NAG Library Function Document

nag_dormrz (f08bkc)

1 Purpose

nag_dormrz (f08bkc) multiplies a general real \( m \) by \( n \) matrix \( C \) by the real orthogonal matrix \( Z \) from an \( RZ \) factorization computed by nag_dtzrzf (f08bhc).

2 Specification

```c
#include <nag.h>
#include <nagf08.h>
void nag_dormrz (Nag_OrderType order, Nag_SideType side, 
    Nag_TransType trans, Integer m, Integer n, Integer k, Integer l, 
    const double a[], Integer pda, const double tau[], double c[], 
    Integer pdc, NagError *fail)
```

3 Description

nag_dormrz (f08bkc) is intended to be used following a call to nag_dtzrzf (f08bhc), which performs an \( RZ \) factorization of a real upper trapezoidal matrix \( A \) and represents the orthogonal matrix \( Z \) as a product of elementary reflectors.

This function may be used to form one of the matrix products

\[
ZC, \quad Z^TC, \quad CZ, \quad CZ^T,
\]

overwriting the result on \( C \), which may be any real rectangular \( m \) by \( n \) matrix.

4 References


5 Arguments

1: \textbf{order} – Nag_OrderType

*Input*

On entry: the \texttt{order} argument specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by \texttt{order} = Nag_RowMajor. See Section 3.2.1.3 in the Essential Introduction for a more detailed explanation of the use of this argument.

Constraint: \texttt{order} = Nag_RowMajor or Nag_ColMajor.

2: \textbf{side} – Nag_SideType

*Input*

On entry: indicates how \( Z \) or \( Z^T \) is to be applied to \( C \).

\texttt{side} = Nag_LeftSide

\( Z \) or \( Z^T \) is applied to \( C \) from the left.

\texttt{side} = Nag_RightSide

\( Z \) or \( Z^T \) is applied to \( C \) from the right.

Constraint: \texttt{side} = Nag_LeftSide or Nag_RightSide.
trans – Nag_TransType

On entry: indicates whether $Z$ or $Z^T$ is to be applied to $C$.

$\text{trans} = \text{Nag_NoTrans}$

$Z$ is applied to $C$.

$\text{trans} = \text{Nag_Trans}$

$Z^T$ is applied to $C$.

Constraint: $\text{trans} = \text{Nag_NoTrans}$ or $\text{Nag_Trans}$.

$m$ – Integer

On entry: $m$, the number of rows of the matrix $C$.

Constraint: $m \geq 0$.

$n$ – Integer

On entry: $n$, the number of columns of the matrix $C$.

Constraint: $n \geq 0$.

$k$ – Integer

On entry: $k$, the number of elementary reflectors whose product defines the matrix $Z$.

Constraints:

if $\text{side} = \text{Nag_LeftSide}$, $m \geq k \geq 0$;
if $\text{side} = \text{Nag_RightSide}$, $n \geq k \geq 0$.

$l$ – Integer

On entry: $l$, the number of columns of the matrix $A$ containing the meaningful part of the Householder reflectors.

Constraints:

if $\text{side} = \text{Nag_LeftSide}$, $m \geq l \geq 0$;
if $\text{side} = \text{Nag_RightSide}$, $n \geq l \geq 0$.

$a[\text{dim}]$ – const double

Input

Note: the dimension, $\text{dim}$, of the array $a$ must be at least

$\max(1, pda \times m)$ when $\text{side} = \text{Nag_LeftSide}$ and $\text{order} = \text{Nag_ColMajor}$;
$\max(1, k \times pda)$ when $\text{side} = \text{Nag_LeftSide}$ and $\text{order} = \text{Nag_RowMajor}$;
$\max(1, pda \times n)$ when $\text{side} = \text{Nag_RightSide}$ and $\text{order} = \text{Nag_ColMajor}$;
$\max(1, k \times pda)$ when $\text{side} = \text{Nag_RightSide}$ and $\text{order} = \text{Nag_RowMajor}$.

The $(i,j)$th element of the matrix $A$ is stored in

$a[(j-1) \times pda + i - 1]$ when $\text{order} = \text{Nag_ColMajor}$;
a$[(i-1) \times pda + j - 1]$ when $\text{order} = \text{Nag_RowMajor}$.

On entry: the $i$th row of $a$ must contain the vector which defines the elementary reflector $H_i$, for

$i = 1, 2, \ldots, k$, as returned by $\text{nag_dtzrzf (f08bhc)}$.

$pda$ – Integer

Input

On entry: the stride separating row or column elements (depending on the value of $\text{order}$) in the
array $a$. 
Constraints:
  if order = Nag_ColMajor, pda ≥ max(1, k);
  if order = Nag_RowMajor,
    if side = Nag_LeftSide, pda ≥ max(1, m);
    if side = Nag_RightSide, pda ≥ max(1, n).

10: \( \text{tau}[\text{dim}] \) – const double 
    \text{Input}

Note: the dimension, \( \text{dim} \), of the array tau must be at least max(1, k).

On entry: \( \text{tau}[i-1] \) must contain the scalar factor of the elementary reflector \( H_i \), as returned by nag_dtzrzf (f08bhc).

11: \( \text{c}[\text{dim}] \) – double 
    \text{Input/Output}

Note: the dimension, \( \text{dim} \), of the array c must be at least
  max(1, pdc \times n) when order = Nag_ColMajor;
  max(1, m \times pdc) when order = Nag_RowMajor.

The \((i,j)\)th element of the matrix \( C \) is stored in
  \( c[(j-1) \times pdc + i - 1] \) when order = Nag_ColMajor;
  \( c[(i-1) \times pdc + j - 1] \) when order = Nag_RowMajor.

On entry: the \( m \) by \( n \) matrix \( C \).

On exit: c is overwritten by \( ZC \) or \( Z^T C \) or \( CZ \) or \( Z^T C \) as specified by side and trans.

12: \( \text{pdc} \) – Integer 
    \text{Input}

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

Constraints:
  if order = Nag_ColMajor, pdc ≥ max(1, m);
  if order = Nag_RowMajor, pdc ≥ max(1, n).

13: \( \text{fail} \) – NagError * 
    \text{Input/Output}

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL
Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

NE_BAD_PARAM
On entry, argument \langle value \rangle had an illegal value.

NE_ENUM_INT_3
On entry, side = \langle value \rangle, m = \langle value \rangle, n = \langle value \rangle and k = \langle value \rangle.
Constraint: if side = Nag_LeftSide, m ≥ k ≥ 0;
if side = Nag_RightSide, n ≥ k ≥ 0.

On entry, side = \langle value \rangle, m = \langle value \rangle, n = \langle value \rangle and l = \langle value \rangle.
Constraint: if side = Nag_LeftSide, m ≥ l ≥ 0;
if side = Nag_RightSide, n ≥ l ≥ 0.
On entry, side = (value), pda = (value), m = (value) and n = (value).
Constraint: if side = Nag_LeftSide, pda ≥ max(1, m);
if side = Nag_RightSide, pda ≥ max(1, n).

NE_INT
On entry, m = (value).
Constraint: m ≥ 0.
On entry, n = (value).
Constraint: n ≥ 0.
On entry, pda = (value).
Constraint: pda > 0.
On entry, pdc = (value).
Constraint: pdc > 0.

NE_INT_2
On entry, pda = (value) and k = (value).
Constraint: pda ≥ max(1, k).
On entry, pdc = (value) and m = (value).
Constraint: pdc ≥ max(1, m).
On entry, pdc = (value) and n = (value).
Constraint: pdc ≥ max(1, n).

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 contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in the Essential Introduction for further information.

NE_NO LICENCE
Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

7 Accuracy
The computed result differs from the exact result by a matrix E such that
\[ ||E||_2 = O(\epsilon)||C||_2 \]
where \( \epsilon \) is the machine precision.

8 Parallelism and Performance
nag_dormrz (f08bkc) is not threaded by NAG in any implementation.
nag_dormrz (f08bkc) makes calls to BLAS and/or LAPACK routines, which may be threaded within the
vendor library used by this implementation. Consult the documentation for the vendor library for further
information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the
OpenMP environment used within this function. Please also consult the Users’ Note for your
implementation for any additional implementation-specific information.
9 Further Comments

The total number of floating-point operations is approximately $4nlk$ if $\text{side} = \text{Nag\_LeftSide}$ and $4mlk$ if $\text{side} = \text{Nag\_RightSide}$.

The complex analogue of this function is nag_zunmrz (f08bxc).

10 Example

See Section 10 in nag_dtzrzf (f08bhc).