Package ‘stokes’

May 6, 2021

Type Package
Title The Exterior Calculus
Version 1.0-8
Depends spray (>= 1.0-11)
Suggests knitr,
Deriv,
testthat,
markdown,
rmarkdown,
emulator
VignetteBuilder knitr
Imports permutations (>= 1.0-4), partitions, magrittr, methods, mathjaxr
Maintainer Robin K. S. Hankin <hankin.robin@gmail.com>
Description Provides functionality for working with tensors, alternating
tensors, wedge products, Stokes's theorem, and related concepts
from the exterior calculus. Functionality for Grassman algebra
is provided. The canonical reference would be:
License GPL-2
URL https://github.com/RobinHankin/stokes
BugReports https://github.com/RobinHankin/stokes/issues
RdMacros mathjaxr

R topics documented:

<table>
<thead>
<tr>
<th>stokes-package</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alt</td>
<td>4</td>
</tr>
<tr>
<td>as.1form</td>
<td>6</td>
</tr>
<tr>
<td>consolidate</td>
<td>7</td>
</tr>
<tr>
<td>contract</td>
<td>8</td>
</tr>
<tr>
<td>cross</td>
<td>9</td>
</tr>
<tr>
<td>hodge</td>
<td>11</td>
</tr>
<tr>
<td>inner</td>
<td>12</td>
</tr>
<tr>
<td>issmall</td>
<td>13</td>
</tr>
<tr>
<td>keep</td>
<td>13</td>
</tr>
</tbody>
</table>
The Exterior Calculus

Description

Provides functionality for working with tensors, alternating tensors, wedge products, Stokes’s theorem, and related concepts from the exterior calculus. Functionality for Grassman algebra is provided. The canonical reference would be: M. Spivak (1965, ISBN:0-8053-9021-9) "Calculus on Manifolds".

Details

The DESCRIPTION file:

<table>
<thead>
<tr>
<th>Package</th>
<th>stokes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Title</td>
<td>The Exterior Calculus</td>
</tr>
<tr>
<td>Version</td>
<td>1.0-8</td>
</tr>
<tr>
<td>Depends</td>
<td>spray (&gt;= 1.0-11)</td>
</tr>
<tr>
<td>Suggests</td>
<td>knitr, Deriv, testthat, markdown, rmarkdown, emulator</td>
</tr>
<tr>
<td>VignetteBuilder</td>
<td>knitr</td>
</tr>
<tr>
<td>Imports</td>
<td>permutations (&gt;= 1.0-4), partitions, magrittr, methods, mathjaxr</td>
</tr>
<tr>
<td>Authors@R</td>
<td>person( given=c(&quot;Robin&quot;, &quot;K. S.&quot;), family=&quot;Hankin&quot;, role = c(&quot;aut&quot;,&quot;cre&quot;), email=&quot;<a href="mailto:hankin.robin@gmail.com">hankin.robin@gmail.com</a>&quot;)</td>
</tr>
<tr>
<td>Maintainer</td>
<td>Robin K. S. Hankin <a href="mailto:hankin.robin@gmail.com">hankin.robin@gmail.com</a></td>
</tr>
<tr>
<td>Description</td>
<td>Provides functionality for working with tensors, alternating tensors, wedge products, Stokes’s theorem</td>
</tr>
<tr>
<td>License</td>
<td>GPL-2</td>
</tr>
<tr>
<td>URL</td>
<td><a href="https://github.com/RobinHankin/stokes">https://github.com/RobinHankin/stokes</a></td>
</tr>
<tr>
<td>BugReports</td>
<td><a href="https://github.com/RobinHankin/stokes/issues">https://github.com/RobinHankin/stokes/issues</a></td>
</tr>
<tr>
<td>RdMacros</td>
<td>mathjaxr</td>
</tr>
<tr>
<td>Author</td>
<td>Robin K. S. Hankin [aut, cre] (<a href="https://orcid.org/0000-0001-5982-0415">https://orcid.org/0000-0001-5982-0415</a>)</td>
</tr>
</tbody>
</table>

Index of help topics:

<table>
<thead>
<tr>
<th>Alt</th>
<th>Alternating multilinear forms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ops.kform</td>
<td>Arithmetic Ops Group Methods for 'kform' and 'ktensor' objects</td>
</tr>
</tbody>
</table>
stokes-package

as.1form Coerce vectors to 1-forms
consolidate Various low-level helper functions
contract Contractions of k-forms
cross Cross products of k-tensors
hodge Hodge star operator
inner Inner product operator
issmall Is a form zero to within numerical precision?
keep Keep or drop variables
kform k-forms
ktensor k-tensors
print.stokes Print methods for k-tensors and k-forms
rform Random kforms and ktensors
scalar Lose attributes
stokes-package The Exterior Calculus
symbolic Symbolic form
transform Linear transforms of k-forms
volume The volume element
wedge Wedge products
zap Zap small values in k-forms and k-tensors
zeroform Zero tensors and zero forms

Generally in the package, arguments that are k-forms are denoted \( K \), k-tensors by \( U \), and spray objects by \( S \). Multilinear maps (which may be either k-forms or k-tensors) are denoted by \( M \).

Author(s)

NA

Maintainer: Robin K. S. Hankin <hankin.robin@gmail.com>

References


See Also

spray

Examples

```r
## Some k-tensors:
U1 <- as.ktensor(matrix(1:15,5,3))
U2 <- as.ktensor(cbind(1:3,2:4),1:3)

## Coerce a tensor to functional form, here mapping \( V^3 \) \( \rightarrow \) \( R \) (here \( V=R^{15} \)):
as.function(U1)(matrix(rnorm(45),15,3))

## Tensor cross-product is cross() or %X%:
U1 %X% U2

## A k-form is an alternating k-tensor:
K1 <- as.kform(cbind(1:5,2:6),rnorm(5))
```
K2 <- kform_general(3:6,2,1:6)
K3 <- rform(9,3,9,runif(9))

## The distributive law is true
(K1 + K2) %*% K3 == K1 %*% K3 + K2 %*% K3 # TRUE to numerical precision

## Wedge product is associative (non-trivial):
K1 %*% K2 %*% K3
K1 %*% (K2 %*% K3)

## k-forms can be coerced to a function and wedge product:
f <- as.function(K1 %*% K2 %*% K3)

## E is a random point in V^k:
E <- matrix(rnorm(63),9,7)

## f() is alternating:
f(E)
f(E[,7:1])

## The package blurs the distinction between symbolic and numeric computing:
dx <- as.kform(1)
dy <- as.kform(2)
dz <- as.kform(3)
dx %*% dy %*% dz
K3 %*% dx %*% dy %*% dz

---

**Alt**

*Alternating multilinear forms*

**Description**

Converts a k-tensor to alternating form

**Usage**

Alt(S,give_kform)

**Arguments**

S

A multilinear form, an object of class ktensor
give_kform

Boolean, with default FALSE meaning to return an alternating k-tensor [that is, an object of class ktensor that happens to be alternating] and TRUE meaning to return a k-form [that is, an object of class kform]
Details

Given a k-tensor $T$, we have

$$\text{Alt}(T) (v_1, \ldots, v_k) = \frac{1}{k!} \sum_{\sigma \in S_k} \text{sgn}(\sigma) \cdot T(v_{\sigma(1)}, \ldots, v_{\sigma(k)})$$

Thus for example if $k = 3$:

$$\text{Alt}(T) (v_1, v_2, v_3) = \frac{1}{6} \begin{pmatrix} +T(v_1, v_2, v_3) & -T(v_1, v_3, v_2) \\ -T(v_2, v_1, v_3) & +T(v_2, v_3, v_1) \\ +T(v_3, v_1, v_2) & -T(v_3, v_2, v_1) \end{pmatrix}$$

and it is reasonably easy to see that $\text{Alt}(T)$ is alternating, in the sense that

$$\text{Alt}(T) (v_1, \ldots, v_i, \ldots, v_j, \ldots, v_k) = -\text{Alt}(T) (v_1, \ldots, v_j, \ldots, v_i, \ldots, v_k)$$

Function $\text{Alt}()$ is intended to take and return an object of class $\text{ktensor}$; but if given a $\text{kform}$ object, it just returns its argument unchanged.

See also the 'Alt' vignette which contains more details and examples.

Value

Returns an alternating $k$-tensor. To work with $k$-forms, which are a much more efficient representation of alternating tensors, use $\text{as.kform}()$.

Author(s)

Robin K. S. Hankin

See Also

$kform$

Examples

```r
S <- as.ktensor(expand.grid(1:3,1:3),rnorm(9))
S
Alt(S)
issmall(Alt(S) - Alt(Alt(S))) # should be TRUE

a <- rtensor()
a
V <- matrix(rnorm(21),ncol=3)
c(as.function(Alt(a))(V), as.function(Alt(a,give_kform=TRUE))(V)) # should match
```
Coerce vectors to 1-forms

Description

Given a vector, return the corresponding 1-form; the exterior derivative of a 0-form (that is, a scalar function). Function \texttt{grad()} is a synonym.

Usage

\begin{align*}
\texttt{as.1form(v)} \\
\texttt{grad(v)}
\end{align*}

Arguments

\begin{itemize}
\item \texttt{v} \quad A vector with element \(i\) being \(\partial f/\partial x_i\)
\end{itemize}

Details

The exterior derivative of a \(k\)-form \(\phi\) is a \((k+1)\)-form \(d\phi\) given by

\[
d\phi \left( P_x (v_i, \ldots, v_{k+1}) \right) = \lim_{h \to 0} \frac{1}{h^{k+1}} \int_{\partial P_x (hv_1, \ldots, hv_{k+1})} \phi
\]

We can use the facts that

\[
d (f dx_{i_1} \wedge \cdots \wedge dx_{i_k}) = df \wedge dx_{i_1} \wedge \cdots \wedge dx_{i_k}
\]

and

\[
d f = \sum_{j=1}^{n} (D_j f) \ dx_j
\]

to calculate differentials of general \(k\)-forms. Specifically, if

\[
\phi = \sum_{1 \leq i_1 < \cdots < i_k \leq n} a_{i_1 \ldots i_k} dx_{i_1} \wedge \cdots \wedge dx_{i_k}
\]

then

\[
d \phi = \sum_{1 \leq i_1 < \cdots < i_k \leq n} \left[ \sum_{j=1}^{n} D_j a_{i_1 \ldots i_k} dx_j \right] \wedge dx_{i_1} \wedge \cdots \wedge dx_{i_k}.
\]

The entry in square brackets is given by \texttt{grad()}. See the examples for appropriate R idiom.

Value

A one-form

Author(s)

Robin K. S. Hankin
**consolidate**

**See Also**

kform

**Examples**

```r
as.1form(1:9)  # note ordering of terms

as.1form(rnorm(20))

grad(c(4,7)) %*% grad(1:4)
```

---

<table>
<thead>
<tr>
<th>consolidate</th>
<th>Various low-level helper functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description**

Various low-level helper functions used in `Alt()` and `kform()`

**Usage**

```r
consolidate(S)
kill_trivial_rows(S)
include_perms(S)
kform_to_ktensor(S)
```

**Arguments**

- `S` Object of class `spray`

**Details**

Low-level helper functions.

- Function `consolidate()` takes a spray object, and combines any rows that are identical up to a permutation, respecting the sign of the permutation
- Function `kill_trivial_rows()` takes a spray object and deletes any rows with a repeated entry (which have `k`-forms identically zero)
- Function `include_perms()` replaces each row of a spray object with all its permutations, respecting the sign of the permutation
- Function `kform_to_ktensor()` coerces a `k`-form to a `k`-tensor

**Value**

The functions documented here all return a spray object.
Author(s)

Robin K. S. Hankin

See Also

ktensor, kform, Alt

Examples

```r
S <- spray(matrix(c(1,1,2,2,1,3,3,1,3,5),nrow=2,byrow=TRUE),1:5)

kill_trivial_rows(S)  # (rows 1 and 3 killed, repeated entries)
consolidate(S)        # (merges rows 2 and 4)
include_perms(S)      # returns a spray object, not alternating tensor.
```

\[
\text{contract} \quad \text{Contractions of } k\text{-forms}
\]

Description

A contraction is a natural linear map from \( k \)-forms to \( k-1 \)-forms.

Usage

```
contract(K,v,lose=TRUE)
contract_elementary(o,v)
```

Arguments

- \( K \): A \( k \)-form
- \( o \): Integer-valued vector corresponding to one row of an index matrix
- \( \text{lose} \): Boolean, with default \( \text{TRUE} \) meaning to coerce a \( 0 \)-form to a scalar and \( \text{FALSE} \) meaning to return the formal \( 0 \)-form
- \( v \): A vector; in function `contract()`, if a matrix, interpret each column as a vector to contract with

Details

Given a \( k \)-form \( \phi \) and a vector \( v \), the \textit{contraction} \( \phi_v \) of \( \phi \) and \( v \) is a \( k-1 \)-form with

\[
\phi_v \left( v^1, \ldots, v^{k-1} \right) = \phi \left( v, v^1, \ldots, v^{k-1} \right)
\]

if \( k > 1 \); we specify \( \phi_v = \phi(v) \) if \( k = 1 \).

Function `contract_elementary()` is a low-level helper function that translates elementary \( k \)-forms with coefficient 1 (in the form of an integer vector corresponding to one row of an index matrix) into its contraction with \( v \).

Value

Returns an object of class `kform`. 
cross

Author(s)
Robin K. S. Hankin

References

See Also
wedge, lose

Examples

```r
cross(as.kform(1:5),1:8)
cross(as.kform(1),3)  # 0-form

## Now some verification:
o <- kform(spray(t(replicate(2, sample(9,5))), runif(2)))
V <- matrix(rnorm(45),ncol=5)
jj <- c(
as.function(o)(V),
as.function(contract(o,V[,1,drop=TRUE]))(V[,1:2]), # scalar
as.function(contract(o,V[,2:3]))(V[,-(1:2),drop=FALSE]),
as.function(contract(o,V[,3:4]))(V[,-(1:3),drop=FALSE]),
as.function(contract(o,V[,4:5],lose=FALSE))(V[,-(1:4),drop=FALSE])
)
max(jj) - min(jj) # zero to numerical precision
```

cross

Cross products of k-tensors

Description
Cross products of k-tensors

Usage

```r
cross(U, ...)
cross2(U1,U2)
```

Arguments

U,U1,U2 Object of class ktensor

... Further arguments, currently ignored
Details

Given a $k$-tensor $S$ and an $l$-tensor $T$, we can form the cross product $S \otimes T$, defined as

$$S \otimes T(v_1, \ldots, v_k, v_{k+1}, \ldots, v_{k+l}) = S(v_1, \ldots, v_k) \cdot T(v_{k+1}, \ldots, v_{k+l}).$$

Package idiom for this includes `cross(S, T)` and `S %X% T`; note that the cross product is not commutative. Function `cross()` can take any number of arguments (the result is well-defined because the cross product is associative); it uses `cross2()` as a low-level helper function.

Value

The functions documented here all return a spray object.

Note

The binary form `%X%` uses uppercase X to avoid clashing with `%x%` which is the Kronecker product in base R.

Author(s)

Robin K. S. Hankin

References

Spivak 1961

See Also

`ktensor`

Examples

```r
M <- cbind(1:4, 2:5)
U1 <- as.ktensor(M, rnorm(4))
U2 <- as.ktensor(t(M), 1:2)

cross(U1, U2)
cross(U2, U1) # not the same!

U1 %X% U2 - U2 %X% U1
```
**hodge**

**Hodge star operator**

**Description**
Given a $k$-form, return its Hodge dual

**Usage**

```
hodge(K, n=max(index(K)), g=rep(1,n), lose=TRUE)
```

**Arguments**

- **K**: Object of class `kform`
- **n**: Dimensionality of space, defaulting to the largest element of the index
- **g**: Diagonal of the metric tensor, defaulting to the standard metric
- **lose**: Boolean, with default `TRUE` meaning to coerce to a scalar if appropriate

**Value**
Given a $k$-form, in an $n$-dimensional space, returns a $(n-k)$-form.

**Author(s)**
Robin K. S. Hankin

**See Also**

- `wedge`

**Examples**

```
hodge(rform())
hodge(kform_general(4,2),g=c(-1,1,1,1))
```

```r
## Some edge-cases:
hodge(zero(5),9)
hodge(volume(5))
hodge(volume(5),lose=TRUE)
hodge(scalar(7),n=9)
```
inner product operator

Description
The inner product

Usage
inner(M)

Arguments
M square matrix

Details
The inner product of two vectors \( x \) and \( y \) is usually written \( \langle x, y \rangle \) or \( x \cdot y \), but the most general form would be \( x^T M y \) where \( M \) is a matrix. Noting that inner products are multilinear, that is \( \langle x, ay + bz \rangle = a \langle x, y \rangle + b \langle x, z \rangle \) and \( \langle ax + by, z \rangle = a \langle x, z \rangle + b \langle y, z \rangle \), we see that the inner product is indeed a multilinear map, that is, a tensor.

Given a square matrix \( M \), function \( \text{inner}(M) \) returns the 2-form that maps \( x, y \) to \( x^T M y \).

See also the \texttt{inner} vignette which contains more details and examples.

Value
Returns a \( k \)-tensor, an inner product

Author(s)
Robin K. S. Hankin

See Also
kform

Examples
inner(diag(7))
inner(matrix(1:9,3,3))

### Compare the following two:
Alt(inner(matrix(1:9,3,3))) # An alternating k tensor
as.kform(inner(matrix(1:9,3,3))) # Same thing coerced to a kform

f <- as.function(inner(diag(7)))
X <- matrix(rnorm(14),ncol=2) # random element of \( (R^7)^2 \)
f(X) - sum(X[,1]*X[,2]) # zero to numerical precision

### verify positive-definiteness:
g <- as.function(inner(crossprod(matrix(rnorm(56),8,7))))
stopifnot(g(kronecker(rnorm(7),t(c(1,1))))>0)
### issmall

**Is a form zero to within numerical precision?**

**Description**

Given a \( k \)-form, return TRUE if it is “small”

**Usage**

\[
\text{issmall}(M, \text{tol}=1e^{-8})
\]

**Arguments**

- **M**: Object of class `kform` or `ktensor`
- **tol**: Small tolerance, defaulting to \( 1e^{-8} \)

**Value**

Returns a logical

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
o <- kform_general(4,2,runif(6))
M <- matrix(rnorm(36),6,6)

discrepancy <- o - transform(transform(o,M),solve(M))

issmall(discrepancy) # should be TRUE
is.zero(discrepancy) # might be FALSE
```

### keep

**Keep or drop variables**

**Description**

Keep or drop variables

**Usage**

```r
\text{keep}(K, \text{yes})
\text{discard}(K, \text{no})
```

**Arguments**

- **K**: Object of class `kform`
- **yes, no**: Specification of dimensions to either keep (yes) or discard (no), coerced to a free object
Details

Function `keep(omega, yes)` keeps the terms specified and `discard(omega, no)` discards the terms specified. It is not clear to me what these functions mean from a mathematical perspective.

Value

The functions documented here all return a `kform` object.

Author(s)

Robin K. S. Hankin

See Also

`lose`

Examples

```r
keep(kform_general(7,3),1:4)  # keeps only terms with dimensions 1-4
discard(kform_general(7,3),1)  # loses any term with a "1" in the index
```

<table>
<thead>
<tr>
<th>kform</th>
<th>k-forms</th>
</tr>
</thead>
</table>

Description

Functionality for dealing with $k$-forms

Usage

```r
kform(S)
as.kform(M,coeffs,lose=TRUE)
kform_basis(n, k)
kform_general(W,k,coeffs,lose=TRUE)
is.kform(x)
## S3 method for class 'kform'
as.function(x,...)
```

Arguments

- `n`  
  Dimension of the vector space $V = R^n$
- `k`  
  A $k$-form maps $V^k$ to $R$
- `W`  
  Integer vector of dimensions
- `M,coeffs`  
  Index matrix and coefficients for a $k$-form
- `S`  
  Object of class `spray`
- `lose`  
  Boolean, with default `TRUE` meaning to coerce a 0-form to a scalar and `FALSE` meaning to return the formal 0-form
- `x`  
  Object of class `kform`
- `...`  
  Further arguments, currently ignored
**Details**

A $k$-form is an alternating $k$-tensor.

Recall that a $k$-tensor is a multilinear map from $V^k$ to the reals, where $V = \mathbb{R}^n$ is a vector space. A multilinear $k$-tensor $T$ is *alternating* if it satisfies

$$T(v_1, \ldots, v_i, \ldots, v_j, \ldots, v_k) = T(v_1, \ldots, v_j, \ldots, v_i, \ldots, v_k)$$

Function `kform_basis()` is a low-level helper function that returns a matrix whose rows constitute a basis for the vector space $\Lambda^k(\mathbb{R}^n)$ of $k$-tensors:

$$\phi = \sum_{1 \leq i_1 < \cdots < i_k \leq n} \frac{a_{i_1 \ldots i_k}}{i_1 \ldots i_k} dx_{i_1} \wedge \cdots \wedge dx_{i_k}$$

and indeed we have:

$$a_{i_1 \ldots i_k} = \phi(e_{i_1}, \ldots, e_{i_k})$$

where $e_j, 1 \leq j \leq k$ is a basis for $V$.

In the `stokes` package, $k$-forms are represented as sparse arrays (`spray` objects), but with a class of `c("kform", "spray")`. The constructor function (`kform()`) ensures that rows of the index matrix are strictly nonnegative integers, have no repeated entries, and are strictly increasing.

**Value**

All functions documented here return a `kform` object except `as.function.kform()`, which returns a function.

**Note**

Hubbard and Hubbard use the term “$k$-form”, but Spivak does not.

**Author(s)**

Robin K. S. Hankin

**References**

Hubbard and Hubbard; Spivak

**See Also**

`ktensor`, `lose`

**Examples**

```r
as.kform(cbind(1:5,2:6), rnorm(5))
kform_general(1:4, 2, coeffs=1:6)  # used in electromagnetism
K1 <- as.kform(cbind(1:5,2:6), rnorm(5))
K2 <- kform_general(5:8,2,1:6)
wedge(K1,K2)
```
f <- as.function(wedge(K1,K2))
E <- matrix(rnorm(32),8,4)
f(E) + f(E[,c(1,3,2,4)])  # should be zero

---

**ktensor**  

**k-tensors**

---

**Description**

Functionality for k-tensors

**Usage**

```r
ktensor(S)
as.ktensor(M,coeffs)
is.ktensor(x)
```  

**Arguments**

- `M,coeffs` Matrix of indices and coefficients, as in `spray(M,coeffs)`
- `S` Object of class `spray`
- `x` Object of class `ktensor`
- `...` Further arguments, currently ignored

**Details**

A k-tensor object S is a map from $V^k$ to the reals $R$, where $V$ is a vector space (here $R^n$) that satisfies multilinearity:

$$S(v_1, \ldots, av_i, \ldots, v_k) = a \cdot S(v_1, \ldots, v_i, \ldots, v_k)$$

and

$$S(v_1, \ldots, v_i + v'_i, \ldots, v_k) = S(v_1, \ldots, v_i, \ldots, x_v) + S(v_1, \ldots, v'_i, \ldots, v_k).$$

Note that this is not equivalent to linearity over $V^{nk}$ (see examples).

In the **stokes** package, k-tensors are represented as sparse arrays (spray objects), but with a class of c("ktensor","spray"). This is a natural and efficient representation for tensors that takes advantage of sparsity using `spray` package features.

Function `as.ktensor()` will coerce a k-form to a k-tensor via `kform_to_ktensor()`.

**Value**

All functions documented here return a `ktensor` object except `as.function.ktensor()`, which returns a function.
Author(s)
Robin K. S. Hankin

References
Spivak 1961

See Also

cross, kform, wedge

Examples

ktensor(rspray(4, powers=1:4))
as.ktensor(cbind(1:4,2:5,3:6),1:4)

## Test multilinearity:
k <- 4
n <- 5
u <- 3

## Define a randomish k-tensor:
S <- ktensor(spray(matrix(1+sample(u*k)%*%n,u,k),seq_len(u)))

## And a random point in V^k:
E <- matrix(rnorm(n*k),n,k)

E1 <- E2 <- E3 <- E

x1 <- rnorm(n)
x2 <- rnorm(n)
r1 <- rnorm(1)
r2 <- rnorm(1)

# change one column:
E1[,2] <- x1
E2[,2] <- x2
E3[,2] <- r1*x1 + r2*x2

f <- as.function(S)

r1*f(E1) + r2*f(E2) - f(E3) # should be small

## Note that multilinearity is different from linearity:
r1*f(E1) + r2*f(E2) - f(r1*E1 + r2*E2) # not small!
Description

Allows arithmetic operators to be used for $k$-forms and $k$-tensors such as addition, multiplication, etc, where defined.

Usage

```r
## S3 method for class 'kform'
Ops(e1, e2 = NULL)
## S3 method for class 'ktensor'
Ops(e1, e2 = NULL)
```

Arguments

- `e1`, `e2` Objects of class `kform` or `ktensor`

Details

The functions `Ops.kform()` and `Ops.ktensor()` pass unary and binary arithmetic operators ("+", "-", ",", and "/") to the appropriate specialist function by coercing to spray objects.

For wedge products of $k$-forms, use `wedge()` or `^%`; and for cross products of $k$-tensors, use `cross()` or `%X`.

Value

All functions documented here return an object of class `kform` or `ktensor`.

Note

A plain asterisk, "\(*\), given two kforms or two ktensors, will return the wedge product or the cross product respectively, on the grounds that the idiom has only one natural interpretation. But its use is discouraged.

Author(s)

Robin K. S. Hankin

Examples

```r
## dx_1 ^ dx_2 + 6dx_5 ^ dx_6:
as.kform(1) ^%% as.kform(2) + 6*as.kform(5) ^%% as.kform(6)

k1 <- kform_general(4,2,rnorm(6))
k2 <- kform_general(4,2,rnorm(6))

E <- matrix(rnorm(8),4,2)
as.function(k1+k2)(E)

as.function(2*k1+3*k2)(E)-(2*as.function(k1)(E) + 3*as.function(k1)(E))
## should be small
```
Print methods for objects with options for printing in matrix form or multivariate polynomial form

## S3 method for class 'kform'
print(x, ...)

## S3 method for class 'ktensor'
print(x, ...)

Arguments

x  

k-form or k-tensor

...  

Further arguments (currently ignored)

Details

The print method is designed to tell the user that an object is a tensor or a k-form. It prints a message to this effect (with special dispensation for zero tensors), then calls the spray print method.

Value

Returns its argument invisibly.

Note

The print method asserts that its argument is a map from \( R^n \) to \( R \), where \( n \) is the largest element in the index matrix. However, such a map naturally furnishes a map from \( R^m \) to \( R \) provided that \( m \geq n \) via the natural projection from \( R^n \) to \( R^m \). Formally this would be \( (x_1, \ldots, x_n) \mapsto (x_1, \ldots, x_n, 0, \ldots, 0) \in R^m \). In the case of the zero k-form or k-tensor, “\( n \)” is to be interpreted as “any \( n \geq 0 \).”

Author(s)

Robin K. S. Hankin

Examples

rform()
rtensor()

## spray print options work:
options(polyform = TRUE)
rtensor()

## reset to default
options(polyform = FALSE)
Random k-form objects and k-tensors, intended as quick “get you going” examples

Usage

```r
rform(terms=9,k=3,n=7,coeffs)
rtensor(terms=9,k=3,n=7,coeffs)
```

Arguments

- `terms` Number of distinct terms
- `k, n` A k-form maps $V^k$ to $R$, where $V = R^n$
- `coeffs` The coefficients of the form; if missing use `seq_len(terms)`

Details

What you see is what you get, basically.
Note that argument `terms` is an upper bound, as the index matrix might contain repeats which are combined.

Value

All functions documented here return an object of class kform or ktensor.

Author(s)

Robin K. S. Hankin

Examples

```r
rform()
rform() %% rform()

rtensor() %X% rtensor()

dx <- as.kform(1)
dy <- as.kform(2)
rform() %% dx
rform() %% dx %% dy
```
Description

 Scalars: 0-forms and 0-tensors

Usage

 scalar(s, lose=FALSE)  
is.scalar(M)  
`0form'(s, lose=FALSE)  
## S3 method for class 'kform'  
lose(M)  
## S3 method for class 'ktensor'  
lose(M)

Arguments

 s A scalar value; a number  
 M Object of class ktensor or kform  
 lose In function scalar(). Boolean with TRUE meaning to return a normal scalar, and default FALSE meaning to return a formal 0-form or 0-tensor

Details

 A k-tensor (including k-forms) maps k vectors to a scalar. If k = 0, then a 0-tensor maps no vectors to a scalar, that is, mapping nothing at all to a scalar, or what normal people would call a plain old scalar. Such forms are created by a couple of constructions in the package, specifically scalar(), kform_general(1,0) and contract(). These functions take a lose argument that behaves much like the drop argument in base extraction.

 Function lose() takes an object of class ktensor or kform and, if of arity zero, returns the coefficient.

 Note that function kform() always returns a kform object, it never loses attributes.

 A 0-form is not the same thing as a zero tensor. A 0-form maps V^0 to the reals; a scalar. A zero tensor maps V^k to zero.

Value

 The functions documented here return an object of class kform or ktensor, except for is.scalar(), which returns a Boolean.

Author(s)

 Robin K. S. Hankin

See Also

 zeroform, lose
symbolic

Examples

```r
o <- scalar(5)
lose(o)
kform_general(1,0)
kform_general(1,0,lose=FALSE)
```

symbolic  Symbolic form

Description

Prints $k$-tensor and $k$-form objects in symbolic form

Usage

```r
as.symbolic(M,symbols=letters,d="")
```

Arguments

- **M**: Object of class `kform` or `ktensor`; a map from $V^k$ to $R$, where $V = R^n$
- **symbols**: A character vector giving the names of the symbols
- **d**: String specifying the appearance of the differential operator

Value

Returns a noquote character string.

Author(s)

Robin K. S. Hankin

Examples

```r
as.symbolic(rtensor())
as.symbolic(rform())
as.symbolic(kform_general(3,2,1:3),d="d",symbols=letters[23:26])
```
Linear transforms of k-forms

Description

Given a k-form, express it in terms of linear combinations of the \( dx_i \)

Usage

\[
\text{transform}(K, M) \\
\text{stretch}(K, d)
\]

Arguments

- \( K \): Object of class kform
- \( M \): Matrix of transformation
- \( d \): Numeric vector representing the diagonal elements of a diagonal matrix

Details

Suppose we are given a two-form

\[
\omega = \sum_{i<j} a_{ij} dx_i \wedge dx_j
\]

and relationships

\[
dx_i = \sum_r M_{ir} dy_r
\]

then we would have

\[
\omega = \sum_{i<j} a_{ij} \left( \sum_r M_{ir} dy_r \right) \wedge \left( \sum_r M_{jr} dy_r \right).
\]

The general situation would be a k-form where we would have

\[
\omega = \sum_{i_1 < \cdots < i_k} a_{i_1 \cdots i_k} dx_{i_1} \wedge \cdots \wedge dx_{i_k}
\]

giving

\[
\omega = \sum_{i_1 < \cdots < i_k} \left[ a_{i_1 \cdots i_k} \left( \sum_r M_{i_1 r} dy_r \right) \wedge \cdots \wedge \left( \sum_r M_{i_k r} dy_r \right) \right].
\]

The \text{transform()} function does all this but it is slow. I am not 100% sure that there isn’t a much more efficient way to do such a transformation. There are a few tests in \text{tests/testthat} and a discussion in the \text{stokes} vignette.

Function \text{stretch()} carries out the same operation but for \( M \) a diagonal matrix. It is much faster than \text{transform()}. 
The functions documented here return an object of class kform.

Author(s)
Robin K. S. Hankin

References

See Also
wedge

Examples

```r
# Example in the text:
K <- as.kform(matrix(c(1,1,2,3),2,2),c(1,5))
M <- matrix(1:9,3,3)
transform(K,M)

# Demonstrate that the result can be complicated:
M <- matrix(rnorm(25),5,5)
transform(as.kform(1:2),M)

# Numerical verification:
o <- rform(terms=2,n=5)
o2 <- transform(transform(o,M),solve(M))
max(abs(value(o-o2))) # zero to numerical precision

# Following should be zero:
transform(as.kform(1),M)-as.kform(matrix(1:5),c(crossprod(M,c(c(1,rep(0,4))))))

# Following should be TRUE:
issmall(transform(o,crossprod(matrix(rnorm(10),2,5))))

# Some stretch() use-cases:
p <- rform()
p
stretch(p,seq_len(7))
stretch(p,c(1,0,1,1,1)) # kills dimension 2
```

---

### volume

**The volume element**

Description

The volume element in \( n \) dimensions
volume

Usage

volume(n)

is.volume(K)

Arguments

n  Dimension of the space
K  Object of class kform

Details

Spivak phrases it well (theorem 4.6, page 82):

If $V$ has dimension $n$, it follows that $\Lambda^n(V)$ has dimension 1. Thus all alternating $n$-tensors on $V$

are multiples of any non-zero one. Since the determinant is an example of such a member of $\Lambda^n(V)$
it is not surprising to find it in the following theorem:

Let $v_1, \ldots, v_n$ be a basis for $V$ and let $\omega \in \Lambda^n(V)$. If $w_i = \sum_{j=1}^{n} a_{ij} v_j$ then

$$\omega(w_1, \ldots, w_n) = \det (a_{ij}) \cdot \omega(v_1, \ldots v_n)$$

(see the examples for numerical verification of this).

Neither the zero $k$-form, nor scalars, are considered to be a volume element.

Value

Function volume() returns an object of class kform; function is.volume() returns a Boolean.

Author(s)

Robin K. S. Hankin

References

Spivak

See Also

zeroform, as.1form

Examples

as.kform(1) %*% as.kform(2) %*% as.kform(3) == volume(3)  # should be TRUE

o <- volume(5)
M <- matrix(runif(25),5,5)
det(M) - as.function(o)(M)  # should be zero
wedge products

Description

Wedge products of \( k \)-forms

Usage

\[
\text{wedge2}(K1,K2) \\
\text{wedge}(x, \ldots)
\]

Arguments

\( K1,K2,x,\ldots \) \( k \)-forms

Details

Wedge product of \( k \)-forms.

Value

The functions documented here return an object of class \texttt{kform}.

Note

In general use, use \texttt{wedge()} or \%\%\%. Function \texttt{wedge()} uses low-level helper function \texttt{wedge2()}, which takes only two arguments.

Author(s)

Robin K. S. Hankin

Examples

\[
k1 <- \text{as.kform}(\text{cbind}(1:5,2:6),1:5) \\
k2 <- \text{as.kform}(\text{cbind}(5:7,6:8,7:9),1:3) \\
k3 <- \text{Kform\textunderscore general}(1:6,2) \\
\text{a1} <- \text{wedge2}(k1,\text{wedge2}(k2,k3)) \\
\text{a2} <- \text{wedge2}(\text{wedge2}(k1,k2),k3)
\]

\texttt{is.zero(a1-a2)} \# NB terms of \texttt{a1, a2} in a different order!

\# This is why \texttt{wedge(k1,k2,k3)} is well-defined. Can also use \%\%\%:
\texttt{k1 \%\%\% k2 \%\%\% k3}
zap

Zap small values in $k$-forms and $k$-tensors

Description

Equivalent to \texttt{zapsmall()}

Usage

\begin{verbatim}
zap(X)
## S3 method for class 'kform'
zap(X)
## S3 method for class 'ktensor'
zap(X)
\end{verbatim}

Arguments

\begin{itemize}
\item \textit{X} \hspace{1cm} Tensor or $k$-form to be zapped
\end{itemize}

Details

Given an object of class \texttt{ktensor} or \texttt{kform}, coefficients close to zero are ‘zapped’, i.e., replaced by ‘0’, using \texttt{base::zapsmall()}. Note, \texttt{zap()} actually changes the numeric value, it is not just a print method.

Value

Returns an object of the same class

Author(s)

Robin K. S. Hankin

Examples

\begin{verbatim}
S <- rform(7)
S == zap(S)
\end{verbatim}

\begin{verbatim}
zeroform(n)
zerotensor(n)
\end{verbatim}

zero

Zero tensors and zero forms

Description

Correct idiom for generating zero $k$-tensors and $k$-forms

Usage

\begin{verbatim}
zeroform(n)
zerotensor(n)
\end{verbatim}
Arguments

n Arity of the $k$-form or $k$-tensor

Value

Returns an object of class kform or ktensor.

Note

Idiom such as as.ktensor(rep(1, n), 0) and as.kform(rep(1, 5), 0) and indeed as.kform(1:5, 0) is incorrect as the arity of the tensor is lost.

A 0-form is not the same thing as a zero tensor. A 0-form maps $V^0$ to the reals; a scalar. A zero tensor maps $V^k$ to zero.

Author(s)

Robin K. S. Hankin

See Also

scalar

Examples

as.ktensor(1+diag(5)) + zerotensor(5)
as.kform(matrix(1:6,2,3)) + zeroform(3)

## Following idiom is plausible but fails because as.ktensor(coeffs=0)
## and as.kform(coeffs=0) do not retain arity:

## Not run:
as.ktensor(1+diag(5)) + as.ktensor(rep(1,5),0)  # fails
asz.kform(matrix(1:6,2,3)) + as.kform(1:3,0)  # also fails

## End(Not run)
Index

* package
  stokes-package, 2

* symbolmath
  Ops.kform, 17
  print.stokes, 19
  %×% (cross), 9
  %∧% (wedge), 26
  0form (scalar), 21
  Alt, 4, 8
  as.1form, 6, 25
  as.function.kform (kform), 14
  as.function.ktensor (ktensor), 16
  as.kform (kform), 14
  as.ktensor (ktensor), 16
  as.symbolic (symbolic), 22
  consolidate, 7
  contract, 8
  contract_elementary (contract), 8
  cross, 9, 17
  cross2 (cross), 9
  discard (keep), 13
  drop (scalar), 21
  drop.free (keep), 13
  general_kform (kform), 14
  grad (as.1form), 6
  Hodge (hodge), 11
  hodge, 11
  include_perms (consolidate), 7
  inner, 12
  inner_product (inner), 12
  is.form (kform), 14
  is.kform (kform), 14
  is.ktensor (ktensor), 16
  is.scalar (scalar), 21
  is.tensor (ktensor), 16
  is.volume (volume), 24
  issmall, 13
  keep, 13
  kform, 5, 7, 8, 12, 14, 17
  kform_basis (kform), 14
  kform_general (kform), 14
  kform_to_ktensor (consolidate), 7
  kill_trivial_rows (consolidate), 7
  ktensor, 8, 10, 15, 16
  lose, 9, 14, 15, 21
  lose (scalar), 21
  lose_repeats (consolidate), 7
  Ops (Ops.kform), 17
  Ops.kform, 17
  print.kform (print.stokes), 19
  print.ktensor (print.stokes), 19
  print.stokes, 19
  pull-back (transform), 23
  pullback (transform), 23
  push-forward (transform), 23
  pushforward (transform), 23
  retain (keep), 13
  rform, 20
  rkform (rform), 20
  rktensor (rform), 20
  rtensor (rform), 20
  scalar, 21, 28
  spray, 3
  star (hodge), 11
  stokes-package, 2
  stretch (transform), 23
  symbolic, 22
  transform, 23
  volume, 24
  wedge, 9, 11, 17, 24, 26
  wedge2 (wedge), 26
  zap, 27
  zapsmall (zap), 27
  zaptiny (zap), 27
zero, 27
zeroform, 21, 25
zeroform (zero), 27
zerotensor (zero), 27