Package: braids (via r-universe)

August 27, 2024

Title The Braid Groups

Version 1.0.0

Description Deals with the braid groups. Includes creation of some specific braids, group operations, free reduction, and Bronfman polynomials. Braid theory has applications in fluid mechanics and quantum physics. The code is adapted from the 'Haskell' library 'combinat', and is based on Birman and Brendle (2005) <doi:10.48550/arXiv.math/0409205>.

License GPL-3

URL https://github.com/stla/braids

BugReports https://github.com/stla/braids/issues

Imports maybe

Encoding UTF-8

RoxygenNote 7.3.1

Repository https://stla.r-universe.dev

RemoteUrl https://github.com/stla/braids

RemoteRef HEAD

RemoteSha 24232c0e1f25f4c2525a2a71bfe9a1358232f339

Contents

`allBraidWords` *Braid words of given length*

Description

All braid words of the given length.

Usage

```
allBraidWords(n, l)
```

Arguments

`n`	number of strands, integer `>=2`
`l`	length of the words

Value

A list of `braid` objects.

Examples

```
allBraidWords(3, 2)
```

allPositiveBraidWords *Positive braid words of given length*

Description

All positive braid words of the given length.

Usage

```
allPositiveBraidWords(n, l)
```

Arguments

n	number of strands, integer >=2
l	length of the words

Value

A list of braid objects.

Examples

```
allPositiveBraidWords(3, 4)
```

braidASCII *ASCII braid*

Description

Prints an ASCII figure of a braid.

Usage

```
braidASCII(braid)
```

Arguments

braid	a braid object

Value

No value is returned, just prints the ASCII figure.

Examples

```
braid <- mkBraid(4, c(1, -2))
braidASCII(braid)
```

braidPermutation *Braid permutation*

Description

Returns the left-to-right permutation associated to a braid.

Usage

```
braidPermutation(braid)
```

Arguments

braid	a braid object (e.g. created with mkBraid)

Value

A permutation.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
braidPermutation(braid)
```

bronfmanPolynomials *Bronfman polynomials*

Description

The Bronfman polynomial of a braid group is the reciprocal of the growth function of the positive braids. This function computes the Bronfman polynomial of the braid group on n strands for n going to 1 to N.

Usage

```
bronfmanPolynomials(N)
```

Arguments

N	maximum number of strands

Value

A list of integer vectors representing the Bronfman polynomials; each vector represents the polynomial coefficients in increasing order.

Examples

```
bronfmanPolynomials(3) # 1, 1 - X, 1 - 2X + X^3
```

composeManyBraids *Composition of many braids.*

Description

Composes many braids, doing free reduction on the result.

Usage

```
composeManyBraids(braids)
```

Arguments

braids	list of braid objects with the same number of strands

Value

A braid object.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
composeManyBraids(list(braid, braid, braid))
```

composeTwoBraids *Composition of two braids*

Description

Composes two braids, doing free reduction on the result.

Usage

```
composeTwoBraids(braid1, braid2)
```

Arguments

braid1, braid2	braid objects with the same number of strands

Value

A braid object.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
composeTwoBraids(braid, braid)
```

doubleSigma *Double generator*

Description

Generator $\sigma_{s,t}$ in the Birman-Ko-Lee new presentation. It twists the strands `s` and `t` while going over all other strands (for `t=s+1`, this is σ_s).

Usage

```
doubleSigma(n, s, t)
```

Arguments

`n`	number of strands, integer `>=2`
`s`, `t`	indices of two strands, `s < t`

Value

A `braid` object.

Examples

```
doubleSigma(5, 1, 3)
```

freeReduceBraidWord *Free reduction of a braid*

Description

Applies free reduction to a braid, i.e. removes pairs of consecutive generators inverse of each other.

Usage

```
freeReduceBraidWord(braid)
```

Arguments

`braid`	a `braid` object (e.g. created with `mkBraid`)

Value

A `braid` object.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
freeReduceBraidWord(braid)
```

halfTwist *Half-twist*

Description

The (positive) half-twist of all the braid strands, usually denoted by Δ.

Usage

```
halfTwist(n)
```

Arguments

n	number of strands, integer >=2

Value

A braid object.

Examples

```
halfTwist(4)
```

inverseBraid *Inverse braid*

Description

The inverse of a braid (without performing reduction).

Usage

```
inverseBraid(braid)
```

Arguments

braid	a braid object

Value

A braid object.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
ibraid <- inverseBraid(braid)
composeTwoBraids(braid, ibraid)
```

isPermutationBraid *Whether a braid is a permutation braid*

Description

Checks whether a braid is a permutation braid, that is, a positive braid where any two strands cross at most one, and positively.

Usage

```
isPermutationBraid(braid)
```

Arguments

braid	a braid object

Value

A Boolean value.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
isPermutationBraid(braid)
```

isPositiveBraidWord *Whether a braid is positive*

Description

Checks whether a braid has only positive Artin generators.

Usage

```
isPositiveBraidWord(braid)
```

Arguments

braid	a braid object

Value

A Boolean value.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
isPositiveBraidWord(braid)
```

isPureBraid *Whether a braid is pure*

Description

Checks whether a braid is pure, i.e. its permutation is trivial.

Usage

```
isPureBraid(braid)
```

Arguments

braid	a braid object

Value

A Boolean value.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
isPureBraid(braid)
```

linkingMatrix *Linking matrix*

Description

Linking numbers between all pairs of strands of a braid.

Usage

```
linkingMatrix(braid)
```

Arguments

braid	a braid object

Value

A matrix.

See Also

strandLinking to get the linking number between two strands of the braid.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
linkingMatrix(braid)
```

`mkBraid` *Make a braid*

Description

Make a braid.

Usage

```
mkBraid(n, artingens)
```

Arguments

`n`	number of strands, an integer, at least 2
`artingens`	Artin generators given by a vector of non-zero integers; a positive integer `i` corresponds to the standard positive Artin generator of a braid which represents twisting the neighbour strands `i` and `i+1`, such that strand `i` goes *under* strand `i+1`; a negative integer `-i` corresponds to the inverse.

Value

A `braid` object.

Examples

```
mkBraid(n = 4, c(2, -3))
```

`numberOfStrands` *Number of strands*

Description

The number of strands of a braid.

Usage

```
numberOfStrands(braid)
```

Arguments

`braid`	a `braid` object (e.g. created with `mkBraid`)

Value

An integer.

permutationBraid *Permutation braid*

Description

Makes a permutation braid from a permutation.

Usage

```
permutationBraid(perm)
```

Arguments

perm	a permutation

Value

A braid object.

Examples

```
perm <- c(3, 1, 4, 2)
braid <- permutationBraid(perm)
isPermutationBraid(braid)
braidPermutation(braid)
```

strandLinking *Linking number between two strands*

Description

The linking number between two strands of a braid.

Usage

```
strandLinking(braid, i, j)
```

Arguments

braid	a braid object
i, j	indices of two strands

Value

An integer.

See Also

linkingMatrix to get the linking numbers between all pairs of strands of the braid.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
strandLinking(braid, 1, 3)
```

---

tau *Inner automorphism*

Description

The inner automorphism defined by $\tau X = \Delta^{-1} X \Delta$, where Δ is the positive half-twist; it sends each generator σ_j to σ_{n-j}.

Usage

```
tau(braid)
```

Arguments

braid	a braid object

Value

A braid object.

Examples

```
braid <- mkBraid(4, c(2, -3, 3))
tau(braid)
```

Index