Erlang BLAS 1.1.0 Documentation

Tanguy Losseau

August 2023

1 Introduction

This project, funded by the Erlang Ecosystem Foundation (https://erlef.org/), was made possible thanks to Peerst Stritzinger (https://www.stritzinger.com/) and aims to bring the efficiency of the BLAS-LAPACKE library to Erlang.

It can be used by any platform providing a BLAS-LAPACKE library, such as the GRISP platform (https://www.grisp.org/).

This document is a reference for the Erlang BLAS wrapper, and is not a reference of BLAS-LAPACKE. Such references can be found at:

- netlib: https://netlib.org/
- IBM: https://www.ibm.com/docs/en
- intel: https://www.intel.com/content/www/us/en/content-details/671183/developer-reference-for-intel-math-kernel-library-intel-mkl-11-3-c.html?wapkw=BLAS%20mlk

Finally, the project maintainer can be contacted at losseautanguy@gmail.com.

2 Reference

The BLAS library exports the following functions:

```
[run/1, run/2, new/1, new/2, shift/2, copy/2, to_bin/1,
to_bin/2, to_list/2, predictor/0]
```

.

The following record will be later referenced as c_binary:

```
-record(c_binary, {size, offset, resource}).
```

Finally, possible values of blas_name are provided in section "Supported BLAS functions".

2.1 run

2.1.1 run/1

```
run(Tuple)
    Tuple: {blas_name, Arg0, ..., ArgN}
```

Same as run/2. The first time it is executed, runs a benchmark of dgemm (see predictor);BLAS execution duration is then predicted and forwarded to run/2. Due to their diversity/complexity, LAPACKE functions are always sent to a dirty scheduler for one millisecond.

2.1.2 run/2

```
run(Tuple, Scheduling)
    Tuple: {blas\_name, Arg0, ..., ArgN}
    Scheduling: Integer | dirty | clean
```

Tuple groups the name of requested BLAS functions, and its arguments. The section "Representing BLAS-LAPACKE types in Erlang" describes how to construct the latter. Scheduling is either:

- dirty: schedule for 1.5 ms on a dirty scheduler.
- clean: schedule for 0.5 ms on a clean scheduler.
- Integer: the percentage of 1ms expected to be used. If inferior to 100, clean scheduling will be used; otherwise dirty scheduling will be used.

If the blas_name executed without error, returns ok. It might raise the following exceptions:

- "Unknown blas." if the blas_name is not recognised.
- "Array overflow." if one of the inputs arrays is too small. Currently, only BLAS functions check for arrays overflow.
- "Invalid number of arguments." if too many arguments are present in the Tuple.
- "Could not translate argument I." if ArgI of Tuple not be read.

2.2 new

This function is used to create a c_binary. The BLAS library executes in place and require mutable arrays. Thought it could possible to do this with erlang binaries, it is safer not to; instead, a nif resource is used and stored in a c_binary.

2.2.1 new/1

```
new(Type)
    Type: Integer | Binary
```

If Type is Integer, allocates a c_binary of given byte size.
If Type is Binary, copy the input Binary into a c_binary.
Returns a c_binary.

2.2.2 new/2

```
new(Encoding, List)
    Encoding: int32 | int64 | s | float32 | d |float64
                | c | complex64 | z | complex128
```

List is a list of numbers to write in a new c_binary.
Encoding indicates how the numbers should be encoded:

- int32: integers of 32 bits.
- int64: integers of 32 bits.
- s, float32: floats of 32 bits.
- d, float64: floats of 64 bits.
- c, complex64: pair number of float of 32 bits.
- z, complex128: pair number of floats of 64 bits.

Returns a c_binary.

2.3 shift

The BLAS library tended to use interleaved matrices and arrays. In order to access them, shifting pointers/c_binaries around is required.

2.3.1 shift/2

```
shift(Shift, C_binary)
    Shift: integer
    C_binary: c_binary
```

Returns a c_binary witch starts with offset Shift (in bytes) relative to input C_binary.

2.4 copy

This functions copies the content of a Binary into a c_binary.

2.4.1 copy/2

```
copy(Binary, C_binary)
    Binary: binary
    C_binary: c_binary
```

Returns ok on success.

2.5 to_bin

This functions converts a c_binary to a binary.

2.5.1 to_bin/1

```
to_bin(C_binary)
    C_binary: c_binary
```

Returns and Erlang binary copy of the c_binary content.

2.5.2 to_bin/2

```
to_bin(Size, C_binary)
    Size: integer
    C_binary: c_binary
```

Returns the first Size bytes of C_binary copied in a binary.

2.6 to_list

This function converts a Binary to a list with given encoding.

2.6.1 to_list/2

```
to_list(Encoding, Binary)
     Encoding: int32 | int64 | s | float32 | d |float64
                 | c | complex64 | z | complex128
    Binary: binary
```

Encoding indicates how the numbers should be encoded:

- int32: integers of 32 bits.
- int64: integers of 32 bits.
- s, float32: floats of 32 bits.
- d, float64: floats of 64 bits.
- c, complex64: pair number of float of 32 bits.
- z, complex128: pair number of floats of 64 bits.

Returns a list of numbers contained by the Binary.

3 Representing BLAS-LAPACKE types in Erlang

This projects provide a complete interface to all BLAS-LAPACKE variables-types.

3.1 Numbers - Arrays - Characters

```
char:         atom
const int:    int,
const float:  double,
const double: double,
const int*:   binary, c_binary,
const float*: binary, c_binary,
const double*:binary, c_binary,
const void*:  binary, c_binary,
const int*:   c_binary,
void*:        c_binary,
float*:       c_binary,
double*:      c_binary,
```

3.2 Enumerations

Enumeration values are represented as atoms.

```
CBLAS_ORDER        blasRowMajor, blasColMajor
CBLAS_TRANSPOSE    n, blasNoTrans, t, blasTrans, c, blasConjTrans
CBLAS_UPLO         u, blasUpper, l, blasLower
CBLAS_DIAG         n, blasNonUnit, u, blasUnit
CBLAS_SIDE         l, blasLeft, r, blasRight
```

4 Examples

4.1 dscal

Double type, SCALe a vector.
X: Alpha*X.
cblas signature:

```
void cblas_dscal (const int n, const double a, double *x, const int incx);
```

Erlang code:

```
X  = blas:new(float64, [1,2,1,2,1,2,1,2]),
ok = blas:run({dscal, 8, 2.0, X, 1}),
blas:btl(float64, blas:to_bin(X)).
```

4.2 dgemm

Double type, GEneral matrices, Matrix Matrix product.
C: Alpha * A * B + Beta * C
cblas signature:

```
void cblas_dgemm(CBLAS_LAYOUT layout, CBLAS_TRANSPOSE TransA,
            CBLAS_TRANSPOSE TransB, const CBLAS_INT M, const CBLAS_INT N,
            const CBLAS_INT K, const double alpha, const double *A,
            const CBLAS_INT lda, const double *B, const CBLAS_INT ldb,
            const double beta, double *C, const CBLAS_INT ldc);
)
```

Erlang code:

```
A  = blas:new(float64, [1,2,3, 1,2,3, 1,2,3]),
B  = blas:new(float64, [4,5,6, 4,5,6, 4,5,6]),
C  = blas:new(float64, [0,0,0, 0,0,0, 0,0,0]),
ok = blas:run({dgemm, blasRowMajor, n,n, 3,3,3, 1.0, A,3, B,3, 0.0, C,3}).
```

4.3 stpmv

Single real numbers, Triangular Packed matrix, Matrix*Vector operation.
X: A*X.
cblas signature:

```
void cblas_stpmv(CBLAS_LAYOUT layout, CBLAS_UPLO Uplo,
             CBLAS_TRANSPOSE TransA, CBLAS_DIAG Diag,
             const CBLAS_INT N, const float *Ap, float *X, const CBLAS_INT incX);
```

Erlang code:

```
N  = 3,
A  = blas:new(s, [1.0, 2.0, 3.0, 4.0, 5.0, 6.0]),
X  = blas:new(s, [-0.25,-0.125,0.5]),
ok = blas:run(
    {stpmv, blasRowMajor, blasUpper, blasNoTrans, blasNonUnit, N, A, X, 1},
    clean
),
[1.0,2.0,3.0] = blas:to_list(s, X).
```

5 Supported BLAS functions

The following blas_atom are supported:

saxpy, daxpy, caxpy, zaxpy, scopy, dcopy, ccopy, zcopy, sswap, dswap, cswap, zswap, sscal, dscal, cscal, csscal, zscal, zdscal, sdot, ddot, cdotu, zdotu, cdotc, zdotc, dsdot, sdsdot, snrm2, dnrm2, scnrm2, dznrm2, sasum, dasum, scasum, dzasum, isamax, idamax, icamax, izamax, srot, drot, csrot, zdrot, srotg, drotg, crotg, zrotg, srotmg, drotmg, srotm, drotm, isamin, idamin, icamin, izamin, ismax, idmax, icmax, izmax, ismin, idmin, icmin, izmin, sgemv, dgemv, cgemv, zgemv, sgbmv, dgbmv, cgbmv, zgbmv, ssbmv, dsbmv, sger, dger, strmv, dtrmv, ctrmv, ztrmv, strsv, dtrsv, ctrsv, ztrsv, strsm, dtrsm, ctrsm, ztrsm, cgeru, cgerc, zgeru, zgerc, sgemm, dgemm, cgemm, cgemm3m, zgemm, zgemm3m, stbmv, dtbmv, ctbmv, ztbmv, stbsv, dtbsv, ctbsv, ztbsv, stpmv, dtpmv, ctpmv, ztpmv, stpsv, dtpsv, ctpsv, ztpsv, ssymv, dsymv, chemv, zhemv, sspmv, dspmv, sspr, dspr, chpr, zhpr, sspr2, dspr2, chpr2, zhpr2, chbmv, zhbmv, chpmv, zhpmv, cher, zher, chemm, zhemm, cherk, zherk, cher2k, zher2k, ssymm, dsymm, csymm, zsymm, ssyrk, dsyrk, csyrk, zsyrk, ssyr2k, dsyr2k, csyr2k, zsyr2k, ssum, dsum, dzsum, scsum, cher2, zher2, strmm, dtrmm, ctrmm, ztrmm, ssyr, dsyr, ssyr2, dsyr2, sbdsdc , dbdsdc , sbdsqr , dbdsqr , cbdsqr , zbdsqr , sdisna , ddisna , sgbbrd , dgbbrd , cgbbrd , zgbbrd , sgbcon , dgbcon , cgbcon , zgbcon , sgbequ , dgbequ , cgbequ , zgbequ , sgbequb , dgbequb , cgbequb , zgbequb , sgbrfs , dgbrfs , cgbrfs , zgbrfs , sgbsv , dgbsv , cgbsv , zgbsv , sgbtrf , dgbtrf , cgbtrf , zgbtrf , sgbtrs , dgbtrs , cgbtrs , zgbtrs , sgebak , dgebak , cgebak , zgebak , sgebal , dgebal , cgebal , zgebal , sgebrd , dgebrd , cgebrd , zgebrd , sgecon , dgecon , cgecon , zgecon , sgeequ , dgeequ , cgeequ , zgeequ , sgeequb , dgeequb , cgeequb , zgeequb , sgeev , dgeev , cgeev , zgeev , sgeevx , dgeevx

, cgeevx , zgeevx , sgehrd , dgehrd , cgehrd , zgehrd , sgejsv , dgejsv , sgelqf , dgelqf , cgelqf , zgelqf , sgels , dgels , cgels , zgels , sgelsd , dgelsd , cgelsd , zgelsd , sgelss , dgelss , cgelss , zgelss , sgelsy , dgelsy , cgelsy , zgelsy , sgeqlf , dgeqlf , cgeqlf , zgeqlf , sgeqp3 , dgeqp3 , cgeqp3 , zgeqp3 , sgeqpf , dgeqpf , cgeqpf , zgeqpf , sgeqrf , dgeqrf , cgeqrf , zgeqrf , sgeqrfp , dgeqrfp , cgeqrfp , zgeqrfp , sgerfs , dgerfs , cgerfs , zgerfs , sgerqf , dgerqf , cgerqf , zgerqf , sgesdd , dgesdd , cgesdd , zgesdd , sgesv , dgesv , cgesv , zgesv , sgesvd , dgesvd , cgesvd , zgesvd , sgesvj , dgesvj , sgetrf , dgetrf , cgetrf , zgetrf , sgetri , dgetri , cgetri , zgetri , sgetrs , dgetrs , cgetrs , zgetrs , sggbak , dggbak , cggbak , zggbak , sggbal , dggbal , cggbal , zggbal , sggev , dggev , cggev , zggev , sggevx , dggevx , cggevx , zggevx , sggglm , dggglm , cggglm , zggglm , sgghrd , dgghrd , cgghrd , zgghrd , sgglse , dgglse , cgglse , zgglse , sggqrf , dggqrf , cggqrf , zggqrf , sggrqf , dggrqf , cggrqf , zggrqf , sggsvd , dggsvd , cggsvd , zggsvd , sggsvp , dggsvp , cggsvp , zggsvp , sgtcon , dgtcon , cgtcon , zgtcon , sgtrfs , dgtrfs , cgtrfs , zgtrfs , sgtsv , dgtsv , cgtsv , zgtsv , sgtsvx , dgtsvx , cgtsvx , zgtsvx , sgttrf , dgttrf , cgttrf , zgttrf , sgttrs , dgttrs , cgttrs , zgttrs , chbev , zhbev , chbevd , zhbevd , chbevx , zhbevx , chbgst , zhbgst , chbgv , zhbgv , chbgvd , zhbgvd , chbgvx , zhbgvx , chbtrd , zhbtrd , checon , zhecon , cheequb , zheequb , cheev , zheev , cheevd , zheevd , cheevr , zheevr , cheevx , zheevx , chegst , zhegst , chegv , zhegv , chegvd , zhegvd , chegvx , zhegvx , cherfs , zherfs , chesv , zhesv , chesvx , zhesvx , chetrd , zhetrd , chetrf , zhetrf , chetri , zhetri , chetrs , zhetrs , chfrk , zhfrk , shgeqz , dhgeqz , chgeqz , zhgeqz , chpcon , zhpcon , chpev , zhpev , chpevd , zhpevd , chpevx , zhpevx , chpgst , zhpgst , chpgv , zhpgv , chpgvd , zhpgvd , chpgvx , zhpgvx , chprfs , zhprfs , chpsv , zhpsv , chpsvx , zhpsvx , chptrd , zhptrd , chptrf , zhptrf , chptri , zhptri , chptrs , zhptrs , shsein , dhsein , chsein , zhsein , shseqr , dhseqr , chseqr , zhseqr , sopgtr , dopgtr , sopmtr , dopmtr , sorgbr , dorgbr , sorghr , dorghr , sorglq , dorglq , sorgql , dorgql , sorgqr , dorgqr , sorgrq , dorgrq , sorgtr , dorgtr , sormbr , dormbr , sormhr , dormhr , sormlq , dormlq , sormql , dormql , sormqr , dormqr , sormrq , dormrq , sormrz , dormrz , sormtr , dormtr , spbcon , dpbcon , cpbcon , zpbcon , spbequ , dpbequ , cpbequ , zpbequ , spbrfs , dpbrfs , cpbrfs , zpbrfs , spbstf , dpbstf , cpbstf , zpbstf , spbsv , dpbsv , cpbsv , zpbsv , spbtrf , dpbtrf , cpbtrf , zpbtrf , spbtrs , dpbtrs , cpbtrs , zpbtrs , spftrf , dpftrf , cpftrf , zpftrf , spftri , dpftri , cpftri , zpftri , spftrs , dpftrs , cpftrs , zpftrs , spocon , dpocon , cpocon , zpocon , spoequ , dpoequ , cpoequ , zpoequ , spoequb , dpoequb , cpoequb , zpoequb , sporfs , dporfs , cporfs , zporfs , sposv , dposv , cposv , zposv , spotrf , dpotrf , cpotrf , zpotrf , spotri , dpotri , cpotri , zpotri , spotrs , dpotrs , cpotrs , zpotrs , sppcon , dppcon , cppcon , zppcon , sppequ , dppequ , cppequ , zppequ , spprfs , dpprfs , cpprfs , zpprfs , sppsv , dppsv , cppsv , zppsv , spptrf , dpptrf , cpptrf , zpptrf , spptri , dpptri , cpptri , zpptri , spptrs , dpptrs , cpptrs , zpptrs , spstrf , dpstrf , cpstrf , zpstrf , sptcon , dptcon , cptcon , zptcon , spteqr , dpteqr , cpteqr , zpteqr , sptrfs , dptrfs , cptrfs , zptrfs , sptsv , dptsv , cptsv , zptsv , sptsvx , dptsvx , cptsvx , zptsvx , spttrf , dpttrf , cpttrf , zpttrf , spttrs , dpttrs , cpttrs , zpttrs , ssbev , dsbev , ssbevd , dsbevd , ssbevx , dsbevx , ssbgst , dsbgst , ssbgv , dsbgv , ssbgvd , dsbgvd , ssbgvx , dsbgvx , ssbtrd , dsbtrd , ssfrk , dsfrk

, sspcon , dspcon , cspcon , zspcon , sspev , dspev , sspevd , dspevd , sspevx , dspevx , sspgst , dspgst , sspgv , dspgv , sspgvd , dspgvd , sspgvx , dspgvx , ssprfs , dsprfs , csprfs , zsprfs , sspsv , dspsv , cspsv , zspsv , sspsvx , dspsvx , cspsvx , zspsvx , ssptrd , dsptrd , ssptrf , dsptrf , csptrf , zsptrf , ssptri , dsptri , csptri , zsptri , ssptrs , dsptrs , csptrs , zsptrs , sstebz , dstebz , sstedc , dstedc , cstedc , zstedc , sstegr , dstegr , cstegr , zstegr , sstein , dstein , cstein , zstein , sstemr , dstemr , cstemr , zstemr , ssteqr , dsteqr , csteqr , zsteqr , ssterf , dsterf , sstev , dstev , sstevd , dstevd , sstevr , dstevr , sstevx , dstevx , ssycon , dsycon , csycon , zsycon , ssyequb , dsyequb , csyequb , zsyequb , ssyev , dsyev , ssyevd , dsyevd , ssyevr , dsyevr , ssyevx , dsyevx , ssygst , dsygst , ssygv , dsygv , ssygvd , dsygvd , ssygvx , dsygvx , ssyrfs , dsyrfs , csyrfs , zsyrfs , ssysv , dsysv , csysv , zsysv , ssysvx , dsysvx , csysvx , zsysvx , ssytrd , dsytrd , ssytrf , dsytrf , csytrf , zsytrf , ssytri , dsytri , csytri , zsytri , ssytrs , dsytrs , csytrs , zsytrs , stbcon , dtbcon , ctbcon , ztbcon , stbrfs , dtbrfs , ctbrfs , ztbrfs , stbtrs , dtbtrs , ctbtrs , ztbtrs , stfsm , dtfsm , ctfsm , ztfsm , stftri , dtftri , ctftri , ztftri , stfttp , dtfttp , ctfttp , ztfttp , stfttr , dtfttr , ctfttr , ztfttr , stgevc , dtgevc , ctgevc , ztgevc , stgexc , dtgexc , ctgexc , ztgexc , stgsen , dtgsen , ctgsen , ztgsen , stgsja , dtgsja , ctgsja , ztgsja , stgsna , dtgsna , ctgsna , ztgsna , stgsyl , dtgsyl , ctgsyl , ztgsyl , stpcon , dtpcon , ctpcon , ztpcon , stprfs , dtprfs , ctprfs , ztprfs , stptri , dtptri , ctptri , ztptri , stptrs , dtptrs , ctptrs , ztptrs , stpttf , dtpttf , ctpttf , ztpttf , stpttr , dtpttr , ctpttr , ztpttr , strcon , dtrcon , ctrcon , ztrcon , strevc , dtrevc , ctrevc , ztrevc , strexc , dtrexc , ctrexc , ztrexc , strrfs , dtrrfs , ctrrfs , ztrrfs , strsen , dtrsen , ctrsen , ztrsen , strsna , dtrsna , ctrsna , ztrsna , strsyl , dtrsyl , ctrsyl , ztrsyl , strtri , dtrtri , ctrtri , ztrtri , strtrs , dtrtrs , ctrtrs , ztrtrs , strttf , dtrttf , ctrttf , ztrttf , strttp , dtrttp , ctrttp , ztrttp , stzrzf , dtzrzf , ctzrzf , ztzrzf , cungbr , zungbr , cunghr , zunghr , cunglq , zunglq , cungql , zungql , cungqr , zungqr , cungrq , zungrq , cungtr , zungtr , cunmbr , zunmbr , cunmhr , zunmhr , cunmlq , zunmlq , cunmql , zunmql , cunmqr , zunmqr , cunmrq , zunmrq , cunmrz , zunmrz , cunmtr , zunmtr , cupgtr , zupgtr , cupmtr , zupmtr

5.1 GRISP

Thought this project is supported by the GRISP platform, the following functions are currently disabled on it: isamin, idamin, icamin, izamin, ismax, idmax, icmax, izmax, ismin, idmin, icmin, izmin, dsum, ssum, scsum, dzsum, cdotu, zdotu, cdotc, zdotc,