Package 'DyadiCarma'

October 17, 2024

Date 2024-10-11

Type Package

Title Dyadic matrices and their algebra using RcppArmadillo

Version 0.0.3

Description Algebra of dyadic matrices.

Depends R (>= 4.3.2), methods

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.3.2

Imports Rcpp (>= 1.0.12), RcppArmadillo (>= 14.0.2)

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Contents

`*,Dyadic,numeric-method`

Scalar multiplication of dyadic objects

Description

The scalar multiplication of a dyadic object.

Usage

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

Arguments

`e1`	numeric scalar.
`e2`	`Dyadic`-object.

Details

The operations are performed in a way that is consistent with the dyadic structure of the matrices.

Value

The `Dyadic`-object that is the result of the operation.

References

Kos, M., Podgórski, K., Wu, H. (2024) "Sparse"

See Also

`Dyadic-class` for the definition of the `Dyadic`-class; `dyadalg` for the dyadic decomposition of dyadic matrices;

Examples

```
#-----------------------------------------------------------#
#------- Scalar multiplication of dyadic objects --------#
#-----------------------------------------------------------#

# Construct four types of dyadic matrices with made of 1's
V <- construct(N, k, type = "vert") # vertical

mat_V <- as.matrix(V)

as.matrix(2*V)==2*mat_V
```

`+,Dyadic,Dyadic-method` *Addition of dyadic objects*

Description

Addition of the dyadic objects.

Usage

```
## S4 method for signature 'Dyadic,Dyadic'
e1 + e2
```

Arguments

`e1`	`Dyadic`-object.
`e2`	`Dyadic`-object.

Details

The operations are performed in a way that is consistent with the dyadic structure of the matrices.

Value

The `Dyadic`-object that is the result of the operation.

References

Kos, M., Podgórski, K., Wu, H. (2024) "Sparse"

See Also

`Dyadic-class` for the definition of the `Dyadic`-class; `dyadalg` for the dyadic decomposition of dyadic matrices;

Examples

```
#-------------------------------------------------------#
#------- Algebraic operations on dyadic matrices --------#
#-------------------------------------------------------#

# Construct four types of dyadic matrices with made of 1's
V <- construct(N, k, type = "vert") # vertical
H <- construct(N, k, type = "horiz") # horizontal
S <- construct(N, k, type = "symm") # symmetric
AS <- construct(N, k, type = "asymm") # asymmetric

# Addition of dyadic matrices

HpV=H+2*V # horizontal + vertical = asymmetric
HpV@type
HpAS=3*t(H)-6*t(AS) #transpose and linear combination
```

`-,Dyadic,Dyadic-method` *Subtraction of dyadic objects*

Description

The subtraction of dyadic objects.

Usage

```
## S4 method for signature 'Dyadic,Dyadic'
e1 - e2
```

Arguments

`e1`	A `Dyadic`-object.
`e2`	A `Dyadic`-object.

Details

The operations are performed in a way that is consistent with the dyadic structure of the matrices.

Value

The `Dyadic`-object that is the result of the subtraction.

References

Kos, M., Podgórski, K., Wu, H. (2024) "Sparse"

See Also

`Dyadic-class` for the definition of the `Dyadic`-class; `dyadalg` for the dyadic decomposition of dyadic matrices;

Examples

```
#-----------------------------------------------#
#------- Subtraction of dyadic objects --------#
#-----------------------------------------------#

# Construct four types of dyadic matrices with made of 1's
V <- construct(N, k, type = "vert") # vertical
H <- construct(N, k, type = "horiz") # horizontal
S <- construct(N, k, type = "symm") # symmetric
AS <- construct(N, k, type = "asymm") # asymmetric

# Convert the dyadic matrices to matrix format
mat_V <- as.matrix(V)
mat_H <- as.matrix(H)
mat_S <- as.matrix(S)
mat_AS <- as.matrix(AS)
```

```
# Subtract two dyadic matrices

NV=-V # negative of vertical
HpAS=3*t(H)-6*t(AS) #transpose and linear combination
HpAS@type #check the type of the result
```

`as.dyadic` *Extract a* `Dyadic` *object from a numeric matrix*

Description

This function extract a `Dyadic` object of given height and breadth from a classic matrix. If the corresponding sub-matrix extracted is not dyadic, the returned result will be wrong.

Usage

```
as.dyadic(mat, type, height, breadth)
```

Arguments

`mat`	A dyadic matrix with the classic R matrix representation.
`type`	string, one of the following character strings: `horiz`, `vert`,`symm`, and `asymm`, which indicates the type of dyadic object to be extracted;
`height`	The height of the dyadic matrix.
`breadth`	The breadth of the dyadic matrix.

Details

This function converts a dyadic matrix of the classic matrix form into the corresponding `Dyadic` object. If the input matrix is not dyadic it extracts the entries for the dyadic structure of the given height and breadth that fits to the upper-left hand side corner. Entries outside the fitted dyadic structure are neglected even if they are not equal to zero.

Value

A `Dyadic` object of the input type, height, and breadth representing the input matrix.

See Also

`Dyadic-class` for a description of the class;

Examples

```
#-------------------------------------------------------------#
#------------Creating vertically dyadic matrices--------------#
#-------------------------------------------------------------#

N <- 4
k <- 3
d <- k * (2^N - 1)
```

```
mat1 <- matrix(0, nrow = d, ncol = d)
mat2 <- matrix(0, nrow = d, ncol = d)

for (i in 1:N) {
    st_col_id <- (2^(i - 1) - 1) * k + 1
    en_col_id <- (2^(i - 1) - 1) * k + k
    for (j in 1:2^(N - i)) {
        st_row_id <- st_col_id - (2^(i - 1) - 1) * k
        en_row_id <- en_col_id + (2^(i - 1) - 1) * k
        mat1[st_row_id:en_row_id, st_col_id:en_col_id] <-
            as.matrix(rnorm((2^i - 1) * k^2), ncol = k, nrow = (2^i - 1) * k)
        mat2[st_row_id:en_row_id, st_col_id:en_col_id] <-
            as.matrix(rnorm((2^i - 1) * k^2), ncol = k, nrow = (2^i - 1) * k)
        st_col_id <- st_col_id + 2^i * k
        en_col_id <- en_col_id + 2^i * k
    }
}

mat1
mat2

#-------------------------------------------------------------#
#----------Creating corresponding dyadic objects--------------#
#-------------------------------------------------------------#

V1 <- as.dyadic(mat1, "vert", N, k) # A "vert" dyadic object
V2 <- as.dyadic(mat2, "vert", N, k) # A "vert" dyadic object

mat1S <- t(mat1) %*% mat1 # A symmetrically dyadic matrix
mat1AS <- t(mat2) %*% mat1 # An asymmetrically dyadic matrix
S <- as.dyadic(mat1S, "symm", N, k) # A "symm" dyadic object
AS <- as.dyadic(mat1AS, "asymm", N, k) # A "asymm" dyadic object

max(abs(as.matrix(S) - mat1S)) # Should be 0.
max(abs(as.matrix(AS) - mat1AS)) # Should be 0.


#-------------------------------------------------------------#
#---------------Creating a non-dyadic matrices----------------#
#-------------------------------------------------------------#

mat3 <- diag(d + 5)
mat3[1:d, 1:d] <- mat1

V3 <- as.dyadic(mat3, "vert", N, k) # Extract the upper-left dxd dyadic sub-matrix
max(abs(as.matrix(V3) - mat1)) # Should be 0.
```

`as.matrix,Dyadic-method` *Matrix representation of dyadic objects*

Description

Extracting the matrix representation of a `Dyadic`-object.

Usage

```
## S4 method for signature 'Dyadic'
as.matrix(x)
```

Arguments

x	Dyadic-object.

Details

The dyadic structure contains information about the type of matrix and its width and height.

Value

The result is a `width*(2^height-1)` x `width*(2^height-1)` matrix.

References

Kos, M., Podgórski, K., Wu, H. (2024) "Sparse"

See Also

`Dyadic-class` for the definition of the `Dyadic`-class; `dyadalg` for the dyadic decomposition of dyadic matrices;

Examples

```
#-----------------------------------------------------------#
#------- Matrix representation of dyadic objects --------#
#-----------------------------------------------------------#

# Construct four types of dyadic matrices with made of 1's
V <- construct(N, k, type = "vert") # vertical
H <- construct(N, k, type = "horiz") # horizontal
S <- construct(N, k, type = "symm") # symmetric
AS <- construct(N, k, type = "asymm") # asymmetric

# Convert the dyadic matrices to matrix format
mat_V <- as.matrix(V)
mat_H <- as.matrix(H)
mat_S <- as.matrix(S)
mat_AS <- as.matrix(AS)
```

`construct` *Construction of a* `Dyadic` *object*

Description

The function constructs a `Dyadic` object either with random entries (default) or with entries equal to one.

Usage

```
construct(height, breadth, type = "vert", distr = "nonrand", param = c(0, 1))
```

Arguments

height	positive integer, the number of dyadic levels;
breadth	positive integer, the breadth of the dyadic structure;
type	string, one of the following character strings: horiz,vert,symm, asymm, which indicates the type of dyadic matrix;
distr	string, if it is one the strings 'binom', 'unif', 'norm' it indicate the type of the distribution used for obtaining the entries, any other string, for example 'non-rand', results in non-random 1's in all entries.
param	vector of two numeric values, these are parameters for the distributions used to generate the entries.

Details

The function constructs a generic Dyadic-object of any type and in the case of the symm type with random entries the object represents a symmetric matrix.

Value

A Dyadic-object.

References

Kos, M., Podgórski, K., Wu, H. (2024) "Sparse"

See Also

Dyadic-class for a description of the class.

Examples

```
#-------------------------------------------------------------#
#---Building 'Dyadic' objects of arbitrary types and sizes ---#
#-------------------------------------------------------------#
N=5; k=4 #the height and breadth of a dyadic matrix

#Nonrandom vertical dyadic matrix with entries equal to 1
S=construct(N,k)

S@entries[[N]] #The top level entries
S@entries[[1]] #The bottom level entries

S@type='horiz' #'S' becomes horizontaly dyadic matrix, which is the transpose of the original object

#Symmetric dyadic with entries equal to 1
SS=construct(N,k,type='symm')
SS@entries[[2]] #The second bottom level entries

SS@aentries #This list is empty whenever the type is not 'asymm'

#Asymmetric dyadic with entries equal to one
AS=construct(N,k,type='asymm')
AS@entries[[2]] #The second bottom level entries
AS@aentries[[2]] #The asymmetric version (which happens to be also symmetric in this case)
```

```
#Truly asymmetric
AS=construct(N,k,type='asymm',distr='unif')
AS@entries[[2]] #The second bottom level entries
AS@aentries[[2]] #The asymmetric (which is also symmetric in this case)
```

---

dyadalg *Efficient factorization of a positive definite symmetrically dyadic matrix.*

Description

This function implement the efficient factorization of a positive definite symmetrically dyadic matrix $\mathbf{\Sigma}$. It computes the vertically dyadic matrix $\mathbf{P}$ such that $\mathbf{P}^\top \mathbf{\Sigma} \mathbf{P} = \mathbf{I}$.

Usage

```
dyadalg(S, inv = FALSE, band = FALSE)
```

Arguments

S	A Dyadic object of type "symm" representing a positive definite symmetrically dyadic matrix;
inv	The boolean value indicating whether the inverse of $\mathbf{\Sigma}$ should be returned.
band	The boolean value indicating whether the input S is a band matrix. If TRUE, then a optimized band-focused algorithm is called. If band==TRUE, but the input matrix is not a band matrix

Details

This function implement the efficient factorization of a positive definite symmetrically dyadic matrix.

Value

If inv == TRUE, then the inverse of $\mathbf{\Sigma}$, which is a (2^(height)-1)*breadth x (2^(height)-1)*breadth classic matrix, is returned. Otherwise, the vertically Dyadic object for $\mathbf{P}$ is returned.

See Also

Dyadic-class for a description of the class;

Examples

```
#-------------------------------------------------------------#
#---------Inverting a PD symmetrically dyadic matrix----------#
#-------------------------------------------------------------#

N <- 4
k <- 3

# A 45x45 vertically dyadic matrix
V <- construct(N, k, type = "vert", distr = "unif")
# A 45x45 symmetrically dyadic matrix
```

```
S <- t(V) %*% V
S@type <- "symm"
S@aentries <- list() # Convert S from "asymm" to "symm"

# Check what S looks like
matS <- as.matrix(S)
matS

# Find the vertically dyadic matrix that satisfies P^T S P = I
# using a dyadic factorization algorithm.
P <- dyadalg(S)
I1 <- as.matrix(t(P) %*% S %*% P)
I <- diag(dim(I1)[1])
max(abs(I1 - I)) # Should be trivially small

# Obtain the inverse of S via the dyadic algorithm
iS <- dyadalg(S, inv = TRUE)
I2 <- iS %*% matS
max(abs(I2 - I)) # Should be trivially small


#-------------------------------------------------------------#
#-----------------Inverting a PD band matrix------------------#
#-------------------------------------------------------------#

d <- k * (2^N - 1)
half_B <- matrix(0, nrow = d, ncol = d)
for (i in 1:d) {
   half_B[i, i:min(d, (i + k - 1))] <- rnorm(min(d, (i + k - 1)) - i + 1, mean = N, sd = 1 / N)
}
matB <- t(half_B) %*% half_B # matB is a PD band matrix with half bandwidth 3.

# Convert matB into a dyadic object B
B <- as.dyadic(matB, "symm", N, k)
iB <- dyadalg(B, inv = TRUE)
I <- diag(dim(matB)[1])
max(abs(iB %*% matB - I)) # Should be trivially small

iB_band <- dyadalg(B, inv = TRUE, band = TRUE)
max(abs(iB_band %*% matB - I)) # Should be trivially small

iB <- dyadalg(B)
iB_band <- dyadalg(B, band = TRUE)

max(abs(as.matrix(iB) - as.matrix(iB_band))) # Should be trivially small
```

`Dyadic-class` *The class to represent a dyadic matrix*

Description

The main class in the `Dyadic`-package used for representing three types of dyadic matrices: horizontal, vertical, symmetric, and asymmetric.

Value

running `new("Dyadic")` return an object that belongs to the class `Dyadic`, with the initialization of the default values for the fields.

Slots

`height` positive integer, the number of dyadic levels;

`breadth` positive integer, the breadth of the dyadic structure;

`type` string, one of the following character strings: `horiz`,`vert`,`symm`, `asymm` which indicates the type of dyadic matrix

- `horiz` horizontal,
- `vert` vertical,
- `symm` symmetric,
- `asymm` asymmetric,

where the last two types distinguish symmetrically dyadic matrices (they both have symmetric dyadic structure) that correspond to symmetric or not symmetric matrices.

`entries` list (of matrices); a list of the length `height` containing `(2^(l)-1)*breadth` x `2^(height-l)*breadth` matrices, where `l` is the index running through the list. Each matrix in the list includes the entries corresponding to `2^(height-l)` `(2^l-1)*breadth` x `breadth`-matrices put side by side columnwise in the `l`th level of a dyadic structure. In the 'symm'- and 'asymm'-cases, the terms below diagonal on the diagonal blocks are set to zero.

`aentries` list (of matrices); a list which is either empty if the slot `type` is not `'asymm'` or of the length `height` otherwise, in which the case it contains `(2^(l)-1)*breadth` x `2^(height-l)*breadth` matrices, where `l` is the index running through the list. Each matrix in the list includes the entries corresponding to `2^(height-l)`. `(2^l-1)*breadth` x `breadth`-matrices put side by side columnwise in the `l`th horizontal level of an asymmetric dyadic structure. The terms above and on the diagonal in the diagonal blocks are set to zero because they are accounted in the slot `entries`.

References

Kos, M., Podgórski, K., Wu, H. (2024) "Sparse"

See Also

`plot,Dyadic-method` for plotting methods for `Dyadic`-objects;

Examples

```
#-------------------------------------------------------------#
#------- Generating an object from the 'Dyadic' class --------#
#-------------------------------------------------------------#

# The most generic generation of an object of class 'Dyadic':
D <- new("Dyadic") # a generic format for 'Splinets' object
D
# The SLOTs of 'Dyadic' - the default values
D@height
D@breadth
D@type
D@entries[[1]]
D@aentries
```

```
N <- 4
k <- 3 # the height and breadth of a dyadic matrix

# The construction of a horizontally dyadic matrix with height 4 and breadth 3.

E <- list()
for (i in 1:4) {
    E[[i]] <- matrix(1, nrow = (2^(i) - 1) * 3, ncol = 2^(4 - i) * 3)
}


DD <- new("Dyadic", height = N, breadth = k, type = "horiz", entries = E)

DD

# The classic R matrix representation of DD.
mat_DD <- as.matrix(DD)
mat_DD


#-------------------------------------------------------------#
#--------------Transpose of the 'Dyadic' objects--------------#
#-------------------------------------------------------------#

# Construct four types of random dyadic matrices with the same shape.
V <- construct(N, k, type = "vert", distr = "unif")
H <- construct(N, k, type = "horiz", distr = "unif")
S <- construct(N, k, type = "symm", distr = "unif")
AS <- construct(N, k, type = "asymm", distr = "unif")
mat_V <- as.matrix(V)
mat_H <- as.matrix(H)
mat_S <- as.matrix(S)
mat_AS <- as.matrix(AS)

# Transpose of the dyadic object
VT <- t(V)
VT@type # should be 'horiz'
max(abs(as.matrix(VT) - t(mat_V))) # Should be 0

HT <- t(H)
HT@type # should be 'horiz'
max(abs(as.matrix(HT) - t(mat_H))) # Should be 0

ST <- t(S)
ST@type # will still be 'symm'
max(abs(as.matrix(ST) - mat_S)) # Should be 0 due to symmetry

AST <- t(AS)
AST@type # will still be 'asymm'
max(abs(as.matrix(AST) - t(mat_AS))) # Should be 0


#-------------------------------------------------------------#
#-----------Multiplications of the 'Dyadic' objects-----------#
#-------------------------------------------------------------#
```

```
# Any pairs of the four types are supported.

# The multiplication of two vertically dyadic matrix,
# which will result in a vertically dyadic matrix
VV <- V %*% V
VV@type # Should be "vert"

# The multiplication of a horizontally dyadic matrix with a vertically dyadic one,
# which will result in an asymmetrically dyadic matrix
HV <- H %*% V
HV@type # Should be "asymm"

# The multiplication of a horizontally dyadic matrix with a symmetrically dyadic one,
# which will result in an asymmetrically dydaic matrix
HS <- H %*% S
HS@type # Should be "asymm"

# The multiplication of a vertically dyadic matrix with a horizontally dyadic one,
# the result is no longer a dyadic object but a dense d x d matrix, where d = k * (2^N - 1)
VH <- V %*% H

# The multiplication of a symmetrically dyadic matrix with a symmetrically dyadic one,
# the result is no longer a dyadic object but a dense d x d matrix, where d = k * (2^N - 1)
SS <- S %*% S

# The multiplication of a symmetrically dyadic matrix with an asymmetrically dyadic one,
# the result is no longer a dyadic object but a dense d x d matrix, where d = k * (2^N - 1)
SAS <- S %*% AS
```

---

t,Dyadic-method *Construction of a* Dyadic *object*

Description

The Dyadic object transpose of a Dyadic object: (t(Dyadic).

Usage

```
## S4 method for signature 'Dyadic'
t(x)
```

Arguments

x	Dyadic-object;

Details

The operations are performed in a way that is consistent with the dyadic structure of the matrices.

Value

The Dyadic-object that is the result of the operation with properly defined fields.

References

Kos, M., Podgórski, K., Wu, H. (2024) "Sparse"

See Also

Dyadic-class for the definition of the Dyadic-class; dyadalg for the dyadic decomposition of dyadic matrices;

Examples

```
#-------------------------------------------#
#-------Transpose of a dyadic object --------#
#-------------------------------------------#

# Construct four types of dyadic matrices with made of 1's
V <- construct(N, k, type = "vert") # vertical
H <- construct(N, k, type = "horiz") # horizontal
S <- construct(N, k, type = "symm", distr='unif') # symmetric


V@type
H@type
t(H)@type
as.matrix(S)
as.matrix(t(S))
```

%*%,Dyadic,Dyadic-method

Matrix multiplication of dyadic objects

Description

The standard matrix multiplication of two Dyadic-objects.

Usage

```
## S4 method for signature 'Dyadic,Dyadic'
x %*% y
```

Arguments

x	Dyadic-object;
y	Dyadic-object;

Details

Both orders of multiplication are implemented: (scalar * dyadic) and (dyadic * scalar).

Value

Either a Dyadic-object or a regular matrix depending on the structure type of the input objects. The matrix outcome of multiplication is also reported as a message in the command line.

References

Kos, M., Podgórski, K., Wu, H. (2024) "Sparse"

See Also

Dyadic-class for the definition of the Dyadic-class; dyadalg for the dyadic decomposition of dyadic matrices;

Examples

```
#-------------------------------------------------#
#------- Multiplication of dyadic matrices --------#
#-------------------------------------------------#

# Construct four types of dyadic matrices with made of 1's
V <- construct(N, k, type = "vert") # vertical
H <- construct(N, k, type = "horiz") # horizontal
S <- construct(N, k, type = "symm") # symmetric
AS <- construct(N, k, type = "asymm") # asymmetric

# Convert the dyadic matrices to matrix format
mat_V <- as.matrix(V)
mat_H <- as.matrix(H)
mat_S <- as.matrix(S)
mat_AS <- as.matrix(AS)

# Multiplication of dyadic matrices
VV <- V %*% V # vertical * vertical = vertical
HH <- H %*% H # horizontal * horizontal = horizontal
HS <- H %*% S # horizontal * symmetric = asymmetric
HV <- H %*% V # horizontal * vertical = asymmetric
ASV <- AS %*% V # asymmetric * vertical = asymmetric

VH <- V %*% H # vertical * horizontal = non-dyadic
VS <- V %*% S # vertical * symmetric = non-dyadic
VAS <- V %*% AS # vertical * asymmetric = non-dyadic

SS <- S %*% S # symmetric * symmetric = non-dyadic
ASAS <- AS %*% AS # asymmetric * asymmetric = non-dyadic
ASH <- AS %*% H # asymmetric * horizontal = non-dyadic

dim(ASAS) #regular matrix
```

Index