <- lindley_approx_hybrid2(
  x = x_data, r = 3, tc = 1.0, n = 10,
  distribution = dist_exp, USL = 3, LSL = 0
)
print(fit_lnd)
```

---

lindley_gpci_hybrid2 *Lindley Approximation MCMC-Style Chain Generator for GPCIs under Hybrid Type-II Censoring*

Description

Generates posterior parameter draws, GPCI chains, point estimates, HPD intervals, bootstrap confidence intervals, and convergence diagnostics for Hybrid Type-II censored data using the Lindley approximation method combined with sampling, burn-in, and thinning.

Usage

```
lindley_gpci_hybrid2(
  x,
  r,
  tc,
  n,
  distribution = NULL,
  prior = NULL,
  chain_length = 1000,
  burn_in = 200,
  thinning = 1,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  u = 1,
  v = 1,
  C0 = 1,
  C1 = 0,
  C2 = 1,
```

```
  P0 = 0.9973002,
  B = 500,
  ...
)
```

Arguments

x	Numeric vector of observed failure times.
r	Positive integer target number of failures.
tc	Positive numeric fixed censoring time.
n	Positive integer total initial sample size placed on test.
distribution	A gpci_dist_hybrid2 object. Defaults to dist_normal().
prior	Log-prior density function function(params) or hyperparameter list.
chain_length	Desired length of final retained chain after burn-in and thinning. Default is 1000.
burn_in	Number of initial burn-in samples to discard. Default is 200.
thinning	Thinning interval. Default is 1 (no thinning).
USL	Upper Specification Limit.
LSL	Lower Specification Limit.
target	Process target value (defaults to midpoint of USL and LSL).
indices	Character vector of GPCI names to evaluate. Default evaluates all available indices.
u, v	Parameters for Vännman's Cp(u,v) family (default 1).
C0, C1, C2	Parameters for loss function (defaults: 1, 0, 1).
P0	Baseline process yield (default 0.9973002).
B	Number of bootstrap replications for bootstrap CIs. Default is 500.
...	Additional arguments passed to fit_distribution_hybrid2.

Details

Under Hybrid Type-II censoring with sample size n, target failures r, and censoring time T_c, the experiment terminates at $T^* = \max(x_r, T_c)$.

The function first computes Maximum Likelihood Estimates (MLE) and Lindley Bayes point estimates of model parameters and GPCIs using 3rd-order Taylor series approximations. Next, parameter draws are generated centered at the Lindley Bayes posterior mode using the asymptotic covariance matrix. Burn-in and thinning are applied to extract the retained parameter chain, which is then mapped through all requested capability indices to generate the posterior GPCI chain. Finally, parametric or non-parametric bootstrap resampling is executed to provide robust confidence intervals.

Value

An object of S3 class "gpcihybridIILinApp" containing a list with the following components:

- `param_chain` A matrix of retained posterior parameter draws after applying burn-in and thinning.
- `gpci_chain` A data frame of evaluated Generalized Process Capability Index (GPCI) values computed across the retained parameter chain.
- `point_estimates` A named numeric vector of Maximum Likelihood Estimates (MLE) of GPCIs.
- `lindley_estimates` A named numeric vector of Lindley Bayes point estimates of GPCIs.
- `lindley_fit` The underlying "gpciLindleyHybrid2" object returned by `lindley_approx_hybrid2`.
- `bootstrap_results` An object of class "gpciBootstrapHybrid2" containing bootstrap confidence interval estimates.
- `specs` A list containing analysis specifications including specification limits (USL, LSL, target), data summary, distribution object, chain length, burn-in, and thinning settings.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x_data <- c(0.2, 0.5, 0.8, 1.1)
fit_chain <- lindley_gpci_hybrid2(
  x = x_data, r = 3, tc = 1.0, n = 10,
  distribution = dist_exp,
  chain_length = 20,
  burn_in = 5,
  thinning = 1,
  USL = 3,
  LSL = 0,
  indices = c("Cpy", "Cp"),
  B = 5
)
print(fit_chain)
```

---

plot.gpciBootstrapHybrid2

Plot Method for gpciBootstrapHybrid2 Objects

Description

Visualizes bootstrap distributions and confidence limits for GPCIs.

Usage

```
## S3 method for class 'gpciBootstrapHybrid2'
plot(x, indices = NULL, ...)
```

Arguments

x	An object of class "gpciBootstrapHybrid2".
indices	Character vector of index names to plot. Defaults to all evaluated indices.
...	Additional graphical arguments.

Value

Invisible x.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x_data <- c(0.2, 0.5, 0.8, 1.1)
boot_res <- bootstrap_hybrid2_gpci(
  x = x_data, r = 3, tc = 1.0, n = 10,
  distribution = dist_exp, USL = 3, LSL = 0, indices = "Cpy", B = 5
)
plot(boot_res, indices = "Cpy")
```

---

plot.gpcihybridIILinApp

Plot Method for gpcihybridIILinApp Objects

Description

Visualizes posterior parameter and GPCI chains (trace plots, density plots) generated from Lindley approximation analysis under Hybrid Type-II censoring.

Usage

```
## S3 method for class 'gpcihybridIILinApp'
plot(x, which = c("both", "trace", "density"), indices = NULL, ...)
```

Arguments

x	An object of class "gpcihybridIILinApp".
which	Character string specifying plot type: "both" (default), "trace", or "density".
indices	Character vector of GPCI names to plot. If NULL (default), plots the first few available indices.
...	Additional graphical arguments passed to plot or density.

Value

Invisible x.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x_data <- c(0.2, 0.5, 0.8, 1.1)
fit_chain <- lindley_gpci_hybrid2(
  x = x_data, r = 3, tc = 1.0, n = 10,
  distribution = dist_exp,
  chain_length = 20, burn_in = 5,
  USL = 3, LSL = 0, indices = "Cpy", B = 5
)
plot(fit_chain, which = "trace", indices = "Cpy")
```

---

print.gpciBootstrapHybrid2 *Print Method for gpciBootstrapHybrid2 Objects*

Description

Print Method for gpciBootstrapHybrid2 Objects

Usage

```
## S3 method for class 'gpciBootstrapHybrid2'
print(x, ...)
```

Arguments

x	An object of class "gpciBootstrapHybrid2".
...	Additional arguments.

Value

Invisible x.

---

print.gpcifit_hybrid2 *Print Method for gpcifit_hybrid2*

Description

Print Method for gpcifit_hybrid2

Usage

```
## S3 method for class 'gpcifit_hybrid2'
print(x, ...)
```

Arguments

x	An object of class gpcifit_hybrid2.
...	Additional print arguments.

Value

Invisible x.

---

print.gpcihybridIILinApp

Print Method for gpcihybridIILinApp Objects

Description

Print Method for gpcihybridIILinApp Objects

Usage

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

Arguments

x	An object of class "gpcihybridIILinApp".
...	Additional arguments.

Value

Invisible x.

---

print.gpciLindleyHybrid2

Print Method for gpciLindleyHybrid2 Objects

Description

Print Method for gpciLindleyHybrid2 Objects

Usage

```
## S3 method for class 'gpciLindleyHybrid2'
print(x, ...)
```

Arguments

x	An object of class "gpciLindleyHybrid2".
...	Additional arguments.

Value

Invisible x.

---

print.gpci_dist_hybrid2

Print Method for gpci_dist_hybrid2

Description

Print Method for gpci_dist_hybrid2

Usage

```
## S3 method for class 'gpci_dist_hybrid2'
print(x, ...)
```

Arguments

x	An object of class gpci_dist_hybrid2.
...	Additional print arguments.

Value

Invisible x.

---

print.gpci_gof_hybrid2

Print Method for gpci_gof_hybrid2

Description

Print Method for gpci_gof_hybrid2

Usage

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

Arguments

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

Value

Invisible x.

---

summary.gpciBootstrapHybrid2 *Summary Method for gpciBootstrapHybrid2 Objects*

Description

Summary Method for gpciBootstrapHybrid2 Objects

Usage

```
## S3 method for class 'gpciBootstrapHybrid2'
summary(object, ...)
```

Arguments

object	An object of class "gpciBootstrapHybrid2".
...	Additional arguments.

Value

Invisible object.

---

summary.gpcifit_hybrid2 *Summary Method for gpcifit_hybrid2*

Description

Summary Method for gpcifit_hybrid2

Usage

```
## S3 method for class 'gpcifit_hybrid2'
summary(object, ...)
```

Arguments

object	An object of class gpcifit_hybrid2.
...	Additional arguments.

Value

Invisible object.

---

summary.gpcihybridIILinApp

Summary and Diagnostics for gpcihybridIILinApp Objects

Description

Computes point estimates, Lindley Bayes estimates, Bias, MSE, Risk values (Linex + SEL), HPD intervals at 90 convergence diagnostic, convergence probability, and Bootstrap confidence intervals under Hybrid Type-II censoring.

Usage

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

Arguments

object	An object of class "gpcihybridIILinApp".
...	Additional arguments.

Value

A data.frame containing detailed diagnostic metrics for each evaluated Generalized Process Capability Index (GPCI), including MLE estimates, Lindley Bayes estimates, MCMC chain means, bias, mean squared error (MSE), risk values (SEL + Linex), 90%, 95%, and 99% HPD credible intervals, Heidelberger-Welch stationarity test statistics and p-values, convergence probabilities, and 95% bootstrap percentile confidence intervals.

Examples

```
dist_exp <- dist_exponential(rate = 1)
x_data <- c(0.2, 0.5, 0.8, 1.1)
fit_chain <- lindley_gpci_hybrid2(
  x = x_data, r = 3, tc = 1.0, n = 10,
  distribution = dist_exp,
  USL = 3, LSL = 0, chain_length = 20, burn_in = 5,
  indices = c("Cpy", "Cp"), B = 5
)
summary(fit_chain)
```

summary.gpciLindleyHybrid2 *Summary Method for gpciLindleyHybrid2 Objects*

Description

Summary Method for gpciLindleyHybrid2 Objects

Usage

```
## S3 method for class 'gpciLindleyHybrid2'
summary(object, ...)
```

Arguments

object	An object of class "gpciLindleyHybrid2".
...	Additional arguments.

Value

Invisible object.

summary.gpci_dist_hybrid2 *Summary Method for gpci_dist_hybrid2*

Description

Summary Method for gpci_dist_hybrid2

Usage

```
## S3 method for class 'gpci_dist_hybrid2'
summary(object, ...)
```

Arguments

object	An object of class gpci_dist_hybrid2.
...	Additional arguments.

Value

Invisible object.

summary.gpci_gof_hybrid2 *Summary Method for gpci_gof_hybrid2*

Description

Summary Method for gpci_gof_hybrid2

Usage

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

Arguments

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

Value

Invisible object.

vcov.gpcifit_hybrid2 *Vcov Method for gpcifit_hybrid2*

Description

Extract asymptotic variance-covariance matrix of fitted distribution parameters.

Usage

```
## S3 method for class 'gpcifit_hybrid2'
vcov(object, ...)
```

Arguments

object	An object of class gpcifit_hybrid2.
...	Additional arguments.

Value

Variance-covariance matrix.

Index