

NAG C Library Function Document

nag_zggbak (f08wwc)

1 Purpose

nag_zggbak (f08wwc) forms the right or left eigenvectors of the real generalized eigenvalue problem $Ax = \lambda Bx$, by backward transformation on the computed eigenvectors given by nag_ztgevc (f08yxc). It is necessary to call this function only if the optional balancing function nag_zggbal (f08wvc) was previously called to balance the matrix pair (A, B) .

2 Specification

```
void nag_zggbak (Nag_OrderType order, Nag_JobType job, Nag_SideType side, Integer n,
                Integer ilo, Integer ihi, const double lscale[], const double rscale[], Integer m,
                Complex v[], Integer pdv, NagError *fail)
```

3 Description

If the matrix pair has been previously balanced using the function nag_zggbal (f08wvc) then nag_zggbak (f08wwc) backtransforms the eigenvector solution given by nag_ztgevc (f08yxc). This is usually the sixth and last step in the solution of the generalized eigenvalue problem.

For a description of balancing, see the document for nag_zggbal (f08wvc).

4 References

Ward R C (1981) Balancing the generalized eigenvalue problem *SIAM J. Sci. Stat. Comp.* **2** 141–152

5 Parameters

- 1: **order** – Nag_OrderType *Input*
On entry: the **order** parameter specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by **order** = **Nag_RowMajor**. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.
Constraint: **order** = **Nag_RowMajor** or **Nag_ColMajor**.
- 2: **job** – Nag_JobType *Input*
On entry: specifies the backtransformation step required:
 - if **job** = **Nag_DoNothing**, no transformations are done;
 - if **job** = **Nag_Permute**, only do backward transformations based on permutations;
 - if **job** = **Nag_Scale**, only do backward transformations based on scaling;
 - if **job** = **Nag_DoBoth**, do backward transformations for both permutations and scaling.**Note:** this must be identical to the parameter **job** as supplied to nag_zggbal (f08wvc).
Constraint: **job** = **Nag_DoNothing**, **Nag_Permute**, **Nag_Scale** or **Nag_DoBoth**.
- 3: **side** – Nag_SideType *Input*
On entry: indicates whether left or right eigenvectors are to be transformed, as follows:

if **side** = **Nag_LeftSide**, left eigenvectors are transformed;

if **side** = **Nag_RightSide**, right eigenvectors are transformed.

Constraint: **side** = **Nag_LeftSide** or **Nag_RightSide**.

4: **n** – Integer *Input*

On entry: n , the order of the matrices A and B of the generalized eigenvalue problem.

Constraint: $n \geq 0$.

5: **ilo** – Integer *Input*

6: **ihi** – Integer *Input*

On entry: i_{lo} and i_{hi} as determined by a previous call to nag_zggbal (f08wvc).

Constraints:

if $n > 0$, $1 \leq ilo \leq ihi \leq n$;

if $n = 0$, $ilo = 1$ and $ihi = 0$.

7: **lscale**[dim] – const double *Input*

Note: the dimension, dim , of the array **lscale** must be at least $\max(1, n)$.

On entry: details of the permutations and scaling factors applied to the left side of the matrices A and B , as returned by a previous call to nag_zggbal (f08wvc).

8: **rscale**[dim] – const double *Input*

Note: the dimension, dim , of the array **rscale** must be at least $\max(1, n)$.

On entry: details of the permutations and scaling factors applied to the right side of the matrices A and B , as returned by a previous call to nag_zggbal (f08wvc).

9: **m** – Integer *Input*

On entry: m , the required number of left or right eigenvectors.

Constraint: $0 \leq m \leq n$.

10: **v**[dim] – Complex *Input/Output*

Note: the dimension, dim , of the array **v** must be at least $\max(1, pdv \times m)$ when **order** = **Nag_ColMajor** and at least $\max(1, pdv \times n)$ when **order** = **Nag_RowMajor**.

If **order** = **Nag_ColMajor**, the (i, j) th element of the matrix V is stored in $v[(j-1) \times pdv + i - 1]$ and if **order** = **Nag_RowMajor**, the (i, j) th element of the matrix V is stored in $v[(i-1) \times pdv + j - 1]$.

On entry: the matrix of right or left eigenvectors, as returned by nag_zggbal (f08wvc).

On exit: the transformed right or left eigenvectors.

11: **pdv** – Integer *Input*

On entry: the stride separating matrix row or column elements (depending on the value of **order**) in the array **v**.

Constraints:

if **order** = **Nag_ColMajor**, $pdv \geq \max(1, n)$;

if **order** = **Nag_RowMajor**, $pdv \geq \max(1, m)$.

12: **fail** – NagError * *Output*

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **n** = $\langle value \rangle$.

Constraint: **n** ≥ 0 .

On entry, **pdv** = $\langle value \rangle$.

Constraint: **pdv** > 0 .

NE_INT_2

On entry, **m** = $\langle value \rangle$, **n** = $\langle value \rangle$.

Constraint: $0 \leq \mathbf{m} \leq \mathbf{n}$.

On entry, **pdv** = $\langle value \rangle$, **n** = $\langle value \rangle$.

Constraint: **pdv** $\geq \max(1, \mathbf{n})$.

On entry, **pdv** = $\langle value \rangle$, **m** = $\langle value \rangle$.

Constraint: **pdv** $\geq \max(1, \mathbf{m})$.

NE_INT_3

On entry, **n** = $\langle value \rangle$, **ilo** = $\langle value \rangle$, **ihi** = $\langle value \rangle$.

Constraint: if **n** > 0 , $1 \leq \mathbf{ilo} \leq \mathbf{ihi} \leq \mathbf{n}$;

if **n** = 0, **ilo** = 1 and **ihi** = 0.

NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter $\langle value \rangle$ had an illegal value.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

7 Accuracy

The errors are negligible.

8 Further Comments

The number of operations is proportional to n^2 .

The real analogue of this function is nag_dggbak (f08wjc).

9 Example

See Section 9 of the documents for nag_zhgeqz (f08xsc) and nag_ztgevc (f08yxc).