F01ZYFP

NAG Parallel Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check for implementation-dependent details. You are advised to enclose any calls to NAG Parallel Library routines between calls to Z01AAFP and Z01ABFP.

1 Description

F01ZYFP generates and distributes a complex vector v of dimension n on a one-dimensional logical grid of processors.

It distributes the vector in the form required by a number of the routines in Chapter F07 (especially F07JRFP (PZPTTRF), F07JSFP (PZPTTRS) which are used for factorization and solving, complex tridiagonal systems of equations). A user-supplied subroutine is required to generate a block of the vector v on each processor.

2 Specification

```
SUBROUTINE F01ZYFP(GRHS, N, V, IDESCV, IFAIL)
COMPLEX*16         V(*)
INTEGER            N, IDESCV(*), IFAIL
EXTERNAL           GRHS
```

3 Usage

3.1 Definitions

The following definitions are used in describing the data distribution within this document:

p	–	$m_p \times n_p$, the total number of processors in the Library Grid.
m_p	–	the number of rows in the Library Grid, for this routine $m_p = 1$ or $m_p = p$;
n_p	–	the number of columns in the Library Grid, for this routine $n_p = 1$ or $n_p = p$.
N_b^x	–	the blocking factor for the distribution of the elements of a vector x.

3.2 Global and Local Arguments

The following global **input** arguments must have the same value on entry to the routine on each processor and the global **output** arguments will have the same value on exit from the routine on each processor:

Global input arguments:	N, some elements of IDESCV (see Section 4 for a description of IDESCV), IFAIL
Global output arguments:	IFAIL

The remaining arguments are local.

3.3 Distribution Strategy

The vector v is partitioned into N_b^v blocks on a one-dimensional processor grid, and stored in array V. This data distribution (**row block distribution**) is described in more detail in the F07 Chapter Introduction.

4 Arguments

1: GRHS — SUBROUTINE, supplied by the user. *External Procedure*

GRHS must return the block $v(i_1 : i_2)$ of the vector to be distributed, in the array VL (i.e., the elements v_i, such that $i_1 \leq i \leq i_2$).

Its specification is:

```
SUBROUTINE          GRHS(I1, I2, VL)
COMPLEX*16          VL(*)
INTEGER             I1, I2
```

1: I1 — INTEGER *Local Input*
On entry: i_1, the first element of the block of v to be generated.

2: I2 — INTEGER *Local Input*
On entry: i_2, the last element of the block of v to be generated.

3: VL(∗) — COMPLEX*16 array *Local Output*
On exit: VL must contain the part $v(i_1 : i_2)$ of the vector v in the first $(i_2 - i_1 + 1)$ elements.

GRHS must be declared as EXTERNAL in the (sub)program from which F01ZYFP is called. Arguments denoted as *Input* must **not** be changed by this procedure.

2: N — INTEGER *Global Input*
On entry: n, the dimension of the vector v.

Constraint: N ≥ 0.

3: V(∗) — COMPLEX*16 array *Local Output*
Note: the dimension of array V must be at least N_b^v.

On exit: the local part of the vector v which is stored in row block fashion.

4: IDESCV(∗) — INTEGER array *Local Input*
Note: the dimension of the array IDESCV must be at least 8 when IDESCV(1) = 1, and at least 5 when IDESCV(1) = 501 or 502.

Distribution: if IDESCV(1) = 1, the array elements IDESCV(3:8) must be global to each processor on the Libray Grid. If IDESCV(1) = 501 or 502, then only the array elements IDESCV(3:5) must be global. In all cases IDESCV(2) is local to each processor.

On entry: the description array for the vector v. This array must contain details of the distribution of the vector v and the logical processor grid.

IDESCV(1), the descriptor type.
If IDESCV(1) = 501 or 502, then $p = 1 \times n_p$, or $p = m_p \times 1$, and the remaining elements of IDESCA must be set as follows:

IDESCV(2), the Library context, usually returned by a call to the Library Grid initialisation routine Z01AAFP;
IDESCV(3), the length n, of the vector v;
IDESCV(4), the blocking factor, N_b^v, used to distribute the vector v;
IDESCV(5), the processor index over which the first element of the vector v is distributed;
IDESCV(6:8) are not referenced.

If IDESCV(1) = 1, then $p = 1 \times n_p$ and the remaining elements of IDESCA must be set as follows:

IDESCV(2), the Library context, usually returned by a call to the Library Grid initialisation routine Z01AAFP;
IDESCV(3), the length n, of the vector v;
IDESCV(4), the number of columns of the vector v (= 1);
IDESCV(5), the blocking factor used to distribute the rows of the vector v (= 1);
IDESCV(6), the blocking factor, N_b^v, used to distribute the elements of the vector v;

IDESCV(7), the processor row index over which the first row of the vector v is distributed;
IDESCV(8), the processor column index over which the first column of the vector v is distributed.

Suggested value: IDESCV(1) = 501 and $p = 1 \times n_p$.

Constraints:

IDESCV(1) = 1, 501 or 502;
if IDESCV(1) = 1 then $p = 1 \times n_p$;
if IDESCV(1) = 501 or 502; then $p = m_p \times 1$ or $p = 1 \times n_p$;
if IDESCV(1) = 501 or 502, then
  IDESCV(3) = N;
  IDESCV(4) $\geq$ 2 and $p\times$IDESCV(4) $\geq$ N;
  IDESCV(5) $\geq$ 0;
if IDESCV(1) = 1, then
  1 $\leq$ IDESCV(3) $\leq$ N;
  IDESCV(4) = IDESCV(5) = 1;
  IDESCV(6) $\geq$ 2 and $p\times$IDESCV(6) $\geq$ N;
  IDESCV(8) $\geq$ 0.

5: IFAIL — INTEGER *Global Input/Global Output*

The NAG Parallel Library provides a mechanism, via the routine Z02EAFP, to reduce the amount of parameter validation performed by this routine. For a full description refer to the Z02 Chapter Introduction.

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this argument (described in the Essential Introduction) the recommended values are:

IFAIL = 0, if multigridding is **not** employed;
IFAIL = −1, if multigridding is employed.

On exit: IFAIL = 0 (or −9999 if reduced error checking is enabled) unless the routine detects an error (see Section 5).

5 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output from the root processor (or processor {0,0} when the root processor is not available) on the current error message unit (as defined by X04AAF).

5.1 Full Error Checking Mode Only

IFAIL = −2000

The routine has been called with a value of ICNTXT (stored in IDESCV(2)) which was not returned by a call to Z01AAFP on one or more processors.

IFAIL = −1000

The utility routine Z01AAFP has not been called to define the logical processor grid and initialise the internal variables used by the Library.

IFAIL = $-i$

On entry, one of the arguments was invalid:

if the kth argument is a scalar IFAIL = $-k$;
if the kth argument is an array and its jth element is invalid, IFAIL = $-(100 \times k + j)$.

This error occured either because a global argument did not have the same value on all logical processors, or because its value on one or more processors was incorrect. An explanatory message distinguishes between these two cases.

6 Further Comments

6.1 Algorithmic Detail

The routine generates a complex vector, on a one-dimensional grid of logical processor using a row block distribution.

6.2 Parallelism Detail

The routine generates a vector on each logical processor independently.

7 References

None.

8 Example

See Section 8 of the document for F07JRFP (PZPTTRF).