NB (input) INTEGER
NB specifies the column block size of the matrix C and the row block size of the matrix C. NB $\geq$ 1.

A (input) REAL/COMPLEX array of DIMENSION (LDA, Nq).
The leading Mp-by-Nq part of the array A must contain the (local) matrix A.

LDA (input) INTEGER
The leading dimension of the (local) array A. LDA $\geq$ MAX(1, Mp).

BETA (input) REAL/COMPLEX
BETA specifies the scalar beta. When BETA is supplied as zero then C need not be set on input.

C (input/output) REAL/COMPLEX array of DIMENSION (LDC, Mq).
The leading Np-by-Mq part of the array C must contain the (local) matrix C, except when beta is zero, in which case C need not be set on entry.
On exit, the array C is overwritten by the Np-by-Mq matrix ($\mathbf{A}' +$ beta $\cdot$ $\mathbf{C}$).

LDC (input) INTEGER
The leading dimension of the (local) array C. LDC $\geq$ MAX(1, Np).

IMROW (input) INTEGER
IMROW specifies a row of the process template, which holds the first blocks of the matrices, A and C. 0 $\leq$ IMROW $<$ P.

IMCOL (input) INTEGER
IMCOL specifies a column of the process template, which holds the first blocks of the matrices, A and C. 0 $\leq$ IMCOL $<$ Q.

WORK (workspace) REAL/COMPLEX array
See requirements.

IWORK (workspace) INTEGER array, DIMENSION (3 $\times$ MAX(P, Q))

Requirements

Compile with `pbblasinc.f`.

Size(IWORK) = 3 $\times$ MAX(P, Q)
Size(WORK) = 2 $\times$ Size(A)
where
Size(A) = $(\lceil MG/LCM \rceil \cdot MB) \times (\lceil NG/LCM \rceil \cdot NB)$

7. PSTRANS/PCTRANS

```
      SUBROUTINE PSTRANS( TRANS, M, N, MB, NB, A, LDA, BETA, C, LDC,
     $                    IMROW, IMCOL, WORK, IWORK )
      CHARACTER*1         TRANS
      INTEGER             M, N, MB, NB, LDA, LDC, IMROW, IMCOL
      REAL                BETA
      REAL                A( LDA, * ), C( LDC, * ), WORK( * )
      INTEGER             IWORK( 3, 0:* )

      SUBROUTINE PCTRANS( TRANS, M, N, MB, NB, A, LDA, BETA, C, LDC,
     $                    IMROW, IMCOL, WORK, IWORK )
      CHARACTER*1         TRANS
      INTEGER             M, N, MB, NB, LDA, LDC, IMROW, IMCOL
      COMPLEX             BETA
      COMPLEX             A( LDA, * ), C( LDC, * ), WORK( * )
      INTEGER             IWORK( 3, 0:* )
```

Purpose

PSTRANS/PCTRANS is used for the following matrix transposition operation,

$$\mathbf{C} \Leftarrow \mathbf{A}^T + \text{beta} \cdot \mathbf{C},$$

where beta is a scalar, and A and C are matrices, with A an M-by-N matrix and C an N-by-M matrix.

Arguments

TRANS (input) CHARACTER*1

TRANS specifies the form of op(A) to be used in the matrix multiplication.

TRANS = 'T': op(A) = A^T;

TRANS = 'C': op(A) = A^H.

M (input) INTEGER

M specifies the (global) number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER

N specifies the (global) number of columns of the matrix B. N $\geq$ 0.

MB (input) INTEGER

MB specifies the row block size of the matrix A and the column block size of the matrix C. MB $\geq$ 1.

IMROW specifies a row of the process template, which holds the first blocks of the matrices, A, B, and C. $0 \leq$ IMROW $<$ P.

IMCOL (input) INTEGER
IMCOL specifies a column of the process template, which holds the first blocks of the matrices, A, B, and C. $0 \leq$ IMCOL $<$ Q.

WORK (workspace) REAL/COMPLEX array
See requirements.

IWORK (workspace) INTEGER array, DIMENSION ($3 \times$ P)

Requirements

Compile with `pummainc.f` and `pbblasinc.f`.

Size(IWORK) = $3 \times$ P

Size(WORK) = Size(C) + $2 \times$ MAX[Size(A), Size(C)] for PICL

Size(WORK) = Size(A) + $2 \times$ Size(C) for BLACS

where

Size(A) = $(\lceil KG/P \rceil \cdot KB) \times Mq$

Size(C) = $(\lceil MG/LCM \rceil \cdot MB) \times Nq$

M (input) INTEGER
M specifies the (global) number of columns of the matrix A and the (global) number of rows of the matrix C. M $\geq$ 0.

N (input) INTEGER
N specifies the (global) number of columns of the matrices B and C. N $\geq$ 0.

K (input) INTEGER
K specifies the (global) number of rows of the matrices A and B. K $\geq$ 0.

ALPHA (input) REAL/COMPLEX
ALPHA specifies the scalar alpha.

MB (input) INTEGER
MB specifies the column block size of the matrix A and the row block size of the matrix C. MB $\geq$ 1.

NB (input) INTEGER
NB specifies the column block size of the matrices B and C. NB $\geq$ 1.

KB (input) INTEGER
KB specifies the column block size of the matrices A and B. KB $\geq$ 1.

A (input) REAL/COMPLEX array of DIMENSION (LDA, Mq).
The leading Kp-by-Mq part of the array A must contain the (local) matrix A.

LDA (input) INTEGER
The leading dimension of the (local) array A. LDA $\geq$ MAX(1, Kp).

B (input) REAL/COMPLEX array of DIMENSION (LDB, Nq).
The leading Kp-by-Nq part of the array B must contain the (local) matrix B.

LDB (input) INTEGER
The leading dimension of the (local) array B. LDB $\geq$ MAX(1, Kp).

BETA (input) REAL/COMPLEX
BETA specifies the scalar beta. When BETA is supplied as zero then C need not be set on input.

C (input/output) REAL/COMPLEX array of DIMENSION (LDC, Nq).
The leading Mp-by-Nq part of the array C must contain the (local) matrix C, except when beta is zero, in which case C need not be set on entry.
On exit, the array C is overwritten by the Mp-by-Nq matrix (alpha $\cdot$ $\mathbf{A}' \cdot \mathbf{B}$ + beta $\cdot$ $\mathbf{C}$).

LDC (input) INTEGER
The leading dimension of the (local) array C. LDC $\geq$ MAX(1, Mp).

IMROW (input) INTEGER

6. PSMTNC/PCMTNC

```
      SUBROUTINE PSMTNC( TRANSA, M, N, K, MB, NB, KB, ALPHA, A, LDA, B,
     $                   LDB, BETA, C, LDC, IMROW, IMCOL, WORK, IWORK )
      CHARACTER*1        TRANSA
      INTEGER            M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER            IMROW, IMCOL
      REAL               ALPHA, BETA
      REAL               A( LDA, * ), B( LDB, * ), C( LDC, * )
      REAL               WORK( * )
      INTEGER            IWORK( 3, 0:* )

      SUBROUTINE PCMTNC( TRANSA, M, N, K, MB, NB, KB, ALPHA, A, LDA, B,
     $                   LDB, BETA, C, LDC, IMROW, IMCOL, WORK, IWORK )
      CHARACTER*1        TRANSA
      INTEGER            M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER            IMROW, IMCOL
      REAL               ALPHA, BETA
      REAL               A( LDA, * ), B( LDB, * ), C( LDC, * )
      REAL               WORK( * )
      INTEGER            IWORK( 3, 0:* )
```

Purpose

PSMTNC/PCMTNC performs the following matrix-matrix multiplication operation,

$$\mathbf{C} \Leftarrow \text{alpha} \cdot \mathbf{A}' \cdot \mathbf{B} + \text{beta} \cdot \mathbf{C},$$

where $A' = A^T$ or $A' = A^H$, alpha and beta are scalars, and A, B and C are matrices, with A an K-by-M matrix, B a K-by-N matrix, and C an M-by-N matrix.

The basic principles are *Columnwise Rotate C* and *Rowwise Shift A*. The partial sums are accumulated based on ring architecture (not spanning tree structure).

Arguments

TRANSA (input) CHARACTER*1

TRANSA specifies the form of op(A) to be used in the matrix multiplication.

TRANSA = 'T': $op(A) = A^T$;

TRANSA = 'C': $op(A) = A^H$.

For REAL case, the argument is ignored, that is, it is always assumed that TRANSA = 'T'.

IMROW (input) INTEGER
IMROW specifies a row of the process template, which holds the first blocks of the matrices, A, B, and C. $0 \leq$ IMROW $<$ P.

IMCOL (input) INTEGER
IMCOL specifies a column of the process template, which holds the first blocks of the matrices, A, B, and C. $0 \leq$ IMCOL $<$ Q.

WORK (workspace) REAL/COMPLEX array
See requirements.

IWORK (workspace) INTEGER array, DIMENSION (3 $\times$ Q)

Requirements

Compile with `pummainc.f` and `pbblasinc.f`.

Size(IWORK) = 3 $\times$ Q

Size(WORK) = Size(C) + 2 $\times$ MAX[Size(B), Size(C)] for PICL

Size(WORK) = Size(B) + 2 $\times$ Size(C) for BLACS

where

Size(B) = $Np \times (\lceil KG/Q \rceil \cdot KB)$

Size(C) = $Mp \times (\lceil NG/LCM \rceil \cdot NB)$

M (input) INTEGER
M specifies the (global) number of rows of the matrices A and C. M $\geq$ 0.

N (input) INTEGER
N specifies the (global) number of rows of the matrix B and the (global) number of columns of the matrix C. N $\geq$ 0.

K (input) INTEGER
K specifies the (global) number of columns of the matrices A and B. K $\geq$ 0.

MB (input) INTEGER
MB specifies the row block size of the matrices A and C. MB $\geq$ 1.

NB (input) INTEGER
NB specifies the row block size of the matrix B and the column block size of the matrix C. NB $\geq$ 1.

KB (input) INTEGER
KB specifies the column block size of the matrices A and B. KB $\geq$ 1.

ALPHA (input) REAL/COMPLEX
ALPHA specifies the scalar alpha.

A (input) REAL/COMPLEX array of DIMENSION (LDA, Kq).
The leading Mp-by-Kq part of the array A must contain the (local) matrix A.

LDA (input) INTEGER
The leading dimension of the (local) array A. LDA $\geq$ MAX(1, Mp).

B (input) REAL/COMPLEX array of DIMENSION (LDB, Kq).
The leading Np-by-Kq part of the array B must contain the (local) matrix B.

LDB (input) INTEGER
The leading dimension of the (local) array B. LDB $\geq$ MAX(1, Np).

BETA (input) REAL/COMPLEX
BETA specifies the scalar beta. When BETA is supplied as zero then C need not be set on input.

C (input/output) REAL/COMPLEX array of DIMENSION (LDC, Nq).
The leading Mp-by-Nq part of the array C must contain the (local) matrix C, except when beta is zero, in which case C need not be set on entry.
On exit, the array C is overwritten by the Mp-by-Nq matrix (alpha $\cdot$ $\mathbf{A} \cdot \mathbf{B}'$ + beta $\cdot$ $\mathbf{C}$).

LDC (input) INTEGER
The leading dimension of the (local) array C. LDC $\geq$ MAX(1, Mp).

5. PSMNTC/PCMNTC

```
      SUBROUTINE PSMNTC( TRANSB, M, N, K, MB, NB, KB, ALPHA, A, LDA, B,
     $                   LDB, BETA, C, LDC, IMROW, IMCOL, WORK, IWORK )
      CHARACTER*1        TRANSB
      INTEGER            M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER            IMROW, IMCOL
      REAL               ALPHA, BETA
      REAL               A( LDA, * ), B( LDB, * ), C( LDC, * )
      REAL               WORK( * )
      INTEGER            IWORK( 3, 0:* )

      SUBROUTINE PCMNTC( TRANSB, M, N, K, MB, NB, KB, ALPHA, A, LDA, B,
     $                   LDB, BETA, C, LDC, IMROW, IMCOL, WORK, IWORK )
      CHARACTER*1        TRANSB
      INTEGER            M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER            IMROW, IMCOL
      REAL               ALPHA, BETA
      REAL               A( LDA, * ), B( LDB, * ), C( LDC, * )
      REAL               WORK( * )
      INTEGER            IWORK( 3, 0:* )
```

Purpose

PSMNTC/PCMNTC performs the following matrix-matrix multiplication operation,

$$\mathbf{C} \Leftarrow \text{alpha} \cdot \mathbf{A} \cdot \mathbf{B}' + \text{beta} \cdot \mathbf{C},$$

where $B' = B^T$ or $B' = B^H$, alpha and beta are scalars, and A, B and C are matrices, with A an M-by-K matrix, B a N-by-K matrix, and C an M-by-N matrix.

The basic principles are *Rowwise Rotate C* and *Columnwise Shift B*. The partial sums are accumulated based on ring architecture (not spanning tree structure).

Arguments

TRANSB (input) CHARACTER*1

TRANSB specifies the form of $op(B)$ to be used in the matrix multiplication.

TRANSB = 'T': $op(B) = B^T$;

TRANSB = 'C': $op(B) = B^H$.

For REAL case, the argument is ignored, that is, it is always assumed that TRANSB = 'T'.

IWORK (workspace) INTEGER array, DIMENSION (Q)

Requirements

Compile with `pummainc.f` and `pbblasinc.f`.

Size(IWORK) = P

Size(WORK) = Size(B) + 2 × MAX[Size(A), Size(B)] for PICL

Size(WORK) = Size(A) + Size(B) for BLACS

where

Size(A) = $Mp \times (\lceil KG/Q \rceil \cdot KB)$

Size(B) = $(\lceil KG/LCM \rceil \cdot KB) \times Nq$

MB specifies the row block size of the matrices A and C. MB $\geq$ 1.

NB (input) INTEGER
NB specifies the column block size of the matrices B and C. NB $\geq$ 1.

KB (input) INTEGER
KB specifies the column block size of the matrix A and the row block size of the matrix B. KB $\geq$ 1.

ALPHA (input) REAL/COMPLEX
ALPHA specifies the scalar alpha.

A (input) REAL/COMPLEX array of DIMENSION (LDA, Kq).
The leading Mp-by-Kq part of the array A must contain the (local) matrix A.

LDA (input) INTEGER
The leading dimension of the (local) array A. LDA $\geq$ MAX(1, Mp).

B (input) REAL/COMPLEX array of DIMENSION (LDB, Nq).
The leading Kp-by-Nq part of the array B must contain the (local) matrix B.

LDB (input) INTEGER
The leading dimension of the (local) array B. LDB $\geq$ MAX(1, Kp).

BETA (input) REAL/COMPLEX
BETA specifies the scalar beta. When BETA is supplied as zero then C need not be set on input.

C (input/output) REAL/COMPLEX array of DIMENSION (LDC, Nq).
On entry the leading Mp-by-Nq part of the array C must contain the (local) matrix C, except when beta is zero, in which case C need not be set on entry. On exit, the array C is overwritten by the Mp-by-Nq matrix (alpha $\cdot$ **A** $\cdot$ **B** + beta $\cdot$ **C**).

LDC (input) INTEGER
The leading dimension of the (local) array C. LDC $\geq$ MAX(1, Mp).

IMROW (input) INTEGER
IMROW specifies a row of the process template, which holds the first blocks of the matrices, A, B, and C. 0 $\leq$ IMROW $<$ P.

IMCOL (input) INTEGER
IMCOL specifies a column of the process template, which holds the first blocks of the matrices, A, B, and C. 0 $\leq$ IMCOL $<$ Q.

WORK (workspace) REAL/COMPLEX array
See requirements.

4. PSMNNB/PCMNNB

```
      SUBROUTINE PSMNNB( M, N, K, MB, NB, KB, ALPHA, A, LDA, B, LDB,
     $                   BETA, C, LDC, IMROW, IMCOL, WORK, IWORK )
      INTEGER            M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER            IMROW, IMCOL
      REAL               ALPHA, BETA
      REAL               A( LDA, * ), B( LDB, * ), C( LDC, * )
      REAL               WORK( * )
      INTEGER            IWORK( 0:* )

      SUBROUTINE PCMNNB( M, N, K, MB, NB, KB, ALPHA, A, LDA, B, LDB,
     $                   BETA, C, LDC, IMROW, IMCOL, WORK, IWORK )
      INTEGER            M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER            IMROW, IMCOL
      COMPLEX            ALPHA, BETA
      COMPLEX            A( LDA, * ), B( LDB, * ), C( LDC, * )
      COMPLEX            WORK( * )
      INTEGER            IWORK( 0:* )
```

Purpose

PSMNNB/PCMNNB performs the following matrix-matrix multiplication operation,

$$\mathbf{C} \Leftarrow \text{alpha} \cdot \mathbf{A} \cdot \mathbf{B} + \text{beta} \cdot \mathbf{C},$$

where alpha and beta are scalars, and A, B and C are matrices, with A an M-by-K matrix, B a K-by-N matrix, and C an M-by-N matrix.

The basic principles are *Rowwise Shift A* and *Columnwise Broadcast B*.

Arguments

M (input) INTEGER
M specifies the (global) number of rows of the matrices A and C. M $\geq$ 0.

N (input) INTEGER
N specifies the (global) number of columns of the matrices B and C. N $\geq$ 0.

K (input) INTEGER
K specifies the (global) number of columns of the matrix A and the (global) number of rows of the matrix B. K $\geq$ 0.

MB (input) INTEGER

IWORK (workspace) INTEGER array, DIMENSION (P)

Requirements

Compile with `pummainc.f` and `pbblasinc.f`.

Size(IWORK) = Q

Size(WORK) = Size(A) + 2 × MAX(Size(A), Size(B)) for PICL

Size(WORK) = Size(A) + Size(B) for BLACS

where

Size(A) = $(\lceil Mp \times (\lceil KG/LCM \rceil \cdot KB)$

Size(B) = $(\lceil KG/P \rceil \cdot KB) \times Nq$

MB specifies the row block size of the matrices A and C. MB $\geq$ 1.

NB (input) INTEGER
NB specifies the column block size of the matrices B and C. NB $\geq$ 1.

KB (input) INTEGER
KB specifies the column block size of the matrix A and the row block size of the matrix B. KB $\geq$ 1.

ALPHA (input) REAL/COMPLEX
ALPHA specifies the scalar alpha.

A (input) REAL/COMPLEX array of DIMENSION (LDA, Kq).
The leading Mp-by-Kq part of the array A must contain the (local) matrix A.

LDA (input) INTEGER
The leading dimension of the (local) array A. LDA $\geq$ MAX(1, Mp).

B (input) REAL/COMPLEX array of DIMENSION (LDB, Nq).
The leading Kp-by-Nq part of the array B must contain the (local) matrix B.

LDB (input) INTEGER
The leading dimension of the (local) array B. LDB $\geq$ MAX(1, Kp).

BETA (input) REAL/COMPLEX
BETA specifies the scalar beta. When BETA is supplied as zero then C need not be set on input.

C (input/output) REAL/COMPLEX array of DIMENSION (LDC, Nq).
On entry the leading Mp-by-Nq part of the array C must contain the (local) matrix C, except when beta is zero, in which case C need not be set on entry. On exit, the array C is overwritten by the Mp-by-Nq matrix (alpha $\cdot$ **A** $\cdot$ **B** + beta $\cdot$ **C**).

LDC (input) INTEGER
The leading dimension of the (local) array C. LDC $\geq$ MAX(1, Mp).

IMROW (input) INTEGER
IMROW specifies a row of the process template, which holds the first blocks of the matrices, A, B, and C. 0 $\leq$ IMROW $<$ P.

IMCOL (input) INTEGER
IMCOL specifies a column of the process template, which holds the first blocks of the matrices, A, B, and C. 0 $\leq$ IMCOL $<$ Q.

WORK (workspace) REAL/COMPLEX array
See requirements.

3. PSMNNA/PCMNNA

```
      SUBROUTINE PSMNNA( M, N, K, MB, NB, KB, ALPHA, A, LDA, B, LDB,
     $                   BETA, C, LDC, IMROW, IMCOL, WORK, IWORK )
      INTEGER            M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER            IMROW, IMCOL
      REAL               ALPHA, BETA
      REAL               A( LDA, * ), B( LDB, * ), C( LDC, * )
      REAL               WORK( * )
      INTEGER            IWORK( 0:* )

      SUBROUTINE PCMNNA( M, N, K, MB, NB, KB, ALPHA, A, LDA, B, LDB,
     $                   BETA, C, LDC, IMROW, IMCOL, WORK, IWORK )
      INTEGER            M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER            IMROW, IMCOL
      COMPLEX            ALPHA, BETA
      COMPLEX            A( LDA, * ), B( LDB, * ), C( LDC, * )
      COMPLEX            WORK( * )
      INTEGER            IWORK( 0:* )
```

Purpose

PSMNNA/PCMNNA performs the following matrix-matrix multiplication operation,

$$\mathbf{C} \Leftarrow \text{alpha} \cdot \mathbf{A} \cdot \mathbf{B} + \text{beta} \cdot \mathbf{C},$$

where alpha and beta are scalars, and A, B and C are matrices, with A an M-by-K matrix, B a K-by-N matrix, and C an M-by-N matrix.

The basic principles are *Rowwise Broadcast A* and *Columnwise Shift B*.

Arguments

M (input) INTEGER
M specifies the (global) number of rows of the matrices A and C. M $\geq$ 0.

N (input) INTEGER
N specifies the (global) number of columns of the matrices B and C. N $\geq$ 0.

K (input) INTEGER
K specifies the (global) number of columns of the matrix A and the (global) number of rows of the matrix B. K $\geq$ 0.

MB (input) INTEGER

Size(C) = $(\lceil NG/P \rceil \cdot NB) \times (\lceil MG/Q \rceil \cdot MB)$

(5) TRANSA = 'T', TRANSB = 'C' or TRANSA = 'C', TRANSB = 'T'
(COMPLEX CASE ONLY – PCGEMM)

(5a) PCTRAN with PCMNNA if P $\geq$ Q

Size(IWORK) = 3 $\times$ P

Size(WORK) = Size(A)
+ MAX[Size(Ap) + 2 $\times$ MAX[Size(Ap), Size(B)],
Size(B) + MAX[Size(B), 2 $\times$ Size(Bp)]] for PICL

Size(WORK) = Size(A)
+ MAX[Size(Ap) + Size(B),
Size(B) + MAX[Size(B), 2 $\times$ Size(Bp)]] for BLACS

where

Size(A) = $Mp \times (\lceil KG/Q \rceil \cdot KB)$

Size(Ap) = $(\lceil MG/P \rceil \cdot MB) \times (\lceil KG/LCM \rceil \cdot KB)$

Size(B) = $(\lceil KG/P \rceil \cdot KB) \times Nq$

Size(Bp) = $(\lceil KG/LCM \rceil \cdot KB) \times (\lceil NG/LCM \rceil \cdot NB)$

(5b) PCTRAN with PCMNNB if P < Q

Size(IWORK) = 3 $\times$ Q

Size(WORK) = Size(B)
+ MAX[Size(Bp) + 2 $\times$ MAX[Size(A), Size(Bp)],
Size(A) + MAX[Size(A), 2 $\times$ Size(Ap)]] for PICL

Size(WORK) = Size(B)
+ MAX[Size(A) + Size(Bp),
Size(A) + MAX[Size(A), 2 $\times$ Size(Ap)]] for BLACS

where

Size(A) = $(\lceil MG/P \rceil \cdot MB) \times (\lceil KG/Q \rceil \cdot KB)$

Size(Ap) = $(\lceil MG/LCM \rceil \cdot MB) \times (\lceil KG/LCM \rceil \cdot KB)$

Size(B) = $(\lceil KG/P \rceil \cdot KB) \times (\lceil NG/Q \rceil \cdot NB)$

Size(Bp) = $(\lceil KG/LCM \rceil \cdot KB) \times (\lceil NG/Q \rceil \cdot NB)$

Size(WORK) = Size(B) + 2 × MAX[Size(A), Size(B)] for PICL

Size(WORK) = Size(A) + Size(B) for BLACS

where

Size(A) = $Mp \times (\lceil KG/Q \rceil \cdot KB)$

Size(B) = $(\lceil KG/LCM \rceil \cdot KB) \times Nq$

(2) TRANSA = 'N' and TRANSB = 'T'/'C' (PSMNTC/PCMNTC)

Size(IWORK) = 3 × Q

Size(WORK) = Size(C) + 2 × MAX[Size(B), Size(C)] for PICL

Size(WORK) = Size(B) + 2 × Size(C) for BLACS

where

Size(B) = $Np \times (\lceil KG/Q \rceil \cdot KB)$

Size(C) = $Mp \times (\lceil NG/LCM \rceil \cdot NB)$

(3) TRANSA = 'T'/'C' and TRANSB = 'N' (PSMTNC/PCMTNC)

Size(IWORK) = 3 × P

Size(WORK) = Size(C) + 2 × MAX[Size(A), Size(C)] for PICL

Size(WORK) = Size(A) + 2 × Size(C) for BLACS

where

Size(A) = $(\lceil KG/P \rceil \cdot KB) \times Mq$

Size(C) = $(\lceil MG/LCM \rceil \cdot MB) \times Nq$

(4) TRANSA = 'T', TRANSB = 'T' or TRANSA = 'C', TRANSB = 'C'

(4a) PSTRAN/PCTRAN with PSMNNA/PCMNNA if P ≥ Q

Size(IWORK) = 3 × P

Size(WORK) = Size(A) + Size(C) + 2 × MAX[Size(A), Size(B)]

Size(WORK) = Size(A) + Size(B) + Size(C) for BLACS

where

Size(A) = $(\lceil KG/P \rceil \cdot KB) \times Mq$

Size(B) = $Np \times (\lceil KG/LCM \rceil \cdot KB)$

Size(C) = $(\lceil NG/P \rceil \cdot NB) \times (\lceil MG/Q \rceil \cdot MB)$

(4b) PSTRAN/PCTRAN with PSMNNB/PCMNNB if P < Q

Size(IWORK) = 3 × Q

Size(WORK) = Size(B) + Size(C) + 2 × MAX[Size(A), Size(B)] for PICL

Size(WORK) = Size(A) + Size(B) + Size(C) for BLACS

where

Size(A) = $(\lceil KG/LCM \rceil \cdot KB) \times Mq$

Size(B) = $Np \times (\lceil KG/Q \rceil \cdot KB)$

LDB (input) INTEGER
The leading dimension of the (local) array B. When TRANSB = 'N', LDB $\geq$ MAX(1, Kp), otherwise LDB $\geq$ MAX(1, Np).

BETA (input) REAL/COMPLEX
BETA specifies the scalar beta. When BETA is supplied as zero then C need not be set on input.

C (input/output) REAL/COMPLEX array of DIMENSION (LDC, Nq).
On entry, the leading Mp-by-Nq part of the array C must contain the (local) matrix C, except when beta is zero, in which case C need not be set on entry. On exit, the array C is overwritten by the Mp-by-Nq matrix (alpha $\cdot$ $op(\mathbf{A})$ $\cdot$ $op(\mathbf{B})$ + beta $\cdot$ $\mathbf{C}$).

LDC (input) INTEGER
The leading dimension of the (local) array C. LDC $\geq$ MAX(1, Mp).

IMROW (input) INTEGER
IMROW specifies a row of the process template, which holds the first blocks of the matrices, A, B, and C. $0 \leq$ IMROW $<$ P.

IMCOL (input) INTEGER
IMCOL specifies a column of the process template, which holds the first blocks of the matrices, A, B, and C. $0 \leq$ IMCOL $<$ Q.

WORK (workspace) REAL/COMPLEX array
See requirements.

IWORK (workspace) INTEGER array
See requirements.

Requirements

Compile with `pummainc.f`, `pbblasinc.f`, `psmnna.f/pcmnna.f`, `psmnnb.f/pcmnnb.f`, `psmntc.f/pcmntc.f`, `psmtnc.f/pcmtnc.f`, and `pstran.f/pctran.f`.

(1a) TRANSA = 'N' and TRANSB = 'N' (PSMNNA/PCMNNA if P $\geq$ Q)

Size(IWORK) = Q

Size(WORK) = Size(A) + 2 $\times$ MAX[Size(A), Size(B)] for PICL

Size(WORK) = Size(A) + Size(B) for BLACS

where

Size(A) = $Mp \times (\lceil KG/LCM \rceil \cdot KB)$

Size(B) = $(\lceil KG/P \rceil \cdot KB) \times Nq$

(1b) TRANSA = 'N' and TRANSB = 'N' (PSMNNB/PCMNNB if P $<$ Q)

Size(IWORK) = P

TRANSB (input) CHARACTER*1

TRANSB specifies the form of op(B) to be used in the matrix multiplication.

TRANSB = 'N': op(B) = B;

TRANSB = 'T': op(B) = B^T;

TRANSB = 'C': op(B) = B^H.

M (input) INTEGER

M specifies the (global) number of rows of the matrices op(A) and C. M ≥ 0.

N (input) INTEGER

N specifies the (global) number of columns of the matrices op(B) and C. N ≥ 0.

K (input) INTEGER

K specifies the (global) number of columns of the matrix op(A) and the (global) number of rows of the matrix op(B). K ≥ 0.

MB (input) INTEGER

MB specifies the row block size of the matrices op(A) and C. MB ≥ 1.

NB (input) INTEGER

NB specifies the column block size of the matrices op(B) and C. NB ≥ 1.

KB (input) INTEGER

KB specifies the column block size of the matrix op(A) and the row block size of the matrix op(B). KB ≥ 1.

ALPHA (input) REAL/COMPLEX

ALPHA specifies the scalar alpha.

A (input) REAL/COMPLEX array of dimension (LDA, Lx), where Lx is Kq when TRANSA = 'N', and is Mq otherwise.

On entry with TRANSA = 'N', the leading Mp-by-Kq part of the array A must contain the (local) matrix A, otherwise the leading Kp-by-Mq part of the array A must contain the (local) matrix A.

LDA (input) INTEGER

The leading dimension of the (local) array A. When TRANSA = 'N', LDA $\geq$ MAX(1, Mp), otherwise LDA $\geq$ MAX(1, Kp).

B (input) REAL/COMPLEX array of DIMENSION (LDB, Lx), where Lx is Nq when TRANSB = 'N', and is Kq otherwise.

On entry with TRANSB = 'N', the leading Kp-by-Nq part of the array B must contain the (local) matrix B, otherwise the leading Np-by-Kq part of the array B must contain the (local) matrix B.

2. PSMMA/PCMMA

```
      SUBROUTINE PSMMA( TRANSA, TRANSB, M, N, K, MB, NB, KB, ALPHA,
     $                  A, LDA, B, LDB, BETA, C, LDC, IMROW, IMCOL,
     $                  WORK, IWORK )
      CHARACTER*1       TRANSA, TRANSB
      INTEGER           M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER           IMROW, IMCOL
      REAL              ALPHA, BETA
      REAL              A( LDA, * ), B( LDB, * ), C( LDC, * )
      REAL              WORK( * )
      INTEGER           IWORK( * )

      SUBROUTINE PCMMA( TRANSA, TRANSB, M, N, K, MB, NB, KB, ALPHA,
     $                  A, LDA, B, LDB, BETA, C, LDC, IMROW, IMCOL,
     $                  WORK, IWORK )
      CHARACTER*1       TRANSA, TRANSB
      INTEGER           M, N, K, MB, NB, KB, LDA, LDB, LDC
      INTEGER           IMROW, IMCOL
      COMPLEX           ALPHA, BETA
      COMPLEX           A( LDA, * ), B( LDB, * ), C( LDC, * )
      COMPLEX           WORK( * )
      INTEGER           IWORK( * )
```

Purpose

PSMMA/PCMMA performs one of the following matrix-matrix operations,

$$\mathbf{C} \Leftarrow \text{alpha} \cdot op(\mathbf{A}) \cdot op(\mathbf{B}) + \text{beta} \cdot \mathbf{C}$$

where op(X) is one of op(X) = X, op(X) = (X)^T, or op(X) = (X)^H, alpha and beta are scalars, and A, B and C are matrices, with op(A) an M-by-K matrix, op(B) a K-by-N matrix, and C an M-by-N matrix.

Arguments

TRANSA (input) CHARACTER*1

TRANSA specifies the form of op(A) to be used in the matrix multiplication.

TRANSA = 'N': op(A) = A;

TRANSA = 'T': op(A) = A^T;

TRANSA = 'C': op(A) = A^H.

NG = $\lceil$ N / NB $\rceil$,

KG = $\lceil$ K / KB $\rceil$,

Lx : It is a local portion of L owned by me, where me is a row or a column processor position on a 2-dimensional processor template, and $0 \leq$ me $<$ p if me is a row processor position, or $0 \leq$ me $<$ q otherwise.

The value is determined by L, LB, x, and me, src, where LB is a block size, and src is a starting processor. (The actual routine is `NUMROC`.) And L and LB are replaced by M and MB, N and NB, or K and KB, respectively, and x is replaced by either p (= P = NPROW) or q (= Q = NPCOL).

Lx $\leq$ Lx0 = $\lceil$ L / (LB $\cdot$ x) $\rceil$ $\cdot$ LB.

1. Notes

1. The specifications that follow give the calling sequence, purpose, descriptions, and memory requirements of each PUMMA routine.

2. Specifications of pairs of real and complex routines have been merged (for example PSMMA/PCMMA).

3. Specifications are given only for *single precision* routines. To adapt them for the double precision version of the software, simply interpret REAL as DOUBLE PRECISION, COMPLEX as COMPLEX*16 (or DOUBLE COMPLEX), and the initial two letters PS- and PC- of the PUMMA routine names as PD- and PZ-.

4. Specification are arranged in alphabetical order of the real routine name.

5. The PUMMA routines rely either on the PICL communication library or the BLACS communication subroutines. The PICL version of the package is dedicated to the Intel Paragon and Intel Delta computers. Forced type communications and even-odd rotating scheme are used by the PICL version. The BLACS version uses non-forced type communication and simultaneous rotating schemes. The two versions need different amount of working space, and both requirements are specified in this document.

6. The PUMMA package includes two different routines, `psmnna/pcsmma` and `psmnnb/pcsmmb` to compute $\mathbf{C} \Leftarrow \text{alpha} \cdot op(\mathbf{A}) \cdot op(\mathbf{B}) + \text{beta} \cdot \mathbf{C}$. The former broadcasts the matrix A and rotates the matrix B, and the latter broadcasts the matrix B and rotates the matrix A. The last character of the routine name specifies which matrix is broadcast. These routines perform differently depending on the arrangement of the process grid ($P \geq Q$ or $P < Q$). Moreover these two routines require different amount of working space.

7. The first blocks of the matrices, A, B, and C should be located on the same processor, P(IMROW, IMCOL) ($0 \leq \text{IMROW} < P$, $0 \leq \text{IMCOL} < Q$).

8. The user should check the amount of memory (working space) required by the PUMMA routines before calling them.

9. The following parameters are used throughout the text in order to simplify the computation of the working space.

   P = NPROW = number of row processors,
   Q = NPCOL = number of column processors,
   LCM = the least common multiple of P and Q,
   MG = $\lceil$ M / MB $\rceil$,

PUMMA REFERENCE MANUAL (VERSION 2.0)

Jaeyoung Choi

Jack J. Dongarra

Antoine Petitet

David W. Walker

Abstract

The PUMMA (Parallel Universal Matrix Multiplication Algorithms) package provides a set of matrix multiplication subroutines for distributed memory concurrent computers. This package includes not only the non-transposed matrix multiplication $\mathbf{C} = \alpha\mathbf{AB} + \beta\mathbf{C}$, but also transposed matrix multiplications, $\mathbf{C} = \alpha\mathbf{A}'\mathbf{B} + \beta\mathbf{C}$, $\mathbf{C} = \alpha\mathbf{AB}' + \beta\mathbf{C}$, and $\mathbf{C} = \alpha\mathbf{A}'\mathbf{B}' + \beta\mathbf{C}$, where $\mathbf{X}' = \mathbf{X}^T$ or $\mathbf{X}^H$. In addition, the PUMMA package provides a matrix transpose routine $\mathbf{C} = \mathbf{A}' + \beta\mathbf{C}$.

These routines assume that the data is distributed on the processes based on a block cyclic data distribution, The routines perform efficiently for a wide range of process configurations and block sizes.

The PUMMA together provide the same functionality as the Level 3 BLAS routine SGEMM/DGEMM/CGEMM/ZGEMM. This document describes the calling sequences of the PUMMA routines. The algorithms used are described in [1,2].

References

[1] J. Choi, J. J. Dongarra, and D. W. Walker. Parallel Matrix Transpose Algorithms on Distributed Memory Concurrent Computers. Technical Report TM-12309, Oak Ridge National Laboratory, Mathematical Sciences Section, October 1993.

[2] J. Choi, J. J. Dongarra, and D. W. Walker. PUMMA : Parallel universal matrix multiplication algorithms on distributed memory concurrent computers. Technical Report TM-12252, Oak Ridge National Laboratory, Mathematical Sciences Section, August 1993.

Contents

1 Notes 1
2 PSMMA/PCMMA 3
3 PSMNNA/PCMNNA 8
4 PSMNNB/PCMNNB 11
5 PSMNTC/PCMNTC 14
6 PSMTNC/PCMTNC 17
7 PSTRANS/PCTRANS 20

ORNL/TM-12494

Engineering Physics and Mathematics Division

Mathematical Sciences Section

PUMMA REFERENCE MANUAL (VERSION 2.0)

Jaeyoung Choi [§]
Jack J. Dongarra [§†]
Antoine Petitet [†]
David W. Walker [§]

[§] Mathematical Sciences Section
Oak Ridge National Laboratory
P.O. Box 2008, Bldg. 6012
Oak Ridge, TN 37831-6367

[†] Department of Computer Science
University of Tennessee at Knoxville
107 Ayres Hall
Knoxville, TN 37996-1301

Date Published: March 1994

Research was supported by the Applied Mathematical Sciences Research Program of the Office of Energy Research, U.S. Department of Energy, by the Defense Advanced Research Projects Agency under contract DAAL03-91-C-0047, administered by the Army Research Office, and by the Center for Research on Parallel Computing

Prepared by the
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831
managed by
Martin Marietta Energy Systems, Inc.
for the
U.S. DEPARTMENT OF ENERGY
under Contract No. DE-AC05-84OR21400