Package ‘DyadiCarma’

December 27, 2024

Date 2024-12-27

Type Package

Title Dyadic matrices and their algebra using Rcpp and RcppArmadillo

Version 0.1

Description Algebra and factorization of dyadic matrices.

Depends R (>= 4.0.0), methods

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.3.2

Imports Rcpp (>= 1.0.12)

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Contents

as.dyadic 1
as.matrix,Dyadic-method 3
construct 4
dyadFac 6
Dyadic Arithmetic 7
Dyadic-class 9
t,Dyadic-method 10
%*%,Dyadic,Dyadic-method 11

Index **13**

`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

all(as.matrix(S) == mat1S) # Should be TRUE.
all(as.matrix(AS) == mat1AS) # Should be TRUE.


#----------------------------------------------------------#
#---------------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
all(as.matrix(V3) == mat1) # Should be TRUE.
```

---

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). Efficient inversion of sparse positive definite matrices through a dyadic factorization. Manuscript in preparation.

See Also

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

Examples

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

N <- 4
k <- 3

# 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", distr="norm") # 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 'nonrand', 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). Efficient inversion of sparse positive definite matrices through a dyadic factorization. Manuscript in preparation.

See Also

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

Examples

```
#------------------------------------------------------------#
#---Building 'Dyadic' objects of arbitrary types and sizes ---#
#------------------------------------------------------------#
N <- 4
k <- 3 # 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 second bottom level asymmetric entries
```

`dyadFac` *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

```
dyadFac(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 one, the function will return the corresponding result for the band part of the input 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 <- dyadFac(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 <- dyadFac(S, inv = TRUE)
I2 <- iS %*% matS
max(abs(I2 - I)) # Should be trivially small
iS_solve <- solve(matS)
I3 <- iS_solve %*% matS
max(abs(I3 - I)) # The result obtained using built-in method for inversion

#-------------------------------------------------------------#
#-----------------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 <- dyadFac(B, inv = TRUE)
I <- diag(dim(matB)[1])
max(abs(iB %*% matB - I)) # Should be trivially small

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

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

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

`Dyadic Arithmetic` *Arithmetic methods for Dyadic objects*

Description

Implements arithmetic operations for Dyadic objects, including negation, addition, subtraction, and scalar multiplication.

Methods

Unary '-' Negates a Dyadic object.

'+' Adds two Dyadic objects.

'-' Subtracts one Dyadic object from another.

'*' Multiplies a Dyadic object by a scalar or vice versa.

References

Kos, M., Podgórski, K., & Wu, H. (2024). Efficient inversion of sparse positive definite matrices through a dyadic factorization. Manuscript in preparation.

See Also

`Dyadic-class` for the definition of the `Dyadic`-class; `AsMatrix` for extracting the matrix representation of a `Dyadic`-object

Examples

```
#--------------------------------------------------------#
#-------- Arithmetic methods for dyadic objects ---------#
#--------------------------------------------------------#

N <- 4
k <- 3

# 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

# Negation of dyadic objects (matrices)
NegV <- -V
NegV@type
all(as.matrix(NegV) == -as.matrix(V)) # Should be TRUE

# Addition of dyadic objects (matrices)
HpV <- H + V # horizontal + vertical = asymmetric
HpV@type

# Subtraction of dyadic objects (matrices)
SmAS <- S - AS # symmetric - asymmetric = asymmetric
SmAS@type

# Scalar multiplication of dyadic objects (matrices)
DoubleV <- 2 * V # Scalar multiplication does not change the type
VDouble <- V * 2 # Scalar multiplication does not change the type
DoubleV@type
VDouble@type
all(as.matrix(DoubleV) == 2 * as.matrix(V)) # Should be TRUE
all(as.matrix(VDouble) == as.matrix(VDouble)) # Should be TRUE

# Linear combination
linearComb <- -S + 3 * H - 6 * AS + V # linear combination of dyadic matrices
linearComb@type # "asymm"
```

`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). Efficient inversion of sparse positive definite matrices through a dyadic factorization. Manuscript in preparation.

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)
View(mat_DD)
```

`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). Efficient inversion of sparse positive definite matrices through a dyadic factorization. Manuscript in preparation.

See Also

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

Examples

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

N <- 4
k <- 3

# 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

t(V)@type # The transpose of a vertical dyadic matrix is horizontal
t(H)@type # The transpose of a horizontal dyadic matrix is vertical

all(as.matrix(t(V)) == t(as.matrix(V))) # Should be TRUE
all(as.matrix(S) == as.matrix(t(S))) # Should be TRUE
```

%*%,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). Efficient inversion of sparse positive definite matrices through a dyadic factorization. Manuscript in preparation.

See Also

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

Examples

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

N <- 4
k <- 3

# 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

*,Dyadic,numeric-method (Dyadic Arithmetic), 7
*,numeric,Dyadic-method (Dyadic Arithmetic), 7
+,Dyadic,Dyadic-method (Dyadic Arithmetic), 7
-,Dyadic,ANY-method (Dyadic Arithmetic), 7
-,Dyadic,Dyadic-method (Dyadic Arithmetic), 7
-,Dyadic-method (Dyadic Arithmetic), 7
%*%,Dyadic,Dyadic-method, 11

as.dyadic, 1
as.matrix,Dyadic-method, 3
AsMatrix, *8*

construct, 4

dyadFac, *4*, 6, *11*, *12*
Dyadic Arithmetic, 7
Dyadic-class, 9

t,Dyadic-method, 10