HUMAN CAPITAL
NATIONAL COHESION STRATEGY

EUROPEAN UNION
EUROPEAN
SOCIAL FUND

High Performance Computing

Lecture 5 - Standard Libraries

Felicja Okulicka-Dłużewska

F.Okulicka@mini.pw.edu.pl

Warsaw University of Technology

Faculty of Mathematics and Information Science

WARSAW UNIVERSITY OF TECHNOLOGY
DEVELOPMENT PROGRAMME

Go to full-screen mode now by hitting CTRL-L

Overview

- **BLAS**
- ATLAS
- LAPACK
- ScaLAPACK
- BLACS
- PBLAS

BLAS

The BLAS (**B**asic **L**inear **A**lgebra **S**ubprograms) include subroutines for common linear algebra computations such as dot-products, matrix-vector multiplication, and matrix-matrix multiplication.

Using matrix-matrix operations (in particular, matrix multiplication) tuned for a particular architecture can mask the effects of the memory hierarchy (cache misses, TLB misses, etc.) and permit floating-point operations to be performed near peak speed of the machine.

An important aim of the BLAS is to provide a portability layer for computation.

BLAS

Basic Linear Algebra Subprograms

The BLAS are relatively low-level linear algebra operations:

- L-norm of a vector
- Inner (dot) product of two vectors
- Position of the maximum absolute element of a vector
- Matrix-vector product
- Matrix- matrix product

BLAS1

BLAS1 = Level 1 BLAS Vector - vector operations :

O(n) operations on O(n) data (each datum is used once only).

It was formulated in the 1970s for traditional serial machines

BLAS1

Name	Operation	Prefixes
_ROTG	Generate plane rotation	S, D
_ROTMG	Generate modified plane rotation	S, D
_ROT	Apply lane rotation	S, D
_ROTM	Apply modified plane rotation	S, D
_SWAP	$x \leftrightarrow y$	S, D, C, Z
_SCAL	$x \leftarrow \alpha x$	S, D, C, Z, CS, ZD
_COPY	$y \leftarrow x$	S, D, C, Z
_AXPY	$y \leftarrow \alpha x + y$	S, D, C, Z

BLAS1

Name	Operation	Prefixes
_DOT	$dot \leftarrow x^T y$	S, D, DS
_DOTU	$dot \leftarrow x^T y$	C, Z
_DOTC	$dot \leftarrow x^H y$	C, Z
_DOT	$dot \leftarrow \alpha + x^T y$	SDS
_NRM2	$nrm2 \leftarrow \|\|x\|\|_2$	S, D, SC, DZ
_ASUM	$asum \leftarrow \|\|re(x)\|\|_1 + \|\|im(x)\|\|_1$	S, D, SC, DZ
I_AMAX	$amax \leftarrow 1^{st} k \ni \|re(x) + \|im(x)\|$	S, D, C, Z

BLAS1

Meaning of prefixes:

S - REAL

C - COMPLEX

D - DOUBLE PRECISION

Z - COMPLEX*16

D,Z - may not be supported by all machines

BLAS2

BLAS2 - vector-matrix operations

Name	Operation	Prefixes
_GEMV	$y \leftarrow \alpha A x + \beta y$	S, D, C, Z
	$y \leftarrow \alpha A^T x + \beta y$	S, D, C, Z
	$y \leftarrow \alpha A^H x + \beta y$	S, D, C, Z
	$A - m \times n$	
_GBMV	$y \leftarrow \alpha A x + \beta y,$	S, D, C, Z
	$y \leftarrow \alpha A^T x + \beta y$	S, D, C, Z
	$y \leftarrow \alpha A^H x + \beta y$	S, D, C, Z
	$A - m \times n$	

BLAS2

Name	Operation	Prefixes
_HEMV	$y \leftarrow \alpha Ax + \beta y$	C, Z
_HBMV	$y \leftarrow \alpha Ax + \beta y$	C, Z
_HPMV	$y \leftarrow \alpha Ax + \beta y$	C, Z
_SYMV	$y \leftarrow \alpha Ax + \beta y$	S, D
_SBMV	$y \leftarrow \alpha Ax + \beta y$	S, D
_SPMV	$y \leftarrow \alpha Ax + \beta y$	S, D

BLAS2

Name	Operation	Prefixes
_TRMV	$x \leftarrow \alpha A x, x \leftarrow \alpha A^T x, x \leftarrow \alpha A^H x$	S, D, C, Z
_TBMV	$x \leftarrow \alpha A x, x \leftarrow \alpha A^T x, x \leftarrow \alpha A^H x$	S, D, C, Z
_TPMV	$x \leftarrow \alpha A x, x \leftarrow \alpha A^T x, x \leftarrow \alpha A^H x$	S, D, C, Z
_TRSV	$x \leftarrow \alpha A^{-1} x, x \leftarrow \alpha A^{-T} x, x \leftarrow \alpha A^{-H} x$	S, D, C, Z
_TBSV	$x \leftarrow \alpha A^{-1} x, x \leftarrow \alpha A^{-T} x, x \leftarrow \alpha A^{-H} x$	S, D, C, Z
_TPSV	$x \leftarrow \alpha A^{-1} x, x \leftarrow \alpha A^{-T} x, x \leftarrow \alpha A^{-H} x$	S, D, C, Z

BLAS3

Name	Operation	Prefixes
_GER	$A \leftarrow \alpha x y^T + A,\ A - m \times n$	S, D
_GERU	$A \leftarrow \alpha x y^T + A,\ A - m \times n$	C, Z
_GERC	$A \leftarrow \alpha x y^H + A,\ A - m \times n$	C, Z
_HER	$A \leftarrow \alpha x x^H + A$	C, Z
_HPR	$A \leftarrow \alpha x x^H + A$	C, Z
_HER2	$A \leftarrow \alpha x y^H + y(\alpha x)^H + A$	C, Z
_HPR2	$A \leftarrow \alpha x y^H + y(\alpha x)^H + A$	C, Z

BLAS3

Name	Operation	Prefixes
_SYR	$A \leftarrow \alpha x x^T + A$	S, D
_SPR	$A \leftarrow \alpha x x^T + A$	S, D
_SYR2	$A \leftarrow \alpha x y^T + \alpha y x^T + A$	S, D
_SPR2	$A \leftarrow \alpha x y^T + \alpha y x^T + A$	S, D

BLAS3

Matrix types:

GE - GEneral	SY - SYmmetric
GB - General Band	SB - Symmetric Band
	SP - Symmetric Packed
HE - Hermitian	TR - TRiangular
HB - Hermitian Band	TB - Triangular Band
HP - Hermitian Packed	TP - Triangular Packed

Gauss using BLAS

Following BLAS procedures can be used to implement the Gauss elimination:

- Reciprocal Scale x=x/alpha : RSCALE
- Scaled vector accumulation y=alpha*x+beta*y : AXPBY
- Rank one updates A=alpha*x*trans(y)+beta*A : SGER, DGER
- Triangular solver : TRSV, TBSV, TPSV, TRSM

ATLAS

The ATLAS (**A**utomatically **T**uned **L**inear **A**lgebra **S**oftware) project is an ongoing research effort focusing on applying empirical techniques in order to provide portable performance. It provides C and Fortran77 interfaces to a portably efficient BLAS implementation, as well as a few routines from LAPACK. ATLAS is often recommended as a way to automatically generate an optimized BLAS library

BLAS and others

STANDARD LIBRARIES base on the BLAS (Basic Linear Algebra Subprograms) routines are available from www pages on netlib for a variety of architectures. The URL for netlib is

$$http://www.netlib.org/$$

LAPACK

LAPACK (**L**inear **A**lgebra **PACK**age) provides routines for solving systems of

- linear equations
- linear least squares
- eigenvalue problems
- singular value decomposition

LAPACK codes themselves employ standard Fortran 77 only. Parallelism is achieved due to BLAS (Level 3 BLAS mostly). It works on shared memory machines.

ScaLAPACK

The ScaLAPACK (or **Sca**lable **LAPACK**) library includes a subset of LAPACK routines redesigned for distributed memory MIMD parallel computers.

It is currently written in a Single-Program-Multiple-Data style using explicit message passing for interprocessor communication.

It assumes that matrices are laid out in a two-dimensional block cyclic decomposition.

ScaLAPACK is designed for heterogeneous computing and is portable on any computer that supports MPI or PVM.

ScaLAPACK

The ScaLAPACK routines are based on block-partitioned algorithms in order to minimize the frequency of data movement between different levels of the memory hierarchy.

The ScaLAPACK homepage can be accessed on the World Wide Web via the URL address:

$$http://www.netlib.org/scalapack/index.html$$

ScaLAPACK

The fundamental building blocks of the ScaLAPACK library are

- PBLAS - Parallel Basic Linear Algebra Subprograms - distributed memory version of the Level 1,2 and 3 BLAS
- BLACS - Basic Linear Algebra Communication Subprograms - a set of subroutines for communication tasks that arise frequently in parallel linear algebra computations.

ScaLAPACK

In the routines, all interprocessor communication occurs within the PBLAS and the BLACS.

ScaLAPACK can solve:

- systems of linear equations,
- linear least squares problems,
- eigenvalue problems,
- singular value problems.

ScaLAPACK can also handle many associated computations such as matrix factorizations or estimating condition numbers.

BLACS

The BLACS (**B**asic **L**inear **A**lgebra **C**ommunication **S**ubprograms) are a message-passing library designed for linear algebra. The computational model consists of a one- or two-dimensional process grid, where each process stores pieces of the matrices and vectors.

The BLACS include synchronous send/receive routines

- to communicate a matrix or submatrix from one process to another,
- to broadcast submatrices to many processes,
- to compute global reductions (sums, maxima and minima).

BLACS

There are also routines to construct, change, or query the process grid. BLACS permit a process to be a member of several overlapping or disjoint process grids, each one labeled by a context. Some message-passing systems, such as MPI also include this context concept; MPI calls this a communicator.

The BLACS provide facilities for safe inter-operation of system contexts and BLACS contexts. An important aim of the BLACS is to provide a portable, linear algebra specific layer for communication.

BLACS

BLACS (**B**asic **L**inear **A**lgebra **C**ommunication **S**ubprograms) contains:

- broadcast
- point-to-point communication
- combine operations
- globally blocking and locally-blocking operations

PBLAS

To simplify the design of ScaLAPACK, and because the BLAS have proven to be useful tools outside LAPACK, a parallel set of BLAS, called PBLAS were build. PBLAS performs message-passing and has interface similar to the BLAS. This makes the ScaLAPACK code to be quite similar, and sometimes nearly identical, to the analogous LAPACK code.

The PBLAS (Parallel Basic Linear Algebra Subprograms) library:

- allows global view of distributed matrices
- provides load balanced computations
- does both the communication and computation

Gauss with PBLAS

To write the parallel implementation of the Gauss elimination and solver the following PBLAS procedures can be used:

- Reciprocal Scale x=x/alpha : PvSCAL
- Scaled vector accumulation y=alpha*x+beta*y : PvAXPBY
- Rank one updates A=alpha*x*trans(y) +beta*A : PvGER
- Triangular solver : PvTRSV

Data layouts

Parallel matrix computation needs distributed data layouts. Matrices can have 2D block distribution or 2D block-cyclic distribution. Vectors can have block distribution, cyclic distribution or block-cyclic distribution. 2D block-cyclic distribution of matrices is used for proper job scheduling. There is no idle processors with such data layout.

ScaLAPACK

In the Lapack and Scalapack libraries the block LU decomposition is implemented applying the following rules:

- all matrices are partitioned conformally
- several order of block LU decomposition is possible
- the block matrices can be decomposed in the nested way
- the diagonal blocks do not need to be unit matrices

ScaLAPACK

Four basic steps are required to call a ScaLAPACK routine:

- Initialize the process grid
- Distribute the matrix on the process grid
- Call ScaLAPACK routine
- Release the process grid

PETSc

PETSc (**P**ortable **E**xtensible **T**oolkit for **S**cientific **C**omputation) contains:

- LINPACK (matrix factorization and solve)
- PLAPACK - Parallel Linear Algebra Package
- LAPACK
- Matlab
- BLAS

Thank you for your attention!

Any questions are welcome.