Package 'adheaping'

July 24, 2026

Type Package

Title Characteristic-Function De-Heaping Density Estimation

Version 1.0.0

Description Tuning-free kernel density estimation for heaped and rounded data using a characteristic-function theory of heaping. Rounding to a grid is convolution with a box followed by lattice sampling, so the density is recovered by deconvolving the known box and tapering against a data-driven noise floor. Provides a box-deconvolution de-heaping estimator, a superposition variant, and a single combined estimator selected by a band-capacity gate; blind grid, heaped-fraction, and mixed-grain readers; and a spectral higher-order comb detector. Faithful base-R replicas of the Heitjan-Rubin multiple-imputation and measurement-error deconvolution methods are included for comparison, and the 'Kernelheaping' stochastic expectation-maximization estimator is used when installed.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.6)

Imports stats

Suggests Kernelheaping, foreign, testthat (>= 3.0.0), knitr, rmarkdown

URL https://github.com/mitch-thornton/kde-ad-heaping

BugReports https://github.com/mitch-thornton/kde-ad-heaping/issues

VignetteBuilder knitr

RoxygenNote 7.3.1

NeedsCompilation no

Author Mitchell A. Thornton [aut, cre]

Maintainer Mitchell A. Thornton <mitchat@sbcglobal.net>

Repository CRAN

Date/Publication 2026-07-24 09:30:19 UTC

Contents

---

adkde *Combined AD heaping-KDE: band-capacity gate selects de-heap / superposition / iterated.*

Description

The single tuning-free estimator. A band-capacity statistic (de-heaped in-band power in the outer quarter of the identifiable band) selects the de-heaping estimator when the density fits inside the band, the superposition when it exceeds the band, and the iterated superposition at the coarsest grids.

Usage

```
adkde(y, D, grid, rho_lo = 0.06, rho_hi = 0.14, seed = 20260627)
```

Arguments

`y`	heaped (rounded) sample.
`D`	grid width.
`grid`	evaluation grid (equally spaced).
`rho_lo, rho_hi`	band-capacity thresholds selecting the component.
`seed`	random seed passed to the superposition components.

Value

Density values on `grid`, with attributes `pick` (the selected component) and `rho` (the band-capacity statistic).

Examples

```
grid <- seq(-6, 6, length.out = 1024)
x <- ifelse(runif(2000) < 0.5, rnorm(2000, -1.2, 0.5), rnorm(2000, 1.2, 0.5))
f <- adkde(0.5 * round(x / 0.5), 0.5, grid)
attr(f, "pick")
```

---

bin_prob *Bin a sample to a probability vector on an equal grid*

Description

Bin a sample to a probability vector on an equal grid

Usage

```
bin_prob(x, grid)
```

Arguments

x	numeric sample.
grid	numeric vector of equally spaced cell centers.

Value

A list with elements p (cell probabilities) and dx (cell width).

Examples

```
grid <- seq(-5, 5, length.out = 256)
b <- bin_prob(rnorm(500), grid)
sum(b$p)
```

---

deconv_kde *Measurement-error deconvolution KDE (rounding as Uniform(-D/2, D/2) error).*

Description

Divides the empirical characteristic function by sinc(wD/2) inside the grid band with a fixed Silverman Gaussian cutoff and a ridge cap; the classical deconvolution baseline.

Usage

```
deconv_kde(y, D, grid)
```

Arguments

y	heaped (rounded) sample.
D	grid width.
grid	evaluation grid (equally spaced).

Value

Density values on grid.

Examples

```
grid <- seq(-6, 6, length.out = 1024)
f <- deconv_kde(0.5 * round(rnorm(2000) / 0.5), 0.5, grid)
```

---

deheap_kde *Tuning-free de-heaping density estimator (box deconvolution + residue-floor Wiener).*

Description

Rounding to grid D is convolution of the density with a width-D box followed by lattice sampling; in the identifiable band |w| < pi/D the empirical characteristic function is divided by sinc(wD/2) (de-Sheppard) and Wiener-tapered against the sampling floor 1/n, amplified by the deconvolution. No bandwidth is chosen by hand.

Usage

```
deheap_kde(y, D, grid)
```

Arguments

y	heaped (rounded) sample.
D	grid width.
grid	evaluation grid (equally spaced).

Value

Density values on grid.

Examples

```
grid <- seq(-6, 6, length.out = 1024)
x <- ifelse(runif(2000) < 0.5, rnorm(2000, -1.2, 0.5), rnorm(2000, 1.2, 0.5))
f <- deheap_kde(0.5 * round(x / 0.5), 0.5, grid)
```

heap_detect *Detect the rounding grid as a group-matched atom in the spectral basis*

Description

Higher-order comb detection: the symmetry defect of the shift-by-K quotient group on the covariance of the power spectrum (a fourth-order statistic) is minimized at the true replica period; the smallest period with low defect (the largest passing cyclic subgroup) is the detected grid.

Usage

```
heap_detect(y, span = NULL, Mgrid = 512, cand_K = c(8, 16, 24, 32, 48, 64, 96))
```

Arguments

y	heaped sample.
span	numeric length-2 range for the internal analysis grid (defaults from the data).
Mgrid	internal grid size (default 512).
cand_K	candidate replica periods in spectral bins.

Value

A list with D_hat (recovered grid, or NA), K_hat, detected (logical), and defects (symmetry defect per candidate period).

Examples

```
set.seed(1)
x <- ifelse(runif(4000) < 0.5, rnorm(4000, -1.2, 0.5), rnorm(4000, 1.2, 0.5))
heap_detect(0.8 * round(x / 0.8), span = c(-12.8, 12.8))$D_hat
```

heap_fraction *Estimate the heaped fraction from the comb tooth height*

Description

Under partial heaping the leading tooth has height about p^2 over the residue floor, so phat = sqrt(S_peak - floor).

Usage

```
heap_fraction(y, D, grid)
```

Arguments

y	heaped (rounded) sample.
D	grid width.
grid	evaluation grid (equally spaced).

Value

The estimated heaped fraction in [0, 1].

Examples

```
grid <- seq(-6, 6, length.out = 2048)
x <- rnorm(4000); heaped <- runif(4000) < 0.4
y <- x; y[heaped] <- round(x[heaped])
heap_fraction(y, 1.0, grid)
```

---

heap_grid *Recover the rounding grid from the leading spectral replica*

Description

The leading tooth of the power spectrum sits at 2*pi/D, so the grid is Dhat = 2*pi / w_peak.

Usage

```
heap_grid(y, grid, near = NULL)
```

Arguments

y	heaped (rounded) sample.
grid	evaluation grid (equally spaced).
near	optional grid hint; if supplied the tooth is located in a frequency window around 2*pi/near (a verification that the comb sits at the expected grid); if NULL the first strong tooth beyond the signal lobe is taken (fully blind).

Value

The recovered grid width Dhat (or NA if no tooth is found).

Examples

```
grid <- seq(-6, 6, length.out = 2048)
x <- ifelse(runif(4000) < 0.5, rnorm(4000, -1.2, 0.5), rnorm(4000, 1.2, 0.5))
heap_grid(0.5 * round(x / 0.5), grid, near = 0.5)
```

`heap_lattice` *Subgroup-lattice reader for mixed rounding grains (Moebius inversion over the divisor lattice).*

Description

Given integer-count data with a known base unit, reads the replica-center amplitudes at the divisor lattice of the base period and solves a nonnegative system for the grain weights and the unrounded share. A compact reader; see the Python reference for the hardened version.

Usage

```
heap_lattice(y, grains = c(1, 5, 10, 20), span = c(0, 720), Mgrid = 8192)
```

Arguments

`y`	integer-valued heaped sample.
`grains`	candidate grains (e.g. `c(1, 5, 10, 20)`).
`span`	analysis span; base unit assumed 1.
`Mgrid`	internal grid size.

Value

A list with `grains`, their `weights`, and the `unrounded` share.

Examples

```
set.seed(1)
y <- sample(c(5, 10, 15, 20, 7, 13), 500, replace = TRUE)
heap_lattice(y, grains = c(1, 5, 10, 20))
```

`heitjan_mi` *Heitjan-Rubin multiple-imputation de-heaping (faithful base-R replica).*

Description

A faithful replica of the multiple-imputation approach of Heitjan and Rubin (1990) for coarse/heaped data: a pilot density is fit to jitter-dequantized values; then, in each of `M` independent imputations, every heaped value is imputed within its rounding cell in proportion to the pilot density and a kernel density estimate is formed on the imputed data; the imputations are averaged. Unlike the stochastic-EM chain of Kernelheaping, the draws are independent multiple imputations, in the Heitjan-Rubin style. No standalone Heitjan KDE package exists, so this replica stands in for it; the manuscript states as much.

Usage

```
heitjan_mi(y, D, grid, M = 10, seed = 20260627)
```

Arguments

y	heaped (rounded) sample.
D	grid width.
grid	evaluation grid (equally spaced).
M	number of imputations (default 10).
seed	random seed.

Value

Density values on grid.

References

Heitjan, D. F. and Rubin, D. B. (1990) Inference from coarse data via multiple imputation with application to age heaping. JASA 85, 304-314.

Examples

```
grid <- seq(-6, 6, length.out = 512)
f <- heitjan_mi(1.0 * round(rnorm(1000)), 1.0, grid, M = 4)
```

---

naive_kde *Naive Gaussian kernel density estimate on a grid*

Description

Silverman-bandwidth Gaussian kernel density estimate, shown for reference. On heaped data it inherits the rounding comb.

Usage

```
naive_kde(x, grid)
```

Arguments

x	numeric sample.
grid	evaluation grid (equally spaced).

Value

Density values on grid.

Examples

```
grid <- seq(-6, 6, length.out = 512)
f <- naive_kde(rnorm(1000), grid)
```

---

sem_kde *SEM de-heaping via the real Kernelheaping package*

Description

Wrapper around the stochastic expectation-maximization estimator of Gross and Rendtel (the **Kernelheaping** package), interpolated to `grid`. Requires that package; returns `NULL` if it is not installed.

Usage

```
sem_kde(y, D, grid, burnin = 5, samples = 10)
```

Arguments

`y`	heaped (rounded) sample.
`D`	grid width.
`grid`	evaluation grid (equally spaced).
`burnin, samples`	SEM burn-in and sample counts passed to `Kernelheaping::dheaping`.

Value

Density values on `grid`, or `NULL` if **Kernelheaping** is absent.

Examples

```
grid <- seq(-6, 6, length.out = 512)
if (requireNamespace("Kernelheaping", quietly = TRUE))
  f <- sem_kde(0.5 * round(rnorm(800) / 0.5), 0.5, grid)
```

---

superpose_kde *Superposition de-heaping estimator*

Description

Smooth Bayesian-information-criterion Gaussian-mixture base (broad components) plus a band-limited de-heaped residual on the leftover mass; the base carries the non-identifiable band beyond `pi/D`.

Usage

```
superpose_kde(y, D, grid, seed = 20260627)
```

Arguments

y	heaped (rounded) sample.
D	grid width.
grid	evaluation grid (equally spaced).
seed	random seed for the jitter dequantization.

Value

Density values on grid.

Examples

```
grid <- seq(-6, 6, length.out = 1024)
x <- ifelse(runif(2000) < 0.5, rnorm(2000, -1.2, 0.5), rnorm(2000, 1.2, 0.5))
f <- superpose_kde(1.0 * round(x / 1.0), 1.0, grid)
```

Index