Jack Dongarra: Supercomputing Expert and Mathematical Software Specialist

Thomas Haigh
University of Wisconsin

Editor: Thomas Haigh

Jack J. Dongarra was born in Chicago in 1950 to a family of Sicilian immigrants. He remembers himself as an undistinguished student during his time at a local Catholic elementary school, burdened by undiagnosed dyslexia. Only later, in high school, did he begin to connect material from science classes with his love of taking machines apart and tinkering with them. Inspired by his science teacher, he attended Chicago State University and majored in mathematics, thinking that this would combine well with education courses to equip him for a high school teaching career. The first person in his family to go to college, he lived at home and worked in a pizza restaurant to cover the cost of his education.[1]

In 1972, during his senior year, a series of chance events reshaped Dongarra's planned career. On the suggestion of Harvey Leff, one of his physics professors, he applied for an internship at nearby Argonne National Laboratory as part of a program that rewarded a small group of undergraduates with college credit. Dongarra credits his success here, against strong competition from students attending far more prestigious schools, to Leff's friendship with Jim Pool who was then associate director of the Applied Mathematics Division at Argonne.[2]

Eispack

Dongarra was supervised by Brian Smith, a young researcher whose primary concern at the time was the lab's Eispack project. Although budget cuts forced Pool to make substantial layoffs during his time as acting director of the Applied Mathematics Division in 1970–1971, he had made a special effort to find funds to protect the project and hire Smith. Smith was the main Argonne coordinator for Eispack and one of its key technical leaders. Dongarra was pressed into service running test problems through the software and reporting errors to its developers.

Eispack was an effort to produce accurate, robust, efficient, well-documented, and portable Fortran implementations of the pathbreaking mathematical methods for matrix eigenvalue and eigenvector calculation introduced during the late 1960s by James H. Wilkinson and Christian H. Reinsch.[3] Eigenvalue calculation is crucial in many areas of scientific computing and modeling, from mechanics and geology to Google's PageRank algorithm. Packaging dependable and efficient Fortran subroutines relieved application programmers of the need to attempt their own implementations. Eispack eventually included around 70 routines, most specialized for particular kinds of matrices, together with a set of higher-level "driver" programs to give application programmers an easier-to-use interface.

Dongarra was put to work testing Eispack. This was not quite the first time Dongarra had used a computer. He was unaware of any undergraduate computer courses at Chicago State, though he had struck up a brief and unrewarding relationship with an early Wang computer in the physics department. But his experience with Argonne's IBM 360/75 hooked Dongarra on computing, and kindled an interest in numerical methods.

Background of Jack Dongarra

Born: July 18, 1950, Chicago, Illinois.

Education: Chicago State University: BS (mathematics), 1972; Illinois Institute of Technology: MS (computer science), 1973; University of New Mexico: PhD (mathematics), 1980.

Positions held: *Argonne National Laboratory:* resident student associate, 1973; research associate, 1974; assistant computer scientist 1975–1980; senior computer scientist 1980–1989; *University of Tennessee:* distinguished professor, 1989–2000; university distinguished professor 2000–present; *Oak Ridge National Laboratory:* distinguished scientist, 1989–2000; adjunct R&D participant 2000–present; *Visiting appointments*: Los Alamos Scientific Laboratory (1978); Stanford University (1979); IBM T.J. Watson Research Center (1981); University of Illinois Urbana-Champaign (1985–1987); AERE Harwell (1987); IBM ECSEC, Rome (1990); ETH, Zurich (1992); Ecole Normale Supérieure de Lyon (1993); Danish Technical University, Lyngby (1994); Swiss Scientific Computing Center, Manno (1995); Australian National University, Canberra (1996); KTH Royal Institute of Technology, Stockholm (1997); Oxford University (1998); CERFACS Toulouse (1999); Ecole Polytechnique Fédérale de Lausanne (June 1999); SFI Walton Visitor, University College Dublin (2004); Turing Fellow University of Manchester (2007–present).

Selected Honors and Awards: Fellow of American Association for the Advancement of Science (1995); Fellow of the IEEE (1999); elected to the National Academy of Engineering (2001); Fellow of the ACM (2001); IEEE Sid Fernbach Award (2003).

Published by the IEEE Computer Society 1058-6180/08/$25.00

EISPACK included a suite of automatic test programs that ran a set of calculations and tested the results to find errors caused by unexpected interactions with the logic circuitry and compilers in use at different test sites. These were supplied to a range of computer centers, so that the code could be fully tested on IBM, CDC, Univac, DEC, Amdahl, and Honeywell computers. Testing the package to certify its reliability was a crucial part of the effort; indeed, the EISPACK project's official rationale was as part of an initiative called the National Activity to Test Software.[4]

Dongarra's internship extended into summer work as thoughts of a high school teaching career faded. Meanwhile, his part began to develop beyond simply testing the EISPACK routines and into debugging and improving them.

Setting aside plans for a master's degree in physics, he made a late application to the Illinois Institute of Technology for its computer science program. Pool's recommendation won Dongarra a teaching assistantship opened by a last-minute cancellation. He continued to work at Argonne one day a week, and was able to choose Brian Smith as the supervisor of his master's thesis on an algorithm for efficiently handling a matrix too large for core memory by reducing it to tridiagonal form. By the end of his three semesters at IIT in 1973, Dongarra had earned not only a master's degree but also a full-time job at Argonne.

The software team was a tightly knit group, held together by friendship and social ties as well as by a shared commitment to the creation of high-quality mathematical routines. As Pool recalls, "Every once in a while people would holler at each other, but in general it was a great time. We had parties at Jack's house, my house, at the lab on Thursday nights, and so forth."[4]

EISPACK was released in 1974, though work continued for some time to produce improved releases. Surrounded by Argonne researchers and distinguished academic visitors, Dongarra soon realized that he would need a doctoral degree to progress in his new career and legitimate himself within the emerging mathematical software community. Taking a leave from Argonne, Dongarra enrolled in the mathematics PhD program of the University of New Mexico, chosen because its faculty included Cleve Moler whom Dongarra had known since his time as an Argonne intern, where one of his responsibilities had been introducing Moler to the local computing facilities.

Figure 1. Jack Dongarra, a leader of the high-performance computing community, is cocreator of mathematical software packages including EISPACK, LINPACK, LAPACK, and ScaLAPACK.

Dongarra began his PhD course work in 1977 with a semester of night school classes at Los Alamos, where Bill Buzbee's numerical methods group at Los Alamos had provided him with a visiting position. The lab had recently taken delivery of the first Cray 1 computer for evaluation purposes, and like other early users, Dongarra struggled to adapt existing methods and algorithms to function effectively on its novel vector architecture.[5] He continued to spend at least one day a week at Los Alamos during his two semesters in residence at the University of New Mexico, working with Tom Jordan to produce linear algebra routines for the Cray. After finishing his course work, he spent a final semester working with Moler during the latter's 1979 sabbatical at Stanford.

Dongarra then returned to Argonne, where he finished his dissertation in 1980. It explored a method for improving the accuracy of eigenvalue calculations first suggested by Jim Wilkinson.[6] Looking back over his own career, Moler recently said that, "Dongarra may be my most well-known student; he's certainly made a name for himself in supercomputing."[7]

LINPACK

Dongarra's most important activity during the late 1970s was not his dissertation work but his participation in the LINPACK project.[8] The project began in the summer of 1976. Following the success of EISPACK, Pool suggest-

Linpack proved itself on practically every computer model used by scientists and engineers during the 1980s.

ed a project to create an equally capable package for the complementary activity of solving linear equations and linear least-squares problems, another foundational set of capabilities for scientific computation across a vast range of application areas. At Pool's request, G.W. "Pete" Stewart, a numerical analyst visiting the lab from the University of Maryland, chaired a public meeting to seek support for the idea. A small team formed around the idea, and Stewart wrote up a successful National Science Foundation (NSF) grant proposal. During this period Stewart recalls that "A lot of things were happening, a lot of socializing, a lot of going into Chicago for enormous free dinners … and technically exciting, too. It was just the place to spawn Linpack."[9] When work began in earnest, Stewart

> was responsible for things having to do with the QR decomposition, least squares, and singular value decomposition. Cleve [Moler] worked on the things relating to Gaussian elimination, and Jim Bunch worked on stuff relating to symmetric indefinite systems.[9]

All three were academics without regular appointments at Argonne, a remarkable testament to the lab's ability to spawn and support university collaborations.

Stewart remembers the scope of Dongarra's involvement with Linpack as having grown significantly during the course of the project: "Jack initially was basically a systems support person. He was not involved in the design of the package [but] later he did begin to take an important role in it."[9] Dongarra's primary initial responsibility was for developing a framework to test the package and its component parts. He also worked closely with Cleve Moler. The design team reconvened in Argonne each summer until 1979, "to collectively put together various pieces that they had worked on over the year and trying to make it all work. I remember those meetings as being very intense."[10] By the end of the project, according to Stewart, "Jack was showing exceptional promise and that we might as well give him a leg up and put him as the lead author. … I think we all feel that that was a very good investment."[9]

Jim Pool remembers recruiting Dongarra as one of the five most important contributions he made to Argonne. According to Pool, "Jack came in as an undergraduate, matured into a graduate student, and matured into one of the staff members who probably contributed as much or more than anybody else to Linpack as it evolved."[11]

Linpack quickly established itself as one of the most important mathematical software packages of its era. It proved itself on practically every computer model used by scientists and engineers during the 1980s, from supercomputers to Unix workstations and IBM personal computers. The code found still more users when it was resold as part of commercial software packages, including the IMSL library.

Its remarkable flexibility came in large part from its use of the BLAS (Basic Linear Algebra Subprograms). The idea of BLAS originated around 1972 with Charles Lawson and Richard Hanson of NASA's Jet Propulsion Laboratory.[12] The BLAS specifications defined a set of simple vector operations, each with a standard calling sequence. Linpack and other mathematical packages were written to use these BLAS calls. Meanwhile, the work of adapting code to work efficiently with particular machine architectures could be tackled by creating a new BLAS implementation rather than rewriting the package itself. But Linpack was the first major package to commit to BLAS, and the two projects developed a symbiotic relationship. Jim Pool remembers the agreement that Linpack would use BLAS as "a sociological breakthrough" toward the sharing of code between projects.[11,13] Linpack forced computer manufacturers to support the development of highly tuned BLASes for their machines, while BLAS gave Linpack good performance across a huge range of machine architectures without burdening its creators with the need to create and support many different versions.

The Linpack team devoted an exceptional amount of effort to its user guide. As well as calling sequences, the book explained how each routine worked, how to determine the most appropriate technique to use for a particular problem, and how the different parts of the package fit together. "In the appendix to that book," recalls Dongarra, "I collected some timing information about various machines. The idea was to give users a handle on how much time it would take to solve their problem if they used our software." For the sample problem he chose a matrix of order 100, the largest that could be processed on all of the dozen or so machines on the list. The results reflected the combined performance of Linpack in conjunction with a specific BLAS implementation, Fortran compiler, and hardware configuration.

Under Dongarra's supervision, this list of performance times grew and took on an independent life as the standard measure of a machine's floating-point performance. Computer vendors proudly quoted benchmark scores and tweaked their wares to optimize performance. The list grew to encompass hundreds of results, then thousands. Even today, long after Linpack was superseded for all other purposes, the benchmark is still used by the "TOP500" project to create the most widely quoted supercomputer-performance-league table. As one of the leaders of the TOP500 project, Dongarra is frequently called on to interpret shifts in its quarterly rankings of the world's fastest computers. According to Hans Werner Meuer, another of the project's leaders, "[T]here was, and still is, no alternative to Linpack. Any other benchmark … would not have been so easily available for all systems … [R]unning Linpack to measure the performance is kind of a first reliability test for new HPC systems."[14]

Netlib

One of Dongarra's other projects during the 1980s was the creation of the mathematical software library Netlib. The project had its genesis in Dongarra's semester at Stanford. Among the friends he made there were eminent numerical analyst Gene Golub and a young student named Eric Grosse. Golub suggested the idea of an electronic repository of software to ensure that code written by students as part of their research could be found and reused by others. In or around 1985, Dongarra and Grosse, by then working at Bell Labs, decided to turn the idea into a reality. In its early versions, Netlib worked over email—a program would extract requests from incoming email and reply with coded messages from which the requested code could be reassembled.[15]

Clumsy as this sounds, it was a huge step forward from previous libraries such as IBM's Share library, which relied on physical movement of punched cards, magnetic tapes, or microfiche to supply code. Grosse and Dongarra served as editors, deciding which software was worthy of inclusion. Within a few years Netlib had become the leading repository for mathematical software packages, a distinction that its web-based incarnation retains to this day. Code from many existing collections, such as routines from the ACM's *Transactions on Mathematical Software*, was eventually merged into Netlib.

Dongarra remained part of the Argonne mathematical software team until 1989, with an appointment as scientific director of the lab's Advanced Computing Research Facility from 1985 onward. As he recalls it:

> From my first time there as an undergraduate until I left it was a very, very positive experience and one in which we all worked together. … The work environment was terrific, very rich and stimulating for a young person. … It never felt like I had to go to work, it felt like I was going and experiencing something new almost every time I would go into the place. It was almost like being at Christmas all the time, opening things up and finding new exciting things there.[16]

Lapack

Dongarra's next major project, Lapack, incorporated new mathematical methods and implementation techniques, covering the core areas of both Lapack and Eispack. As with the earlier packages, Lapack was built by a team. Dongarra and James Demmel of the University of California, Berkeley, were the primary designers, but the original 1987 prospectus named Jeremy Du Croz, Anne Greenbaum, Sven Hammarling, and Danny Sorensen as investigators. This reflected the growing institutional diversity of the mathematical software field as both Du Croz and Hammarling worked for the Numerical Algorithms Group, a British supplier of commercial mathematical software. The first public release took place in 1992, with subsequent major version releases in 1994 and 1999.

Compared with the two earlier packages, Lapack added new interface features, such as the provision of condition number estimates to users of the routines. But its key motivation was the continuing evolution of computer architectures. Flexible as Linpack was, it could not fully exploit the power of vector processing supercomputers and shared memory parallel machines, the most important of which during the early 1980s were respectively the

Figure 2. Jack Dongarra is a University Distinguished Professor at the University of Tennessee.

Cray-1 and its multiprocessor stable mate, the Cray X-MP. Fast, architecturally elegant, and visually stunning, these were the exotic sports cars of the computing world. But for the US National Laboratories and other supercomputer centers, their novel designs posed significant software challenges, with which Dongarra had been grappling since his time at Los Alamos. Although the multi-ton cooling system and sofa-like casing of the Cray computers were never going to appear in mass-market designs, their long instruction pipelines, vector processing capabilities, and shared-memory multiple-core architectures gradually moved from supercomputing centers into high-performance workstations, eventually becoming standard features of personal computers.[17]

LAPACK was developed in parallel with new BLAS specifications, dubbed Level 2 and Level 3 BLAS. The original Level-1 BLAS covered only scalar, vector, and vector-vector operations (i.e., those processing at most individual columns from two matrices). This insulated application programmers from many machine-dependent factors, but still required them to make major changes to their application to optimize it for specific vector processing, cache memory, or multiprocessor architectures.

The new BLAS implementations covered standard operations on the combination of a matrix and a vector (Level 2) and two matrices (Level 3). Dongarra remembers that "we defined an interface and had a reference implementation and test routines for [Level-2 BLAS for LINPACK], and that was done in 1987."[18] This higher level of abstraction was the key to LAPACK's ability to run efficiently on a huge range of architectures.

Unlike the Level 1 BLAS, the new BLAS levels required fundamental changes to existing matrix algorithms so that they worked at the level of larger data blocks rather than manipulating individual matrix elements. According to Dongarra, LAPACK was "more complicated because its level of modularity is quite a bit more extreme, it has more involvement with the performance which changes from machine to machine and from architecture to architecture."[19] Contributor Hammarling believes that "LAPACK is perhaps the most outstanding example" of a system built using "a lot of state-of-the-art algorithms."[20]

University of Tennessee

In 1989 Dongarra left Argonne to accept dual appointments as a Distinguished Professor at the University of Tennessee, Knoxville, and Distinguished Scientist at the nearby Oak Ridge National Laboratory (see Figure 2). Dongarra felt that "it would allow me to be a professor and to have students and to do other things that were perhaps not as easy to do at the lab." At the same time, Oak Ridge gave him an environment similar to what he had enjoyed at Argonne. Over time, however, his role at Oak Ridge dwindled to a consulting position—primarily because the lab's bureaucracy and high overhead rates on grant money made it an unattractive destination for his external funding. In the university environment, he felt more able to "act as an entrepreneur."[21]

Although LAPACK was begun while Dongarra was at Argonne, much of his work on the project was conducted from his new research group at Tennessee. To begin with, this group consisted of just one graduate assistant. By 2004 his Innovative Computing Laboratory had grown to encompass

> about 50 people including research professors, postdocs, research assistants, graduate students, undergraduate students, programmers, an artist, secretarial staff, people who take care of our computers, and our support staff as well.[22]

Except for Dongarra himself, the entire team has been supported by a series of major grants and research contracts from the NSF, US Defense Advanced Research Projects Agency (DARPA), Department of Energy, and other sources. The scale of the lab and its success at attracting funding has made it unique in the world of academic mathematical software, closer to the model of major hard science or biomedical research than to the traditional pattern in mathematics of a researcher with a few graduate students and a network of collaborators at other institutions. It dwarfs the university's computer science department and, according to Dongarra, brings in more grant money than any other entity in the University of Tennessee system other than its medical school. As the group has grown larger, Dongarra's personal role has evolved. Rather than try to directly manage all 50 staff members, he has delegated responsibility for many aspects of the lab and its day-to-day operations. Since 2001, Dongarra has also directed the university's Center for Information Technology Research, a "center of excellence" currently including two other labs in addition to his own Innovative Computing Laboratory.

Dongarra has co-authored and co-edited a huge and ever-growing body of books, book chapters, and research articles. He has come to serve as a public face of the supercomputing user community. Since 1992, he has been editor in chief of the *International Journal of High Performance Computing* as well as serving the Society for Industrial and Applied Mathematics, founding its interest group on supercomputing and participating in many of the society's other activities.

Since the release of LAPACK, Dongarra has continued to grapple with the challenges new computer architecture has posed for linear algebra. LAPACK worked well with multiprocessor computers based on shared memory, such as the Cray X-MP and its successor, the Y-MP. Like today's high-end personal computers, these machines used up to eight processors working in parallel. But by the end of the 1990s, a wave of enthusiasm for massively parallel supercomputers had swept over the industry. These computers were sometimes called distributed memory machines, because each processor or small cluster of processors had its own memory. The idea of a supercomputer built from thousands of cheap processors was attractive, but existing applications, algorithms, and software tools could unlock only a tiny fraction of the hardware's theoretical performance. Because of these problems, the massively parallel supercomputer industry suffered serious setbacks during the mid-1990s, but similar issues soon surfaced with the rise of cluster computing in which large numbers of personal computers are networked together to create a single virtual computer of enormous power. Many of the computers on the TOP500 list in recent years have been clusters of hundreds of personal computers.

The idea of a supercomputer built from thousands of cheap processors was attractive, but existing tools could unlock only a fraction of the hardware's theoretical performance.

The challenge was in splitting a large job into many small independent tasks and then finding an efficient way to coordinate their operation via the exchange of messages between processes. The ScaLAPACK (Scalable LAPACK) project implemented many of LAPACK's routines in a form suitable for distributed memory computers, yielding an initial release in 1993 followed by improved versions in 1996 and 1997.

Like LAPACK, the project involved collaboration with researchers at several other universities. Dongarra and Demmel were again the principal designers. Retaining the familiar LAPACK structures made it much easier to transfer existing applications and programming techniques to the new architectures. DARPA provided two grants totaling almost six million dollars to support the project. For ScaLAPACK to work, it needed a standard mechanism to insulate application programs from the details of the cluster of machines on which they were being executed by presenting them with a single virtual parallel computer.

The project co-evolved with a new system known as PVM (Parallel Virtual Machine). Adam Beguelin and Robert Manchek from Dongarra's group at the University of Tennessee worked in collaboration with Al Geist at Oak Ridge National Laboratory and Vaidy Sunderam at Emory University to create PVM. Like BLAS, this technology was quickly adopted by many other scientific computing projects.

Dongarra also played a crucial role in the creation of the standard Message Passing Interface (MPI) for distributed computation. Dongarra chaired the first MPI standards group, leading to Version 1.0 of the standard in 1994. He says that

> MPI was designed by a committee of twenty people from commercial settings, academic institutions, research centers, and vendors, all pouring their ideas into this package, and because it was a committee it was done by consensus. It has a lot of ideas in it, some of which are really terrific and some of which are rather obscure, but that's the nature of committee work.[23]

MPI quickly won widespread adoption as the standard interface for application programmers in many languages to control communication and synchronization between distributed processes.

The need for efficient BLAS implementations continued to motivate Dongarra's research group during the 1990s. As he puts it,

> Our software packages get structured on top of these BLAS operations, and we hope that the vendors will go off and write a good implementation of the BLAS. In many cases they did, in some cases they didn't, and sometimes the software was generated but not made freely available. That caused some stress in the sense that we didn't have a way of getting portable performance out of our software, and we started thinking about how we could implement a portable implementation of the BLAS which would be efficient.[24]

The result was the ATLAS (Automatically Tuned Linear Algebra Software) project, undertaken by Dongarra's students Clint Whaley and Antoine Petitet, to automate the process of creating a highly optimized BLAS for a particular computer architecture. After probing the basic capabilities of the machine it is running on, ATLAS creates many different versions of each BLAS routine and determines experimentally which of these give optimum performance.[25]

Dongarra's most recent project, NetSolve, has reflected another architectural shift in technical computation. By 2004, ''grid computing'' was a much discussed idea (albeit one applied to many different approaches). NetSolve reflected the idea of computing power as a service available to users anywhere in a network. Users are provided with a mechanism to submit matrices via the network and receive back solutions without having to download and install software to carry out the actual calculations.

Looking to the future, Dongarra said of his research group, ''I don't see any major changes or shifts that would occur in terms of how we do business.''[26] When asked about research directions for the future, he said there were

> many unanswered questions, primarily dealing today with the large numbers of processors. There are four areas of great research interest: the effective use of multicore architectures, self adapting and auto tuning of software, fault tolerance algorithms and software, and exploiting mixed precision in our numerical computations. These are going to present great challenges and have a major impact over the next ten years.[27]

Acknowledgments

This biography and the oral history interview from which it draws were created under contract to the Society for Industrial and Applied Mathematics as part of the SIAM History Project supported by grant # DE-FG02-01ER25547 awarded by the US Department of Energy. Transcripts and other project documents are available at http://history.siam.org and will be added to the permanent collection of the Charles Babbage Institute.

References and notes

1. J.J. Dongarra, oral history interview by T. Haigh, 26 Apr. 2004, Soc. for Industrial and Applied Mathematics, Philadelphia; available from http://history.siam.org/oralhistories/dongarra.htm. Otherwise unattributed material throughout this biography is taken from this source.
2. The history of scientific computing at Argonne is discussed in C.N. Yood, *Argonne National Laboratory and the Emergence of Computer and Computational Science, 1946–1992*, history, Pennsylvania State Univ., 2005. Pages 177-199 cover EISPACK and Pool's leadership of mathematical software work at Argonne during this period.
3. The EISPACK project is discussed in C.B. Moler and J.J. Dongarra, ''EISPACK—A Package for Solving Matrix Eigenvalue Problems,'' *Sources and Development of Mathematical Software*, W.R. Cowell, ed., Prentice Hall,1984, pp. 68-87. See also T. Haigh, ''Biography of Cleve Moler,'' *IEEE Annals of the History of Computing*, vol. 30, no. 1, 2008, pp. 87-91.
4. See W.J. Cody, ''Observations on the Mathematical Software Effort,'' *Sources and*

Development of Mathematical Software, W.R. Cowell, ed., Prentice Hall,1984, pp. 1-19.

5. The influence of atomic weapons labs on the evolution of supercomputer architecture and the challenge sposed by the array processing architecture of the unsuccessful CDC STAR-100 machine of the 1960s are discussed in D. Mackenzie, ''The Influence of Los Alamos and Livermore National Laboratories on the Development of Supercomputing,'' *Annals of the History of Computing*, vol. 13, no. 2, 1991, pp. 179-291. Cray's interaction with its customers is explored in B. Elzen and D. MacKenzie, ''The Social Limits of Speed: The Development and Use of Supercomputers,'' *IEEE Annals of the History of Computing*, vol. 16, no. 1, 1994, pp. 46-61.
6. Dongarra dissertation findings were published as J.J. Dongarra, C.B. Moler, and J.H. Wilkinson, ''Improving the Accuracy of Computed Eigenvalues and Eigenvectors,'' *SIAM J. Numerical Analysis*, vol. 20, no. 1, 1983, pp. 23-45.
7. C. Moler, oral history interview by T. Haigh, 8–9 Mar., 2004, Santa Barbara, Calif., Soc. for Industrial and Applied Mathematics, Philadelphia; available from http://history.siam.org/oralhistories/moler.htm, p. 61.
8. J.J. Dongarra and G.W. Stewart, ''LINPACK—A Package for Solving Linear Systems,'' *Sources and Development of Mathematical Software*, W.R. Cowell, ed., Prentice Hall,1984, pp. 20-48.
9. G.W.P. Stewart, oral history interview by T. Haigh, 5–6 Mar. 2006, Soc. for Industrial and Applied Mathematics, Philadelphia (publication forthcoming).
10. J.J. Dongarra, oral history interview, p. 23.
11. J. Pool, oral history interview with T. Haigh, 14 July 2004, Soc. for Industrial and Applied Mathematics, Philadelphia, (publication forthcoming).
12. Lawson discussed the BLAS and his role in its development in C.L. Lawson, oral history interview with T. Haigh, 6–7 Nov. 2004, Soc. for Industrial and Applied Mathematics; available from http://history.siam.org/oralhistories/lawson.htm and in C.L. Lawson, ''Background, Motivation and a Retrospective View of the BLAS,'' *Proc. Ninth SIAM Conf. Parallel Processing for Scientific Computing* (CD-ROM), Soc. for Industrial and Applied Mathematics, 1999.
13. Pool believes this agreement was reached during a SIAM meeting in Madison in May 1978, though Lawson remembers contact with Argonne participants several years earlier.
14. See J.J. Dongarra, ''The LINPACK Benchmark: An Explanation,'' *Evaluating Supercomputers: Strategies for Exploiting, Evaluating and Benchmarking Computers with Advanced Architectures*, Chapman & Hill,1990, pp. 1-21.
15. The origins of Netlib are documented in J.J. Dongarra and E. Grosse, ''Distribution of Mathematical Software by Electronic Mail,'' *Comm. ACM*, vol. 30, no. 5, 1987, pp. 403-407.
16. J.J. Dongarra, oral history interview, p. 37.
17. From the Pentium onward, Intel's mainstream processors have included superscalar designs with multiple pipelines. As pipeline length increased (with the Pentium II, Pentium III, and Pentium IV chips), the issues of latency and branch prediction faced by chip designers and compiler creators mirrored those faced by the Cray programmers of the 1970s. Meanwhile, new instructions introduced with 1996's Pentium MMX and expanded with each subsequent chip generation provided the single instruction, multiple data (SIMD) instructions pioneered by vector processing machines like the Cray X-MP. Though conceived as special-purpose multimedia enhancements, these instructions are now used by mathematical libraries. Efforts to harness the powerful SIMD capabilities of modern video game hardware for general-purpose numerical calculations have attracted great attention recently.
18. J.J. Dongarra, oral history interview, p. 54. On the higher levels of BLAS, see J.J. Dongarra et al., ''A Set of Level 3 Basic Linear Algebra Subprograms,'' *ACM Trans. Mathematical Software*, vol. 16, no. 1, 1990, pp. 1-17. Their relationship to LAPACK and Dongarra is discussed in I.S. Duff, oral history by T. Haigh, 30–31 Aug. 2005, Soc. for Industrial and Applied Mathematics, Philadelphia; available from http://history.siam.org/oralhistories/duff.htm.
19. J.J. Dongarra, oral history interview, p. 34.
20. B. Ford, S. Hammarling, and S. Hague, oral history interview by T. Haigh, 29–30 June 2004, Soc. for Industrial and Applied Mathematics, Philadelphia (publication forthcoming).
21. J.J. Dongarra, oral history interview, p. 51.
22. Ibid., p. 67.
23. Ibid., p. 57.
24. Ibid., p. 54.
25. R.C. Whaley, A. Petitet, and J.J. Dongarra, ''Automated Empirical Optimization of Software and the ATLAS Project,'' *Parallel Computing*, vol. 27, no. 1-2, 2001, pp. 3-35.
26. J.J. Dongarra, oral history interview, p. 78.
27. J.J. Dongarra, personal communication, 26 Feb. 2007.

Readers may contact department editor Thomas Haigh at annals-biographies@computer.org.