Appendix D

The MooreRayleigh software package

D.1 Implementation and installation

The Moore-Rayleigh test is a nonparametric test for the spherical symmetry of a sample of vectors under a general alternative. It is introduced and described in Chapter 5. The R package MooreRayleigh has been developed to perform the test in practice and is released under the GNU Public License Version 3.0\(^1\). The Moore-Rayleigh test has been implemented in R code, the related permutation tests of Diks and Tong (1999) have been implemented in C++ code for speed. Installation is straightforward by invoking

```
R CMD INSTALL MooreRayleigh.tar.gz
```

at the command line.

D.2 Reference

The following is a complete description of all functions available in the MooreRayleigh package, version 1.2-0\(^2\).

---

**bisect**

*Numerical bisection to find the root of a function*

**Description**

Iteratively bisects an interval to find the root of a continuous function. The initial points are assumed to bracket the root. Only the first zero found is returned.

**Usage**

```
bisect(x1, x2, f, max.iter = 38, tol = NULL)
```

---

1. URL: [http://www.gnu.org/licenses/lgpl.html](http://www.gnu.org/licenses/lgpl.html)
2. The version number follows the major/minor convention, where the first number indicates significant (major) changes, the second number minor changes (with even numbers indicating stable releases and odd numbers indicating developmental versions), and the last number is used to indicate consecutive bug fixes.
Arguments

- **x1**: One endpoint of the interval.
- **x2**: The other endpoint of the interval.
- **f**: A continuous function with a root between x1 and x2.
- **max.iter**: Maximum number of bisections to try.
- **tol**: Accuracy of the root. Defaults to \((x1+x2)/2.0*Machine\$double.eps\).

Details

An initial bracketing is given if \(f(x_1)*f(x_2) \geq 0.0\). The interval is halved every step after evaluating \(f\) at the midpoint.

Value

The argument for which \(f\) is zero, up to an accuracy of \(tol\).

Examples

```r
f <- function(x) { x }
bisect(-1,1,f)  # returns zero
bisect(2,4,sin) # approximates pi
```

---

diks.test  
*Monte Carlo testing for symmetry of multivariate samples*

Description

Test a multivariate sample for spherical or reflection symmetry.

Usage

```r
diksS.test(x, bw = 0.25, n.mc = 1e3, center = FALSE, return.perms = FALSE)
diksU.test(x, bw = 0.25, n.mc = 1e3, center = FALSE, return.perms = FALSE)
```

Arguments

- **x**: Numerical vector or matrix. If a matrix, each row is expected to contain the coordinates of a sample. A vector will be interpreted as a matrix with a single column.
- **bw**: Bandwidth to use in the distance calculation.
- **n.mc**: Number of Monte Carlo samples to use.
- **center**: Center \(x\) by subtracting the column means before testing.
- **return.perms**: Numerical value to indicate whether the values of the test statistic are to be returned for all Monte Carlo realizations.

The hypothesis tests of Diks and Tong (1999) are permutation tests. The test statistic \( T \) is based on an U-estimator for the squared distance between two multivariate distributions. In terms of a finite sample \( X_i \in \mathbb{R}^k \ (i = 1, 2, \ldots, n) \) it is given by

\[
T(X_1, \ldots, X_n) = \frac{2}{n(n-1)} \sum_{i,j} K(x_i - x_j),
\]

where \( K \) is the kernel function \( K(X - Y) = \exp(-\|X - Y\|^2/(4d^2)) \) that depends on a bandwidth parameter \( d \) (given by the parameter \( \text{bw} \) of the function). Under the null hypothesis the common distribution of the \( X_i \) is invariant under the action of the respective symmetry group. This can be either spherical symmetry (\( \text{diksU.test} \)) or reflection symmetry (\( \text{diksS.test} \)). For each Monte Carlo sample an element of this group is chosen randomly, acts on the sample \( (X_1, \ldots, X_n) \), and the corresponding value of \( T \) is recorded. The significance probability of the test is the fraction of such “permutations” with a value of \( T \) less than or equal to the one for the original sample.

**Value**

A list with class “htest” containing the following components:

- **statistic**: value of the test statistic for the original data \( x \).
- **p.value**: the significance probability for the test.
- **alternative**: a character string describing the alternative hypothesis (‘not symmetric’).
- **method**: a character string describing the type of test.
- **data.name**: a character string giving the names of the data.
- **n.mc**: the number of Monte Carlo samples used.
- **bw**: the bandwidth parameter used.
- **centered**: a logical value describing whether the columns of \( x \) were centered.
- **statistic.mc**: depending on the value of \( \text{return.perms} \) either \( \text{NULL} \) or a vector containing the values of the test statistic for all Monte Carlo samples.

**Examples**

```r
x <- rsphere(100)  # 100 samples from the unit sphere in 3D
diksU.test(x, n.mc = 1e4)  # should accept the null hypothesis
y <- F3(100)  # 100 samples from the Fisher distribution
diksU.test(x, n.mc = 1e4)  # should reject the null hypothesis
x[,1:2] <- 0  # project to a uniform distribution on a line
diksU.test(z, n.mc = 1e4)  # should reject the null hypothesis
diksS.test(z, n.mc = 1e4)  # should accept the null hypothesis
```
Fisher distribution

Description
Generate random variates from the Fisher distribution, also known as the three-dimensional von-Mises distribution.

Usage
F3(n = 1, lambda = 1)

Arguments
- n: Number of samples requested.
- lambda: Concentration parameter. Must be nonnegative.

Details
The Fisher distribution is a singular distribution on the sphere $S^2 \subset \mathbb{R}^3$. Its density $f(x)$ is proportional to $\exp(\lambda \langle x, \xi \rangle)$, where $\xi \in \mathbb{R}^3$ is the mean direction, and $\lambda \geq 0$ is a concentration parameter. In this implementation $\xi = (0, 0, 1)^t$ is fixed as the unit z-vector, and random variates are generated according to the method of Ulrich (1984) and Wood (1994).

Value
A vector if just one numerical sample is requested, otherwise a matrix with one column for each sample.

Examples
```r
m <- F3(500, lambda = 5)  # 500 x 3 matrix
library(lattice)  # load package lattice for 3D plotting
cloud(z ~ x*y, data=data.frame(x=m[,1],y=m[,2],z=m[,3]))  # point cloud
```

Properties of random walk with linearly increasing steps

Description
Calculate the distribution of a symmetric, unbiased one-dimensional random walk with linearly increasing steps.

Usage
- lrw(N, both.sides = FALSE, nonzero = TRUE)
- dlrw(x, N, both.sides = FALSE, scaled = FALSE)
- plrw(q, N, both.sides = FALSE, scaled = FALSE)
- qlrw(p, N, both.sides = FALSE, scaled = FALSE)
- rlrw(n, N, both.sides = FALSE, scaled = FALSE)
Arguments

\( x, q \)  
Vector of quantiles

\( p \)  
Vector of probabilities

\( n \)  
Number of observations. If length(n) > 1, the length is taken to be the required number.

\( N \)  
Number of steps taken by the walk.

\( \text{both.sides} \)  
Logical value indicating whether the distribution is given for both sides of the coordinate system.

\( \text{nonzero} \)  
Logical value indicating whether only sites with nonzero probability should be returned.

\( \text{scaled} \)  
Logical value indicating whether the argument should be scaled by \( N^{3/2} \).

Details

A random walk with \( N \) steps is considered. At the \( n \)-th step, the position of the walker either increases or decreases by \( n \), with equal probability. The probability distribution of this walk is obtained by iterated convolution. Since it is symmetric, only the positive sites (with nonnegative probability) are returned by default.

Value

Function \texttt{lrw} returns various properties of the distribution of this walk:

\( \text{pr} \)  
A vector with the probabilities for the walker to be at a certain site after \( N \) steps.

\( \text{counts} \)  
A vector with the counts of how many possibilities there are for the walker to reach a certain site after \( N \) steps.

\( \text{signs} \)  
A vector with the average sign for each site. For each possible distinct walk to finish at a specific site, the sign is the product of the signs of the individual steps taken by the walker.

\( \text{pr.signs} \)  
Equal to \( \text{pr} \times \text{signs} \).

\( \text{dst} \)  
A vector with the sites.

\texttt{dlrw} gives the density, \texttt{plrw} gives the distribution function, \texttt{qlrw} gives the quantile function, and \texttt{rlwr} generates random variates.

Examples

\texttt{lrw(N=3,both.sides=TRUE,nonzero=FALSE)}

\( x \leftarrow \text{seq}(0,100,0.1) \)

\texttt{plot(x,dlrw(x,N=10),cex=0.5,pch=20)}  
# probability funct. after 10 steps

\texttt{plot(x,plrw(x,N=10),cex=0.5,pch=20)}  
# distribution funct. after 10 steps

\texttt{sum(dlrw(seq(1,55),N=10))}  
# sums to one
Asymptotic Moore-Rayleigh distribution

Description

Density, distribution function, quantile function and random generation for the asymptotic form of the Moore-Rayleigh distribution, i.e., for the length of the resultant of a random flight with \( N \) linearly growing steps, scaled by \( N^{3/2} \), in the limit of \( N \to \infty \).

Usage

\[
\begin{align*}
dmr(x, k = 3) \\
pmr(q, k = 3) \\
qmr(p, k = 3) \\
rmr(n, k = 3)
\end{align*}
\]

Arguments

- \( x, q \) A vector of quantiles.
- \( p \) A vector of probabilities.
- \( n \) The number of variates requested. If \( \text{length}(n) > 1 \) use the length as the required number.
- \( k \) The dimensionality (\( \geq 2 \)).

Value

\( \text{dmr} \) gives the density, \( \text{pmr} \) gives the distribution function, \( \text{qmr} \) gives the quantile function, and \( \text{rmr} \) generates random deviates.

Note

The asymptotic Moore-Rayleigh distribution with distribution function \( F(x) \) in \( k \) dimensions is related to the \( \chi^2 \) distribution with \( k \) degrees of freedom and distribution function \( G(x) \) by \( F(x) = G(3kx^2) \).

Examples

\[
\begin{align*}
x & \leftarrow \text{seq}(0.02, 2.0, 0.001) \\
p & \leftarrow \text{dmr}(x) \\
\text{plot}(x, p, \text{cex}=0.5, \text{pch}=20, \text{title}="Density function in 3D") \\
d & \leftarrow \text{pmr}(x) \\
\text{plot}(x, d, \text{cex}=0.5, \text{pch}=20, \text{title}="Distribution function in 3D")
\end{align*}
\]
Exact Moore-Rayleigh distribution in three dimensions

Description
Density, distribution function, quantile function and random generation for the Moore-Rayleigh distribution in three dimensions, i.e., for the length of the resultant of a random flight with \( N \) linearly growing steps, scaled by \( N^{3/2} \).

Usage

```r
dmr3(x, N)
pmr3(q, N, method = "Borwein")
qmr3(p, N)
rmr3(n, N)
```

Arguments

- `x,q` A vector of quantiles.
- `p` A vector of probabilities.
- `n` Number of observations. If `length(n) > 1`, the length is taken to be the required number.
- `N` Number of steps taken by the walk.
- `method` This can be one of the following:
  - `Borwein` Combinatorial Borwein summation (default).
  - `Integrate` Use `integrate` to evaluate the oscillating integrals directly.

Details

Value

- `dmr3` gives the density, `pmr3` gives the distribution function, `qmr3` gives the quantile function, and `rmr3` generates random deviates.

Examples

```r
x <- seq(0.02,2.0,0.001)
par(mfrow=c(1,2),mar=c(4,4,1,1))  # plot density and distribution
plot(x, dmr(x), type="l", lwd=1/2, lty=2, ylab=expression(pr(R^n==r)), xlab="r")
lines(x, dmr3(x, N=10), lwd=1/2)
plot(x, pmr(x), type="l", lwd=1/2, lty=2, ylab=expression(pr(R^n<=r)), xlab="r")
lines(x, pmr3(x, N=10), lwd=1/2)
```
Description

Test a sample of three-dimensional vectors for spherical symmetry.

Usage

```r
mr3.test(x, exact = NULL, center = FALSE)
```

Arguments

- `x`: A matrix whose rows represent vectors in three dimensions.
- `exact`: A logical value that indicates whether to use the exact or the asymptotic distribution.
- `center`: A logical value that indicates whether to center the vectors before applying the test.

Details

The Moore-Rayleigh test is a hypothesis test for spherical uniformity under a general alternative. It ranks the \( N \) vectors in \( x \) by their lengths. Under the null hypothesis the vectors are assumed to be distributed uniformly on each hypersphere, and the ranks are randomly realized. The test statistic is the length of the resultant of the vectors in \( x \), normalized by their ranks, and corresponds to the distance covered by a uniform random flight with \( N \) linearly increasing steps under the null hypothesis. It is scaled by \( N^{3/2} \) for asymptotic simplicity. The distribution of the null hypothesis is available in closed form, and evaluated by a combinatorial sum for an exact test (valid for \( N \lesssim 60 \) under IEEE 754 arithmetic) or approximated by the asymptotic distribution (see `mr`). For a two-sample test, the vectors need to be paired (see `pairing`).

Value

A list with class “htest” containing the following components:

- `statistic`: value of the test statistic for the data in \( x \).
- `p.value`: the significance probability for the test.
- `alternative`: a character string describing the alternative hypothesis (‘not symmetric’).
- `method`: a character string describing the type of test.
- `data.name`: a character string giving the names of the data.
- `centered`: a logical value describing whether the columns of \( x \) were centered.
Examples

```r
x <- rsphere(10)           # 10 samples from the unit sphere
mr3.test(x, exact=TRUE)    # one-sample test: should reject alternative

y <- rsphere(10)
xy <- pairing.random(x,y)  # two-sample test: should reject alternative

y <- matrix(runif(30), ncol=3)
xy <- pairing.random(x,y)
mr3.test(xy, exact=TRUE)   # two-sample test: should accept alternative
```

Description

Pair two set of vectors for a two-sample test.

Usage

```r
pairing.transport(x, y, minimize = TRUE, normalize = FALSE)
pairing.ranks(x, y, inverse = FALSE)
pairing.random(x, y)
```

Arguments

- **x, y**: Two matrices where each row represents the coordinates of a vector. The number of columns needs to be the same, the number of rows can differ.
- **minimize**: Logical value that indicates whether the total cost is minimized or maximized.
- **normalize**: Logical value that indicates whether to project all vectors to the unit sphere before matching them.
- **inverse**: Logical value that indicates whether to begin matching vectors with the smallest (default) or the largest vectors (if inverse is TRUE).

Details

The preferred pairing is `pairing.random` in which vectors are randomly matched (by sampling without replacement). Function `pairing.ranks` pairs vectors according to their lengths. The value of `inverse` determines whether this proceeds from the smallest to the largest or the other way around. Function `pairing.transport` pairs vectors by optimal transport under Euclidean distance. The value of `minimize` determines whether the total cost is minimized (default) or maximized.
Value

A list with class “pairing” containing the following components:

- **x, y** original data.
- **xy** a matrix representing difference vectors for all pairs.
- **x.pairing** a vector of indices to index the vectors in **x** for the pairing.
- **y.pairing** a vector of indices to index the vectors in **y** for the pairing.
- **cost** value of the total cost (only for **pairing.transport**).
- **dist** the distance matrix (only for **pairing.transport**).

Examples

```r
x <- rsphere(10)
y <- rsphere(10)
d1 <- numeric(1e3)
d2 <- d1
for (i in seq(along=d)) {
  xy <- pairing.random(x,y)$xy
  d1[i] <- sum(apply(xy^2,1,sum)) # squared lengths of difference
  xy <- pairing.ranks(x,y)$xy
  d2[i] <- sum(apply(xy^2,1,sum))
}
plot(density(d2),main="Sum of squared lengths of difference vectors",
     lty=2)
lines(density(d1),lty=1)
```

rsphere

Random variates on the unit sphere

Description

Generate random vectors on the unit sphere in k-dimensional Euclidean space.

Usage

`rsphere(n, k = 3)`

Arguments

- **n** Number of random variates generated. If `length(n) > 1` the length is taken to be the required number.
- **k** Dimension of the space (default = 3).

Details

Uses the method of Knuth, i.e., the fact that a k-dimensional vector of normal variates is uniformly distributed on the unit sphere $S^{k−1}$ after radial projection.
Value

A matrix, where each row represents the coordinates of one random vector.

Examples

```r
rsphere(2)
pplot(rsphere(1000,k=2),cex=0.5,pch=20,xlab="",ylab="",main="Uniform distribution on the circle")
```