**BLAS Level 3 Routines**

Level 3 routines mainly deals with matrix-matrix operations.

# Multiplications and Additions

The following BLAS level 3 routines are relevant to matrix multiplications and additions.

* SGEMM, DGEMM, CGEMM, ZGEMM
* SSYMM, DSYMM, CSYMM, ZSYMM
* CHEMM, ZHEMM

The xGEMM routines are for Combined Matrix Multiplication and Addition for General Matrices, Their Transposes, or Conjugate Transposes.

The xSYMM/xHEMM routines are for Matrix-Matrix Product Where One Matrix is Real or Complex Symmetric or Complex Hermitian.

C ← αAB +βC

C ← αATB +βC

C ← αAHB +βC

The syntax to use these functions in different languages are summarized below.

**For general matrix:**

| **Language** | **Syntax** |
| --- | --- |
| **Fortran** | CALL SGEMM | DGEMM | CGEMM | ZGEMM (*transa*, *transb*, *l*, *n*, *m*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*) |
| **C and C++** | sgemm | dgemm | cgemm | zgemm (*transa*, *transb*, *l*, *n*, *m*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*); |
| **CBLAS** | cblas\_sgemm | cblas\_dgemm | cblas\_cgemm | cblas\_zgemm (*cblas\_layout*, *cblas\_transa*, *cblas\_transb*, *l*, *n*, *m*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*); |

**For Real or Complex Symmetric / Complex Hermitian matrix:**

| **Language** | **Syntax** |
| --- | --- |
| **Fortran** | CALL SSYMM | DSYMM | CSYMM | ZSYMM | CHEMM | ZHEMM (*side*, *uplo*, *m*, *n*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*) |
| **C and C++** | ssymm | dsymm | csymm | zsymm | chemm | zhemm (*side*, *uplo*, *m*, *n*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*); |
| **CBLAS** | cblas\_ssymm | cblas\_dsymm | cblas\_csymm | cblas\_zsymm | cblas\_chemm | cblas\_zhemm (*cblas\_layout*, *cblas\_side*, *cblas\_uplo*, *m*, *n*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*); |

Some arguments are explained here:

cblas\_layout

indicates whether the input and output matrices are stored in row major order or column major order, where:

If cblas\_layout = CblasRowMajor, the matrices are stored in row major order.

If cblas\_layout = CblasColMajor, the matrices are stored in column major order.

Specified as: an object of enumerated type CBLAS\_LAYOUT.

side

indicates whether the triangular matrix A is located to the left or right of rectangular matrix B in the system of equations, where:

If side = 'L', A is to the left of B, resulting in solution 1, 2, or 5.

If side = 'R', A is to the right of B, resulting in solution 3, 4, or 6.

Specified as: a single character. It must be 'L' or 'R'.

cblas\_side

indicates whether matrix A is located to the left or right of rectangular matrix B in the equation used for this computation, where:

If cblas\_side = CblasLeft, A is to the left of B, resulting in solution 1, 2, or 5.

If cblas\_side = CblasRight, A is to the right of B, resulting in solution 3, 4, or 6.

Specified as: an object of enumerated type CBLAS\_SIDE.

uplo

indicates whether matrix A is an upper or lower triangular matrix, where:

If uplo = 'U', A is an upper triangular matrix.

If uplo = 'L', A is a lower triangular matrix.

Specified as: a single character. It must be 'U' or 'L'.

cblas\_uplo

indicates whether matrix A is an upper or lower triangular matrix, where:

If cblas\_uplo = CblasUpper, A is an upper triangular matrix.

If cblas\_uplo = CblasLower, A is a lower triangular matrix.

Specified as: an object of enumerated type CBLAS\_UPLO.

transa

indicates the form of matrix A used in the system of equations, where:

If transa = 'N', A is used, resulting in solution 1 or 3.

If transa = 'T', AT is used, resulting in solution 2 or 4.

If transa = 'C', AH is used, resulting in solution 5 or 6.

Specified as: a single character. It must be 'N', 'T', or 'C'.

cblas\_transa

indicates the form of matrix A to use in the computation, where:

If cblas\_transa = CblasNoTrans, A is used, resulting in solution 1 or 3.

If cblas\_transa = CblasTrans, AT is used, resulting in solution 2 or 4.

If cblas\_transa = CblasConjTrans, AH is used, resulting in solution 5 or 6.

Specified as: an object of enumerated type CBLAS\_TRANSPOSE.

diag

indicates the characteristics of the diagonal of matrix A, where:

If diag = 'U', A is a unit triangular matrix.

If diag = 'N', A is not a unit triangular matrix.

Specified as: a single character. It must be 'U' or 'N'.

cblas\_diag

indicates the characteristics of the diagonal of matrix A, where:

If diag = CblasUnit, A is a unit triangular matrix.

If diag = CblasNonUnit A is not a unit triangular matrix.

Specified as: an object of enumerated type CBLAS\_DIAG.

m,

is the number of rows in rectangular matrix B, and:

If side = 'L', m is the order of triangular matrix A.

Specified as: an integer, where:

If side = 'L', 0 ≤ m ≤ lda and m ≤ ldb.

If side = 'R', 0 ≤ m ≤ ldb.

n

is the number of columns in rectangular matrix B, and:

If side = 'R', n is the order of triangular matrix A.

Specified as: an integer; n ≥ 0, and:

If side = 'R', n ≤ lda.

alpha

is the scalar α. Specified as: a number of the data type for matrix entries.

a

is the triangular matrix A, of which only the upper or lower triangular portion is used, where:

If side = 'L', A is order m.

If side = 'R', A is order n.

Specified as: a two-dimensional array, containing numbers of the data type for matrix entries, where:

If side = 'L', its size must be lda by (at least) m.

If side = 'R', it size must be lda by (at least) n.

lda

is the leading dimension of the array specified for a.

Specified as: an integer; lda > 0, and:

If side = 'L', lda ≥ m.

If side = 'R', lda ≥ n.

b

is the m by n rectangular matrix B, which contains the right-hand sides of the triangular system to be solved.

Specified as: an ldb by (at least) n array, containing numbers of the data type indicated in Table 1.

ldb

is the leading dimension of the array specified for b.

Specified as: an integer; ldb > 0 and ldb ≥ m.

Some examples demonstrating these routines are given. Some examples are located at **test-blas/level-3** in the repository.

**SGEMM Example:**

TRANSA TRANSB L N M ALPHA A LDA B LDB BETA C LDC

| | | | | | | | | | | | |

CALL SGEMM( 'N' , 'N' , 6 , 4 , 5 , 1.0 , A , 8 , B , 6 , 2.0 , C , 7 )

| 1.0 2.0 -1.0 -1.0 4.0 |

| 2.0 0.0 1.0 1.0 -1.0 |

| 1.0 -1.0 -1.0 1.0 2.0 |

A = | -3.0 2.0 2.0 2.0 0.0 |

| 4.0 0.0 -2.0 1.0 -1.0 |

| -1.0 -1.0 1.0 -3.0 2.0 |

| . . . . . |

| . . . . . |

| 1.0 -1.0 0.0 2.0 |

| 2.0 2.0 -1.0 -2.0 |

B = | 1.0 0.0 -1.0 1.0 |

| -3.0 -1.0 1.0 -1.0 |

| 4.0 2.0 -1.0 1.0 |

| . . . . |

| 0.5 0.5 0.5 0.5 |

| 0.5 0.5 0.5 0.5 |

| 0.5 0.5 0.5 0.5 |

C = | 0.5 0.5 0.5 0.5 |

| 0.5 0.5 0.5 0.5 |

| 0.5 0.5 0.5 0.5 |

| . . . . |

Output:

| 24.0 13.0 -5.0 3.0 |

| -3.0 -4.0 2.0 4.0 |

| 4.0 1.0 2.0 5.0 |

C = | -2.0 6.0 -1.0 -9.0 |

| -4.0 -6.0 5.0 5.0 |

| 16.0 7.0 -4.0 7.0 |

| . . . . |

**SSYMM Example:**

SIDE UPLO M N ALPHA A LDA B LDB BETA C LDC

| | | | | | | | | | | |

CALL SSYMM( 'L' , 'U' , 5 , 4 , 2.0 , A , 8 , B , 6 , 1.0 , C , 5 )

| 1.0 2.0 -1.0 -1.0 4.0 |

| . 0.0 1.0 1.0 -1.0 |

| . . -1.0 1.0 2.0 |

A = | . . . 2.0 0.0 |

| . . . . -1.0 |

| . . . . . |

| . . . . . |

| . . . . . |

| 1.0 -1.0 0.0 2.0 |

| 2.0 2.0 -1.0 -2.0 |

B = | 1.0 0.0 -1.0 1.0 |

| -3.0 -1.0 1.0 -1.0 |

| 4.0 2.0 -1.0 1.0 |

| . . . . |

| 23.0 12.0 -6.0 2.0 |

| -4.0 -5.0 1.0 3.0 |

C = | 5.0 6.0 -1.0 -4.0 |

| -4.0 1.0 0.0 -5.0 |

| 8.0 -4.0 -2.0 13.0 |

Output:

| 69.0 36.0 -18.0 6.0 |

| -12.0 -15.0 3.0 9.0 |

C = | 15.0 18.0 -3.0 -12.0 |

| -12.0 3.0 0.0 -15.0 |

| 8.0 -20.0 -2.0 35.0 |

**CHEMM Example:**

SIDE UPLO M N ALPHA A LDA B LDB BETA C LDC

| | | | | | | | | | | |

CALL CHEMM( 'R' , 'L' , 2 , 3 , ALPHA , A , 4 , B , 3 , BETA , C , 5 )

ALPHA = (2.0, 3.0)

BETA = (1.0, 6.0)

| (1.0, . ) . . |

A = | (3.0, 2.0) (4.0, . ) . |

| (-1.0, 6.0) (1.0, 4.0) (2.0, . ) |

| . . . |

| (1.0, 1.0) (-3.0, 2.0) (3.0, 3.0) |

B = | (2.0, 6.0) (4.0, 5.0) (-1.0, 4.0) |

| . . . |

| (13.0, 6.0) (-18.0, 6.0) (10.0, 7.0) |

| (-11.0, 8.0) (11.0, 1.0) (-4.0, 2.0) |

C = | . . . |

| . . . |

| . . . |

# Rank-K/Rank-2K Update of a Real or Complex Symmetric or a Complex Hermitian Matrix

The following BLAS level 3 routines are relevant:

* SSYRK, DSYRK, CSYRK, ZSYRK, CHERK, ZHERK
* SSYR2K, DSYR2K, CSYR2K, ZSYR2K, CHER2K, ZHER2K

The rank-k updates compute one of the following rank-k updates, where matrix C is stored in upper or lower storage mode.

C ← αAAT+βC C ← αATA+βC

C ← αAAH+βC C ← αAHA+βC

The rank-2k updates involves two different matrices, where matrix **C** is stored in upper or lower storage mode.

C ← αABT+αBAT+βC

C ← αATB+αBTA+βC

The syntax to use these functions in different languages are summarized below.

**For rank-k update:**

| **Language** | **Syntax** |
| --- | --- |
| **Fortran** | CALL SSYRK | DSYRK | CSYRK | ZSYRK | CHERK | ZHERK (*uplo*, *trans*, *n*, *k*, *alpha*, *a*, *lda*, *beta*, *c*, *ldc*) |
| **C and C++** | ssyrk | dsyrk | csyrk | zsyrk | cherk | zherk (*uplo*, *trans*, *n*, *k*, *alpha*, *a*, *lda*, *beta*, *c*, *ldc*); |
| **CBLAS** | cblas\_ssyrk | cblas\_dsyrk | cblas\_csyrk | cblas\_zsyrk | cblas\_cherk | cblas\_zherk (*cblas\_layout*, *cblas\_uplo*, *cblas\_trans*, *n*, *k*, *alpha*, *a*, *lda*, *beta*, *c*, *ldc*); |

**For rank-2k update:**

| **Language** | **Syntax** |
| --- | --- |
| **Fortran** | CALL SSYR2K | DSYR2K | CSYR2K | ZSYR2K | CHER2K | ZHER2K (*uplo*, *trans*, *n*, *k*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*) |
| **C and C++** | ssyr2k | dsyr2k | csyr2k | zsyr2k | cher2k | zher2k (*uplo*, *trans*, *n*, *k*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*); |
| **CBLAS** | cblas\_ssyr2k | cblas\_dsyr2k | cblas\_csyr2k | cblas\_zsyr2k | cblas\_cher2k | cblas\_zher2k (*cblas\_layout*, *cblas\_uplo*, *cblas\_trans*, *n*, *k*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*); |

Some examples demonstrating these routines are given. The examples are located at **test-blas/level-3** in the repository.

**SSYRK Example:**

UPLO TRANS N K ALPHA A LDA BETA C LDC

| | | | | | | | | |

CALL SSYRK( 'U' , 'N' , 8 , 2 , 1.0 , A , 9 , 1.0 , C , 10 )

| 0.0 8.0 |

| 1.0 9.0 |

| 2.0 10.0 |

| 3.0 11.0 |

A = | 4.0 12.0 |

| 5.0 13.0 |

| 6.0 14.0 |

| 7.0 15.0 |

| . . |

| 0.0 1.0 3.0 6.0 10.0 15.0 21.0 28.0 |

| . 2.0 4.0 7.0 11.0 16.0 22.0 29.0 |

| . . 5.0 8.0 12.0 17.0 23.0 30.0 |

| . . . 9.0 13.0 18.0 24.0 31.0 |

C = | . . . . 14.0 19.0 25.0 32.0 |

| . . . . . 20.0 26.0 33.0 |

| . . . . . . 27.0 34.0 |

| . . . . . . . 35.0 |

| . . . . . . . . |

| . . . . . . . . |

Output:

| 64.0 73.0 83.0 94.0 106.0 119.0 133.0 148.0 |

| . 84.0 96.0 109.0 123.0 138.0 154.0 171.0 |

| . . 109.0 124.0 140.0 157.0 175.0 194.0 |

| . . . 139.0 157.0 176.0 196.0 217.0 |

C = | . . . . 174.0 195.0 217.0 240.0 |

| . . . . . 214.0 238.0 263.0 |

| . . . . . . 259.0 286.0 |

| . . . . . . . 309.0 |

| . . . . . . . . |

| . . . . . . . . |

**CSYR2K Example:**

UPLO TRANS N K ALPHA A LDA B LDB BETA C LDC

| | | | | | | | | | | |

CALL CSYR2K( 'L' , 'N' , 3 , 5 , ALPHA , A , 3 , B , 3 , BETA , C , 4 )

ALPHA = (1.0, 1.0)

BETA = (1.0, 1.0)

| (2.0, 5.0) (3.0, 2.0) (4.0, 1.0) (1.0, 7.0) (0.0, 0.0) |

A = | (3.0, 3.0) (8.0, 5.0) (2.0, 5.0) (2.0, 4.0) (1.0, 2.0) |

| (1.0, 3.0) (2.0, 1.0) (6.0, 5.0) (3.0, 2.0) (2.0, 2.0) |

| (1.0, 5.0) (6.0, 2.0) (3.0, 1.0) (2.0, 0.0) (1.0, 0.0) |

B = | (2.0, 4.0) (7.0, 5.0) (2.0, 5.0) (2.0, 4.0) (0.0, 0.0) |

| (3.0, 5.0) (8.0, 1.0) (1.0, 5.0) (1.0, 0.0) (1.0, 1.0) |

| (2.0, 3.0) . . |

C = | (1.0, 9.0) (3.0, 3.0) . |

| (4.0, 5.0) (6.0, 7.0) (8.0, 3.0) |

| . . . |

Output:

| (-101.0, 121.0) . . |

C = | (-182.0, 192.0) (-274.0, 248.0) . |

| (-98.0, 146.0) (-163.0, 205.0) (-151.0, 115.0) |

| . . . |

# Triangular Matrix-Matrix Product

The following BLAS level 3 routines are relevant:

* STRMM, DTRMM, CTRMM, and ZTRMM

STRMM and DTRMM compute one of the following matrix-matrix products, using the scalar α, rectangular matrix B, and triangular matrix A or its transpose:

| **Matrix-Matrix Products Computed by STRMM and DTRMM** | |
| --- | --- |
| 1. ***B***←α***AB*** | 3. ***B***←α***BA*** |
| 2. ***B***←α***A***T***B*** | 4. ***B***←α***BA***T |

| **Matrix-Matrix Products Computed by CTRMM and ZTRMM** | | |
| --- | --- | --- |
| 1. ***B***←α***AB*** | 3. ***B***←α***BA*** | 5. ***B***←α***A***H***B*** |
| 2. ***B***←α***A***T***B*** | 4. ***B***←α***BA***T | 6. ***B***←α***BA***H |

The syntax to use these functions in different languages are summarized below.

| **Language** | **Syntax** |
| --- | --- |
| **Fortran** | CALL STRMM | DTRMM | CTRMM | ZTRMM (*side*, *uplo*, *transa*, *diag*, *m*, *n*, *alpha*, *a*, *lda*, *b*, *ldb*) |
| **C and C++** | strmm | dtrmm | ctrmm | ztrmm (*side*, *uplo*, *transa*, *diag*, *m*, *n*, *alpha*, *a*, *lda*, *b*, *ldb*); |
| **CBLAS** | cblas\_strmm | cblas\_dtrmm | cblas\_ctrmm | cblas\_ztrmm (*cblas\_layout*, *cblas\_side*, *cblas\_uplo*, *cblas\_transa*, *cblas\_diag*, *m*, *n*, *alpha*, *a*, *lda*, *b*, *ldb*); |

Some examples demonstrating these routines are given. They can be found in **test-blas/level-3** in the repository.

**STRMM Example:**

SIDE UPLO TRANSA DIAG M N ALPHA A LDA B LDB

| | | | | | | | | | |

CALL STRMM( 'L' , 'U' , 'N' , 'N' , 5 , 3 , 1.0 , A , 7 , B , 6 )

| 3.0 -1.0 2.0 2.0 1.0 |

| . -2.0 4.0 -1.0 3.0 |

| . . -3.0 0.0 2.0 |

A = | . . . 4.0 -2.0 |

| . . . . 1.0 |

| . . . . . |

| . . . . . |

| 2.0 3.0 1.0 |

| 5.0 5.0 4.0 |

B = | 0.0 1.0 2.0 |

| 3.0 1.0 -3.0 |

| -1.0 2.0 1.0 |

| . . . |

Output:

| 6.0 10.0 -2.0 |

| -16.0 -1.0 6.0 |

B = | -2.0 1.0 -4.0 |

| 14.0 0.0 -14.0 |

| -1.0 2.0 1.0 |

| . . . |

# Solution of Triangular Systems of Equations with Multiple Right-Hand Sides

The following BLAS level 3 routines are relevant:

* STRSM, DTRSM, CTRSM, and ZTRSM

STRSM and DTRSM perform one of the following solves for a triangular system of equations with multiple right-hand sides, using scalar α, rectangular matrix B, and triangular matrix A or its transpose:

|  |
| --- |
|  |
| **Solution** | **Equation** |
| 1. ***B***←α(***A***-1)***B*** | ***AX*** = α***B*** |
| 2. ***B***←α(***A***-T)***B*** | ***A***T***X*** = α***B*** |
| 3. ***B***←α***B***(***A***-1) | ***XA*** = α***B*** |
| 4. ***B***←α***B***(***A***-T) | ***XA***T = α***B*** |

|  |
| --- |
|  |
| **Solution** | **Equation** |
| 1. ***B***←α(***A***-1)***B*** | ***AX*** = α***B*** |
| 2. ***B***←α(***A***-T)***B*** | ***A***T***X*** = α***B*** |
| 3. ***B***←α***B***(***A***-1) | ***XA*** = α***B*** |
| 4. ***B***←α***B***(***A***-T) | ***XA***T = α***B*** |
| 5. ***B***←α(***A***-H)***B*** | ***A***H***X*** = α***B*** |
| 6. ***B***←α***B***(***A***-H) | ***XA***H = α***B*** |

The syntax to use these functions in different languages are summarized below.

| **Language** | **Syntax** |
| --- | --- |
| **Fortran** | CALL STRSM | DTRSM | CTRSM | ZTRSM (*side*, *uplo*, *transa*, *diag*, *m*, *n*, *alpha*, *a*, *lda*, *b*, *ldb*) |
| **C and C++** | strsm | dtrsm | ctrsm | ztrsm (*side*, *uplo*, *transa*, *diag*, *m*, *n*, *alpha*, *a*, *lda*, *b*, *ldb*); |
| **CBLAS** | cblas\_strsm | cblas\_dtrsm | cblas\_ctrsm | cblas\_ztrsm (*cblas\_layout*, *cblas\_side*, *cblas\_uplo*, *cblas\_transa*, *cblas\_diag*, *m*, *n*, *alpha*, *a*, *lda*, *b*, *ldb*); |

Some examples demonstrating these routines are given. They can be found in **test-blas/level-3** in the repository.

**STRSM Example:**

SIDE UPLO TRANSA DIAG M N ALPHA A LDA B LDB

| | | | | | | | | | |

CALL STRSM( 'L' , 'U' , 'N' , 'N' , 5 , 3 , 1.0 , A , 7 , B , 6 )

| 3.0 -1.0 2.0 2.0 1.0 |

| . -2.0 4.0 -1.0 3.0 |

| . . -3.0 0.0 2.0 |

A = | . . . 4.0 -2.0 |

| . . . . 1.0 |

| . . . . . |

| . . . . . |

| 6.0 10.0 -2.0 |

| -16.0 -1.0 6.0 |

B = | -2.0 1.0 -4.0 |

| 14.0 0.0 -14.0 |

| -1.0 2.0 1.0 |

| . . . |

Output:

| 2.0 3.0 1.0 |

| 5.0 5.0 4.0 |

B = | 0.0 1.0 2.0 |

| 3.0 1.0 -3.0 |

| -1.0 2.0 1.0 |

| . . . |

**STRSM Example:**

SIDE UPLO TRANSA DIAG M N ALPHA A LDA B LDB

| | | | | | | | | | |

CALL CTRSM( 'R' , 'L' , 'N' , 'N' , 3 , 5 , ALPHA , A , 7 , B , 4 )

ALPHA = (1.0, 0.0)

| (2.0, -3.0) . . . . |

| (2.0, -4.0) (3.0, -1.0) . . . |

| (2.0, 2.0) (1.0, 2.0) (1.0, 1.0) . . |

A = | (0.0, 0.0) (3.0, -1.0) (0.0, -1.0) (-2.0, 1.0) . |

| (2.0, 2.0) (4.0, 0.0) (-1.0, 2.0) (2.0, -4.0) (-1.0, -4.0) |

| . . . . . |

| . . . . . |

| (22.0, -41.0) (7.0, -26.0) (9.0, 0.0) (-15.0, -3.0) (-15.0, 8.0) |

B = | (29.0, -18.0) (24.0, -10.0) (9.0, 6.0) (-12.0, -24.0) (-19.0, -8.0) |

| (-15.0, 2.0) (-3.0, -21.0) (-2.0, 4.0) (-4.0, -12.0) (-10.0, -6.0) |

| . . . . . |

Output:

| (3.0, 0.0) (4.0, 0.0) (-1.0, -2.0) (-1.0, -1.0) (-1.0, -4.0) |

B = | (2.0, -1.0) (1.0, 2.0) (-1.0, -3.0) (0.0, 2.0) (3.0, -4.0) |

| (-2.0, 1.0) (-1.0, -3.0) (-3.0, 1.0) (0.0, 0.0) (2.0, -2.0) |

| . . . . . |