

# NAG Toolbox for MATLAB

## f01za

### 1 Purpose

f01za copies a real triangular matrix stored in a packed one-dimensional array into an unpacked two-dimensional array, or vice versa.

### 2 Syntax

```
[a, b, ifail] = f01za(job, uplo, diag, a, b, 'n', n)
```

### 3 Description

f01za unpacks a triangular matrix stored in a vector into a two-dimensional array, or packs a triangular matrix stored in a two-dimensional array into a vector. The matrix is packed by column. This function is intended for possible use in conjunction with functions from Chapters F06, F07 and F08 where some functions that use triangular matrices store them in the packed form described below.

### 4 References

None.

### 5 Parameters

#### 5.1 Compulsory Input Parameters

1: **job** – string

Specifies whether the triangular matrix is to be packed or unpacked.

**job** = 'P' (Pack)

The matrix is to be packed into array **b**.

**job** = 'U' (Unpack)

The matrix is to be unpacked into array **a**.

*Constraint:* **job** must be one of 'P' or 'U'.

2: **uplo** – string

Specifies the type of the matrix to be copied

**uplo** = 'L' (Lower)

The matrix is lower triangular. In this case the packed vector holds, or will hold on exit, the matrix elements in the following order:  $(1, 1), (2, 1), \dots, (\mathbf{n}, 1), (2, 2), (3, 2), \dots, (\mathbf{n}, 2), \dots$

**uplo** = 'U' (Upper)

The matrix is upper triangular. In this case the packed vector holds, or will hold on exit, the matrix elements in the following order:  $(1, 1), (1, 2), (2, 2), (1, 3), (2, 3), (3, 3), (1, 4), \dots$

*Constraint:* **uplo** must be one of 'L' or 'U'.

3: **diag** – string

Must specify whether the diagonal elements of the matrix are to be copied.

**diag** = 'B' (Blank)

The diagonal elements of the matrix are not referenced and not copied.

**diag** = 'U' (Unit diagonal)

The diagonal elements of the matrix are not referenced, but are assumed all to be unity, and are copied as such.

**diag** = 'N' (Non-unit diagonal)

The diagonal elements of the matrix are referenced and copied.

*Constraint:* **diag** must be one of 'B', 'U' or 'N'.

4: **a(lda,n)** – double array

**lda**, the first dimension of the array, must be at least **n**.

If **job** = 'P', then the leading **n** by **n** part of **a** must contain the matrix to be copied, stored in unpacked form, in the upper or lower triangle depending on parameter **uplo**. The opposite triangle of **a** is not referenced and need not be assigned.

5: **b((n × (n + 1))/2)** – double array

If **job** = 'U', then **b** must contain the triangular matrix packed by column.

## 5.2 Optional Input Parameters

1: **n** – int32 scalar

*Default:* The dimension of the array **a**.

**n**, the number of rows and columns of the triangular matrix.

*Constraint:* **n** > 0.

## 5.3 Input Parameters Omitted from the MATLAB Interface

**lda**

## 5.4 Output Parameters

1: **a(lda,n)** – double array

If **job** = 'U', then the leading **n** by **n** part of array **a** contains the copied matrix, stored in unpacked form, in the upper or lower triangle depending on parameter **uplo**. The opposite triangle of **a** is not referenced.

2: **b((n × (n + 1))/2)** – double array

If **job** = 'P', then **b** contains the triangular matrix packed by column.

Note that **b** must have space for the diagonal elements of the matrix, even if these are not stored.

3: **ifail** – int32 scalar

0 unless the function detects an error (see Section 6).

## 6 Error Indicators and Warnings

Errors or warnings detected by the function:

**ifail** = 1

On entry, **job** ≠ 'P' or 'U'.

**ifail** = 2On entry, **uplo**  $\neq$  'L' or 'U'.**ifail** = 3On entry, **diag**  $\neq$  'N', 'U' or 'B'.**ifail** = 4On entry, **n** < 1.**ifail** = 5On entry, **lda** < **n**.

## 7 Accuracy

Not applicable.

## 8 Further Comments

None.

## 9 Example

```

job = 'Pack';
uplo = 'U';
diag = 'N';
a = [1.1, 1.2, 1.3, 1.4;
     0, 2.2, 2.3, 2.4;
     0, 0, 3.3, 3.4;
     0, 0, 0, 4.4];
b = zeros(10, 1);
[aOut, bOut, ifail] = f01za(job, uplo, diag, a, b)

```

```

aOut =
    1.1000    1.2000    1.3000    1.4000
         0    2.2000    2.3000    2.4000
         0         0    3.3000    3.4000
         0         0         0    4.4000

```

```

bOut =
    1.1000
    1.2000
    2.2000
    1.3000
    2.3000
    3.3000
    1.4000
    2.4000
    3.4000
    4.4000

```

```

ifail =
         0

```