Package ‘weyl’

January 21, 2022

Type Package
Title The Weyl Algebra
Version 0.0-1
Depends spray (>= 1.0-19), methods, R (>= 3.5.0)
Maintainer Robin K. S. Hankin <hankin.robin@gmail.com>
Description A suite of routines for Weyl algebras. Notation follows
License GPL (>= 2)
LazyData yes
Suggests knitr,rmarkdown,testthat
VignetteBuilder knitr
Imports mathjaxr, disordR (>= 0.0-8), freealg (>= 1.0-4)
URL https://github.com/RobinHankin/weyl
BugReports https://github.com/RobinHankin/weyl/issues
RdMacros mathjaxr

R topics documented:

weyl-package ......................................................... 2
coeffs ............................................................... 3
constant ........................................................... 4
degree ............................................................... 5
derivation ........................................................... 6
dim ................................................................. 6
don-class ........................................................... 7
drop ............................................................... 8
grade ............................................................... 9
identity ........................................................... 10
Ops ............................................................... 10
print.weyl .......................................................... 12
rweyl ............................................................. 13
weyl ............................................................... 14
weyl-class .......................................................... 14
x_and_d ............................................................ 15
zero ............................................................... 15
Description


Details

The DESCRIPTION file:

Package: weyl
Type: Package
Title: The Weyl Algebra
Version: 0.0-1
Depends: spray (>= 1.0-19), methods, R (>= 3.5.0)
Authors@R: person(given=c("Robin", "K. S."), family="Hankin", role = c("aut","cre"), email="hankin.robin@gmail.com")
Maintainer: Robin K. S. Hankin <hankin.robin@gmail.com>
License: GPL (>= 2)
LazyData: yes
Suggests: knitr,rmarkdown,testthat
VignetteBuilder: knitr
Imports: mathjaxr, disordR (>= 0.0-8), freealg (>= 1.0-4)
URL: https://github.com/RobinHankin/weyl
BugReports: https://github.com/RobinHankin/weyl/issues
RdMacros: mathjaxr
Author: Robin K. S. Hankin [aut, cre] (<https://orcid.org/0000-0001-5982-0415>)

Index of help topics:

- Ops
  - Arithmetic Ops Group Methods for the Weyl algebra
- coeffs
  - Manipulate the coefficients of a weyl object
- constant
  - The constant term
- degree
  - The degree of a 'weyl' object
- derivation
  - Derivations
- dim
  - The dimension of a 'weyl' object
- dot-class
  - Class "dot"
- drop
  - Drop redundant information
- grade
  - The grade of a weyl object
- identity
  - The identity operator
- print.weyl
  - Print methods for weyl objects
- rweyl
  - Random weyl objects
- weyl
  - The algebra and weyl objects
- weyl-class
  - Class "weyl"
- weyl-package
  - The Weyl Algebra
- x_and_d
  - Generating elements for the first Weyl algebra
The zero operator

Author(s)
NA
Maintainer: Robin K. S. Hankin <hankin.robin@gmail.com>

Examples

```r
x <- rweyl(d=1)
y <- rweyl(d=1)
z <- rweyl(d=1)

is.zero(x*(y*z) - (x*y)*z) # should be TRUE
```

```
coeffs
Manipulate the coefficients of a weyl object

Description
Manipulate the coefficients of a weyl object. The coefficients are disord objects.

Usage
coeffs(S) <- value

Arguments
S               A weyl object
value           Numeric

Details
To access coefficients of a weyl object S, use spray::coeffs(S) [package idiom is coeffs(S)].
Similarly to access the index matrix use index(s).
The replacement method is package-specific; use coeffs(S) <- value.

Value
Extraction methods return a disord object (possibly dropped); replacement methods return a weyl object.

Author(s)
Robin K. S. Hankin

Examples
a <- rweyl()
coeffs(a)
coeffs(a)[coeffs(a)<3] <- 100
```
constant

The constant term

Description

The constant of a weyl object is the coefficient of the term with all zeros.

Usage

constant(x, drop = TRUE)
constant(x) <- value

Arguments

x Object of class weyl
drop Boolean with default TRUE meaning to return the value of the coefficient, and FALSE meaning to return the corresponding Weyl object
value Constant value to replace existing one

Value

Returns a numeric or weyl object

Note

The constant.weyl() function is somewhat awkward because it has to deal with the difficult case where the constant is zero and drop=FALSE.

Author(s)

Robin K. S. Hankin

Examples

a <- rweyl()+5
constant(a)
costant(a,drop=FALSE)

constant(a) <- 0
constant(a)
costant(a,drop=FALSE)

constant(a+66) == constant(a) + 66
The degree of a monomial weyl object $x^a \partial^b$ is defined as $a + b$. The degree of a general weyl object expressed as a linear combination of monomials is the maximum of the degrees of these monomials. Following Coutinho we have:

- $\deg(d_1 + d_2) \leq \max(\deg(d_1) + \deg(d_2))$
- $\deg(d_1 d_2) = \deg(d_1) + \deg(d_2)$
- $\deg(d_1 d_2 - d_2 d_1) \leq \deg(d_1) + \deg(d_2) - 2$

**Usage**

```r
deg(S)
```

**Arguments**

- `S` Object of class `weyl`

**Value**

Nonnegative integer (or $-\infty$ for the zero Weyl object)

**Note**

The degree of the zero object is conventionally $-\infty$.

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
d1 <- rweyl(n=2)
d2 <- rweyl(n=2)
deg(d1+d2) <= deg(d1) + deg(d2)
deg(d1*d2) == deg(d1) + deg(d2)
deg(d1*d2-d2*d1) <= deg(d1) + deg(d2) -2
```
### Derivations

**Description**

A derivation $D$ of an algebra $A$ is a linear operator that satisfies $D(d_1d_2) = d_1D(d_2) + D(d_1)d_2$, for every $d_1, d_2 \in A$. If a derivation is of the form $D(d) = [d, f] = df - fd$ for some fixed $f \in A$, we say that $D$ is an inner derivation.

Function `as.der()` returns a derivation with $\text{as.der}(f)(g) = fg - gf$.

**Usage**

`as.der(S)`

**Arguments**

- `S` : Weyl object

**Value**

Returns a function, a derivation

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
o <- rweyl(n=2,d=2)
f <- as.der(o)
d1 <-rweyl(n=1,d=2)
d2 <-rweyl(n=2,d=2)

f(d1*d2) == d1*f(d2) + f(d1)*d2 # should be TRUE
```

---

### dim

**Description**

The dimension of a weyl algebra is the number of variables needed; it is half the `spray::arity()`.

The dimension of a Weyl algebra generated by $\{x_1, x_2, \ldots, x_n, \partial_{x_1}, \partial_{x_2}, \ldots, \partial_{x_n}\}$ is $n$. It is the number of variables needed for the operators; it is half the `spray::arity()`.

**Usage**

```r
## S3 method for class 'weyl'
dim(x)
```
**dot-class**

**Arguments**

- `x` Object of class `weyl`

**Value**

Integer

**Note**

Empty spray objects give zero-dimensional `weyl` objects.

**Author(s)**

Robin K. S. Hankin

**Examples**

```
dim(rweyl())
```

---

**dot-class**

Class “dot”

**Description**

The dot object is defined in the `freecal` package, and imported here, so that idiom like `[x, y]` returns the commutator, that is, `xy-yx`.

**Arguments**

- `x` Object of any class
- `i, j` elements to commute
- `...` Further arguments to `dot_error()`, currently ignored

**Value**

Always returns an object of the same class as `xy`.

**Author(s)**

Robin K. S. Hankin

**Examples**

```
x <- rweyl(n=1, d=2)
y <- rweyl(n=1, d=2)
z <- rweyl(n=1, d=2)

.[x,.[y,z]] + .[y,.[z,x]] + .[z,.[x,y]] # Jacobi identity
```
Description

Coerce constant weyl objects to numeric

Usage

drop(x)

Arguments

x Weyl object

Details

If its argument is a constant weyl object, coerce to numeric.

Value

Returns either a length-one numeric vector or its argument, a weyl object

Note

Many functions in the package take `drop` as an argument which, if TRUE, means that the function returns a dropped value.

Author(s)

Robin K. S. Hankin

Examples

```r
a <- rweyl() + 67
drop(a)

drop(idweyl(9))

drop(constant(a, drop=FALSE))
```
The grade of a homogenous term of a Weyl algebra is the sum of the powers. Thus the grade of $4xy^2\partial_x^3\partial_y^4$ is $1 + 2 + 3 + 4 = 10$.

The functionality documented here closely follows the equivalent in the clifford package. Coutinho calls this the symbol map.

**Usage**

```r
grade(C, n, drop=TRUE)
g grade(C,n) <- value
grades(x)
```

**Arguments**

- `C, x` Weyl object
- `n` Integer vector specifying grades to extract
- `value` Replacement value, a numeric vector
- `drop` Boolean, with default TRUE meaning to coerce a constant operator to numeric, and FALSE meaning not to

**Details**

Function `grades()` returns an (unordered) vector specifying the grades of the constituent terms. Function `grades<-()` allows idiom such as `grade(x, 1:2) <- 7` to operate as expected [here to set all coefficients of terms with grades 1 or 2 to value 7].

Function `grade(C, n)` returns a Weyl object with just the elements of grade $g$, where $g \in n$.

The zero grade term, `grade(C, 0)`, is given more naturally by `constant(C)`.

**Value**

Integer vector or weyl object

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
a <- rweyl(30)

grades(a)
g grade(a, 1:4)
g grade(a, 5:9) <- -99
```
### identity

**The identity operator**

**Description**

The identity operator maps any function to itself.

**Usage**

```r
idweyl(d)
## S3 method for class 'weyl'
as.id(S)
is.id(S)
```

**Arguments**

- `d` Integer specifying dimensionality of the weyl object (twice the spray arity)
- `S` A weyl object

**Value**

A weyl object corresponding to the identity operator

**Note**

The identity function cannot be called “`id()`” because then R would not know whether to create a spray or a weyl object.

**Examples**

```r
idweyl(7)
a <- rweyl(d=5)
is.id(a)
is.id(1+a=a)

a == a*1
a == a*as.id(a)
```

### Ops

*Arithmetic Ops Group Methods for the Weyl algebra*

**Description**

Allows arithmetic operators to be used for spray calculations, such as addition, multiplication, division, integer powers, etc.

Idiom such as `x^2 + y*z/5` should work as expected. Operations are the same as those of the spray package except for *, which is interpreted as functional composition. A number of helper functions are documented here (which are not designed for the end-user).
Usage

```r
## S3 method for class 'weyl'
Ops(e1, e2 = NULL)
weyl_prod_helper1(a,b,c,d)
weyl_prod_helper2(a,b,c,d)
weyl_prod_helper3(a,b,c,d)
weyl_prod_univariate_onerow(S1,S2,func)
weyl_prod_univariate_nrow(S1,S2)
weyl_prod_multivariate_onerow_singlecolumn(S1,S2,column)
weyl_prod_multivariate_onerow_allcolumns(S1,S2)
weyl_prod_multivariate_nrow_allcolumns(S1,S2)
weyl_power_scalar(S,n)
```

Arguments

- `S, S1, S2, e1, e2` Objects of class `weyl`, elements of a Weyl algebra
- `a, b, c, d` Integers, see details
- `column` Column to be multiplied
- `n` Integer power (non-negative)
- `func` Function used for products

Details

All arithmetic is as for `spray` objects, apart from `*` and `^`. Here, `*` is interpreted as operator concatenation: Thus, if \( w_1 \) and \( w_2 \) are Weyl objects, then \( w_1 w_2 \) is defined as the operator that takes \( f \) to \( w_1(w_2 f) \).

Functions such as `weyl_prod_multivariate_nrow_allcolumns()` are low-level helper functions with self-explanatory names. In this context, “univariate” means the first Weyl algebra, generated by \( \{ x, \partial \} \), subject to \( x \partial - \partial x = 1 \); and “multivariate” means the algebra generated by \( \{ x_1, x_2, \ldots, x_n, \partial x_1, \partial x_2, \ldots, \partial x_n \} \).

The product is somewhat user-customisable via option `prodfunc`, which affects function `weyl_prod_univariate_onerow()`.

Currently the package offers three examples: `weyl_prod_helper1()`, `weyl_prod_helper2()`, and `weyl_prod_helper3()`. These are algebraically identical but occupy different positions on the efficiency-readability scale. The option defaults to `weyl_prod_helper3()`, which is the fastest but most opaque. The vignette provides further details, motivation, and examples.

Value

Generally, return a `weyl` object

Note

Function `weyl_prod_univariate_nrow()` is present for completeness, it is not used in the package

Author(s)

Robin K. S. Hankin
Examples

```r
x <- rweyl(n=1,d=2)
y <- rweyl(n=1,d=2)
z <- rweyl(n=2,d=2)

x*(y+z) == x*y + x*z
is.zero(x*(y*z) - (x*y)*z)
```

Description

Printing methods for weyl objects follow those for the spray package, with some additional functionality.

Usage

```r
## S3 method for class 'weyl'
print(x, ...)
```

Arguments

- `x`: A weyl object
- `...`: Further arguments, currently ignored

Details

Option polyform determines whether the object is to be printed in matrix form or polynomial form: as in the spray package, this option governs dispatch to either `print_spray_polyform()` or `print_spray_matrixform()`.

Option weylvars controls the variable names by changing the sprayvars option which is used in the spray package. If NULL (the default), then sensible values are used; either [xyz] if the dimension is three or less, or integers.

If the user sets weylvars, the print method tries to do the Right Thing (tm). If set to c("a","b","c"), for example, the generators are named c(" a"," b"," c"," da"," db"," dc") [note the spaces]. If the algebra is univariate, the names will be something like d and x. No checking is performed and if the length is not equal to the dimension, undesirable behaviour may occur. For the love of God, do not use a variable named d.

Note that, as for the spray package, this option has no algebraic significance: it only affects the print method.

Value

Returns a weyl object.

Author(s)

Robin K. S. Hankin
**rweyl**

**Random weyl objects**

**Description**

Creates random weyl objects: quick-and-dirty examples of Weyl algebra elements

**Usage**

```r
rweyl(nterms = 3, vals = seq_len(nterms), dim = 3, powers = 0:2)
```

**Arguments**

- `nterms`: Number of terms in output
- `vals`: Values of coefficients
- `dim`: Dimension of weyl object
- `powers`: Set from which to sample the entries of the index matrix

**Value**

Returns a weyl object

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
rweyl()
```

```r
rweyl(d=7)
```
The algebra and weyl objects

Description

Basic functions for weyl objects

Usage

weyl(M)

is.weyl(M)

as.weyl(val,d)

is.ok.weyl(M)

Arguments

M           A weyl or spray object
val,d       Value and dimension for weyl object

Details

Function weyl() is the formal creator method; is.weyl() tests for weyl objects and is.ok.weyl() checks for well-formed sprays. Function as.weyl() tries (but not very hard) to infer what the user intended and return the right thing.

Value

Return a weyl or a Boolean

Author(s)

Robin K. S. Hankin

Examples

weyl(spray(matrix(1:36,6,6),1:6))

as.weyl(15,d=3)

Class “weyl”

The formal S4 class for weyls.

Objects from the Class

Objects can be created by calls of the form new("weyl",...) but this is not encouraged. Use functions weyl() or as.weyl() instead.
Author(s)
Robin K. S. Hankin

Description
Variables \( x \) and \( d \) correspond to operator \( x \) and \( \partial_x \); they are provided for convenience. These elements generate the one-dimensional Weyl algebra.

Note that a similar system for multivariate Weyl algebras is not desirable. We might want to consider the Weyl algebra generated by \( \{x, y, z, \partial_x, \partial_y, \partial_z\} \) and correspondingly define \( R \) variables \( x, y, z, dx, dy, dz \). But then variable \( x \) is ambiguous: is it a member of the first Weyl algebra or the third?

Usage
\[
\text{data}(\text{\textunderscore} \text{\textunderscore} \text{\textunderscore} \text{\textunderscore} \text{\textunderscore} \text{\textunderscore} \text{\textunderscore} \text{\textunderscore} \text{x\textunderscoreand\textunderscored})
\]

Author(s)
Robin K. S. Hankin

Examples
\[
d*x-x*d
\]
\[
(1-d*x*d)*(x^2-d^3)
\]

zero

The zero operator

Description
The zero operator maps any function to the zero function (which maps anything to zero). To test for being zero, use \texttt{spray::is.zero()}; package idiom would be \texttt{is.zero()}.

Usage
\[
\text{zero}(d)
\]

Arguments
\[
d \quad \text{Integer specifying dimensionality of the weyl object (twice the spray arity)}
\]

Value
A weyl object corresponding to the zero operator (or a Boolean for \texttt{is.zero()})
Examples

```r
a <- rweyl(d=5)
is.zero(a)
is.zero(a-a)
is.zero(a*0)

a == a + zero(dim(a))
```
Index

* classes
  weyl-class, 14
* datasets
  x_and_d, 15
* package
  weyl-package, 2
    .(dot-class), 7
    [, dot, ANY, ANY-method (dot-class), 7
    [, dot, ANY, missing-method (dot-class), 7
    [, dot, matrix, matrix-method (dot-class), 7
      [, dot, missing, ANY-method (dot-class), 7
      [, dot, missing, missing-method (dot-class), 7
      [, dot-method (dot-class), 7
    as.der (derivation), 6
    as.id (identity), 10
    as.identity (identity), 10
    as.one.weyl (identity), 10
    as.weyl (weyl), 14
    coeff (coeffs), 3
    coeffs, 3
    coeffs.weyl-method (coeffs), 3
    coeff.weyl (coeffs), 3
    coeffs<-(coeffs), 3
    coeffs<-.weyl-method (coeffs), 3
    coeff.<-.weyl (coeffs), 3
    commutator (dot-class), 7
    const (constant), 4
    constant, 4
    constant<-(constant), 4
    d (x_and_d), 15
    deg (degree), 5
    degree, 5
    derivation, 6
    derivations (derivation), 6
    dim, 6
    dimension (dim), 6
    dot (dot-class), 7
    dot-class, 7
    dot_error (dot-class), 7
    drop, 8
    drop.weyl-method (drop), 8
    empty (zero), 15
    extract (dot-class), 7
    grade, 9
    grade<- (grade), 9
    grades (grade), 9
    id (identity), 10
    identity, 10
    idweyl (identity), 10
    index (coeffs), 3
    is.empty (zero), 15
    is.id (identity), 10
    is.identity (identity), 10
    is.ok.weyl (weyl), 14
    is.weyl (weyl), 14
    is.zero (zero), 15
    jacobi (dot-class), 7
    Ops, 10
    print (print.weyl), 12
    print.weyl, 12
    prodfunc (Ops), 10
    rweyl, 13
    symbol_map (grade), 9
    value (coeffs), 3
    value.weyl-method (coeffs), 3
    value.weyl (coeffs), 3
    value.<- (coeffs), 3
    values (coeffs), 3
    weyl, 14
    weyl-class, 14
    weyl-package, 2
    weyl_power_scalar (Ops), 10
    weyl_prod (Ops), 10
weyl_prod_helper1 (Ops), 10
weyl_prod_helper2 (Ops), 10
weyl_prod_helper3 (Ops), 10
weyl_prod_multivariate_nrow_allcolumns (Ops), 10
weyl_prod_multivariate_onerow_allcolumns (Ops), 10
weyl_prod_multivariate_onerow_singlecolumn (Ops), 10
weyl_prod_univariate_nrow (Ops), 10
weyl_prod_univariate_onerow (Ops), 10

x (x_and_d), 15
x_and_d, 15

zero, 15