Package ‘QZ’

April 13, 2025

Version 0.2-4

Date 2025-04-12

Title Generalized Eigenvalues and QZ Decomposition

Depends R (>= 3.6.0), methods, Matrix

Suggests fda

LazyLoad yes

LazyData yes

Description Generalized eigenvalues and eigenvectors use QZ decomposition (generalized Schur decomposition). The decomposition needs an N-by-N non-symmetric matrix A or paired matrices (A,B) with eigenvalues reordering mechanism. The decomposition functions are mainly based Fortran subroutines in complex*16 and double precision of LAPACK library (version 3.10.0 or later).

NeedsCompilation yes

Maintainer Wei-Chen Chen <wccsnow@gmail.com>

Author Wei-Chen Chen [aut, cre],
LAPACK authors [aut, cph]

Repository CRAN

Date/Publication 2025-04-13 04:40:02 UTC

Contents

`QZ-package` *Generalized Eigenvalues and QZ Decomposition*

Description

QZ package provides generalized eigenvalues and QZ decomposition (generalized Schur form) for an N-by-N non-symmetric matrix A or paired matrices (A,B) with eigenvalues reordering mechanism. The package is mainly based complex*16 and double precision of LAPACK library (version 3.4.2.)

Details

The QZ package contains R functions for generalized eigenvalues and QZ decomposition (generalized Schur form) for an N-by-N non-symmetric matrix A or paired matrices (A,B) via two main functions, `qz.geigen()` and `qz()`. The `qz()` function also provides an option for eigenvalues reordering.

The QZ package is also based on a minimum set of complex*16 and double precision of LAPACK and BLAS Fortran libraries. Most functions are wrapped in C via `.Call()` to avoid extra memory copy and to improve performance and memory usage.

Author(s)

Wei-Chen Chen <`wccsnow@gmail.com`>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://en.wikipedia.org/wiki/Schur_decomposition

https://www.netlib.org/lapack/

See Also

qz.geigen, qz,
qz.zgges, qz.zggev, qz.ztgsen, qz.dgges, qz.dggev, qz.dtgsen,
qz.zgees, qz.zgeev, qz.ztrsen, qz.dgees, qz.dgeev, qz.dtrsen.

Examples

```
## Not run:
demo(ex1_geigen, "QZ")
demo(ex2_qz, "QZ")
demo(ex3_ordqz, "QZ")
demo(ex4_fda_geigen, "QZ")

## End(Not run)
```

Conjugate transpose — *Conjugate Transpose for Complex Matrix*

Description

Conjugate transpose, Hermitian transpose, or Hermitian conjugate.

Usage

```
H(x)
```

Arguments

x	a complex matrix or vector.

Details

This is equivalent to `Conj(t.default(x))`.

Value

This returns a conjugate transpose of `x`.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

Examples

```
library(QZ, quiet = TRUE)

A <- matrix(c(-21.10 -22.50i, 53.50 -50.50i, -34.50 +127.50i,   7.50  +0.50i,
               -0.46  -7.78i, -3.50 -37.50i, -15.50  +58.50i, -10.50  -1.50i,
                4.30  -5.50i, 39.70 -17.10i, -68.50  +12.50i,  -7.50  -3.50i,
                5.50  +4.40i, 14.40 +43.30i, -32.50  -46.00i, -19.00 -32.50i),
            nrow = 4, byrow = TRUE)
H(A)
```

`Example datasets` *Small example datasets*

Description

These datasets are small for test operations and functions in complex and double precision/matrices.

Format

Each dataset contains information where it is from and two matrices in pair of (A,B) or single matrix (A) for testing functions `qz.*` or related functions, either in complex or in double precision.

Details

The example datasets are

Examples	Source
`exAB1`	https://www.netlib.org/lapack/lug/node124.html
`exAB2`	https://www.netlib.org/lapack/lug/node119.html
`exAB3`	https://support.nag.com/numeric/fl/nagdoc_fl23/xhtml/f08/f08yuf.xml
`exAB4`	https://support.nag.com/numeric/fl/nagdoc_fl23/xhtml/f08/f08ygf.xml
`exA1`	https://www.netlib.org/lapack/lug/node94.html
`exA2`	https://www.netlib.org/lapack/lug/node89.html
`exA3`	https://support.nag.com/numeric/fl/nagdoc_fl23/xhtml/f08/f08quf.xml
`exA4`	https://support.nag.com/numeric/fl/nagdoc_fl22/xhtml/f08/f08qgf.xml

The elements of dataset are (if any)

Elements	Usage
`description`	the source of data
`A`	the first matrix A
`B`	the second matrix B
`S`	the Shur form
`T`	the Shur form
`Q`	the left Shur vectors

Z	the right Shur vectors

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://en.wikipedia.org/wiki/Schur_decomposition

`fda.geigen` *Generalized Eigen Analysis as in fda Package*

Description

This is an equivalent function to `fda::geigen` which finds matrices L and M to maximize

tr(L'AM) / sqrt(tr(L'BL) tr(M'CM))

where A = a p x q matrix, B = p x p symmetric, positive definite matrix, B = q x q symmetric positive definite matrix, L = p x s matrix, and M = q x s matrix, where s = the number of non-zero generalized eigenvalues of A.

Usage

```
fda.geigen(Amat, Bmat, Cmat)
```

Arguments

Amat	a numeric matrix
Bmat	a symmetric, positive definite matrix with dimension = number of rows of A
Cmat	a symmetric, positive definite matrix with dimension = number of columns of A

Details

This function is equivalent to `fda::geigen(Amat, Bmat, Cmat)` except that this is rewritten and utilizes LAPACK functions via `qz.dggev`.

Also, `Lmat` and `Mmat` are both scaled such that L'BL and M'CM are identity matrices.

Value

list(values, Lmat, Mmat)

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

See Also

qz.geigen, qz.dggev.

Examples

```
library(QZ, quiet = TRUE)

A <- matrix(as.double(1:6), 2)
B <- matrix(as.double(c(2, 1, 1, 2)), 2)
C <- diag(as.double(1:3))

ret.qz <- fda.geigen(A, B, C)

### Verify
library(fda, quiet = TRUE)
ret.fda <- fda::geigen(A, B, C)
```

Generalized Eigenvalues

Generalized Eigen Values

Description

This function obtains generalized eigen values on input paired matrices (A,B) or a single matrix A.

Usage

```
geigen(A, B = NULL, only.values = FALSE, ...)

qz.geigen(A, B = NULL, only.values = FALSE, ...)
```

Arguments

A	a 'complex/real' matrix, dim = c(N, N).
B	a 'complex/real' matrix, dim = c(N, N).
only.values	if 'TRUE', only the eigenvalues are computed and returned, otherwise both eigenvalues and eigenvectors are returned.
...	options to qz.* functions.

Details

Call one of qz.zggev, qz.dggev, qz.zgeev, or qz.dgeev depending on the input arguments and types.

Value

Returns a list from the call.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz, ordqz.

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node122.html
(ret <- qz.geigen(exAB1$A, exAB1$B))

### https://www.netlib.org/lapack/lug/node117.html
(ret <- qz.geigen(exAB2$A, exAB2$B))

### https://www.netlib.org/lapack/lug/node92.html
(ret <- qz.geigen(exA1$A))

### https://www.netlib.org/lapack/lug/node87.html
(ret <- qz.geigen(exA2$A))
```

Print methods *Functions for Printing Objects According to Classes*

Description

Several classes are declared in **QZ**, and these are functions to print objects.

Usage

```
## S3 method for class 'zgges'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'zggev'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'ztgsen'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'dgges'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'dggev'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'dtgsen'
```

```
print(x, digits = max(4, getOption("digits") - 3), ...)

## S3 method for class 'zgees'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'zgeev'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'ztrsen'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'dgees'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'dgeev'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'dtrsen'
print(x, digits = max(4, getOption("digits") - 3), ...)
```

Arguments

`x`	an object with the class attributes.
`digits`	for printing out numbers.
`...`	other possible options.

Details

These are useful functions for summarizing and debugging. Use `names` or `str` to explore the details.

Value

The results will cat or print on the STDOUT by default.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

`qz.zgges`, `qz.zggev`, `qz.ztgsen`, `qz.dgges`, `qz.dggev`, `qz.dtgsen`, `qz.zgees`, `qz.zgeev`, `qz.ztrsen`, `qz.dgees`, `qz.dgeev`, `qz.dtrsen`.

Examples

```
## Not run:
# Functions applied by directly type the names of objects.

## End(Not run)
```

`QZ Decomposition` *QZ Decomposition*

Description

This function performs QZ decomposition on input paired matrices (A,B) or a single matrix A.

Usage

```
qz(A, B = NULL, select = NULL, only.values = FALSE, ...)
```

Arguments

`A`	a 'complex/real' matrix, dim = c(N, N).
`B`	a 'complex/real' matrix, dim = c(N, N).
`select`	specifies the eigenvalues in the selected cluster.
`only.values`	if 'TRUE', only the eigenvalues are computed and returned, otherwise both eigenvalues and eigenvectors are returned.
`...`	options to `qz.*` functions.

Details

If `select` is `NULL`, then call one of `qz.zgges`, `qz.dgges`, `qz.zgees`, or `qz.dgees` depending on the input arguments and types.

If `select` is not `NULL`, then call one of `qz.zgges` + `qz.ztgsen`, `qz.dgges` + `qz.dtgsen`, `qz.zgees` + `qz.ztrsen`, or `qz.dgees` + `qz.dtrsen` depending on the input arguments and types.

Value

Returns a list from the call.

Author(s)

Wei-Chen Chen <`wccsnow@gmail.com`>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

`ordqz`, `geigen`.

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node124.html
(ret <- qz(exAB1$A, exAB1$B))

### https://www.netlib.org/lapack/lug/node119.html
(ret <- qz(exAB2$A, exAB2$B))

### https://www.netlib.org/lapack/lug/node94.html
(ret <- qz(exA1$A))

### https://www.netlib.org/lapack/lug/node89.html
(ret <- qz(exA2$A))

# Reordering eigenvalues
select1 <- c(TRUE, FALSE, FALSE, TRUE)
select2 <- c(FALSE, TRUE, TRUE, FALSE)
(ret <- qz(exAB1$A, exAB1$B, select = select1))
(ret <- qz(exAB2$A, exAB2$B, select = select2))
(ret <- qz(exA1$A, select = select1))
(ret <- qz(exA2$A, select = select1))
```

`QZ Decomposition Reordering`

Reordering QZ Decomposition

Description

This function performs QZ decomposition on input paired matrices (A,B) or a single matrix A with reordering.

Usage

```
ordqz(A, B = NULL, cluster = NULL,
      keyword = c("lhp", "rhp", "udi", "udo", "ref", "cef",
                  "lhp.fo", "rhp.fo", "udi.fo", "udo.fo"),
      ...)
```

Arguments

`A`	a 'complex/real' matrix, dim = c(N, N).
`B`	a 'complex/real' matrix, dim = c(N, N).
`cluster`	specifies the eigenvalues in the selected cluster.
`keyword`	as similarly used in MATLAB.
`...`	options to `qz.*` functions.

Details

Either `cluster` or `keyword` should be specified.

`cluster` actually is the same as `select` in all `qz.*` functions.

`keywork` actually is similar as MATLAB.

keyword	Selected Region
'lhp'	Left-half plane (real(E) < 0)
'rhp'	Right-half plane (real(E) >= 0)
'udi'	Interior of unit disk (abs(E) < 1)
'udo'	Exterior of unit disk (abs(E) >= 1)
'ref'	Real eigenvalues first (top-left conner)
'cef'	Complex eigenvalues first (top-left conner)
'lhp'	Left-half plane (real(E) < 0) and finite only
'rhp'	Right-half plane (real(E) >= 0) and finite only
'udi'	Interior of unit disk (abs(E) < 1) and finite only
'udo'	Exterior of unit disk (abs(E) >= 1) and finite only

Value

Returns a list from the call.

Author(s)

Wei-Chen Chen <`wccsnow@gmail.com`>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

`qz`, `geigen`.

Examples

```
library(QZ, quiet = TRUE)

# Reordering eigenvalues
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "lhp"))
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "rhp"))
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "udi"))
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "udo"))
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "ref"))
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "cef"))
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "lhp.fo"))
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "rhp.fo"))
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "udi.fo"))
```

```
(ret <- ordqz(exAB1$A, exAB1$B, keyword = "udo.fo"))
```

---

qz.dgees *QZ Decomposition for a Real Matrix*

Description

This function call 'dgees' in Fortran to decompose a 'real' matrix A.

Usage

```
qz.dgees(A, vs = TRUE, LWORK = NULL)
```

Arguments

A	a 'real' matrix, dim = c(N, N).
vs	if compute 'real' Schur vectors. (Q)
LWORK	optional, dimension of array WORK for workspace. (>= 3N)

Details

See 'dgees.f' for all details.

DGEES computes for an N-by-N real non-symmetric matrix A, the eigenvalues, the real Schur form T, and, optionally, the matrix of Schur vectors Q. This gives the Schur factorization A = Q*T*(Q**T).

Optionally, it also orders the eigenvalues on the diagonal of the real Schur form so that selected eigenvalues are at the top left. The leading columns of Q then form an orthonormal basis for the invariant subspace corresponding to the selected eigenvalues.

A matrix is in real Schur form if it is upper quasi-triangular with 1-by-1 and 2-by-2 blocks. 2-by-2 blocks will be standardized in the form

[a b] [c a]

where b*c < 0. The eigenvalues of such a block are a +- sqrt(bc).

Value

Return a list contains next:

'T'	A's generalized Schur form.
'WR'	original returns from 'dgees.f'.
'WI'	original returns from 'dgees.f'.
'VS'	original returns from 'dgees.f'.
'WORK'	optimal LWORK (for dgees.f only)

'INFO' = 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. <= N: QZ iteration failed. =N+1: reordering problem. =N+2: reordering failed.

Extra returns in the list:

'W' WR + WI * i.

'Q' the Schur vectors.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/double/dgees.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.dgeev

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node89.html
A <- exA2$A
ret <- qz.dgees(A)

# Verify 1
A.new <- ret$Q %*% ret$T %*% solve(ret$Q)
round(A - A.new)

# verify 2
round(ret$Q %*% solve(ret$Q))
```

---

qz.dgeev *Generalized Eigenvalues Decomposition for a Real Matrix*

Description

This function call 'dgeev' in Fortran to decompose a 'real' matrix A.

Usage

```
qz.dgeev(A, vl = TRUE, vr = TRUE, LWORK = NULL)
```

Arguments

`A`	a 'real' matrix, dim = c(N, N).
`vl`	if compute left 'real' eigen vector. (U)
`vr`	if compute right 'real' eigen vector. (V)
`LWORK`	optional, dimension of array WORK for workspace. (>= 4N)

Details

See 'dgeev.f' for all details.

DGEEV computes for an N-by-N real non-symmetric matrix A, the eigenvalues and, optionally, the left and/or right eigenvectors.

The right eigenvector v(j) of A satisfies

A * v(j) = lambda(j) * v(j)

where lambda(j) is its eigenvalue. The left eigenvector u(j) of A satisfies

u(j)**T * A = lambda(j) * u(j)**T

where u(j)**T denotes the transpose of u(j).

The computed eigenvectors are normalized to have Euclidean norm equal to 1 and largest component real.

Value

Return a list contains next:

`'WR'`	original returns from 'dgeev.f'.
`'WI'`	original returns from 'dgeev.f'.
`'VL'`	original returns from 'dgeev.f'.
`'VR'`	original returns from 'dgeev.f'.
`'WORK'`	optimal LWORK (for dgeev.f only)
`'INFO'`	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. > 0: QZ iteration failed.

Extra returns in the list:

`'W'`	WR + WI * i.
`'U'`	the left eigen vectors.
`'V'`	the right eigen vectors.

If WI[j] is zero, then the j-th eigenvalue is real; if positive, then the j-th and (j+1)-st eigenvalues are a complex conjugate pair, with WI[j+1] negative.

If the j-th eigenvalue is real, then U[, j] = VL[, j], the j-th column of VL. If the j-th and (j+1)-th eigenvalues form a complex conjugate pair, then U[, j] = VL[, j] + i * VL[, j+1] and U[, j+1] = VL[, j] - i * VL[, j+1].

Similarly, for the right eigenvectors of V and VR.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/double/dgeev.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.dgees

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node87.html
A <- exA2$A
ret <- qz.dgeev(A)

# Verify 1
diff.R <- A %*% ret$V - matrix(ret$W, 4, 4, byrow = TRUE) * ret$V
diff.L <- t(ret$U) %*% A - matrix(ret$W, 4, 4) * t(ret$U)
round(diff.R)
round(diff.L)

# Verify 2
round(ret$U %*% solve(ret$U))
round(ret$V %*% solve(ret$V))
```

---

qz.dgges *QZ Decomposition for Real Paired Matrices*

Description

This function call 'dgges' in Fortran to decompose 'real' matrices (A,B).

Usage

```
qz.dgges(A, B, vsl = TRUE, vsr = TRUE, LWORK = NULL)
```

Arguments

A	a 'real' matrix, dim = c(N, N).
B	a 'real' matrix, dim = c(N, N).
vsl	if compute left 'real' Schur vectors. (Q)
vsr	if compute right 'real' Schur vectors. (Z)
LWORK	optional, dimension of array WORK for workspace. (>= 8N+16)

Details

See 'dgges.f' for all details.

DGGES computes for a pair of N-by-N real non-symmetric matrices (A,B), the generalized eigenvalues, the generalized real Schur form (S,T), optionally, the left and/or right matrices of Schur vectors (VSL and VSR). This gives the generalized Schur factorization

(A,B) = ((VSL)*S*(VSR)**T, (VSL)*T*(VSR)**T)

Optionally, it also orders the eigenvalues so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the upper quasi-triangular matrix S and the upper triangular matrix T.The leading columns of VSL and VSR then form an orthonormal basis for the corresponding left and right eigenspaces (deflating subspaces).

(If only the generalized eigenvalues are needed, use the driver DGGEV instead, which is faster.)

A generalized eigenvalue for a pair of matrices (A,B) is a scalar w or a ratio alpha/beta = w, such that A - w*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0 or both being zero.

A pair of matrices (S,T) is in generalized real Schur form if T is upper triangular with non-negative diagonal and S is block upper triangular with 1-by-1 and 2-by-2 blocks. 1-by-1 blocks correspond to real generalized eigenvalues, while 2-by-2 blocks of S will be "standardized" by making the corresponding elements of T have the form:

[a 0]
[0 b]

and the pair of corresponding 2-by-2 blocks in S and T will have a complex conjugate pair of generalized eigenvalues.

Value

Return a list contains next:

`'S'`	A's generalized Schur form.
`'T'`	B's generalized Schur form.
`'ALPHAR'`	original returns from 'dgges.f'.
`'ALPHAI'`	original returns from 'dgges.f'.
`'BETA'`	original returns from 'dgges.f'.
`'VSL'`	original returns from 'dgges.f'.
`'VSR'`	original returns from 'dgges.f'.
`'WORK'`	optimal LWORK (for dgges.f only)
`'INFO'`	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. =1,...,N: QZ iteration failed. =N+1: other than QZ iteration failed in DHGEQZ. =N+2: reordering problem. =N+3: reordering failed.

Extra returns in the list:

`'ALPHA'`	ALPHAR + ALPHAI * i.
`'Q'`	the left Schur vectors.
`'Z'`	the right Schur vectors.

The ALPHA[j]/BETA[j] are generalized eigenvalues.

If ALPHAI[j] is zero, then the j-th eigenvalue is real; if positive, then the j-th and (j+1)-st eigenvalues are a complex conjugate pair, with ALPHAI[j+1] negative.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/double/dgges.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.dggev

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node119.html
A <- exAB2$A
B <- exAB2$B
ret <- qz.dgges(A, B)

# Verify 1
A.new <- ret$Q %*% ret$S %*% t(ret$Z)
B.new <- ret$Q %*% ret$T %*% t(ret$Z)
round(A - A.new)
round(B - B.new)

# verify 2
round(ret$Q %*% t(ret$Q))
round(ret$Z %*% t(ret$Z))
```

---

qz.dggev *Generalized Eigenvalues Decomposition for Real Paired Matrices*

Description

This function call 'dggev' in Fortran to decompose 'real' matrices (A,B).

Usage

```
qz.dggev(A, B, vl = TRUE, vr = TRUE, LWORK = NULL)
```

Arguments

`A`	a 'real' matrix, dim = c(N, N).
`B`	a 'real' matrix, dim = c(N, N).
`vl`	if compute left 'real' eigen vector. (U)
`vr`	if compute right 'real' eigen vector. (V)
`LWORK`	optional, dimension of array WORK for workspace. (>= 8N)

Details

See 'dggev.f' for all details.

DGGEV computes for a pair of N-by-N real non-symmetric matrices (A,B) the generalized eigenvalues, and optionally, the left and/or right generalized eigenvectors.

A generalized eigenvalue for a pair of matrices (A,B) is a scalar lambda or a ratio alpha/beta = lambda, such that A - lambda*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0, and even for both being zero.

The right eigenvector v(j) corresponding to the eigenvalue lambda(j) of (A,B) satisfies

A * v(j) = lambda(j) * B * v(j).

The left eigenvector u(j) corresponding to the eigenvalue lambda(j) of (A,B) satisfies

u(j)**H * A = lambda(j) * u(j)**H * B .

where u(j)**H is the conjugate-transpose of u(j).

Value

Return a list contains next:

`'ALPHAR'`	original returns from 'dggev.f'.
`'ALPHAI'`	original returns from 'dggev.f'.
`'BETA'`	original returns from 'dggev.f'.
`'VL'`	original returns from 'dggev.f'.
`'VR'`	original returns from 'dggev.f'.
`'WORK'`	optimal LWORK (for dggev.f only)
`'INFO'`	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. =1,...,N: QZ iteration failed. =N+1: other than QZ iteration failed in DHGEQZ. =N+2: reordering problem. =N+3: reordering failed.

Extra returns in the list:

`'ALPHA'`	ALPHAR + ALPHAI * i.
`'U'`	the left eigen vectors.
`'V'`	the right eigen vectors.

If ALPHAI[j] is zero, then the j-th eigenvalue is real; if positive, then the j-th and (j+1)-st eigenvalues are a complex conjugate pair, with ALPHAI[j+1] negative.

If the j-th eigenvalue is real, then U[, j] = VL[, j], the j-th column of VL. If the j-th and (j+1)-th eigenvalues form a complex conjugate pair, then U[, j] = VL[, j] + i * VL[, j+1] and U[, j+1] = VL[, j] - i * VL[, j+1]. Each eigenvector is scaled so the largest component has abs(real part) + abs(imag. part) = 1.

Similarly, for the right eigenvectors of V and VR.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/double/dggev.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.dgges

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node117.html
A <- exAB2$A
B <- exAB2$B
ret <- qz.dggev(A, B)

# Verify
(lambda <- ret$ALPHA / ret$BETA)    # Unstable
diff.R <- matrix(ret$BETA, 4, 4, byrow = TRUE) * A %*% ret$V -
          matrix(ret$ALPHA, 4, 4, byrow = TRUE) * B %*% ret$V
diff.L <- matrix(ret$BETA, 4, 4) * H(ret$U) %*% A -
          matrix(ret$ALPHA, 4, 4) * H(ret$U) %*% B
round(diff.R)
round(diff.L)

# Verify 2
round(ret$U %*% solve(ret$U))
round(ret$V %*% solve(ret$V))
```

qz.dtgsen *Reordered QZ Decomposition for Real Paired Matrices*

Description

This function call 'dtgsend' in Fortran to reorder 'double' matrices (S,T,Q,Z).

Usage

```
qz.dtgsen(S, T, Q, Z, select, ijob = 4L,
          want.Q = TRUE, want.Z = TRUE, LWORK = NULL, LIWORK = NULL)
```

Arguments

S	a 'double' generalized Schur form, dim = c(N, N).
T	a 'double' generalized Schur form, dim = c(N, N).
Q	a 'double' left Schur vectors, dim = c(N, N).
Z	a 'double' right Schur vectors, dim = c(N, N).
select	specifies the eigenvalues in the selected cluster.
ijob	specifies whether condition numbers are required for the cluster of eigenvalues (PL and PR) or the deflating subspaces (Difu and Difl).
want.Q	if update Q.
want.Z	if update Z.
LWORK	optional, dimension of array WORK for workspace. (>= max(4N+16, N(N+1)))
LIWORK	optional, dimension of array IWORK for workspace. (>= max(N+6, N(N+1)/2))

Details

See 'dtgsen.f' for all details.

DTGSEN reorders the generalized real Schur decomposition of a real matrix pair (S,T) (in terms of an orthonormal equivalence transformation Q**T * (S,T) * Z), so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the upper quasi-triangular matrix S and the upper triangular T. The leading columns of Q and Z form orthonormal bases of the corresponding left and right eigenspaces (deflating subspaces). (S,T) must be in generalized real Schur canonical form (as returned by DGGES), i.e. S is block upper triangular with 1-by-1 and 2-by-2 diagonal blocks. T is upper triangular.

Note for 'ijob':
=0: Only reorder w.r.t. SELECT. No extras.
=1: Reciprocal of norms of "projections" onto left and right eigenspaces w.r.t. the selected cluster (PL and PR).
=2: Upper bounds on Difu and Difl. F-norm-based estimate (DIF(1:2)).
=3: Estimate of Difu and Difl. 1-norm-based estimate (DIF(1:2)). About 5 times as expensive as ijob = 2.
=4: Compute PL, PR and DIF (i.e. 0, 1 and 2 above): Economic version to get it all.

=5: Compute PL, PR and DIF (i.e. 0, 1 and 3 above).

In short, if (A,B) = Q * (S,T) * Z**T from `qz.zgges` and input (S,T,Q,Z) to `qz.ztgsen` with appropriate `select` option, then it yields

(A,B) = Q_n * (S_n,T_n) * Z_n**T

where (S_n,T_n,Q_n,Z_n) is a new set of generalized Schur decomposition of (A,B) according to the `select`.

Value

Return a list contains next:

`'S'`	S's reorded generalized Schur form.
`'T'`	T's reorded generalized Schur form.
`'ALPHAR'`	original returns from 'dtgsen.f'.
`'ALPHAI'`	original returns from 'dtgsen.f'.
`'BETA'`	original returns from 'dtgsen.f'.
`'M'`	original returns from 'dtgsen.f'.
`'PL'`	original returns from 'dtgsen.f'.
`'PR'`	original returns from 'dtgsen.f'.
`'DIF'`	original returns from 'dtgsen.f'.
`'WORK'`	optimal LWORK (for dtgsen.f only)
`'IWORK'`	optimal LIWORK (for dtgsen.f only)
`'INFO'`	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. =1: reordering of (S,T) failed.

Extra returns in the list:

`'ALPHA'`	ALPHAR + ALPHAI * i.
`'Q'`	the reorded left Schur vectors.
`'Z'`	the reorded right Schur vectors.

Warning(s)

There is no format checking for `S`, `T`, `Q`, and `Z` which are usually returned by `qz.dgges`.

There is also no checking for `select` which is usually according to the returns of `qz.dggev`.

Author(s)

Wei-Chen Chen <`wccsnow@gmail.com`>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/double/dtgsen.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.zgges, qz.dgges, qz.ztgsen.

Examples

```
library(QZ, quiet = TRUE)

### https://support.nag.com/numeric/fl/nagdoc_fl23/xhtml/f08/f08ygf.xml
S <- exAB4$S
T <- exAB4$T
Q <- exAB4$Q
Z <- exAB4$Z
select <- c(FALSE, TRUE, TRUE, FALSE)
ret <- qz.dtgsen(S, T, Q, Z, select)

# Verify 1
S.new <- ret$Q %*% ret$S %*% t(ret$Z)
T.new <- ret$Q %*% ret$T %*% t(ret$Z)
round(S - S.new)
round(T - T.new)

# verify 2
round(ret$Q %*% t(ret$Q))
round(ret$Z %*% t(ret$Z))
```

---

qz.dtrsen *Reordered QZ Decomposition for a Real Matrix*

Description

This function call 'dtrsend' in Fortran to reorder 'double' matrices (T,Q).

Usage

```
qz.dtrsen(T, Q, select, job = c("B", "V", "E", "N"),
          want.Q = TRUE, LWORK = NULL, LIWORK = NULL)
```

Arguments

T	a 'double' generalized Schur form, dim = c(N, N).
Q	a 'double' Schur vectors, dim = c(N, N).
select	specifies the eigenvalues in the selected cluster.
job	Specifies whether condition numbers are required for the cluster of eigenvalues (S) or the invariant subspace (SEP).
want.Q	if update Q.
LWORK	optional, dimension of array WORK for workspace. (>= N(N+1)/2)
LIWORK	optional, dimension of array IWORK for workspace. (>= N(N+1)/4)

Details

See 'dtrsen.f' for all details.

DTRSEN reorders the real Schur factorization of a real matrix A = Q*T*Q**T, so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the upper quasi-triangular matrix T, and the leading columns of Q form an orthonormal basis of the corresponding right invariant subspace.

Optionally the routine computes the reciprocal condition numbers of the cluster of eigenvalues and/or the invariant subspace.

T must be in Schur canonical form (as returned by DHSEQR), that is, block upper triangular with 1-by-1 and 2-by-2 diagonal blocks; each 2-by-2 diagonal block has its diagonal elements equal and its off-diagonal elements of opposite sign.

Value

Return a list contains next:

'T'	T's reorded generalized Schur form.
'WR'	original returns from 'dtrsen.f'.
'WI'	original returns from 'dtrsen.f'.
'M'	original returns from 'dtrsen.f'.
'S'	original returns from 'dtrsen.f'.
'SEP'	original returns from 'dtrsen.f'.
'WORK'	optimal LWORK (for dtrsen.f only)
'IWORK'	optimal LIWORK (for dtrsen.f only)
'INFO'	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. =1: reordering of T failed.

Extra returns in the list:

'W'	WR + WI * i.
'Q'	the reorded Schur vectors.

Warning(s)

There is no format checking for `T` and `Q` which are usually returned by `qz.dgees`.

There is also no checking for `select` which is usually according to the returns of `qz.dgeev`.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/double/dtrsen.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.zgees, qz.dgees, qz.ztrsen.

Examples

```
library(QZ, quiet = TRUE)

### https://support.nag.com/numeric/fl/nagdoc_fl22/xhtml/f08/f08qgf.xml
T <- exA4$T
Q <- exA4$Q
select <- c(TRUE, FALSE, FALSE, TRUE)
ret <- qz.dtrsen(T, Q, select)

# Verify 1
A <- Q %*% T %*% solve(Q)
A.new <- ret$Q %*% ret$T %*% solve(ret$Q)
round(A - A.new)

# verify 2
round(ret$Q %*% t(ret$Q))
```

---

qz.zgees — *QZ Decomposition for a Complex Matrix*

Description

This function call 'zgees' in Fortran to decompose a 'complex' matrix A.

Usage

```
qz.zgees(A, vs = TRUE, LWORK = NULL)
```

Arguments

A	a 'complex' matrix, dim = c(N, N).
vs	if compute 'complex' Schur vectors. (Q)
LWORK	optional, dimension of array WORK for workspace. (>= 2N)

Details

See 'zgees.f' for all details.

ZGEES computes for an N-by-N complex non-symmetric matrix A, the eigenvalues, the Schur form T, and, optionally, the matrix of Schur vectors Q. This gives the Schur factorization A = Q*T*(Q**H).

Optionally, it also orders the eigenvalues on the diagonal of the Schur form so that selected eigenvalues are at the top left. The leading columns of Q then form an orthonormal basis for the invariant subspace corresponding to the selected eigenvalues.

A complex matrix is in Schur form if it is upper triangular.

Value

Return a list contains next:

'T'	A's generalized Schur form.
'W'	generalized eigenvalues.
'VS'	original returns from 'zgees.f'.
'WORK'	optimal LWORK (for zgees.f only)
'INFO'	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. =1,...,N: QZ iteration failed. =N+1: reordering problem. =N+2: reordering failed.

Extra returns in the list:

'Q'	the Schur vectors.

Warning(s)

The results may not be consistent on 32 bits and 64 bits Windows systems, but may be valid on both systems.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/complex16/zgees.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.zgeev

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node94.html
A <- exA1$A
ret <- qz.zgees(A)

# Verify 1
A.new <- ret$Q %*% ret$T %*% H(ret$Q)
round(A - A.new)

# verify 2
round(ret$Q %*% H(ret$Q))
```

qz.zgeev *Generalized Eigenvalues Decomposition for a Complex Matrix*

Description

This function call 'zgeev' in Fortran to decompose a 'complex' matrix A.

Usage

```
qz.zgeev(A, vl = TRUE, vr = TRUE, LWORK = NULL)
```

Arguments

`A`	a 'complex' matrix, dim = c(N, N).
`vl`	if compute left 'complex' eigen vectors. (U)
`vr`	if compute right 'complex' eigen vectors. (V)
`LWORK`	optional, dimension of array WORK for workspace. (>= 2N)

Details

See 'zgeev.f' for all details.

ZGEEV computes for an N-by-N complex non-symmetric matrix A, the eigenvalues and, optionally, the left and/or right eigenvectors.

The right eigenvector v(j) of A satisfies

A * v(j) = lambda(j) * v(j)

where lambda(j) is its eigenvalue. The left eigenvector u(j) of A satisfies

u(j)**H * A = lambda(j) * u(j)**H

where u(j)**H denotes the conjugate transpose of u(j).

The computed eigenvectors are normalized to have Euclidean norm equal to 1 and largest component real.

Value

Return a list contains next:

`'W'`	original returns from 'zgeev.f'.
`'VL'`	original returns from 'zgeev.f'.
`'VR'`	original returns from 'zgeev.f'.
`'WORK'`	optimal LWORK (for zgeev.f only)
`'INFO'`	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. > 0: QZ iteration failed.

Extra returns in the list:

`'U'`	the left eigen vectors.
`'V'`	the right eigen vectors.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/complex16/zgeev.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.zgees

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node92.html
A <- exA1$A
ret <- qz.zgeev(A)

# Verify 1
diff.R <- A %*% ret$V - matrix(ret$W, 4, 4, byrow = TRUE) * ret$V
diff.L <- H(ret$U) %*% A - matrix(ret$W, 4, 4) * H(ret$U)
round(diff.R)
round(diff.L)

# Verify 2
round(ret$U %*% H(ret$U))
round(ret$V %*% H(ret$V))
```

---

qz.zgges   *QZ Decomposition for Complex Paired Matrices*

Description

This function call 'zgges' in Fortran to decompose 'complex' matrices (A,B).

Usage

```
qz.zgges(A, B, vsl = TRUE, vsr = TRUE, LWORK = NULL)
```

Arguments

A	a 'complex' matrix, dim = c(N, N).
B	a 'complex' matrix, dim = c(N, N).
vsl	if compute left 'complex' Schur vectors. (Q)
vsr	if compute right 'complex' Schur vectors. (Z)
LWORK	optional, dimension of array WORK for workspace. (>= 2N)

Details

See 'zgges.f' for all details.

ZGGES computes for a pair of N-by-N complex non-symmetric matrices (A,B), the generalized eigenvalues, the generalized complex Schur form (S, T), and optionally left and/or right Schur vectors (VSL and VSR). This gives the generalized Schur factorization

(A,B) = ((VSL)*S*(VSR)**H, (VSL)*T*(VSR)**H)

where (VSR)**H is the conjugate-transpose of VSR.

Optionally, it also orders the eigenvalues so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the upper triangular matrix S and the upper triangular matrix T. The leading columns of VSL and VSR then form an unitary basis for the corresponding left and right eigenspaces (deflating subspaces).

(If only the generalized eigenvalues are needed, use the driver ZGGEV instead, which is faster.)

A generalized eigenvalue for a pair of matrices (A,B) is a scalar w or a ratio alpha/beta = w, such that A - w*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0, and even for both being zero.

A pair of matrices (S,T) is in generalized complex Schur form if S and T are upper triangular and, in addition, the diagonal elements of T are non-negative real numbers.

Value

Return a list contains next:

`'S'`	A's generalized Schur form.
`'T'`	B's generalized Schur form.
`'ALPHA'`	ALPHA[j]/BETA[j] are generalized eigenvalues.
`'BETA'`	ALPHA[j]/BETA[j] are generalized eigenvalues.
`'VSL'`	original returns from 'zgges.f'.
`'VSR'`	original returns from 'zgges.f'.
`'WORK'`	optimal LWORK (for zgges.f only)
`'INFO'`	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. =1,...,N: QZ iteration failed. =N+1: other than QZ iteration failed in ZHGEQZ. =N+2: reordering problem. =N+3: reordering failed.

Extra returns in the list:

`'Q'`	the left Schur vectors.
`'Z'`	the right Schur vectors.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/complex16/zgges.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.zggev

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node124.html
A <- exAB1$A
B <- exAB1$B
ret <- qz.zgges(A, B)

# Verify 1
A.new <- ret$Q %*% ret$S %*% H(ret$Z)
B.new <- ret$Q %*% ret$T %*% H(ret$Z)
round(A - A.new)
round(B - B.new)

# verify 2
round(ret$Q %*% H(ret$Q))
round(ret$Z %*% H(ret$Z))
```

---

qz.zggev *Generalized Eigenvalues Decomposition for Complex Paired Matrices*

Description

This function call 'zggev' in Fortran to decompose 'complex' matrices (A,B).

Usage

```
qz.zggev(A, B, vl = TRUE, vr = TRUE, LWORK = NULL)
```

Arguments

A	a 'complex' matrix, dim = c(N, N).
B	a 'complex' matrix, dim = c(N, N).
vl	if compute left 'complex' eigen vectors. (U)
vr	if compute right 'complex' eigen vectors. (V)
LWORK	optional, dimension of array WORK for workspace. (>= 2N)

Details

See 'zggev.f' for all details.

ZGGEV computes for a pair of N-by-N complex non-symmetric matrices (A,B), the generalized eigenvalues, and optionally, the left and/or right generalized eigenvectors.

A generalized eigenvalue for a pair of matrices (A,B) is a scalar lambda or a ratio alpha/beta = lambda, such that A - lambda*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0, and even for both being zero.

The right generalized eigenvector v(j) corresponding to the generalized eigenvalue lambda(j) of (A,B) satisfies

A * v(j) = lambda(j) * B * v(j).

The left generalized eigenvector u(j) corresponding to the generalized eigenvalues lambda(j) of (A,B) satisfies

u(j)**H * A = lambda(j) * u(j)**H * B

where u(j)**H is the conjugate-transpose of u(j).

Value

Return a list contains next:

'ALPHA'	original returns from 'zggev.f'.
'BETA'	original returns from 'zggev.f'.
'VL'	original returns from 'zggev.f'.
'VR'	original returns from 'zggev.f'.
'WORK'	optimal LWORK (for zggev.f only)
'INFO'	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. =1,...,N: QZ iteration failed. =N+1: other than QZ iteration failed in ZHGEQZ. =N+2: reordering problem. =N+3: reordering failed.

Extra returns in the list:

'U'	the left eigen vectors.
'V'	the right eigen vectors.

Note that 'VL' and 'VR' are scaled so the largest component has abs(real part) + abs(imag. part) = 1.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/complex16/zggev.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.zgges

Examples

```
library(QZ, quiet = TRUE)

### https://www.netlib.org/lapack/lug/node122.html
A <- exAB1$A
B <- exAB1$B
ret <- qz.zggev(A, B)

# Verify 1
(lambda <- ret$ALPHA / ret$BETA)    # Unstable
diff.R <- matrix(ret$BETA, 4, 4, byrow = TRUE) * A %*% ret$V -
          matrix(ret$ALPHA, 4, 4, byrow = TRUE) * B %*% ret$V
diff.L <- matrix(ret$BETA, 4, 4) * H(ret$U) %*% A -
          matrix(ret$ALPHA, 4, 4) * H(ret$U) %*% B
round(diff.R)
round(diff.L)

# Verify 2
round(ret$U %*% solve(ret$U))
round(ret$V %*% solve(ret$V))
```

---

qz.ztgsen *Reordered QZ Decomposition for Complex Paired Matrices*

Description

This function call 'ztgsend' in Fortran to reorder 'complex' matrices (S,T,Q,Z).

Usage

```
qz.ztgsen(S, T, Q, Z, select, ijob = 4L,
          want.Q = TRUE, want.Z = TRUE, LWORK = NULL, LIWORK = NULL)
```

Arguments

S	a 'complex' generalized Schur form, dim = c(N, N).
T	a 'complex' generalized Schur form, dim = c(N, N).
Q	a 'complex' left Schur vectors, dim = c(N, N).
Z	a 'complex' right Schur vectors, dim = c(N, N).
select	specifies the eigenvalues in the selected cluster.
ijob	specifies whether condition numbers are required for the cluster of eigenvalues (PL and PR) or the deflating subspaces (Difu and Difl).
want.Q	if update Q.

want.Z	if update Z.
LWORK	optional, dimension of array WORK for workspace. (>= N(N+1))
LIWORK	optional, dimension of array IWORK for workspace. (>= max(N+2, N(N+1)/2))

Details

See 'ztgsen.f' for all details.

ZTGSEN reorders the generalized Schur decomposition of a complex matrix pair (S,T) (in terms of an unitary equivalence transformation Q**H * (S,T) * Z), so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the pair (S,T). The leading columns of Q and Z form unitary bases of the corresponding left and right eigenspaces (deflating subspaces). (S,T) must be in generalized Schur canonical form, that is, S and T are both upper triangular.

ZTGSEN also computes the generalized eigenvalues

w(j)= ALPHA(j) / BETA(j)

of the reordered matrix pair (S,T).

Note for 'ijob':
=0: Only reorder w.r.t. SELECT. No extras.
=1: Reciprocal of norms of "projections" onto left and right eigenspaces w.r.t. the selected cluster (PL and PR).
=2: Upper bounds on Difu and Difl. F-norm-based estimate (DIF(1:2)).
=3: Estimate of Difu and Difl. 1-norm-based estimate (DIF(1:2)). About 5 times as expensive as ijob = 2.
=4: Compute PL, PR and DIF (i.e. 0, 1 and 2 above): Economic version to get it all.
=5: Compute PL, PR and DIF (i.e. 0, 1 and 3 above).

In short, if (A,B) = Q * (S,T) * Z**H from `qz.zgges` and input (S,T,Q,Z) to `qz.ztgsen` with appropriate `select` option, then it yields

(A,B) = Q_n * (S_n,T_n) * Z_n**H

where (S_n,T_n,Q_n,Z_n) is a new set of generalized Schur decomposition of (A,B) according to the `select`.

Value

Return a list contains next:

'S'	S's reorded generalized Schur form.
'T'	T's reorded generalized Schur form.
'ALPHA'	ALPHA[j]/BETA[j] are generalized eigenvalues.
'BETA'	ALPHA[j]/BETA[j] are generalized eigenvalues.
'M'	original returns from 'ztgsen.f'.
'PL'	original returns from 'ztgsen.f'.
'PR'	original returns from 'ztgsen.f'.
'DIF'	original returns from 'ztgsen.f'.
'WORK'	optimal LWORK (for ztgsen.f only)

'IWORK' optimal LIWORK (for ztgsen.f only)

'INFO' = 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value. =1: reordering of (S,T) failed.

Extra returns in the list:

'Q' the reorded left Schur vectors.

'Z' the reorded right Schur vectors.

Warning(s)

There is no format checking for S, T, Q, and Z which are usually returned by qz.zgges.

There is also no checking for select which is usually according to the returns of qz.zggev.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/complex16/ztgsen.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

qz.zgges, qz.dgges, qz.dtgsen.

Examples

```
library(QZ, quiet = TRUE)

### https://support.nag.com/numeric/fl/nagdoc_fl23/xhtml/f08/f08yuf.xml
S <- exAB3$S
T <- exAB3$T
Q <- exAB3$Q
Z <- exAB3$Z
select <- c(FALSE, TRUE, TRUE, FALSE)
ret <- qz.ztgsen(S, T, Q, Z, select)

# Verify 1
S.new <- ret$Q %*% ret$S %*% H(ret$Z)
T.new <- ret$Q %*% ret$T %*% H(ret$Z)
round(S - S.new)
round(T - T.new)

# verify 2
round(ret$Q %*% H(ret$Q))
round(ret$Z %*% H(ret$Z))
```

`qz.ztrsen` *Reordered QZ Decomposition for a Complex Matrix*

Description

This function call 'ztrsend' in Fortran to reorder 'complex' matrix (T,Q).

Usage

```
qz.ztrsen(T, Q, select, job = c("B", "V", "E", "N"),
          want.Q = TRUE, LWORK = NULL)
```

Arguments

`T`	a 'complex' generalized Schur form, dim = c(N, N).
`Q`	a 'complex' Schur vectors, dim = c(N, N).
`select`	specifies the eigenvalues in the selected cluster.
`job`	Specifies whether condition numbers are required for the cluster of eigenvalues (S) or the invariant subspace (SEP).
`want.Q`	if update Q.
`LWORK`	optional, dimension of array WORK for workspace. (>= N(N+1)/2)

Details

See 'ztrsen.f' for all details.

ZTRSEN reorders the Schur factorization of a complex matrix A = Q*T*Q**H, so that a selected cluster of eigenvalues appears in the leading positions on the diagonal of the upper triangular matrix T, and the leading columns of Q form an orthonormal basis of the corresponding right invariant subspace.

Optionally the routine computes the reciprocal condition numbers of the cluster of eigenvalues and/or the invariant subspace.

Value

Return a list contains next:

`'T'`	T's reorded generalized Schur form.
`'W'`	generalized eigenvalues.
`'M'`	original returns from 'ztrsen.f'.
`'S'`	original returns from 'ztrsen.f'.
`'SEP'`	original returns from 'ztrsen.f'.
`'WORK'`	optimal LWORK (for ztrsen.f only)
`'INFO'`	= 0: successful. < 0: if INFO = -i, the i-th argument had an illegal value.

Extra returns in the list:

`'Q'`	the reorded Schur vectors.

Warning(s)

There is no format checking for `T` and `Q` which are usually returned by `qz.zgees`.

There is also no checking for `select` which is usually according to the returns of `qz.zgeev`.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>

References

Anderson, E., et al. (1999) *LAPACK User's Guide,* 3rd edition, SIAM, Philadelphia.

https://www.netlib.org/lapack/complex16/ztrsen.f

https://en.wikipedia.org/wiki/Schur_decomposition

See Also

`qz.zgees`, `qz.dgees`, `qz.dtrsen`.

Examples

```
library(QZ, quiet = TRUE)

### https://support.nag.com/numeric/fl/nagdoc_fl23/xhtml/f08/f08quf.xml
T <- exA3$T
Q <- exA3$Q
select <- c(TRUE, FALSE, FALSE, TRUE)
ret <- qz.ztrsen(T, Q, select)

# Verify 1
A <- Q %*% T %*% solve(Q)
A.new <- ret$Q %*% ret$T %*% solve(ret$Q)
round(A - A.new)

# verify 2
round(ret$Q %*% solve(ret$Q))
```

Index