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

```c
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


5 Parameters

1: \textbf{order} – Nag_OrderType \hspace{5em} \textit{Input}

\textit{On entry}: the \textbf{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 \textbf{order} = Nag_RowMajor. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.

\textit{Constraint}: \textbf{order} = Nag_RowMajor or Nag_ColMajor.

2: \textbf{job} – Nag_JobType \hspace{5em} \textit{Input}

\textit{On entry}: specifies the backtransformation step required:

- \textbf{job} = Nag_DoNothing, no transformations are done;
- \textbf{job} = Nag_Permute, only do backward transformations based on permutations;
- \textbf{job} = Nag_Scale, only do backward transformations based on scaling;
- \textbf{job} = Nag_DoBoth, do backward transformations for both permutations and scaling.

\textit{Note}: this must be identical to the parameter \textbf{job} as supplied to nag_zggbal (f08wvc).

\textit{Constraint}: \textbf{job} = Nag_DoNothing, Nag_Permute, Nag_Scale or Nag_DoBoth.

3: \textbf{side} – Nag_SideType \hspace{5em} \textit{Input}

\textit{On entry}: indicates whether left or right eigenvectors are to be transformed, as follows:
if \( \text{side} = \text{Nag} \_\text{LeftSide} \), left eigenvectors are transformed;
if \( \text{side} = \text{Nag} \_\text{RightSide} \), right eigenvectors are transformed.

**Constraint:** \( \text{side} = \text{Nag} \_\text{LeftSide} \) or \( \text{Nag} \_\text{RightSide} \).

4: \( n \) – Integer \hspace{1cm} \text{Input}

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

*Constraint:* \( n \geq 0 \).

5: \( \text{ilo} \) – Integer \hspace{1cm} \text{Input}
6: \( \text{ihi} \) – Integer \hspace{1cm} \text{Input}

*On entry:* \( \text{ilo} \) and \( \text{ihi} \) as determined by a previous call to \text{nag}\_zggbal (f08wvc).

*Constraints:*

- if \( n > 0 \), \( 1 \leq \text{ilo} \leq \text{ihi} \leq n \);
- if \( n = 0 \), \( \text{ilo} = 1 \) and \( \text{ihi} = 0 \).

7: \( \text{lscale} \) – const double \hspace{1cm} \text{Input}

*Note:* the dimension, \( \text{dim} \), of the array \( \text{lscale} \) 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 \text{nag}\_zggbal (f08wvc).

8: \( \text{rscale} \) – const double \hspace{1cm} \text{Input}

*Note:* the dimension, \( \text{dim} \), of the array \( \text{rscale} \) 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 \text{nag}\_zggbal (f08wvc).

9: \( m \) – Integer \hspace{1cm} \text{Input}

*On entry:* \( m \), the required number of left or right eigenvectors.

*Constraint:* \( 0 \leq m \leq n \).

10: \( v \) – Complex \hspace{1cm} \text{Input/Output}

*Note:* the dimension, \( \text{dim} \), of the array \( v \) must be at least \( \max(1, \text{pdv} \times m) \) when \text{order} = \text{Nag} \_\text{ColMajor} \) and at least \( \max(1, \text{pdv} \times n) \) when \text{order} = \text{Nag} \_\text{RowMajor} \).

If \text{order} = \text{Nag} \_\text{ColMajor} \), the \( (i,j) \)th element of the matrix \( V \) is stored in \( v[(j - 1) \times \text{pdv} + i - 1] \) and if \text{order} = \text{Nag} \_\text{RowMajor} \), the \( (i,j) \)th element of the matrix \( V \) is stored in \( v[(i - 1) \times \text{pdv} + j - 1] \).

*On entry:* the matrix of right or left eigenvectors, as returned by \text{nag}\_zggbal (f08wvc).

*On exit:* the transformed right or left eigenvectors.

11: \( \text{pdv} \) – Integer \hspace{1cm} \text{Input}

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

*Constraints:*

- if \text{order} = \text{Nag} \_\text{ColMajor} \), \( \text{pdv} \geq \max(1, n) \);
- if \text{order} = \text{Nag} \_\text{RowMajor} \), \( \text{pdv} \geq \max(1, m) \).

12: \( \text{fail} \) – NagError * \hspace{1cm} \text{Output}

The NAG error parameter (see the Essential Introduction).
6 Error Indicators and Warnings

NE_INT
On entry, n = <value>.  
Constraint: n ≥ 0.

On entry, pdv = <value>.  
Constraint: pdv > 0.

NE_INT_2
On entry, m = <value>, n = <value>.  
Constraint: 0 ≤ m ≤ n.

On entry, pdv = <value>, n = <value>.  
Constraint: pdv ≥ max(1, n).

On entry, pdv = <value>, m = <value>.  
Constraint: pdv ≥ max(1, m).

NE_INT_3
On entry, n = <value>, ilo = <value>, ihi = <value>.  
Constraint: if n > 0, 1 ≤ ilo ≤ ihi ≤ n;  
if n = 0, ilo = 1 and ihi = 0.

NE_ALLOC_FAIL
Memory allocation failed.

NE_BAD_PARAM
On entry, parameter <value> 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).