

# Package ‘orientlib’

October 26, 2009

**Title** Support for orientation data

**Version** 0.10.2

**Author** Duncan Murdoch

**Description** Representations, conversions and display of orientation SO(3) data. See the orientlib help topic for details.

**Maintainer** Duncan Murdoch <murdoch@stats.uwo.ca>

**License** GPL

**Depends** R (>= 2.10.0), methods

**Suggests** rgl, scatterplot3d

**LazyLoad** yes

**SystemRequirements** Will use djmrgl or rgl packages for rendering if present

**URL** <http://www.stats.uwo.ca/faculty/murdoch/software>

**Repository** CRAN

**Date/Publication** 2009-10-26 21:43:49

## R topics documented:

boat3d . . . . .	2
coerce-methods . . . . .	3
eulerzxz . . . . .	4
eulerzyx . . . . .	5
index-methods . . . . .	6
length-methods . . . . .	6
matrix-classes . . . . .	6
matrix-methods . . . . .	7
mean-methods . . . . .	7
nearest . . . . .	8

orientation-class . . . . .	9
orientlib . . . . .	10
orientlm . . . . .	11
quaternion . . . . .	12
rotation.distance . . . . .	13
rotmatrix . . . . .	14
rotvector . . . . .	15
skewmatrix . . . . .	16
skewvector . . . . .	17
vector-classes . . . . .	18
weighted.mean-methods . . . . .	19
<b>Index</b>	<b>20</b>

---

boat3d	<i>Draw boat glyphs for orientation data</i>
--------	--

---

## Description

Draws a stylized sailboat to represent an orientation.

## Usage

```
boat3d(orientation, x = 1:length(orientation), y = 0,
       z = 0, scale = 0.25, col = 'red', add = FALSE, box = FALSE, axes = TRUE,
       graphics = c('rgl', 'scatterplot3d'), ...)
```

## Arguments

orientation	An <a href="#">orientation</a> object to be shown.
x, y, z	Coordinates where boats should be shown.
scale	Size of boats
col	Colour of boats
add	Context in which to continue drawing, or FALSE to clear first.
box	Whether to draw a box around the plot
axes	Whether to draw axes
graphics	Which graphics package to use
...	Additional graphics parameters; see Details below

## Details

For the identity orientation, the sailboats will be shown upright. Other orientations are shown as rotations of this glyph.

The (x,y,z) coordinate appears in the middle of the sail, at the top of the gunwales of the boat.

If the `rgl` package is installed, it will be used to draw solid faces on the boats which can be moved by the user. If not, but the `scatterplot3d` package is installed, it will be used to draw fixed wireframe boats. This search order can be changed by modifying the `graphics` parameter.

Additional graphics parameters may be passed. If `scatterplot3d` is used, these are passed to the `scatterplot3d` function (and ignored when adding to an existing plot). Extra parameters are not passed to `rgl`.

To add to a `scatterplot3d` plot, you must pass the return value from the initial plot as the value of `add`. See the `orientlm` function for an example.

## Value

A current plot number for `rgl`, or a `scatterplot3d` drawing context. In any case, an attribute named `graphics` is added to indicate the drawing device type.

## Note

Requires the `rgl` or `scatterplot3d` package.

## Author(s)

Duncan Murdoch

## Examples

```
x <- eulerzyx(psi=c(0,pi/4,0,0), theta=c(0,0,pi/4,0), phi=c(0,0,0,pi/4))

# Need a 3D renderer; assume scatterplot3d, but others could be used

s <- boat3d(x, 0:3, axes = FALSE, graphics = 'scatterplot3d')
text(s$xyz.convert(0:3, rep(-0.5,4), rep(-0.5,4)),
      label = c('Id','z','y','x'))

## Not run:

# if the rgl package is loaded, this code will work

boat3d(x, 0:3, axes = FALSE, graphics = 'rgl')
rgl.bbox(xat=0:3,xlab=c('Id','z','y','x'),yat=1,zat=1,color='grey')
## End(Not run)
```

---

`coerce-methods`*Methods for Function coerce in Package 'orientlib'*

---

**Description**

Coercion methods are provided between all types of `orientation` objects, and from matrices to the orientation classes.

---

`eulerzxz`*Create an orientation using Euler angles*

---

**Description**

Creates an `eulerzxz-class` object.

**Usage**

```
eulerzxz(phi, theta, psi)
```

**Arguments**

<code>phi</code>	Rotation about Z axis
<code>theta</code>	Rotation about X axis
<code>psi</code>	Further rotation about Z axis

**Details**

The rotations are expressed in radians and applied in the order Z, X, Z.

If `theta` and `psi` are missing, `phi` is taken to be an  $n \times 3$  matrix (or 3 element vector) holding all 3 Euler angles; alternatively, it may be an orientation object.

**Value**

An `eulerzxz-class` object.

**Author(s)**

Duncan Murdoch

**See Also**

`eulerzxz-class`, `eulerzyx-class`, `rotmatrix`, `rotvector`, `quaternion`, `skewvector`, `skewmatrix`

**Examples**

```
x <- eulerzzz(c(1,0,0), c(0,1,0), c(0,0,1))
x
rotmatrix(x)
```

---

`eulerzyx`*Create an orientation using Euler angles*

---

**Description**

Creates an `eulerzyx-class` object.

**Usage**

```
eulerzyx(psi, theta, phi)
```

**Arguments**

<code>psi</code>	Rotation about Z axis
<code>theta</code>	Rotation about Y axis
<code>phi</code>	Rotation about X axis

**Details**

The rotations are expressed in radians and applied in the order Z, Y, X.

If `theta` and `phi` are missing, `psi` is taken to be an  $n \times 3$  matrix (or 3 element vector) holding all 3 Euler angles; alternatively, any orientation object may be used.

**Value**

An `eulerzyx-class` object.

**Author(s)**

Duncan Murdoch

**See Also**

`eulerzyx-class`, `rotmatrix`, `rotvector`, `quaternion`, `skewvector`, `skewmatrix`

**Examples**

```
x <- eulerzyx(c(1,0,0), c(0,1,0), c(0,0,1))
x
rotmatrix(x)
```

---

index-methods      *Methods for indexing orientations*

---

### Description

Methods are defined for indexing all types of orientations.

### Details

Single bracket indexing (e.g. `x[1:3]`) creates a new orientation object of the same class as the original by selecting the appropriate entries. Double bracket indexing (e.g. `x[[3]]`) extracts the chosen data as a matrix or vector, depending on the class of the orientation.

---

length-methods      *Length of orientation object*

---

### Description

The generic `length()` function has methods for orientations; it counts the number of orientations in the object.

---

matrix-classes      *Matrix orientation classes*

---

### Description

An orientation represented by 3 x 3 SO(3) matrices or 3 x 3 skew symmetric matrices

### Objects from the Class

Objects can be created by calls of the form `rotmatrix(x)` or `skewmatrix(x)`. The objects store the matrices in a 3 x 3 x n array.

### Slots

**x:** 3 x 3 x n array holding the matrices.

### Extends

Class "orientation", directly. Class "vector", by class "orientation".

**Methods**

`[, [<-` Extract or assign to subvector

`[[, [[<-` Extract or assign to an entry

**length** The length of the `orientation` vector

**coerce** Coerce methods are defined to convert all `orientation` descendants from one to another, and to coerce an appropriately shaped matrix or array to a `rotmatrix`

**Author(s)**

Duncan Murdoch

**See Also**

`orientation-class`, `vector-classes`, `rotmatrix`, `skewmatrix`

**Examples**

```
x <- rotmatrix(matrix(c(1,0,0, 0,1,0, 0,0,1), 3, 3))
x
skewmatrix(x)
```

---

matrix-methods

*Methods for matrix operations in 'orientlib'*

---

**Description**

Methods are defined for matrix multiplication `%*%` transposition `t()`, and real powers `^`. These operate on the orientations term by term.

---

mean-methods

*Methods for calculating the mean*

---

**Description**

The mean function.

**Methods**

`x = "ANY"` the standard mean function

`x = "orientation"` find the nearest SO(3) matrix to the mean `rotmatrix-class` representation of the orientations

---

`nearest`*Find nearest SO(3) or orthogonal matrix.*

---

## Description

Converts arbitrary 3 x 3 matrices into the nearest SO(3) or orthogonal matrix.

## Usage

```
nearest.SO3(x)
nearest.orthog(x)
```

## Arguments

`x` 3 x 3 matrices stored in a 3 x 3 x n array)

## Details

Uses Stephens' (1979) algorithm to find the nearest (in entry-wise Euclidean sense) SO(3) or orthogonal matrix to a given matrix.

## Value

`nearest.SO3` produces an `orientation-class` object holding the closest orientations.  
`nearest.orthog` produces a 3 x 3 x n array of orthogonal matrices.

## Author(s)

Duncan Murdoch

## References

Stephens (1979). Vector correlation. *Biometrika* 66, 41-48.

## See Also

`orientation-class`

## Examples

```
x <- matrix(rnorm(9), 3, 3)
nearest.orthog(x)
nearest.SO3(x)
x <- -x
nearest.orthog(x)
nearest.SO3(x)
```

---

orientation-class *Class "orientation"*

---

## Description

Abstract class for vectors of various representations of SO(3) (orientation) objects.

## Objects from the Class

A virtual Class: No objects may be created from it.

## Methods

**coerce** Methods are defined to coerce `orientation` objects to any concrete descendant class.

**%\*%** Matrix multiplication acts on `orientation` objects component by component, producing compositions of the rotations.

**^** An orientation is raised to a power by multiplying its component rotation angles by that power.

**t** The transpose of an `orientation` object is its component by component inverse.

**mean** The mean of an `orientation` object is the nearest SO(3) matrix to the element-by-element mean of its 3 x 3 rotation matrix representation.

**weighted.mean** The weighted mean, defined analogously to the mean.

## Author(s)

Duncan Murdoch

## See Also

[matrix-classes](#), [vector-classes](#)

## Examples

```
x <- rotmatrix(diag(3))
x
rotvector(x)
eulerzyx(x)
eulerzxx(x)
quaternion(x)
```

---

orientlib

*Orientation Library*

---

## Description

Representations, conversions and display of orientation data.

## Details

This package contains methods for working with orientation data, i.e. data from  $SO(3)$ . The basic abstract class is the `orientation`; there are several concrete classes (`rotmatrix`, `rotvector`, `eulerzyx`, `eulerzxxz`, `quaternion`, `skewmatrix` and `skewvector`) storing different representations of orientations.

Methods are defined to get the length of a vector of orientations, as well as to extract and replace elements, and to multiply orientations and raise them to real powers.

There are also utility functions `rotation.distance`, `rotation.angle`, `nearest.orthog`, `nearest.SO3`.

There is a plotting method `boat3d` to display orientation data in a 3D plot, and a linear modelling function `orientlm`.

## Release History

- Version 0.9 - Added CITATION file, dropped djmrgl support.
- Versions 0.3 to 0.8 - Changes for CRAN compliance and minor corrections.
- Version 0.2 - Added `mean` and `weighted.mean`.
  - Made `orientation` descend from `vector`.
  - Added `[]` methods.
  - Changed default look of boats.
  - Made `rotmatrix` etc. into conversion functions between orientation types.
  - Added `eulerzxxz` class.
  - Added various parameters to `boat3d`.
  - Added `orientlm` regression function plus transpose `t()` method.
  - Added `rgl` and `scatterplot3d` support to `boat3d` function.
  - Added `skewmatrix`.
- Version 0.1 - First release.

## Note

Plots require either the `rgl` or `scatterplot3d` package.

## Author(s)

Duncan Murdoch

---

orientlm *Linear models for orientation data*


---

## Description

Regression models for matched pairs of orientations.

## Usage

```
orientlm(observed, leftformula, trueorient = rotmatrix(diag(3)),
         rightformula, data = list(), subset, weights, na.action,
         iterations = 5)
```

## Arguments

observed	Observed orientations
leftformula	Formula for “left” model (see below)
trueorient	“True” orientation (see below)
rightformula	Formula for “right” model (see below)
data	Optional data frame for predictors in linear model
subset	Optional logical vector indicating subset of data
weights	Optional weights
na.action	Optional NA function for predictors
iterations	How many iterations to use. Ignored unless both <code>leftformula</code> and <code>rightformula</code> are specified.

## Details

The Prentice (1989) model for matched pairs of orientations was

$$E(V_i) = kA_1^t U_i A_2$$

where  $V_i$  is the observed orientation,  $A_1$  and  $A_2$  are orientation matrices, and  $U_i$  is the “true” orientation, and  $k$  is a constant. It was assumed that errors were symmetrically distributed about the identity matrix.

This function generalizes this model, allowing  $A_1$  and  $A_2$  to depend on regressor variables through `leftformula` and `rightformula` respectively. These formulas should include the predictor variables (right hand side) only, e.g. use `~ x + y + z` rather than `response ~ x + y + z`. Specify the response using the `observed` argument. If both formulas are `~ 1`, i.e. intercepts only, then Prentice’s original model is recovered. More general models are fit by coordinatewise linear regression in the `rotmatrix` representation of the orientation, with fitted values projected onto  $SO(3)$  using the `nearest.SO3` function.

When both left and right models are given, Prentice’s iterative approach is used with a fixed number of iterations. Note that Shin (1999) found that Prentice’s scheme sometimes fails to find the global minimum; this function presumably suffers from the same failing.

**Value**

Returns a list containing the following components:

<code>leftfit</code>	Result of <code>lm</code> call based on <code>leftformula</code>
<code>rightfit</code>	Result of <code>lm</code> call based on <code>rightformula</code>
<code>A1</code>	Fitted values of $A_1$ for each observation
<code>A2</code>	Fitted values of $A_2$ for each observation
<code>predict</code>	Fitted values of $A_1^t U_i A_2$ for each observation

**Author(s)**

Duncan Murdoch

**References**

Prentice, M.J. (1989). Spherical regression on matched pairs of orientation statistics. JRSS B 51, 241-248.

Shin, H.S.H. (1999). Experimental Design for Orientation Models. PhD thesis, Queen's University.

**Examples**

```
x <- rep(1:10,10)
y <- rep(1:10,each=10)
A1 <- skewvector(cbind(x/10,y/10,rep(0,100)))
A2 <- skewvector(c(1,1,1))
trueorientation <- skewvector(matrix(rnorm(300),100))
noise <- skewvector(matrix(rnorm(300)/10,100))
obs <- t(A1) %**% trueorientation %**% A2 %**% noise

fit <- orientlm(obs, ~ x + y, trueorientation, ~ 1)

context <- boat3d(A1, x, z=y, col = 'green', graphics='scatterplot3d')
boat3d(fit$A1, x, z=y, add=context)
```

---

quaternion

*Create an orientation using quaternions*

---

**Description**

Creates a `quaternion-class` object.

**Usage**

```
quaternion(m)
```

**Arguments**

`m` n x 4 matrix or 4 element vector containing a unit quaternion, or an orientation object

**Details**

The rows of `m` are 4 element unit vectors interpreted as follows: the first 3 (`x`, `y`, `z`) define the axis of rotation, and the last element gives the cosine of half the angle of rotation in a counter-clockwise direction when looking down the axis towards the origin.

**Value**

A `quaternion-class` object.

**Author(s)**

Duncan Murdoch

**See Also**

`quaternion-class`, `rotmatrix`, `rotvector`, `eulerzyx`, `eulerzxx`, `skewvector`, `skewmatrix`

**Examples**

```
x <- quaternion(c(1,0,0,0))
x
rotmatrix(x)
```

---

`rotation.distance` *Rotation angle or distance*

---

**Description**

Calculates the angle (in radians) of the rotation taking one orientation to another.

**Usage**

```
rotation.angle(x)
rotation.distance(x, y)
```

**Arguments**

`x`, `y` Two orientation objects

**Details**

If `y` is missing in a call to `rotation.distance`, it is treated as the identity, i.e. `rotation.angle(x)` is calculated.

**Value**

`rotation.distance` returns a vector of length  $\max(\text{length}(x), \text{length}(y))$  containing the angle of the rotation taking corresponding elements of `x` to `y` (with the usual recycling rules if they are different lengths).

`rotation.angle` is equivalent to calculating the `rotation.distance` to the identity matrix.

**Author(s)**

Duncan Murdoch

**See Also**

[orientation-class](#), [rotation.angle](#)

**Examples**

```
rotation.angle(eulerzyx(1,0,0))
rotation.distance(eulerzyx(1,0,0), eulerzyx(0,1,0))
```

---

rotmatrix

*Create an orientation using Euler angles*

---

**Description**

Creates a [rotmatrix-class](#) object.

**Usage**

```
rotmatrix(a)
```

**Arguments**

`a` A 3 x 3 matrix or 3 x 3 x n array of matrices or an orientation object.

**Value**

A [rotmatrix-class](#) object.

**Author(s)**

Duncan Murdoch

**See Also**

[rotmatrix-class](#), [rotvector](#), [eulerzyx](#), [eulerzxz](#), [quaternion](#), [skewvector](#), [skewmatrix](#)

**Examples**

```
x <- rotmatrix(matrix(c(1,0,0, 0,1,0, 0,0,1), 3, 3))
x
```

---

`rotvector`*Create an orientation using vectorized 3x3 matrices*

---

**Description**

Creates a `rotvector-class` object.

**Usage**

```
rotvector(m)
```

**Arguments**

`m` n x 9 matrix or 9 element vector whose rows are vectorized 3x3 matrices, or an orientation object.

**Details**

Converts a matrix whose rows are vectorized 3x3 matrices (in column-major form) into an `rotvector-class` object.

**Value**

A `rotvector-class` object.

**Author(s)**

Duncan Murdoch

**See Also**

`rotvector-class`, `rotmatrix`, `eulerzyx`, `eulerzxx`, `quaternion`, `skewvector`, `skewmatrix`

**Examples**

```
x <- rotvector(c(0,1,0,-1,0,0,0,0,1))
x
rotmatrix(x)
```

---

skewmatrix	<i>Create an orientation using the entries in a skew-symmetric matrix representation</i>
------------	--

---

## Description

Creates a `skewmatrix-class` object.

## Usage

```
skewmatrix(a)
```

## Arguments

`a` 3 x 3 x n array or 3 x 3 matrix containing the entries of a skew-symmetric matrix, or an orientation object.

## Details

The entries `a[, , i]` are 3 x 3 skew-symmetric matrices. The matrix exponential of these give SO(3) matrices.

## Value

A `skewmatrix-class` object.

## Author(s)

Duncan Murdoch

## See Also

`skewvector-class`, `skewvector`, `rotmatrix`, `rotvector`, `eulerzyx`, `eulerzxz`, `quaternion`

## Examples

```
x <- skewmatrix(matrix(c(0, 1, 2, -1, 0, 3, -2, -3, 0), 3, 3))
x
rotmatrix(x)
skewvector(x)
rotation.angle(x)
```

---

skewvector	<i>Create an orientation using the entries in a skew-symmetric matrix representation</i>
------------	--

---

## Description

Creates a `skewvector-class` object.

## Usage

```
skewvector(m)
```

## Arguments

`m` `n` x 3 matrix or 3 element vector containing a the entries of a skew-symmetric matrix, or an orientation object.

## Details

The rows of `m` are 3 element vectors (x,y,z) interpreted as follows: the matrix exponential of the matrix  $((0, -z, y), (z, 0, -x), (-y, x, 0))$  is the SO(3) matrix.

## Value

A `skewvector-class` object.

## Author(s)

Duncan Murdoch

## See Also

`skewvector-class`, `skewmatrix`, `rotmatrix`, `rotvector`, `eulerzyx`, `eulerzxx`, `quaternion`

## Examples

```
x <- skewvector(c(1,0,0))
x
rotmatrix(x)
rotation.angle(x)
```

**Description**

An vector of orientations, each represented by a vector of numbers. Each of these types stores orientations as rows of a matrix in slot `x`.

The `eulerzyx` class uses 3 Euler angles in the roll-pitch-yaw scheme (rotation about Z axis, then Y axis, then X axis).

The `eulerzxx` class uses 3 Euler angles in the X system scheme (rotation about Z axis, then X axis, then Z axis again).

The `rotvector` class uses the 9 components of a 3 x 3 rotation matrix, stored in column-major order.

The `quaternion` class uses the 4 components of a unit quaternion.

The `skewvector` class uses the 3 non-zero components of a skew-symmetric matrix, where  $(x, y, z)$  stores the matrix  $\begin{pmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{pmatrix}$ .

**Objects from the Class**

Objects of each class can be created by calls to the corresponding constructor functions: `eulerzyx`, `eulerzxx`, `rotvector`, `quaternion`, `skewmatrix` and `skewvector`.

**Slots**

**x:** An  $n \times m$  matrix object holding the vector representations, where  $m$  is 3, 4, or 9.

**Extends**

Class "orientation", directly. Class "vector", by class "orientation".

**Methods**

`[, [<-` Extract or assign to subvector

`[[, [[<-` Extract or assign to an entry

**length** The length of the `orientation` vector

**coerce** Coerce methods are defined to convert all `orientation` descendants from one to another, and to coerce an appropriately shaped matrix or array to a `rotmatrix`

**Author(s)**

Duncan Murdoch

**See Also**

Constructor and coercion functions `rotmatrix`, `eulerzyx`, `eulerzxx`, `rotvector`, `quaternion`, and `skewvector`.

Classes `matrix-classes`, `orientation-class`.

**Examples**

```
x <- eulerzyx(0,pi/4,0)
x
eulerzxx(x)
rotmatrix(x)
rotvector(x)
quaternion(x)
skewvector(x)
```

---

weighted.mean-methods

*Weighted mean method*

---

**Description**

The weighted mean function.

**Details**

The weighted mean for orientations is the nearest SO(3) matrix to the entrywise weighted mean of the `rotmatrix-class` matrices.

**Methods**

`x = "ANY", w = "ANY"` the standard `stats::weighted.mean`

`x = "orientation", w = "numeric"` weighted mean for orientations

# Index

- \*Topic **algebra**
  - eulerzxz, 3
  - eulerzyx, 4
  - nearest, 7
  - quaternion, 12
  - rotation.distance, 13
  - rotmatrix, 14
  - rotvector, 14
  - skewmatrix, 15
  - skewvector, 16
- \*Topic **array**
  - eulerzxz, 3
  - eulerzyx, 4
  - nearest, 7
  - orientlib, 9
  - quaternion, 12
  - rotation.distance, 13
  - rotmatrix, 14
  - rotvector, 14
  - skewmatrix, 15
  - skewvector, 16
- \*Topic **classes**
  - matrix-classes, 6
  - orientation-class, 8
  - vector-classes, 17
- \*Topic **dynamic**
  - boat3d, 2
  - orientlib, 9
- \*Topic **hplot**
  - boat3d, 2
  - orientlib, 9
- \*Topic **methods**
  - coerce-methods, 3
  - index-methods, 5
  - length-methods, 5
  - matrix-methods, 7
  - mean-methods, 7
  - weighted.mean-methods, 18
- \*Topic **regression**
  - orientlm, 10
  - [,eulerzxz-method  
(index-methods), 5
  - [,eulerzyx-method  
(index-methods), 5
  - [,quaternion-method  
(index-methods), 5
  - [,rotmatrix-method  
(index-methods), 5
  - [,rotvector-method  
(index-methods), 5
  - [,skewmatrix-method  
(index-methods), 5
  - [,skewvector-method  
(index-methods), 5
  - [<-,eulerzxz-method  
(index-methods), 5
  - [<-,eulerzyx-method  
(index-methods), 5
  - [<-,quaternion-method  
(index-methods), 5
  - [<-,rotmatrix-method  
(index-methods), 5
  - [<-,rotvector-method  
(index-methods), 5
  - [<-,skewmatrix-method  
(index-methods), 5
  - [<-,skewvector-method  
(index-methods), 5
  - [[,eulerzxz-method  
(index-methods), 5
  - [[,eulerzyx-method  
(index-methods), 5
  - [[,quaternion-method  
(index-methods), 5
  - [[,rotmatrix-method  
(index-methods), 5
  - [[,rotvector-method  
(index-methods), 5

- [[, skewmatrix-method  
(*index-methods*), 5
- [[, skewvector-method  
(*index-methods*), 5
- [[<- , eulerzxz-method  
(*index-methods*), 5
- [[<- , eulerzyx-method  
(*index-methods*), 5
- [[<- , quaternion-method  
(*index-methods*), 5
- [[<- , rotmatrix-method  
(*index-methods*), 5
- [[<- , rotvector-method  
(*index-methods*), 5
- [[<- , skewmatrix-method  
(*index-methods*), 5
- [[<- , skewvector-method  
(*index-methods*), 5
- %\*%, orientation, orientation-method  
(*matrix-methods*), 7
- ^, orientation, numeric-method  
(*matrix-methods*), 7
- boat3d, 2, 9, 10
- coerce, array, orientation-method  
(*coerce-methods*), 3
- coerce, eulerzxz, rotmatrix-method  
(*coerce-methods*), 3
- coerce, eulerzyx, rotmatrix-method  
(*coerce-methods*), 3
- coerce, matrix, eulerzxz-method  
(*coerce-methods*), 3
- coerce, matrix, eulerzyx-method  
(*coerce-methods*), 3
- coerce, matrix, orientation-method  
(*coerce-methods*), 3
- coerce, orientation, eulerzxz-method  
(*coerce-methods*), 3
- coerce, orientation, eulerzyx-method  
(*coerce-methods*), 3
- coerce, orientation, quaternion-method  
(*coerce-methods*), 3
- coerce, orientation, rotvector-method  
(*coerce-methods*), 3
- coerce, orientation, skewmatrix-method  
(*coerce-methods*), 3
- coerce, orientation, skewvector-method  
(*coerce-methods*), 3
- coerce, quaternion, rotmatrix-method  
(*coerce-methods*), 3
- coerce, rotvector, rotmatrix-method  
(*coerce-methods*), 3
- coerce, skewmatrix, rotmatrix-method  
(*coerce-methods*), 3
- coerce, skewmatrix, skewvector-method  
(*coerce-methods*), 3
- coerce, skewvector, quaternion-method  
(*coerce-methods*), 3
- coerce, skewvector, rotmatrix-method  
(*coerce-methods*), 3
- coerce-methods, 3
- eulerzxz, 3, 9, 12, 14–18
- eulerzxz, orientation, missing, missing-method  
(*eulerzxz*), 3
- eulerzxz-class, 3, 4
- eulerzxz-class (*vector-classes*),  
17
- eulerzyx, 4, 9, 12, 14–18
- eulerzyx, orientation, missing, missing-method  
(*eulerzyx*), 4
- eulerzyx-class, 4, 5
- eulerzyx-class (*vector-classes*),  
17
- index-methods, 5
- length, eulerzxz-method  
(*length-methods*), 5
- length, eulerzyx-method  
(*length-methods*), 5
- length, quaternion-method  
(*length-methods*), 5
- length, rotmatrix-method  
(*length-methods*), 5
- length, rotvector-method  
(*length-methods*), 5
- length, skewmatrix-method  
(*length-methods*), 5
- length, skewvector-method  
(*length-methods*), 5
- length-methods, 5
- lm, 11
- matrix-classes, 9, 18
- matrix-classes, 6
- matrix-methods, 7

- mean, ANY-method (*mean-methods*), 7
- mean, orientation-method  
(*mean-methods*), 7
- mean-methods, 7
- nearest, 7
- nearest.orthog, 9
- nearest.SO3, 9, 11
- orientation, 2, 6, 9, 18
- orientation (*orientation-class*), 8
- orientation-class, 6, 8, 13, 18
- orientation-class, 8
- orientlib, 9
- orientlm, 2, 9, 10, 10
- quaternion, 4, 5, 9, 12, 14–18
- quaternion, orientation-method  
(*quaternion*), 12
- quaternion-class, 12
- quaternion-class  
(*vector-classes*), 17
- rgl, 2, 3, 10
- rotation.angle, 9, 13
- rotation.angle  
(*rotation.distance*), 13
- rotation.distance, 9, 13
- rotmatrix, 4–6, 9, 11, 12, 14, 15–18
- rotmatrix, orientation-method  
(*rotmatrix*), 14
- rotmatrix-class, 14, 18
- rotmatrix-class, 7
- rotmatrix-class (*matrix-classes*),  
6
- rotvector, 4, 5, 9, 12, 14, 14, 16–18
- rotvector, orientation-method  
(*rotvector*), 14
- rotvector-class, 14, 15
- rotvector-class (*vector-classes*),  
17
- scatterplot3d, 2, 3, 10
- skewmatrix, 4–6, 9, 10, 12, 14, 15, 15, 17
- skewmatrix, orientation-method  
(*skewmatrix*), 15
- skewmatrix-class, 15, 16
- skewmatrix-class  
(*matrix-classes*), 6
- skewvector, 4, 5, 9, 12, 14, 15, 16, 16–18
- skewvector, orientation-method  
(*skewvector*), 16
- skewvector-class, 16, 17
- skewvector-class  
(*vector-classes*), 17
- t, orientation-method  
(*matrix-methods*), 7
- vector-classes, 6, 9
- vector-classes, 17
- weighted.mean, 18
- weighted.mean, ANY, ANY-method  
(*weighted.mean-methods*), 18
- weighted.mean, orientation, numeric-method  
(*weighted.mean-methods*), 18
- weighted.mean-methods, 18