Package ‘jointDiag’

September 9, 2017

Version 0.3
Date 2017-09-09
Title Joint Approximate Diagonalization of a Set of Square Matrices
Author Cedric Gouy-Pailler <cedric.gouypailler@gmail.com>
Maintainer Cedric Gouy-Pailler <cedric.gouypailler@gmail.com>

Depends

Suggests

Description Different algorithms to perform approximate joint diagonalization of a finite set of square matrices. Depending on the algorithm, orthogonal or non-orthogonal diagonalizer is found. These algorithms are particularly useful in the context of blind source separation.

License GPL (>= 2)
URL https://github.com/gouypailler/jointDiag
RoxygenNote 6.0.1
NeedsCompilation yes
Repository CRAN
Date/Publication 2017-09-09 21:51:24 UTC

R topics documented:

ajd .................................................. 2
ffdiag ........................................... 3
jadiag ........................................... 4
jedi ............................................... 6
qdiag ............................................ 7
uwedge .......................................... 9

Index 11
ajd  

Wrapper: Joint approximate diagonalization of a set of matrices

Description

This function is mainly a wrapper to the different algorithms provided in the package. So see the help of the different algorithms for the details.

Usage

ajd(M, A0 = NULL, B0 = NULL, eps = .Machine$double.eps, itermax = 200, keepTrace = FALSE, methods = c("jedi"))

Arguments

M  DOUBLE ARRAY (KxKxN). Three-dimensional array with dimensions KxKxN representing the set of square and real-valued matrices to be jointly diagonalized. N is the number of matrices. Matrices are KxK square matrices.

A0  DOUBLE MATRIX (KxK). The initial guess of the inverse of a joint diagonalizer. If NULL, an initial guess is automatically generated by the algorithm.

B0  DOUBLE MATRIX (KxK). The initial guess of a joint diagonalizer. If NULL, an initial guess is automatically generated by the algorithm.

eps  DOUBLE. The algorithm stops when the criterion difference between two iterations is less than eps.

itermax  INTEGER. Alternatively, the algorithm stops when itermax sweeps have been performed without reaching convergence. If the maximum number of iterations is performed, a warning appears.

keepTrace  BOOLEAN. Do we want to keep the successive estimations of the joint diagonalizer.

methods  STRING. One or more methods, choosen among the set of available algorithms. Possible values are: jedi, ffdiag, jadiag, uwedge, qdiag

Details

This function is mainly a wrapper to use the different algorithms provided in the package (see help of the different functions).

Value

If the number of methods is one, the result is the structure provided by the algorithm used.

If the number of methods is more than one, a list of results provided by each algorithm is given. Names of the list correspond to methods.

Author(s)

Cedric Gouy-Pailler (cedric.gouypailler@gmail.com)
Examples

# generating diagonal matrices
D <- replicate(30, diag(rchisq(df=1,n=10)), simplify=FALSE)
# mixing and demixing matrices
B <- matrix(rnorm(100), 10, 10)
A <- solve(B)
C <- array(NA, dim=c(10, 10, 30))
for (i in 1:30) C[,,i] <- A %*% D[[i]] %*% t(A)
adj(C, method=c("jedi", "ffdiag"))

ffdiag

Joint Approximate Diagonalization of a set of square, symmetric and real-valued matrices

Description

This function performs a Joint Approximate Diagonalization of a set of square and real-valued matrices.

Usage

ffdiag(C0, V0 = NULL, eps = .Machine$double.eps, itermax = 200, keepTrace = FALSE)

Arguments

C0

DOUBLE ARRAY (KxKxN). Three-dimensional array with dimensions KxKxN representing the set of square and real-valued matrices to be jointly diagonalized. N is the number of matrices. Matrices are KxK square matrices.

V0

DOUBLE MATRIX (KxK). The initial guess of a joint diagonalizer. If NULL, an initial guess is automatically generated by the algorithm.

eps

DOUBLE. The algorithm stops when the criterium difference between two iterations is less than eps.

itermax

INTEGER. Alternatively, the algorithm stops when itermax sweeps have been performed without reaching convergence. If the maximum number of iteration is performed, a warning appears.

keepTrace

BOOLEAN. Do we want to keep the successive estimations of the joint diagonalizer.

Details

Given a set $C_i$ of N KxK real-valued matrices, the algorithm is looking for a matrix $B$ such that $\forall i \in [1, N]$, $BC_iB^T$ is as close as possible of a diagonal matrix.
Value

- **B**
  Estimation of the Joint Diagonalizer.
- **criter**
  Successive estimates of the cost function across sweeps.
- **B_trace**
  Array of the successive estimates of B across iterations.

Author(s)

Cedric Gouy-Pailler (cedric.gouypailler@gmail.com), from the initial matlab code by A. Ziehe.

References


Examples

```r
# generating diagonal matrices
D <- replicate(30, diag(rchisq(df=1,n=10)), simplify=FALSE)
# Mixing and demixing matrices
B <- matrix(rnorm(100),10,10)
A <- solve(B)
C <- array(NA,dim=c(10,10,30))
for (i in 1:30) C[,,i] <- A %*% D[[i]] %*% t(A)
B_est <- ffdiag(C)$B
# B_est should be an approximate of B=solve(A)
B_est %*% A
# close to a permutation matrix (with random scales)
```

---

**jadiag**

_Joint Approximate Diagonalization of a set of square, symmetric and real-valued matrices_

Description

This function performs a Joint Approximate Diagonalization of a set of square, symmetric and real-valued matrices.

Usage

```r
jadiag(M, W_est0 = NULL, eps = .Machine$double.eps, itermax = 200, keepTrace = FALSE)
```
**Arguments**

- **M**
  - DOUBLE ARRAY (KxKxN). Three-dimensional array with dimensions KxKxN representing the set of square, symmetric and real-valued matrices to be jointly diagonalized. N is the number of matrices. Matrices are KxK square matrices.

- **W_est0**
  - DOUBLE MATRIX (KxK). The initial guess of a joint diagonalizer. If NULL, an initial guess is automatically generated by the algorithm.

- **eps**
  - DOUBLE. The algorithm stops when the criterium difference between two iterations is less than eps.

- **itermax**
  - INTEGER. Alternatively, the algorithm stops when itermax sweeps have been performed without reaching convergence. If the maximum number of iterations is performed, a warning appears.

- **keepTrace**
  - BOOLEAN. Do we want to keep the successive estimations of the joint diagonalizer.

**Details**

Given a set $C_i$ of N KxK symmetric and real-valued matrices, the algorithm is looking for a matrix $B$ such that $\forall i \in [1, N]$, $BC_iB^T$ is as close as possible of a diagonal matrix.

**Value**

- **B**
  - Estimation of the Joint Diagonalizer.

- **criter**
  - Successive estimates of the cost function across sweeps.

- **B_trace**
  - Array of the successive estimates of B across iterations.

**Author(s)**

Cedric Gouy-Pailler (cedric.gouypailler@gmail.com), from the initial C code by Dinh-Tuan Pham.

**References**


**Examples**

```r
# generating diagonal matrices
D <- replicate(30, diag(rchisq(df=1,n=10)), simplify=FALSE)
# Mixing and demixing matrices
B <- matrix(rnorm(100),10,10)
A <- solve(B)
C <- array(NA,dim=c(10,10,30))
for (i in 1:30) C[,,i] <- A %*% D[[i]] %*% t(A)
B_est <- jadiag(C)$B
# B_est should be an approximate of B=solve(A)
B_est %*% A
# close to a permutation matrix (with random scales)
```
Approximate non-orthogonal joint diagonalization of a set of square real-valued matrices

Description

This function performs a Joint Approximate Diagonalization of a set of square and real-valued matrices (not necessarily symmetric). The algorithm seeks the inverse of the joint diagonalizer (the mixing matrix in terms of source separation).

The algorithm uses Givens and hyperbolic rotations to find the inverse of a non-orthogonal joint diagonalizer. It is an extension of the JADE method (orthogonal joint diagonalization).

Usage

\[
jedi(M, A0 = NULL, eps = .Machine$double.eps, itermax = 200, \text{keepTrace} = \text{FALSE})\]

Arguments

- **M**
  - DOUBLE ARRAY (KxKxN). Three-dimensional array with dimensions KxKxN representing the set of square and real-valued matrices to be jointly diagonalized. N is the number of matrices. Matrices are KxK square matrices.

- **A0**
  - DOUBLE MATRIX (KxK). The initial guess of the inverse of a joint diagonalizer. If NULL, an initial guess is automatically generated by the algorithm.

- **eps**
  - DOUBLE. The algorithm stops when the criterium difference between two iterations is less than eps.

- **itermax**
  - INTEGER. Alternatively, the algorithm stops when itermax sweeps have been performed without reaching convergence. If the maximum number of iteration is performed, a warning appears.

- **keepTrace**
  - BOOLEAN. Do we want to keep the successive estimations of the joint diagonalizer.

Details

Given a set \( M \) of \( N \times K \) square and real-valued matrices, the algorithm is looking for a matrix \( A \) such that \( \forall i \in [1, N], A^{-1} C_i A^{-T} \) is as close as possible of a diagonal matrix.

Value

- **A**
  - Estimation of the Joint Diagonalizer.

- **criter**
  - Successive estimates of the cost function across sweeps.

- **A_trace**
  - Array of the successive estimates of A across iterations.
### qdiag

**Joint Approximate Diagonalization of a set of square, symmetric and real-valued matrices**

**Description**

This function performs a Joint Approximate Diagonalization of a set of square, symmetric and real-valued matrices.

**Usage**

```r
qdiag(C, W0 = NULL, eps = .Machine$double.eps, itermax = 200, keepTrace = FALSE)
```

**Arguments**

- **C**
  - DOUBLE ARRAY (KxKxN). Three-dimensional array with dimensions KxKxN representing the set of square, symmetric and real-valued matrices to be jointly diagonalized. N is the number of matrices. Matrices are KxK square matrices.

- **W0**
  - DOUBLE MATRIX (KxK). The initial guess of a joint diagonalizer. If NULL, an initial guess is automatically generated by the algorithm.
eps DOUBLE. The algorithm stops when the criterium difference between two iterations is less than eps.

itermax INTEGER. Alternatively, the algorithm stops when itermax sweeps have been performed without reaching convergence. If the maximum number of iteration is performed, a warning appears.

keepTrace BOOLEAN. Do we want to keep the successive estimations of the joint diagonalizer.

Details

Given a set $C_i$ of $N$ KxK symmetric and real-valued matrices, the algorithm is looking for a matrix $B$ such that $\forall i \in [1, N], BC_iB^T$ is as close as possible of a diagonal matrix.

Value

$B$ Estimation of the Joint Diagonalizer.

criter Successive estimates of the cost function across sweeps.

$B$-trace Array of the successive estimates of $B$ across iterations.

Note

Two versions of the quadratic optimization are present in the paper referenced below. These two versions have different complexities, $O(N \cdot K^3)$ and $O(K^5)$. Currently only the version with $O(N \cdot K^3)$ is implemented.

Author(s)

Cedric Gouy-Pailler (cedric.gouypailler@gmail.com), from the initial matlab code by R. Vollgraf.

References

R. Vollgraf and K. Obermayer; Quadratic Optimization for Approximate Matrix Diagonalization; IEEE Transaction on Signal Processing, 2006

Examples

# generating diagonal matrices
D <- replicate(30, diag(rchisq(df=1,n=10)), simplify=FALSE)
# Mixing and demixing matrices
B <- matrix(rnorm(100),10,10)
A <- solve(B)
C <- array(NA,dim=c(10,10,30))
for (i in 1:30) C[,,i] <- A %*% D[[i]] %*% t(A)
B_est <- qdiag(C)$B
# B_est should be an approximate of B=solve(A)
B_est %*% A
# close to a permutation matrix (with random scales)
Joint Approximate Diagonalization of a set of square, symmetric and real-valued matrices

Description

This function performs a Joint Approximate Diagonalization of a set of square, symmetric and real-valued matrices.

Usage

uwedge(M, W_est0 = NULL, eps = .Machine$double.eps, itermax = 200, keepTrace = FALSE)

Arguments

- **M**: DOUBLE ARRAY (KxKxN). Three-dimensional array with dimensions KxKxN representing the set of square, symmetric and real-valued matrices to be jointly diagonalized. N is the number of matrices. Matrices are KxK square matrices.
- **W_est0**: DOUBLE MATRIX (KxK). The initial guess of a joint diagonalizer. If NULL, an initial guess is automatically generated by the algorithm.
- **eps**: DOUBLE. The algorithm stops when the criterium difference between two iterations is less than eps.
- **itermax**: INTEGER. Alternatively, the algorithm stops when itermax sweeps have been performed without reaching convergence. If the maximum number of iteration is performed, a warning appears.
- **keepTrace**: BOOLEAN. Do we want to keep the successive estimations of the joint diagonalizer.

Details

Given a set $C_i$ of N KxK symmetric and real-valued matrices, the algorithm is looking for a matrix $B$ such that $\forall i \in [1, N], BC_iB^T$ is as close as possible of a diagonal matrix.

Value

- **B**: Estimation of the Joint Diagonalizer.
- **criter**: Successive estimates of the cost function across sweeps.
- **B_trace**: Array of the successive estimates of B across iterations.

Author(s)

Cedric Gouy-Pailler (cedric.gouypailler@gmail.com), from the initial matlab code by P. Tichavsky.
References


Examples

# generating diagonal matrices
D <- replicate(30, diag(rchisq(df=1, n=10)), simplify=FALSE)
# Mixing and demixing matrices
B <- matrix(rnorm(100),10,10)
A <- solve(B)
C <- array(NA, dim=c(10,10,30))
for (i in 1:30) C[,,i] <- A %*% D[[i]] %*% t(A)
B_est <- uwedge(C)$B
# B_est should be an approximate of B=solve(A)
B_est %*% A
# close to a permutation matrix (with random scales)
Index

*Topic algebra
    ajd, 2
    ffdiag, 3
   jadiag, 4
    jedi, 6
    qdiag, 7
    uwedge, 9

ajd, 2
ffdiag, 3
jadiag, 4
jedi, 6
qdiag, 7
uwedge, 9