Package ‘Directional’

June 28, 2019

Type Package
Title Directional Statistics
Version 3.9
URL
Date 2019-06-29
Author Michail Tsagris, Giorgos Athineou, Anamul Sajib, Eli Amson, Micah J. Waldstein
Maintainer Michail Tsagris <mtsagris@yahoo.gr>

Description

License GPL-2
Imports bigstatsr, parallel, doParallel, foreach, MASS, Rfast, Rfast2, RcppZiggurat, rgl,

NeedsCompilation no
Repository CRAN
Date/Publication 2019-06-28 20:30:03 UTC

R topics documented:

Directional-package ................................................................. 3
A test for testing the equality of the concentration parameters for circular data .... 5
Angular central Gaussian random values simulation .................................. 6
Anova for (hyper-)spherical data .................................................. 7
Anova for circular data ............................................................ 8
BIC for the model based clustering using mixtures of von Mises-Fisher distributions . 10
<table>
<thead>
<tr>
<th>R topics documented:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check visually whether matrix Fisher samples is correctly generated or not</td>
</tr>
<tr>
<td>Circular or angular regression</td>
</tr>
<tr>
<td>Circular-linear correlation</td>
</tr>
<tr>
<td>Circular correlations between one and many circular variables</td>
</tr>
<tr>
<td>Circular correlations between two circular variables</td>
</tr>
<tr>
<td>Contour plot of a mixture of von Mises-Fisher distributions model</td>
</tr>
<tr>
<td>Contour plot of spherical data using a von Mises-Fisher kernel density estimate</td>
</tr>
<tr>
<td>Contour plot of the Kent distribution for some data</td>
</tr>
<tr>
<td>Contour plot of the Kent distribution without any data</td>
</tr>
<tr>
<td>Conversion plots of the von Mises-Fisher distribution</td>
</tr>
<tr>
<td>Conversion of cosines to azimuth and plunge</td>
</tr>
<tr>
<td>Converting a rotation matrix on SO(3) to an unsigned unit quaternion</td>
</tr>
<tr>
<td>Converting an unsigned unit quaternion to rotation matrix on SO(3)</td>
</tr>
<tr>
<td>Cross validation for estimating the classification rate</td>
</tr>
<tr>
<td>Cross validation in von Mises-Fisher discriminant analysis</td>
</tr>
<tr>
<td>Cross validation with ESAG discriminant analysis</td>
</tr>
<tr>
<td>Density of some (hyper-)spherical distributions</td>
</tr>
<tr>
<td>Density of some circular distributions</td>
</tr>
<tr>
<td>Density of the spherical Kent and ESAG distributions</td>
</tr>
<tr>
<td>Euclidean transformation</td>
</tr>
<tr>
<td>Euler angles from a rotation matrix on SO(3)</td>
</tr>
<tr>
<td>Generate random folds for cross-validation</td>
</tr>
<tr>
<td>Goodness of fit test for grouped data</td>
</tr>
<tr>
<td>Habeck’s rotation matrix generation</td>
</tr>
<tr>
<td>Hypothesis test for IAG distribution over the ESAG distribution</td>
</tr>
<tr>
<td>Hypothesis test for von Mises-Fisher distribution over Kent distribution</td>
</tr>
<tr>
<td>Interactive 3D plot of spherical data</td>
</tr>
<tr>
<td>Inverse of Lambert’s equal area projection</td>
</tr>
<tr>
<td>Inverse of the Euclidean transformation</td>
</tr>
<tr>
<td>k-NN algorithm using the arc cosinus distance</td>
</tr>
<tr>
<td>k-NN regression</td>
</tr>
<tr>
<td>Lambert’s equal area projection</td>
</tr>
<tr>
<td>Logarithm of the Kent distribution normalizing constant</td>
</tr>
<tr>
<td>Mixtures of Von Mises-Fisher distributions</td>
</tr>
<tr>
<td>MLE of some circular distributions</td>
</tr>
<tr>
<td>MLE of the angular central Gaussian distribution</td>
</tr>
<tr>
<td>MLE of the ESAG distribution</td>
</tr>
<tr>
<td>MLE of the generalised von Mises distribution</td>
</tr>
<tr>
<td>MLE of the Kent distribution</td>
</tr>
<tr>
<td>Mle of the Matrix Fisher distribution on SO(3)</td>
</tr>
<tr>
<td>Mle of the Purkayashita distribution</td>
</tr>
<tr>
<td>MLE of the spherical projected normal distribution</td>
</tr>
<tr>
<td>MLE of the von Mises-Fisher distribution</td>
</tr>
<tr>
<td>MLE of the Wood bimodal distribution on the sphere</td>
</tr>
<tr>
<td>Prediction in discriminant analysis based on ESAG distribution</td>
</tr>
<tr>
<td>Prediction in discriminant analysis based on von Mises-Fisher distribution</td>
</tr>
<tr>
<td>Probability density function of the von Mises-Fisher distribution</td>
</tr>
<tr>
<td>Random sample of matrices in SO(p)</td>
</tr>
</tbody>
</table>
This is an R package that provides methods for the statistical analysis of directional data, including massive (very large scale) directional data.

Circular-linear regression, spherical-spherical regression, spherical regression, discriminant analysis, ANOVA for circular and (hyper-)spherical data, tests for equality of concentration parameters, fitting distributions, random values generation, contour plots and many more functions are included in this package.
Details

Package: Directional
Type: Package
Version: 3.9
Date: 2019-06-29
License: GPL-2

Maintainers

Michail Tsagris <mtsagris@yahoo.gr>

Note

Acknowledgments:

Professor Andy Wood and Dr Simon Preston from the university of Nottingham are highly appre-
ciated for being my supervisors during my post-doc in directional data analysis.

Dr Georgios Pappas (former postDoc at the university of Nottingham) helped me construct the
contour plots of the von Mises-Fisher and the Kent distribution.

Dr Christopher Fallaize and Dr Theo Kypraios from the university of Nottingham have provided a
function for simulating from the Bingham distribution using rejection sampling. So any questions
regarding this function should be addressed to them.

Dr Kwang-Rae Kim (post-doc at the university of Nottingham) answered some of my questions.

Giorgos Borboudakis (PhD student at the university of Crete) pointed out to me a not so clear
message in the algorithm of generating random values from the von Mises-Fisher distribution.

Panagiotis (pronounced Panayiotis) Tzirakis (master student at the department of computer science
in Heraklion during the 2013-2015 seasons) showed me how to perform parallel computing in R
and he is greatly acknowledged and appreciated not only from me but from all the readers of this
document. He also helped me with the vectorization of some contour plot functions.

Professor John Kent from the university of Leeds is acknowledged for clarifying one thing with the
ovalness parameter in his distribution.

Phillip Paine (postdoc at the university of Nottingham) spotted that the function rfb is rather slow
and he suggested me to change it. The function has changed now and this is also due to Joshua
Davis (from Carleton College, Northfield, MN) who spotted that mistakes could occur, due a vector
not being a matrix.

Professor Kurt Hornik from the Vienna university of economics and business is greatly acknowl-
edged for his patience and contest help with this (and not only) R package.

Manos Papadakis, undergraduate student in the department of computer science at university of
Crete, is also acknowledged for his programming tips.

Dr Mojgan Golzy spotted a mistake in the ESGdensity and Michail is very happy for that.

If you want more information on many of these algorithms see Chapters 9 and 10 in the following
A test for testing the equality of the concentration parameters for circular data

Author(s)
Michail Tsagris <mtsagris@yahoo.gr>, Giorgos Athineou <gioathineou@gmail.com>, Anamul Sajib <pmxahsa@nottingham.ac.uk>, Eli Amson <eli.amson1988@gmail.com> and Micah J. Waldstein <micah@waldste.in>.

References

Description
A test for testing the equality of the concentration parameter among g samples, where g >= 2 for circular data. It is a tangential approach.

Usage
tangNconc(u, ina, rads = FALSE)

Arguments
u A numeric vector containing the values of all samples.
ina A numerical variable or factor indicating the groups of each value.
rads If the data are in radians this should be TRUE and FALSE otherwise.

Details
This test works for circular data.

Value
A vector including:
test The value of the test statistic.
p-value The p-value of the test.

Author(s)
Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>
Angular central Gaussian random values simulation

References


See Also

embed.circaov, hcf.circaov, lr.circaov, het.circaov, conc.test

Examples

x <- rvonmises(100, 2.4, 15)
ina <- rep(1:4, each = 25)
tang.conc(x, ina, rads = TRUE)

Description

Angular central Gaussian random values simulation.

Usage

racg(n, sigma)

Arguments

n The sample size, a numerical value.
sigma The covariance matrix in $\mathbb{R}^d$.

Details

The algorithm uses univariate normal random values and transforms them to multivariate via a spectral decomposition. The vectors are then scaled to have unit length.

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>
Anova for (hyper-)spherical data

References


See Also

acg, rvmf, rvonmises

Examples

s <- cov(iris[, 1:4])
x <- racg(100, s)
acg(x)
vmf(x)  # the concentration parameter, kappa, is very low, close to zero, as expected.

Description

Analysis of variance for (hyper-)spherical data.

Usage

hcfaov(x, ina, fc = TRUE)

lr.aov(x, ina)

embed.aov(x, ina)

het.aov(x, ina)

Arguments

x A matrix with the data in Euclidean coordinates, i.e. unit vectors.

ina A numerical variable or a factor indicating the group of each vector.

fc A boolean that indicates whether a corrected F test should be used or not.

Details

The high concentration (hcfaov), log-likelihood ratio (lr.aov), embedding approach (embed.aov) or the non equal concentration parameters approach (het.aov) is used.
Value
A vector including:

- **test**: The test statistic value.
- **p-value**: The p-value of the F test.
- **kappa**: The common concentration parameter kappa based on all the data.

Author(s)
Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

See Also
lr.aov, embed.aov, het.aov, spherconc.test, conc.test

Examples
```r
x <- rvmf(60, rnorm(3), 15)
ina <- rep(1:3, each = 20)
hcf.aov(x, ina)
hcf.aov(x, ina, fc = FALSE)
lr.aov(x, ina)
embed.aov(x, ina)
het.aov(x, ina)
```

Anova for circular data

Analysis of variance for circular data

Description
Analysis of variance for circular data.

Usage
```r
hcf.circaov(u, ina, rads = FALSE)

lr.circaov(u, ina, rads = FALSE)

het.circaov(u, ina, rads = FALSE)

embed.circaov(u, ina, rads = FALSE)
```
Arguments

u A numeric vector containing the data.
ina A numerical or factor variable indicating the group of each value.
rads If the data are in radians, this should be TRUE and FALSE otherwise.

Details

The high concentration (hcf.circaov), log-likelihood ratio (lr.circaov), embedding approach (embed.circaov) or the non equal concentration parameters approach (het.circaov) is used.

Value

A vector including:

test The value of the test statistic.
p-value The p-value of the test.
kappa The concentration parameter based on all the data. If the het.circaov is used this argument is not returned.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

conc.test, hcf.aov, lr.aov, het.aov, embed.aov

Examples

x <- rvonmises(100, 2.4, 15)
ina <- rep(1:4,each = 25)
hcf.circaov(x, ina, rads = TRUE)
lr.circaov(x, ina, rads = TRUE)
het.circaov(x, ina, rads = TRUE)
embed.circaov(x, ina, rads = TRUE)
BIC for the model based clustering using mixtures of von Mises-Fisher distributions

BIC to choose the number of components in a model based clustering using mixtures of von Mises-Fisher distributions

Description

BIC to choose the number of components in a model based clustering using mixtures of von Mises-Fisher distributions

Usage

bic.mixvmf(x, A, n.start = 20)

Arguments

x A matrix containing directional data.
A The maximum number of clusters to be tested. Default value is 3.
n.start The number of random starts to try. See also R’s built-in function kmeans for more information about this.

Details

If the data are not unit vectors, they are transformed into unit vectors.

Value

A list including:

BIC The BIC values for all the models tested.
A plot A plot of the BIC values.
runtime The run time of the algorithm. A numeric vector. The first element is the user time, the second element is the system time and the third element is the elapsed time.

Author(s)

Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

Check visually whether matrix Fisher samples is correctly generated or not

See Also
mix.vmf, rmixvmf, mixvmf.contour

Examples

```r
x <- as.matrix( iris[, 1:4] )
x <- x / sqrt( rowSums(x^2) )
bic.mixvmf(x, 5)
```

Description

It plots the log probability trace of matrix Fisher distribution which should close to the maximum value of the logarithm of matrix Fisher distribution, if samples are correctly generated.

Usage

```r
visual.check(x, Fa)
```

Arguments

- `x` The simulated data. An array with at least 2 3x3 matrices.
- `Fa` An arbitrary 3x3 matrix represents the parameter matrix of this distribution.

Details

For a given parameter matrix Fa, maximum value of the logarithm of matrix Fisher distribution is calculated via the form of singular value decomposition of $Fa = U \Lambda V^T$ which is $tr(\Lambda)$. Multiply the last column of $U$ by $-1$ and replace small eigenvalue, say, $\lambda_3$ by $-\lambda_3$ if $|UV^T| = -1$.

Value

A plot which shows log probability trace of matrix Fisher distribution. The values are also returned.

Author(s)

Anamul Sajib<pmxahsa@nottingham.ac.uk>
R implementation and documentation: Anamul Sajib<pmxahsa@nottingham.ac.uk>

References

Circular or angular regression

Examples

```r
Fa <- matrix(c(85, 11, 41, 78, 39, 60, 43, 64, 48), ncol = 3) / 10
x <- rmatrixfisher(1000, Fa)
a <- visual.check(x, Fa)
```

Description

Regression with circular dependent variable and Euclidean or categorical independent variables.

Usage

```r
spml.reg(y, x, rads = TRUE, xnew = NULL, seb = FALSE, tol = 1e-07)
```

Arguments

- `y` The dependent variable, a numerical vector, it can be in radians or degrees.
- `x` The independent variable(s). Can be Euclidean or categorical (factor variables).
- `rads` If the dependent variable is expressed in rads, this should be TRUE and FALSE otherwise.
- `xnew` The new values of some independent variable(s) whose circular values you want to predict. Can be Euclidean or categorical. If they are categorical, the user must provide them as dummy variables. It does not accept factor variables. If you have no new x values, leave it NULL (default).
- `seb` a boolean variable. If TRUE, the standard error of the coefficients will be returned. Set to FALSE in case of simulation studies or in other cases such as a forward regression setting for example. In these cases, it can save some time. Leave this FALSE as we currently have a problem with this, but will fix it in our next update.
- `tol` The tolerance value to terminate the Newton-Raphson algorithm.

Details

The Newton-Raphson algorithm is fitted in this regression as described in Presnell et al. (1998).

Value

A list including:

- `runtime` The runtime of the procedure.
- `iters` The number of iterations required until convergence of the EM algorithm.
- `beta` The regression coefficients.
Circular-linear correlation

It calculates the squared correlation between a circular and one or more linear variables.

Usage

```r
circlin.cor(theta, x, rads = FALSE)
```

Arguments

- `theta` The circular variable.
- `x` The linear variable or a matrix containing many linear variables.
- `rads` If the circualr variable is in rads, this should be TRUE and FALSE otherwise.
**Circular correlations between one and many circular variables**

**Details**

The squared correlation between a circular and one or more linear variables is calculated.

**Value**

A matrix with as many rows as linear variables including:

- $r^2$-squared The value of the squared correlation.
- p-value The p-value of the zero correlation hypothesis testing.

**Author(s)**

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

**References**


**See Also**

circ.cor1, circ.cor2, spml.reg

**Examples**

```r
phi <- rvonmises(50, 2, 20, rads = TRUE)
x <- 2 * phi + rnorm(50)
y <- matrix(rnorm(50 * 5), ncol = 5)
circlin.cor(phi, x, rads = TRUE)
circlin.cor(phi, y, rads = TRUE)
```

---

**Circular correlations between two circular variables**

**Description**

Circular correlations between two circular variables.

**Usage**

`circ.cors1(theta, phi)`

**Arguments**

- `theta` The first circular variable expressed in radians, not degrees.
- `phi` The other circular variable. In the case of "circ.cors1" this is a matrix with many circular variables. In either case, the values must be in radians, not degrees.
**Details**

Correlation for circular variables using the cosinus and sinus formula of Jammalamadaka and Sengupta (1988).

**Value**

A matrix with two columns, the correlations and the p-values.

**Author(s)**

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

**References**


**See Also**

[spmlNreg](#)

**Examples**

```r
y <- runif(50, 0, 2 * pi)
x <- matrix(runif(50 * 10, 0, 2 * pi), ncol = 10)
circ.cors1(y, x)
```

---

**Description**

Circurlar correlations between two circular variables.

**Usage**

```r
circ.cor1(theta, phi, rads = FALSE)
circ.cor2(theta, phi, rads = FALSE)
```
Cirrcular correlations between two circular variables

Arguments

theta  The first circular variable.
phi   The other circular variable.
rads If the data are expressed in rads, then this should be TRUE. If the data are in
degrees, then this is FALSE.

Details

circ.cor1: Correlation for circular variables using the cosinus and sinus formula of Jammalamadaka
circ.cor2: Correlation for circular variables using the cosinus and sinus formula of Mardia and Jupp
(2000).

Value

A vector including:
rho The value of the correlation coefficient.
p-value The p-value of the zero correlation hypothesis testing.

Author(s)

Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

Statistical Theory and Data Analysis, 2:349–364.

See Also

circlin.cor, circ.cor2, spml.reg

Examples

y <- runif(50, 0, 2 * pi)
x <- runif(50, 0, 2 * pi)
circ.cor1(x, y, rads = TRUE)
circ.cor2(x, y, rads = TRUE)
Description

Contour lines are produced of mixture model for spherical data only.

Usage

mixvmf.contour(u, mod)

Arguments

u
A two column matrix. The first column is the longitude and the second is the latitude.

mod
This is mix.vmf object. Run a mixture model and save it as mod for example, mod = mix.vmf(x, 3).

Details

The contour plot is displayed with latitude and longitude in the axes. No Lambert projection is used here. This works for spherical data only which are given as longitude and latitude.

Value

A plot including: The points and the contour lines.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

vmf.kerncontour, vmf.contour, mix.vmf
Contour plot of spherical data using a von Mises-Fisher kernel density estimate

Examples

```r
k <- runif(2, 4, 20)
prob <- c(0.4, 0.6)
mu <- matrix(rnorm(6), ncol = 3)
mu <- mu / sqrt(rowSums(mu^2))
x <- rmixvmf(200, prob, mu, k)$x
mod <- mix.vmf(x, 2)
y <- euclid.inv(x)
mixvmf.contour(y, mod)
```

Description

Contour plot of spherical data using a von Mises-Fisher kernel density estimate.

Usage

```r
vmf.kerncontour(u, thumb = "none", den.ret = FALSE, full = FALSE, ngrid = 100)
```

Arguments

- **u**: A two column matrix. The first column is the latitude and the second is the longitude.
- **thumb**: This is either 'none' (default), or 'rot' for the rule of thumb suggested by Garcia-Portugues (2013). If it is "none" it is estimated via cross validation, with the fast function "vmfkde.tune_2".
- **den.ret**: If FALSE (default), plots the contours of the density along with the individual points. If TRUE, will instead return a list with the Longitudes, Latitudes and Densities. Look at the 'value' section for details.
- **full**: If FALSE (default), uses the range of positions from 'u' to calculate and optionally plot densities. If TRUE, calculates densities covering the entire sphere.
- **ngrid**: Sets the resolution of the density calculation.

Details

It calculates the contour plot using a von Mises-Fisher kernel for spherical data only.
Contour plot of the Kent distribution for some data

Value

The contour lines of the data. If "den.ret" was set to TRUE a list including:

lat: The latitude values.
long: The longitude values.
h: The optimal bandwidth.
den: The kernel density estimate contour points.

Author(s)

Michail Tsagris and Micah J. Waldstein.

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>, Giorgos Athineou <gioathineou@gmail.com> and Micah J. Waldstein <micah@waldste.in>.

References


See Also

vmf.kde, vmfkde.tune, vmf.contour, kent.datacontour

Examples

```r
x <- rvmf(100, rnorm(3), 15)
x <- euclid.inv(x)
par(mfrow = c(1, 2))
vmf.kerncontour(x, "rot")
vmf.kerncontour(x, "none")
```

Description

The contour plot of the Kent distribution on the sphere for some data is produced.

Usage

kent.datacontour(x)

Arguments

x: A two column matrix, where the first column is the latitude and the second column is the longitude. If the matrix has two columns, it is assumed to have unit vectors and in this case it is turned into latitude and longitude.
Details

MLE of the parameters of the Kent distribution are calculated, then the contour plot is plotted using
these estimates and finally the data are also plotted.

Value

A plot containing the contours of the distribution along with the data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

kent.contour, kent.mle, vmf.kerncontour

Examples

x <- rvmf(100, rnorm(3), 10)
kent.mle(x)
y <- euclid.inv(x)
kent.datacontour(y)
vmf.kerncontour(y, thumb = "none")

Description

The contour plot of the Kent distribution on the sphere is produced. The user can see how the shape
and ovalness change as he/she changes the ovalness parameter.

Usage

kent.contour(k, b)

Arguments

k The concentration parameter.
b The ovalness parameter. It has to be less than k/2 in order for the distribution to
be unimodal. Otherwise it is bimodal.
Details

The goal of this function is for the user to see how the Kent distribution looks like.

Value

A plot containing the contours of the distribution.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athi-
neou <gioathineou@gmail.com>

References


See Also

kent.datacontour, kent.mle, vmf.contour, vmf.kerncontour

Examples

par( mfrow = c(1, 2 )

kent.contour(10, 2)

kent.contour(10, 4)

Description

Contour plots of the von Mises-Fisher distribution on the sphere.

Usage

vmf.contour(k)

Arguments

k

The concentration parameter.

Details

The user specifies the concentration parameter only and not the mean direction or data. This is for il-

Illustration purposes only. The graph will always contain circles, as the von Mises-Fisher distribution

is the analogue of a bivariate normal in two dimensions with a zero covariance.
Conversion of cosines to azimuth and plunge

Value
A contour plot of the von Mises-Fisher distribution.

Author(s)
Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

See Also
rvmf, vmf, vmf.kerncontour, kent.contour, sphereplot

Examples
par(mfrow = c(1,3))
vmf.contour(1)
vmf.contour(5)
vmf.contour(10)

Conversion of cosines to azimuth and plunge

Conversion of cosines to azimuth and plunge

Description
Conversion of cosines to azimuth and plunge.

Usage
cosap(x, y, z)

Arguments
x x component of cosine.
y y component of cosine.
z z component of cosine.

Details
Orientation: x>0 is 'eastward', y>0 is 'southward', and z>0 is 'downward'.

Value
A The azimuth
P The plunge
Converting a rotation matrix on SO(3) to an unsigned unit quaternion

Author(s)
Eli Amson
R implementation and documentation: Eli Amson <eli.amson1988@gmail.com>

References

See Also
euclid, euclid.inv, eul2rot

Examples
cosap(-0.505, 0.510, -0.696)

Description
It returns an unsigned unite quaternion in $S^3$ (the four-dimensional sphere) from a $3 \times 3$ rotation matrix on SO(3).

Usage
rot2quat(X)

Arguments
X A rotation matrix in SO(3).

Details
Firstly construct a system of linear equations by equating the corresponding components of the theoretical rotation matrix proposed by Prentice (1986), and given a rotation matrix. Finally, the system of linear equations are solved by following the tricks mentioned in second reference here in order to achieve numerical accuracy to get quaternion values.

Value
A unsigned unite quaternion.
Converting an unsigned unit quaternion to rotation matrix on SO(3)

Author(s)

Anamul Sajib

R implementation and documentation: Anamul Sajib <pmxahsa@nottingham.ac.uk>

References


See Also

quat2rot, rotation, Arotation \ link{rot.matrix}

Examples

x <- rnorm(4)
x <- x/sqrt(sum(x^2))  ## an unit quaternion in R4
R <- quat2rot(x)
R
x
rot2quat(R)  ## sign is not exact as you can see

Description

It forms a (3 x 3) rotation matrix on SO(3) from an unsigned unit quaternion in S^3 (the four-dimensional sphere).

Usage

quat2rot(x)

Arguments

x

An unsigned unit quaternion in S^3.

Details

Given an unsigned unit quaternion in S^3 it forms a rotation matrix on SO(3), according to the transformation proposed by Prentice (1986).

Value

A rotation matrix.
Cross validation for estimating the classification rate

Author(s)
Anamul Sajib
R implementation and documentation: Anamul Sajib <pmxahsa@nottingham.ac.uk>

References

See Also
rot2quat, rotation, Arotation rot.matrix

Examples
```
x <- rnorm(4)
x <- x/sqrt(sum(x^2))
x ## an unit quaternion in R4
quat2rot(x)
```

Description
Cross validation for estimating the classification rate.

Usage
dirdaNcv(x, ina, folds = NULL, nfolds = 10, k = 2:10, stratified = FALSE, type = c("vmf", "iag", "esag", "kent", "sknn", "nsknn"), seed = FALSE, B = 1000, parallel = FALSE)

Arguments
- **x**: A matrix with the data in Euclidean coordinates, i.e. unit vectors. The matrix must have three columns, only spherical data are currently supported.
- **ina**: A variable indicating the groupings.
- **folds**: Do you already have a list with the folds? If not, leave this NULL.
- **nfolds**: How many folds to create?
- **k**: If you choose to use k-NN, what will be the k values?
- **stratified**: Should the folds be created in a stratified way? i.e. keeping the distribution of the groups similar through all folds?
- **seed**: If seed is TRUE, the results will always be the same.
Cross validation for estimating the classification rate

`type`

The type of classifier to use. The available options are "vmf" (von Mises-Fisher distribution), "esag" (ESAG distribution), "kent" (Kent distribution), "sknn" (standard k-NN) and "nsknn" (non standard k-NN). You can choose any of them or all of them. Note that "esag" and "kent" work only with spherical data.

`B`

If you used k-NN, should a bootstrap correction of the bias be applied? If yes, 1000 is a good value.

`parallel`

If you want the standard k-NN algorithm to take place in parallel set this equal to TRUE.

Details

Cross-validation for the estimation of the performance of a classifier.

The estimated performance of the best classifier is overestimated. After the cross-validation procedure, the predicted values produced by all classifiers are collected, from all folds, in an $n \times M$ matrix, where $n$ is the number of samples and $M$ the number of all classifiers used. We sample rows (predictions) with replacement from P and denote them as the in-sample values. The non resampled rows are denoted as out-of-sample values. The performance of each classifier in the insample rows is calculated and the classifier with the optimal performance is selected, followed by the calculation of performance in the out-of-sample values. This process is repeated B times and the average performance is returned. The only computational overhead is with the repetitive resampling and calculation of the performance, i.e. no model or classifier is fitted nor trained. For more information see Tsamardinos et al. (2018). This procedure though takes place only for the k-NN algorithm and is applied to each version standard ("sknn") or non-standard ("nsknn").

The good thing with the function is that you can run any method you want by supplying the folds yourselves using the command `makefolds`. Then suppose you want to run another method. By supplying the same folds you will be able to have comparative results for all methods.

Value

A list including:

`perf` A vector with the estimated performance of each classifier.

`best` The classifier with the optimal performance.

`boot.perf` The bootstrap bias corrected performance.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>.

References


Cross validation in von Mises-Fisher discriminant analysis


See Also

ESAG.da, vmfda.pred, dirknn, knn.reg

Examples

```r
x <- rvmf(300, rnorm(3), 10)
ina <- sample.int(4, 300, replace = TRUE)
dirda.cv(x, ina, B = 1000)
```

Cross validation in von Mises-Fisher discriminant analysis

Cross validation for estimating the classification rate of a discriminant analysis for directional data assuming a von Mises-Fisher distribution.

Description

Cross validation for estimating the classification rate of a discriminant analysis for directional data assuming a von Mises-Fisher distribution.

Usage

```r
vmf.da(x, ina, fraction = 0.2, R = 200, seed = FALSE)
```

Arguments

- `x`: A matrix with the data in Euclidean coordinates, i.e. unit vectors.
- `ina`: A variable indicating the groupings.
- `fraction`: The fraction of data to be used as test set.
- `R`: The number of repetitions.
- `seed`: If seed is TRUE, the results will always be the same.

Details

A repeated cross validation procedure is performed to estimate the rate of correct classification.

Value

A list including:

- `percent`: The estimated percent of correct classification and two estimated standard deviations. The one is the standard deviation of the rates and the other is assuming a binomial distribution.
- `ci`: Three types of confidence intervals, the standard one, another one based on the binomial distribution and the third one is the empirical one, which calculates the upper and lower 2.5% of the rates.
Cross validation with ESAG discriminant analysis

Author(s)

Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

vmfda.pred, mix.vmf, vmf, dirknn

Examples

```r
x <- rvmf(100, rnorm(4), 15)
ina <- rep(1:2, each = 50)
vmf.da(x, ina, fraction = 0.2, R = 200, seed = FALSE)
```

Description

Cross validation for estimating the classification rate of a discriminant analysis for directional data assuming an ESAG distribution.

Usage

```r
ESAG.da(y, ina, fraction = 0.2, R = 100, seed = FALSE)
```

Arguments

- `y`: A matrix with the data in Euclidean coordinates, i.e. unit vectors. The matrix must have three columns, only spherical data are currently supported.
- `ina`: A variable indicating the groupings.
- `fraction`: The fraction of data to be used as test set.
- `R`: The number of repetitions.
- `seed`: If seed is TRUE, the results will always be the same.

Details

A repeated cross validation procedure is performed to estimate the rate of correct classification.
Density of some (hyper-)spherical distributions

Value

A list including:

- **percent**: The estimated percent of correct classification and two estimated standard deviations. The one is the standard deviation of the rates and the other is assuming a binomial distribution.
- **ci**: Three types of confidence intervals, the standard one, another one based on the binomial distribution and the third one is the empirical one, which calculates the upper and lower 2.5% of the rates.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References


See Also

- vmf.da, vmfda.pred, dirknn

Examples

```r
x <- rvmf(100, rnorm(3), 15)
ina <- rep(1:2, each = 50)
ESAG.da(x, ina, fraction = 0.2, R = 50, seed = FALSE)
```

Density of some (hyper-)spherical distributions

Usage

```r
vmf.density(y, k, mu, logden = FALSE )
iag.density(y, mu, logden = FALSE)
purka.density(y, a, theta, logden = FALSE)
```
Density of some (hyper-)spherical distributions

Arguments

- **y**: A matrix or a vector with the data expressed in Euclidean coordinates, i.e. unit vectors.
- **k**: The concentration parameter of the von Mises-Fisher distribution.
- **a**: The concentration parameter of the Purkayastha distribution.
- **mu**: The mean direction (unit vector) of the von Mises-Fisher distribution or the mean direction of the IAG distribution.
- **theta**: The median direction for the Purkayastha distribution.
- **logden**: If you the logarithm of the density values set this to TRUE.

Details

The density of the von Mises-Fisher, of the IAG or of the Purkayastha distribution is computed.

Value

A vector with the (log) density values of y.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References


See Also

kent.mle, rkent, ESAGmle

Examples

```r
m <- colMeans( as.matrix( iris[,1:3] ) )
y <- rvmf(1000, m = m, k = 10)
vmf.density(y, k=10, m )
```
Density of some circular distributions

Description

Density of some circular distributions.

Usage

vm.density(x, m, k, rads = FALSE, logden = FALSE)
spml.density(x, mu, rads = FALSE, logden = FALSE)
wrapcauchy.density(x, m, rho, rads = FALSE, logden = FALSE)
circpurka.density(x, m, a, rads = FALSE, logden = FALSE)

Arguments

x A vector with circular data.
m The mean value, a scalar. This is the median for the circular Purkayastha distribution.
mu The mean vector, a vector with two values.
k The concentration parameter.
rho The $\rho$ parameter of the wrapped Cauchy distribution.
a The $\alpha$ parameter of the circular Purkayastha distribution.
rads If the data are in rads, then this should be TRUE, otherwise FALSE.
logden If you the logarithm of the density values set this to TRUE.

Details

The density of the von Mises, bivariate projected normal, wrapped Cauchy or the circular Purkayastha distributions is computed.

Value

A vector with the (log) density values of x.

Author(s)

Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References

Density of the spherical Kent and ESAG distributions

See Also

kent.\texttt{density}, \texttt{rvonmises}, ESAG\texttt{density}

Examples

\begin{verbatim}
x <- rvonmises(500, m = 2.5, k = 10, rads = TRUE) 
mod <- circ.summary(x, rads = TRUE, plot = FALSE) 
den <- vm.density(x, mod$mesos, mod$kappa, rads = TRUE, logden = TRUE) 
mod$loglik 
sum(den)
\end{verbatim}

Description

Density of the spherical Kent and ESAG distributions.

Usage

\begin{verbatim}
kent.\texttt{density}(y, G, param, logden = FALSE) 
ESAG\texttt{density}(y, param, logden = FALSE)
\end{verbatim}

Arguments

\begin{itemize}
  \item \texttt{y} A matrix or a vector with the data expressed in Euclidean coordinates, i.e. unit vectors.
  \item \texttt{G} For the Kent distribution only, a 3 x 3 matrix whose first column is the mean direction. The second and third columns are the major and minor axes respectively.
  \item \texttt{param} For the Kent distribution a vector with the concentration $\kappa$ and ovalness $\beta$ parameters. The $\psi$ has been absorbed inside the matrix $G$.
  \item \texttt{logden} For the ESAG distribution, its parameters, the first three are the mean vector in $\mathbb{R}^3$ and the next two are the two gammas.
  \item \texttt{logden} If you the logarithm of the density values set this to \texttt{TRUE}.
\end{itemize}

Details

The density of the spherical Kent or spherical ESAG distribution is computed.

Value

A vector with the (log) density values of \texttt{y}. 
Euclidean transformation

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References


See Also

kent.mle, rkent, ESAGmle

Examples

m <- colMeans(as.matrix(iris[,1:3]))
y <- rkent(1000, k = 10, m = m, b = 4)
mod <- kent.mle(y)
  kent.density(y, G = mod$G, param = mod$param)

Euclidean transformation

Euclidean transformation

Description

It transforms the data from the spherical coordinates to Euclidean coordinates.

Usage

euclid(u)

Arguments

u A two column matrix or even one single vector, where the first column (or element) is the latitude and the second is the longitude. The order is important.

Details

It takes the matrix of unit vectors of latitude and longitude and transforms it to unit vectors.
Euler angles from a rotation matrix on SO(3)

Value

A three column matrix:

\[ u \]

The Euclidean coordinates of the latitude and longitude.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athi-
neou <gioathineou@gmail.com>

See Also

euclid.inv, Arotation, lambert

Examples

```r
x <- rvmf(10, rnorm(3), 10)
u <- euclid.inv(x)
euclid(u)
x
```

---

Euler angles from a rotation matrix on SO(3)

*Compute the Euler angles from a rotation matrix on SO(3).*

Description

It calculates three euler angles \((\theta_1, \theta_2, \theta_3)\) from a \((3 \times 3)\) rotation matrix \(X\), where \(X\) is defined as \(X = R_z(\theta_1) \times R_y(\theta_2) \times R_x(\theta_3)\). Here \(R_x(\theta_3)\) means a rotation of \(\theta_3\) radians about the x axis.

Usage

```r
rot2eul(X)
```

Arguments

\(X\)

A rotation matrix which is defined as a product of three elementary rotations mentioned above. Here \(\theta_1, \theta_3 \in (-\pi, \pi)\) and and \(\theta_3 \in (-\pi/2, \pi/2)\).

Details

Given a rotation matrix \(X\), euler angles are computed by equating each element in \(X\) with the corresponding element in the matrix product defined above. This results in nine equations that can be used to find the euler angles.
Generate random folds for cross-validation

Value
For a given rotation matrix, there are two equivalent sets of euler angles.

Author(s)
Anamul Sajib<pmxahsa@nottingham.ac.uk>
R implementation and documentation: Anamul Sajib<pmxahsa@nottingham.ac.uk>

References
http://www.staff.city.ac.uk/~sbbh653/publications/euler.pdf

See Also
eul2rot

Examples

# three euler angles
theta.12 <- sample( seq(-3, 3, 0.3), 1 )
theta.23 <- sample( seq(-3, 3, 0.3), 1 )
theta.13 <- sample( seq(-1.4, 1.4, 0.3), 1 )
theta.12 ; theta.23 ; theta.13

X <- eul2rot(theta.12, theta.23, theta.13)
X # A rotation matrix

e <- rot2eul(X)$v1
theta.12 <- e[3]
theta.23 <- e[2]
theta.13 <- e[1]

theta.12 ; theta.23 ; theta.13

Description
Random folds for use in a cross validation are generated. There is the option for stratified splitting as well.
Usage

makefolds(ina, nfolds = 10, stratified = TRUE, seed = FALSE)

Arguments

ina A variable indicating the groupings.
nfolds The number of folds to produce.
stratified A boolean variable specifying whether stratified random (TRUE) or simple random (FALSE) sampling is to be used when producing the folds.
seed A boolean variable. If set to TRUE, the folds will always be the same.

Details

I was inspired by the command in the package TunePareto in order to do the stratified version.

Value

A list with nfolds elements where each elements is a fold containing the indices of the data.

Author(s)

Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

See Also

dirda.cv

Examples

a <- makefolds(iris[, 5], nfolds = 5, stratified = TRUE)
table(iris[a[1]], 5))  # 10 values from each group
Arguments

- **g**: A vector with the group points, either in radians or in degrees.
- **ni**: The frequency of each or group class.
- **m**: The mean direction in radians or in degrees.
- **k**: The concentration parameter, $\kappa$.
- **dist**: The distribution to be tested, it can be either "vm" or "uniform".
- **rads**: If the data are in radians, this should be TRUE and FALSE otherwise.
- **R**: The number of bootstrap simulations to perform, set to 999 by default.
- **ncores**: The number of cores to use.

Details

When you have grouped data, you can test whether the data come from the von Mises-Fisher distribution or from a uniform distribution.

Value

A list including:

- **info**: A vector with two elements, the test statistic value and the bootstrap p-value.
- **runtime**: The runtime of the procedure.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

`pvm`, `circ.summary`, `rvonmises`

Examples

```r
x <- rvonmises(100, 2, 10)
g <- seq(min(x) - 0.1, max(x) + 0.1, length = 6)
ni <- as.vector(table(cut(x, g)))
group.gof(g, ni, R = 299, ncores = 1, dist = "vm", rads = TRUE)
group.gof(g, ni, R = 299, ncores = 1, dist = "vm", rads = TRUE)
```
Habeck’s rotation matrix generation

*Generation of three-dimensional random rotations using Habeck’s algorithm.*

Description

It generates random rotations in three-dimensional space that follow a probability distribution, matrix Fisher distribution, arising in fitting and matching problem.

Usage

`habeck.rot(F)`

Arguments

F

An arbitrary 3 x 3 matrix represents the parameter matrix of this distribution.

Details

Firstly rotation matrices $X$ are chosen which are the closest to $F$, and then parameterized using euler angles. Then a Gibbs sampling algorithm is implemented to generate rotation matrices from the resulting distribution of the euler angles.

Value

A simulated rotation matrix.

Author(s)

Anamul Sajib<pmxahsa@nottingham.ac.uk>

R implementation and documentation: Anamul Sajib<pmxahsa@nottingham.ac.uk>

References


Examples

```r
F <- 10^(-1) * matrix(c(85, 11, 41, 78, 39, 60, 43, 64, 48), ncol = 3)  ## Arbitrary F matrix
X <- habeck.rot(F)
det(X)
```
Hypothesis test for IAG distribution over the ESAG distribution

Description

The null hypothesis is whether an IAG distribution fits the data well, where the alternative is that ESAG distribution is more suitable.

Usage

iagesag(x, B = 1, tol = 1e-07)

Arguments

x
A numeric matrix with three columns containing the data as unit vectors in Euclidean coordinates.

B
The number of bootstrap re-samples. By default is set to 999. If it is equal to 1, no bootstrap is performed and the p-value is obtained through the asymptotic distribution.

tol
The tolerance to accept that the Newton-Raphson algorithm used in the IAG distribution has converged.

Details

Essentially it is a test of rotational symmetry, whether the two $\gamma$ parameters are equal to zero. This works for spherical data only.

Value

A vector including:

test
The value of the test statistic.

p-value or Bootstrap p-value
The p-value of the test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References

Hypothesis test for von Mises-Fisher distribution over Kent distribution

See Also

fishkent, ESAGmle, kent.mle, iag.mle

Examples

```r
x <- rvmf(100, rnorm(3), 15)
iagesag(x)
fishkent(x, B = 1)
```

Description

The null hypothesis is whether a von Mises-Fisher distribution fits the data well, where the alternative is that Kent distribution is more suitable.

Usage

```r
fishkent(x, B = 999)
```

Arguments

- **x**: A numeric matrix containing the data as unit vectors in Euclidean coordinates.
- **B**: The number of bootstrap re-samples. By default is set to 999. If it is equal to 1, no bootstrap is performed and the p-value is obtained through the asymptotic distribution.

Details

Essentially it is a test of rotational symmetry, whether Kent's ovalness parameter (beta) is equal to zero. This works for spherical data only.

Value

A vector including:

- **test**: The value of the test statistic
- **p-value or Bootstrap p-value**: The p-value of the test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>
Interactive 3D plot of spherical data

References


See Also

iagesag, vmf, kent.mle, rkent

Examples

```r
x <- rvmf(100, rnorm(3), 15)
fishkent(x)
fishkent(x, B = 1)
iagesag(x)
```

Description

Interactive 3D plot of spherical data.

Usage

```r
sphereplot(x, col = NULL)
```

Arguments

- `x`: A matrix with three columns, unit-vectors, spherical data.
- `col`: If you want the points to appear with different colours put numbers here, otherwise leave it NULL.

Value

An interactive 3D plot of the spherical data will appear.

Author(s)

Michail Tsagris R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

See Also

lambert, vmf.contour, euclid
Inverse of Lambert’s equal area projection

Examples

```r
## Not run:
x <- rvmf(100, rnorm(3), 5)
sphereplot(x)
ing <- rbinom(100, 1, 0.5) + 1
sphereplot(x, col = ina)

## End(Not run)
```

Description

It takes some points from the cartesian coordinates and maps them onto the sphere. The inverse os the Lambert’s equal area projection.

Usage

```r
lambert.inv(z, mu)
```

Arguments

- `z`  
  A two- column matrix containing the Lambert’s equal rea projected data.
- `mu`  
  The mean direction of the data on the sphere.

Details

The data are first mapped on the sphere with mean direction equal to the north pole. Then, they are rotated to have the given mean direction. It is the inverse of the Lambert’s equal are projection.

Value

A matrix containing spherical data (unit vectors).

Author(s)

Michail Tsagris  
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

Inverse of the Euclidean transformation

See Also

lambert

Examples

```r
m <- rnorm(3)
m <- m / sqrt(sum(m^2))
x <- rvmf(20, m, 19)
mu <- vmf(x)$mu
y <- lambert(euclid.inv(x))
lambert.inv(y, mu)
euclid.inv(x)
```

Description

It transforms the data from the Euclidean coordinates to latitude and longitude.

Usage

euclid.inv(U)

Arguments

U

A matrix of unit vectors, or even one single unit vector in three dimensions.

Details

It takes the matrix of unit vectors and back transforms it to latitude and longitude.

Value

A two column matrix:

u

The first column is the latitude and the second is the longitude, both expressed in degrees.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

See Also

euclid, Arotation, lambert
**k-NN algorithm using the arc cosinus distance**

**Examples**

```r
x <- rvmf(10, rnorm(3), 10)
euclid.inv(x)
euclid( euclid.inv(x) )
x
```

**Description**

It classifies new observations to some known groups via the k-NN algorithm.

**Usage**

```r
dirknn(x, xnew, k = 5, ina, type = "S", mesos = TRUE, parallel = FALSE)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>x</code></td>
<td>The data, a numeric matrix with unit vectors.</td>
</tr>
<tr>
<td><code>xnew</code></td>
<td>The new data whose membership is to be predicted, a numeric matrix with unit vectors.</td>
</tr>
<tr>
<td><code>k</code></td>
<td>The number of nearest neighbours, set to 5 by default. It can also be a vector with many values.</td>
</tr>
<tr>
<td><code>ina</code></td>
<td>A variable indicating the groups of the data x.</td>
</tr>
<tr>
<td><code>type</code></td>
<td>If type is &quot;S&quot;, the standard k-NN algorithm is to be used, else &quot;NS&quot; for the non standard one. See below (details) for more information.</td>
</tr>
<tr>
<td><code>mesos</code></td>
<td>A boolean variable used only in the case of the non standard algorithm (type=&quot;NS&quot;). Should the average of the distances be calculated (TRUE) or not (FALSE)? If it is FALSE, the harmonic mean is calculated.</td>
</tr>
<tr>
<td><code>parallel</code></td>
<td>If you want the standard -NN algorithm to take place in parallel set this equal to TRUE.</td>
</tr>
</tbody>
</table>

**Details**

The standard algorithm is to keep the k nearest observations and see the groups of these observations. The new observation is allocated to the most frequent seen group. The non standard algorithm is to calculate the classical mean or the harmonic mean of the k nearest observations for each group. The new observation is allocated to the group with the smallest mean distance.

**Value**

A vector including:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>g</code></td>
<td>A matrix with the predicted group(s). It has as many columns as the values of k.</td>
</tr>
</tbody>
</table>
**k-NN regression**

**Author(s)**

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

**See Also**

dirknn.tune, vmfda.pred, mix.vmf

**Examples**

```r
k <- runif(4, 4, 20)
prob <- c(0.2, 0.4, 0.3, 0.1)
mu <- matrix(rnorm(16), ncol = 4)
mux <- mu / sqrt(rowSums(mu^2))
da <- rmixvfm(200, prob, mu, k)
nu <- sample(1:200, 180)
x <- da$x[nu, ]
ina <- da$id[nu]
xx <- da$x[-nu, ]
id <- da$id[-nu]
a1 <- dirknn(x, xx, k = 5, ina, type = "S", mesos = TRUE)
a2 <- dirknn(x, xx, k = 5, ina, type = "NS", mesos = TRUE)
a3 <- dirknn(x, xx, k = 5, ina, type = "S", mesos = FALSE)
a4 <- dirknn(x, xx, k = 5, ina, type = "NS", mesos = FALSE)
b <- vmfda.pred(xx, x, ina)
table(id, a1)
table(id, a2)
table(id, a3)
table(id, a4)
```

---

**Description**

k-NN regression with Euclidean or (hyper-)spherical response and or predictor variables.

**Usage**

```r
knn.reg(xnew, y, x, k = 5, res = "eucl", type = "euclidean", estim = "arithmetic")
```
**Arguments**

- **xnew**
  The new data, new predictor variables values. A matrix with either euclidean (univariate or multivariate) or (hyper-)spherical data. If you have a circular response, say u, transform it to a unit vector via \((\cos(u), \sin(u))\). If \(x_{\text{new}} = x\), you will get the fitted values.

- **y**
  The currently available data, the response variables values. A matrix with either euclidean (univariate or multivariate) or (hyper-)spherical data. If you have a circular response, say u, transform it to a unit vector via \((\cos(u), \sin(u))\).

- **x**
  The currently available data, the predictor variables values. A matrix with either euclidean (univariate or multivariate) or (hyper-)spherical data. If you have a circular response, say u, transform it to a unit vector via \((\cos(u), \sin(u))\).

- **k**
  The number of nearest neighbours, set to 5 by default. This can also be a vector with many values.

- **res**
  The type of the response variable. If it is Euclidean, set this argument equal to "res". If it is a unit vector set it to res="spher".

- **type**
  The type of distance to be used. This determines the nature of the predictor variables. This is actually the argument "method" of the distance function in R. The default value is "euclidean". If you use the Euclidean distance, the package "Rfast" is used. The "dista" function of that package is about 3 times faster than the standard built-in "dist". R has several options the type of the distance. Just type ?Rfast:Dist in R and see the methods. Any method can be given here. If you have unit vectors in general, you should put type="spher", so that the cosinus distance is calculated.

- **estim**
  Once the k observations with the smallest distance are discovered, what should the prediction be? The arithmetic average of the corresponding y values be used estim="arithmetic" or their harmonic average estim="harmonic".

**Details**

This function covers a broad range of data, Euclidean and spherical, along with their combinations.

**Value**

A list with as many elements as the number of values of k. Each element in the list contains a matrix (or a vector in the case of Euclidean data) with the predicted response values.

**Author(s)**

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsgiris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

**See Also**

knnreg.tune, spher.reg, spml.reg
**Lambert’s equal area projection**

**Examples**

```r
y <- iris[, 1]
x <- as.matrix(iris[, 2:4])
x <- x/ sqrt(rowSums(x^2))  # Euclidean response and spherical predictors
a <- knn.reg(x, y, x, k = 5, res = "eucl", type = "spher", estim = "arithmetic")

y <- iris[, 2:4]
y <- y / sqrt(rowSums(y^2))  # Spherical response and Euclidean predictor
x <- iris[, 1]
a <- knn.reg(x, y, x, k = 5, res = "spher", type = "euclidean", estim = "arithmetic")
```

**Description**

It calculates the Lambert’s equal area projection.

**Usage**

`lambert(y)`

**Arguments**

`y`  
A two column matrix with the data. The first column is the altitude and the second is the longitude.

**Details**

The spherical data are first rotated so that their mean direction is the north pole and then are projected on the plane tangent to the sphere at the north pole.

**Value**

A two-column matrix with the projected points.

**Author(s)**

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

**References**

Logarithm of the Kent distribution normalizing constant

See Also
euclid, lambert.inv

Examples

```r
x <- rvMF(100, rnorm(3), 20)
x <- euclid.inv(x)
a <- lambert(x)
plot(a)
```

Logarithm of the Kent distribution normalizing constant

Logarithm of the Kent distribution normalizing constant

Description

Logarithm of the Kent distribution normalizing constant.

Usage

```r
kent.logcon(k, b, j = 100)
```

Arguments

- `k` The conencration parameter, \( \kappa \).
- `b` The ovalness parameter, \( \beta \).
- `j` The number of the terms in the sum to use. By default this is 100.

Details

It calculates logarithm of the normalising constant of the Kent distribution.

Value

The value of the logarithm of the normalising constant of the Kent distribution.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

Mixtures of Von Mises-Fisher distributions

Description

It performs model based clustering for circular, spherical and hyperspherical data assuming von Mises-Fisher distributions.

Usage

```r
mix.vmf(x, g, n.start = 20)
```

Arguments

- `x`: A matrix with the data expressed as unit vectors.
- `g`: The number of groups to fit. It must be greater than or equal to 2.
- `n.start`: The number of random starts to try. See also R’s built-in function `kmeans` for more information about this.

Details

The initial step of the algorithm is not based on a spherical k-means, but on simple k-means. The results are comparable to the package movMF.

Value

A list including:

- `param`: A matrix with the mean direction, the concentrations parameter and mixing probability of each group.
- `loglik`: The value of the maximised log-likelihood.
- `pred`: The predicted group of each observation.
- `runtime`: The run time of the algorithm. A numeric vector. The first element is the user time, the second element is the system time and the third element is the elapsed time.

See Also

- `fb.saddle`, `kent.mle`

Examples

```r
kent.logcon(10, 2)
fbsaddle( c(0, 10, 0), c(0, -2, 2) )
```
MLE of some circular distributions

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

rmixvmf, bic.mixvmf, mixvmf.contour

Examples

```r
k <- runif(4, 4, 20)
prob <- c(0.2, 0.4, 0.3, 0.1)
mu <- matrix(rnorm(16), ncol = 4)
mu <- mu / sqrt(rowSums(mu^2))
x <- rmixvmf(200, prob, mu, k)$x
mix.vmf(x, 3)
mix.vmf(x, 4)
mix.vmf(x, 5)
```

Description

MLE of some circular distributions.

Usage

```r
spml.mle(x, rads = FALSE, tol = 1e-07)
wrapcauchy(x, rads = FALSE, tol = 1e-07)
```

Arguments

- `x`: A numerical vector with the circular data. They must be expressed in radians.
- `rads`: If the data are in radians set this to TRUE.
- `tol`: The tolerance level to stop the iterative process of finding the MLEs.
Detailed

The parameters of the bivariate angular Gaussian and wrapped Cauchy distributions are estimated. For the Wrapped Cauchy, the iterative procedure described by Kent and Tyler (1988) is used. The Newton-Raphson algorithm for the angular Gaussian is described in the regression setting in Presnell et al. (1998).

Value

A list including:

- `iters` : The iterations required until convergence.
- `loglik` : The value of the maximised log-likelihood.
- `param` : A vector consisting of the estimates of the two parameters, the mean direction for both distributions and the concentration parameter kappa and the rho for the von Mises and wrapped Cauchy respectively.
- `gamma` : The norm of the mean vector of the angular Gaussian distribution.
- `mu` : The mean vector of the angular Gaussian distribution.
- `mumu` : In the case of "spml.mle" this is the mean angle in radians.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References


See Also

`vmf`, `rvonmises`, `rvmf`

Examples

```r
x <- rvonmises(1000, 3, 9)
spml.mle(x)
wrapcauchy(x)
```
MLE of the angular central Gaussian distribution

**Description**

MLE of the angular central Gaussian distribution.

**Usage**

```r
acg(x, tol = 1e-07)
```

**Arguments**

- `x`: A matrix with directional data, i.e. unit vectors.
- `tol`: The tolerance value at which to terminate the iterations.

**Details**

There is a constraint on the estimated covariance matrix; its trace is equal to the number of variables. An iterative algorithm takes place and convergence is guaranteed.

**Value**

A list including:

- `iter`: The number of iterations required by the algorithm to converge to the solution.
- `cova`: The estimated covariance matrix.

**Author(s)**

Michail Tsagris R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

**References**


**See Also**

`rbingham, rfb, f.rbing, fb.saddle`
MLE of the ESAG distribution

Examples

```r
m <- c(0, 0, 0, 0)
s <- cov(iris[, 1:4])
x <- MASS::mvrnorm(500, m, s)
x <- x / sqrt(Rfast::rowsums(x^2))
mod <- acg(x)
mod
cov2cor(mod%*%cov) ## estimated covariance matrix turned into a correlation matrix
cov2cor(s) ## true covariance matrix turned into a correlation matrix
```

Description

MLE of the ESAG distribution.

Usage

```r
ESAGmle(y, full = FALSE, tol = 1e-06)
```

Arguments

- `y`: A matrix with the data expressed in Euclidean coordinates, i.e. unit vectors.
- `full`: If you want some extra information, the inverse of the covariance matrix, the `rho` parameter (smallest eigenvalue of the covariance matrix) and the angle of rotation $\psi$ set this equal to TRUE. Otherwise leave it FALSE.
- `tol`: A tolerance value to stop performing successive optimizations.

Details

MLE of the MLE of the ESAG distribution, on the sphere, is implemented. ESAG stands for Elliptically Symmetric Angular Gaussian and it was suggested by Paine et al. (2017). Unlike the projected normal distribution this is rotationally symmetric and is a competitor of the spherical Kent distribution (which is also non rotational symmetric).

Value

A list including:

- `mu`: The mean vector in $R^3$.
- `gam`: The two gamma parameters.
- `loglik`: The log-likelihood value.
- `vinv`: The inverse of the covariance matrix. It is returned if the argument “full” is TRUE.
rho  The \( \rho \) parameter (smallest eigenvalue of the covariance matrix). It is returned if the argument "full" is TRUE.

psi  The angle of rotation \( \psi \) set this equal to TRUE. It is returned if the argument "full" is TRUE.

iag.loglik  The log-likelihood value of the isotropic angular Gaussian distribution. That is, the projected normal distribution which is rotational symmetric.

Author(s)
Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References


See Also
ESAgdensity, ESAGsim, iag.mle, kent.mle, acg, circ.summary, sphereplot

Examples

```r
m <- colMeans( as.matrix( iris[,1:3] )
y <- ESAgSim(1000, c(m, 1,0.5) )
ESAgMle(y)
```

Description
MLE of the generalised von Mises distribution.

Usage
```r
ggvm(phi, rads = FALSE)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phi</td>
<td>A numerical vector with the circular data.</td>
</tr>
<tr>
<td>rads</td>
<td>Whether the data are in rads (TRUE) or not (FALSE).</td>
</tr>
</tbody>
</table>
MLE of the Kent distribution

Details

The generalised von Mises distribution (Dietrich and Richter, 2016) is fitted to some data and its parameters are estimated.

Value

A list including:

- `loglik`  
  The value of the maximised log-likelihood.
- `param`  
  A vector consisting of the $\zeta$, $\kappa$, $\mu$ and $\alpha$ parameters of the generalised von Mises distribution as described in Equation (2.7) of Dietrich and Richter (2016).

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References


See Also

circ.summary, rvonmises

Examples

```r
x <- rvonmises(100, 2, 25, rads = TRUE)
circ.summary(x, rads = TRUE)
ggm(x, rads = TRUE)
```

Description

It estimates the concentration and the ovalness parameter of some directional data assuming the Kent distribution. The mean direction and major and minor axes are also estimated.

Usage

`kent.mle(x)`

Arguments

- `x`  
  A matrix containing spherical data in Euclidean coordinates.
Details

The Kent distribution is fitted to some data and its parameters are estimated.

Value

A list including:

- `runtime` The run time of the procedure.
- `G` A 3 x 3 matrix whose first column is the mean direction. The second and third columns are the major and minor axes respectively.
- `param` A vector with the concentration $\kappa$ and ovalness $\beta$ parameters and the angle $\psi$ used to rotate $\mathbf{H}$ and hence estimate $\mathbf{G}$ as in Kent (1982).
- `logcon` The logarithm of the normalising constant, using the third type approximation (Kume and Wood, 2005).
- `loglik` The value of the log-likelihood.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

- `kent.mle`, `fb.saddle`, `vmf`, `wood.mle`, `sphereplot`

Examples

```r
x <- rvmf(200, rnorm(3), 15)
kent.mle(x)
vmf(x)
A <- diag(c(-5, 0, 5))
x <- rfb(200, 15, rnorm(3), A)
kent.mle(x)
vmf(x)
```
Description

It returns the maximum likelihood estimate of the Matrix Fisher parameter \( F(3\times3) \).

Usage

\[
\text{matrixfisher.mle}(X)
\]

Arguments

\( X \) An array containing rotation matrices in \( SO(3) \).

Value

The components of \( svd(\bar{X}) \).

Author(s)

Anamul Sajib & Chris Fallaize.

R implementation and documentation: Anamul Sajib <pmxahsa@nottingham.ac.uk> & Chris Fallaize.

References


See Also

\text{rmatrixfisher}

Examples

\[
F \leftarrow 10^{-1} \times \text{matrix}(\text{c}(85, 11, 41, 78, 39, 60, 43, 64, 48), \text{ncol} = 3) \quad \#\#\text{An arbitrary } F \text{ matrix}
\]

\[
X \leftarrow \text{rmatrixfisher}(5000, F)
\]

\[
\text{matrixfisher.mle}(X)
\]

\[
\text{svd}(F)
\]
MLE of the Purkayashta distribution

Description

MLE of the Purkayashta distribution.

Usage

purka.mle(x, tol = 1e-07)

Arguments

x A numerical vector with data expressed in radians or a matrix with spherical data.

 tol The tolerance value to terminate the Brent algorithm.

Details

MLE of the Purkayastha distribution is performed.

Value

A list including:

theta The median direction.
alpha The concentration parameter.
loglik The log-likelihood.
alpha.sd The standard error of the concentration parameter.

Author(s)

Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References


See Also
circ.cor1
Examples

```r
x <- cbind(rnorm(100,1,1), rnorm(100, 2, 1))
x <- x / sqrt(rowSums(x^2))
purka.mle(x)
```

Description

MLE of the spherical projected normal distribution.

Usage

```r
iag.mle(y, tol = 1e-07)
```

Arguments

- `y`: A matrix with the data expressed in Euclidean coordinates, i.e. unit vectors.
- `tol`: The tolerance to accept that the E-M algorithm used to estimate the concentration parameter has converged.

Details

MLE of the projected normal distribution, on the sphere, is implemented.

Value

A list including:

- `iters`: The number of iteration required by the Newton-Raphson.
- `mesi`: A matrix with two rows. The first row is the mean direction and the second is the mean vector. The first comes from the second by normalising to have unit length.
- `param`: A vector with the elements, the norm of mean vector, the log-likelihood and the log-likelihood of the spherical uniform distribution. The third value helps in case you want to do a log-likelihood ratio test for uniformity.
- `kappa`: The concentration parameter.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>
MLE of the von Mises-Fisher distribution

References


See Also

ESAGmle, spml.mle, acg, circ.summary

Examples

```r
x <- as.matrix(iris[,1:3])
x <- x / sqrt( rowSums(x^2) )
igm.mle(x)
```

Description

MLE of the von Mises-Fisher distribution.

Usage

`vmf(x, fast = FALSE, tol = 1e-07)`

Arguments

- **x**: A matrix or an FBM matrix (Filebacked Big Matrix, for big data) with the data expressed in Euclidean coordinates, i.e. unit vectors.
- **fast**: A boolean variable to do a slightly faster implementation. This does not support Filebacked Big Matrix.
- **tol**: The tolerance to accept that the E-M algorithm used to estimate the concentration parameter has converged.

Details

The mean direction and concentration of a fitted von Mises-Fisher distribution are estimated.
It estimates the parameters of the Wood bimodal distribution.

Usage

`wood.mle(y)`
MLE of the Wood bimodal distribution on the sphere

Arguments

y
A matrix containing two columns. The first one is the latitude and the second is the longitude, both expressed in degrees.

Details

The Wood distribution is fitted to some data and its parameters are estimated. It is a bimodal distribution which contains 5 parameters, just like the Kent distribution.

Value

A list including:

info
A 5 x 3 matrix containing the 5 parameters, gamma, delta, alpha, beta and kappa along with their corresponding 95% confidence intervals all expressed in degrees.

modes
The two axis of the modes of the distribution expressed in degrees.

unitvectors
A 3 x 3 matrix with the 3 unitvectors associated with the gamma and delta parameters.

loglik
The value of the log-likelihood.

Author(s)

Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

kent.mle, ESAGmle, vmf, rvmf, sphereplot

Examples

x <- rvmf(100, rnorm(3), 15)
x <- euclid.inv(x)
wood.mle(x)
Prediction in discriminant analysis based on ESAG distribution

Description
Prediction of a new observation using discriminant analysis based on ESAG distribution.

Usage
ESAGda.pred(ynew, y, ina)

Arguments
- ynew: The new observation(s) (unit vector(s)) whose group is to be predicted.
- y: A data matrix with unit vectors, i.e. spherical directional data.
- ina: A vector indicating the groups of the data y.

Details
Prediction of the class of a new spherical vector assuming ESAG distribution.

Value
A vector with the predicted group of each new observation.

Author(s)
Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>.

References

See Also
ESAG.da, vmfda.pred, dirknn, knn.reg
Prediction in discriminant analysis based on von Mises-Fisher distribution

Examples

```r
m1 <- rnorm(3)
m2 <- rnorm(3) + 0.5
y <- rbind( rvmf(100, m1, 3), rvmf(80, m2, 5) )
ina <- c( rep(1,100), rep(2, 80) )
ynew <- rbind(rvmf(10, m1, 10), rvmf(10, m2, 5))
id <- rep(1:2, each = 10)
g <- ESA Gda.pred(ynew, y, ina)
table(id, g)
```

Description

Prediction of the class of a new observation using discriminant analysis based on von Mises-Fisher distribution.

Usage

```r
vmfdaNpred(xnew, x, ina)
```

Arguments

- `xnew` The new observation(s) (unit vector(s)) whose group is to be predicted.
- `x` A data matrix with unit vectors, i.e. directional data.
- `ina` A vector indicating the groups of the data `x`.

Details

Discriminant analysis assuming von Mises-Fisher distributions.

Value

A vector with the predicted group of each new observation.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

Probability density function of the von Mises-Fisher distribution

See Also

vmf.da, mix.vmf, mix.vmf, dirknn, knn.reg

Examples

```r
m1 <- rnorm(5)
m2 <- rnorm(5)
x <- rbind(rvmf(100, m1, 5), rvmf(80, m2, 10))
ina <- c(rep(1,100), rep(2, 80))
y <- rbind(rvmf(10, m1, 10), rvmf(10, m2, 5))
id <- rep(1:2, each = 10)
g <- vmfda.pred(y, x, ina)
table(id, g)
```

Description

Probability density function of the von Mises-Fisher distribution.

Usage

```r
pvm(theta, m, k, rads = FALSE)
```

Arguments

- `theta`: A numerical value, either in radians or in degrees.
- `m`: The mean direction in radians or in degrees.
- `k`: The concentration parameter, $\kappa$.
- `rads`: If the data are in radians, this should be TRUE and FALSE otherwise.

Details

This value calculates the probability of $x$ being less than $\theta$ and is used by `group.gof`.

Value

The probability that of $x$ being less than $\theta$, where $x$ follows the von Mises-Fisher distribution.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>
Random sample of matrices in SO(p)

References


See Also

group.gof, circ.summary, rvonmises

Examples

pvm(1, 2, 10, rads = TRUE)
pvm(2, 2, 10, rads = TRUE)

Description

Random sample of matrices in SO(p).

Usage

rsop(n, p)

Arguments

n The sample size, the number of matrices you want to generate.
p The dimensionality of the matrices.

Details

The idea is very simple. Start with a unit vector pointing at the north pole (1,0,...,0). Then generate random numbers from a standard normal and scale them so that they have a unit length. To put it differently, a sample of n values from the uniform distribution on the sphere is generated. Then calculate the rotation matrix required to go from the north pole to each of a generated vector.

Value

If n = 1 one matrix is returned. If n is greater than 1, an array with n matrices inside.

Author(s)

Michail Tsagris R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>
Random values generation from the ESAG distribution

References

See Also
rotation, Arotation, rot.matrix

Examples

```r
x1 <- rsop(1, 3)
x2 <- rsop(10, 3)
x3 <- rsop(100, 10)
```

Description
Random values generation from the ESAG distribution.

Usage

```r
ESAGsim(n, param)
```

Arguments

- `n`: A number; how many vectors you want to generate.
- `param`: The parameters of the ESAG distribution, the first three are the mean vector in $\mathbb{R}^3$ and the next two are the two gammas.

Details
A random sample from the ESAG distribution is generated. In case the gammas are zero the sample is drawn from Independent Angular Gaussian (IAG) or projected normal.

Value
An nx3 matrix with the simulated unit vectors.

Author(s)
Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>
Random values simulation from some circular distributions

References


See Also

ESAGmle, ESAGdensity, spml.mle, acg, circ.summary

Examples

```r
m <- colMeans(as.matrix(iris[,1:3]))
y <- ESAGsim(1000, c(m, 1, 0.5))
ESAGmle(y)
```

Description

Random values simulation from some circular distributions.

Usage

```r
rvonmises(n, m, k, rads = TRUE)
rwrapcauchy(n, m, rho, rads = TRUE)
```

Arguments

- `n`: The sample size.
- `m`: The mean angle expressed in radians or degrees.
- `k`: The concentration parameter of the von Mises distribution. If k is zero the sample will be generated from the uniform distribution over $$(0, 2\pi)$$. 
- `rho`: The $\rho$ parameter of the Wrapped Cauchy distribution.
- `rads`: If the mean angle is expressed in radians, this should be TRUE and FALSE otherwise. The simulated data will be expressed in radians or degrees depending on what the mean angle is expressed.

Details

For the von Mises distribution, the mean direction is transformed to the Euclidean coordinates (i.e. unit vector) and then the rvmf function is employed. It uses a rejection sampling as suggested by Andrew Wood in 1994. I have mentioned the description of the algorithm as I found it in Dhillon and Sra in 2003. Finally, the data are transformed to radians or degrees.

For the wrapped Cauchy distribution the function generates Cauchy values $x$ and then wraps around the circle $x = x(\mod 2\pi)$. For the circular beta the function has some extra steps (see Zheng Sun’s master thesis).
Value

A vector with the simulated data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

circ.summary, rvmf, racg

Examples

```r
x <- rvonmises(100, 2, 25, rads = TRUE)
circ.summary(x, rads = TRUE)
```

Description

It checkes whether the data are uniformly distributed on the sphere or hypersphere.

Usage

```r
rayleigh(x, modif = TRUE, B = 999)
```

Arguments

- `x` A matrix containing the data, unit vectors.
- `modif` If `modif` is `TRUE`, the modification as suggested by Jupp (2001) is used.
- `B` If `B` is greater than 1, bootstrap calibration os performed. If it is equal to 1, classical theory is used.
Details

The Rayleigh test of uniformity is not the best, when there are two antipodal mean directions. In this case it will fail. It is good to test whether there is one mean direction or not. To put it differently, it tests whether the concentration parameter of the Fisher distribution is zero or not.

Value

A vector including:

test
The value of the test statistic.

p-value or Bootstrap p-value
The (bootstrap) p-value of the test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

vmf, meandir.test, acg

Examples

x <- rvmf(100, rnorm(5), 1)  ## Fisher distribution with low concentration
rayleigh(x)
Rotation axis and angle of rotation given a rotation matrix

Usage

read.fbm(file, select)

Arguments

file

The File to read.

select

Indices of columns to read (sorted). The length of select will be the number of columns of the resulting FBM.

Details

The functions read a file as a Filebacked Big Matrix object. For more information see the "bigstatsr" package.

Value

A Filebacked Big Matrix object.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

See Also

vmf, kent.mle

Examples

## Not run:
dataset <- matrix( runif(100 * 50, 1, 100), ncol = 50 )
write.csv(dataset, "dataset.csv")
a <- read.fbm("dataset.csv", select = 1:50)
## End(Not run)

Description

Given a 3 x 3 rotation matrix, the angle and the axis of rotation are calculated.

Usage

Arotation(A)
Rotation matrix from a rotation axis and angle of rotation

Arguments

A  A 3 x 3 rotation matrix.

Details

If the user does not supply a rotation matrix a message will appear.

Value

A list including:

angle  The angle of rotation expressed in degrees.
axis  The axis of rotation. A vector of two components, latitude and longitude, ex-
       pressed in degrees.

Author(s)

Michail Tsagris R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and
Giorgos Athineou <gioathineou@gmail.com>

References


See Also

rot.matrix, rotation, rsop

Examples

ksi <- c(25.31, 24.29)
theta <- 2.38
A <- rot.matrix(ksi, theta, rads = FALSE)
A
A rotation(A)

Description

It calculates a rotation matrix from a rotation axis and angle of rotation.

Usage

rot.matrix(ksi, theta, rads = FALSE)
Rotation matrix from a rotation axis and angle of rotation

Arguments

ksi
The rotation axis, a vector with two elements, the first is the latitude and the second is the longitude.

theta
The angle of rotation.

rads
If both the ksi and theta are in rads, this should be TRUE. If both the ksi and theta are in degrees, this should be FALSE.

Details

The function accepts as arguments the rotation axis and the angle of rotation and it calculates the requested rotation matrix.

Value

A 3 x 3 rotation matrix.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

Arotation, rotation, rsop

Examples

ksi <- c(25.31, 24.29)
theta <- 2.38
A <- rot.matrix(ksi, theta, rads = FALSE)
A
Arotation(A)
Rotation matrix on SO(3) from three Euler angles

Construct a rotation matrix on SO(3) from the Euler angles.

Description

It forms a rotation matrix X on SO(3) by using three Euler angles \((\theta_{12}, \theta_{13}, \theta_{23})\), where X is defined as

\[
X = R_z(\theta_{12}) \times R_y(\theta_{13}) \times R_x(\theta_{23}).
\]

Here \(R_x(\theta_{23})\) means a rotation of \(\theta_{23}\) radians about the x axis.

Usage

\[\text{eul2rot(theta.12, theta.23, theta.13)}\]

Arguments

\begin{itemize}
  \item theta.12 An Euler angle, a number which must lie in \((-\pi, \pi)\).
  \item theta.23 An Euler angle, a number which must lie in \((-\pi, \pi)\).
  \item theta.13 An Euler angle, a number which must lie in \((-\pi/2, \pi/2)\).
\end{itemize}

Details

Given three euler angles a rotation matrix X on SO(3) is formed using the transformation according to Green and Mardia (2006) which is defined above.

Value

A rotation matrix.

Author(s)

Anamul Sajib<pmxahsa@nottingham.ac.uk>

R implementation and documentation: Anamul Sajib<pmxahsa@nottingham.ac.uk>

References


See Also

\[\text{rot2eul}\]
Rotation matrix to rotate a spherical vector along the direction of another

Examples

# three euler angles

```r	heta.12 <- sample( seq(-3, 3, 0.3), 1 )
theta.23 <- sample( seq(-3, 3, 0.3), 1 )
theta.13 <- sample( seq(-1.4, 1.4, 0.3), 1 )

theta.12 ; theta.23 ; theta.13

X <- eul2rot(theta.12, theta.23, theta.13)
X # A rotation matrix
det(X)

e <- rot2eul(X)$v1

theta.12 <- e[3]
theta.23 <- e[2]
theta.13 <- e[1]

theta.12 ; theta.23 ; theta.13
```

Description

A rotation matrix is calculated to rotate a unit vector along the direction of another.

Usage

`rotation(a, b)`

Arguments

- `a` The initial unit vector.
- `b` The target unit vector.

Details

The function calculates a rotation matrix given two vectors. This rotation matrix is the connection between the two spherical only, vectors.

Value

A rotation matrix whose dimension is equal to the length of the unit vectors.
Saddlepoint approximations of the Fisher-Bingham distributions

Author(s)
Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

See Also
Arotation, rot.matrix, lambert, lambert.inv, rsop

Examples
a <- rnorm(3)
a <- a/sqrt(sum(a^2))
b <- rnorm(3)
b <- b/sqrt(sum(b^2))
A <- rotation(a, b)
A
a ; b
a%% t(A)

a <- rnorm(7)
a <- a/sqrt(sum(a^2))
b <- rnorm(7)
b <- b/sqrt(sum(b^2))
A <- rotation(a, b)
A
a ; b
a%% t(A)

Saddlepoint approximations of the Fisher-Bingham distributions

Description
It calculates the logarithm of the normalising constant of the Fisher-Bingham distribution.

Usage
fb.saddle(gam, lam)
Arguments

\texttt{gam} 
A numeric vector containing the parameters of the Fisher part.

\texttt{lam} 
All the eigenvalues of the Bingham part. Not just the non zero ones.

Details

It calculate the three approximations given by Kume and Wood (2005) and it uses the Fisher-Bingham parametrization of that paper.

Value

A list including:

- \texttt{first oder} 
The first order approximation
- \texttt{second oder} 
The second order approximation
- \texttt{third oder} 
The third order approximation

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

\texttt{kent.logcon}, \texttt{rfb}, \texttt{kent.mle}, \texttt{rbingham}

Examples

```r
p <- 3 ; k <- 1
0.5 * p * log(2 * pi) - (p/2 - 1) * log(k) + log( besselI(k, p/2 - 1, expon.scaled = TRUE) ) + k
## normalising constant of the
## von Mises-Fisher distribution
fb.saddle( c(0, k, 0), c(0, 0, 0) ) ## saddlepoint approximation

## Normalising constant of the Kent distribution
fb.saddle( c(0, 10, 0), c(0, -2, 2) )
kent.logcon(10, 2)
```
Simulation from a Bingham distribution using any symmetric matrix $A$

Description

It simulates random values from a Bingham distribution with any given symmetric matrix.

Usage

```r
rbingham(n, A)
```

Arguments

- `n`: The sample size.
- `A`: A symmetric matrix.

Details

The eigenvalues are first calculated and then Chris Fallaize and Theo Kypraios's code (f.rbing) is used. The resulting simulated data are then right multiplied by the eigenvectors of the matrix $A$.

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

- `f.rbing`, `rfb`, `rvmf`, `rkent`

Examples

```r
A <- cov(iris[, 1:3])
x <- rbingham(100, A)
```
Description

It simulates random samples (rotation matrices) from a Matrix Fisher distribution with any given parameter matrix, $F$ (3x3).

Usage

```r
rmatrixfisher(n, F)
```

Arguments

- `n` the sample size.
- `F` An arbitrary 3x3 matrix.

Details

Firstly corresponding Bingham parameter $A$ is determined for a given Matrix Fisher parameter $F$ using John Kent (2013) algorithm and then Bingham samples for parameter $A$ are generated using `rbingham` code. Finally convert Bingham samples to Matrix Fisher samples according to the Kent (2013) transformation.

Value

An array with simulated rotation matrices.

Author(s)

Anamul Sajib & Chris Fallaize

R implementation and documentation: Anamul Sajib <pmxahsa@nottingham.ac.uk> & Chris Fallaize

References


Examples

```r
F <- matrix(c(85, 11, 41, 78, 39, 60, 43, 64, 48), ncol = 3) / 10  ### An arbitrary F matrix
a <- rmatrixfisher(10, F)
```
Simulation of random values from a Bingham distribution

Description

It simulates from a Bingham distribution using the code suggested by Kent et al. (2013).

Usage

f.rbing(n, lam, fast = FALSE)

Arguments

n
Sample size.

lam
Eigenvalues of the diagonal symmetric matrix of the Bingham distribution.

fast
If you want a fast, efficient simulation set this to TRUE.

Details

The user must have calculated the eigenvalues of the diagonal symmetric matrix of the Bingham distribution. The function accepts the q-1 eigenvalues only. This means, that the user must have subtracted the lowest eigenvalue from the rest and give the non zero ones. The function uses rejection sampling and it was written by Chris Fallaize and Theo Kypraios (University of Nottingham) and kindly offered. Any questions on the code can be addressed to one of the two aforementioned people. It is slightly different than the one Kent et al. (2013) suggests.

Value

A list including:

x
The simulated data.

avtry
The estimate of M in the rejection sampling. The average number of simulated values before a value is accepted. If the argument fast is set to TRUE this information will not appear.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris &lt;mtsagris@yahoo.gr&gt; and Giorgos Athineou &lt;gioathineou@gmail.com&gt;

References


Simulation of random values from a mixture of von Mises-Fisher distributions

See Also

rfb, rvmf, rbingham, rkent, link{rsop}

Examples

```r
x <- f.rbingm(100, c(1, 0.6, 0.1))
x
```

Description

The function simulates random values simulation from a given mixture of von Mises-Fisher distributions.

Usage

`rmixvmf(n, prob, mu, k)`

Arguments

- `n`: The sample size.
- `prob`: This is a vector with the mixing probability of each group.
- `mu`: A matrix with the mean direction of each group.
- `k`: A vector with the concentration parameter of each group.

Details

The function simulates random values simulation from a given mixture of von Mises-Fisher distributions using the rvmf function.

Value

A list including:

- `id`: An indicator of the group of each simulated vector.
- `x`: A matrix with the simulated data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>
Simulation of random values from a spherical Fisher-Bingham distribution

References

See Also
mix.vmf, rvmf, rvmf, bic.mixvmf

Examples

```r
k <- runif(4, 4, 20)
prob <- c(0.2, 0.4, 0.3, 0.1)
mu <- matrix(rnorm(16), ncol = 4)
mu <- mu / sqrt(rowSums(mu^2))
x <- rmixvmf(200, prob, mu, k)$x
bic.mixvmf(x, 5)
```

Simulation of random values from a spherical Fisher-Bingham distribution

Description
Simulation of random values from a spherical Fisher-Bingham distribution.

Usage

```r
rfb(n, k, m, A)
```

Arguments

- `n` The sample size.
- `k` The concentration parameter (Fisher part). It has to be greater than 0.
- `m` The mean direction (Fisher part).
- `A` A symmetric matrix (Bingham part).

Details
Random values from a spherical Fisher-Bingham distribution are generated. This function includes the option of simulating from a Kent distribution also.

Value
A matrix with the simulated data.
Simulation of random values from a spherical Kent distribution

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

rbingham, rvmf, rkent, f.rbing

Examples

```r
k <- 15
mu <- rnorm(3)
mu <- mu / sqrt(sum(mu^2))
A <- cov(iris[, 1:3])
x <- rfb(50, k, mu, A)
vmf(x) # fits a von Mises-Fisher distribution to the simulated data
```

```r
# Next we simulate from a Kent distribution
n <- 100
x <- rfb(n, k, mu, A) # data follow a Kent distribution
kent.mle(x) # gives a Kent distribution
vmf(x) # fits a von Mises-Fisher distribution
```

```r
A <- diag(c(-5, 0, 5))
n <- 100
x <- rfb(n, k, mu, A) # data follow a Kent distribution
kent.mle(x) # gives a Kent distribution
vmf(x) # fits a von Mises-Fisher distribution
```

Description

Simulation of random values from a spherical Kent distribution.

Usage

rkent(n, k, m, b)
Simulation of random values from a spherical Kent distribution

Arguments

- **n**: The sample size.
- **k**: The concentration parameter $\kappa$. It has to be greater than 0.
- **m**: The mean direction (Fisher part).
- **b**: The ovalness parameter, $\beta$.

Details

Random values from a Kent distribution on the sphere are generated. The function generates from a spherical Kent distribution using `rfb` with an arbitrary mean direction and then rotates the data to have the desired mean direction.

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

`rfb`, `rbingham`, `rvmf`, `fRbinging`

Examples

```r
k <- 15
mu <- rnorm(3)
mu <- mu / sqrt(sum(mu^2))
A <- diag(c(-5, 0, 5))
x <- rfb(500, k, mu, A)
kent.mle(x)
y <- rkent(500, k, mu, A[3, 3])
kent.mle(y)
```
Simulation of random values from a von Mises-Fisher distribution

Random values simulation from a von Mises-Fisher distribution

Description

It generates random vectors following the von Mises-Fisher distribution. The data can be spherical or hyper-spherical.

Usage

rvmf(n, mu, k)

Arguments

n
The sample size.

mu
The mean direction.

k
The concentration parameter. If k = 0, random values from the spherical uniform will be drawn. Values from a multivariate normal distribution with zero mean vector and the identity matrix as the covariance matrix. Then each vector becomes a unit vector.

Details

It uses a rejection sampling as suggested by Andrew Wood (1994).

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

vmf, rfb, racg, rvonmises, rmixvmf
Examples

```r
m <- rnorm(4)
m <- m/sqrt(sum(m^2))
x <- rvmf(100, m, 25)
m
vmf(x)
```

Spherical and hyperspherical median

*Fast calculation of the spherical and hyperspherical median*

Description

It calculates, very fast, the (hyper-)spherical median of a sample.

Usage

```r
mediandir(x)
mediandir_2(x)
```

Arguments

- **x**: The data, a numeric matrix with unit vectors.

Details

The "mediandir" employs a fixed point iterative algorithm stemming from the first derivative (Cabrera and Watson, 1990) to find the median direction as described by Fisher (1985) and Fisher, Lewis and Embleton (1987). In the big samples this is much much faster than "mediandir_2", since the search is based on iterations.

Value

The median direction.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


Spherical regression using the projected normal or the von Mises-Fisher distribution

See Also

vmf, kent.mle

Examples

m <- rnorm(3)
m <- m / sqrt(sum(m^2))
x <- rvmf(100, m, 10)
mediandir(x)
mediandir_2(x)

Spherical regression using the projected normal or the von Mises-Fisher distribution

Description

Spherical regression using the projected normal or the von Mises-Fisher distribution.

Usage

iag.reg(y, x, con = TRUE, xnew = NULL, tol = 1e-06)
vmf.reg(y, x, con = TRUE, xnew = NULL, tol = 1e-06)

Arguments

y A matrix with 3 columns containing the (unit vector) spherical data.
x The predictor variable(s), they can be continuous, spherical, categorical or a mix of them.
con Do you want the constant term in the regression?
xnew If you have new data use it, otherwise leave it NULL.
tol A tolerance value to decide when to stop the successive optimalizations.

Details

The second parametrization of the projected normal and of the von Mises-Fisher regression (Paine et al., 2019) is applied. For more information see the paper by Paine et al. (2019).

Value

A list including:

loglik The log-likelihood of the regression model.
fit This is a measure of fit of the estimated values, defined as $\sum_{i=1}^{n} y_i^T \hat{y}_i$. This appears if the argument "xnew" is NULL.
Spherical-spherical correlation

**Description**

Correlation between two spherical variables.

**Usage**

`spher.cor(x, y)`

**Arguments**

- `x` A spherical variable. A matrix with three columns, each row is a unit vector.
- `y` A spherical variable. A matrix with three columns, each row is a unit vector.
Spherical-spherical regression

Details

A very similar to the classical correlation is calculated. In addition, a hypothesis test for no correlation is performed. Note, that this is a squared correlation actually, so negative values will never be returned.

Value

A vector including:

R-squared
The value of the squared correlation.

p-value
The p-value of the no correlation hypothesis testing.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

spher.reg, vmf, circ.cor1, circ.cor2

Examples

x <- rvmf(100, rnorm(3), 10)
y <- rvmf(100, rnorm(3), 10)
spher.cor(x, y)
**Arguments**

- **y**
  The dependent variable; a matrix with either two columns, latitude and longitude, either in radians or in degrees. Alternatively it is a matrix with three columns, unit vectors.

- **x**
  The dependent variable; a matrix with either two columns, latitude and longitude, either in radians or in degrees. Alternatively it is a matrix with three columns, unit vectors. The two matrices must agree in the scale and dimensions.

- **rads**
  If the data are expressed in latitude and longitude then it matters to know if they are in radians or degrees. If they are in radians, then this should be TRUE and FALSE otherwise. If the previous argument, euclidean, is TRUE, this one does not matter what its value is.

**Details**

Spherical regression as proposed by Chang (1986) is implemented. If the estimated rotation matrix has a determinant equal to -1, singular value decomposition is performed and the last unit vector of the second matrix is multiplied by -1.

**Value**

A list including:

- **A**
  The estimated rotation matrix.

- **fitted**
  The fitted values in Euclidean coordinates (unit vectors).

**Author(s)**

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

**References**


**See Also**

- `spher.cor`, `spml.reg`, `circ.cor1`, `circ.cor2`, `sphereplot`

**Examples**

```r
mx <- rnorm(3)
x <- mx/sqrt( sum(mx^2) )
my <- rnorm(3)
my <- my/sqrt( sum(my^2) )
x <- rvmf(100, mx, 15)
A <- rotation(mx, my)
y <- x %*% t(A)
```
Summary statistics for circular data

Description

It produces a few summary measures for circular data.

Usage

```
circ.summary(u, rads = FALSE, fast = FALSE, tol = 1e-09, plot = TRUE)
```

Arguments

- `u`: A vector with circular data.
- `rads`: If the data are in rads, then this should be TRUE, otherwise FALSE.
- `fast`: A boolean variable to do a faster implementation.
- `tol`: The tolerance level to stop the Newton-Raphson algorithm for finding kappa.
- `plot`: If you want to see the data plotted on a circle make this TRUE.

Details

It returns the circular mean, mean resultant length, variance, standard deviation and concentration parameter. So, basically it returns the estimated values of the parameters of the von Mises distribution.

Value

If `fast = FALSE` a list including all the following. If `fast = TRUE` less items are returned.

- `mesos`: The circular mean direction.
- `confint`: The 95% confidence interval for the circular mean direction.
- `kappa`: The concentration parameter.
- `MRL`: The mean resultant length.
- `circvariance`: The circular variance.
- `circstd`: The circular standard deviation.
- `loglik`: The log-likelihood of the fitted von Mises distribution.
Summary statistics for grouped circular data

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

rvonmises, vm.kde, vmf, group.vm, hcf.circaov

Examples

x <- rvonmises(50, 2.5, 15, rads = TRUE)
circ.summary(x, rads = TRUE, plot = TRUE)

Summary statistics for grouped circular data

Summary statistics for grouped circular data

Description

It produces a few summary measures for grouped circular data.

Usage

group.vm(group, fi, rads = FALSE)

Arguments

group A matrix denoting the classes. Each row consists of two numbers, the lower and upper points of each class.
fi The frequency of each class of data.
rads If the data are in rads, then this should be TRUE, otherwise FALSE.

Details

It returns the circular mean, mean resultant length, variance, standard deviation and concentration parameter. So, basically it returns the estimated values of the parameters of the von Mises distribution. The mean resultant length though is group corrected.
Summary statistics for grouped circular data

Value

A list including:

- **mesos**: The circular mean direction.
- **confint**: The 95% confidence interval for the circular mean direction.
- **kappa**: The concentration parameter.
- **MRL**: The mean resultant length.
- **circvariance**: The circular variance.
- **circstd**: The circular standard deviation.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

- `circ.summary`, `rvonmises`, `vm.kde`

Examples

```r
x <- rvonmises(200, 3, 10)
a <- circ.summary(x, rads = TRUE, plot = FALSE)
group <- seq(min(x) - 0.1, max(x) + 0.1, length = 6)
y <- cut(x, breaks = group, length = 6)
group <- matrix(c(group[1], rep(group[2:5], each = 2), group[6]), ncol = 2, byrow = TRUE)
fi <- as.vector(table(y))
b <- group.vm(group, fi, rads = TRUE)
a
b
```
Test for a given mean direction

Description

A log-likelihood ratio test for testing whether the sample mean direction is equal to some predefined one.

Usage

meandir.test(x, mu, B = 999)

Arguments

x     A matrix with the data, unit vectors.
mu    A unit vector with the hypothesized mean direction.
B     A number either 1, so no bootstrap calibration is performed or more than 1, so bootstrap calibration is performed.

Details

The log-likelihood ratio test is performed.

Value

A list including:

mean.dir    The sample mean direction
pvalue      The p-value of the test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

vmf, kent.mle, rayleigh
Test for equality of concentration parameters for spherical data

Examples

```r
mu <- rnorm(5)
mu <- mu / sqrt(sum(mu^2))
x <- rvmf(100, mu, 10)
meandir.test(x, mu, 1)
meandir.test(x, mu, 499)
```

Description

This tests the equality of concentration parameters for spherical data only.

Usage

```r
spherconc.test(x, ina)
```

Arguments

- `x` A matrix with the data in Euclidean coordinates, i.e. unit vectors
- `ina` A variable indicating the groupings of the observations.

Details

The test is designed for spherical data only.

Value

A list including:

- `mess` A message stating the value of the mean resultant and which test statistic was used, U1, U2 or U3.
- `res` A vector containing the test statistic and its p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

Test of equality of the concentration parameters for circular data

See Also
  het.aov, lr.aov, embed.aov, hcf.aov, conc.test, sphereplot

Examples

```r
x <- rvmf(100, rnorm(3), 15)
ina <- rep(1:4, each = 25)
spherconc.test(x, ina)
```

Description

A test for testing the equality of the concentration parameter among g samples, where g \( \geq 2 \) for circular data.

Usage

```
conc.test(u, ina, rads = FALSE)
```

Arguments

- **u**: A numeric vector containing the values of all samples.
- **ina**: A numerical variable or factor indicating the groups of each value.
- **rads**: If the data are in radians this should be TRUE and FALSE otherwise.

Details

This test works for circular data.

Value

A list including:

- **mess**: A message informing the use of the test statistic used.
- **res**: A numeric vector containing the value of the test statistic and its associated p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>
Tuning of the bandwidth parameter in the von Mises kernel

References


See Also

embed.circaov, hcf.circaov, lr.circaov, het.circaov

Examples

```r
x <- rvonmises(100, 2.4, 15)
ina <- rep(1:4,each = 25)
conc.test(x, ina, rads = TRUE)
```

Tuning of the bandwidth parameter in the von Mises kernel

Tuning of the bandwidth parameter in the von Mises kernel for circular data

Description

Tuning of the bandwidth parameter in the von Mises kernel for circular data. Cross validation is used.

Usage

`vmkde.tune(u, low = 0.1, up = 1, rads = TRUE)`

Arguments

- `u` The data, a numerical vector.
- `low` The lower value of \( h \) to search.
- `up` The lower value of \( h \) to search.
- `rads` If the data are in radians this should be TRUE and FALSE otherwise.

Details

Tuning of the bandwidth parameter in the von Mises kernel for circula data via cross validation. The procedure is fast because an optimiser is used.

Value

A vector including two elements:

- `Optimal h` The best \( H \) found.
- `cv` The value of the maximised pseudo-likelihood.
Tuning of the bandwidth parameter in the von Mises-Fisher kernel

Author(s)
Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

See Also
vm.kde, vmfkde.tune, vmf.kde

Examples
u <- rvmnives(100, 2.4, 10, rads = TRUE)
vmkde.tune(u)

Description
Tuning of the bandwidth parameter in the von Mises-Fisher kernel for (hyper-)spherical data.

Usage
vmfkde.tune(x, low = 0.1, up = 1)

Arguments
x A matrix with the data in Euclidean cordinates, i.e. unit vectors.
low The lower value of the bandwidth to search.
up The upper value of the bandwidth to search.

Details
Fast tuning of the bandwidth parameter in the von Mises-Fisher kernel for (hyper-)spherical data via cross validation.
Tuning of the k-NN algorithm using the arc cosinus distance

Value

A vector including two elements:

- `optimal h` The best H found.
- `cv` The value of the maximised pseudo-likelihood.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

`vmf.kde`, `vmf.kerncontour`, `vm.kde`, `vmkde.tune`

Examples

```r
x <- rvmf(100, rnorm(3), 15)
vmfkde.tune(x)
```

Description

It estimates the percentage of correct classification via an m-fold cross validation. The bias is estimated as well using the algorithm suggested by Tibshirani and Tibshirani (2009) and is subtracted.

Usage

```r
dirknn.tune(z, nfolds = 10, A = 5, ina, type = "S", mesos = TRUE, folds = NULL, parallel = FALSE, stratified = TRUE, seed = FALSE)
```
Tuning of the k-NN algorithm using the arc cosinus distance

Arguments

- **z**: The data, a numeric matrix with unit vectors.
- **nfolds**: How many folds to create?
- **A**: The maximum number of nearest neighbours, set to 5 by default. The 1 nearest neighbour is not used.
- **ina**: A variable indicating the groups of the data x.
- **type**: If type is "S", the standard k-NN algorithm is to be used, else "NS" for the non standard one. See below (details) for more information.
- **mesos**: A boolean variable used only in the case of the non standard algorithm (type="NS"). Should the average of the distances be calculated (TRUE) or not (FALSE)? If it is FALSE, the harmonic mean is calculated.
- **folds**: Do you already have a list with the folds? If not, leave this NULL.
- **parallel**: If you want the standard -NN algorithm to take place in parallel set this equal to TRUE.
- **stratified**: Should the folds be created in a stratified way? i.e. keeping the distribution of the groups similar through all folds?
- **seed**: If seed is TRUE, the results will always be the same.

Details

The standard algorithm is to keep the k nearest observations and see the groups of these observations. The new observation is allocated to the most frequent seen group. The non standard algorithm is to calculate the classical mean or the harmonic mean of the k nearest observations for each group.

We have made an efficient (not very much efficient though) memory allocation. Even if you have hundreds of thousands of observations, the computer will not clush, it will only take longer. Instead of calculate the distance matrix once in the beginning we calculate the distances of the out-of-sample observations from the rest. If we calculated the distance matrix in the beginning, once, the resulting matrix could have dimensions thousands by thousands. This would not fit into the memory. If you have a few hundred of observations, the runtime is about the same (maybe less, maybe more) as calculating the distance matrix in the first place.

Value

A list including:

- **per**: The average percent of correct classification across the neighbours.
- **percent**: The bias corrected percent of correct classification.
- **runtime**: The run time of the algorithm. A numeric vector. The first element is the user time, the second element is the system time and the third element is the elapsed time.

Author(s)

Michail Tsagris R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>.
Tuning of the k-NN regression

See Also
dirknn, vmf.da, mix.vmf

Examples

```r
k <- runif(4, 4, 20)
prob <- c(0.2, 0.4, 0.3, 0.1)
mu <- matrix(rnorm(16), ncol = 4)
mu <- mu / sqrt(rowSums(mu^2))
da <- rmixvMF(200, prob, mu, k)
x <- da$x
ina <- da$id
dirknn.tune(x, nfolds = 5, A = 10, ina, type = "S", mesos = TRUE)
dirknn.tune(x, nfolds = 10, A = 5, ina, type = "S", mesos = TRUE)
```

Description

Tuning of the k-NN regression with Euclidean or (hyper-)spherical response and or predictor variables. It estimates the percentage of correct classification via an m-fold cross validation. The bias is estimated as well using the algorithm suggested by Tibshirani and Tibshirani (2009) and is subtracted.

Usage

```r
knnreg.tune(y, x, nfolds = 10, A = 10, ncores = 1, res = "eucl", type = "euclidean", estim = "arithmetic", folds = NULL, seed = FALSE, graph = FALSE)
```

Arguments

- **y**: The currently available data, the response variables values. A matrix with either euclidean (univariate or multivariate) or (hyper-)spherical data. If you have a circular response, say u, transform it to a unit vector via (cos(u), sin(u)).

- **x**: The currently available data, the predictor variables values. A matrix with either euclidean (univariate or multivariate) or (hyper-)spherical data. If you have a circular response, say u, transform it to a unit vector via (cos(u), sin(u)).

- **nfolds**: How many folds to create?

- **A**: The maximum number of nearest neighbours, set to 5 by default, starting from the 1 nearest neighbor.

- **ncores**: How many cores to use. This is taken into account only when the predictor variables are spherical.
Tuning of the k-NN regression

res
The type of the response variable. If it is Euclidean, set this argument equal to "res". If it is a unit vector set it to res="spher".

type
The type of distance to be used. This determines the nature of the predictor variables. This is actually the argument "method" of the distance function in R. The default value is "euclidean". R has several options the type of the distance. Just type ?dist in R and see the methods. Any method can be given here. If you have unit vectors in general, you should put type="spher", so that the cosinus distance is calculated.
estim
Once the k observations whith the smallest distance are discovered, what should the prediction be? The arithmetic average of the corresponding y values be used estim="arithmetic" or their harmonic average estim="harmonic".
folds
Do you already have a list with the folds? If not, leave this NULL.
seed
If seed is TRUE, the results will always be the same.
graph
If this is TRUE a graph with the results will appear.

Details
Tuning of the k-NN regression with Euclidean or (hyper-)spherical response and or predictor variables. It estimates the percentage of correct classification via an m-fold cross validation. The bias is estimated as well using the algorithm suggested by Tibshirani and Tibshirani (2009) and is subtracted. The sum of squares of prediction is used in the case of Euclidean responses. In the case of spherical responses the $\sum \hat{y}^T y_i$ is calculated.

Value
A list including:
crit
The value of the criterion to minimise/maximise for all values of the nearest neighbours.

best_k
The best value of the nearest neighbours.

performance
The bias corrected optimal value of the criterion, along wit the estimated bias. For the case of Euclidean response this will be higher than the crit and for the case or spherical responses it will be lower than crit.

runtime
The run time of the algorithm. A numeric vector. The first element is the user time, the second element is the system time and the third element is the elapsed time.

Author(s)
Michail Tsagris R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

See Also

knn.reg, spher.reg, dirknn.tune
Examples

```r
y <- iris[, 1]
x <- iris[, 2:4]
x <- x / sqrt(rowSums(x^2))  ## Euclidean response and spherical predictors
knnreg.tune(y, x, A = 5, res = "eucl", type = "spher", estim = "arithmetic")

y <- iris[, 1:3]
y <- y / sqrt(rowSums(y^2))  ## Spherical response and Euclidean predictor
x <- iris[, 2]
knnreg.tune(y, x, A = 5, res = "spher", type = "euclidean", estim = "arithmetic")
```

Description

Hypothesis tests of uniformity for circular data.

Usage

```r
kuiper(u, rads = FALSE, R = 1)

watson(u, rads = FALSE, R = 1)
```

Arguments

- `u`: A numeric vector containing the circular data, which can be expressed in degrees or radians.
- `rads`: A boolean variable. If the data are in radians, put this TRUE. If the data are expressed in degrees make this FALSE.
- `R`: If `R = 1` the asymptotic p-value will be calculated. If `R` is greater than 1 the bootstrap p-value is returned.

Details

The high concentration (hcf.circaov), log-likelihood ratio (lr.circaov), embedding approach (embed.circaov) or the non equal concentration parameters approach (het.circaov) is used.

Value

A vector including:

- **Test**: The value of the test statistic.
- **p-value**: The p-value of the test (bootstrap or asymptotic depends upon the value of the argument `R`).
Unit vector(s) with a given angle

Author(s)
Michail Tsagris
R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References

See Also
rayleigh, vmf, rvmnises

Examples

```r
x <- rvmnises(n = 40, m = 2, k = 10)
kiper(x, rads = TRUE)
watson(x, rads = TRUE)
x <- rvmnises(40, m = 2, k = 0)
kiper(x, rads = TRUE)
watson(x, rads = TRUE)
```

Description
Unit vector with a given angle from a given unit vector.

Usage

```r
vec(x, n = 1, deg = 90)
```

Arguments

- **x**: A unit vector. If it is not a unit vector it becomes one.
- **n**: The number of unit vectors to return.
- **deg**: The angle between the given vector and the n vectors to be returned. This must be in degrees and it has to be between 0 and 180 degrees. If the angle is 0, the same unit vector will be returned. If the angle is 180, the same unit vector with the signs changed will be returned.
von Mises kernel density estimation

Details

The user provides a unit vector and the degrees. The function will return n unit vectors whose angle with the given unit vector equals the degrees given. For example, if you want 10 unit vectors perpendicular to the x put vec(x, 10, 90).

Value

A list including:

- runtime The runtime of the procedure.
- crit The calculated angle between the given unit vector and each of the generated unit vectors.
- mat A matrix with the n unit vectors.

Author(s)

Michail Tsagris R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

See Also

rvmf, rbingham, rfb

Examples

```r
x <- rnorm(10)
x <- x / sqrt(sum(x^2))
a <- vec(x, 20, 90)
```

Description

Kernel density estimation of circular data with a von Mises kernel.

Usage

```r
vm.kde(u, h, thumb = "none", rads = TRUE)
```

Arguments

- `u` A numeric vector containing the data.
- `h` The bandwidth.
- `thumb` It can be either "none", so the bandwidth the user has set will be used. "tay" for the method of Taylor (2008) or "rot" for the method of Garcia-Portugues (2013).
- `rads` If the data are in radians, this should be TRUE and FALSE otherwise.
Details

The user has the option to use a bandwidth he/she has found in some way (cross-validation) or estimate it as Taylor (2008) or Garcia-Portugues (2013).

Value

A list including:

- **h** The bandwidth. If the user chose one of "tay" or "rot" the estimated bandwidth will be returned.
- **f** The kernel density estimate at the observed points.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou<gioathineou@gmail.com>

References


See Also

vmkde.tune, vmfkde.tune, vmf.kde

Examples

```r
x <- rvonmises(100, 2.4, 10, rads = TRUE)
hist(x, freq = FALSE)
f1 <- vm.kde(x, h = 0.1, thumb = "rot", rads = TRUE)$f
f2 <- vm.kde(x, h = 0.1, thumb = "tay", rads = TRUE)$f
h <- vmkde.tune(x)[1]
f3 <- vm.kde(x, h = h, thumb = "none", rads = TRUE)$f
points(x, f1, col = 1)
points(x, f2, col = 2)
points(x, f3, col = 3)
```
A von Mises-Fisher kernel is used for the non parametric density estimation.

Usage

```r
vmf.kde(x, h, thumb = "none")
```

Arguments

- `x`: A matrix with unit vectors, i.e. the data being expressed in Euclidean coordinates.
- `h`: The bandwidth to be used.
- `thumb`: If this is "none", the given bandwidth is used. If it is "rot" the rule of thumb suggested by Garcia-Portugues (2013) is used.

Details

A von Mises-Fisher kernel is used for the non parametric density estimation.

Value

A list including:

- `h`: The bandwidth used.
- `f`: A vector with the kernel density estimate calculated for each of the data points.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <gioathineou@gmail.com>

References


See Also

`vmfkde.tune`, `vm.kde`, `vmf`, `vmkde.tune`
Examples

```r
x <- rvmf(100, rnorm(5), 15)
h <- vmfkde.tune(x)[1]
f1 <- vmf.kde(x, h = h, thumb = "none")
f2 <- vmf.kde(x, h = h, thumb = "rot")
f1$h ; f2$h
```
Index

*Topic (Hyper-)spherical data
Anova for (hyper-)spherical data, 7

*Topic Analysis of variance
Anova for (hyper-)spherical data, 7
Anova for circular data, 8

*Topic Angle of rotation
Rotation axis and angle of rotation given a rotation matrix, 71
Rotation matrix from a rotation axis and angle of rotation, 72

*Topic Angular central Gaussian distribution
Angular central Gaussian random values simulation, 6

*Topic Anova
Directional-package, 3

*Topic Axis of rotation
Rotation axis and angle of rotation given a rotation matrix, 71
Rotation matrix from a rotation axis and angle of rotation, 72

*Topic BIC of the mixture models
BIC for the model based clustering using mixtures of von Mises-Fisher distributions, 10

*Topic Bimodal distribution on the sphere
MLE of the Wood bimodal distribution on the sphere, 61

*Topic Bingham distribution
Simulation from a Bingham distribution using any symmetric matrix A, 78
Simulation of random values from a Bingham distribution, 80

*Topic Circular correlation type II
Circular correlations between two circular variables, 15

*Topic Circular correlation type I
Circular correlations between two circular variables, 15

*Topic Circular data
A test for testing the equality of the concentration parameters for circular data, 5
Anova for circular data, 8
Test of equality of the concentration parameters for circular data, 96
Uniformity test for circular data, 103

*Topic Circular regression
Circular or angular regression, 12

*Topic Circular-linear correlation
Circular-linear correlation, 13

*Topic Concentration parameters
Test for equality of concentration parameters for spherical data, 95

*Topic Contour plot
Contour plot of a mixture of von Mises-Fisher distributions model, 17
Contour plot of spherical data using a von Mises-Fisher kernel density estimate, 18
Contour plot of the Kent distribution for some data, 19
Contour plot of the Kent distribution without any data, 20
Contour plots of the von Mises-Fisher distribution, 21

*Topic Cross validation
Cross validation in von Mises-Fisher discriminant
analysis, 27
Cross validation with ESAG
discriminant analysis, 28
*Topic Cross-validation
    Tuning of the k-NN regression, 101
*Topic Directional data
    Directional-package, 3
*Topic Directional k-NN algorithm
    k-NN algorithm using the arc
cosinus distance, 44
    Tuning of the k-NN algorithm using
    the arc cosinus distance, 99
*Topic Discriminant analysis
    Cross validation for estimating
    the classification rate, 25
    Cross validation in von
    Mises-Fisher discriminant
    analysis, 27
    Cross validation with ESAG
discriminant analysis, 28
    Directional-package, 3
    Prediction in discriminant
    analysis based on ESAG
distribution, 63
    Prediction in discriminant
    analysis based on von
    Mises-Fisher distribution, 64
*Topic ESAG distribution
    MLE of the ESAG distribution, 53
    Random values generation from the
    ESAG distribution, 67
*Topic Equality of concentrations
    A test for testing the equality of
    the concentration parameters
    for circular data, 5
    Test of equality of the
    concentration parameters for
    circular data, 96
*Topic Euclidean coordinates
    Euclidean transformation, 33
    Inverse of the Euclidean
    transformation, 43
*Topic Euclidean data
    k-NN regression, 45
    Tuning of the k-NN regression, 101
*Topic Fisher-Bingham distribution
    Saddlepoint approximations of the
    Fisher-Bingham distributions,
    76
    Simulation of random values from a
    spherical Fisher-Bingham
distribution, 82
*Topic Generalised von Mises
distribution
    MLE of the generalised von Mises
distribution, 54
*Topic Goodness of fit test
    Hypothesis test for IAG
distribution over the ESAG
distribution, 39
    Hypothesis test for von
    Mises-Fisher distribution
    over Kent distribution, 40
*Topic Graphs
    Directional-package, 3
*Topic Grouped data
    Goodness of fit test for grouped
data, 36
    Summary statistics for grouped
    circular data, 92
*Topic Hypothesis testing
    A test for testing the equality of
    the concentration parameters
    for circular data, 5
    Anova for circular data, 8
    Test for equality of
    concentration parameters for
    spherical data, 95
    Test of equality of the
    concentration parameters for
    circular data, 96
    Uniformity test for circular data, 103
*Topic Hypothesis test
    Rayleigh’s test of uniformity, 69
*Topic Inverse transformation
    Inverse of Lambert’s equal area
    projection, 42
*Topic Kent distribution
    Contour plot of the Kent
distribution for some data, 19
    Contour plot of the Kent
distribution without any data, 20
    Hypothesis test for IAG
distribution over the ESAG
distribution, 39
Hypothesis test for von Mises-Fisher distribution over Kent distribution, 40
Logarithm of the Kent distribution normalizing constant, 48
MLE of the Kent distribution, 55
Simulation of random values from a spherical Fisher-Bingham distribution, 82
Simulation of random values from a spherical Kent distribution, 83

*Topic Kernel density estimate
Tuning of the bandwidth parameter in the von Mises kernel, 97
Tuning of the bandwidth parameter in the von Mises-Fisher kernel, 98
von Mises-Fisher kernel density estimation for (hyper-)spherical data, 107

*Topic Kernel density
von Mises kernel density estimation, 105

*Topic Lambert’s equal area projection
Inverse of Lambert’s equal area projection, 42
Lambert’s equal area projection, 47

*Topic Matrix Fisher distribution
MLE of the Matrix Fisher distribution on SO(3), 57

*Topic Maximum likelihood estimation
MLE of the Matrix Fisher distribution on SO(3), 57
MLE of the spherical projected normal distribution, 59
MLE of the von Mises-Fisher distribution, 60

*Topic Mean direction
Test for a given mean direction, 94

*Topic Median direction
Spherical and hyperspherical median, 86

*Topic Mixtures of von Mises-Fisher distributions
BIC for the model based clustering using mixtures of von Mises-Fisher distributions, 10
Contour plot of a mixture of von Mises-Fisher distributions, 17
Simulation of random values from a mixture of von Mises-Fisher distributions, 81

*Topic Model based clustering
Mixtures of Von Mises-Fisher distributions, 49

*Topic Normalising constant
Logarithm of the Kent distribution normalizing constant, 48
Saddlepoint approximations of the Fisher-Bingham distributions, 76

*Topic One sample hypothesis test
Test for a given mean direction, 94

*Topic Probability density function
Probability density function of the von Mises-Fisher distribution, 65

*Topic Projected normal
Circular or angular regression, 12

*Topic Random values simulation
Random values simulation from some circular distributions, 68
Simulation of random values from a Bingham distribution, 80
Simulation of random values from a mixture of von Mises-Fisher distributions, 81
Simulation of random values from a von Mises-Fisher distribution, 85

*Topic Regression
Directional-package, 3

*Topic Rejection sampling
Simulation of random values from a Bingham distribution, 80

*Topic Rotation matrix
Random sample of matrices in SO(p), 66
Rotation matrix from a rotation axis and angle of rotation, 72
Rotation matrix to rotate a spherical vector along the
direction of another, 75
*Topic SO(p)
  Random sample of matrices in SO(p), 66
*Topic Saddlepoint approximation
  Logarithm of the Kent distribution normalizing constant, 48
  Saddlepoint approximations of the Fisher-Bingham distributions, 76
*Topic Simulated data
  Simulation of random values from a spherical Fisher-Bingham distribution, 82
  Simulation of random values from a spherical Kent distribution, 83
*Topic Simulation of random values
  Simulation from a Bingham distribution using any symmetric matrix A, 78
*Topic Simulation
  Directional-package, 3
*Topic Spherical coordinates
  Euclidean transformation, 33
  Inverse of the Euclidean transformation, 43
*Topic Spherical data
  Directional-package, 3
  k-NN regression, 45
  Lambert’s equal area projection, 47
  Spherical-spherical correlation, 88
  Spherical-spherical regression, 89
  Test for equality of concentration parameters for spherical data, 95
  Tuning of the k-NN regression, 101
*Topic Spherical-Spherical regression
  Spherical-spherical regression, 89
*Topic Squared correlation
  Spherical-spherical correlation, 88
*Topic Summary statistics
  Summary statistics for circular data, 91
  Summary statistics for grouped circular data, 92
*Topic Supervised classification
  k-NN algorithm using the arc cosinus distance, 44
  Tuning of the k-NN algorithm using the arc cosinus distance, 99
*Topic Tuning of the bandwidth
  Tuning of the bandwidth parameter in the von Mises-Fisher kernel, 98
*Topic Tuning the bandwidth
  Tuning of the bandwidth parameter in the von Mises kernel, 97
*Topic Uniform distribution on the (hyper)-sphere
  Rayleigh’s test of uniformity, 69
*Topic Uniformity test
  Uniformity test for circular data, 103
*Topic Von Mises distribution
  Summary statistics for circular data, 91
  Summary statistics for grouped circular data, 92
*Topic Von Mises-Fisher distributions
  Cross validation for estimating the classification rate, 25
  Prediction in discriminant analysis based on ESAG distribution, 63
  Prediction in discriminant analysis based on von Mises-Fisher distribution, 64
*Topic Wood distribution
  MLE of the Wood bimodal distribution on the sphere, 61
*Topic bivariate angular Gaussian
  MLE of some circular distributions, 50
*Topic circular data
  Directional-package, 3
  MLE of some circular distributions, 50
*Topic directional data
  Angular central Gaussian random values simulation, 6
  Conversion of cosines to azimuth and plunge, 22
  MLE of the angular central Gaussian distribution, 52
**Topic** high concentration F test  
Anova for (hyper-)spherical data, 7

**Topic** k-NN regression  
k-NN regression, 45  
Tuning of the k-NN regression, 101

**Topic** maximum likelihood estimation  
MLE of the ESAG distribution, 53  
MLE of the Kent distribution, 55

**Topic** projected normal distribution  
MLE of the spherical projected normal distribution, 59

**Topic** projected normal  
MLE of the angular central Gaussian distribution, 52

**Topic** random values simulation  
Angular central Gaussian random values simulation, 6

**Topic** simulation  
Random values generation from the ESAG distribution, 67

**Topic** spherical data  
MLE of the ESAG distribution, 53  
MLE of the spherical projected normal distribution, 59  
Random values generation from the ESAG distribution, 67

**Topic** unit vectors  
Unit vector(s) with a given angle, 104

**Topic** von Mises distribution  
Tuning of the bandwidth parameter in the von Mises kernel, 97

**Topic** von Mises kernel  
von Mises kernel density estimation, 105

**Topic** von Mises-Fisher distribution  
Contour plots of the von Mises-Fisher distribution, 21  
Cross validation in von Mises-Fisher discriminant analysis, 27  
Cross validation with ESAG discriminant analysis, 28  
Goodness of fit test for grouped data, 36  
Hypothesis test for IAG distribution over the ESAG distribution, 39  
Hypothesis test for von Mises-Fisher distribution over Kent distribution, 40  
Mixtures of Von Mises-Fisher distributions, 49  
MLE of the von Mises-Fisher distribution, 60  
Probability density function of the von Mises-Fisher distribution, 65  
Random values simulation from some circular distributions, 68  
Simulation of random values from a von Mises-Fisher distribution, 85  
Tuning of the bandwidth parameter in the von Mises-Fisher kernel, 98

**Topic** von Mises-Fisher kernel  
Contour plot of spherical data using a von Mises-Fisher kernel density estimate, 18

**Topic** von Mises-Fisher  
von Mises-Fisher kernel density estimation for (hyper-)spherical data, 107

**Topic** wrapped Cauchy distribution  
MLE of some circular distributions, 50

A test for testing the equality of the concentration parameters for circular data, 5  
acg, 7, 54, 60, 68, 70  
acg (MLE of the angular central Gaussian distribution), 52  
Angular central Gaussian random values simulation, 6  
Anova for (hyper-)spherical data, 7  
Anova for circular data, 8  
Arotation, 24, 25, 34, 43, 67, 73, 76  
Arotation (Rotation axis and angle of rotation given a rotation matrix), 71

BIC for the model based clustering using mixtures of von Mises-Fisher distributions, 10
bic.mixvmf, 50, 82
bic.mixvmf (BIC for the model based clustering using mixtures of von Mises-Fisher distributions), 10

Check visually whether matrix Fisher samples is correctly generated or not, 11
circ.cor1, 13, 14, 58, 89, 90
circ.cor1 (Circular correlations between two circular variables), 15
circ.cor2, 13, 14, 16, 89, 90
circ.cor2 (Circular correlations between two circular variables), 15
circ.cor1 (Circular correlations between one and many circular variables), 14
circ.summary, 37, 54, 55, 60, 66, 68, 69, 93
circ.summary (Summary statistics for circular data), 91
circlin.cor, 13, 16
circlin.cor (Circular-linear correlation), 13
circpurka.density (Density of some circular distributions), 31
Circular or angular regression, 12
Circular-linear correlation, 13
Circular correlations between one and many circular variables, 14
Circular correlations between two circular variables, 15
cconc.test, 6, 8, 9, 96
conc.test (Test of equality of the concentration parameters for circular data), 96
Contour plot of a mixture of von Mises-Fisher distributions model, 17
Contour plot of spherical data using a von Mises-Fisher kernel density estimate, 18
Contour plot of the Kent distribution for some data, 19
Contour plot of the Kent distribution without any data, 20
Contour plots of the von Mises-Fisher distribution, 21
Conversion of cosines to azimuth and plunge, 22
Converting a rotation matrix on SO(3) to an unsigned unit quaternion, 23
Converting an unsigned unit quaternion to rotation matrix on SO(3), 24
cosap (Conversion of cosines to azimuth and plunge), 22
Cross validation for estimating the classification rate, 25
Cross validation in von Mises-Fisher discriminant analysis, 27
Cross validation with ESAG discriminant analysis, 28
Density of some (hyper-)spherical distributions, 29
Density of some circular distributions, 31
Density of the spherical Kent and ESAG distributions, 32
dirda.cv, 36
dirda.cv (Cross validation for estimating the classification rate), 25
Directional-package, 3
dirknn, 27–29, 63, 65, 101
dirknn (k-NN algorithm using the arc cosinus distance), 44
dirknn.tune, 45, 102
dirknn.tune (Tuning of the k-NN algorithm using the arc cosinus distance), 99
eembed.aov, 8, 9, 96
eembed.aov (Anova for (hyper-)spherical data), 7
eembed.circaov, 6, 97
eembed.circaov (Anova for circular data), 8
ESAG.da, 27, 63
ESAG.da (Cross validation with ESAG discriminant analysis), 28
ESAGda.pred (Prediction in discriminant analysis based on ESAG distribution), 63
INDEX

ESAGdensity, 4, 32, 54, 68
ESAGdensity (Density of the spherical Kent and ESAG distributions), 32
ESAGmle, 30, 33, 40, 60, 62, 68, 88
ESAGmle (MLE of the ESAG distribution), 53
ESAGsim, 54
ESAGsim (Random values generation from the ESAG distribution), 67
euclid, 23, 41, 43, 48
euclid (Euclidean transformation), 33
euclid.inv, 23, 34
euclid.inv (Inverse of the Euclidean transformation), 43
Euclidean transformation, 33
eul2rot, 23, 35
eul2rot (Rotation matrix on SO(3) from three Euler angles), 74
Euler angles from a rotation matrix on SO(3), 34
f.rbing, 52, 78, 83, 84
f.rbing (Simulation of random values from a Bingham distribution), 80
fb.saddle, 49, 52, 56
fb.saddle (Saddlepoint approximations of the Fisher-Bingham distributions), 76
fishkent, 40
fishkent (Hypothesis test for von Mises-Fisher distribution over Kent distribution), 40
Generate random folds for cross-validation, 35
ggvm (MLE of the generalised von Mises distribution), 54
Goodness of fit test for grouped data, 36
group.gof, 65, 66
group.gof (Goodness of fit test for grouped data), 36
group.vm, 92
group.vm (Summary statistics for grouped circular data), 92
Habeck’s rotation matrix generation, 38
habeck.rot (Habeck’s rotation matrix generation), 38
hcf.aov, 9, 96
hcf.aov (Anova for (hyper-)spherical data), 7
hcf.circaov, 6, 92, 97
hcf.circaov (Anova for circular data), 8
het.aov, 8, 9, 96
het.aov (Anova for (hyper-)spherical data), 7
het.circaov, 6, 97
het.circaov (Anova for circular data), 8
Hypothesis test for IAG distribution over the ESAG distribution, 39
Hypothesis test for von Mises-Fisher distribution over Kent distribution, 40
iag.density (Density of some (hyper-)spherical distributions), 29
iag.mle, 40, 54, 61
iag.mle (MLE of the spherical projected normal distribution), 59
iag.reg (Spherical regression using the projected normal or the von Mises-Fisher distribution), 87
iagesag, 41
iagesag (Hypothesis test for IAG distribution over the ESAG distribution), 39
Interactive 3D plot of spherical data, 41
Inverse of Lambert’s equal area projection, 42
Inverse of the Euclidean transformation, 43
k-NN algorithm using the arc cosinus distance, 44
k-NN regression, 45
kent.contour, 20, 22
kent.contour (Contour plot of the Kent distribution without any data), 20
kent.datacontour, 19, 21
kent.datacontour (Contour plot of the Kent distribution for some data), 19
kent-density, 32
kent-density (Density of the spherical Kent and ESAG distributions), 32
kent.logcon, 77
kent.logcon (Logarithm of the Kent distribution normalizing constant), 48
kent.mle, 20, 21, 30, 33, 40, 41, 49, 54, 56, 61, 62, 71, 77, 87, 94
kent.mle (MLE of the Kent distribution), 55
kmeans, 10, 49
knn.reg, 27, 63, 65, 102
knn.reg (k-NN regression), 45
knnreg.tune, 46
knnreg.tune (Tuning of the k-NN regression), 101
kuiper (Uniformity test for circular data), 103
lambert, 34, 41, 43, 76
lambert (Lambert’s equal area projection), 47
Lambert’s equal area projection, 47
lambert.inv, 48, 76
lambert.inv (Inverse of Lambert’s equal area projection), 42
Logarithm of the Kent distribution normalizing constant, 48
lr.aov, 8, 9, 96
lr.aov (Anova for (hyper-)spherical data), 7
lr.circaov, 6, 97
lr.circaov (Anova for circular data), 8
makefolds, 26
makefolds (Generate random folds for cross-validation), 35
matrixfisher.mle (MLE of the Matrix Fisher distribution on SO(3)), 57
meandir.test, 70
meandir.test (Test for a given mean direction), 94
mediandir (Spherical and hyperspherical median), 86
mediandir_2 (Spherical and hyperspherical median), 86
mix.vmf, 11, 17, 28, 43, 65, 82, 101
mix.vmf (Mixtures of Von Mises-Fisher distributions), 49
Mixtures of Von Mises-Fisher distributions, 49
mixvmf.contour, 11, 50
mixvmf.contour (Contour plot of a mixture of von Mises-Fisher distributions model), 17
MLE of some circular distributions, 50
MLE of the angular central Gaussian distribution, 52
MLE of the ESAG distribution, 53
MLE of the generalised von Mises distribution, 54
MLE of the Kent distribution, 55
Mle of the Matrix Fisher distribution on SO(3), 57
MLE of the Purkayashta distribution, 58
MLE of the spherical projected normal distribution, 59
MLE of the von Mises-Fisher distribution, 60
MLE of the Wood bimodal distribution on the sphere, 61
Prediction in discriminant analysis based on ESAG distribution, 63
Prediction in discriminant analysis based on von Mises-Fisher distribution, 64
Probability density function of the von Mises-Fisher distribution, 65
purka-density (Density of some (hyper-)spherical distributions), 29
purka.mle (MLE of the Purkayashta distribution), 58
pvm, 37
pvm (Probability density function of the von Mises-Fisher distribution), 65
quat2rot, 24
quat2rot (Converting an unsigned unit quaternion to rotation matrix on SO(3)), 24

racg, 69, 85
racg (Angular central Gaussian random values simulation), 6
Random sample of matrices in SO(p), 66
Random values generation from the ESAG distribution, 67
Random values simulation from some circular distributions, 68
rayleigh, 94, 104
rayleigh (Rayleigh's test of uniformity), 69
Rayleigh's test of uniformity, 69
rbingham, 52, 77, 81, 83, 84, 105
rbingham (Simulation from a Bingham distribution using any symmetric matrix A), 78
Read a file as a Filebacked Big Matrix, 70
read.fbm (Read a file as a Filebacked Big Matrix), 70
rfb, 4, 52, 77, 78, 81, 84, 85, 105
rfb (Simulation of random values from a spherical Fisher-Bingham distribution), 82
rkent, 30, 33, 41, 78, 81, 83
rkent (Simulation of random values from a spherical Kent distribution), 83
rmatrixfisher, 57
rmatrixfisher (Simulation from a Matrix Fisher distribution on SO(3)), 79
rvmf, 11, 50, 85
rvmf (Simulation of random values from a mixture of von Mises-Fisher distributions), 81
rot.matrix, 23, 67, 72, 76
rot.matrix (Rotation matrix from a rotation axis and angle of rotation), 72
rot2eul, 74
rot2eul (Euler angles from a rotation matrix on SO(3)), 34
rot2quat, 25
rot2quat (Converting a rotation matrix on SO(3) to an unsigned unit quaternion), 23
rotation, 24, 25, 67, 72, 73
rotation (Rotation matrix to rotate a spherical vector along the direction of another), 75
Rotation axis and angle of rotation given a rotation matrix, 71
Rotation matrix from a rotation axis and angle of rotation, 72
Rotation matrix on SO(3) from three Euler angles, 74
Rotation matrix to rotate a spherical vector along the direction of another, 75
rsop, 72, 73, 76
rsop (Random sample of matrices in SO(p)), 66
rvmf, 7, 22, 51, 61, 62, 69, 78, 81–84, 105
rvmf (Simulation of random values from a von Mises-Fisher distribution), 85
rvonmises, 7, 32, 37, 51, 55, 66, 85, 92, 93, 104
rvonmises (Random values simulation from some circular distributions), 68
rwrapcauchy (Random values simulation from some circular distributions), 68
Saddlepoint approximations of the Fisher-Bingham distributions, 76
Simulation from a Bingham distribution using any symmetric matrix A, 78
Simulation from a Matrix Fisher distribution on SO(3), 79
Simulation of random values from a Bingham distribution, 80
Simulation of random values from a mixture of von Mises-Fisher distributions, 81
Simulation of random values from a spherical Fisher-Bingham distribution, 82
Simulation of random values from a spherical Kent distribution, 83
Simulation of random values from a von Mises-Fisher distribution, 85
spher.cor, 13, 90
spher.cor (Spherical-spherical correlation), 88
spher.reg, 13, 46, 89, 102
spher.reg (Spherical-spherical regression), 89
spherconc.test, 8
spherconc.test (Test for equality of concentration parameters for spherical data), 95
sphereplot, 22, 54, 56, 62, 90, 96
sphereplot (Interactive 3D plot of spherical data), 41
Spherical and hyperspherical median, 86
Spherical regression using the projected normal or the von Mises-Fisher distribution, 87
Spherical-spherical correlation, 88
Spherical-spherical regression, 89
spml.density (Density of some circular distributions), 31
spml.mle, 60, 68
spml.mle (MLE of some circular distributions), 50
spml.reg, 14–16, 46, 88, 90
spml.reg (Circular or angular regression), 12
Summary statistics for circular data, 91
Summary statistics for grouped circular data, 92
tang.conc (A test for testing the equality of the concentration parameters for circular data), 5
Test for a given mean direction, 94
Test for equality of concentration parameters for spherical data, 95
Test of equality of the concentration parameters for circular data, 96
Tuning of the bandwidth parameter in the von Mises kernel, 97
Tuning of the bandwidth parameter in the von Mises-Fisher kernel, 98
Tuning of the k-NN algorithm using the arc cosinus distance, 99
Tuning of the k-NN regression, 101
Uniformity test for circular data, 103
Unit vector(s) with a given angle, 104
vec (Unit vector(s) with a given angle), 104
visual.check (Check visually whether matrix Fisher samples is correctly generated or not), 11
vm.density (Density of some circular distributions), 31
vm.kde, 92, 93, 98, 99, 107
vm.kde (von Mises kernel density estimation), 105
vmf, 22, 28, 41, 51, 56, 62, 70, 71, 85, 87–89, 92, 94, 104, 107
vmf (MLE of the von Mises-Fisher distribution), 60
vmf.contour, 17, 19, 21, 41
vmf.contour (Contour plots of the von Mises-Fisher distribution), 21
vmf.da, 29, 65, 101
vmf.da (Cross validation in von Mises-Fisher discriminant analysis), 27
vmf.density (Density of some (hyper-)spherical distributions), 29
vmf.kde, 19, 61, 98, 99, 106
vmf.kde (von Mises-Fisher kernel density estimation for (hyper-)spherical data), 107
vmf.kerncontour, 17, 20–22, 99
vmf.kerncontour (Contour plot of spherical data using a von Mises-Fisher kernel density estimate), 18
vmf.reg (Spherical regression using the projected normal or the von Mises-Fisher distribution), 87
vmfda.pred, 27–29, 45, 63
vmfda.pred (Prediction in discriminant analysis based on von
INDEX

Mises-Fisher distribution, 64
vmfkde.tune, 19, 98, 106, 107
vmfkde.tune (Tuning of the bandwidth parameter in the von Mises-Fisher kernel), 98
vmkde.tune, 99, 106, 107
vmkde.tune (Tuning of the bandwidth parameter in the von Mises kernel), 97
von Mises kernel density estimation, 105
von Mises-Fisher kernel density estimation for (hyper-)spherical data, 107
watson (Uniformity test for circular data), 103
wood.mle, 56, 61
wood.mle (MLE of the Wood bimodal distribution on the sphere), 61
wrapcauchy (MLE of some circular distributions), 50
wrapcauchy.density (Density of some circular distributions), 31