Package ‘Rknots’

October 21, 2016

Type Package

Title Topological Analysis of Knotted Proteins, Biopolymers and 3D Structures

Version 1.3.2

Date 2016-10-12

Depends R (>= 2.14.0), methods, rgl, rSymPy, bio3d

Suggests knitr

VignetteBuilder knitr

Author Federico Comoglio and Maurizio Rinaldi

Maintainer Federico Comoglio <federico.comoglio@gmail.com>

Description
Contains functions for the topological analysis of polymers, with a focus on protein structures.

License GPL-2

LazyData yes

NeedsCompilation no

Repository CRAN

Date/Publication 2016-10-21 14:09:36

R topics documented:

Rknots-package *Rknots: topological characterization of knotted 3D structures with R*

Description

The Rknots package provides functions for the topological analysis of either biological or synthetic knotted polymers, with a major focus on proteins.

Details

Package:	Rknots
Type:	Package
Version:	1.2.2
Date:	2014-12-29
License:	GPL-2
LazyLoad:	yes

The Rknots package contains functions for the topological analysis of polymers, with a major focus on proteins. Notice that Rknots depends on the package bio3d (it can be downloaded from http://mccammon.ucsd.edu/~bgrant/bio3d/) not available through CRAN. The package provides the following set of tools:

1)Functions to import 3D structures or fetch pdb file from the Protein Data Bank (PDB) are implemented with a set of utilities.

2)Structure reduction algorithms: the Alexander-Briggs reduction scheme for knots and links and the Minimal Structure Reduction (MSR) algorithm.

3)Polynomial computation: Alexander, Jones and HOMFLY polynomial via geometrical construction of the Conway's skein triple. Multivariate Alexander polynomial of links. Linking number.

4)Graphical representation of knot and link diagrams. 3D knots and links representation.

5)Large set of utilities.

Credits are due to the bio3d package, to the google code project SymPy and to Rsympy.

Rknots serves as a general purpose framework to identify knots in 3D structures and is not limited to proteins. We have therefore described part of the low-level functions to maximize their integration within other frameworks.

For any further details or feedback please feel free to contact us. Finally, we encourage external contributions to the package.

License: GPL-2

LazyLoad: yes

Author(s)

Federico Comoglio and Maurizio Rinaldi

Maintainer: Federico Comoglio <federico.comoglio@bsse.ethz.ch>

References

Comoglio F. and Rinaldi M. (2012) Rknots: topological analysis of knotted biopolymers with R, Bioinformatics 28 (10), 1400-1401

Comoglio F. and Rinaldi M. (2011) A Topological Framework for the Computation of the HOMFLY Polynomial and Its Application to Proteins PLoS ONE 6(4): e18693, doi:10.1371/journal.pone.0018693 ArXiv:1104.3405

`AlexanderBriggs` *Alexander-Briggs reduction of a polygonal knot or link*

Description

Apply the Alexander-Briggs reduction to a polygonal knot or link. This method is based on the concept of elementary deformation, which consists in the replacement of two sides of a triangle with the third provided that the triangle is empty. From version 1.1 a fast implementation for links is provided.

Usage

```
AlexanderBriggs(points3D, ends = c())
```

Arguments

`points3D` an N x 3 matrix of the x, y, z coordinates of a polygonal link

`ends` a vector of positive integers defining the separators of the polygonal link

Value

A list of two slots:

`points3D` an M x 3 matrix of the x, y, z coordinates of the reduced structure, $M \leq N$

`ends` if a non empty `ends` has been provided as an input, a vector of positive integers defining the separators of the reduced structure

Note

This is a low-level function.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

Maurizio Rinaldi, maurizio.rinaldi@pharm.unipmn.it

References

Reidemeister K (1926), Abh Math Sem Univ Hamburg 5: 24-32.

Alexander JW, Briggs GB (1926) On types of knotted curves. Ann of Math 28: 562-586.

See Also

`msr`

Examples

```
#reducing a knot
k <- makeExampleKnot(k = TRUE)
AlexanderBriggs(points3D = k)

#reducing a link
k <- makeExampleKnot(k = FALSE)
AlexanderBriggs(points3D = k$points3D, ends = k$ends)
```

centroidClosure *Structure closure with the centroid method*

Description

For being treated as a mathematical knot, the protein backbone needs to be closed. This function implements the centroid method (also called CENTER), where the protein endpoints (C-terminus and N-terminus) are extended outside the sphere containing the protein backbone in the direction of the centroid of the structure.

Usage

```
centroidClosure(points3D, w = 2)
```

Arguments

`points3D`	an N x 3 matrix of the x, y, z coordinates of a polygonal link
`w`	given the radius r of the sphere that contains the structure, the endpoints will be extended by `w` r.

Value

an $(N + 2)$ x 3 matrix of the x, y, z coordinates of a polygonal link. The first and the last rows contain the coordinates of the extended endpoints.

Note

This is a low-level function.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Lua RC, Grosberg AY (2006) Statistics of Knots, Geometry of Conformations, and Evolution of Proteins. PLoS Comput Biol 2(5): e45. doi:10.1371/journal.pcbi.0020045

Examples

```
## Not run:
## Import a PDB file from the web
protein <- makeExampleProtein()
closed <- centroidClosure(protein$A)

##Plot the result
plotKnot3D(closed, ends = c(), text = TRUE, showNC = TRUE,
    radius = 0.01, lwd = 5)

## End(Not run)
```

closeAndProject *Close the protein backbone and perform a Principal Component Analysis (requires an object of class 'Knot')*

Description

Close the protein backbone and perform a Principal Component Analysis, This function formally prepares a protein for the computation of an invariant. It requires the protein structure as an object of class 'Knot'.

Usage

```
closeAndProject(protein, ...)
```

Arguments

protein	an object of class 'Knot'
...	additional parameters to be passed to lower level functions (e.g. w for controlling the extension of the endpoints in centroidClosure

Value

an object of class 'Knot' containing the processed structure.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Lua RC, Grosberg AY (2006) Statistics of Knots, Geometry of Conformations, and Evolution of Proteins. PLoS Comput Biol 2(5): e45. doi:10.1371/journal.pcbi.0020045

See Also

Knot-class, centroidClosure

Examples

```
## Not run:
## Loading from the file system
fn <- system.file("extdata", "1AJC_chainA.pdb", package="Rknots")
protein <- loadProtein(fn)
protein <- newKnot(protein$A)
protein.cp <- closeAndProject(protein)

# Plot the results
par(mfrow = c(1,2))
```

```
plot(protein, lwd = 2)
plot(protein.cp, lwd = 2)

## End(Not run)
```

computeInvariant *Compute an invariant of an object of class 'Knot'*

Description

Compute one polynomial invariant (HOMFLY, Jones, Alexander, multivariable Alexander) of a knot or link, or the linking number of a link for object of class 'Knot'.

Usage

```
computeInvariant(knot, invariant, ...)
```

Arguments

`knot` an object of class 'Knot'

`invariant` 'HOMFLY' for the HOMFLY polynomial, 'Alexander' for the Alexander polynomial (knots) or the multivariable Alexander polynomial (links), 'Jones' for the Jones polynomial, 'LK' for the linking number of a link

`...` additional parameters to be passed to lower level functions (e.g. `skein.sign` for the HOMFLY polynomial computation

Value

the computed invariant

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Freyd P, Yetter D, Hoste J, Lickorish WBR, Millett K, et al. (1985) A new polynomial invariant of knots and links. Bull Amer Math Soc (NS) 12: 239-246.

Kauffman, L. Knots and Physics. Teaneck, NJ: World Scientific, p. 19, 1991.

Comoglio F. and Rinaldi M. (2011) A Topological Framework for the Computation of the HOMFLY Polynomial and Its Application to Proteins PLoS ONE 6(4): e18693, doi:10.1371/journal.pone.0018693 ArXiv:1104.3405

Alexander J. W. (1928) Topological invariants of knots and links. Trans. Amer. Math. Soc. 30: 275-306.

Conway J. H. (1970) An enumeration of knots and links, and some of their algebraic properties. Computational Problems in Abstract Algebra (Proc. Conf.,Oxford, 1967), Pergamon, Oxford: 329-358.

Murakami J. (1993) A state model for the multivariable Alexander polynomial. Pacific J. Math. 157, no. 1: 109-135.

Archibald J, (2008) The weight system of the multivariable Alexander polynomial. Acta Math. Vietnamica. 33: 459-470.

Archibald J. (2010) The Multivariable Alexander Polynomial on Tangles. PhD Thesis,Department of Mathematics University of Toronto

Torres G. (1953) On the Alexander polynomial Ann. Math. 57: 57-89.

See Also

Knot-class

Examples

```
## Not run:
knot <- makeExampleKnot(k = TRUE)
knot <- newKnot(knot)

##compute the polynomials
computeInvariant(knot, 'HOMFLY', skein.sign = -1)
computeInvariant(knot, 'Alexander')

## End(Not run)
```

findGaps *Find gaps in proteins based on euclidean distance between residues of the backbone*

Description

Find gaps in proteins based on euclidean distance between residues of the backbone. Return gap positions and accordingly split a chain in subchains.

Usage

```
findGaps(points3D, cutoff = 7)
```

Arguments

points3D	an N x 3 matrix of the x, y, z coordinates of a polygonal link
cutoff	numeric (in Angstrom), distances between consecutive alpha-Carbon atoms greater than cutoff are considered as gaps.

Details

Default cutoff is set to 7, approximately twice the average distance between two consecutive alpha-Carbon atoms.

Value

a list of matrices containing the x, y, z coordinates of the split chains.

Note

Gap finding is highly recommended. Notice that it is performed by default by `loadProtein` and that in the following example this option has been disabled by purpose. This is a low-level function.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

See Also

`loadProtein`

Examples

```
## Import a PDB file from the file system
if(require(bio3d)) {
fn <- system.file("extdata", "1AJC_chainA.pdb", package="Rknots")
protein <- loadProtein(fn, join.gaps = TRUE)
str(protein)
protein.g <- findGaps(protein$A, cutoff = 7)
str(protein.g)
}
```

---

getCoordinates-methods

Accessors for the 'points3D' and 'ends' slots of a Knot object.

Description

The points3D slot holds the N x 3 matrix containing the x, y, z coordinates of the points of a knot or a link.

The ends slot is a vector of integers containing the separators of the link components.

Methods

`signature(object = "Knot")` an object of class Knot.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

See Also

`Knot-class`

Examples

```
knot <- makeExampleKnot(k = TRUE)
knot <- newKnot(knot)

getCoordinates(knot)
getEnds(knot)
```

getKnotType *Given a polynomial invariant, obtain information on the knot type*

Description

Given a polynomial invariant, obtain information on the knot type. Additionally, other polynomials are returned if wished

Usage

```
getKnotType( polynomial, invariant = 'HOMFLY', full.output = FALSE )
```

Arguments

polynomial	a single element character vector containing a polynomial invariant as returned by computeInvariant
invariant	the type of polynomial. It can assume values among 'HOMFLY', 'Jones' and 'Alexander'
full.output	logical, define the output type. If TRUE, the polynomial provided in input is converted to other polynomials. The knot type, an URL to retrieve additional information on the knot type and the full set of invariants are returned. If FALSE, only the knot type is returned. Default to FALSE.

Value

The knot type (the simplest knot having the input polynomial). Optionally, additional informations on it.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

The Rolfsen Knot Table on Knot Atlas, http://www.math.toronto.edu/~drorbn/KAtlas/Knots

See Also

computeInvariant

Examples

```
#suppose you obtain the following HOMFLY polynomial (e.g. via computeInvariant)
polynomial <- 'l^(-4) - 1/l^2 + l^2 - m^2 - m^2/l^2'
getKnotType(polynomial = polynomial, invariant='HOMFLY')
#get more information of the Stevedore's knot
getKnotType(polynomial = polynomial, invariant='HOMFLY', full.output = TRUE)
```

`HOMFLY2Jones` *Convert the HOMFLY polynomial into the Alexander or Jones polynomials*

Description

Convert the HOMFLY polynomial into the Alexander or Jones polynomials

Usage

```
HOMFLY2Alexander(HOMFLY)
HOMFLY2Jones(HOMFLY)
```

Arguments

`HOMFLY`	a single element character vector containing the HOMFLY polynomial

Details

The Alexander polynomial is obtained by substituting into the HOMFLY polynomial

$$l = 1$$

and

$$m = x^{1/2} - x^{-1/2}$$

The following substitutions lead instead to the Jones polynomial

$$l = t^{-1}$$

and

$$m = t^{1/2} - t^{-1/2}$$

Value

The Alexander or the Jones polynomial

Note

These are low-level functions.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

Maurizio Rinaldi, maurizio.rinaldi@pharm.unipmn.it

References

Freyd P, Yetter D, Hoste J, Lickorish WBR, Millett K, et al. (1985) A new polynomial invariant of knots and links. Bull Amer Math Soc (NS) 12: 239-246.

Kauffman, L. Knots and Physics. Teaneck, NJ: World Scientific, p. 19, 1991.

See Also

HOMFLYpolynomial

Examples

```
## Not run:
HOMFLY.trefoil <- "-1/l**4 + 2/l**2 + m**2/l**2"
HOMFLY2Alexander(HOMFLY.trefoil)
HOMFLY2Jones(HOMFLY.trefoil)

## End(Not run)
```

HOMFLY2mirror *Convert the HOMFLY polynomial of a knot into the polynomial of its mirror image*

Description

The HOMFLY polynomial of the mirror image of a knot is computed by substituting

$$l = -l^{-}1$$

Usage

```
HOMFLY2mirror(HOMFLY)
```

Arguments

HOMFLY	a single element character vector containing the HOMFLY polynomial

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

Maurizio Rinaldi, maurizio.rinaldi@pharm.unipmn.it

References

Freyd P, Yetter D, Hoste J, Lickorish WBR, Millett K, et al. (1985) A new polynomial invariant of knots and links. Bull Amer Math Soc (NS) 12: 239-246.

Kauffman, L. Knots and Physics. Teaneck, NJ: World Scientific, p. 19, 1991.

See Also

HOMFLYpolynomial, HOMFLY2Alexander, HOMFLY2Jones

Examples

```
## Not run:
##convert the right-handed trefoil knot polynomial into the left-handed trefoil polynomial
HOMFLY.trefoil <- "-1/l**4 + 2/l**2 + m**2/l**2"
HOMFLY2mirror(HOMFLY.trefoil)

## End(Not run)
```

HOMFLYpolynomial *Compute the HOMFLY polynomial of a polygonal link*

Description

Compute the HOMFLY polynomial of a polygonal link, given its skein tree (generated by skeinIterator)

Usage

```
HOMFLYpolynomial(leaves, tree, skein.sign = -1)
```

Arguments

leaves	a list containing the binary indices of the tree leaves
tree	a list containing the skein tree as returned by skeinIterator.
skein.sign	the skein sign to be used (+1 or -1)

Details

The polynomial computation relies on rSymPy. Please notice that the first time sympy is invoked is expected to be much slower than subsequent ones.

Value

the HOMFLY polynomial

Note

This is a low-level function.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Freyd P, Yetter D, Hoste J, Lickorish WBR, Millett K, et al. (1985) A new polynomial invariant of knots and links. Bull Amer Math Soc (NS) 12: 239-246.

Kauffman, L. Knots and Physics. Teaneck, NJ: World Scientific, p. 19, 1991.

Comoglio F. and Rinaldi M. (2011) A Topological Framework for the Computation of the HOMFLY Polynomial and Its Application to Proteins PLoS ONE 6(4): e18693, doi:10.1371/journal.pone.0018693 ArXiv:1104.3405

See Also

HOMFLY2mirror, HOMFLY2Alexander, HOMFLY2Jones

Examples

```
## Not run:
## Import a PDB file from the file system
protein <- makeExampleProtein()

## Reduce to minimal structure
protein.reduced <- AlexanderBriggs(protein$A)

## Compute the skein tree
tree <- skeinIterator(protein.reduced$points3D, protein.reduced$ends)

## Compute the HOMFLY polynomial
HOMFLYpolynomial(tree$leaves, tree$tree, skein.sign = -1)

## End(Not run)
```

intersectionMatrix *Compute the intersection matrix of a polygonal link*

Description

Compute the intersection matrix of a polygonal link. See details.

Usage

```
intersectionMatrix(points3D, ends = c())
```

Arguments

points3D	an N x 3 matrix of the x, y, z coordinates of a three-dimensional structure
ends	a vector of positive integers defining the separators of the polygonal link

Details

The entries of the intersection matrix are defined as follows. Given two sets of edges A and B we can compute the intersection matrix $I = I(A, B)$ by setting

$$(I(A, B))_{i,j} = 0$$

if A_i and B_j do not intersect transversally.

$$(I(A, B))_{i,j} = +1$$

if A_i lays over B_j.

$$(I(A, B))_{i,j} = -1$$

if A_i lays under B_j. Finally, if $A = B$ we get the skew symmetric square matrix $I(A)$

Value

An N-1 x N-1 matrix

Note

This is a low-level function.

Note

If ends is not null, the corresponding rows and columns of the intersection matrix are set to zero.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Comoglio F. and Rinaldi M. A Topological Framework for the Computation of the HOMFLY Polynomial and Its Application to Proteins (2011) PLoS ONE 6(4): e18693, doi:10.1371/journal.pone.0018693 ArXiv:1104.3405

Examples

```
##Compute the intersection matrix of a random structure of 20 points
points <- matrix(runif(60,-1,1), ncol = 3)

intersectionMatrix(points)

##Compute the intersection matrix of the trefoil knot
data(Rolfsen.table, package = "Rknots")
trefoil <- Rolfsen.table$"3.1"

intersectionMatrix(trefoil)
```

Knot-class *Class "Knot" – a container for knot and link coordinates and ends*

Description

This is the main class for the present package.

Objects from the Class

Objects should be created with calls to newKnot.

Note

Note: This is a summary for reference. For a detailed explanation, please see the vignette. The slot points3D contains the 3D coordinates of points of a polygonal knot or link as an N x 3 matrix. The slot ends contains a vector of link separators. This is automatically set to numeric(0) for knots.

See Also

newKnot, makeExampleKnot

Examples

```
# create an object of class 'Knot' by using new
link <- makeExampleKnot( k = FALSE )
new('Knot', points3D = link$points3D, ends = link$ends)

#or by means of the constructor
newKnot(points3D = link$points3D, ends = link$ends)

#for knots, it is sufficient to specify the 3D coordinates
#ends are set by default to numeric(0)
knot <- makeExampleKnot( k = TRUE )
newKnot(points3D = knot)

#for creating an example, use makeExampleKnot.
#knot:
makeExampleKnot(k = TRUE)
#link:
makeExampleKnot(k = FALSE)
```

`link.table` *Three dimensional coordinates and separators of polygonal links.*

Description

This data set contains the three dimensional coordinates of 130 polygonal links (up to 4 components) along with the relevant separators.

Usage

```
data(link.table)
```

Format

A list of lists. Each element is a matrix containing coordinates and separators of a polygonal link.

Note

A single entry can be accessed by [[or using the corresponding link name.

`names(link.table)` returns the available link names. For example, the Hopf link can be accessed either by `link.table[[1]]` or by `link.table$"2.2.1"`.

References

The COLAB SFU website, http://www.colab.sfu.ca

Examples

```
data(link.table)
names(link.table)
str(link.table)
```

`linkingNumber` *Compute the linking number of a polygonal link*

Description

Compute the linking number of a polygonal link

Usage

```
linkingNumber(points3D, ends, M = c())
```

Arguments

points3D an N x 3 matrix of the x, y, z coordinates of a polygonal link

ends a vector of positive integers defining the separators of the polygonal link

M the intersection matrix of the polygonal link. If no matrix is provided, the function will compute it (default)

Details

The linking number is defined for a two-component oriented link as the sum of +1 crossings and -1 crossing over all crossings between the two links divided by 2. For components α and β,

$$lk(\alpha, \beta) = \frac{1}{2} \sum_{c \in \alpha \cap \beta} \epsilon(c)$$

where $\alpha \cap \beta$ is the set of crossings of α with β, and $\epsilon(c)$ is the sign of the crossing.

Value

lk the linking number of the polygonal link

Note

This is a low-level function.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Weisstein, Eric W. "Linking Number." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/LinkingNumber.html

Kauffman, L. Knots and Physics. Teaneck, NJ: World Scientific, p. 19, 1991.

See Also

intersectionMatrix

Examples

```
link <- makeExampleKnot(k = FALSE)
linkingNumber(points3D = link$points3D, ends = link$ends)
```

loadProtein *Import a pdb file or fetch it from the Protein Data Bank. Check for structural gaps. Build and return a backbone trace for each polypeptide chain.*

Description

Import a pdb file either by fetching the Protein Data Bank or from the file system and return the alpha-Carbon atoms coordinates.

Usage

```
loadProtein(pdbID, join.gaps = FALSE, cutoff = 7, ...)
```

Arguments

`pdbID`	a single element character vector containing the path to the file in the file system and the name of the PDB file to be read, or the four letter PDB identifier for online file access (fetched from the PDB). For the latter, the `pdbID` is not case sensitive.
`join.gaps`	logical (default FALSE). If TRUE, gaps are joined together with a straight line.
`cutoff`	numeric (in Angstrom), distances between consecutive alpha-Carbon atoms greater than `cutoff` are considered as gaps.
`...`	any other parameter for the `read.pdb` function of bio3d. See the manual for details.

Details

If the selected PDB entry contains more than one polypeptide chain, all the available chains are loaded or downloaded. The PDB atom field is then filtered individually to retain the coordinates of the protein C-alpha trace. If `join.gaps` is FALSE, additional chains (one for each split) will be produced and labeled according to the following: if C is a chain and C is split in j subchains due to gaps, the resulting subchains will be labeled with $C_1, \dots, C_j$.

Notice that any other model or any three-dimensional structure can be directly loaded in R as an N x 3 matrix, where the columns represents the x, y, z coordinates. To this purpose, please see the help pages of `read.table`, `read.delim`, `read.csv`, etc.

Value

A list of length equal to the number of polypeptide chain (if gaps are not present or not considered) or greater (if gaps are present), where each element contains an N x 3 matrix with the x, y, z coordinates of the chain (or subchain) model.

Note

This function makes use of the `read.pdb` function in the `bio3d` package. See references for additional details and credits.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Grant, Rodrigues, ElSawy, McCammon, Caves (2006) *Bio3D: An R package for the comparative analysis of protein structures.*, Bioinformatics 22, 2695-2696

Examples

```
## Not run:
## Loading from the file system
fn <- system.file("extdata", "1AJC_chainA.pdb", package="Rknots")
protein <- loadProtein(fn)

## Fetching the PDB
protein<- loadProtein('2K0A')
#more than one chain (notice the split)
protein <- loadProtein('1GZ0')
str(protein)

#A protein with two subunits. The first one is split in two subchains.
protein <- loadProtein('1AJC', join.gaps = FALSE, cutoff = 7)
str(protein)

## End(Not run)
```

makeExampleKnot *Randomly select a knot or a link from the local repository*

Description

This function returns a randomly selected knot or link from the Rolfsen table or from the link repository. It is used for the examples in some of the package help pages and can be used to test the different functions.

Usage

```
makeExampleKnot(k = TRUE)
```

Arguments

k	logical, if TRUE a knot is returned, otherwise it returns a link.

Value

A matrix with the 3D coordinates of a knot if k is FALSE. A list with two slots: the 3D coordinates of a link and a vector of separators if k is TRUE.

See Also

Knot-class, newKnot

Examples

```
## Not run:
#an example knot
knot <- makeExampleKnot(k = TRUE)

#an example link
link <- makeExampleKnot(k = FALSE)

## End(Not run)
```

---

makeExampleProtein — *Load the coordinates of a protein structure used in many of the package help pages*

Description

This function returns the coordinates of the Rds3p protein (PDB ID: 2K0A). It is used for the examples in some of the package help pages and can be used to test the different functions.

Usage

```
makeExampleProtein()
```

Value

A list containing a single element, namely a matrix with the 3D coordinates of a the Rds3p protein (109 aminoacids / points)

Examples

```
## Not run:
makeExampleProtein()

## End(Not run)
```

msr *Minimal Structure Reduction*

Description

Reduce a polygonal link to its minimal structure by applying Generalized Reidemeister Moves.

Usage

```
msr(points3D, ends = c(), n = 100)
```

Arguments

`points3D`	an N x 3 matrix of the x, y, z coordinates of a polygonal link
`ends`	a vector of positive integers defining the separators of the polygonal link
`n`	the number of iterations

Details

A minimal structure for a polygonal link L is a nested sequence of subsets of L:

$$L \supset L_1 \supset \ldots \supset L_N$$

that cannot be extended. Each inclusion corresponds to a Generalized Reidemeister Move.

Value

Returns a list of three elements

`points3D`	an M x 3 matrix of the x, y, z coordinates of the reduced structure
`ends`	a vector of positive integers (if a non empty `ends` has been provided as an argument defining the separators of the reduced structure
`M`	the intersection matrix of the reduced structure

Note

This is a low-level function.

Note

The default number of iterations is 100. This value almost always allows to reach the minimal structure. If wished, a partial reduction can be achieved with `n` between 2 and 5. This is particularly suitable for graphical representations of the reduction process.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Comoglio F. and Rinaldi M. A Topological Framework for the Computation of the HOMFLY Polynomial and Its Application to Proteins (2011) PLoS ONE 6(4): e18693, doi:10.1371/journal.pone.0018693 ArXiv:1104.3405

Examples

```
## Not run:
knot <- makeExampleKnot()
reduced <- msr(points3D = knot)

## 3D plot of both the trefoil and its reduced structure
plotKnot3D(knot, ends = c(), radius = 0.01, lwd = 2.5)
plotKnot3D(reduced$points3D, reduced$ends, colors = 'red', radius = 0.1, lwd = 2.5)

## End(Not run)
```

---

mVA *Compute the multivariable Alexander polynomial of a polygonal link*

Description

Computes the multivariable Alexander polynomial (MVA) of a polygonal link.

Usage

```
mVA(points3D, ends, normalized = FALSE, return.A = FALSE)
```

Arguments

`points3D`	an N x 3 matrix of the x, y, z coordinates of a polygonal link
`ends`	a vector of positive integers defining the separators of the polygonal link
`normalized`	logical, if FALSE (default) the multivariable non normalized MVA is returned, the normalized MVA otherwise
`return.A`	logical, if TRUE the Alexander matrix is returned in a format that can be parsed to sympy

Details

The polynomial computation relies on rSymPy. Please notice that the first time sympy is invoked is expected to be much slower than subsequent ones.

Value

the multivariable Alexander polynomial

Note

This is a low-level function. If you wish to make computations faster, reduce the structure first with `AlexanderBriggs` or `msr`.

Author(s)

Maurizio Rinaldi, maurizio.rinaldi@pharm.unipmn.it

References

Alexander J. W. (1928) Topological invariants of knots and links. Trans. Amer. Math. Soc. 30: 275-306.

Conway J. H. (1970) An enumeration of knots and links, and some of their algebraic properties. Computational Problems in Abstract Algebra (Proc. Conf.,Oxford, 1967), Pergamon, Oxford: 329-358.

Murakami J. (1993) A state model for the multivariable Alexander polynomial. Pacific J. Math. 157, no. 1: 109-135.

Archibald J. (2008) The weight system of the multivariable Alexander polynomial. Acta Math. Vietnamica. 33: 459-470.

Archibald J. (2010) The Multivariable Alexander Polynomial on Tangles. PhD Thesis, Department of Mathematics University of Toronto

Torres G. (1953) On the Alexander polynomial Ann. Math. 57: 57-89.

Comoglio F. and Rinaldi M. (2011) A Topological Framework for the Computation of the HOMFLY Polynomial and Its Application to Proteins, PLoS ONE 6(4): e18693, doi:10.1371/journal.pone.0018693 ArXiv:1104.3405

See Also

`msr`, `AlexanderBriggs`

Examples

```
## Not run:
link <- makeExampleKnot(k = FALSE)
mVA(points3D = link$points3D, ends = link$ends)
mVA(points3D = link$points3D, ends = link$ends, normalized = TRUE)
mVA(points3D = link$points3D, ends = link$ends, normalized = TRUE, return.A = TRUE)

## End(Not run)
```

newKnot *Create a Knot object*

Description

This function creates a knot object from a matrix of 3D points coordinates and a vector of separators (for links)

Usage

```
newKnot(points3D, ends)
```

Arguments

`points3D`	An N x 3 matrix containing the x, y, z coordinates of points of a polygonal knot or link. Each row then contains the 3D coordinates of a single point of the structure.
`ends`	A vector of integers containing the separators of the link components. A separator can be seen as the index of the edge that if not removed would connect a component with the following. This slot is automatically set to `numeric(0)` for knots

Value

an object of class Knot

Note

Note: This is a summary for reference. For a more detailed explanation, please see the vignette.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

See Also

`Knot-class`, `makeExampleKnot`

Examples

```
knot <- makeExampleKnot(k = TRUE)
newKnot(knot)

link <- makeExampleKnot(k = FALSE)
newKnot(link$points3D, link$ends)
```

PCAProjection *Two-dimensional projection by Principal Component Analysis*

Description

Two-dimensional projection by Principal Component Analysis

Usage

```
PCAProjection(points3D)
```

Arguments

`points3D`	an N x 3 matrix of the x, y, z coordinates of a polygonal link

Value

`points3D.rot`	an N x 3 matrix containing the x, y, z coordinates of the rotated structure. The first two columns represent the 2D projection of the structure.

Note

This is a low-level function.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Comoglio F. and Rinaldi M. A Topological Framework for the Computation of the HOMFLY Polynomial and Its Application to Proteins (2011) PLoS ONE 6(4): e18693, doi:10.1371/journal.pone.0018693 ArXiv:1104.3405

Examples

```
## Not run:
protein <- loadProtein('3MDZ')
par(mfrow = c(1,2))
plotDiagram(protein$A, ends = c(), lwd = 2.5, main = 'Original')

protein.rot <- PCAProjection(protein$A)
plotDiagram(protein.rot, ends = c(), lwd = 2.5, main = 'Reduced')

## End(Not run)
```

plot-methods *Plot the diagram of a Knot object.*

Description

Plot method that produces the knot or link diagram for an object of class 'Knot'

Methods

```
signature(x = "Knot")
```

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

See Also

`Knot-class`

Examples

```
knot <- makeExampleKnot(k = TRUE)
knot <- newKnot(knot)
link <- makeExampleKnot(k = FALSE)
link <- newKnot(link$points3D, link$ends)

plot(knot)
plot(knot, lwd = 2.5) #to emphasize the overcrossings
plot(link)
plot(link, lwd = 2.5) #to emphasize the overcrossings
```

plot3D *3D plot of an object of class 'Knot'*

Description

Returns the 3D plot of an object of class 'Knot'

Usage

```
plot3D(knot, ...)
```

Arguments

`knot`	an object of class 'Knot'
`...`	other parameters for `lines3d` or `spheres3d`

Value

called for its effect.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

Examples

```
## Not run:
link <- makeExampleKnot(k = FALSE)
link <- newKnot(link$points3D, link$ends)

##plot with rgl
require(rgl)
plot3D(link)
plot3D(link, radius = 0.1)
plot3D(link, radius = 0.1, lwd = 10)

## End(Not run)
```

`plotDiagram` *Plot a knot or a link diagram*

Description

Plot a knot or a link diagram, along with component orientation.

Usage

```
plotDiagram(points3D, ends, pca = FALSE, size = 1, colors = c(), return.vars=FALSE, ...)
```

Arguments

`points3D`	an N x 3 matrix of the x, y, z coordinates of a polygonal link
`ends`	a vector of positive integers defining the separators of the polygonal link
`pca`	logical, if TRUE a Principal Component Analysis is applied to rotate the points. Generally, results in a diagram simpler to be interpreted. This is especially true for large structures.
`size`	a scaling factor s of the plot region to be used for the plot. Default 1. The plot region within the active plot window is defined by $$\frac{min(\Delta X, \Delta Y)}{s}$$. Values of $s > 1$ reduce the plot region.
`colors`	a vector of colors, one for each component. If not supplied, integers from 1 to n are used, where n is the number of link components
`return.vars`	logical, if TRUE returns the extended structure described in details.
`...`	other graphical parameters for `plot`.

Details

Internally,this function localizes the crossings and insert auxiliary points in proximity of each undercross. The resulting extended structure can be accessed using `return.vars=TRUE`. The use of the parameter `lwd` is recommended in order to increase the line width of the overcrossings. This can be particularly useful when large structures are represented. See the examples below.

Value

Called for its effect. If `return.vars` is TRUE, it returns a list with the following elements

`points3D` an $(N+2k)$ x 3 matrix of the x, y, z coordinates of the extended polygonal link, where k is the number of crossings

`ends` a vector of positive integers defining the separators of the extended polygonal link

`undercross` a vector of positive integers defining the row in `points3D` containing the coordinates of the undercrossings

`overcross` a vector of positive integers defining the row in `points3D` containing the coordinates of the overcrossings

`component` a vector of positive integers of length $N+2k$ where each entry i corresponds to the component the i-th row of `points3D`.

Note

This is a low-level function.

Author(s)

Maurizio Rinaldi, maurizio.rinaldi@pharm.unipmn.it Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Weisstein, Eric W. "Knot diagram." From MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/KnotDiagram.html

See Also

`plotKnot3D`

Examples

```
## Not run:
##Represent 12 randomly selected link diagrams
par(mfrow = c(3,4))

for(i in 1 : 12) {
link <- makeExampleKnot( k = FALSE )
link <- AlexanderBriggs(link$points3D, link$ends)
plotDiagram(link$points3D, link$ends, main = i, lwd = 1.5)
}
```

```
## End(Not run)
```

plotKnot3D *3D plot of a polygonal link*

Description

3D plot of a polygonal link

Usage

```
plotKnot3D(points3D, ends = c(), text = FALSE, showNC = FALSE, colors = list(), ...)
```

Arguments

`points3D` an N x 3 matrix of the x, y, z coordinates of a polygonal link

`ends` a vector of positive integers defining the separators of the polygonal link. Default empty (polygonal knot)

`text` logical, if TRUE the numbering of the points is shown

`showNC` logical, if TRUE the N-terminus and C-terminus of a protein are labeled in red

`colors` a list of color vectors, one for each component. Each vector i can be of length 1, containing the color to be used for the component i or of length equal to the number of points of the component i, thus specifying the color to be used for each edge. If not supplied, single colors from 1 to n are used, where n is the number of link components

`...` other parameters for `lines3d` or `spheres3d`

Value

Called for its effect.

Note

This is a low-level function.

Author(s)

Federico Comoglio, federico.comogli@bsse.ethz.ch

See Also

`plotDiagram`

Examples

```
## Not run:
##3D plot of the Hopf link

hopf <- matrix(c(0.000084, -1.667027, -0.000300,
    0.179585, 0.249735, 0.991541,
    -0.179478, 0.250355, -0.991211,
    0.000084, -1.667027, -0.000300,
    -0.991561, -0.250457, 0.179413,
    -0.000415, 1.666817, 0.000107,
    0.991784, -0.249435, -0.179541,
    -0.991561, -0.250457, 0.179413),
    ncol = 3, byrow = TRUE)

plotKnot3D(hopf, ends = 4, text = FALSE, showNC = FALSE, lwd = 5, radius = 0.05)

##Stevedore knot
data(Rolfsen.table, package = ”Rknots”)
stevedore <- Rolfsen.table$”6.1”
plotKnot3D(stevedore, ends = c(), text = FALSE, showNC = FALSE,
    radius = 0.05, lwd = 5)

#The same plot, without specifying any parameter for the rgl primitive shapes
# (the default sphere radius is 1).
plotKnot3D(stevedore, ends = c(), text = FALSE)

## End(Not run)
```

print-methods *print the content of the slots of a Knot object.*

Description

Print method for an object of class ’Knot’

Methods

```
signature(x = ”Knot”)
```

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

See Also

Knot-class

Examples

```
knot <- makeExampleKnot(k = TRUE)
knot <- newKnot(knot)
print(knot)

link <- makeExampleKnot(k = FALSE)
link <- newKnot(link$points3D, link$ends)
print(link)
```

---

reduceStructure *Structure reduction of an object of class ’Knot’*

Description

Apply the Alexander-Briggs reduction or the Minimal Structure Reduction algorithm to a knot or a link stored in a ’Knot’ object.

Usage

```
reduceStructure(knot, algorithm = 'AB')
```

Arguments

knot	an object of class ’Knot’
algorithm	’AB’ (default) for the Alexander-Briggs reduction scheme, ’MSR’ for the Minimal Structure Reduction algorithm.

Value

an object of class ’Knot’ containing the reduced structure

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Reidemeister K (1926), Abh Math Sem Univ Hamburg 5: 24-32.

Alexander JW, Briggs GB (1926) On types of knotted curves. Ann of Math 28: 562-586.

Comoglio F. and Rinaldi M. A Topological Framework for the Computation of the HOMFLY Polynomial and Its Application to Proteins (2011) PLoS ONE 6(4): e18693, doi:10.1371/journal.pone.0018693 ArXiv:1104.3405

See Also

Knot-class

Examples

```
knot <- makeExampleKnot(k = TRUE)
knot <- newKnot(knot)

##reduce the structure
reduceStructure(knot, 'AB')
reduceStructure(knot, 'MSR')
```

`rMatrix` *Rotation matrix along the z axis*

Description

Compute the rotation matrix by an angle θ along the z axis

Usage

```
rMatrix(theta)
```

Arguments

`theta`	The rotation angle, in radians

Details

The counterclockwise 3 x 3 rotation matrix by an angle θ along the z axis is defined as:

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Value

A 3 x 3 matrix

Note

This is a low-level function.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Weisstein, Eric W. "Rotation Matrix." From MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/RotationMatrix.html

Examples

```
##Rotation matrix by theta = 45 degrees
rMatrix(pi/4)

##to rotate clockwise by 45 degrees
rMatrix(-pi/4)
```

Rolfsen.table *Knots (up to 10 crossings) three dimensional coordinates, enumerated accordingly to Rolfsen.*

Description

This data set contains the three dimensional coordinates of any knot with a number o crossing less or equal to 10. The knots are labeled according to the Rolfsen's table of knots (see references).

The knot minimal sticks representation for each knot is provided in the data set `Rolfsen.table.ms`.

Usage

```
data(Rolfsen.table)
Rolfsen.table
Rolfsen.table.ms
```

Format

A list of 250 matrices (N x 3)

Note

A single knot entry can be accessed by indexing the list with the corresponding knot name.

`names(Rolfsen.table)` returns the available knot names. For example, the minimal stickies representation of the trefoil knot can be accessed by `Rolfsen.table.ms$"3.1"`.

References

The Knot Server, http://www.colab.sfu.ca/KnotPlot/KnotServer

The COLAB SFU website, http://www.colab.sfu.ca

The Rolfsen Knot Table on Knot Atlas, http://www.math.toronto.edu/~drorbn/KAtlas/Knots

`setCoordinates<--methods` *Setters for the 'points3D' and 'ends' slots of a Knot object.*

Description

The points3D slot holds the N x 3 matrix containing the x, y, z coordinates of the points.

The ends slot is a vector of integers containing the separators of the link components.

Methods

```
signature(object = "Knot")
```

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

See Also

`Knot-class`

Examples

```
link <- makeExampleKnot(k = FALSE)
link <- newKnot(link$points3D, link$ends)

setCoordinates(link) <- matrix(runif(90), ncol = 3)
setEnds(link) <- c(10, 20)
```

`skeinIterator` *Iterate the skein relation to build a skein tree of a polygonal link*

Description

This function is required for the computation of the skein tree of a polygonal link. The tree is built by iterating the skein relation and geometrically constructing the L_0 and L_{sw} configuration (of a Conway skein triple).

Usage

```
skeinIterator(points3D, ends, M = c())
```

Arguments

`points3D`	an N x 3 matrix of the x, y, z coordinates of a polygonal link
`ends`	a vector of positive integers defining the separators of the polygonal link
`M`	the intersection matrix of the polygonal link

Value

`leaves` a list containing the binary indices of the tree leaves

`tree` a list containing the skein tree. Each slot contains the slots `points3D`, `ends`, `signs` and `M`, which are respectively the coordinates, separators of the current configuration, the skein signs of the ancestors (inner vertices) and the intersection matrix of the current configuration.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Freyd P, Yetter D, Hoste J, Lickorish WBR, Millett K, et al. (1985) A new polynomial invariant of knots and links. Bull Amer Math Soc (NS) 12: 239-246.

Kauffman, L. Knots and Physics. Teaneck, NJ: World Scientific, p. 19, 1991.

Comoglio F. and Rinaldi M. A Topological Framework for the Computation of the HOMFLY Polynomial and Its Application to Proteins (2011) PLoS ONE 6(4): e18693, doi:10.1371/journal.pone.0018693 ArXiv:1104.3405

See Also

`HOMFLYpolynomial`,

Examples

```
protein <- makeExampleProtein()
protein <- AlexanderBriggs(protein$A)

## Compute the skein tree
tree <- skeinIterator(protein$points3D, protein$ends)
str(tree)
```

Index