C Interface Conventions for LAPACK Routines

The C interfaces are implemented for most of the Intel MKL LAPACK driver and computational routines.

Function Prototypes

Intel MKL supports four distinct floating-point precisions. Each corresponding prototype looks similar, usually differing only in the data type. C interface LAPACK function names follow the form `<?><name>`, where `<?>` is:

- `LAPACKE_s` for float
- `LAPACKE_d` for double
- `LAPACKE_c` for lapack_complex_float
- `LAPACKE_z` for lapack_complex_double

A specific example follows. To solve a system of linear equations with a packed Cholesky-factored Hermitian positive-definite matrix with complex precision, use the following:

lapack_int LAPACKE_**c**pptrs(int *matrix_layout*, char *uplo*, lapack_int *n*, lapack_int *nrhs*, const **lapack_complex_float*** *ap*, **lapack_complex_float*** *b*, lapack_int *ldb*);

Workspace Arrays

In contrast to the Fortran interface, the LAPACK C interface omits workspace parameters because workspace is allocated during runtime and released upon completion of the function operation.

If you prefer to allocate workspace arrays yourself, the LAPACK C interface provides alternate interfaces with *work* parameters. The name of the alternate interface is the same as the LAPACK C interface with `_work` appended. For example, the syntax for the singular value decomposition of a real bidiagonal matrix is:

Fortran:	call sbdsdc (uplo, compq, n, d, e, u, ldu, vt, ldvt, q, iq, **work**, **iwork**, info)
C LAPACK interface:	lapack_int LAPACKE_sbdsdc (int matrix_layout, char uplo, char compq, lapack_int n, float* d, float* e, float* u, lapack_int ldu, float* vt, lapack_int ldvt, float* q, lapack_int* iq);
Alternate C LAPACK interface with *work* parameters:	lapack_int LAPACKE_sbdsdc_work(int matrix_layout, char uplo, char compq, lapack_int n, float* d, float* e, float* u, lapack_int ldu, float* vt, lapack_int ldvt, float* q, lapack_int* iq, float* **work**, lapack_int* **iwork**);

See the *install_dir*`/include/mkl_lapacke.h` file for the full list of alternative C LAPACK interfaces.

The Intel MKL Fortran-specific documentation contains details about workspace arrays.

Mapping Fortran Data Types against C Data Types

Fortran Data Types vs. C Data Types

INTEGER	lapack_int
LOGICAL	lapack_logical
REAL	float
DOUBLE PRECISION	double
COMPLEX	lapack_complex_float
COMPLEX*16/DOUBLE COMPLEX	lapack_complex_double
CHARACTER	char

C Type Definitions

You can find type definitions specific to Intel MKL such as `MKL_INT`, `MKL_Complex8`, and `MKL_Complex16` in *`install_dir`*`/mkl_types.h`.

C types

```
#ifndef lapack_int
#define lapack_int MKL_INT
#endif

#ifndef lapack_logical
#define lapack_logical lapack_int
#endif
```

Complex Type Definitions

Complex type for single precision:

```
#ifndef lapack_complex_float
#define lapack_complex_float MKL_Complex8
#endif
```

Complex type for double precision:

```
#ifndef lapack_complex_double
#define lapack_complex_double MKL_Complex16
#endif
```

Matrix Layout Definitions

```
#define LAPACK_ROW_MAJOR 101
#define LAPACK_COL_MAJOR 102
```

See Matrix Layout for LAPACK Routines (/node/26131c10-423a-480a-8a09-52ac6cdecc22) for an explanation of row-major order and column-major order storage.

Error Code
Definitions

```
#define LAPACK_WORK_MEMORY_ERROR -1010 /* Failed to allocate memory
for a working array */
#define LAPACK_TRANSPOSE_MEMORY_ERROR -1011 /* Failed to allocate memor
y
for transposed matrix */
```

Parent topic: LAPACK Routines (/node/6a025ea8-0cbd-42ac-9c75-1a4022663879)