Package 'gpciLindleyApprox'

August 21, 2026

Type Package

Title Lindley Approximation Method for Generalized Process Capability Indices

Version 0.1.0

Description Provides a comprehensive framework for estimating Generalized Process Capability Indices (GPCIs) using the Lindley approximation method for uncensored data under Bayesian inference. Evaluates point estimates and posterior expectations for classical and non-normal capability indices, including Cpy (Maiti et al., 2010), Spmk (Dey & Saha, 2019), CpTk (Saha et al., 2019), Cpc (Saha et al., 2022), CNpmc (Alotaibi et al., 2022), CNpmkc (Saha et al., 2024), CNpk (Saha et al., 2018), and Vannman's Cp(u,v) family. Computes parametric and non-parametric bootstrap confidence intervals at 90%, 95%, and 99% levels of significance. Supports MCMC chain generation with burn-in and thinning, Highest Posterior Density (HPD) intervals, Bias, MSE, Risk values, and Heidelberger and Welch's MCMC Convergence Diagnostic with convergence probabilities.
References:
Lindley (1980) <doi:10.2307/2345271>,
Maiti, Saha & Nanda (2010) <doi:10.1080/16843703.2010.11673233>,
Saha, Dey & Maiti (2018) <doi:10.1080/21681015.2018.1437793>,
Dey & Saha (2019) <doi:10.1007/s41872-019-00081-4>,
Saha, Dey & Maiti (2019),
Alotaibi, Dey & Saha (2022) <doi:10.1155/2022/3135264>,
Saha, Dey & Nadarajah (2022) <doi:10.1080/02664763.2021.1971632>,
Saha, Tripathi & Dey (2024) <doi:10.1142/S021853932450013X>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.0.0)

Imports stats, graphics, ggplot2, numDeriv, boot

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 12:50:25 UTC

Contents

`bootstrap_gpci` *Parametric and Non-Parametric Bootstrap Confidence Intervals for GPCIs*

Description

Computes bootstrap confidence intervals for Generalized Process Capability Indices (GPCIs) using parametric or non-parametric resampling. Supports Percentile, Normal, Basic, BCa, BCp, and Studentized bootstrap methods at 90%, 95%, and 99% confidence levels.

Usage

```
bootstrap_gpci(
  data,
  distribution = NULL,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  type = c("nonparametric", "parametric"),
  B = 1000,
  conf_levels = c(0.9, 0.95, 0.99),
  u = 1,
  v = 1
)
```

Arguments

`data`	Numeric vector of process observations.
`distribution`	A `gpci_dist` object.
`USL`	Upper Specification Limit.
`LSL`	Lower Specification Limit.
`target`	Process target value (default `(USL + LSL) / 2`).
`indices`	Character vector of GPCI names to evaluate.
`type`	Type of bootstrap: `"nonparametric"` (default) or `"parametric"`.
`B`	Number of bootstrap replications. Default is 1000.
`conf_levels`	Confidence levels for intervals (default `c(0.90, 0.95, 0.99)`).
`u, v`	Parameters for `Cp(u,v)` family (default 1).

Value

A list containing bootstrap point estimates, standard errors, and confidence intervals for each index.

Examples

```
data_x <- stats::rnorm(40, mean = 10, sd = 1)
fit_boot <- bootstrap_gpci(data = data_x, distribution = dist_normal(),
                           USL = 13, LSL = 7, B = 200)
```

`capability` *Compute Generalized Process Capability Indices (GPCIs)*

Description

Computes Generalized Process Capability Indices (GPCIs) for uncensored process data, given a `gpci_dist` object or data and specification limits.

Usage

```
capability(
  data = NULL,
  distribution = NULL,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  fit = TRUE,
  fit_method = "mle",
  u = 1,
  v = 1,
  C0 = 1,
  C1 = 0,
  C2 = 1,
  P0 = 0.9973002
)
```

Arguments

`data`	Numeric vector of uncensored process observations. Optional if `distribution` has parameters set.
`distribution`	A `gpci_dist` object with parameter values.
`USL`	Upper Specification Limit.
`LSL`	Lower Specification Limit.
`target`	Target process value. Default is `(USL + LSL) / 2`.
`indices`	Character vector of GPCI names to calculate. Default calculates all available indices.
`fit`	Logical. If `TRUE`, fits the distribution to `data` first. Default `TRUE`.
`fit_method`	Method used for distribution fitting (default `"mle"`).
`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`	Parameter C0 for loss function (default 1).
`C1`	Parameter C1 for loss function (default 0).
`C2`	Parameter C2 for loss function (default 1).
`P0`	Baseline process yield (default 0.9973002).

Value

A named numeric vector of computed GPCI values.

Examples

```
data_x <- stats::rnorm(50, mean = 10, sd = 1)
dist_norm <- dist_normal()
capability(data = data_x, distribution = dist_norm, USL = 13, LSL = 7)
```

---

define_distribution *Define a Process Probability Distribution Object*

Description

Creates a `gpci_dist` object containing probability density function (PDF), cumulative distribution function (CDF), survival function (SF), and optional quantile function (QF).

Usage

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

Arguments

`name`	Character string specifying the name of the distribution.
`pdf`	Function representing probability density `function(x, ...)`.
`cdf`	Function representing cumulative distribution `function(x, ...)`.
`sf`	Function representing survival function `function(x, ...)`.
`qf`	Optional function representing quantile function `function(p, ...)`.
`support`	Numeric vector of length 2 giving lower and upper support bounds. Default `c(-Inf, Inf)`.
`params`	List of initial parameter values or parameter names.

Value

An object of class `"gpci_dist"`.

Examples

```
dist_norm <- define_distribution(
  name = "Normal",
  pdf = function(x, mean = 0, sd = 1) stats::dnorm(x, mean, sd),
  cdf = function(x, mean = 0, sd = 1) stats::pnorm(x, mean, sd),
  sf  = function(x, mean = 0, sd = 1) 1 - stats::pnorm(x, mean, sd),
  params = list(mean = 0, sd = 1)
)
```

---

dist_exponentiated_exponential

Built-in Exponentiated-Exponential Distribution for GPCI

Description

Built-in Exponentiated-Exponential Distribution for GPCI

Usage

```
dist_exponentiated_exponential()
```

Value

A `gpci_dist` object for Exponentiated Exponential distribution (Saha et al. 2022).

---

dist_gamma

Built-in Gamma Distribution for GPCI

Description

Built-in Gamma Distribution for GPCI

Usage

```
dist_gamma()
```

Value

A `gpci_dist` object for Gamma distribution.

dist_logistic_exponential *Built-in Logistic-Exponential Distribution for GPCI*

Description

Built-in Logistic-Exponential Distribution for GPCI

Usage

```
dist_logistic_exponential()
```

Value

A `gpci_dist` object for Logistic-Exponential distribution (Lan & Leemis 2008, Alotaibi et al. 2022).

dist_normal *Built-in Normal Distribution for GPCI*

Description

Built-in Normal Distribution for GPCI

Usage

```
dist_normal()
```

Value

A `gpci_dist` object for Normal distribution.

dist_weibull *Built-in Weibull Distribution for GPCI*

Description

Built-in Weibull Distribution for GPCI

Usage

```
dist_weibull()
```

Value

A `gpci_dist` object for Weibull distribution.

fit_distribution *Fit Probability Distribution to Uncensored Data*

Description

Fits a gpci_dist object to uncensored process data using Maximum Likelihood Estimation (MLE).

Usage

```
fit_distribution(data, distribution, method = "mle", init = NULL)
```

Arguments

data	Numeric vector of process observations.
distribution	A gpci_dist object.
method	Fitting method (default is "mle").
init	Optional initial parameter values list.

Value

An updated gpci_dist object containing estimated parameters.

Examples

```
data_x <- stats::rnorm(50, mean = 10, sd = 2)
dist_fitted <- fit_distribution(data_x, dist_normal())
```

gpci_lindley *High-Level User Interface Function for 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.

Usage

```
gpci_lindley(
  data,
  pdf = NULL,
  cdf = NULL,
  sf = NULL,
  prior = NULL,
  chain_length = 1000,
```

```
  burn_in = 200,
  thinning = 1,
  USL,
  LSL,
  target = (USL + LSL)/2,
  indices = NULL,
  B = 500
)
```

Arguments

`data`	Numeric vector of process observations.
`pdf`	Custom PDF function `function(x, ...)`.
`cdf`	Custom CDF function `function(x, ...)`.
`sf`	Custom Survival Function `function(x, ...)`.
`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.
`indices`	Character vector of GPCI names to evaluate.
`B`	Number of bootstrap replications (default 500).

Value

An object of class `"gpciLindleyApprox"`.

Examples

```
gpci_lindley(
  data = stats::rnorm(50, 10, 1),
  pdf = function(x, mean = 0, sd = 1) stats::dnorm(x, mean, sd),
  cdf = function(x, mean = 0, sd = 1) stats::pnorm(x, mean, sd),
  chain_length = 300, burn_in = 50,
  USL = 13, LSL = 7, B = 100
)
```

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).

Value

A list containing Cramér-von Mises statistic (stat), p-value (pvalue), stationarity test status (passed), half-width test status (hw_passed), half-width statistic (hw_stat), and convergence probability (convergence_prob).

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 / significance levels (e.g., 90%, 95%, 99%).

Usage

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

Arguments

x	Numeric vector of sample draws.
prob	Credibility level (1 - significance level). Default is 0.95.

Value

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

Examples

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

`lindley_approx` *Lindley Approximation Method for Parameter and GPCI Bayesian Point Estimation*

Description

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

Usage

```
lindley_approx(
  data,
  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

`data`	Numeric vector of uncensored process observations.
`distribution`	A `gpci_dist` object.
`prior`	Log-prior density function `function(params)` or a hyperparameter list. Default assumes non-informative flat or conjugate log-priors.
`USL`	Upper Specification Limit.
`LSL`	Lower Specification Limit.
`target`	Process target value. Default is `(USL + LSL) / 2`.
`indices`	Character vector of GPCI names to evaluate.

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.
P0	Baseline process yield (default 0.9973002).

Value

A list of class "gpciLindley" containing:

mle_params	Maximum Likelihood Estimates of parameters.
lindley_params	Lindley Bayes point estimates of parameters.
mle_gpci	GPCI point estimates computed at MLE.
lindley_gpci	Lindley Bayes point estimates of GPCIs.
cov_matrix	Estimated covariance matrix σ_{ij}.
hessian	Observed Hessian matrix.

Examples

```
data_x <- stats::rnorm(50, mean = 10, sd = 1)
dist_norm <- dist_normal()
fit_lindley <- lindley_approx(data = data_x, distribution = dist_norm, USL = 13, LSL = 7)
```

---

lindley_gpci *Lindley Approximation MCMC-Style Chain Generator for GPCIs*

Description

Generates posterior parameter draws, GPCI chains, point estimates, HPD intervals, bootstrap confidence intervals, and convergence diagnostics for uncensored data using the Lindley approximation method combined with sampling and thinning.

Usage

```
lindley_gpci(
  data,
  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

data	Numeric vector of uncensored process data.
distribution	A gpci_dist object.
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.

Value

An object of class "gpciLindleyApprox" containing:

param_chain	Matrix of retained parameter draws after burn-in and thinning.
gpci_chain	Data frame of retained GPCI chain values.
point_estimates	Point estimates of GPCIs under MLE.
lindley_estimates	Bayes point estimates of GPCIs under Lindley approximation.
bootstrap_results	Bootstrap confidence interval results.
specs	Specification limits and MCMC options.

Examples

```
data_x <- stats::rnorm(50, mean = 10, sd = 1)
dist_norm <- dist_normal()
fit_chain <- lindley_gpci(
  data = data_x,
  distribution = dist_norm,
  chain_length = 500,
  burn_in = 100,
  thinning = 1,
  USL = 13, LSL = 7, B = 200
)
```

---

plot.gpciLindleyApprox

Plot Visualizations for gpciLindleyApprox Objects

Description

Generates trace plots, posterior density distributions, and HPD / Bootstrap interval plots for Generalized Process Capability Indices (GPCIs) using ggplot2 or base R graphics.

Usage

```
## S3 method for class 'gpciLindleyApprox'
plot(x, index = NULL, type = c("trace", "density", "both"), ...)
```

Arguments

x	An object of class "gpciLindleyApprox".
index	Character string specifying which GPCI to plot. Default plots the first index.
type	Type of plot: "trace" for trace plot, "density" for posterior density, or "both".
...	Additional graphic parameters.

Value

A ggplot2 object or plot list.

Examples

```
data_x <- stats::rnorm(40, mean = 10, sd = 1)
fit_chain <- lindley_gpci(data = data_x, distribution = dist_normal(),
                          USL = 13, LSL = 7, chain_length = 200, burn_in = 50, B = 50)
plot(fit_chain, index = "Cpy")
```

summary.gpciLindleyApprox *Summary and Diagnostics for gpciLindleyApprox Objects*

Description

Computes point estimates, Lindley Bayes estimates, Bias, MSE, Risk values (Linex + SEL), HPD intervals at 90%, 95%, and 99% confidence levels, Heidelberger and Welch MCMC convergence diagnostic, convergence probability, and Bootstrap confidence intervals.

Usage

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

Arguments

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

Value

A data frame containing detailed diagnostic metrics for each evaluated GPCI.

Examples

```
data_x <- stats::rnorm(40, mean = 10, sd = 1)
fit_chain <- lindley_gpci(data = data_x, distribution = dist_normal(),
                          USL = 13, LSL = 7, chain_length = 300, burn_in = 50, B = 100)
summary(fit_chain)
```

Index