Package 'DPrivStats'

September 8, 2026

Type Package

Title Differentially Private Classical Statistical Inference

Version 0.1.0

Description Implements differentially private (DP) versions of common classical statistical procedures, including descriptive statistics (mean, variance, quantiles, histograms), hypothesis tests (t-test, chi-square, Kolmogorov-Smirnov, one-way ANOVA), and regression (closed-form DP linear regression and DP-SGD for generalized linear models). Provides Laplace and Gaussian mechanisms with analytic calibration, exponential mechanism for medians, privacy-aware confidence intervals that account for both sampling and privacy noise, and privacy budget accounting via basic, advanced, and Renyi differential privacy (RDP) composition. Designed for official statistics and privacy-preserving data analysis research.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.0.0)

Imports stats, MASS, Rcpp

Suggests testthat (>= 3.0.0), knitr, rmarkdown, ggplot2

LinkingTo Rcpp

VignetteBuilder knitr

Config/testthat/edition 3

URL https://github.com/MukulBijalwan/DPrivStats

BugReports https://github.com/MukulBijalwan/DPrivStats/issues

LazyData true

Config/roxygen2/version 8.1.0

NeedsCompilation yes

Author Mukul Bijalwan [aut, cre]

Maintainer Mukul Bijalwan <mukulbijalwan555@gmail.com>

Repository CRAN

Date/Publication 2026-09-08 13:30:29 UTC

Contents

`advanced_composition_epsilon` *Advanced composition*

Description

Computes the advanced-composition bound for k homogeneous releases:

$$\varepsilon' = \sqrt{2k \log(1/\delta')}\varepsilon + k\varepsilon(e^{\varepsilon} - 1).$$

Usage

```
advanced_composition_epsilon(prior_epsilons, new_epsilon, delta_prime = 1e-06)
```

Arguments

`prior_epsilons`	Numeric vector of epsilons already spent.
`new_epsilon`	Epsilon of the new release.
`delta_prime`	Target overall failure parameter δ' in (0,1).

Value

Total epsilon under advanced composition.

`analytic_gaussian_mechanism` *Analytic Gaussian mechanism*

Description

Releases `f(x)` plus Gaussian noise calibrated with the analytic Gaussian mechanism (Balle & Wang, 2018).

Usage

```
analytic_gaussian_mechanism(f, x, epsilon, delta, l2_sensitivity)
```

Arguments

f	Function computing the target statistic on data x.
x	The private dataset.
epsilon	Privacy parameter epsilon (> 0).
delta	Privacy parameter delta in (0, 1).
l2_sensitivity	L2 sensitivity of f.

Value

The privatized value of f(x) (scalar or vector).

---

analytic_gaussian_sigma

Analytic Gaussian calibration (Balle & Wang, 2018)

Description

Solves for the smallest sigma satisfying the analytic Gaussian mechanism privacy guarantee:

$$\Phi(\Delta/(2\sigma) - \epsilon\sigma/\Delta) - e^{\epsilon}\Phi(-\Delta/(2\sigma) - \epsilon\sigma/\Delta) \leq \delta$$

via bisection. Always at least as tight as the classic calibration.

Usage

```
analytic_gaussian_sigma(epsilon, delta, l2_sensitivity)
```

Arguments

epsilon	Privacy parameter epsilon (> 0).
delta	Privacy parameter delta in (0, 1).
l2_sensitivity	L2 sensitivity of f.

Value

Scalar noise standard deviation.

References

Balle, B., & Wang, Y. (2018). Improving the Gaussian Mechanism for Differential Privacy via Analytic Concentration of the Gaussian Distribution. *NeurIPS 2018*.

Examples

```
analytic_gaussian_sigma(1.0, 1e-6, 1.0)
gaussian_sigma(1.0, 1e-6, 1.0) # looser
```

basic_composition *Basic (sequential) composition*

Description

Basic (sequential) composition

Usage

```
basic_composition(epsilons, deltas = rep(0, length(epsilons)))
```

Arguments

epsilons	Numeric vector of per-release epsilons.
deltas	Numeric vector of per-release deltas.

Value

List with total epsilon and delta.

can_spend *Check whether a release fits in the remaining budget*

Description

Check whether a release fits in the remaining budget

Usage

```
can_spend(budget, epsilon, delta = 0)
```

Arguments

budget	A "privacy_budget" object.
epsilon	Proposed epsilon spend.
delta	Proposed delta spend (default 0).

Value

Logical; TRUE if the proposed release fits.

clip_gradients *Per-sample gradient clipping*

Description

Clips each row of a gradient matrix to a maximum L2 norm: $\tilde{g}_i = g_i \min(1, C/\|g_i\|_2)$. Used by DP-SGD.

Usage

```
clip_gradients(grads, max_grad_norm)
```

Arguments

grads	Numeric matrix (n x p) of per-sample gradients.
max_grad_norm	Positive clipping constant C.

Value

List with clipped (the clipped gradients) and norms (original norms).

Examples

```
g <- matrix(rnorm(30), 10, 3)
clip_gradients(g, 1)$norms
```

compare_composition *Compare composition rules over a workflow*

Description

Given a sequence of per-release epsilons, returns total privacy cost under basic, advanced, and RDP composition — useful for the research question of which rule dominates in practice.

Usage

```
compare_composition(epsilons, delta = 1e-06)
```

Arguments

epsilons	Numeric vector of per-release epsilons.
delta	Target delta in (0, 1).

Value

Data frame with one row per composition rule and total epsilon.

Examples

```
compare_composition(c(1, 0.5, 0.5, 0.25), 1e-6)
```

`compare_utility` *Compare utility of non-private and private fits*

Description

Fits `dp_lm` at each epsilon on an epsilon grid and reports MSE of coefficients relative to the ordinary least squares fit.

Usage

```
compare_utility(
  formula,
  data,
  epsilon_grid,
  delta = 1e-06,
  bounds = NULL,
  n_reps = 10
)
```

Arguments

`formula`	Model formula.
`data`	Data frame.
`epsilon_grid`	Numeric vector of epsilon values to compare.
`delta`	Delta for Gaussian mechanism.
`bounds`	Bounds list passed to `dp_lm`.
`n_reps`	Number of repetitions per epsilon (default 10).

Value

Data frame with columns `epsilon`, `rep`, `mse_coef`, `max_abs_error`.

Examples

```
set.seed(1)
d <- data.frame(x = rnorm(200)); d$y <- 1 + d$x + rnorm(200)
compare_utility(y ~ x, d, c(0.5, 2), 1e-6, list(y = c(-15, 15)),
                n_reps = 3)
```

confint.dp_lm *Confint interface for dp_lm objects*

Description

Convenience wrapper calling dp_confint with default settings.

Usage

```
## S3 method for class 'dp_lm'
confint(object, parm, level = 0.95, method = "analytical", B = 200, ...)
```

Arguments

object	A dp_lm object.
parm	Coefficient indices or names (default all).
level	Confidence level (default 0.95).
method	One of "analytical", "parametric_bootstrap", "privacy_aware_bootstrap".
B	Number of bootstrap replicates for bootstrap methods.
...	Additional arguments (ignored).

Value

Numeric matrix with columns estimate, lower, upper.

cpp_clip_gradients *Fast per-sample gradient clipping (C++)*

Description

Clips rows of a gradient matrix to a maximum L2 norm; equivalent to clip_gradients but implemented in C++.

Usage

```
cpp_clip_gradients(grads, max_grad_norm)
```

Arguments

grads	Numeric matrix of per-sample gradients.
max_grad_norm	Positive clipping constant.

Value

Numeric matrix of clipped gradients.

Examples

```
g <- matrix(rnorm(30), 10, 3)
cpp_clip_gradients(g, 1)
```

---

cpp_rlaplace *Fast Laplace sampling (C++)*

Description

Vectorized inverse-CDF Laplace sampler implemented in C++ via Rcpp. Equivalent to `rlaplace` but faster for large draws.

Usage

```
cpp_rlaplace(n, scale = 1)
```

Arguments

`n`	Number of draws.
`scale`	Scale parameter (positive).

Value

Numeric vector of length `n`.

Examples

```
cpp_rlaplace(5, 1.0)
```

---

dp_anova *DP one-way ANOVA F-test*

Description

Privatizes group counts, means and variances (equal budget splits) and computes the one-way ANOVA F statistic from privatized summaries only.

Usage

```
dp_anova(formula, data, epsilon, bounds, delta = NULL)
```

Arguments

formula	Formula of the form response ~ group.
data	Data frame.
epsilon	Total privacy parameter epsilon (> 0).
bounds	Common bounds c(L, U) assumed known for both samples.
delta	Optional Gaussian-mechanism delta; if NULL Laplace is used.

Value

An object of class c("dp_htest", "htest").

Examples

```
dat <- data.frame(y = c(rnorm(60), rnorm(60, 1)),
                  g = factor(rep(c("A", "B"), each = 60)))
dp_anova(y ~ g, dat, 2.0, bounds = c(-5, 5))
```

---

dp_chisq_test *DP chi-square test of independence*

Description

Privatizes each cell of a contingency table with Laplace noise (sensitivity 1 per cell), truncates negative counts (post-processing), then computes the standard chi-square test on the privatized table.

Usage

```
dp_chisq_test(table, epsilon)
```

Arguments

table	A numeric matrix or table of non-negative counts.
epsilon	Privacy parameter epsilon (> 0).

Value

An object of class c("dp_htest", "htest") wrapping a standard chi-square test on the privatized table.

Examples

```
tab <- matrix(c(30, 20, 10, 40), nrow = 2)
dp_chisq_test(tab, 1.0)
```

`dp_confint` *Privacy-aware confidence intervals*

Description

Confidence intervals for `dp_lm` coefficients that account for both sampling variability and privacy noise.

Usage

```
dp_confint(
  object,
  parm,
  level = 0.95,
  method = c("analytical", "parametric_bootstrap", "privacy_aware_bootstrap"),
  B = 200,
  ...
)
```

Arguments

`object`	A `dp_lm` object.
`parm`	Coefficient indices or names (default all).
`level`	Confidence level (default 0.95).
`method`	One of `"analytical"`, `"parametric_bootstrap"`, `"privacy_aware_bootstrap"`.
`B`	Number of bootstrap replicates for bootstrap methods.
`...`	Additional arguments (ignored).

Details

Three methods:

`"analytical"` Normal intervals using $\sqrt{\text{diag}(\Sigma_{\text{sampling}}) + v_{\text{privacy}}}$, where privacy noise variance is known exactly from the mechanism calibration.

`"parametric_bootstrap"` Simulates $\beta^* \sim N(\hat{\beta}_{\text{priv}}, \Sigma_{\text{sampling}})$ and adds fresh privacy noise; returns percentile intervals.

`"privacy_aware_bootstrap"` Resamples the data and refits the DP estimator on each resample with per-resample budget $\epsilon/\sqrt{B}$ (advanced composition heuristic); returns percentile intervals. Most conservative but computationally heavy.

Value

Numeric matrix of class `c("dp_confint", "matrix")` with columns `estimate`, `lower`, `upper`.

Examples

```
set.seed(11)
d <- data.frame(x = rnorm(300))
d$y <- 1 + 2 * d$x + rnorm(300)
fit <- dp_lm(y ~ x, d, epsilon = 2.0, delta = 1e-6,
             bounds = list(y = c(-20, 20)))
dp_confint(fit)
```

`dp_glm` *DP-SGD for generalized linear models*

Description

Fits a GLM with differentially private stochastic gradient descent: per-sample gradients are clipped to `max_grad_norm`, averaged, and Gaussian noise is added at each iteration. Privacy is accounted with the advanced-composition heuristic over `n_iter` iterations.

Usage

```
dp_glm(
  formula,
  data,
  family = stats::gaussian(),
  epsilon,
  delta,
  max_grad_norm = 1,
  n_iter = 1000,
  lr = 0.01,
  batch_size = NULL,
  bounds = NULL
)
```

Arguments

`formula`	Model formula.
`data`	Data frame.
`family`	A GLM family (e.g. `binomial()`).
`epsilon`	Privacy parameter epsilon (> 0).
`delta`	Privacy parameter delta in (0, 1); required.
`max_grad_norm`	Per-sample gradient clipping norm C (> 0).
`n_iter`	Number of SGD iterations.
`lr`	Learning rate.
`batch_size`	Mini-batch size (default full batch).
`bounds`	Named list: `bounds$y = c(L_y, U_y)` for the response (required); optional `bounds$x_norm` giving a priori bound C on row norms of the design matrix when the design itself is private.

Value

A list of class "dp_glm" with coefficients, epsilon, delta, and convergence details.

Examples

```
set.seed(12)
d <- data.frame(x = rnorm(400))
d$y <- rbinom(400, 1, plogis(0.5 * d$x))
fit <- dp_glm(y ~ x, d, binomial(), epsilon = 2.0, delta = 1e-6,
              bounds = list(y = c(0, 1)))
fit$coefficients
```

---

dp_histogram *DP histogram*

Description

Releases bin counts of a bounded variable using Laplace noise with sensitivity 1 per bin, followed by non-negativity post-processing and optional normalization to probabilities.

Usage

```
dp_histogram(x, epsilon, breaks = NULL, normalize = FALSE)
```

Arguments

x	Numeric vector (NAs ignored).
epsilon	Privacy parameter epsilon (> 0).
breaks	Numeric vector of cut points (monotone). Defaults to deciles of the observed data.
normalize	Logical; return relative frequencies instead of counts.

Value

An object of class dp_estimate with element estimate (named numeric vector of privatized bin counts/proportions).

Examples

```
set.seed(4)
fit <- dp_histogram(rnorm(500), 1.0, breaks = seq(-3, 3, by = 1))
fit$estimate
```

dp_ks_test *DP Kolmogorov-Smirnov test*

Description

Releases the ECDFs of two bounded samples on a common grid under DP (ECDF sensitivity $1/n$ per sample; budget split evenly between the two samples) and computes the KS statistic from the privatized ECDFs.

Usage

```
dp_ks_test(
  x,
  y,
  epsilon,
  bounds = c(min(c(x, y), na.rm = TRUE), max(c(x, y), na.rm = TRUE)),
  n_grid = 100
)
```

Arguments

x, y	Numeric vectors (NAs ignored).
epsilon	Total privacy parameter epsilon (> 0).
bounds	Common bounds c(L, U) assumed known for both samples.
n_grid	Number of grid points for ECDF evaluation (default 100).

Value

An object of class `c("dp_htest", "htest")`.

Examples

```
set.seed(8)
dp_ks_test(rnorm(150), rnorm(150, 1), 1.0, bounds = c(-6, 6))
```

dp_lm *Differentially private linear regression*

Description

Fits OLS on clipped data and releases the coefficient vector under the Gaussian mechanism, calibrated to the L2 sensitivity of $\hat{\beta} = (X^\top X)^{-1} X^\top y$ given response bounds `bounds$y` (and optionally a bound C on row norms of X; see `dp_lm_sensitivity`).

Usage

```
dp_lm(
  formula,
  data,
  epsilon,
  delta,
  bounds = NULL,
  assume_public_design = TRUE
)
```

Arguments

formula	Model formula.
data	Data frame.
epsilon	Privacy parameter epsilon (> 0).
delta	Privacy parameter delta in (0, 1); required.
bounds	Named list: bounds$y = c(L_y, U_y) for the response (required); optional bounds$x_norm giving a priori bound C on row norms of the design matrix when the design itself is private.
assume_public_design	Logical; see dp_lm_sensitivity.

Value

An object of class c("dp_lm", "lm") with privatized coefficients, residuals and fitted values computed from the private fit, plus elements epsilon, delta, noise_variance (privacy noise variance per coefficient), and sampling_vcov (the non-private sampling covariance estimate used by dp_confint).

Examples

```
set.seed(11)
d <- data.frame(x = rnorm(300), z = rnorm(300))
d$y <- 1 + 2 * d$x - d$z + rnorm(300)
fit <- dp_lm(y ~ x + z, d, epsilon = 2.0, delta = 1e-6,
             bounds = list(y = c(-30, 30)))
coef(fit)
```

---

dp_lm_sensitivity *Global L2 sensitivity of OLS coefficients*

Description

Under the assumption that the design matrix X is public/fixed and the response is bounded ($y_i \in [L_y, U_y]$), neighbouring datasets differ in one response value only. With $C = \max_i \|x_i\|_2$ and $D = U_y - L_y$,

$$\Delta_2(\hat{\beta}) = C \cdot D / \lambda_{\min}(X^\top X).$$

Usage

```
dp_lm_sensitivity(
  X,
  y_bounds,
  x_norm_bound = NULL,
  assume_public_design = TRUE
)
```

Arguments

X	Numeric model matrix (n x p).
y_bounds	Bounds c(L_y, U_y) on the response.
x_norm_bound	Optional bound $C = \max_i \|x_i\|_2$. If NULL it is computed from X (use this when X is public). When the design is private, supply an a priori bound instead.
assume_public_design	Logical; if TRUE (default) the sensitivity accounts only for changes in y.

Details

If the design matrix itself is bounded per-row by x_norm_bound and treated as private, a conservative multiplicative factor of 3 is applied to account for simultaneous changes in $X'X$ and $X'y$ (documented heuristic; see vignette "regression-guide" for discussion).

Value

Scalar L2 sensitivity of the OLS coefficient vector.

Examples

```
X <- cbind(1, 1:10)
dp_lm_sensitivity(X, y_bounds = c(0, 10))
```

---

dp_mean *DP mean*

Description

Differentially private sample mean of a bounded variable. Data are clipped to bounds before the statistic is computed.

Usage

```
dp_mean(
  x,
  epsilon,
  bounds = range(x, na.rm = TRUE),
  mechanism = c("laplace", "gaussian", "analytic_gaussian"),
  delta = NULL
)
```

Arguments

x	Numeric vector (NAs ignored).
epsilon	Privacy parameter epsilon (> 0).
bounds	Bounds c(L, U) assumed known; data are clipped to them. Defaults to the observed range (use a priori bounds in practice).
mechanism	One of "laplace", "gaussian", "analytic_gaussian".
delta	Required (in (0,1)) when a Gaussian mechanism is used.

Value

An object of class dp_estimate.

Examples

```
set.seed(1)
dp_mean(rnorm(500), 1.0, bounds = c(-5, 5))$estimate
```

---

dp_median *DP median via the exponential mechanism*

Description

Releases a differentially private estimate of the median using the exponential mechanism over a grid of candidate values, with score $s(t; x) = -|\#\{i : x_i \le t\}| - n/2|$ which has L1 sensitivity 1.

Usage

```
dp_median(x, epsilon, bounds, n_bins = 1000)
```

Arguments

x	Numeric vector (may contain NAs, ignored).
epsilon	Privacy parameter epsilon (> 0).
bounds	Bounds c(L, U) assumed known for the data.
n_bins	Number of candidate grid points (default 1000).

Value

A list of class dp_estimate with element estimate.

Examples

```
set.seed(42)
fit <- dp_median(rnorm(200), 1.0, bounds = c(-5, 5))
fit$estimate
```

dp_quantile *DP quantile via the exponential mechanism*

Description

Releases arbitrary quantiles `probs` of a bounded variable using the exponential mechanism with score based on distance from the target order statistic. Each quantile receives budget $\epsilon/|\text{probs}|$.

Usage

```
dp_quantile(x, epsilon, bounds, probs = 0.5, n_bins = 1000)
```

Arguments

`x`	Numeric vector (may contain NAs, ignored).
`epsilon`	Privacy parameter epsilon (> 0).
`bounds`	Bounds c(L, U) assumed known for the data.
`probs`	Numeric vector of probabilities in (0, 1).
`n_bins`	Number of candidate grid points (default 1000).

Value

An object of class `dp_estimate`; `estimate` is a named vector with one entry per probability in `probs`.

Examples

```
set.seed(3)
dp_quantile(rnorm(200), 1.0, bounds = c(-5, 5), probs = c(.25, .5, .75))
```

dp_t_test *DP two-sample t-test*

Description

Differentially private two-sample t-test. The privacy budget is split evenly across the DP group counts, means and variances; the test statistic is computed entirely by post-processing of privatized quantities.

Usage

```
dp_t_test(
  x,
  y,
  epsilon,
  bounds = c(min(c(x, y), na.rm = TRUE), max(c(x, y), na.rm = TRUE)),
  alternative = c("two.sided", "less", "greater"),
  delta = NULL
)
```

Arguments

x, y	Numeric vectors (NAs ignored).
epsilon	Total privacy parameter epsilon (> 0).
bounds	Common bounds c(L, U) assumed known for both samples.
alternative	Character, "two.sided", "less" or "greater".
delta	Optional Gaussian-mechanism delta; if NULL Laplace is used.

Value

An object of class c("dp_htest", "htest") with elements statistic, estimate, std.error, p.value, alternative, and epsilon.

Examples

```
set.seed(7)
dp_t_test(rnorm(200), rnorm(200, 0.5), 1.0, bounds = c(-5, 5))
```

---

dp_utility_diagnostics

Utility diagnostics for DP estimators

Description

Computes bias, mean squared error, RMSE, and (optionally) CI coverage of a DP estimator relative to its non-private counterpart or a known truth, across Monte Carlo replicates.

Usage

```
dp_utility_diagnostics(estimates, truth, ci_matrix = NULL)
```

Arguments

estimates	Numeric vector of estimates from repeated runs.
truth	Scalar true value (or the non-private estimate).
ci_matrix	Optional B x 2 matrix of interval endpoints (lower, upper) aligned with estimates, for coverage.

Value

List with bias, mse, rmse and optional coverage.

Examples

```
ests <- replicate(100, dp_mean(rnorm(200), 1.0, c(-5, 5))$estimate)
dp_utility_diagnostics(ests, 0)
```

---

dp_variance *DP variance*

Description

Differentially private variance of a bounded variable, released as the DP second moment minus the square of the DP mean under a split budget ($\epsilon/2$ each), with post-processing to ensure non-negativity.

Usage

```
dp_variance(
  x,
  epsilon,
  bounds = range(x, na.rm = TRUE),
  mechanism = c("laplace", "gaussian"),
  delta = NULL
)
```

Arguments

x	Numeric vector (NAs ignored).
epsilon	Privacy parameter epsilon (> 0).
bounds	Bounds c(L, U) assumed known; data are clipped to them. Defaults to the observed range (use a priori bounds in practice).
mechanism	One of "laplace", "gaussian", "analytic_gaussian".
delta	Privacy parameter for Gaussian mechanisms (if chosen).

Value

An object of class dp_estimate.

Examples

```
set.seed(2)
dp_variance(runif(300), 1.0, bounds = c(0, 1))$estimate
```

empirical_delta_from_losses *Empirical delta from simulated privacy losses*

Description

Estimates the delta needed so that $P(L > \epsilon) \leq \delta$ given a sample of simulated losses.

Usage

```
empirical_delta_from_losses(losses, epsilon)
```

Arguments

losses	Numeric vector of simulated privacy losses.
epsilon	Privacy parameter threshold.

Value

Empirical tail probability.

Examples

```
empirical_delta_from_losses(simulate_gaussian_losses(1000, 5), 2.0)
```

epsilon_to_rdp *Convert epsilon-DP to zCDP (RDP) parameter*

Description

An (epsilon, 0)-DP mechanism satisfies (epsilon^2 / 2)-zCDP (Bun & Steinke, 2016).

Usage

```
epsilon_to_rdp(epsilon)
```

Arguments

epsilon	Pure-DP epsilon.

Value

zCDP rho parameter.

example_microdata *Census-like example microdata*

Description

A synthetic census-style microdata set of 2000 individuals generated for use in examples and vignettes. Income is a linear function of education, age, hours worked, region, plus noise; it is not real personal data.

Usage

```
example_microdata
```

Format

A data frame with 2000 rows and 5 variables:

education Years of education (integer, 0-20).

age Age in years (integer, 18-80).

hours Weekly working hours (numeric, 0-80).

region Factor with levels North, South, East, West.

income Annual income (numeric, non-negative).

Source

Simulated; see `data-raw/generate_data.R`.

exponential_mechanism *Exponential mechanism sampler*

Description

Samples a candidate from `candidates` with probability proportional to $\exp(\epsilon\, s(c)/(2\Delta s))$, where Δs is the sensitivity of the score function.

Usage

```
exponential_mechanism(candidates, scores, epsilon, score_sensitivity = 1)
```

Arguments

`candidates`	Numeric vector of candidate outputs.
`scores`	Numeric vector of scores (higher is better).
`epsilon`	Privacy parameter epsilon (> 0).
`score_sensitivity`	L1 sensitivity of the score function.

Value

One sampled candidate.

Examples

```
exponential_mechanism(1:10, -(1:10 - 5)^2, 1.0, 1)
```

`gaussian_mechanism` *Gaussian mechanism*

Description

Releases `f(x)` plus Gaussian noise using the classic $\sigma = \Delta_2 \sqrt{2 \log(1.25/\delta)}/\epsilon$ calibration for (epsilon, delta)-DP.

Usage

```
gaussian_mechanism(f, x, epsilon, delta, l2_sensitivity)
```

Arguments

`f`	Function computing the target statistic on data `x`.
`x`	The private dataset.
`epsilon`	Privacy parameter epsilon (> 0).
`delta`	Privacy parameter delta in (0, 1).
`l2_sensitivity`	L2 sensitivity of `f`.

Value

The privatized value of `f(x)` (scalar or vector).

Examples

```
gaussian_mechanism(sum, c(1:100), 1.0, 1e-6, l2_sensitivity = 100)
```

gaussian_sigma *Classic Gaussian noise scale*

Description

Computes the standard deviation used by the classic Gaussian mechanism.

Usage

```
gaussian_sigma(epsilon, delta, l2_sensitivity)
```

Arguments

epsilon	Privacy parameter epsilon (> 0).
delta	Privacy parameter delta in (0, 1).
l2_sensitivity	L2 sensitivity of f.

Value

Scalar noise standard deviation.

laplace_mechanism *Laplace mechanism*

Description

Releases f(x) plus Laplace noise calibrated to the L1 sensitivity, providing pure epsilon-DP.

Usage

```
laplace_mechanism(f, x, epsilon, l1_sensitivity)
```

Arguments

f	Function computing the target statistic on data x.
x	The private dataset.
epsilon	Privacy parameter epsilon (> 0).
l1_sensitivity	L1 sensitivity of f.

Value

The privatized value of f(x) (scalar or vector).

Examples

```
laplace_mechanism(mean, c(1:100), 1.0, l1_sensitivity = 100 / 99)
```

laplace_plr_tail *Privacy loss random variables for the Laplace mechanism*

Description

The privacy loss of one Laplace release with scale s is $L = |d|/s$ with $d \sim \mathrm{Laplace}(0, s)$, i.e. $L \sim \mathrm{Exp}(\epsilon)$ when $s = \Delta_1/\epsilon$. These helpers expose its tail probabilities and quantiles for PLRV-based accounting and simulation studies.

Usage

```
laplace_plr_tail(t, epsilon)

laplace_plr_quantile(p, epsilon)
```

Arguments

t	Numeric vector of thresholds on the privacy loss.
epsilon	Privacy parameter (> 0).
p	Numeric vector of probabilities in [0, 1].

Value

Tail probabilities $P(L > t)$ (`laplace_plr_tail`) or quantiles at probabilities `p` (`laplace_plr_quantile`).

Examples

```
laplace_plr_tail(3, 1.0) # exp(-3)
```

new_privacy_budget *Create a privacy budget object*

Description

An S3 object tracking cumulative privacy loss across multiple DP releases under basic, advanced, or Renyi DP (RDP) composition.

Usage

```
new_privacy_budget(
  epsilon,
  delta = 1e-06,
  composition = c("basic", "advanced", "rdp")
)
```

Arguments

epsilon	Total epsilon available (> 0).
delta	Total delta available (default 1e-6; must be in (0,1)).
composition	One of "basic", "advanced", "rdp".

Value

An object of class "privacy_budget".

Examples

```
b <- new_privacy_budget(3.0, 1e-6)
b <- spend(b, 1.0, description = "DP mean")
b$epsilon_spent
```

---

print.dp_confint *Print method for dp_confint objects*

Description

Print method for dp_confint objects

Usage

```
## S3 method for class 'dp_confint'
print(x, digits = 4, ...)
```

Arguments

x	A dp_confint matrix.
digits	Number of digits to print.
...	Additional arguments (ignored).

Value

The object x, invisibly.

print.dp_estimate *Print method for dp_estimate objects*

Description

Print method for dp_estimate objects

Usage

```
## S3 method for class 'dp_estimate'
print(x, digits = 4, ...)
```

Arguments

x	A dp_estimate object.
digits	Number of digits to print.
...	Additional arguments (ignored).

Value

The object x, invisibly.

print.dp_glm *Print method for dp_glm objects*

Description

Print method for dp_glm objects

Usage

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

Arguments

x	A dp_glm object.
...	Additional arguments (ignored).

Value

The object x, invisibly.

print.dp_htest *Print method for dp_htest objects*

Description

Print method for dp_htest objects

Usage

```
## S3 method for class 'dp_htest'
print(x, digits = 4, ...)
```

Arguments

x	A dp_htest object.
digits	Number of digits to print.
...	Additional arguments (ignored).

Value

The object x, invisibly.

print.dp_lm *Print method for dp_lm objects*

Description

Print method for dp_lm objects

Usage

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

Arguments

x	A dp_lm object.
...	Additional arguments (ignored).

Value

The object x, invisibly.

print.privacy_budget *Print method for privacy_budget objects*

Description

Print method for privacy_budget objects

Usage

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

Arguments

x	A "privacy_budget" object.
...	Additional arguments (ignored).

Value

The object x, invisibly.

print.summary.dp_lm *Print method for summary.dp_lm objects*

Description

Print method for summary.dp_lm objects

Usage

```
## S3 method for class 'summary.dp_lm'
print(x, digits = 4, ...)
```

Arguments

x	A summary.dp_lm object.
digits	Number of digits to print.
...	Additional arguments (ignored).

Value

The object x, invisibly.

rdp_composition *RDP composition*

Description

Composes a sequence of pure-DP releases via the zCDP/RDP accumulator and converts the result back to (epsilon, delta)-DP. Typically much tighter than basic composition for many small releases.

Usage

```
rdp_composition(epsilons, delta = 1e-06)
```

Arguments

epsilons	Numeric vector of per-release epsilons.
delta	Target delta in (0, 1).

Value

List with elements rho and epsilon.

Examples

```
rdp_composition(rep(0.1, 20), 1e-6)$epsilon
basic_composition(rep(0.1, 20))$epsilon
```

rdp_to_epsilon *Convert zCDP (RDP) parameter to (epsilon, delta)-DP*

Description

Uses $\varepsilon = \rho + \sqrt{2\rho \log(1/\delta)}$.

Usage

```
rdp_to_epsilon(rho, delta)
```

Arguments

rho	zCDP parameter (>= 0).
delta	Target delta in (0, 1).

Value

Equivalent epsilon.

rlaplace *Sample Laplace noise*

Description

Draws `n` independent Laplace(0, scale) deviates using inverse-CDF sampling with base R's uniform generator.

Usage

```
rlaplace(n, location = 0, scale = 1)
```

Arguments

`n`	Number of draws.
`location`	Location parameter (default 0).
`scale`	Scale parameter; must be positive.

Value

Numeric vector of length `n`.

Examples

```
rlaplace(5, 0, 1)
```

simulate_data *Simulate synthetic microdata*

Description

Generates a census-like dataset with correlated predictors used in examples and vignettes: income as a function of education, age, and hours worked.

Usage

```
simulate_data(n = 1000)
```

Arguments

`n`	Sample size (default 1000).

Value

A data frame with columns `education` (years, 0-20), `age` (18-80), `hours` (weekly hours, 0-80), `income` (annual income, >= 0), and `region` (factor).

Examples

```
head(simulate_data(5))
```

`simulate_gaussian_losses` *Simulate privacy losses of Gaussian releases*

Description

Draws the realized privacy loss $L = z^2/(2\sigma^2) + ...$ approximated by $L = |z|/\sigma + 1/(2\sigma^2)$ for two neighbouring datasets under the Gaussian mechanism, useful for empirical delta estimation.

Usage

```
simulate_gaussian_losses(n, sigma)
```

Arguments

`n`	Number of draws.
`sigma`	Noise standard deviation.

Value

Numeric vector of simulated losses.

Examples

```
l <- simulate_gaussian_losses(1000, 5); mean(l)
```

`spend` *Spend privacy budget*

Description

Records one DP release against the budget and updates the cumulative privacy loss under the configured composition rule.

Usage

```
spend(budget, epsilon, delta = 0, description = "")
```

Arguments

`budget`	A `"privacy_budget"` object.
`epsilon`	Epsilon spent by this release (> 0).
`delta`	Delta spent by this release (default 0).
`description`	Optional text label.

Value

The updated "privacy_budget" object (invisibly also sets remaining_epsilon).

See Also

new_privacy_budget

---

stat_sensitivities *Sensitivity of common bounded-data statistics*

Description

Computes the L1/L2 sensitivities under bounded-DP (neighbouring datasets differ in one record, same size n) for a record $x_i \in [L, U]$.

Usage

```
stat_sensitivities(bounds, n)
```

Arguments

bounds	Numeric vector c(L, U) giving the data bounds.
n	Sample size (positive integer).

Value

A list with elements l1 and l2, each a named numeric vector of sensitivities for "mean", "sum", "count", "variance" and "ecdf".

Examples

```
sens <- stat_sensitivities(c(0, 500000), 1000)
sens$l1["mean"]
```

summary.dp_lm *Summarize a dp_lm fit*

Description

Prints privatized coefficients with analytical privacy-aware standard errors and confidence intervals.

Usage

```
## S3 method for class 'dp_lm'
summary(object, level = 0.95, ...)
```

Arguments

object	A dp_lm object.
level	Confidence level (default 0.95).
...	Additional arguments (ignored).

Value

Invisibly, the list with coefficients table.

validate_coverage *Validate confidence-interval coverage by Monte Carlo*

Description

Repeatedly simulates data, fits dp_lm, builds privacy-aware CIs with dp_confint, and reports empirical coverage and mean interval width against the true generating coefficients.

Usage

```
validate_coverage(
  formula,
  true_betas,
  sigma = 1,
  data_gen = NULL,
  n = 500,
  n_sims = 200,
  epsilon = 1,
  delta = 1e-06,
  y_bounds = c(-50, 50),
  confint_method = "analytical",
  level = 0.95
)
```

Arguments

formula	Model formula matching true_betas names.
true_betas	Named numeric vector of true coefficients.
sigma	Error standard deviation of the simulated model.
data_gen	Function returning a data frame with columns needed by formula; defaults to a Gaussian design built from n.
n	Sample size per replicate (used by the default generator).
n_sims	Number of Monte Carlo replicates.
epsilon	Privacy parameter epsilon.
delta	Privacy parameter delta.
y_bounds	Response bounds passed to dp_lm.
confint_method	Method passed to dp_confint.
level	Confidence level.

Value

List with coverage_rate, mean_ci_width, target_coverage, n_sims.

Examples

```
set.seed(1)
validate_coverage(y ~ x, c(`(Intercept)` = 1, x = 2), sigma = 1,
                  n = 200, n_sims = 30, epsilon = 3, delta = 1e-6,
                  y_bounds = c(-15, 15))
```

Index