Some notes on efficient computing and high performance computing environments

Andrew Finley[1] & Jeffrey Doser[2]

May 15, 2023

[1] Department of Forestry, Michigan State University.

[2] Department of Integrative Biology, Michigan State University.

Code implementation

Very useful libraries for efficient matrix computation:

1. Fortran BLAS (Basic Linear Algebra Subprograms, see Blackford et al. 2001). Started in late 70s at NASA JPL by Charles L. Lawson. See `http://www.netlib.org/blas`.

2. Fortran LAPACK (Linear Algebra Package, see Anderson et al. 1999). Started in mid 80s at Argonne and OakRidge National Laboratories. See `http://www.netlib.org/lapack`.

Modern math software has a heavy reliance on these libraries, e.g., Matlab and *R*. Routines are also accessible via C, C++, Python, etc.

Many improvements on the standard BLAS and LAPACK functions, see, e.g.,

- Intel Math Kernel Library (MKL)

- AMD Core Math Library (ACML)

- Automatically Tuned Linear Algebra Software (ATLAS)

- Matrix Algebra on GPU and Multicore Architecture (MAGMA)

- OpenBLAS `http://www.openblas.net`

- vecLib (for Mac users only)

Key BLAS and LAPACK functions used in our setting.

Function	Description
dpotrf	LAPACK routine to compute the Cholesky factorization of a real symmetric positive definite matrix.
dtrsv	Level 2 BLAS routine to solve the systems of equations $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{x}$ and $\mathbf{b}$ are vectors and $\mathbf{A}$ is a triangular matrix.
dtrsm	Level 3 BLAS routine to solve the matrix equations $\mathbf{AX} = \mathbf{B}$, where $\mathbf{X}$ and $\mathbf{B}$ are matrices and $\mathbf{A}$ is a triangular matrix.
dgemv	Level 2 BLAS matrix-vector multiplication.
dgemm	Level 3 BLAS matrix-matrix multiplication.

Computing environments

Consider different environments:

1. A distributed system consists of multiple autonomous computers (nodes) that communicate through a network. A computer program that runs in a distributed system is called a distributed program. Message Passing Interface (MPI) is a specification for an Application Programming Interface (API) that allows many computers to communicate.

Consider different environments:

1. A distributed system consists of multiple autonomous computers (nodes) that communicate through a network. A computer program that runs in a distributed system is called a distributed program. Message Passing Interface (MPI) is a specification for an Application Programming Interface (API) that allows many computers to communicate.

2. A shared memory multiprocessing system consists of a single computer with memory that may be simultaneously accessed by one or more programs running on multiple Central Processing Units (CPUs). OpenMP (Open Multi-Processing) is an API that supports shared memory multiprocessing programming.

3. A heterogeneous system uses more than one kind of processor, e.g., CPU & (Graphics Processing Unit) GPU or CPU & Intel's Xeon Phi Many Integrated Core (MIC).

Which environments are right for large *n* settings?

- MCMC necessitates iterative evaluation of the likelihood which requires operations on large matrices.

- A specific hurdle is factorization to computing determinant and inverse of large dense covariance matrices.

- We try to model our way out and use computing tools to overcome the complexity (e.g., covariance tapering, Kaufman et al. 2008; low-rank methods, Cressie and Johannesson 2008; Banerjee et al. 2008, etc.).

- Due to slow network communication and transport of submatrices among nodes distributed systems are not ideal for these types of iterative large matrix operations.

- My lab currently favors shared memory multiprocessing and heterogeneous systems.

- Newest unit is a Dell Poweredge with 384 GB of RAM, 2 threaded 10-core Xeon CPUs, and 2 Intel Xeon Phi Coprocessor with 61-cores (244 threads) running a Linux operating systems.

- Software includes OpenMP coupled with Intel MKL. MKL is a library of highly optimized, extensively threaded math routines designed for Xeon CPUs and Phi coprocessors (e.g., BLAS, LAPACK, ScaLAPACK, Sparse Solvers, Fast Fourier Transforms, and vector RNGs).

So what kind of speed up to expect from threaded BLAS and
LAPACK libraries.
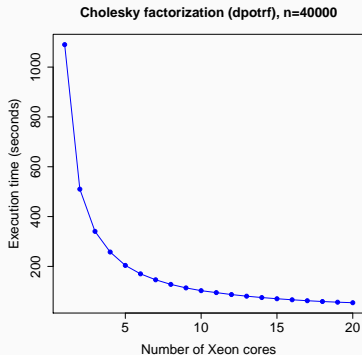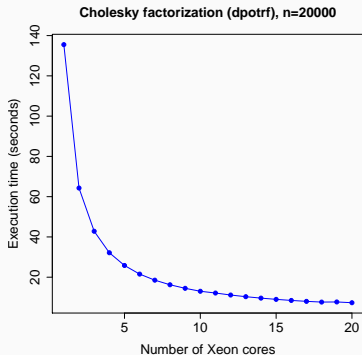
R and threaded BLAS and LAPACK

The BLAS and LAPACK that "ships" with R is single-threaded,
but these can be replaced with multi-threaded libraries.

Windows
- Microsoft R Open: The Enhanced R Distribution
  https://mran.microsoft.com/open comes with MLK
  https://software.intel.com/en-us/mkl.
- Replace existing R's libRblas.so with OpenBLAS library
  libopenblas.so. OpenBLAS is available here
  http://www.openblas.net.

Max OS X
- Mac vecLib obtained via XCode. Use install notes (▸ here).

Linux/Unix
- MKL, OpenBLAS, ACML (compile R against MLK or post
  compile symbolic link of libRblas.so to libopenblas.so).
- Some additional gains using Intel icc and ifort compilers.