

NAG Library Routine Document

F11MKF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details.

1 Purpose

F11MKF computes a matrix-matrix or transposed matrix-matrix product involving a real, square, sparse nonsymmetric matrix stored in compressed column (Harwell–Boeing) format.

2 Specification

```

SUBROUTINE F11MKF (TRANS, N, M, ALPHA, ICOLZP, IROWIX, A, B, LDB, BETA,      &
                  C, LDC, IFAIL)
INTEGER          N, M, ICOLZP(*), IROWIX(*), LDB, LDC, IFAIL
REAL (KIND=nag_wp) ALPHA, A(*), B(LDB,*), BETA, C(LDC,*)
CHARACTER(1)    TRANS

```

3 Description

F11MKF computes either the matrix-matrix product $C \leftarrow \alpha AB + \beta C$, or the transposed matrix-matrix product $C \leftarrow \alpha A^T B + \beta C$, according to the value of the parameter TRANS, where A is a real n by n sparse nonsymmetric matrix, of arbitrary sparsity pattern with nnz nonzero elements, B and C are n by m real dense matrices. The matrix A is stored in compressed column (Harwell–Boeing) storage format. The array A stores all nonzero elements of A , while arrays ICOLZP and IROWIX store the compressed column indices and row indices of A respectively.

4 References

None.

5 Parameters

- 1: TRANS – CHARACTER(1) *Input*
On entry: specifies whether or not the matrix A is transposed.
 TRANS = 'N'
 $\alpha AB + \beta C$ is computed.
 TRANS = 'T'
 $\alpha A^T B + \beta C$ is computed.
Constraint: TRANS = 'N' or 'T'.
- 2: N – INTEGER *Input*
On entry: n , the order of the matrix A .
Constraint: $N \geq 0$.
- 3: M – INTEGER *Input*
On entry: m , the number of columns of matrices B and C .
Constraint: $M \geq 0$.

- 4: ALPHA – REAL (KIND=nag_wp) Input
On entry: α , the scalar factor in the matrix multiplication.
- 5: ICOLZP(*) – INTEGER array Input
Note: the dimension of the array ICOLZP must be at least $N + 1$.
On entry: ICOLZP(i) contains the index in A of the start of a new column. See Section 2.1.3 in the F11 Chapter Introduction.
- 6: IROWIX(*) – INTEGER array Input
Note: the dimension of the array IROWIX must be at least $ICOLZP(N + 1) - 1$, the number of nonzeros of the sparse matrix A .
On entry: the row index array of sparse matrix A .
- 7: A(*) – REAL (KIND=nag_wp) array Input
Note: the dimension of the array A must be at least $ICOLZP(N + 1) - 1$, the number of nonzeros of the sparse matrix A .
On entry: the array of nonzero values in the sparse matrix A .
- 8: B(LDB, *) – REAL (KIND=nag_wp) array Input
Note: the second dimension of the array B must be at least $\max(1, M)$.
On entry: the n by m matrix B .
- 9: LDB – INTEGER Input
On entry: the first dimension of the array B as declared in the (sub)program from which F11MKF is called.
Constraint: $LDB \geq \max(1, N)$.
- 10: BETA – REAL (KIND=nag_wp) Input
On entry: the scalar factor β .
- 11: C(LDC, *) – REAL (KIND=nag_wp) array Input/Output
Note: the second dimension of the array C must be at least $\max(1, M)$.
On entry: the n by m matrix C .
On exit: C is overwritten by $\alpha AB + \beta C$ or $\alpha A^T B + \beta C$ depending on the value of TRANS.
- 12: LDC – INTEGER Input
On entry: the first dimension of the array C as declared in the (sub)program from which F11MKF is called.
Constraint: $LDC \geq \max(1, N)$.
- 13: IFAIL – INTEGER Input/Output
On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.
 For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry, TRANS \neq 'N' or 'T',
or $N < 0$,
or $M < 0$,
or $LDB < \max(1, N)$,
or $LDC < \max(1, N)$.

IFAIL = -99

An unexpected error has been triggered by this routine. Please contact NAG.
See Section 3.8 in the Essential Introduction for further information.

IFAIL = -399

Your licence key may have expired or may not have been installed correctly.
See Section 3.7 in the Essential Introduction for further information.

IFAIL = -999

Dynamic memory allocation failed.
See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

F11MKF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

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

9 Further Comments

None.

10 Example

This example reads in a sparse matrix A and a dense matrix B . It then calls F11MKF to compute the matrix-matrix product $C = AB$ and the transposed matrix-matrix product $C = A^T B$, where

$$A = \begin{pmatrix} 2.00 & 1.00 & 0 & 0 & 0 \\ 0 & 0 & 1.00 & -1.00 & 0 \\ 4.00 & 0 & 1.00 & 0 & 1.00 \\ 0 & 0 & 0 & 1.00 & 2.00 \\ 0 & -2.00 & 0 & 0 & 3.00 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0.70 & 1.40 \\ 0.16 & 0.32 \\ 0.52 & 1.04 \\ 0.77 & 1.54 \\ 0.28 & 0.56 \end{pmatrix}.$$

10.1 Program Text

```

Program f11mkfe

!      F11MKF Example Program Text

!      Mark 25 Release. NAG Copyright 2014.

!      .. Use Statements ..
Use nag_library, Only: f11mkf, nag_wp, x04caf
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Real (Kind=nag_wp), Parameter      :: alpha = 1.E0_nag_wp
Real (Kind=nag_wp), Parameter      :: beta = 0.E0_nag_wp
Integer, Parameter                  :: nin = 5, nout = 6
!      .. Local Scalars ..
Integer                              :: i, ifail, j, ldb, ldc, m, n, nnz
Character (1)                          :: trans
!      .. Local Arrays ..
Real (Kind=nag_wp), Allocatable      :: a(:), b(:,,:), c(:,:)
Integer, Allocatable                  :: icolzp(:), irowix(:)
!      .. Executable Statements ..
Write (nout,*) 'F11MKF Example Program Results'
!      Skip heading in data file
Read (nin,*)

!      Read order of matrix

Read (nin,*) n, m
ldb = n
ldc = n

Allocate (b(ldb,m),c(ldc,m),icolzp(n+1))

!      Read the matrix A

Read (nin,*) icolzp(1:n+1)
nnz = icolzp(n+1) - 1

Allocate (a(nnz),irowix(nnz))

Do i = 1, nnz
  Read (nin,*) a(i), irowix(i)
End Do

!      Read the matrix B

Do j = 1, m
  Read (nin,*) b(1:n,j)
End Do

!      Calculate matrix-matrix product

trans = 'N'

!      ifail: behaviour on error exit
!            =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft

```

```

        ifail = 0
        Call f11mkf(trans,n,m,alpha,icolzp,irowix,a,b,ldb,beta,c,ldc,ifail)

!       Output results
        Write (nout,*)
        Flush (nout)

        Call x04caf('G',' ',n,m,c,ldc,'Matrix-matrix product',ifail)

!       Calculate transposed matrix-matrix product
        trans = 'T'

        ifail = 0
        Call f11mkf(trans,n,m,alpha,icolzp,irowix,a,b,ldb,beta,c,ldc,ifail)

!       Output results
        Write (nout,*)
        Flush (nout)

        Call x04caf('G',' ',n,m,c,ldc,'Transposed matrix-matrix product',ifail)

End Program f11mkfe

```

10.2 Program Data

F11MKF Example Program Data

```

5 2          N, M
1
3
5
7
9
12  ICOLZP(I) I=1,..,N+1
2.  1
4.  3
1.  1
-2. 5
1.  2
1.  3
-1. 2
1.  4
1.  3
2.  4
3.  5      A(I), IROWIX(I) I=1,...,NNZ
0.70 0.16 0.52 0.77 0.28
1.40 0.32 1.04 1.54 0.56      matrix B

```

10.3 Program Results

F11MKF Example Program Results

Matrix-matrix product

	1	2
1	1.5600	3.1200
2	-0.2500	-0.5000
3	3.6000	7.2000
4	1.3300	2.6600
5	0.5200	1.0400

Transposed matrix-matrix product

	1	2
1	3.4800	6.9600
2	0.1400	0.2800
3	0.6800	1.3600
4	0.6100	1.2200
5	2.9000	5.8000
