Package 'madness'

August 21, 2023

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

Version 0.2.8

Date 2023-08-20

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

LazyData true

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 7.1.1

NeedsCompilation no

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

Repository CRAN

Date/Publication 2023-08-21 07:42:32 UTC

R topics documented:

`accessor` *Accessor methods.*

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>

---

arithops *Basic Arithmetic Operations.*

Description

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

Usage

```
## 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
```

```
## 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))
```

```
anobj <- + obj0
anobj <- - obj0
anobj <- 6 - obj0
anobj <- 1 + obj0
anobj <- obj0 - 3
anobj <- z + obj0
anobj <- obj0 - z

obj1 <- obj0 ^ 2
anobj <- (0.3 * obj0) + (5.1 * obj1)

anobj <- 2 ^ obj0
anobj <- obj1 ^ obj0
anobj <- obj1 / obj0
anobj <- z / obj0
```

---

as *Coerce 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` *Coerce to a madness object.*

Description

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

```
\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>

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)

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

---

colsums *Form Row and Column Sums and Means*

Description

Form Row and Column Sums and Means for madness objects.

Usage

```
## 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.

Value

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

Author(s)

Steven E. Pav <shabbychef@gmail.com>

det *Matrix Determinant*

Description

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

Usage

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

det(x, ...)

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

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

## 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>

`eigen` *Spectral Decomposition of a Matrix*

Description

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

Usage

```
## 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, `isSymmetric(x)` is used.
`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 `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 <- eigen(A)`, and `V <- r$vectors; lam <- r$values`, then

$$A = VLmbdV^{-1}$$

(up to numerical fuzz), where `Lmbd =diag(lam)`.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Izenman, Alan Julian. "Reduced-Rank Regression for the Multivariate Linear Model." Journal of Multivariate Analysis 5, pp 248-264 (1975). https://www.sciencedirect.com/science/article/pii/0047259X75900421

Kato, Tosio. "Perturbation Theory for Linear Operators." Springer (1995). https://www.maths.ed.ac.uk/~v1ranick/papers/kato1.pdf

See Also

eigen.

---

elwise *Element-wise Multivariate Operations*

Description

Element-wise multivariate operations.

Usage

```
## 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)

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

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

## 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

---

madness-class *Madness Class.*

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

```
## 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 $\mathrm{vec}\,(Y)$ with respect to the mn vector $\mathrm{vec}\,(X)$. Moreover, derivatives follow the 'numerator layout' convention: this derivative is a $ij \times mn$ matrix whose first column is the derivative of $\mathrm{vec}\,(Y)$ 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

Petersen, Kaare Brandt and Pedersen, Michael Syskind. "The Matrix Cookbook." Technical University of Denmark (2012). http://www2.imm.dtu.dk/pubdb/pubs/3274-full.html

Magnus, Jan R. and Neudecker, H. "Matrix Differential Calculus with Applications in Statistics and Econometrics." 3rd Edition. Wiley Series in Probability and Statistics: Texts and References Section (2007).

Examples

```
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")
```

`madness-NEWS` *News for package 'madness':*

Description

News for package 'madness'.

madness Version 0.2.8 (2023-08-20)

- emergency CRAN release to fix package documentation.

madness Version 0.2.7 (2020-02-07)

- emergency CRAN release to deal with ellipsis in documentation.

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.

madness Version 0.2.1 (2017-04-13)

- emergency CRAN release for failed build.
- no new functionality.

madness Version 0.2.0 (2016-01-19)

- add static vignette.
- modify `twomoments`.
- release to CRAN.

madness Version 0.1.0.400 (2016-01-12)

- adding `max` and `min`.

madness Version 0.1.0.300 (2016-01-10)

- adding `eigen`.

madness Version 0.1.0.200 (2016-01-07)

- exporting `diag`.

madness Version 0.1.0 (2015-12-15)

- first CRAN release.

madness Initial Version 0.0.0.5000 (2015-12-01)

- first github release.

`marithops` *Basic Matrix Arithmetic Operations.*

Description

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

Usage

```
## 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)

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

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

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

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

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

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

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

Arguments

x, y	madness or numeric matrix 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))

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

```
## 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 x max(x) == -Inf and min(x) == +Inf whenever length(x) == 0 (after removing missing values if requested).

Author(s)

Steven E. Pav <shabbychef@gmail.com>

---

norm *Matrix and vector norms.*

Description

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

Usage

```
maxeig(x)

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

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

## 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. "O", "o" **or** "1" specifies the **o**ne norm, (maximum absolute column sum); "I" **or** "i" specifies the **i**nfinity norm (maximum absolute row sum); "F" **or** "f" specifies the **F**robenius norm (the Euclidean norm of x treated as if it were a vector); "M" **or** "m" specifies the **m**aximum 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 "O". Only the first character of type[1] is used.

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'),...)

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

## S4 method for signature 'ANY,madness'
numderiv(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

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

---

outer *Outer product.*

Description

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

Usage

```
## 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'
kronecker(X, Y)

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

## S4 method for signature 'array,madness'
kronecker(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

```
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,'/')
```

---

reshapes *Basic Reshape Operations*

Description

Basic Reshape Operations

Usage

```
## 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.
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 dimensions.
resize	a flag indicating whether the vector should be resized as well as having its elements 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.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

---

show *Show a madness object.*

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

```
## 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

```
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

Data were collated from Quandl on August 25, 2018. This data is no longer freely available from Quandl, but may be available directly from Nasdaq, see: https://www.nasdaq.com/market-activity/stocks/aapl/historical and https://www.nasdaq.com/market-activity/stocks/ibm/historical.

Examples

```
data(stock_returns)
str(stock_returns)
```

`sumprod` *Sum and Product.*

Description

Compute sum or product of `madness` objects.

Usage

```
## 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.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

Examples

```
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

```
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 $k_1 \times k_2 \times ... \times k_l \times k_1 \times k_2 ... k_l$ array of the mean of $\mathrm{outer}(X, X)$ 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

```
set.seed(123)
X <- matrix(rnorm(1000*3),ncol=3)
th <- 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=runif(100),b=rnorm(100),c=1)
th <- theta(X)
```

---

todiag *Diagonal Operations*

Description

Diagonal Operations

Usage

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

todiag(x)

## S4 method for signature 'madness'
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`

`to_objective` *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

```
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

```
# 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)
```

`twomoments` *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 $n \times k_1 \times k_2 \times ... \times k_l$ array whose 'rows' are independent observations of X, computes the $k_1 \times k_2 \times ... \times k_l$ array of the mean of X and the $k_1 \times k_2 \times ... \times k_l \times k_1 \times k_2 ... k_l$ 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$mu / sqrt(alst$sigmasq)
```

`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 `varx` 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 Σ 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

$$\frac{\mathrm{d}Y}{\mathrm{d}X}\Sigma\left(\frac{\mathrm{d}Y}{\mathrm{d}X}\right)^{\top},$$

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

```
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,dvdx=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

```
vec(x)

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

## S4 method for signature 'array'
vec(x)

vech(x, k = 0)

## S4 method for signature 'array'
vech(x, k = 0)

## S4 method for signature 'madness'
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

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

---

wff3 *Weekly Fama French 3 Factor Returns*

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 data, 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

Kenneth French data library, via Quandl. See http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html, data description at http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/Data_Library/f-f_factors.html.

Examples

```
data(wff3)
str(wff3)
```

[*Extract parts of a* madness *value.*

Description

Extract parts of a madness value.

Usage

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

## 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 or whole numbers as by as.integer or for large values by trunc (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