Package ‘DirStats’

August 26, 2020

Type Package

Title Nonparametric Methods for Directional Data

Version 0.1.6

Date 2020-08-25

Description Nonparametric kernel density estimation, bandwidth selection, and other utilities for analyzing directional data. Implements the estimator in Bai, Rao and Zhao (1987) <doi:10.1016/0047-259X(88)90113-3>, the cross-validation bandwidth selectors in Hall, Watson and Cabrera (1987) <doi:10.1093/biomet/74.4.751> and the plug-in bandwidth selectors in García-Portugués (2013) <doi:10.1214/13-ejs821>.

License GPL-3

LazyData true

Depends R (>= 3.6.0)

Imports movMF, rotasym

Suggests viridisLite

URL https://github.com/egarpor/DirStats

BugReports https://github.com/egarpor/DirStats

Encoding UTF-8

RoxygenNote 7.1.0

NeedsCompilation yes

Author Eduardo García-Portugués [aut, cre] (<https://orcid.org/0000-0002-9224-4111>)

Maintainer Eduardo García-Portugués <edgarcia@est-econ.uc3m.es>

Repository CRAN

Date/Publication 2020-08-26 13:20:03 UTC

R topics documented:

`DirStats-package` `DirStats` – *Nonparametric Methods for Directional Data*

Description

Nonparametric kernel density estimation, bandwidth selection, and other utilities for analyzing directional data. Implements the estimator in Bai, Rao and Zhao (1987) <doi:10.1016/0047-259X(88)90113-3>, the cross-validation bandwidth selectors in Hall, Watson and Cabrera (1987) <doi:10.1093/biomet/74.4.751> and the plug-in bandwidth selectors in García-Portugués (2013) <doi:10.1214/13-ejs821>.

Author(s)

Eduardo García-Portugués.

`bic_vmf_mix` *Fitting mixtures of von Mises–Fisher distributions*

Description

Fitting mixtures of von Mises–Fisher distributions by the Expectation-Maximization algorithm, with determination of the optimal number of mixture components.

Usage

```
bic_vmf_mix(data, M_bound = ceiling(log(nrow(data))), M_neig = 3,
  crit = "BIC", iterative = TRUE, plot_it = FALSE, verbose = FALSE,
  kappa_max = 250)
```

Arguments

data	directional data, a matrix of size c(n,q + 1).
M_bound	bound for the number of components in the mixtures. If it is not enough, the search for the mixture with minimum crit will continue from M_bound + 1 if iterative = TRUE. Defaults to ceiling(log(nrow(data))).
M_neig	number of neighbors explored around the optimal number of mixture components. Defaults to 3.
crit	information criterion employed, either "BIC" (default), "AICc" or "AIC".
iterative	keep exploring higher number of components if the optimum is attained at M_bound? Defaults to TRUE.
plot_it	display an informative plot on the optimization's grid search? Defaults to FALSE.
verbose	display fitting progress? Defaults to FALSE.
kappa_max	maximum value of allowed concentrations, to avoid numerical instabilities. Defaults to 250.

Details

See Algorithm 3 in García-Portugués (2013). The Expectation-Maximization fit is performed with movMF.

Value

A list with entries:

- best_fit: a list with estimated mixture parameters mu_hat, kappa_hat, and p_hat of the best-fitting mixture according to crit.
- fit_mixs: a list with of the fitted mixtures.
- BICs: a vector with the BICs (or other information criterion) of the fitted mixtures.

References

García-Portugués, E. (2013). Exact risk improvement of bandwidth selectors for kernel density estimation with directional data. *Electronic Journal of Statistics*, 7:1655–1685. https://doi.org/10.1214/13-ejs821

Hornik, K. and Grün, B. (2014). movMF: An R Package for Fitting Mixtures of von Mises–Fisher Distributions. *Journal of Statistical Software*, 58(10):1–31. https://doi.org/10.18637/jss.v058.i10

Examples

```
# Sample
q <- 2
n <- 300
set.seed(42)
samp <- rbind(rotasym::r_vMF(n = n / 3, mu = c(rep(0, q), 1), kappa = 5),
              rotasym::r_vMF(n = n / 3, mu = c(rep(0, q), -1), kappa = 5),
              rotasym::r_vMF(n = n / 3, mu = c(1, rep(0, q)), kappa = 5))
```

```
# Mixture fit
bic_vmf_mix(data = samp, plot_it = TRUE, verbose = TRUE)
```

---

bw_dir_cv *Cross-validation bandwidth selectors for directional data*

Description

Likelihood and least squares cross-validation bandwidth selectors for kernel density estimation with directional data.

Usage

```
bw_dir_lcv(data, h_grid = exp(seq(log(0.05), log(1.5), l = 100)), L = NULL,
  plot_it = FALSE, optim = TRUE, optim_par = 0.25, optim_lower = 0.06,
  optim_upper = 10)

bw_dir_lscv(data, h_grid = exp(seq(log(0.05), log(1.5), l = 100)),
  L = NULL, plot_it = FALSE, optim = TRUE, R_code = FALSE,
  optim_par = 0.25, optim_lower = 0.06, optim_upper = 10)
```

Arguments

data	directional data, a matrix of size `c(n,q + 1)`.
h_grid	vector of bandwidths for performing a grid search. Defaults to `exp(seq(log(0.05),log(1.5),l = 100))`.
L	kernel function. Set internally to `function(x) exp(-x)` (von Mises–Fisher kernel) if `NULL` (default).
plot_it	display an informative plot on the optimization's grid search? Defaults to `FALSE`.
optim	run an optimization? Defaults to `TRUE`. Otherwise, a grid search on `h` is done. Only effective if `L = NULL`.
optim_par, optim_lower, optim_upper	parameters passed to `par`, `lower`, and `upper` in `optim` when using the `"L-BFGS-B"` method. Default to `0.25`, `0.06` (to avoid numerical instabilities), and `10`.
R_code	use slower R code when `L = NULL`? Defaults to `FALSE`.

Details

`data` is not checked to have unit norm, so the user must be careful. When `L = NULL`, faster FORTRAN code is employed.

`bw_dir_lscv` employs Monte Carlo integration for $q > 2$, which results in a random output. Use `set.seed` before to avoid it.

Value

A list with entries:

- `h_opt`: cross-validation bandwidth.
- `h_grid`: `h_grid`, if used (otherwise `NULL`).
- `CV_opt`: minimum of the CV loss.
- `CV_grid`: value of the CV function at `h_grid`, if used (otherwise `NULL`).

Source

The function `bw_dir_lscv` employs Netlib's subroutine `ribesl` for evaluating the modified Bessel function of the first kind. The subroutine is based on a program by Sookne (1973) and was modified by W. J. Cody and L. Stoltz. An earlier version was published in Cody (1983).

References

Cody, W. J. (1983). Algorithm 597: Sequence of modified Bessel functions of the first kind. *ACM Transactions on Mathematical Software*, 9(2):242–245. https://doi.org/10.1145/357456.357462

Hall, P., Watson, G. S., and Cabrera, J. (1987). Kernel density estimation with spherical data. *Biometrika*, 74(4):751–762. https://doi.org/10.1093/biomet/74.4.751

Sookne, D. J. (1973). Bessel functions of real argument and integer order. *Journal of Research of the National Bureau of Standards*, 77B:125–132.

Examples

```
# Sample
n <- 25
q <- 2
set.seed(42)
samp <- rotasym::r_vMF(n = n, mu = c(1, rep(0, q)), kappa = 2)

# bw_dir_lcv
bw_dir_lcv(data = samp, optim = TRUE)$h_opt
bw_dir_lcv(data = samp, optim = FALSE, plot_it = TRUE)$h_opt
bw_dir_lcv(data = samp, L = function(x) exp(-x))$h_opt

# bw_dir_lscv
set.seed(42)
bw_dir_lscv(data = samp, optim = TRUE)$h_opt
bw_dir_lscv(data = samp, optim = FALSE, plot_it = TRUE)$h_opt
bw_dir_lscv(data = samp, optim = FALSE, R_code = TRUE)$h_opt
bw_dir_lscv(data = samp, L = function(x) exp(-x))$h_opt
```

bw_dir_pi *Plug-in bandwidth selectors for directional data*

Description

Plug-in bandwidth selectors for kernel density estimation with directional data, including Rule-Of-Thumb (ROT), Asymptotic MIxtures (AMI), and Exact MIxtures (EMI).

Usage

```
bw_dir_rot(data)

bw_dir_ami(data, fit_mix = NULL, L = NULL)

R_Psi_mixvmf(q, mu, kappa, p)

bw_dir_emi(data, fit_mix = NULL, optim = TRUE,
  h_grid = exp(seq(log(0.05), log(1.5), l = 100)), plot_it = TRUE,
  optim_par = 0.25, optim_lower = 0.06, optim_upper = 10)
```

Arguments

`data`	directional data, a matrix of size `c(n,q + 1)`.
`fit_mix`	output from `bic_vmf_mix`. Computed internally if `NULL` (default).
`L`	kernel function. Set internally to `function(x) exp(-x)` (von Mises–Fisher kernel) if `NULL` (default).
`q`	dimension of S^q, $q \geq 1$.
`mu, kappa, p`	mixture parameters. `mu` is the mean matrix of size `c(length(p),q + 1)`, `kappa` is vector of `length(p)` concentration parameters, and `p` is the vector of mixture proportions.
`optim`	run an optimization? Defaults to `TRUE`. Otherwise, a grid search on `h` is done. Only effective if `L = NULL`.
`h_grid`	vector of bandwidths for performing a grid search. Defaults to `exp(seq(log(0.05),log(1.5),l = 100))`.
`plot_it`	display an informative plot on the optimization's grid search? Defaults to `FALSE`.
`optim_par, optim_lower, optim_upper`	parameters passed to `par`, `lower`, and `upper` in `optim` when using the `"L-BFGS-B"` method. Default to `0.25`, `0.06` (to avoid numerical instabilities), and `10`.

Details

See Algorithms 1 (AMI) and 2 (EMI) in García-Portugués (2013). The ROT selector is implemented according to Proposition 2, **but** without the paper's typo in equation (6), case $q = 2$, where an incorrect extra $\hat{\kappa}$ appears premultiplying $(1 + 4\hat{\kappa}^2)\sinh(2\hat{\kappa})$ in the denominator.

bw_dir_ami uses R_Psi_mixvmf for computing the curvature term of a mixture of von Mises–Fisher densities.

bw_dir_emi employs Monte Carlo integration for $q > 2$, which results in a random output. Use set.seed before to avoid it.

Value

Selected bandwidth for bw_dir_rot and bw_dir_ami. bw_dir_emi returns a list with entries:

- h_opt: cross-validation bandwidth.
- h_grid: h_grid, if used (otherwise NULL).
- MISE_opt: minimum of the MISE loss.
- MISE_grid: value of the MISE function at h_grid, if used (otherwise NULL).

References

García-Portugués, E. (2013). Exact risk improvement of bandwidth selectors for kernel density estimation with directional data. *Electronic Journal of Statistics*, 7:1655–1685. https://doi.org/10.1214/13-ejs821

Examples

```
# Sample
n <- 25
q <- 2
set.seed(42)
samp <- rotasym::r_vMF(n = n, mu = c(1, rep(0, q)), kappa = 2)

# Mixture fit
fit_mix <- bic_vmf_mix(data = samp, plot_it = TRUE)

# ROT
bw_dir_rot(samp)

# AMI
bw_dir_ami(samp)
bw_dir_ami(samp, fit_mix = fit_mix)
bw_dir_ami(samp, fit_mix = fit_mix, L = function(x) exp(-x))

# EMI
bw_dir_emi(samp)
bw_dir_emi(samp, fit_mix = fit_mix, optim = FALSE, plot_it = TRUE)
```

conv *Convenience functions*

Description

Normalization of data in R^{q+1} to S^q. Transformations between S^1 and $[0, 2\pi)$, and between S^2 and $[0, 2\pi) \times [0, \pi]$.

Usage

```
norm2(x)

normalize(x)

to_cir(th)

to_rad(x)

to_sph(th, ph)
```

Arguments

`x`	matrix or vector, in S^1 for `to_cir`.
`th`	vector of angles in $[0, 2\pi)$.
`ph`	vector of angles in $[0, \pi]$.

Value

Euclidean norm (`norm`) and normalized data (`normalize`). Position in S^1 (`to_cir`) or in $[0, 2\pi)$ (`to_rad`). Position in S^2 (`to_sph`) or in $[0, 2\pi) \times [0, \pi]$ (`to_rad`).

Examples

```
# Normalization
x <- 1:3
norm2(x)
normalize(x)
x <- rbind(1:3, 3:1)
norm2(x)
normalize(x)

# Circular transformations
th <- 1
x <- c(0, 1)
to_rad(to_cir(th))
to_rad(to_cir(c(th, th + 1)))
to_cir(to_rad(x))
to_cir(to_rad(rbind(x, -x)))
```

```
# Spherical transformations
th <- 2
ph <- 1
x <- c(0, 1, 0)
to_rad(to_sph(th, ph))
to_rad(to_sph(c(th, th + 1),
              c(ph, ph + 1)))
to_sph(to_rad(x)[, 1], to_rad(x)[, 2])
to_sph(to_rad(rbind(x, -x))[, 1], to_rad(rbind(x, -x))[, 2])
```

`int` *Integration routines*

Description

Several quadrature rules for integration of functions on S^1, S^2, and S^q, $q \geq 3$.

Usage

```
int_cir(f, N = 500, na.rm = TRUE, f_vect = TRUE, ...)

int_sph(f, na.rm = TRUE, f_vect = TRUE, ...)

int_hypsph(f, q, M = 1e+05, na.rm = TRUE, f_vect = TRUE, ...)
```

Arguments

f	function to be integrated on S^q. Must be vectorized and accept matrix inputs of size c(nx,q + 1).
N	Defaults to 5e2.
na.rm	ignore possible NAs arising from the evaluation of f? Defaults to TRUE.
f_vect	can f be called in a vectorized form, with matrix input? Defaults to TRUE.
...	further arguments passed to f.
q	dimension of S^q, $q \geq 1$.
M	number of Monte Carlo replicates. Defaults to 1e5.

Details

int_cir is an extension of equation (4.1.11) in Press et al. (1997), a periodic trapezoidal rule. int_sph employs the Lebedev quadrature on S^2. int_hypsph implements a Monte Carlo integration on S^q.

Value

A scalar approximating the integral.

References

Lebedev, V. I. and Laikov, D. N. (1999). A quadrature formula for the sphere of the 131st algebraic order of accuracy. *Doklady Mathematics*, 59(3):477–481.

Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery B. P. (1997). *Numerical Recipes in Fortran 77: The Art of Scientific Computing*. Volume 1. Cambridge University Press, Cambridge. Second edition.

Examples

```
# S^1, trapezoidal rule
f <- function(x) rotasym::d_vMF(x = x, mu = c(0, 1), kappa = 2)
int_cir(f = f)

# S^2, Lebedev rule
f <- function(x) rotasym::d_vMF(x = x, mu = c(0, 0, 1), kappa = 2)
int_sph(f = f)

# S^2, Monte Carlo
f <- function(x) rotasym::d_vMF(x = x, mu = c(0, 0, 1), kappa = 2)
int_hypsph(f = f, q = 2)
```

`kde_dir` *Directional kernel density estimator*

Description

Kernel density estimation with directional data as in the estimator of Bai et al. (1988).

Usage

```
kde_dir(x, data, h, L = NULL)

c_h(h, q, L = NULL)

lambda_L(L = NULL, q)

b_L(L = NULL, q)

d_L(L = NULL, q)
```

Arguments

`x`	evaluation points, a matrix of size `c(nx,q + 1)`.
`data`	directional data, a matrix of size `c(n,q + 1)`.
`h`	bandwidth, a scalar for `kde_dir`. Can be a vector for `c_h`.
`L`	kernel function. Set internally to `function(x) exp(-x)` (von Mises–Fisher kernel) if `NULL` (default).
`q`	dimension of S^q, $q \geq 1$.

Details

`data` is not checked to have unit norm, so the user must be careful. When `L = NULL`, faster FORTRAN code is employed.

Value

`kde_dir` returns a vector of size `nx` with the evaluated kernel density estimator. `c_h` returns the normalizing constant for the kernel, a vector of length `length(h)`. `lambda_L`, `b_L`, and `d_L` return moments of `L`.

References

Bai, Z. D., Rao, C. R., and Zhao, L. C. (1988). Kernel estimators of density function of directional data. *Journal of Multivariate Analysis*, 27(1):24–39. https://doi.org/10.1016/0047-259X(88)90113-3

Examples

```
# Sample
n <- 50
q <- 3
samp <- rotasym::r_vMF(n = n, mu = c(1, rep(0, q)), kappa = 2)

# Evaluation points
x <- rbind(diag(1, nrow = q + 1), diag(-1, nrow = q + 1))

# kde_dir
kde_dir(x = x, data = samp, h = 0.5, L = NULL)
kde_dir(x = x, data = samp, h = 0.5, L = function(x) exp(-x))

# c_h
c_h(h = 0.5, q = q, L = NULL)
c_h(h = 0.5, q = q, L = function(x) exp(-x))

# b_L
b_L(L = NULL, q = q)
b_L(L = function(x) exp(-x), q = q)

# d_L
d_L(L = NULL, q = q)
d_L(L = function(x) exp(-x), q = q)

# lambda_L
lambda_L(L = NULL, q = q)
lambda_L(L = function(x) exp(-x), q = q)
```

lebedev *Lebedev quadrature on the sphere*

Description

Nodes and weights for Lebedev quadrature on the sphere S^2. The rule has 5810 points and is exact up to polynomials of order 131.

Usage

```
lebedev
```

Format

A data frame with 5810 rows and two variables:

xyz nodes for quadrature, a matrix with three columns.

w weights for quadrature, a vector.

Details

The approximation to the integral of f has the form

$$\int_{S^2} f(x,y,z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z = 4\pi \sum_{i=1}^{N} w_i f(x_i, y_i, z_i)$$

where $N = 5810$. The nodes (in spherical coordinates) and weights are processed from lebedev_131.txt.

Source

https://people.sc.fsu.edu/~jburkardt/datasets/sphere_lebedev_rule/sphere_lebedev_rule.html

References

Lebedev, V. I. and Laikov, D. N. (1999). A quadrature formula for the sphere of the 131st algebraic order of accuracy. *Doklady Mathematics*, 59(3):477–481.

Examples

```
# Load data
data("lebedev")

# Integrate x_1 * x_2^2 (zero integral)
f_1 <- function(x) x[, 1] * x[, 2]^2
4 * pi * sum(lebedev$w * f_1(lebedev$xyz))
```

vmf — *Von Mises–Fisher distribution utilities*

Description

Maximum likelihood estimation for the von Mises–Fisher distribution and evaluation of density mixtures.

Usage

```
kappa_ml(data)

mu_ml(data)

d_mixvmf(x, mu, kappa, p, norm = FALSE)
```

Arguments

`data`	directional data, a matrix of size `c(n,q + 1)`.
`x`	evaluation points, a matrix of size `c(nx,q + 1)`.
`mu, kappa, p`	mixture parameters. `mu` is the mean matrix of size `c(length(p),q + 1)`, `kappa` is vector of `length(p)` concentration parameters, and `p` is the vector of mixture proportions.
`norm`	enforce normalization of `x` internally? Defaults to `FALSE`.

Value

Estimated vector mean (`mu_ml`) or concentration parameter (`kappa_ml`). A vector of length `nx` for `d_mixvmf`.

Examples

```
# Sample
n <- 50
q <- 2
samp <- rotasym::r_vMF(n = n, mu = c(1, rep(0, q)), kappa = 2)

# Estimates
mu_ml(samp)
kappa_ml(samp)

# Mixture
x <- to_cir(seq(0, 2 * pi, l = 200))
dens <- d_mixvmf(x = x, mu = rbind(c(-1, 0), c(0, 1), c(1, 0)),
                 kappa = 1:3, p = c(0.5, 0.2, 0.3))
plot(to_rad(x), dens, type = "l")
```

Index