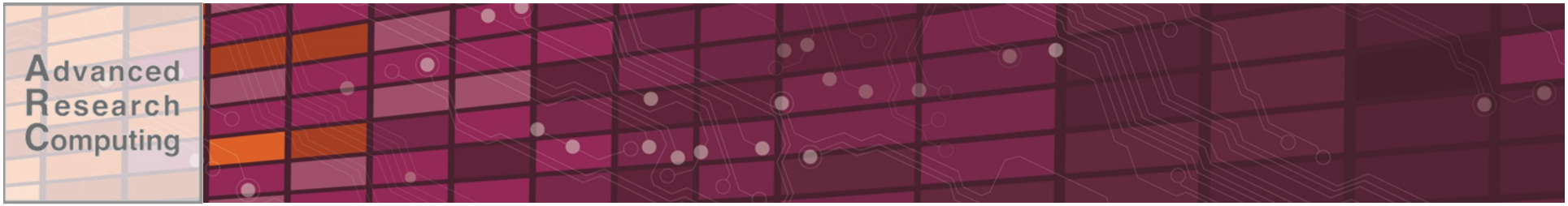


Faster Code for Free: Linear Algebra Libraries

Advanced Research Computing

26 Feb 2016

Outline

- Introduction
- Implementations
- Using them
  - Use on ARC systems
- Hands on session
- Conclusions

Introduction

BLAS

Level	Operations	Complexity	Examples
1	Vector-vector	O(N)	Vector addition, scaling, norm
2	Matrix-vector	$O(N^2)$	Matrix-vector multiply
3	Matrix-matrix	$O(N^3)$	Matrix-matrix multiplication

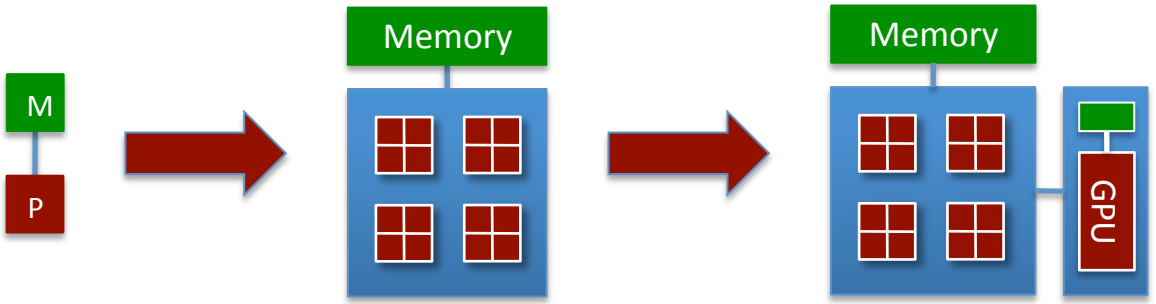
LAPACK

- Linear Algebra Package for solving:
  - Simultaneous linear equations and least-squares solutions of linear systems of equations
  - Eigenvalue and singular value problems
  - Matrix factorizations: LU, Cholesky, QR, SVD, Schur
- Designed to use as many Level 3 BLAS as possible

Related Packages

- BLACS (Basic Linear Algebra Communication Subprograms): Linear algebra message passing
- SCALAPACK: LAPACK for distributed memory machines (uses BLACS)
- MAGMA: LAPACK but for heterogeneous/ hybrid architectures (e.g. CPU/GPU)

HPC Trends



Architecture	Code
Single core	Serial
Multicore	BLAS, LAPACK
GPU	CUBLAS, MAGMA
Cluster	MPI

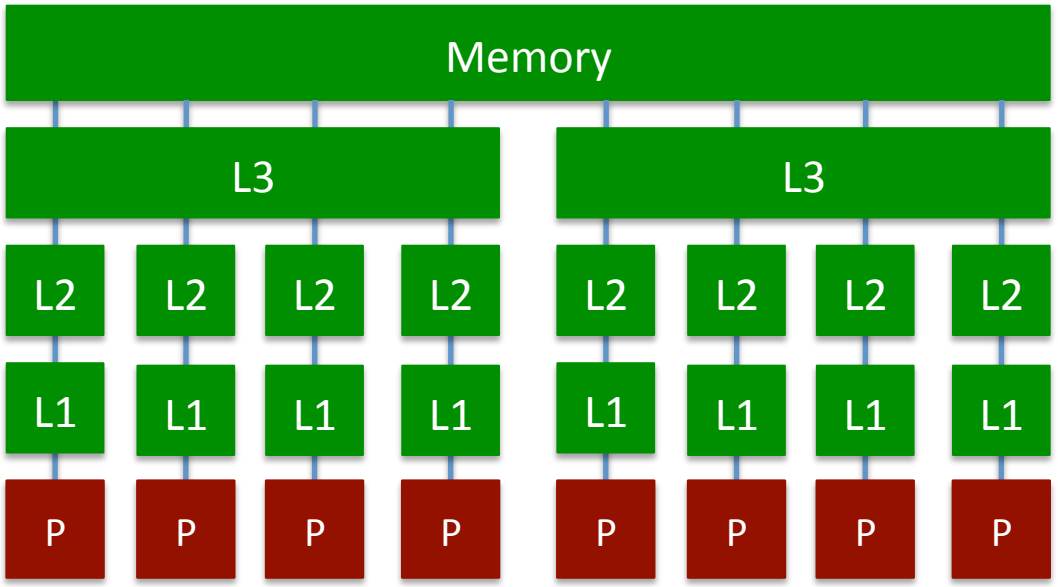
Implementations

Implementations

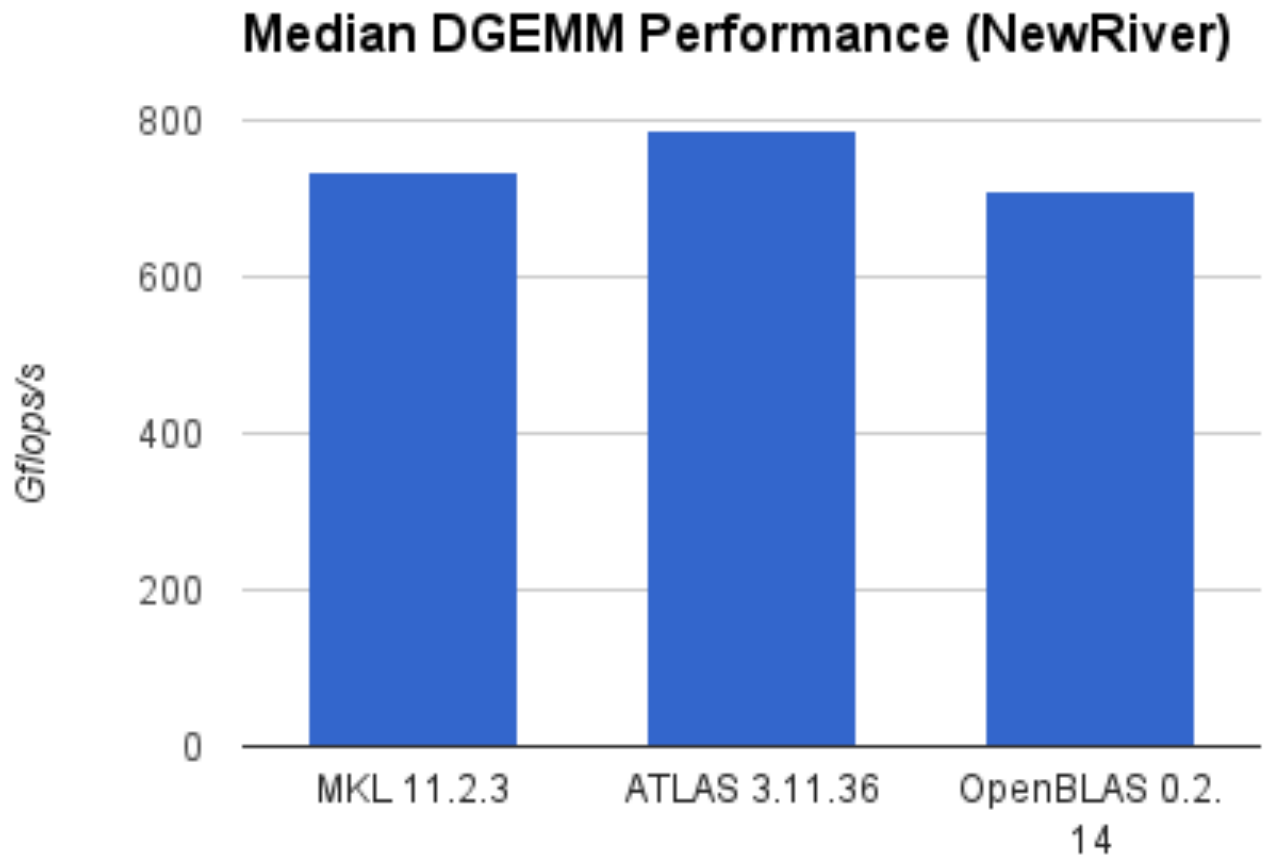
- MKL: Intel, Not free but fast
  - Broader scientific library, including e.g. FFTW
  - Some built-in capability for offload to Xeon Phi
- OpenBLAS: Free
  - Fork of famous GotoBLAS2 project
- ATLAS: Free, automatically tunes to hardware
  - Tough to build and link against
  - Faster in recent versions
  - No thread control at runtime
- There are others

Memory Hierarchy

- Tuning to size of caches has a big impact on performance



Performance Comparison



ATLAS Versions

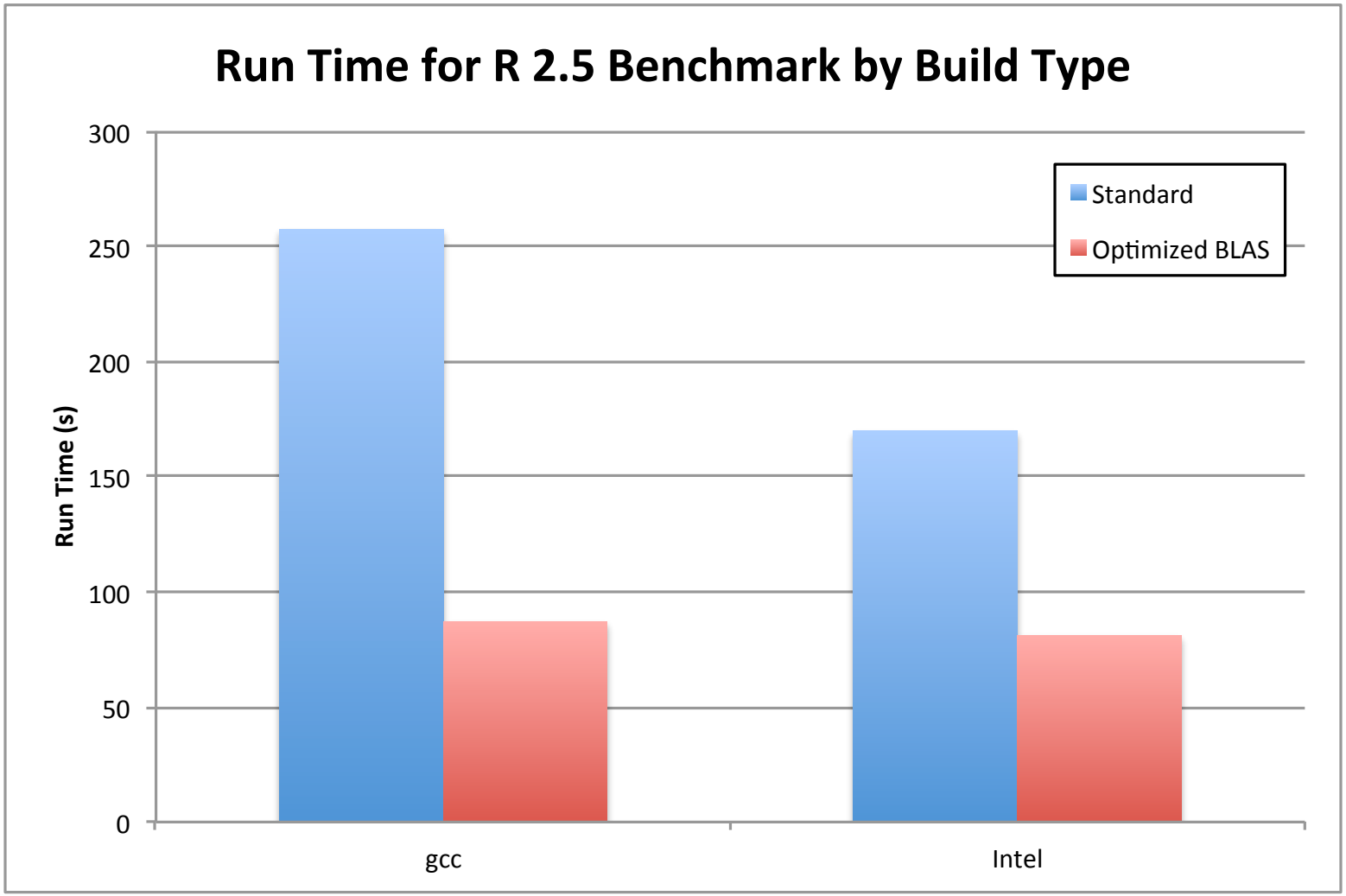
- ARC testing has shown substantial performance improvements from the last stable version of ATLAS (3.10.2) to more recent "unstable" versions (e.g. 3.11.34+)
- Ex: DGEMM of 4224 x 4224 matrices on BlueRidge:
  - 3.10.2: ~210 Gflops/s
  - 3.11.34: ~275 Gflops/s

Third-party Packages

ARC-installed packages that use MKL include:

- Aspect: Mantle convection
- Deal II: Finite elements
- Espresso: Electronic-structure calculations
- Gromacs: Molecular dynamics
- Julia: Numerical computing
- Lammps: Molecular dynamics
- NWChem: Electronic-structure calculations
- PETSc: PDE solver
- Python (numpy): Numerical computing
- R: Numerical computing
- Trilinos: Scalable algorithms
- Vasp: Electronic-structure calculations

Example: R and BLAS



Using Linear Algebra Libraries

Function Naming Scheme

- Six letters: XYYZZZ
- First letter indicates the data type:
  - –S Real
  - –D Double
  - –C Complex
  - –Z Double complex
- Next two letters are matrix type, e.g.:
  - –GE General
  - –SY Symmetric
  - –etc

Naming Convention (continued)

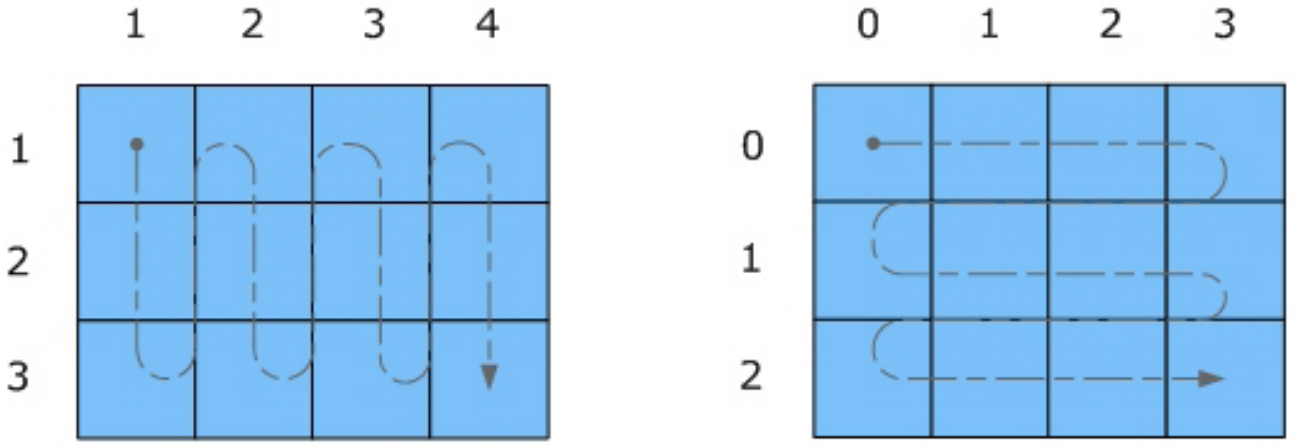
- Six letters: XYYZZZ
- Last 2-3 letters are the computation, e.g.:
  - –MM Matrix multiply
  - –EV Eigenvalue
  - –TRF Factorize
  - –TRS Solve using factorization
  - –etc
- Examples:
  - –dgemm() is double-precision general matrix multiply
  - –ssyev() is single-precision symmetric eigenvalue

Function Parameters

```
dgemm  (              //calculates alpha*op( A )*op( B ) + beta*C
      character                          TRANSA,          //transpose A? 'n' or 't'
      character                          TRANSB,          //transpose B? 'n' or 't'
      integer                            M,               //rows of C
      integer                            N,               //columns of C
      integer                            K,               //columns of A
      double precision                   ALPHA,           //scalar
      double precision, dimension(lda,*) A,               //first matrix
      integer                            LDA,             //rows (C) or cols (Fort) of A
      double precision, dimension(ldb,*) B,               //second matrix
      integer                            LDB,             //rows (C) or cols (Fort) of B
      double precision                   BETA,            //scalar
      double precision, dimension(ldc,*) C,               //C, overwritten by answer
      integer                            LDC              //rows (C) or cols (Fort) of C
)
```

A Caution for C Programmers

- LAPACK written in Fortran (column-major)
  - C/C++ stores matrices in row-major format.
- "Built in transpose" if you call them from C



A: Column-major order (Fortran-style)

B: Row-major order (C-style)

A Caution for C Programmers

- Result: Matrices are interpreted as their transpose
- For matrix-multiply, this means that we call the arguments in reverse:

  `dgemm(…,B,…,A,…)` = $B^T * A^T = (AB)^T = C^T$

- And with the dimensions of B^T and A^T:

```
dgemm('n','n',n,m,k,…)
```

Linking

- MKL:
  - Lots of options, can get complicated, e.g.

```
-L${MKLROOT}/lib/intel64 -lmkl_intel_lp64 -
lmkl_core -lmkl_intel_thread -lpthread -lm
```

  - MKL Link Line Advisor can help with this
- OpenBLAS:

```
-L$OPENBLAS_LIB -lopenblas
```

- ATLAS:
  - Threaded:

```
-L$ATLAS_LIB -llapack -lptf77blas -ltatlas
```

  - Replace `-ltatlas` with `-latlas` for sequential

Thread Control

- MKL:
  - Link to sequential or threaded libraries
  - MKL_NUM_THREADS environment variable
- OpenBLAS:
  - OPENBLAS_NUM_THREADS environment variable
- ATLAS:
  - Link to sequential or threaded (all cores) libraries
  - No thread control at runtime

Linear Algebra Libraries on ARC Systems

Module System

```
[arcadm@nr123 gemm]$ module purge; module load gcc openblas
[arcadm@nr123 gemm]$ module list

Currently Loaded Modules:
  1) gcc/5.2.0   2) openblas/0.2.14

[arcadm@nr123 gemm]$ module help openblas
-----------------Module Specific Help for "openblas/0.2.14" -----------------
OpenBLAS is an open source BLAS library forked from the GotoBLAS2-1.13 BSD
version.

Define Environment Variables:

            $OPENBLAS_DIR - root
            $OPENBLAS_LIB - libraries

Prepend Environment Variables:

    LD_LIBRARY_PATH += $OPENBLAS_LIB
```

Getting Started on ARC's Systems

- Request an account (anyone with a VT PID):

  http://www.arc.vt.edu/account

  – Can also request for external collaborators

- Request a system unit allocation:

  http://www.arc.vt.edu/allocations

  – MIC nodes are "charged" the same as normal nodes

Conclusions

Conclusions

- Linear algebra libraries make your code:
  - Simpler
  - Faster
  - More standardized

References

- LAPACK User Guide: http://www.netlib.org/lapack/lug/
- ATLAS Project: http://math-atlas.sourceforge.net/
- OpenBLAS Project: http://www.openblas.net/
- MKL Link Line Advisor: https://software.intel.com/en-us/articles/intel-mkl-link-line-advisor

Advanced Research Computing

Questions?

This is a new class, so we would appreciate your feedback: https://virginiatech.qualtrics.com/jfe/form/SV_5cpaTH2OhXT8yjz