Package ‘madness’

April 19, 2019

Maintainer  Steven E. Pav <shabbychef@gmail.com>
Version  0.2.6
Date  2019-04-18
License  LGPL-3
Title  Automatic Differentiation of Multivariate Operations

BugReports  https://github.com/shabbychef/madness/issues

Description  An object that supports automatic differentiation of matrix- and multidimensional-valued functions with respect to multidimensional independent variables. Automatic differentiation is via 'forward accumulation'.

Depends  R (>= 3.2.0)
Imports  Matrix, matrixcalc, expm, methods
Suggests  testthat, dplyr, tidyr, lubridate, SharpeR, sandwich, formatR, knitr

URL  https://github.com/shabbychef/madness

VignetteBuilder  knitr

Collate  'AllClass.r' 'utils.r' 'Ops.r' 'bind.r' 'blockrep.r'
  'coerce.r' 'data.r' 'det.r' 'diag.r' 'eigen.r' 'elwise.r'
  'madness_pkg.r' 'matwise.r' 'max.r' 'norm.r' 'numderiv.r'
  'reshape.r' 'solve.r' 'sum.r' 'sums.r' 'theta.r'
  'to_objective.r' 'trace.r' 'twomoments.r' 'vcov.r' 'vec.r'

RoxygenNote  6.1.1

NeedsCompilation  no

Author  Steven E. Pav [aut, cre] (<https://orcid.org/0000-0002-4197-6195>)

Repository  CRAN

Date/Publication  2019-04-19 05:10:03 UTC
**R topics documented:**

accessor .......................................................... 2
arithops ........................................................... 3
as ................................................................. 6
as.madness ....................................................... 7
bind ............................................................... 8
blockrep ......................................................... 9
colsums ........................................................ 10
det ............................................................... 11
eigen ........................................................... 12
elwise ............................................................ 13
madness-class ................................................... 14
madness-NEWS .................................................. 16
madness-pkg .................................................... 17
marithops ....................................................... 18
matrix.trace .................................................... 20
matwise ........................................................ 20
max ............................................................... 21
norm ............................................................. 22
numderiv ....................................................... 23
outer ............................................................ 24
reshapes ........................................................ 25
setter ............................................................ 26
show ............................................................. 27
solve ............................................................ 27
stock_returns .................................................. 28
sumprod ........................................................ 29
theta .............................................................. 30
todiag .......................................................... 31
to_objective .................................................... 32
twomoments .................................................... 33
vcov.madness .................................................. 34
vec ............................................................... 35
wff3 ............................................................. 37
[ .............................................................. 38

**Index**

<table>
<thead>
<tr>
<th>accessor</th>
<th>Accessor methods.</th>
</tr>
</thead>
</table>

**Description**

Access slot data from a `madness` object.
Usage

val(x)

## S4 method for signature 'madness'
val(x)

## S4 method for signature 'madness'
dim(x)

## S4 method for signature 'madness'
length(x)

dvdx(x)

## S4 method for signature 'madness'
dvdx(x)

xtag(x)

## S4 method for signature 'madness'
xtag(x)

vtag(x)

## S4 method for signature 'madness'
vtag(x)

varx(x)

## S4 method for signature 'madness'
varx(x)

Arguments

x a madness object.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

Description

These perform basic arithmetic operations on madness objects: unary plus and minus, addition, subtraction, multiplication, division and power.
Usage

```r
## S4 method for signature 'madness,missing'
e1 + e2

## S4 method for signature 'madness,missing'
e1 - e2

## S4 method for signature 'madness,madness'
e1 + e2

## S4 method for signature 'madness,numeric'
e1 + e2

## S4 method for signature 'madness,array'
e1 + e2

## S4 method for signature 'numeric,madness'
e1 + e2

## S4 method for signature 'array,madness'
e1 + e2

## S4 method for signature 'madness,madness'
e1 - e2

## S4 method for signature 'madness,numeric'
e1 - e2

## S4 method for signature 'madness,array'
e1 - e2

## S4 method for signature 'numeric,madness'
e1 - e2

## S4 method for signature 'array,madness'
e1 - e2

## S4 method for signature 'madness,madness'
e1 * e2

## S4 method for signature 'madness,numeric'
e1 * e2

## S4 method for signature 'madness,array'
e1 * e2

## S4 method for signature 'numeric,madness'
e1 * e2
```

```
```r
## S4 method for signature 'array,madness'
e1 * e2

## S4 method for signature 'madness,madness'
e1 / e2

## S4 method for signature 'madness,numeric'
e1 / e2

## S4 method for signature 'madness,array'
e1 / e2

## S4 method for signature 'numeric,madness'
e1 / e2

## S4 method for signature 'array,madness'
e1 / e2

## S4 method for signature 'madness,madness'
e1 ^ e2

## S4 method for signature 'madness,numeric'
e1 ^ e2

## S4 method for signature 'madness,array'
e1 ^ e2

## S4 method for signature 'numeric,madness'
e1 ^ e2

## S4 method for signature 'array,madness'
e1 ^ e2

Arguments

e1, e2

madness or numeric values

Author(s)

Steven E. Pav <shabbychef@gmail.com>

Examples

set.seed(123)
y <- array(rnorm(3*3),dim=c(3,3))
dy <- matrix(rnorm(length(y)*2),ncol=2)
dx <- crossprod(matrix(rnorm(ncol(dy)*100),nrow=100))
obj0 <- madness(val=y,vtag='y',xtag='x',dvdx=dy, varx=dx)
z <- array(rnorm(3*3),dim=c(3,3))
```
as

as madness to something else

Description

Coerce as something else

Usage

## S4 method for signature 'madness'
as.array(x, ...)

## S4 method for signature 'madness'
as.matrix(x, ...)

## S4 method for signature 'madness'
as.numeric(x, ...)

Arguments

- x: a madness object
- ... further arguments passed to or from other methods.

Author(s)

Steven E. Pav <shabbychef@gmail.com>
as.madness

Convert model to a madness object.

Usage

as.madness(x, vtag=NULL, xtag=NULL)

## Default S3 method:
as.madness(x, vtag = NULL, xtag = NULL)

Arguments

x an object which can be fed to coef, and possibly vcov
vtag an optional name for the val variable.
xtag an optional name for the X variable.

Details

Attempts to stuff the coefficients and variance-covariance matrix of a model into a madness object.

Value

A madness object.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

Examples

xy <- data.frame(x=rnorm(100), y=runif(100), z=runif(100))
amod <- lm(z ~ x + y, xy)
amad <- as.madness(amod)
bind           Row and Column Bind

Description

Row and Column Bind

Usage

```r
\method{c}{madness}(...)

## S4 method for signature 'madness,missing'
cbind2(x, y, ...)

## S4 method for signature 'madness,madness'
cbind2(x, y, ...)

## S4 method for signature 'madness,ANY'
cbind2(x, y, ...)

## S4 method for signature 'ANY,madness'
cbind2(x, y, ...)

## S4 method for signature 'madness,missing'
rbind2(x, y, ...)

## S4 method for signature 'madness,madness'
rbind2(x, y, ...)

## S4 method for signature 'madness,ANY'
rbind2(x, y, ...)

## S4 method for signature 'ANY,madness'
rbind2(x, y, ...)
```

Arguments

- ... optional arguments for methods (ignored here).
- x, y madness or array, numeric, matrix objects.

Author(s)

Steven E. Pav <shabbychef@gmail.com>
blockrep

Replicate blocks of multidimensional value.

Description
Replicates a multidimensional object a number of times along given dimensions.

Usage
blockrep(x, nreps)
repto(x, newdim)
repto(x, newdim)

Arguments
x a madness object, representing a k-dimensional object.
nreps an l-vector of positive integers, representing how many times to copy the object.
newdim an l-vector of positive integers of the new dimension of the output object. These must be integer multiples of the input dimensions.

Details
Given a k-dimensional object, and an l-vector of positive integers, for l >= k, copy the input object l_i times in the ith dimension. Useful for replication and (slow, fake) outer products.

repto replicates to the given dimension, assuming the given dimension are integer multiples of the input dimensions.

Value
A madness object replicated out.

Note
An error will be thrown if nreps or newdim are improper.

Author(s)
Steven E. Pav <shabbychef@gmail.com>
colsums

Examples

```r
set.seed(123)
y <- array(rnorm(3*3),dim=c(3,3))
dy <- matrix(rnorm(length(y)*2),ncol=2)
dx <- crossprod(matrix(rnorm(ncol(dy)*100),nrow=100))
obj0 <- madness(val=y,vtag='y',xtag='x',dvdx=dy,varx=dx)

anobj <- blockrep(obj0,c(1,2,1))
anobj <- blockrep(obj0,c(1,1,2))
anobj <- repto(obj0,c(9,12,4))
```

Description

Form Row and Column Sums and Means for `madness` objects.

Usage

```r
# S4 method for signature 'madness'
colSums(x, na.rm = FALSE, dims = 1)
# S4 method for signature 'madness'
colMeans(x, na.rm = FALSE, dims = 1)
# S4 method for signature 'madness'
rowSums(x, na.rm = FALSE, dims = 1)
# S4 method for signature 'madness'
rowMeans(x, na.rm = FALSE, dims = 1)
```

Arguments

- `x` : madness object.
- `na.rm` : logical. Should missing values (including NaN) be omitted from the calculations?
- `dims` : integer: Which dimensions are regarded as ‘rows’ or ‘columns’ to sum over. For `row*`, the sum or mean is over dimensions `dims+1, ...`; for `col*` it is over dimensions `1:dims`.
- `...` : potentially further arguments, for method ‘<-’ generic compatibility.

Value

a `madness` object. Note that the sums are flattened to a column vector.

Author(s)

Steven E. Pav <shabbychef@gmail.com>
**Description**

Compute the determinant of a matrix. As for `base::determinant`, a list of the modulus and sign are returned.

**Usage**

```r
## S3 method for class 'madness'
determinant(x, logarithm = TRUE, ...)

det(x, ...)
```

```r
## S4 method for signature 'madness,ANY'
determinant(x, logarithm = TRUE, ...)
```

```r
## S4 method for signature 'madness,missing'
determinant(x, logarithm = TRUE, ...)
```

```r
## S4 method for signature 'madness,logical'
determinant(x, logarithm = TRUE, ...)
```

**Arguments**

- `x` madness object.
- `logarithm` logical; if TRUE (default) return the logarithm of the modulus of the determinant.
- `...` Optional arguments. At present none are used. Previous versions of `det` allowed an optional `method` argument. This argument will be ignored but will not produce an error.

**Value**

a list with elements modulus and sign, which are madness objects.

**Note**

throws an error for non-square matrices or non-matrix input.

**Author(s)**

Steven E. Pav <shabbychef@gmail.com>
**Description**

Computes eigenvalues and eigenvectors of numeric (double, integer, logical) or complex madness matrices.

**Usage**

```r
## S4 method for signature 'madness'
eigen(x, symmetric, only.values = FALSE,
     EISPACK = FALSE)
```

**Arguments**

- `x` madness object representing a numeric matrix whose spectral decomposition is to be computed.
- `symmetric` if TRUE, the matrix is assumed to be symmetric (or Hermitian if complex) and only its lower triangle (diagonal included) is used. If symmetric is not specified, the matrix is inspected for symmetry.
- `only.values` if TRUE, only the eigenvalues are computed and returned, otherwise both eigenvalues and eigenvectors are returned.
- `EISPACK` logical. Defunct and ignored.

**Details**

The singular value decomposition of the matrix $X$ is

$$X = UDV',$$

where $U$ and $V$ are orthogonal, $V'$ is $V$ transposed, and $D$ is a diagonal matrix with the singular values on the diagonal.

**Value**

a list with components

- `values` a madness object of a vector containing the $p$ eigenvalues of $x$, sorted in decreasing order, according to $\text{Mod(value)}$ in the assymetric case when they might be complex (even for real matrices). For real asymmetric matrices the vector will be complex only if complex conjugate pairs of eigenvalues are detected.
- `vectors` either a $p \times p$ matrix whose columns contain the eigenvectors of $x$ or NULL if only.values is TRUE. The vectors are normalized to unit length.
Recall that the eigenvectors are only defined up to a constant: even when the length is specified they are still only defined up to a scalar of modulus one (the sign for real matrices). If \( r \leftarrow \text{eigen}(A) \), and \( V \leftarrow r\text{vectors} \), \( \lambda \leftarrow r\text{values} \), then

\[
A = V \Lambda d V^{-1}
\]

(up to numerical fuzz), where \( \Lambda = \text{diag}(\lambda) \).

**Author(s)**

Steven E. Pav <shabbychef@gmail.com>

**References**


**See Also**

eigen.

---

**elwise**  
*Element-wise Multivariate Operations*

**Description**

Element-wise multivariate operations.

**Usage**

```r
## S4 method for signature 'madness'
abs(x)

## S4 method for signature 'madness'
exp(x)

## S4 method for signature 'madness'
log(x)

## S4 method for signature 'madness'
log10(x)

## S4 method for signature 'madness'
sqrt(x)
```
## madness-class

```r
## S4 method for signature 'madness'
sin(x)
```

```r
## S4 method for signature 'madness'
cos(x)
```

```r
## S4 method for signature 'madness'
tan(x)
```

### Arguments

- `x`  
  madness object.

### Details

These operations are scalar-to-scalar operations applied to each element of a multidimensional array.

### Note

The `exp`, `log`, and `sqrt` functions are not to be confused with the matrix-wise operations, `expm`, `logm` and `sqrtm`.

### Author(s)

Steven E. Pav <shabbychef@gmail.com>

### See Also

- `matwise`

---

### Description

An S4 class to enable forward differentiation of multivariate computations. Think of 'madness' as 'multivariate automatic differentiation -ness.' There is also a constructor method for `madness` objects, and a wrapper method.

### Usage

```r
## S4 method for signature 'madness'
initialize(.Object, val, dvdx, xtag = NA_character_,
  vtag = NA_character_, varx = matrix(nrow = 0, ncol = 0))

madness(val, dvdx = NULL, vtag = NULL, xtag = NULL, varx = NULL)
```
Arguments

- `.Object` a madness object, or proto-object.
- `val` an array of some numeric value, of arbitrary dimension.
- `dvdx` a matrix of the derivative of (the vector of) `val` with respect to some independent variable, `X`.
- `xtag` an optional name for the `X` variable.
- `vtag` an optional name for the `val` variable.
- `varx` an optional variance-covariance matrix of the independent variable, `X`.

Details

A madness object contains a (multidimensional) value, and the derivative of that with respect to some independent variable. The purpose is to simplify computation of multivariate derivatives, especially for use in the Delta method. Towards this usage, one may store the covariance of the independent variable in the object as well, from which the approximate variance-covariance matrix can easily be computed. See `vcov`.

Note that derivatives are all implicitly 'flattened'. That is, when we talk of the derivative of `i \times j` matrix `Y` with respect to `m \times n` matrix `X`, we mean the derivative of the `ij` vector `vec(Y)` with respect to the `mn` vector `vec(X)`. Moreover, derivatives follow the 'numerator layout' convention: this derivative is a `ij \times mn` matrix whose first column is the derivative of `vec(Y)` with respect to `X_{1,1}`. Numerator layout feels unnatural because it makes a gradient vector of a scalar-valued function into a row vector. Despite this deficiency, it makes the product rule feel more natural. (2FIX: is this so?)

Value

An object of class `madness`.

Slots

- `val` an array of some numeric value. (Note that `array` includes `matrix` as a subclass.) The numeric value can have arbitrary dimension.
- `dvdx` a matrix of the derivative of (the vector of) `val` with respect to some independent variable, `X`. A Derivative is indeed a 2-dimensional matrix. Derivatives have all been 'flattened'. See the details. If not given, defaults to the identity matrix, in which case `val = X`, which is useful to initialization. Note that the derivative is with respect to an 'unrestricted' `X`.
- `xtag` an optional name for the `X` variable. Operations between two objects of the class with distinct `xtag` data will result in an error, since they are considered to have different independent variables.
- `vtag` an optional name for the `val` variable. This will be propagated forward.
- `varx` an optional variance-covariance matrix of the independent variable, `X`.

Author(s)

Steven E. Pav `<shabbychef@gmail.com>`
References


Examples

```r
obj <- new("madness", val=matrix(rnorm(10*10),nrow=10), dvdx=diag(100), xtag="foo", vtag="foo")
obj2 <- madness(val=matrix(rnorm(10*10),nrow=10), xtag="foo", vtag="foo"2")
```

News for package ‘madness’:

madness Version 0.2.6 (2019-04-19)

- emergency CRAN release to deal with change in generic signature for colSums, colMeans, rowSums, rowMeans.

madness Version 0.2.5 (2018-08-27)

- emergency CRAN release to deal with failing vignette on alternative BLAS.

madness Version 0.2.4 (2018-08-26)

- adding to unit tests.
- fix scalar to array promotion.
- fix broken vtag in aperm.
- add FF3 and stock returns data to build vignette.

madness Version 0.2.3 (2018-02-14)

- emergency CRAN release to deal with failing tests under alternative BLAS/LAPACK libraries.

madness Version 0.2.2 (2017-04-26)

- emergency CRAN release for upstream changes to diag. thanks to Martin Maechler for the patch.
Description

Automatic Differentiation of Matrix Operations.

Legal Mumbo Jumbo

madness is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

Note

This package is maintained as a hobby.
marithops

Basic Matrix Arithmetic Operations.

Description

These perform basic matrix arithmetic on madness objects: matrix multiplication, cross product, Kronecker product.

Usage

```r
## S4 method for signature 'madness,madness'
x %%%% y

## S4 method for signature 'madness,array'
x %%%% y

## S4 method for signature 'array,madness'
x %%%% y

## S4 method for signature 'madness,madness'
crossprod(x, y)
```

References


MM: DO use @importFrom (whimps use full imports ..)!
## Arguments

- **x**, **y**
  - madness or numeric matrix values.
- **...**
  - ignored here.

## Author(s)

Steven E. Pav &lt;shabbychef@gmail.com&gt;

## Examples

```r
set.seed(123)
y <- array(rnorm(3*3), dim=c(3,3))
dy <- matrix(rnorm(length(y)*2), ncol=2)
dx <- crossprod(matrix(rnorm(ncol(dy)*100), nrow=100))
obj0 <- madness(val=y, vtag='y', xtag='x', dvdx=dy, varx=dx)
z <- array(rnorm(3*3), dim=c(3,3))

anobj <- obj0 %*% obj0
anobj <- z %*% obj0
anobj <- crossprod(obj0)
anobj <- crossprod(obj0, z)
anobj <- tcrossprod(obj0, obj0)
# NYI:
# anobj <- obj0 %x% obj0
```
matrix.trace  

Matrix Trace

Description
Matrix Trace

Usage

matrix.trace(x)

## S4 method for signature 'ANY'
matrix.trace(x)

## S4 method for signature 'madness'
matrix.trace(x)

Arguments
x  madness object.

Author(s)
Steven E. Pav <shabbychef@gmail.com>

matwise  

Matrix-wise Multivariate Operations

Description
Element-wise multivariate operations.

Usage

## S4 method for signature 'madness'
sqrtm(x)

## S3 method for class 'madness'
chol(x, ...)

Arguments
x  madness object.
...  further arguments passed to or from other methods.
Details

These operations are operations on matrices: compute the symmetric square root or the Cholesky factor. In the future, the matrix exponent and logarithm may be implemented?

Author(s)

Steven E. Pav <shabbychef@gmail.com>

max

Maxima and Minima

Description

Return the maxima and minima of the input values.

Usage

```r
## S4 method for signature 'madness'
max(x, ..., na.rm = FALSE)

## S4 method for signature 'madness'
min(x, ..., na.rm = FALSE)
```

Arguments

- `x`: madness object arguments.
- `...`: madness object arguments.
- `na.rm`: a logical indicating whether missing values should be removed.

Details

`max` and `min` return the maximum or minimum of all the values present in their arguments.

If `na.rm` is FALSE and NA value in any of the arguments will cause a value of NA to be returned, otherwise NA values are ignored.

The minimum and maximum of a numeric empty set are `+Inf` and `-Inf` (in this order!) which ensures transitivity, e.g., \( \min(x1, \min(x2)) = \min(x1, x2) \). For numeric \( \max(x) = -\inf \) and \( \min(x) = +\inf \) whenever \( \text{length}(x) = 0 \) (after removing missing values if requested).

Author(s)

Steven E. Pav <shabbychef@gmail.com>
**Description**

Compute the norm of a vector or matrix, as determined by the `type`.

**Usage**

```r
maxeig(x)
```

```r
## S4 method for signature 'madness'
maxeig(x)
```

```r
## S4 method for signature 'madness,missing'
norm(x)
```

```r
## S4 method for signature 'madness,ANY'
norm(x, type = "One")
```

**Arguments**

- `x` : madness object.
- `type` : character string, specifying the `type` of matrix norm to be computed. A character indicating the type of norm desired.
  
  "0", "o" or "1" specifies the one norm, (maximum absolute column sum);
  "I" or "i" specifies the infinity norm (maximum absolute row sum);
  "F" or "f" specifies the Frobenius norm (the Euclidean norm of `x` treated as if it were a vector);
  "M" or "m" specifies the maximum modulus of all the elements in `x` and
  "2" specifies the “spectral” or 2-norm, which is the largest singular value (`svd`) of `x`.

  The default is "0". Only the first character of `type[1]` is used.

  ... further arguments passed to or from other methods.

**Value**

the matrix norm, a non-negative number.

**Note**

This should probably be fixed to return a scalar, not a 1 by 1 matrix?

**Author(s)**

Steven E. Pav <shabbychef@gmail.com>
**numderiv**  
**Numerical (approximate) Differentiation.**

**Description**  
Approximates the derivative of a function at the input by numerical methods.

**Usage**  
numderiv(f, x, eps=1e-8, type=c('forward','central','backward'), ...)

```r
## S4 method for signature 'ANY,array'
umderiv(f, x, eps = 1e-08, type = c("forward",  
    "central", "backward"), ...)

## S4 method for signature 'ANY,madness'
umderiv(f, x, eps = 1e-08, type = c("forward",  
    "central", "backward"), ...)
```

**Arguments**
- **f** a function, to be evaluated at and near \( x \).
- **x** array, matrix, or madness object.
- **eps** the 'epsilon', a small value added or subtracted from \( x \) to compute the first differences.
- **type** the type of first difference, case-insensitive, substrings ok.
- **...** arguments passed on to \( f \).

**Details**  
For a multivariate-valued function of multivariate data, approximates the derivative at a point via the forward, central, or backward first differences, returning a madness object.

**Value**  
A matrix if \( x \) is numeric; a madness object if \( x \) is a madness object.

**Author(s)**  
Steven E. Pav <shabbychef@gmail.com>

**Examples**
```r
f <- function(x,h) {  
    cos(x + h)
}
x <- array(rnorm(100),dim=c(10,10))
madx <- numderiv(f,x,1e-8,h=pi)
```
Outer product.

Description
Computes the outer product (or sum, quotient, etc) of the Cartesian product of two inputs.

Usage

```r
## S4 method for signature 'ANY,ANY'
outer(X, Y, FUN = "*", ...)

## S4 method for signature 'madness,madness'
outer(X, Y, FUN = "*", ...)

## S4 method for signature 'madness,array'
outer(X, Y, FUN = "*", ...)

## S4 method for signature 'array,madness'
outer(X, Y, FUN = "*", ...)

X %o% Y

## S4 method for signature 'madness,madness'
kroncker(X, Y)

## S4 method for signature 'madness,array'
kroncker(X, Y)

## S4 method for signature 'array,madness'
kroncker(X, Y)
```

Arguments

- `X`, `Y`: madness or numeric matrix values.
- `FUN`: a function to use on the outer products, found via `match.fun` (except for the special case "*").
- `...`: optional arguments to be passed to `FUN`.

Value

- a madness object.

Author(s)

Steven E. Pav <shabbychef@gmail.com>
Examples

```r
set.seed(123)
y <- array(rnorm(3*3), dim = c(3,3))
dy <- matrix(rnorm(length(y)*2), ncol = 2)
dx <- crossprod(matrix(rnorm(ncol(dy)*100), nrow = 100))
obj0 <- madness(val = y, vtag = 'y', xtag = 'x', dvdx = dy, varx = dx)

y1 <- array(rnorm(3*3), dim = c(3,3))
dy1 <- matrix(rnorm(length(y1)*2), ncol = 2)
dx1 <- crossprod(matrix(rnorm(ncol(dy1)*100), nrow = 100))
obj1 <- madness(val = y1, vtag = 'y1', xtag = 'x', dvdx = dy1, varx = dx1)

anobj <- outer(obj0, obj0, '*')
anobj <- outer(obj0, obj0, '+')
anobj <- outer(obj0, obj1, '-')
anobj <- outer(obj0, obj1, '/')
```

---

**Description**

Basic Reshape Operations

**Usage**

```r
## S4 method for signature 'madness'
t(x)

## S4 method for signature 'madness'
tril(x, k = 0)

## S4 method for signature 'madness'
triu(x, k = 0)

## S4 replacement method for signature 'madness'
dim(x) <- value

## S3 method for class 'madness'
aperm(a, perm = NULL, resize = TRUE, ...)
```

**Arguments**

- `x` madness object.
- `k` the index of the diagonal number from which to extract madness object.
- `value` an array of the new dimensions of the object value.
setter

a  the array to be transposed.
perm the subscript permutation vector, usually a permutation of the integers 1:n,
where n is the number of dimensions of a. When a has named dimnames, it can be
a character vector of length n giving a permutation of those names. The
default (used whenever perm has zero length) is to reverse the order of the di-
mensions.
resize a flag indicating whether the vector should be resized as well as having its ele-
ments reordered (default TRUE).
... Optional arguments used by specific methods. (None used at present.)

Author(s)
Steven E. Pav <shabbychef@gmail.com>

See Also
vec, todiag

setter  Setter methods.

Description
Modify slot data of a madness object. Note that the value and the derivative cannot easily be
changed, as allowing this form of access would likely result in badly computed derivatives.

Usage
xtag(x) <- value

## S4 replacement method for signature 'madness'
xtag(x) <- value
vtag(x) <- value

## S4 replacement method for signature 'madness'
vtag(x) <- value
varx(x) <- value

## S4 replacement method for signature 'madness'
varx(x) <- value

Arguments
x a madness object.
value the new value of the tag or derivative.
show

Author(s)

Steven E. Pav <shabbychef@gmail.com>

Description

Displays the madness object.

Usage

show(object)

### S4 method for signature 'madness'
show(object)

Arguments

object a madness object.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

Examples

obj <- madness(val=matrix(rnorm(10*10), nrow=10), xtag="foo", vtag="foo^2")
obj

solve

Basic Matrix Inversion

Description

Basic Matrix Inversion
Usage

```r
## S4 method for signature 'ANY,missing'
solve(a, b)

## S4 method for signature 'madness,missing'
solve(a, b)

## S4 method for signature 'madness,madness'
solve(a, b)

## S4 method for signature 'madness,array'
solve(a, b)

## S4 method for signature 'madness,ANY'
solve(a, b)

## S4 method for signature 'array,madness'
solve(a, b)

## S4 method for signature 'ANY,madness'
solve(a, b)
```

Arguments

- `a, b` madness object or matrix value.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

---

stock_returns Stock Returns Data

Description

Historical weekly relative returns of common shares of IBM and AAPL, downloaded from Quandl.

Usage

```r
data(stock_returns)
```

Format

A `data.frame` object with 1930 observations and 3 columns. The columns are defined as follows:

- **Date** The closing date at which the return was observed, as a `Date` object. These are Friday dates, ranging from January 1981 through December 2017.
AAPL The simple returns of AAPL common shares, based on weekly (adjusted) close prices. A value of 0.01 corresponds to a one percent return. Close prices are adjusted for splits and dividends by Quandl.

IBM The simple returns of IBM common shares, based on weekly (adjusted) close prices. A value of 0.01 corresponds to a one percent return. Close prices are adjusted for splits and dividends by Quandl.

Author(s)
Steven E. Pav <shabbychef@gmail.com>

Source

Examples
data(stock_returns)
str(stock_returns)

Description
Compute sum or product of madness objects.

Usage
```r
## S4 method for signature 'madness'
sum(x, ..., na.rm = FALSE)

## S4 method for signature 'madness'
prod(x, ..., na.rm = FALSE)
```

Arguments
- `x` a numeric or madness object.
- `...` ignored here.
- `na.rm` logical. Should missing values (including ‘NaN’) be removed?

Value
a madness object representing a scalar value.
**Examples**

```r
xv <- matrix(rnorm(5*5), ncol=5)
xmad <- madness(xv)
prod(xv)
sum(xv)
```

---

**theta**  
_Estimate the symmetric second moment array of values._

**Description**

Given rows of observations of some vector (or multidimensional data), estimates the second moment by taking a simple mean, returning a `madness` object.

**Usage**

```r
theta(X, vcov.func=vcov, xtag=NULL)
```

**Arguments**

- `X`  
a multidimensional array (or a data frame) of observed values.
- `vcov.func`  
a function which takes an object of class `lm`, and computes a variance-covariance matrix. If equal to the string "normal", we assume multivariate normal returns.
- `xtag`  
an optional string tag giving the name of the input data. defaults to figuring it out from the input expression.

**Details**

Given a \( n \times k_1 \times k_2 \times ... \times k_l \) array whose ‘rows’ are independent observations of \( X \), computes the \( k_1 \times k_2 \times ... \times k_l \times k_1 \times k_2 \times ... \times k_l \) array of the mean of \( \text{outer}(X, X) \) based on \( n \) observations, returned as a `madness` object. The variance-covariance is also estimated, and stored in the object.

One may use the default method for computing covariance, via the `vcov` function, or via a 'fancy' estimator, like `sandwich::vcovHAC`, `sandwich::vcovHC`, etc.

**Value**

A `madness` object representing the mean of the outer product of the tail dimensions of \( X \).

**Author(s)**

Steven E. Pav <shabbychef@gmail.com>
See Also
twomoments

Examples

```r
set.seed(123)
X <- matrix(rnorm(1000*3), ncol=3)
theta(X)

## Not run:
if (require(sandwich)) {
  th2 <- theta(X, vcov.func=vcovHC)
}

## End(Not run)
# works on data frames too:
set.seed(456)
X <- data.frame(a=rnorm(100), b=rnorm(100), c=1)
theta(X)
```

---

todiag   

Diagonal Operations

Description

Diagonal Operations

Usage

```r
## S4 method for signature 'madness'
diag(x)

todiag(x)
```

Arguments

- `x` madness object.

Note

the (somewhat odd) use of stats::diag for two different functions is not repeated here, at least for now.

Author(s)

Steven E. Pav <shabbychef@gmail.com>
See Also

- reshapes

\[
\text{to_objective} \quad \text{Convert a madness object into an objective value with gradient}
\]

Description

Given a madness object representing a scalar value, strip out that value and attach an attribute of its derivative as a gradient. This is a convenience method that simplifies construction of objective functions for optimization routines.

Usage

\[
\text{to_objective}(X)
\]

Arguments

- \(X\) a madness object representing a scalar.

Value

A scalar numeric with a gradient attribute of the derivative.

Note

An error will be thrown if the value is not a scalar.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

Examples

```r
# an objective function for matrix factorization with penalty:
fitfun <- function(R,L,Y,nu=-0.1) {
  dim(R) <- c(length(R),1)
  Rmad <- madness(R)
  dim(Rmad) <- c(ncol(L),ncol(Y))
  Err <- Y - L %*% Rmad
  penalty <- sum(exp(nu * Rmad))
  fit <- norm(Err,'F') + penalty
  to_objective(fit)
}
set.seed(1234)
L <- array(runif(30*5),dim=c(30,5))
Y <- array(runif(nrow(L)*20),dim=c(nrow(L),20))
R0 <- array(runif(ncol(L)*ncol(Y)),dim=c(ncol(L),ncol(Y)))
obj0 <- fitfun(R0,L,Y)
fooz <- nlm(fitfun, R0, L, Y, iterlim=3)
```
Estimate the mean and covariance of values.

Description
Given rows of observations of some vector (or multidimensional data), estimates the mean and covariance of the values, returning two madness objects. These have a common covariance and 'xtag', so can be combined together.

Usage
twomoments(X, diag.only=FALSE, vcov.func=vcov, xtag=NULL, df=NULL)

Arguments
X a multidimensional array (or a data frame) of observed values.
diag.only logical flag, defaulting to FALSE. When TRUE, only the diagonal of the covariance is computed, and returned instead of the entire covariance. This should be used for reasons of efficiency when only the marginal variances are needed.
vcov.func a function which takes an object of class lm, and computes a variance-covariance matrix. If equal to the string "normal", we assume multivariate normal returns.
xtag an optional string tag giving the name of the input data. defaults to figuring it out from the input expression.
df the number of degrees of freedom to subtract from the sample size in the denominator of the covariance matrix estimate. The default value is the number of elements in the mean, the so-called Bessel’s correction.

Details
Given a n × k1 × k2 × ... × k_l array whose 'rows' are independent observations of X, computes the k1 × k2 × ... × k_l array of the mean of X and the k1 × k2 × ... × k_l × k1 × k2...k_l array of the covariance, based on n observations, returned as two madness objects. The variance-covariance of each is estimated. The two objects have the same 'xtag', and so may be combined together. When the diag.only=TRUE, only the diagonal of the covariance is computed and returned.
One may use the default method for computing covariance, via the vcov function, or via a 'fancy' estimator, like sandwich::vcovHAC, sandwich::vcovHC, etc.

Value
A two element list. When diag.only=FALSE, the first element of the list is mu, representing the mean, a madness object, the second is Sigma, representing the covariance, also a madness object. When diag.only=TRUE, the first element is mu, but the second is sigmasq, a madness object representing the diagonal of the covariance matrix.

Author(s)
Steven E. Pav <shabbychef@gmail.com>
See Also

theta

Examples

set.seed(123)
X <- matrix(rnorm(1000*8),ncol=8)
alst <- twomoments(X)
markowitz <- solve(alst$Sigma,alst$mu)
vcov(markowitz)

# now compute the Sharpe ratios:
alst <- twomoments(X,diag.only=TRUE,df=1)
srs <- alst$mu / sqrt(alst$Sigma)

vcov.madness

Calculate Variance-Covariance Matrix for a model.

Description

Returns the variance-covariance matrix of the parameters computed by a madness object.

Usage

## S3 method for class 'madness'
vcov(object, ...)

Arguments

object

a madness object. A var x matrix must have been set on the object, otherwise an error will be thrown.

... additional arguments for method functions. Ignored here.

Details

Let $X$ represent some quantity which is estimated from data. Let $\Sigma$ be the (known or estimated) variance-covariance matrix of $X$. If $Y$ is some computed function of $X$, then, by the Delta method (which is a first order Taylor approximation), the variance-covariance matrix of $Y$ is approximately

$$
\frac{dY}{dX} \Sigma \left( \frac{dY}{dX} \right)^\top,
$$

where the derivatives are defined over the 'unrolled' (or vectorized) $Y$ and $X$.

Note that $Y$ can represent a multidimensional quantity. Its variance covariance matrix, however, is two dimensional, as it too is defined over the 'unrolled' $Y$. 
Value

A matrix of the estimated covariances between the values being estimated by the madness object. While \( Y \) may be multidimensional, the return value is a square matrix whose side length is the number of elements of \( Y \).

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

vcov.

Examples

```r
y <- array(rnorm(2*3), dim=c(2,3))
dy <- matrix(rnorm(length(y)*2), ncol=2)
dx <- crossprod(matrix(rnorm(ncol(dy)*100), nrow=100))
obj <- madness(val=y, dvy=dy, varx=dx)
print(vcov(obj))
```

---

vec

vectorize a multidimensional array.

Description

Turn a multidimensional array into a (column) vector. Turn a (typically symmetric) matrix into a (column) vector of the lower triangular part. Or reverse these operations.

Usage

```r
vec(x)
```

## S4 method for signature 'madness'

```r
vec(x)
```

## S4 method for signature 'array'

```r
vec(x)
```

vech(x, k = 0)

## S4 method for signature 'array'

```r
vech(x, k = 0)
```

## S4 method for signature 'madness'

```r
vech(x, k = 0)
```
ivech(x, k = 0, symmetric = FALSE)

## S4 method for signature 'ANY'
ivech(x, k = 0, symmetric = FALSE)

## S4 method for signature 'madness'
ivech(x, k = 0, symmetric = FALSE)

### Arguments

- **x**
  - a `madness` object or multidimensional array or matrix.

- **k**
  - the diagonal from which to subselect.

- **symmetric**
  - logical whether to put the array on the antidiagonal as well. Will throw an error if k > 0.

### Value

- a `madness` object or an array, of the vectorized array or the subselected part. For the inverse operations, promotes to a `madness` of a matrix, or a matrix.

### Author(s)

Steven E. Pav <shabbychef@gmail.com>

### See Also

- `reshapes`, in particular `tril`.

### Examples

```r
y <- matrix(rnorm(16), ncol=4)
sy <- y + t(y)
vx <- vec(sy)
vmy <- vec(madness(sy))
vy <- vech(sy)
vmhy <- vech(madness(sy))

ivech(c(1,2,3))
ivech(c(1,2,3),-1)
ivech(c(1,2,3),-1,symmetric=TRUE)
ivech(c(1,2,3,4,5,6,7,8),1)
```
Description
The weekly returns of the 3 Fama French Factors: Market, the cap factor SMB, and the growth factor HML.

Usage
wff3

Format
A data.frame object with 4800 observations and 5 columns. The data run from July, 1926 through June, 2018. As in the upstream source, the data are given in percents, meaning a value of 1.00 corresponds to a 1% movement. Note also that returns presumably are ‘simple’ returns, not log returns, though this is not clarified by the upstream source. The columns are defined as follows:

Date  The closing date, as a Date object. These are typically Saturdays.
Mkt  The Market weekly return. Note that the risk free rate has been added back to the excess returns published by the upstream source.
SMB  The cap factor weekly return.
HML  The growth factor weekly return.
RF   The risk-free rate, presumably as an weekly rate, though note that no corrections have been made for weekend effects when adding the risk-free rate back to the market rate.

Author(s)
Steven E. Pav <shabbychef@gmail.com>

Source

Examples

data(wff3)
str(wff3)
Extract parts of a madness value.

**Description**

Extract parts of a madness value.

**Usage**

```r
## S4 method for signature 'madness,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]
```

```r
## S4 method for signature 'madness,ANY,missing,ANY'
x[i, j, ..., drop = TRUE]
```

**Arguments**

- `x`  
  a madness object.

- `i`  
  indices specifying elements to extract or replace. Indices are numeric or character vectors or empty (missing) or NULL. Numeric values are coerced to integer as by `as.integer` (and hence truncated towards zero). Character vectors will be matched to the names of the object (or for matrices/arrays, the `dimnames`); see ‘Character indices’ below for further details.

  For `-indexing only: i, j, ... can be logical vectors, indicating elements/slices to select. Such vectors are recycled if necessary to match the corresponding extent. i, j, ... can also be negative integers, indicating elements/slices to leave out of the selection.

  When indexing arrays by `[` a single argument `i` can be a matrix with as many columns as there are dimensions of `x`; the result is then a vector with elements corresponding to the sets of indices in each row of `i`.

  An index value of `NULL` is treated as if it were `integer(0)`.

- `j, ...`  
  further indices specifying elements to extract or replace.

- `drop`  
  For matrices and arrays. If `TRUE` the result is coerced to the lowest possible dimension (see the examples). This only works for extracting elements, not for the replacement. See `drop` for further details.

**Author(s)**

Steven E. Pav <shabbychef@gmail.com>
Index

*Topic **data**
  stock_returns, 28
  wff3, 37

*Topic **differentiation**
  madness-class, 14

*Topic **multivariate**
  madness-class, 14

*Topic **package**
  madness-pkg, 17
  *, array, madness-class (arithops), 3
  *, array, madness-method (arithops), 3
  *, madness, array-class (arithops), 3
  *, madness, array-method (arithops), 3
  *, madness, madness-class (arithops), 3
  *, madness, madness-method (arithops), 3
  *, madness, numeric-class (arithops), 3
  *, madness, numeric-method (arithops), 3
  *, numeric, madness-class (arithops), 3
  *, numeric, madness-method (arithops), 3
  +, array, madness-class (arithops), 3
  +, array, madness-method (arithops), 3
  +, madness, array-class (arithops), 3
  +, madness, array-method (arithops), 3
  +, madness, madness-class (arithops), 3
  +, madness, madness-method (arithops), 3
  +, madness, numeric-class (arithops), 3
  +, madness, numeric-method (arithops), 3
  +, numeric, madness-class (arithops), 3
  +, numeric, madness-method (arithops), 3
  -, array, madness-class (arithops), 3
  -, array, madness-method (arithops), 3
  -, madness, array-class (arithops), 3
  -, madness, array-method (arithops), 3
  -, madness, madness-class (arithops), 3
  -, madness, madness-method (arithops), 3
  -, madness, numeric-class (arithops), 3
  -., madness, numeric-method (arithops), 3
  -., madness-class (arithops), 3
  -., numeric, madness-class (arithops), 3
  /., array, madness-class (arithops), 3
  /., array, madness-method (arithops), 3
  /., madness, array-class (arithops), 3
  /., madness, array-method (arithops), 3
  /., madness, madness-class (arithops), 3
  /., madness, madness-method (arithops), 3
  /., madness, numeric-class (arithops), 3
  /., madness, numeric-method (arithops), 3
  /., numeric, madness-class (arithops), 3
  /., numeric, madness-method (arithops), 3
  \[, madness, ANY, ANY, ANY-method [], 38
  \[, madness, ANY, missing, ANY-method [], 38
  %\%, array, madness-class (arithops), 18
  %\%, array, madness-method (arithops), 18
  %\%, madness, array-class (arithops), 18
  %\%, madness, array-method (arithops), 18
  %\%, madness, madness-class (arithops), 18
  %\%, madness, madness-method (arithops), 18
  %\% (outer), 24
  ^, array, madness-class (arithops), 3
  ^, array, madness-method (arithops), 3
  ^, madness, array-class (arithops), 3
  ^, madness, array-method (arithops), 3
  ^, madness, madness-class (arithops), 3
  ^, madness, madness-method (arithops), 3
  ^, madness, numeric-class (arithops), 3
  ^, numeric, madness-class (arithops), 3
  ^, numeric, madness-method (arithops), 3
  '%%', madness, madness-class (arithops), 18
  abs, madness-method (elwise), 13
  accessor, 2
  ANY, array-method (numderiv), 23

39
c.madness (bind), 8
cbind2, ANY, madness-method (bind), 8
cbind2, madness, ANY-method (bind), 8
cbind2, madness, madness-method (bind), 8
cbind2, madness, missing-method (bind), 8
colMeans (colsums), 10
colMeans, madness-method (colsums), 10
colSums (colsums), 10
colSums, 10
colSums, madness-method (colsums), 10
crossprod, madness-method (elwise), 13
crossprod, ANY, madness-method (marithops), 18
crossprod, ANY, madness-method (marithops), 18
crossprod, madness, ANY-method (marithops), 18
crossprod, madness, madness-method (marithops), 18
crossprod, madness, missing-method (marithops), 18
det, 11
determinant (det), 11
determinant, madness, ANY-method (det), 11
determinant, madness, logical-method (det), 11
determinant, madness, missing-method (det), 11
determinant.madness (det), 11
diag (todiag), 31
diag, madness-method (todiag), 31
dim, madness-method (accessor), 2
dim<-, madness, ANY-method (reshapes), 25
dim<-, madness, madness-method (reshapes), 25
dimnames, 38
drop, 38
dvdx (accessor), 2
dvdx, madness-method (accessor), 2
eigen, 12, 13
eigen, madness-method (eigen), 12
elwise, 13
exp, madness-method (elwise), 13
initialize, madness-class (madness-class), 14
initialize, madness-method (madness-class), 14
ivech (vec), 35
ivech, ANY-method (vec), 35
ivech, madness-method (vec), 35
kronecker, array, madness-class (outer), 24
kronecker, array, madness-method (outer), 24
kronecker, madness, array-class (outer), 24
kronecker, madness, array-method (outer), 24
kronecker, madness, madness-class (outer), 24
length, madness-method (accessor), 2
log, madness-method (elwise), 13
log10, madness-method (elwise), 13
madness, 30, 33
madness (madness-class), 14
madness-class, 14
madness-NEWS, 16
madness-pkg, 17
madness-pkg-package (madness-pkg), 17
marithops, 18
match.fun, 24
matrix.trace, 20
matrix.trace, ANY-method (matrix.trace), 20
matrix.trace, madness-method (matrix.trace), 20
matrix.trace, matrix-method (matrix.trace), 20
matwise, 14, 20
max, 21
max, madness-method (max), 21
maxeig (norm), 22
INDEX

maxeig,madness-method (norm), 22
min (max), 21
min,madness-method (max), 21
names, 38
norm, 22
norm,madness,ANY-method (norm), 22
norm,madness,missing-method (norm), 22
norm,madness-method (norm), 22
numderiv,23
numderiv,ANY,array-method (numderiv), 23
numderiv,ANY,madness-method (numderiv), 23
outer, 24
outer,ANY,ANY-method (outer), 24
outer,array,madness-method (outer), 24
outer,madness,array-method (outer), 24
outer,madness,madness-method (outer), 24
prod (sumprod), 29
prod,madness-method (sumprod), 29
rbind2,ANY,madness-method (bind), 8
rbind2,madness,ANY-method (bind), 8
rbind2,madness,madness-method (bind), 8
rbind2,madness,missing-method (bind), 8
repto (blockrep), 9
reshapes, 25, 32, 36
rowMeans (colsums), 10
rowMeans,madness-method (colsums), 10
rowSums (colsums), 10
rowSums,madness-method (colsums), 10
setter, 26
show, 27
show,madness-method (show), 27
sin,madness-method (elwise), 13
solve, 27
solve,ANY,madness-method (solve), 27
solve,ANY,missing-method (solve), 27
solve,array,madness-method (solve), 27
solve,madness,ANY-method (solve), 27
solve,madness,madness-method (solve), 27
solve,madness,missing-method (solve), 27
sqrt,madness-method (elwise), 13
sqrtm (matwise), 20
sqrtm,madness-method (matwise), 20
stock_returns, 28
sum (sumprod), 29
sum,madness-method (sumprod), 29
sumprod, 29
svd, 22
t (reshapes), 25
t,madness-method (reshapes), 25
tan,madness-method (elwise), 13
tcrossprod (marithops), 18
tcrossprod,ANY,madness-method (marithops), 18
tcrossprod,madness,ANY-method (marithops), 18
tcrossprod,madness,madness-method (marithops), 18
tcrossprod,madness,missing-method (marithops), 18
theta, 30, 34
to_objective, 32
todiag, 26, 31
todiag,madness-method (todiag), 31
tril (reshapes), 25
tril,madness-method (reshapes), 25
triu (reshapes), 25
triu,madness-method (reshapes), 25
twomoments, 31, 33
val (accessor), 2
val,madness-method (accessor), 2
varx (accessor), 2
varx,madness-method (accessor), 2
varx<- (setter), 26
varx<-,madness-method (setter), 26
vcov, 15, 30, 33, 35
vcov,madness, 34
vec, 26, 35
vec,array-method (vec), 35
vec,madness-method (vec), 35
vech, 35
vech,vec, 35
vech,vec, 35
vtag (accessor), 2
vtag,madness-method (accessor), 2
vtag<- (setter), 26
vtag<-,madness-method (setter), 26
wff3, 37
xtag (accessor), 2
xtag, madness-method (accessor), 2
xtag<- (setter), 26
xtag<-, madness-method (setter), 26