LAPACK Working Note 109
BLAS Technical Workshop[1]

Jack Dongarra[2], Sven Hammarling[3] and Susan Ostrouchov[4]

December 12, 1995

[1]This workshop was hosted by the University of Tennessee with partial support from the NSF S&T Center for Research on Parallel Computation.
[2]The University of Tennessee, Knoxville and Oak Ridge National Laboratory
[3]The University of Tennessee, Knoxville and The Numerical Algorithms Group Ltd, Oxford
[4]The University of Tennessee, Knoxville

Abstract

The BLAS Technical Workshop was held on November 13-14, 1995, in Knoxville, Tennessee. Its focus was on developing a set of Parallel BLAS and related interfaces for linear algebra, specifically the Sparse BLAS, Sparse PBLAS, BLACS, and extensions to the BLAS. Fifty-two people were in attendance.

Contents

1 Introduction

This is an informal report of a technical workshop on the BLAS held in Knoxville on November 13-14, 1995. The workshop was organized by Jack Dongarra of the University of Tennessee, Knoxville, and the Oak Ridge National Laboratory, Iain Duff of Rutherford Appleton Laboratory, and Mike Heroux of Cray Research, Inc. It was attended by 52 people. The purpose of the workshop was to study two related topics for software involving linear algebra computation.

The existing BLAS have proven to be very effective in assisting portable, efficient software for sequential and some of the current class of high-performance computers. The purpose of the workshop was to investigate the possibility of extending the currently accepted standards to provide greater coverage of sparse matrices and provide additional facilities for parallel computing. In particular to consider standardizing a set of Parallel BLAS along the lines of the existing BLAS for the dense and sparse cases.

The goal of the workshop was to stimulate thought, discussion, and comment on the future development of a set of standards for basic matrix data structures, both dense and sparse, as well as calling sequences for a set of low-level computational kernels for the parallel and sequential settings. These new "standards" are needed to complement and supplement the existing ones for sparse and parallel computation. One of the major aims of these standards will be to enable linear algebra libraries (both public domain and commercial) to interoperate efficiently and easily.

There were eight sessions of talks with a total of 28 talks. Each talk was limited to 15 minutes, and each session was followed by a 30 minute discussion session. A Birds-of-a-Feather session was held in the evening of November 13, and there were 3 topics for discussion – Sparse BLAS chaired by Iain Duff and Mike Heroux; Matrix Distribution chaired by Jack Dongarra, Bo Kågström, and Robert van de Geijn; and Extensions to the BLAS chaired by Jeremy Du Croz and Linda Kaufman.

The principle observations and conclusions reached in this workshop were as follows:

- It may be too early to establish a standard for the Parallel BLAS, but there is sufficient interest in forming a group to investigate further.
- We need to consider the facilities and functionality offered by the BLAS following the experience with using them in existing software packages.
- The audience for the BLAS has expanded to encompass application scientists in general.
- Similar meeting style as in HPF and MPI forum should be adopted.
- Lower-level routines are needed to insure high-performance on small problems.
- Interoperability between programming languages needs to be stressed.
- Extensions and alternative entry points (to allow different levels of error checking and profiling) to the BLAS seem appropriate.

Additional information, including many of the presentations for the workshop, can be found on the URL:

http://www.netlib.org/utk/papers/sblas-meeting.html

2 BLAS

Sven Hammarling of NAG Ltd. gave a historical view of the BLAS and the Sparse BLAS. He stressed the importance of not ignoring the past when making decisions for the future. He summarized the standardization process as involving background experience, extensive consultation of facilities, preparation of model and associated software, publication, and promotion.

Jeremy Du Croz of NAG Ltd. presented a proposal for the Fortran 90 version of the BLAS. This Fortran 90 implementation would exploit the following features of the language: generic interfaces, assumed shape arrays, optional arguments, and modules.

Craig Douglas of the IBM T.J. Watson Research Center presented a talk on the OBLAS (Objective Basic Linear Algebra Subprograms). He stressed multiple entry points for BLAS routines.

The primary topics of the discussion sessions were as follows:

- Importance of layering of BLAS packages.
  The idea that the BLAS-related activities need to be considered as a whole, especially since they are part of libraries, was repeated by several speakers at the workshop.
- Is a Fortran 90 interface for the BLAS needed?
  With the current state of Fortran 90 compilers, a data copy is required preceding a call to a Fortran 77 routine. If a Fortran 90 interface to the BLAS is derived, the unnecessary performance hit for the data copy could be avoided.
- Is Fortran 90 discounted by the computer science community?
  Fortran 90 does not yet have a defined interface to other programming languages. Users are driven to use other languages to perform operations which are not supported.
- The dense and sparse BLAS are not uniform in functionality.
- How much of a performance hit are we willing to accept to accommodate input error-checking?
- Efficient BLAS operations needed for small matrices.
  It was argued that small matrix computations naturally occur in sparse matrix calculations, so there is a need for kernels that have little overhead.
- Two entry point for each routine allowing for error checking and profiling, if desired.

2.1 Extensions to the existing BLAS

Linda Kaufman of Bell Labs began her talk by saying that the sparse and parallel BLAS should not be specified without taking into consideration the specific attributes of a problem/platform. She argued that there is no need for xSYRK in sparse matrix computations and it is little used in dense computations. It is not useful for quasi-Newton methods, and cannot be used for block and unblocked indefinite linear system codes in LAPACK. She also proposed that several of the LAPACK auxiliary routines become BLAS routines – SLACPY, SLARFG, SLANGE, xSPR2. She proposed that we need parallel BLAS routines to perform simultaneous Givens transformations, parallel application of Givens transformations, routines for simultaneous ISAMAX and SAXPY operations, and routines for diagonal matrices. Other participants in the workshop seconded this need to solve multiple instance problems.

During the Birds-of-a-Feather session, several proposals were made for new BLAS routines.

Jeremy Du Croz of NAG Ltd addressed the issue:

- Should we propose a standard for FFT's?

Ed Anderson of Cray Research Inc. proposed the following additions:

- SAXPBY/CAXPBY: $y \leftarrow \alpha x + \beta y$ (x, y vectors)
- SGESUM/CGESUM: $B \leftarrow \alpha op(A) + \beta B$ (A, B matrices, op(A) may be A or transpose(A))
- SHAD/CHAD: $z \leftarrow \alpha x * y + \beta z$ (Hadamard product)
- Matrix norms, like the LAPACK SLANxx routines. Could we combine SNRM2, SASUM, and XMAX = X(ISAMAX(N, X, INCX)) into a single interface?
- BLASification of the complex symmetric Level 2 BLAS, and CROT, already distributed with LAPACK

It was suggested that perhaps "SGESUM" is not very mnemonic if the user is only interested in copying, scaling, or transposing a matrix. Perhaps separate routines for these functions would be the best alternative. Barry Smith and Craig Douglas also requested the inclusion of SAXPBY and SGESUM. Linda Kaufman also endorsed the inclusion of SHAD and matrix norms.

Craig Douglas proposed the following extensions:

- SGEMMS (Strassen) or SGEMMW (Winograd); says they are available on netlib
- Mixed real/complex data types, particularly for Level 2 BLAS
- A routine to compute $P\,A\,R$, where A is dense, P and R may be sparse, but may also be dense in applications from economics.

Linda Kaufman of Bell Labs proposed the following additions:

- BLAS definitions of commonly used LAPACK auxiliary routines:
  SLARGV, SLARTV
  SLARF, SLARFG, SLARTG, CLARTG
  SLACPY
  SLANxx
  Complaints all around that even the most commonly used auxiliary routines are not documented in the LUG.

- Generally wants multiple instances of Level 1 BLAS (not all were carried over into Level 2 BLAS). SLARGV is one example.
- Update operation for the symmetric pivoting factorization (a.k.a., the "Bunch-Kaufman factorization"): $X \leftarrow X + A\,B\,A^T$, where X is symmetric, A is $n-by-k$, B is $k-by-k$ tridiagonal (really just needs B to be block diagonal with $1-by-1$ and $2-by-2$ diagonal blocks)

Barry Smith of Argonne National Laboratory proposed the following:

- WAXPY: $z \leftarrow \alpha x + [\beta]y$
- Multiple dot products or SAXPY's with one vector the same. This is not the same as SGEMV, because the other vectors may not be in a matrix (no problem in C, but impossible in Fortran).

3 Sparse BLAS

Iain Duff of Rutherford Laboratory and Mike Heroux of Cray Research, Inc. presented talks on the Sparse BLAS. The proposal for the Level 1 Sparse BLAS has been published in ACM Transactions on Mathematical Software [1]. The proposals for the Level 2 and 3 Sparse BLAS are still in review. It was stressed that the proposals for the Sparse BLAS were motivated by their potential use in iterative methods. Iain invited comments from the community on

- mixed language use
- error handling
- name changes
- work arrays
- more scaling possibilities
- Fortran90 interface

Mike Heroux of Cray Research, Inc. presented an overview of the Sparse BLAS Toolkit and its latest features, Sparse BLAS primitives, and the Sparse BLAS layers. The Sparse BLAS primitives are a very small set of macro dependent loop sets that support the Sparse BLAS Toolkit. A macro version of the loops can be optimized for a particular architecture since the definition of a macro does not impact optimization strategies. And finally, the Sparse BLAS primitives minimize the amount of code that hardware vendors must optimize. He ended his talk by reiterating the layered view of BLAS related software.

Yousef Saad of the University of Minnesota presented a talk on P_SPARSLIB, a library of iterative methods for the solution of sparse systems on distributed-memory machines. P_SPARSLIB aims to provide the basic kernels, preprocessing tools and preconditioners, as well as the iterative solvers. Emphasis is placed on the algorithms, data structures and portability.

Roldan Pozo of NIST gave a talk on sparse BLAS interface issues in C and C++. He felt that there is a need for more general matrix structures, and proposed a simplication of the current interface to remove work arrays and character string arguments. Function overloading could be used to provide a single interface call. Templates for block structures and matrices of generic types could be used. The use of optional parameters would simply function calls. Operator overloading could be used to replace function calls. Exception handling could be used for better error control. Validation tests would only be performed at object construction to reduce the need for repeated integrity checking. And finally, he proposed the use of ANSI C prototyping for safety and proper linking.

The primary topics of the discussion sessions were as follows:

- Should character strings be eliminated from the interface?
  It was argued that it should not be kept for historical Fortran-ish reasons, and only creates difficulties in the mixing of programming languages. On the other hand, it does provide greater readability of the code.

- How much we are willing to sacrifice in performance for error-checking?
  Overhead and performance issues related to error-checking were then discussed and it was suggested to provide two interfaces to the library. One interface would provide error-checking for the user and should be used during the debug phase of program development. The second interface would not provide any error-checking and would be used for optimal performance.

- Should we allow multiple interfaces to low-level routines?
  Mike Heroux argued that multiple interfaces to the same routine was too much for vendors to support.

The primary theme of the Sparse BLAS Birds-of-a-Feather session was the concept of layers or levels. We identified four distinct levels of sparse BLAS. Starting from the lowest level we have:

- Sparse BLAS primitives
  These were also referred to as part of a set of RISC BLAS or a Basic Linear Algebra Instruction Set. These are the kernels that capture the essential operations using specific data structures. We would expect hardware vendors to implement these routines very efficiently. It is required to be a very small set of computationally intensive kernels. User-friendliness is not an issue for these kernels, only performance and simplicity.

- Sparse BLAS Toolkit
  A set of higher level kernels built on top of the primitives, these routines are still data structure specific, but have interfaces that are callable from any language (all data structures are simple scalar values or one dimensional arrays). These routines can deal with matrix properties such as symmetry, triangularity, unit diagonals, etc. and handle some of the finer details that might be otherwise tedious for the user to deal with explicitly. However, they are still not “object-oriented” in design.

- User level (Object Oriented) Sparse BLAS
  Built on top of the sparse BLAS Toolkit, primitives, or other appropriate routines, this interface treats sparse matrices as objects and hides data structure complexities from the user interface by using the appropriate facilities of the given language. We saw a proposal for this type of interface in Fortran 77, Fortran 90 and C/C++. It is expected that application developers writing portable code would access the sparse BLAS from this level using the language specific interface that matches the rest of the application code.

- Distributed memory sparse BLAS
  Implicit in the previous three levels is a single processor or SMP programming model. In a distributed memory application, the application developer would use routines from the other three sparse BLAS levels as appropriate to perform the local computations. However, it would also be useful to have a set of routines to deal with boundary data in domain decomposition and domain partitioning schemes. These routines would handle communication between processes using sparse matrix representations.

In addition to the layers theme, two other relevant points came up:

- All kernel routines should have an alternate entry point at exactly the same location as the standard entry point. This "P" (P for profile) entry point could then be used after intercepting the call to a kernel and collecting some data. We then call the "P" entry point to continue the kernel execution. For example, if DGEMM has a second entry point called PDGEMM at the same location, I could provide my own routine called DGEMM which would look like this:

  ```
  SUBROUTINE DGEMM( ... )
  ⋮
  Collect statistics, analyze input data for errors, etc.
  ⋮
  CALL PDGEMM( ... )
  ```

  This structure can yield great insight into the execution of the code and still ensure the proper execution of the original kernel.

- We should develop a sparse matrix data file standard for C/C++ much like that provided by the Harwell Boeing format defined for Fortran 77. Although it is possible to access HB files from C/C++, it is cumbersome and platform dependent.

4 Future Design Issues

Andrew Lumsdaine from the University of Notre Dame gave a talk stressing the shortcomings of current mathematical libraries. One of his main points was that the BLAS related activities need to be considered as a whole, especially since they are part of libraries. Several speakers later in the workshop also reiterated this view. He pointed out that the dense and sparse BLAS are not uniform in functionality, and they should be. The semantics for routines are sometimes incompletely specified. Subtle numerical properties are not expressed in the standard. No provisions are made for promotion, demotion, or conversion. The memory layout is implicitly specified via Fortran77. He wanted to accommodate routines designed for row-major orderings as well. Fred Gustavson of IBM added that a good vendor implementation of the BLAS should circumvent the need for this. Lumsdaine's proposed fix to address the shortcomings of the BLAS was the Basic Linear Algebra Instruction Set (BLAIS).

Bill Gropp from the Argonne National Laboratory presented a talk about the requirements for a Parallel BLAS library. He stressed four key areas: correctness, latency control, nestability, and the need to use subsets of processors. For correctness, he stressed the need and use of MPI communicators to ensure separate spheres of computation (non-interference from other computations), and the need for careful attention to buffering issues. For latency control, the splitting of operations is needed to allow for separate entry points for basic and "combined" operations. For nestability, the routines must not assume a single environment. Libraries at different levels should not interfere with each other.

Barry Smith of Argonne National Laboratory presented a talk on the importance of highly efficient computational kernels for all block sizes. He cited several examples where small block sizes naturally occur in sparse matrix computations. Since the blocks are very small, the computational kernels must be lightweight and have little overhead. He listed several key factors to obtain good performance for small block sizes:

- No run-time function overloading with additional arguments (fat interfaces) should be allowed.
- No error-checking should be performed.
- No character string arguments should be allowed.
- Code should be inlineable.
- Code should allow caching of reusable structure information.
- Separate code for stride 1 operations is needed.
- Many of the "BLAS" operations should be optimized for small block sizes.

Tony Skjellum of Mississippi State University presented a talk on the future directions of scalable parallel libraries. He felt that current parallel libraries tend to ignore software engineering issues, the cost of data remapping, polyalgorithms, and sequential kernel design for concurrency. Library use should be application driven, and the needs of the "customers" will determine what is done in the future.

Steve Huss-Lederman of the SuperComputing Research Center presented a talk on the issues related to standardizing the Parallel BLAS. He spoke about how to create a standard appropriate for a large group of people. Diversity must be accommodated between library writers and tool developers, and application users. Data layout is one of the central issues in parallel library development. HPF was fixed to a two-dimensional block-cyclic layout, so it was natural for ScaLAPACK to choose the same data layout. However, he felt that the evolution of data layouts is still continuing, and that we should support alternative data layouts, and provide as much flexibility as possible.

4.1 Parallel BLAS

Antoine Petitet of the University of Tennessee presented a proposal for the Levels 1, 2, and 3 PBLAS, as used in ScaLAPACK. The PBLAS assume a SPMD programming model and a two-dimensional block-cyclic data decomposition. They provide similar functionality to the BLAS: distributed vector-vector, matrix-vector, and matrix-matrix operations. They allow software re-use of dense linear algebra codes, especially when such software is implemented on top of the BLAS. They also provide clarity, modularity, and program portability. Since the PBLAS rely on the BLACS for communication, there is a safe co-existence with other parallel libraries. The PBLAS do not perform "inter-context" operations. The PBLAS specification does not preclude the use of other data distributions, and can be trivially extended to allow different data layouts. Depending upon the shape and locality of their operands, the PBLAS routines employ the use of polyalgorithms in their design to select the most appropriate and efficient algorithm available to them. All software described is available on netlib.

4.2 Matrix Distribution

Robert van de Geijn of the University of Texas, Austin, presented a different perspective on matrix distributions in which vectors, not matrices, are the key component, the rationale being that vectors are the natural quantity of concern to applications. Matrix distributions, which he calls "physically based," are induced from the vector distributions, resulting in blocking that is natural to the application problem and not imposed by library software. The resulting matrix distributions are Cartesian but allow variable block sizes, and thus generalize two-dimensional block cyclic mappings. He argued that this paradigm unifies dense and sparse linear algebra and results in better load balancing for certain problems. He also outlined how these ideas could be combined with previously defined collective communication routines to implement a modular and compact linear algebra library.

Carter Edwards of the University of Texas, Austin, presented the theory behind the physically based matrix distribution scheme. He defined semantics for matrix distribution – index space, distributed index space (DIS) for dense systems, and unassembled distributed index space for sparse systems. He illustrated this with an example application in which solution points in a domain space are mapped to processors using a space-filling curve to maintain locality and the matrix distribution is induced from this vector mapping.

The primary topics of the discussion sessions and the Birds-of-a-Feather session were as follows:

- Should we support alternative data layouts? If so, how many?
  Several participants felt that parallel data layouts are still evolving, and that we should not commit ourselves to only supporting two-dimensional block-cyclic data decomposition.

- How are the PBLAS used outside of the ScaLAPACK project?

- Do the PBLAS and BLACS preclude the overlapping of communication with computation?
  Tony Skjellum felt that the PBLAS and BLACS preclude the overlapping of communication and computation. Clint Whaley pointed out that there is some overlap through pipelining in the ScaLAPACK codes, but the modularity desired for the ScaLAPACK codes does essentially preclude such abilities. Fred Gustavson pointed out that some current hardware does not support the overlap of communication and computation. Clint said that he had experimented with an asynchronous version of the BLACS but because the hardware could not support such an overlap, the added complexity of the calling sequences did not justify the inclusion of such abilities at this time.

- Should ScaLAPACK, and the PBLAS in particular, allow "inter-context" operations? Some participants felt that the added complexity for "inter-context" operations is warranted and thus should be supported in parallel libraries.

- How much of a performance hit are we willing to accept to accommodate input error-checking?
  This issue was debated for all BLAS libraries. In distributed-memory programming, more severe consequences can occur as a result of a mistake, but the overhead for error-checking is also much higher.

4.3 Message Passing (Communication)

Clint Whaley of the University of Tennessee presented the basic calling interfaces in the BLACS for point-to-point, broadcast, and collective communication. The BLACS context (similar to an MPI communicator) provides the ability to have separate "universes" of message passing, i.e., safe co-existence with other parallel libraries. Versions of the BLACS on top of CMMD, MPL, NX, PVM, and MPI are available on netlib. He gave a brief discussion of why the BLACS are used in ScaLAPACK instead of MPI. MPI did not exist when the ScaLAPACK project was begun. A slightly different set of "wrappers" (BLACS) would still have been developed even if MPI had existed at that time because there is not a one-to-one correspondence between BLACS and MPI calls. For our implementation, the BLACS calling sequences are patterned after the BLAS and are well-suited for linear algebra computations performed in ScaLAPACK. He finished by discussing the difficulties he encountered when implementing the BLACS on top of MPI.

David Walker of the Oak Ridge National Laboratory presented a talk on the design of out-of-core routines for ScaLAPACK. He focused on the need to establish a standard for parallel input/output, and proposed a set of primitives called BLAPIOS (Basic Linear Algebra Parallel I/O Subprograms).

5 Implementation Issues

Bo Kägstrom of Umeå University presented a talk about the implementation and performance evaluation benchmark for his GEMM-based Level 3 BLAS. (A GEMM-based Level 3 BLAS is a reorganization of the Level 3 BLAS to be rich in general matrix multiplication and addition operations). The benchmark evaluates the performance of vendor-supplied

BLAS through comparisons with the GEMM-based BLAS performance, and identifies the computational areas where improvements could be made.

Jeff Bilmes of the University of California, Berkeley presented a talk on automatic generation of BLAS routines using PHIPAC (Portable HIgh Performance ANSI C). The goals of the PHIPAC project are: to produce a fast, portable, public domain, tunable matrix-vector library in C; maintain a BLAS-friendly interface; and, automatically produce high performance routines. For his methodology, first he makes the compiler's job as easy as possible by only relying on it for instruction selection, scheduling, and register allocation. Second, he generates C code that is easy to optimize by removing false dependencies and using machine sympathetic constructs. And finally, he writes parameterized C code generators and tuning scripts to achieve optimal performance. The target machine for this package is a RISC-based workstation. This package makes it possible to offer a high performance BLAS package for many machines, and can be used for comparison against a vendor-supplied BLAS library.

Michel Daydé of ENSEEIHT-IRIT presented a talk about a tuned version of the Level 2 and 3 BLAS for RISC processors. It is based on data copying, loop unrolling and blocking techniques and is publicly available. The block size is chosen according to machine characteristics such as the cache size. Small matrices are the object of particular attention as far as performance is concerned. This tuned BLAS implementation achieves approximately 80% and 60% of the peak performance per node for single and double precision arithmetic respectively on the Meiko CS-2. Similar experiments were conducted on a variety of machines and results reported as well. He also installed the ScaLAPACK library on a 16-node Meiko CS-2. By using the tuned BLAS implementation discussed earlier, his timing results showed that classic factorizations such as LU or QR decompositions achieve reasonable efficiency even on relatively small problem sizes. These results are very encouraging for his current development of (direct multifrontal) sparse solvers on distributed memory MIMD computers.

Fred Gustavson of IBM presented a talk on the implementation and performance of an optimal three-dimensional parallel matrix multiply. The three-dimensional parallel matrix multiplication approach has a factor of $P^{1/6}$ less communication than the two-dimensional parallel algorithms, but does in general require more workspace than some of the two-dimensional algorithms. This three-dimensional algorithm has been implemented on an IBM SP2 and has yielded close to the peak performance of the machine, with the caveat that an initial data rearrangement is typically necessary. Extensions using Strassen's method at either an inner or outer level were outlined. An analysis of current Level 3 BLAS showed that the vast majority, if not all, of these routines allow for three-dimensional formulations, raising the question as to whether future PBLAS should incorporate three-dimensional mappings as a standard.

Jin Li of Mississippi State University presented a talk on a polyalgorithm approach for parallel dense matrix multiplication. The algorithms that can be selected are Cannon's approach, Fox's approach, and the Broadcast-Broadcast approach. The decision of which algorithm to choose depends primarily upon the size of the matrix. It was implemented in C and MPI on an IBM SP2.

Almadena Chtchelkanova of the University of Texas, Austin, presented the performance results of a Level 3 Parallel BLAS implementation with the physically based matrix distribution scheme.

Ken Stanley of the University of California, Berkeley, gave a talk about performance issues for LU decomposition and reduction to tridiagonal form using ScaLAPACK and the PBLAS. He also discussed the impact of changing the two-dimensional block cyclic distribution to a virtual two-dimensional torus wrap data layout. The virtual two-dimensional torus wrap data layout is used in the PRISM project and is proposed as a possible extension to HPF. His conclusions were the following:

- Performance considerations need not be the primary decision criteria because the competing options do not offer a compelling performance advantage.
- PBLAS error checks, if they cannot be made five times faster, should be optional.
- Collective communications implemented at a lower level will greatly improve performance on some problem sizes and machines.
- Special casing for square processor layouts offers the potential for improved performance.

6 Other Related Topics

Melody Ivory of the University of California, Berkeley, presented ideas and plans for a GAM (Generic Active Messages) implementation of the BLACS. A GAM implementation of the BLACS is necessary to integrate ScaLAPACK into the Castle and NOW projects. She proposed several approaches to the implementation. Her goals were to be portable and to obtain performance comparable to the MPL BLACS with the least amount of effort.

Vipin Kumar of the University of Minnesota gave a talk on his work with scalable parallel algorithms for sparse linear systems. Specifically, he presented his most recent performance figures for sparse multifrontal Cholesky factorization and a new multilevel graph partitioning scheme.

7 Summary

Pete Stewart of the University of Maryland gave the closing address and summary of the workshop. He began by thanking all of the speakers for a fine group of talks, and jokingly commented that he thought we should impose a fifteen-minute time limit on all talks at conferences. He presented a brief outline of the issues that were discussed, admitting that some of the issues are difficult to categorize since there is so much overlap of ideas between the various BLAS packages.

The key issues that he discussed were the following:

- Small issues
  There was a lot of quibbling at the beginning of the workshop about issues such as naming schemes and the removal of character strings from calling sequences. While these are very important issues that must be addressed in any standardization process, we wished to concentrate more on the fundamental issues of functionality that should be supported.

- Design issues
  Several speakers stressed that more computer science related design issues such as software engineering should be incorporated into the development of mathematical libraries. The idea of "layering" of the BLAS packages in a hierarchical view of development was also presented. However, we must make a strong effort to not over-complicate the libraries, and provide the vendors with as easy a job as possible to implement these libraries efficiently. Many language issues pertaining to Fortran 77, C, C++, and Fortran 90, were heavily discussed. Some speakers felt that the BLAS are too Fortran-centric. These are tricky issues that must be considered very carefully.

- Error-checking
  Error-checking was a very strongly debated issue. Most participants felt that error-checking should only be performed at the higher level routines, not in a low-level library. Some participants proposed having multiple interfaces to a library to allow the user to debug with error-checking turned on, and gain optimal performance with error-checking turned off. However, we must make a strong effort to not over-complicate the libraries, and provide the vendors with as easy a job as possible to implement these libraries efficiently.
  This issue of error-checking and overhead also brought to the forefront the issue of good library performance for small matrices. The term "small" refers to small in either matrix dimension. Several participants pointed out that small matrix computations naturally occur in sparse matrix calculations, and should be taken into account when considering library performance.

- Functionality
  Functionality is always an issue of division among participants in a workshop. It is important that we re-evaluate the functionality of the BLAS as a result of experience with using them in existing software packages. There was an effort reflected in the presentations at the workshop to produce GEMM-based BLAS, thus reducing the Level 3 BLAS to matrix multiplication. This is an attempt to reduce the functionality of the BLAS. Other participants argued for increased functionality and extensions to the existing BLAS routines.

- Modeling performance
  Modeling the performance of an algorithm has always been an important issue in software development. These "back-of-the-envelope" calculations are always encouraged and we welcome new tools to aid in this evaluation.

- Sparse matrices
  It is important to note that real progress was made in the area of developing Sparse BLAS when the developers stopped trying to implement everything. When the focus was taken from direct methods to iterative methods, considerable progress was made.

- Parallel efforts
  Alternative but similar views were presented for the concept of matrix distribution. Several participants felt that the area of data layout is still evolving and alternative data layouts for parallel applications should be supported.

- Standardization
  Many issues are still unresolved in the area of parallel library development. Quick decisions should not be made simply in the effort to establish a standard. Further implementation needs to be done before a better feel can be established for the needs of a parallel library.
  However, an effort can be made in areas such as extensions to the BLAS.

8 Future Meetings

It was agreed to arrange a Birds-of-a-Feather session at SuperComputing '95 in San Diego, California, to accommodate additional discussion on these BLAS related issues. The session has been scheduled for Thursday, November 7 at 12:15pm PST in Room 8 of the Conference Center. It is expected that a series of meetings, perhaps styled after the HPF and MPI Forums, will be initiated.

Appendix A

Agenda for the Workshop

Additional information for the workshop can be found on the web page.

http://www.netlib.org/utk/papers/sblas-meeting.html

A.1 Session 1

- Jack Dongarra, Chair
- "Historical Perspective",
  Sven Hammarling, NAG Ltd.
- "Standard Sequential Mathematical Libraries: Promises and Pitfalls, Opportunities and Challenges",
  Andrew Lumsdaine, University of Notre Dame
- "Requirements for Parallel BLAS: A Library Writer's Perspective",
  Bill Gropp, Argonne National Laboratory
- "On the Sparse BLAS Work",
  Iain Duff, Rutherford Appleton Laboratory
- Discussion

A.2 Session 2

- Iain Duff, Chair
- "Sparse BLAS, Toolkits and Primitives",
  Mike Heroux, Cray Research, Inc.
- "Basic Linear Algebra Communication Subprograms",
  Clint Whaley, University of Tennessee
- "Parallel BLAS Used by ScaLAPACK",
  Antoine Petitet, University of Tennessee

- Discussion

A.3 Session 3

- Mike Heroux, Chair
- "Parallel Givens and a Better Symmetric Update",
  Linda Kaufman, Bell Labs
- "Parallel Matrix Distributions: Have We Been Doing It All Wrong?",
  Robert van de Geijn, University of Texas, Austin
- "Physically Based Matrix Distribution: Theory and Interface",
  Carter Edwards, University of Texas, Austin
- "OBLAS: Objective Basic Linear Algebra Subprograms (One Call for all LAS)",
  Craig C. Douglas, IBM T.J. Watson Research
- Discussion

A.4 Session 4

- Sven Hammarling, Chair
- "Fortran90 Version of the BLAS",
  Jeremy Du Croz, The Numerical Algorithms Group, Ltd.
- "Key Concepts for Parallel Out-of-Core LU Factorization",
  David Walker, Oak Ridge National Laboratory
- "The Importance of Highly Efficient Computational Kernels for All Block Sizes",
  Barry Smith, Argonne National Laboratory
- "Future Research Directions in Scalable Software Libraries",
  Tony Skjellum, Mississippi State University
- "P_SPARSLIB: A Parallel Sparse Iterative Solution Package",
  Yousef Saad, University of Minnesota
- Discussion

A.5 Birds-of-a-Feather: Sparse BLAS

- Iain Duff and Mike Heroux, Co-chairs.

A.6 Birds-of-a-Feather: Matrix Distribution

- Jack Dongarra, Bo Kågström, and Robert van de Geijn, Co-chairs.

A.7 Birds-of-a-Feather: Extensions to the BLAS

- Jeremy Du Croz and Linda Kaufman, Co-chairs.

A.8 Session 5

- Jeremy Du Croz, Chair
- "GEMM-Based Level 3 BLAS: High Performance Model Implementations and Performance Evaluations Benchmark",
  Bo Kågström, Umeå University
- "Portable Automatic Generation of Fast BLAS-GEMM Compatible Matrix-Matrix Multiply Using PHiPAC Techniques",
  Jeff Bilmes, University of California, Berkeley
- "A GAM Implementation of the BLACS",
  Melody Ivory, University of California, Berkeley
- Discussion

A.9 Session 6

- Bo Kågström, Chair
- "Parallel BLAS and ScaLAPACK Results on Meiko",
  Michel Daydé, ENSEEIHT-IRIT, Toulouse
- "3-D Parallel Matrix Multiply",
  Fred Gustavson, IBM
- "A Poly-Algorithm for Parallel Dense Matrix Multiplication on Two-Dimensional Process Grid Topologies",
  Jin Li, Mississippi State University
- "Issues in Standardizing the Parallel BLAS",
  Steve Huss-Lederman, SuperComputing Research Center
- Discussion

A.10 Session 7

- Tony Skjellum, Chair
- "Efficient Parallel Level 3 BLAS Implementation",
  Almadena Chtchelkanova, University of Texas, Austin

- "Performance impacts of PBLAS interface and implementation decisions on LU decomposition and reduction to tridiagonal form",
  Ken Stanley, University of California, Berkeley
- "On C/C++ Work on the Sparse BLAS",
  Roldan Pozo, NIST
- Discussion

A.11 Session 8

- Jack Dongarra, Chair
- "Highly Parallel Formulations of Sparse Matrix Computations",
  Vipin Kumar, University of Minnesota
- "Summary and Wrap-up",
  Pete Stewart, University of Maryland
- Discussion

Appendix B

List of Attendees

Ed Anderson
Cray Research, Inc.
eca@cray.com

Zhaojun Bai
University of Kentucky
bai@ms.uky.edu

Mike Berry
University of Tennessee
berry@cs.utk.edu

Jeff Bilmes
University of California, Berkeley
bilmes@icsi.berkeley.edu

Almadena Chtchelkanova
University of Texas, Austin

Andy Cleary
University of Tennessee
cleary@cs.utk.edu

Michel Daydé
ENSEEIHT-IRIT, Toulouse
dayde@enseeiht.fr

Dave Dodson
Convex Corporation
dodson@convex.com

Jack Dongarra
University of Tennessee and ORNL
dongarra@cs.utk.edu

Craig Douglas
CERFACS and IBM
douglas-craig@cs.yale.edu

Jeremy Du Croz
NAG, Ltd.
jeremy@nag.co.uk

Iain Duff
Rutherford Appleton Laboratory
isd@letterbox.rl.ac.uk

Carter Edwards
University of Texas, Austin

Salvatore Filippone
IBM Rome
filippon@vnet.ibm.com

Rob Gjertsen
UIUC
gjertsen@ncsa.uiuc.edu

Bill Gropp
Argonne National Laboratory
gropp@mcs.anl.gov

John Gunnels
University of Texas, Austin

Fred Gustavson
IBM
gustav@watson.ibm.com

Sven Hammarling
NAG, Ltd.
hammarli@cs.utk.edu

Greg Henry
Intel Corporation
ghenry@cs.utk.edu

Mike Heroux
Cray Research Inc.
mamh@appsdiv.cray.com

Steve Huss-Lederman
SuperComputing Research Center
lederman@super.org

Melody Ivory
University of California, Berkeley
ivory@maxima.cs.berkeley.edu

Bo Kågström
Umeå University
bokg@cs.umu.se

Chandrika Kamath
DEC
kamath@caldec.enet.dec.com

Linda Kaufman
Bell Labs
lck@research.att.com

David Kincaid
University of Texas, Austin
na.kincaid@na-net.ornl.gov

Vipin Kumar
University of Minnesota
kumar@cs.umn.edu

Jin Li
Mississippi State University

Steve Lee
Oak Ridge National Laboratory
slee@msr.epm.ornl.gov

Andrew Lumsdaine
University of Notre Dame
lumsdaine.1@nd.edu

Dave Mackay
Intel Corporation
mackay@intel2.ccs.ornl.gov

Kristyn Marshoffe
Rice University
kristyn@caam.rice.edu

Brian McCandless
University of Notre Dame

Joan McComb
IBM
mccomb@vnet.ibm.com

Noel Nachtigal
Oak Ridge National Laboratory
santa@msr.epm.ornl.gov

Susan Ostrouchov
University of Tennessee
sost@cs.utk.edu

Antoine Petitet
University of Tennessee
petitet@cs.utk.edu

Roldan Pozo
NIST
pozo@cam.nist.gov

Karin Remington
NIST
karin@cam.nist.gov

Yousef Saad
University of Minnesota

saad@cs.umn.edu

Majed Sidani
Cray Research, Inc.
sidani@cray.com

Barry Smith
Argonne National Laboratory
bsmith@merlin.mcs.anl.gov

Tony Skjellum
Mississippi State University
tony@aurora.cs.msstate.edu

Ken Stanley
University of California, Berkeley
stanley@orodruin.cs.berkeley.edu

Pete Stewart
University of Maryland
stewart@cs.umd.edu

Anna Tsao
SuperComputing Research Center
anna@super.org

Robert van de Geijn
University of Texas, Austin
rvdg@cs.utexas.edu

Phuong Vu
SGI
pav@houst.sgi.com

David Walker
Oak Ridge National Laboratory
walker@rios2.epm.ornl.gov

Bob Ward
University of Tennessee
ward@cs.utk.edu

Clint Whaley
University of Tennessee
rwhaley@cs.utk.edu

Bibliography

[1] D. S. Dodson, R. G. Grimes, and J. G. Lewis, "Sparse Extensions to the FORTRAN Basic Linear Algebra Subprograms," *ACM Trans. Math. Soft.*, 17, 1:253-272, 1991.

[2] D. S. Dodson and J. G. Lewis, "Issues Relating to Extension of the Basic Linear Algebra Subprograms," *ACM Signum Newsletter*, 20, 2-18, 1985.

[3] J. Dongarra, J. Du Croz, I. Duff, and S. Hammarling, "A Set of Level 3 Basic Linear Algebra Subprograms," *ACM Trans. Math. Soft.*, 16, 1:1-17, March 1990.

[4] J. Dongarra, J. Du Croz, S. Hammarling, and R. Hanson, "An Extended Set of Fortran Basic Linear Algebra Subprograms," *ACM Trans. Math. Soft.*, 14, 1:1-17, March 1988.

[5] C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh, "Basic Linear Algebra Subprograms for Fortran Usage," *ACM Trans. Math. Soft.*, 5, 3:308-323, September 1979.