Notes on the `bigalgebra` Package

Emmerson, Kane, Lewis

November 29, 2010

1 Introduction

The `bigalgebra` package provides linear algebra operations for "Big Matrices" defined in the `bigmemory` package. The `bigalgebra` package presently only supports double-precision big matrices. The package includes matrix and vector arithmetic operations and several standard matrix factorizations. Infix arithmetic operators are overloaded to use `bigalgebra` functions, keeping the syntax familiar and simple.

The `bigalgebra` package may be optionally compiled with high-performance, multithreaded numeric libraries, which also must provide large-index support. Large indices support linear algebra computation on matrices and vectors with up to 2^{51} total entries (the usual R limit constrains objects to less than 2^{31} total entries). The package also supports computations on arrays that are larger than available RAM using `bigmemory`'s file-backed big matrix objects.

2 Installation

The default installation uses a subset of the refererence BLAS and LAPACK libraries available from Netlib[1] with support for large indices. The `bigalgebra` package may be optionally compiled with alternate libraries that offer performance and threading optimizations.

Installation assumes the standard R toolchain and GNU compiler suite, and requires setting one or more configuration options:

`--with-incDir` Extra required include directories and compiler options.

`--with-blasHeader` Alternate BLAS header file.

`--with-blas` The BLAS library linker instructions.

[1] http://netlib.org

`--with-lapcakHeader` Alternate LAPACK header file.

`--with-lapack` The LAPACK library linker instructions.

`--with-int64` The integer index type (usually `long` or `long long`).

2.1 Installation with Reference BLAS (with large index support)

The `bigalgebra` package includes reference BLAS and LAPACK libraries from Netlib with large index support. The libraries are not tuned for performance–use high-performance libraries instead when possible.

The following example illustrates installation with the reference large-index BLAS and LAPACK libraries:

```
R CMD INSTALL bigalgebra
```

2.2 Installation with Intel MKL

The Intel® Math Kernel Library (MKL), available from http://software.intel.com/en-us/intel-mkl/, is a set of highly optimized and extensively threaded numeric routines for x86, x86-64, and Intel architecture (Itanium) processors. The library is available in standard and large-index versions.

We assume below that the MKL is installed in the `/opt/intel/composerxe-2011.0.084/mkl/` directory–substitute your installation path as required. We illustrate the command-line installation procedure below. Installation from the R console with the `install.packages` command follows similarly. We outline dynamic linkage against MKL which requires that the MKL libraries lie in the system library path. That condition may be temporarily satisfied for the following installation example with:

```
export LD_LIBRARY_PATH=/opt/intel/composerxe-2011.0.084/mkl/lib/intel64/
```

2.2.1 Example: MKL Large-index support on x86-64 systems using GNU Open MP

```
R CMD INSTALL --configure-args='
              --with-incDir="-DMKL_ILP64 -I/opt/intel/composerxe/mkl/include"
              --with-blasHeader="mkl.h"
              --with-blas="-L/opt/intel/composerxe/mkl/lib/intel64/
                           -lmkl_gf_ilp64 -lmkl_core -lmkl_def
                           -lmkl_gnu_thread -lgomp"
              --with-lapack="-L/opt/intel/composerxe/mkl/lib/intel64/
                            -lmkl_gf_ilp64 -lmkl_core -lmkl_def
```

```
                    -lmkl_gnu_thread -lgomp"
        --with-lapackHeader="mkl.h"
        --with-int64="long long"'
        bigalgebra
```

Additional MKL compilation examples can be found in the Intel MKL documentation.

2.3 Installation with AMD ACML

The AMD® Core Math Library (ACML) is available from http://developer.amd.com/cpu/libraries/acml/pages/default.aspx. The ACML provides a highly-threaded set of numeric libraries optimized for performance on x86 and x86-64 processor architectures. The library is available in single and multi-threaded versions.

We assume that the ACML is installed in the `/opt/acml4.4.0/` directory. As in the previous Intel MKL example, we assume that the AMD ACML libraries are available in the library path, for example with:

```
export LD_LIBRARY_PATH=/opt/acml4.4.0/gfortran_mp_int64/lib
```

2.3.1 Example: ACML Large-index support on x86-64 systems

```
R CMD INSTALL --configure-args='
              --with-incDir="-I/opt/acml4.4.0/gfortran64_mp_int64/include/"
              --with-blasHeader="acml.h"
              --with-blas="-L/opt/acml4.4.0/gfortran64_mp_int64/lib/
                          -lacml_mp -lacml_mv"
              --with-lapack="-L/opt/acml4.4.0/gfortran64_mp_int64/lib/
                            -lacml_mp -lacml_mv"
              --with-lapackHeader="acml.h"
              --with-int64="long"'
              bigalgebra
```

3 Examples

The following trivial example sums two vectors with 2^{31} entries, larger than can be indexed by standard R operations. Note that the output vector y will consume approximately 16 GB of disk space.

```
> library('bigalgebra')
> x <- big.matrix(nrow=2^31, ncol=1, type='double', backingfile='x')
> y <- big.matrix(nrow=2^31, ncol=1, type='double', backingfile='y')
> x[5,1] <- 5
```

```
> y[5,1] <- 5
> daxpy(Y=y, X=x)
> print(y[1:10,])
 [1]  0  0  0  0 10  0  0  0  0  0
```