R documentation

of rotate

July 29, 2012

R topics documented:

rotate-package 1
angle_axis 1
arith.mean 2
arsample 2
arsample.unif 3
dcayley 4
dfisher 4
dhaar 5
dvmises 5
EAtoSO3 6
eskew 7
eyeBall 7
genR 8
GuessLs 8
HartleyL1 9
is.SOn 10
MantonL2 10
matrixExp 11
matrixLog 12
projMatrix 12
QtoSO3 13
rar 13
rcayley 14
rfisher 15
riedist 15
rmedian 16
rvmises 17
SumDist 17
tLogMat 18
trim 19
vecNorm 19

`rotate-package` *A package to create and analyze data in the n dimensional special orthogonal group SO(n)*

Description

A package to create and analyze data in the n dimensional special orthogonal group SO(n)

Author(s)

Bryan Stanfill `<stanfill@iastate.edu>`

`angle_axis` *A function Dr. Hofmann wrote*

Description

A function Dr. Hofmann wrote

Usage

```
angle_axis(U, theta)
```

Arguments

`U`	a vector
`theta`	an angle

Value

Used in `\lin{eyeBall}` to orient the data properly

`arith.mean` *Projected Arithmetic Mean $\hat{S}$_P*

Description

This function takes a sample of 3×3 rotations (in the form of a $n \times 9$ matrix where n is the sample size) and returns the projected arithmetic mean denoted $\widehat{\boldsymbol{S}}_P$. For a sample of n random rotations $\boldsymbol{R}_i \in SO(3)$, $i = 1, 2, \dots, n$, this mean-type estimator is defined as

$$\widehat{\boldsymbol{S}}_P =_{\boldsymbol{S} \in SO(3)} \sum_{i=1}^{n} d_E^2(\boldsymbol{R}_i, \boldsymbol{S}) =_{\boldsymbol{S} \in SO(3)} (\boldsymbol{S}^\top \bar{\boldsymbol{R}})$$

where $\bar{\boldsymbol{R}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{R}_i$. First the mean of each element is calculated then that matrix is projected to SO(3) in accordance with the procedure presented in Moahker's 2003 paper

Usage

```
arith.mean(Rs)
```

Arguments

Rs	A sample of n 3×3 random rotations

Value

S3 `arith.mean` object; A 3×3 matrix in SO(3) called the Projected arithmetic mean

See Also

`MantonL2`, `HartleyL1`, `rmedian`

Examples

```
r<-rvmises(20,0.01)
Rs<-genR(r)
arith.mean(Rs)
```

---

arsample — *Accept/reject algorithm written by Dr. Hofmann*

Description

Accept/reject algorithm written by Dr. Hofmann

Usage

```
arsample(f, g, M, kappa, ...)
```

Arguments

f	target density
g	sampling density
M	maximum in uniform density
kappa	second parameter in the target density
...	additional arguments passed to samping density, g

Value

a random observation from target density

Author(s)

Heike Hofmann

`arsample.unif` *Accept reject ratio of uniforms? Written by Dr. Hofmann*

Description

Accept reject ratio of uniforms? Written by Dr. Hofmann

Usage

```
arsample.unif(f, M, ...)
```

Arguments

`f`	target density
`M`	maximum value for one of the uniforms
`...`	additional arguments sent to f

Value

x an observation from the target density

Author(s)

Heike Hofmann

`dcayley` *Cayley distribution for angular data*

Description

The symmetric Cayley distribution has a density of the form

$$C_{\mathrm{C}}(r|\kappa) = \frac{1}{\sqrt{\pi}}\frac{\Gamma(\kappa+2)}{\Gamma(\kappa+1/2)}2^{-(\kappa+1)}(1+\cos r)^{\kappa}(1-\cos r)$$

. It was orignally given in the material sciences literature by Schaben 1997 and called the de la Vall\'ee Poussin distribution but was more recently discussed and introduced in a more general manner by Leon 06.

Usage

```
dcayley(r, kappa = 1, Haar = F)
```

Arguments

`r`	Where the density is being evaluated
`kappa`	The concentration paramter, taken to be zero
`Haar`	logical, if density is evaluated with respect to Haar measure or Lebesgue

Value

value of Cayley distribution with concentration κ evaluated at r

See Also

rcayley,dfisher,dhaar

`dfisher` *von Mises-Fisher distribution for angular data*

Description

The symmetric matrix fisher distribution has the density

$$C_{\mathrm{F}}(r|\kappa)=\frac{1}{2\pi[\mathrm{I}_0(2\kappa)-\mathrm{I}_1(2\kappa)]}e^{2\kappa\cos(r)}[1-\cos(r)]$$

where $\mathrm{I}_{\mathrm{p}}(\cdot)$ denotes the Bessel function of order p defined as $\mathrm{I}_{\mathrm{p}}(\kappa)=\frac{1}{2\pi}\int_{-\pi}^{\pi}\cos(pr)e^{\kappa\cos r}dr$. This function allows the user to evaluate the function $C_{\mathrm{F}}(r|\kappa)$ at r with κ provided by the user.

Usage

```
dfisher(r, kappa = 1, Haar = F)
```

Arguments

r	Where the density is being evaluated
kappa	The concentration paramter, taken to be zero
Haar	logical, if density is evaluated with respect to Haar measure or Lebesgue

Value

value of Fisher matrix distribution with concentration κ evaluated at r

---

`dhaar` *Evaluate the uniform distribution on the circle at r*

Description

The uniform distribution on the sphere is also know as the Haar measure and has the density function

$$C_U(r)=\frac{1-cos(r)}{2\pi}$$

Usage

```
dhaar(r)
```

Arguments

r	Where the density is being evaluated

Value

the probability density evaluated at r

`dvmises` *Density function for circular von Mises distribution*

Description

The circular von Mises-based distribution has the density

$$C_\mathrm{M}(r|\kappa) = \frac{1}{2\pi \mathrm{I}_0(\kappa)} e^{\kappa \cos(r)}$$

. This function allows the use to evaluate $C_\mathrm{M}(r|\kappa)$ at angle r given a concentration parameter κ.

Usage

```
dvmises(r, kappa = 1, Haar = F)
```

Arguments

`r`	value at which to evaluate the distribution function
`kappa`	concentration paramter
`Haar`	logical, if density is evaluated with respect to Haar measure or Lebesgue

Value

value of circular-von Mises distribution with concentration κ evaluated at r

See Also

`rvmises`, `dfisher`,`dhaar`,`dcayley`

`EAtoSO3` *A function that will take in a Euler angle and return a rotation matrix in vector format*

Description

A function that will take in a Euler angle and return a rotation matrix in vector format

Usage

```
EAtoSO3(eur)
```

Arguments

`eur`	numeric Euler angle representation of an element in SO(3)

Value

numeric 9×1 vector of a matrix in SO(3)

See Also

`is.SOn` can be used to check the output of this function

Examples

```
eaExample<-c(pi/2,3*pi/4,0)
SO3Dat<-EAtoSO3(eaExample)
is.SOn(SO3Dat)
```

---

eskew *A function Dr. Hofmann wrote*

Description

A function Dr. Hofmann wrote

Usage

```
eskew(U)
```

Arguments

U	a vector of size 3 it looks like

Value

I have no idea

Author(s)

Heike Hofmann

---

eyeBall *A novel approach to visualizing random rotations.*

Description

This function produces a three-dimensional globe onto which the on column of the provided sample is drawn. The data are centered around a provided matrix and the user can choose to display this center or not.

Usage

```
eyeBall(Rs, center = diag(1, 3, 3), column = 1,
  show.estimates = FALSE, ...)
```

Arguments

`Rs`	the sample of n random rotations
`center`	point about which to center the observations
`column`	integer 1 to 3 indicating which column to display
`show.estimates`	rather to display the four estimates of the principal direction or not
`...`	Additional arguments passed to ggplot2

Value

a ggplot2 object with the data dispalyed on a blank sphere

Examples

```
r<-rvmises(20,1.0)
Rs<-genR(r)
eyeBall(Rs,center=arith.mean(Rs),show.estimates=TRUE,shape=4)
```

---

genR *Generate rotation matrix given misorientation angle, r*

Description

A function that generates a random rotation in $SO(3)$ following a Uniform-Axis random roation distribution with central direction S The exact form of the UARS distribution depends upon the distribution of the roation r

Usage

```
genR(r, S = diag(1, 3, 3))
```

Arguments

`r`	The angle through which all three dimensions are rotated after the axis was picked uniformly on the unit sphere
`S`	the principle direction

Value

a $n \times 9$ matrix in SO(3) with misorientation angle r and principal direction S

Examples

```
r<-rvmises(20,0.01)
genR(r)
```

GuessLs *d_R^p based estimators of the Central Direction*

Description

This functions is slow but it computes the element of SO(3) that minimizes the sum of the pth order Riemannian distances. It returns the the random roatation Shat. It calls the function SumDistR which calculates the sum of the pth order Riemannian distances between the sample Rs and S. This needs to be trashed, most likely.

Usage

```
GuessLs(Rs, maxe = 0.001, p)
```

Arguments

Rs	The sample of random rotations in the form of an nx9 matrix
maxe	The stopping rule of the optimization process
p	The order of the riemannian distance to minimize

Value

S3

Shatp The estimated random rotation $\widehat{\boldsymbol{S}}_P$

HartleyL1 *Compute the geometric median of a sample of random rotations*

Description

This function uses the algorithm published by Hartley to estimate the principle direction of a sample of random rotaions with the point in $SO(3)$ that minimizes the sum of first order Riemannian distances, aka the geometric median and denoted $\widetilde{\boldsymbol{S}}_G$. More explicitly

$$\widetilde{\boldsymbol{S}}_G = \widetilde{\boldsymbol{S}}_G =_{\boldsymbol{S}\in SO(3)} \sum_{i=1}^{n} d_G(\boldsymbol{R}_i, \boldsymbol{S})$$

.

Usage

```
HartleyL1(Rs, epsilon = 1e-05, maxIter = 1000)
```

Arguments

Rs	the sample $n \times 9$ matrix with rows corresponding to observations
epsilon	the stopping rule for the iterative algorithm
maxIter	integer, the maximum number of iterations allowed

Value

list

S	the element in SO(3) minimizing the sum of first order Riemannian distances for sample Rs
iter	the number of iterations needed to converge or not

See Also

MantonL2, arith.mean, rmedian

Examples

```
r<-rvmises(20,0.1)
Rs<-genR(r)
HartleyL1(Rs)
```

---

is.SOn *A function to determine if a given matrix is in SO(n) or not.*

Description

A function to determine if a given matrix is in SO(n) or not.

Usage

```
is.SOn(x)
```

Arguments

x	numeric $n \times n$ matrix or vector of length n^2

Value

logical T if the matrix is in SO(n) and false otherwise

Examples

```
is.SOn(diag(1,3,3))
is.SOn(1:9)
```

MantonL2 — *This is the same as MantonL2 except it starts at the S_p rather than an arbitrary sample element*

Description

The intrisic approach to the arithmetic mean is given by the estimatot

$$\widehat{\boldsymbol{S}}_G =_{\boldsymbol{S}\in SO(3)} \sum_{i=1}^{n} d_G^2(\boldsymbol{R}_i, \boldsymbol{S})$$

. That is, the matrix $\widehat{\boldsymbol{S}}_G$ minimizes the sum of squared distances in the intrensic sense, or Riemannian distances.

Usage

```
MantonL2(Rs, epsilon = 1e-05, maxIter = 2000,
  startSp = T, si = 1)
```

Arguments

Rs	the sample $n \times 9$ matrix with rows corresponding to observations
epsilon	the stopping rule for the iterative algorithm
maxIter	integer, the maximum number of iterations allowed
startSp	logical 'TRUE' if first guess is $\widehat{\boldsymbol{S}}_P$, a random sample observation otherwise
si	which sample point to start at

Value

a list

S	the element in SO(3) minimizing the sum of squared Riemannian distances for sample Rs
iter	the number of iterations needed to converge or not

See Also

arith.mean, HartleyL1, rmedian

Examples

```
r<-rcayley(20,1)
Rs<-genR(r)
MantonL2(Rs)
```

matrixExp *This fuction will compute the natural exponential of skew-symmetric matrix. It uses the special case of the Taylor expansion for SO(n) matrices.*

Description

This fuction will compute the natural exponential of skew-symmetric matrix. It uses the special case of the Taylor expansion for SO(n) matrices.

Usage

```
matrixExp(A)
```

Arguments

A	3-dimensional skew-symmetric matrix, i.e., $\boldsymbol{A} = -\boldsymbol{A}^\top$

Value

numeric matrix $e^{\boldsymbol{A}}$

matrixLog *This fuction will compute the natural logarithm of a matrix in SO(n). It uses the special case of the Taylor expansion for SO(n) matrices.*

Description

This fuction will compute the natural logarithm of a matrix in SO(n). It uses the special case of the Taylor expansion for SO(n) matrices.

Usage

```
matrixLog(R)
```

Arguments

R	numeric matrix in $SO(n)$

Value

mlog numeric matrix $\log(R)$

projMatrix — *The projection of an arbitrary 3×3 matrix into $SO(3)$*

Description

This function uses the process given in Moakher 2002 to project an arbitrary 3×3 matrix into $SO(3)$.

Usage

```
projMatrix(M)
```

Arguments

M	3×3 matrix to project

Value

projection of M into $SO(3)$

See Also

arith.mean, rmedian

Examples

```
M<-matrix(rnorm(9),3,3)
projMatrix(M)
```

QtoSO3 — *A function to translate from unit quaternion representation to SO(3) representation of a rotation matrix*

Description

A function to translate from unit quaternion representation to SO(3) representation of a rotation matrix

Usage

```
QtoSO3(q)
```

Arguments

q	numeric unit vector, i.e. $q^\top q = 1$, representing an element in SO(3)

Value

vector representation of a rotation matrix in SO(3)

See Also

is.SOn can be used to check the return vector

Examples

```
is.SOn(QtoSO3(c(1/sqrt(2),0,0,1/sqrt(2))))
```

---

rar *Call arsample 'n' times to get a sample of size n from target density f*

Description

Call arsample 'n' times to get a sample of size n from target density f

Usage

```
rar(n, f, g, M, ...)
```

Arguments

n	number of sample wanted
f	target density
g	sampling distribution
M	maximum number in uniform proposal density
...	additional arguments sent to arsample

Value

a vector of size n of observations from target density

Author(s)

Heike Hofmann

---

rcayley *Simulate misorientation angles from Cayley distribtuion*

Description

This function allows the user to simulate n misorientation angles from the Cayley distribution symmetric about 0 on interval $(-\pi, \pi]$. The relationship between Cayley and Beta distribution is used. The symmetric Cayley distribution has a density of the form

$$C_{\mathrm{C}}(r|\kappa) = \frac{1}{\sqrt{\pi}} \frac{\Gamma(\kappa+2)}{\Gamma(\kappa+1/2)} 2^{-(\kappa+1)} (1+\cos r)^{\kappa} (1-\cos r)$$

. It was orignally given in the material sciences literature by Schaben 1997 and called the de la Vall\'ee Poussin distribution but was more recently discussed and introduced in a more general manner by Leon 06.

Usage

```
rcayley(n, kappa = 1)
```

Arguments

n	sample size
kappa	The concentration paramter

Value

vector of n observations from Cayley(kappa) distribution

Examples

```
r<-rcayley(20,0.01)
```

---

rfisher — *Simulate a data set of size n from the matrix Fisher angular distribution*

Description

The symmetric matrix fisher distribution has the density

$$C_{\mathrm{F}}(r|\kappa) = \frac{1}{2\pi[\mathrm{I}_0(2\kappa) - \mathrm{I}_1(2\kappa)]} e^{2\kappa\cos(r)}[1 - \cos(r)]$$

where $\mathrm{I}_{\mathrm{p}}(\cdot)$ denotes the Bessel function of order p defined as $\mathrm{I}_{\mathrm{p}}(\kappa) = \frac{1}{2\pi}\int_{-\pi}^{\pi}\cos(pr)e^{\kappa\cos r}dr$. This function allows for simulation of n random deviates with density $C_{\mathrm{F}}(r|\kappa)$ and κ provided by the user.

Usage

```
rfisher(n, kappa = 1)
```

Arguments

n	sample size
kappa	the concentration parameter

Value

a sample of size n from the matrix Fisher distribution with concentration κ

See Also

dfisher,rvmises,rcayley

`riedist` *Riemannian Distance Between Two Random Rotations*

Description

This function will calculate the riemannian distance between an estimate of the central direction (in matrix or vector form) and the central direction. By default the central direction is taken to be the identity matrix, but any matrix in SO(3) will work. It calls the matrix log and matrix exponential functions also given here.

Usage

```
riedist(R, S = diag(1, 3, 3))
```

Arguments

R	The estimate of the central direction
S	The true central direction

Value

S3 `riedist` object; a number between 0 and pi that is the shortest geodesic curve connecting two matrices, i.e., the Riemannian distance

Examples

```
r<-rvmises(20,0.01)
Rs<-genR(r)
Sp<-arith.mean(Rs)
riedist(Sp,diag(1,3,3))
```

`rmedian` *Compute the minimizer of the first order Euclidean distances.*

Description

The embeded median type estimator we call the projected median and is given by

$$\widetilde{\boldsymbol{S}}_P =_{\boldsymbol{S}\in SO(3)} \sum_{i=1}^{n} d_E(\boldsymbol{R}_i, \boldsymbol{S})$$

. The algorithm used is a modified Weiszfeld algorithm and is similar to the algorithm proposed by Hartley to compute the geometric median $\widetilde{\boldsymbol{S}}_G$.

Usage

```
rmedian(Rs, epsilon = 1e-05, maxIter = 2000)
```

Arguments

`Rs`	the sample $n \times 9$ matrix with rows corresponding to observations
`epsilon`	the stopping rule for the iterative algorithm
`maxIter`	integer, the maximum number of iterations allowed

Value

a list

`S`	the element in SO(3) minimizing the sum of first order Euclidean distances for sample Rs
`iter`	the number of iterations needed to converge or not

See Also

`MantonL2`, `HartleyL1`, `arith.mean`

Examples

```
r<-rcayley(50,1)
 Rs<-genR(r)
 rmedian(Rs)
```

`rvmises` *Generate a vector of angles(r) from the von Mises Circular distribution*

Description

The circular von Mises-based distribution has the density

$$C_{\mathrm{M}}(r|\kappa) = \frac{1}{2\pi \mathrm{I}_0(\kappa)} e^{\kappa \cos(r)}$$

. This function allows the use to simulate n random deviates from $C_{\mathrm{M}}(r|\kappa)$ given a concentration parameter κ.

Usage

```
rvmises(n, kappa = 1)
```

Arguments

`kappa`	The concentration parameter of the distribution
`n`	The number of angles desired

Value

S3 `rvmises` object; a vector of n angles following the von Mises Circular distribution with concentration kappa and mean/mode 0

Examples

```
r<-rvmises(20,0.01)
```

SumDist *Compute the sum of the $p\hat{}th$ order distances between Rs and S*

Description

Compute the sum of the p^{th} order distances between Rs and S

Usage

```
SumDist(Rs, S = diag(1, 3, 3), p)
```

Arguments

Rs	numeric matrix with sample size n rows and m columns
S	the matrix to compute the sum of distances between each row of Rs with
p	the order of the distances to compute

Value

list of size two

Rieman	the sum of p^{th} order Riemannian distances
Euclid	the sum of p^{th} order Euclidean distances

Examples

```
r<-rvmises(20,0.01)
Rs<-genR(r)
Sp<-arith.mean(Rs)
SumDist(Rs,S=Sp,2)
```

tLogMat *Log of a matrix times some center S*

Description

Used to speed up the Riemannian based estimators

Usage

```
tLogMat(x, S)
```

Arguments

x	vector of length 9
S	3×3 matrix

Value

skew-symmetic matrix $\log(\boldsymbol{S}^\top \boldsymbol{x})$

See Also

MantonL2, HartleyL1

Examples

```
rs<-rvmises(20,1)
Rs<-genR(rs)
apply(Rs,1,tLogMat,S=diag(1,3,3))
```

`trim` *Function to Trim the Sample*

Description

This function will take a sample of size n random rotations, find the observations beyond the alpha/2 percentile and delete them from the sample. To determine which observations to remove, the average distance of each sample point from the estimated central direction is used.

Usage

```
trim(Rs, alpha)
```

Arguments

`Rs`	The sample of random rotations
`alpha`	The percent of observations to be trimeed

Value

S3 `trim` object; a sample of size n-(n*alpha) of random roatations

---

`vecNorm` *Turn a vector into a matrix and compute the the norm between x and S*

Description

Turn a vector into a matrix and compute the the norm between x and S

Usage

```
vecNorm(x, S, ...)
```

Arguments

`x`	numeric vector
`S`	numeric vector or matrix
`...`	additional arguments passed to `norm` function

Value

the norm of x-S