DISTRIBUTION OF MATHEMATICAL SOFTWARE VIA ELECTRONIC MAIL

A large collection of public-domain mathematical software is now available via electronic mail. Messages sent to "netlib@anl-mcs" (on the Arpanet/CSNET) or to "research!netlib" (on the Unix® network) wake up a server that distributes items from the collection. For example, the one-line message "send index" gets a library catalog by return mail.

JACK J. DONGARRA and ERIC GROSSE

A large pool of high-quality mathematical software is in use at educational, research, and industrial institutions around the country. At present this software is available from a number of distribution agents — for example, AT&T for the PORT library, IMSL, the National Energy Software Center (NESC), and the Numerical Algorithms Group (NAG). All do a fine job with the distribution of large packages of mathematical software, but there is no provision for convenient distribution of small pieces of software. Currently scientists transmit such software by magnetic tapes, but contacting authors and deciphering alien tape formats wastes an intolerable amount of time.

A new system, *netlib*, provides quick, easy, and efficient distribution of public-domain software to the scientific computing community on an as-needed basis.

A user sends a request by electronic mail to *netlib@anl-mcs* on the Arpanet or to *research!netlib* on the Unix uucp network. (Gateways are available to forward mail from other networks such as CSNET, Telenet, and BITNET). The two addresses mentioned are respectively at Argonne National Lab in Chicago and at AT&T Bell Labs in Murray Hill, New Jersey. A request is made up of lines of one of the following forms:

send index.
send index from *library* **.**
send *routines* **from** *library* **.**
find *keywords* **.**

Examples and a few variants of these forms are described in the next section.

This work was supported was supported in part by the National Science Foundation under Agreement No. DCR-8419437. Any opinions, findings, conclusions, and recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

NETLIB IN USE

Imagine an engineer who needs to compute several integrals numerically. He consults the resident numeric expert, who advises trying the routine *dqag* for some preliminary estimates and then using *gaussq* for the production runs. The engineer types at his terminal

mail research!netlib
send dqag from quadpack
send gaussq from go
.

In a short time, he receives back two pieces of mail from *netlibd*. The first contains the double precision Fortran subroutine *dqag* and all the routines from *quadpack* that *dqag* calls; the second contains *gaussq* and the routines it calls. A utility routine *d*1*mach* called by *gaussq* was not included, since it is probably already installed on his system; if he had wanted it, he could have changed his request to "*send gaussq from go core*" to include the ‘‘core library’’ of machine constants and basic linear algebra modules in the search list.

Should the engineer later decide that the routine *dqags* would be more effective, he could ask "*send dqags but not dqag from quadpack*" to get *dqags* and any subroutines not already sent with *dqag*.

Meanwhile, the numerical expert decides she should check on the current contents of netlib. She types

mail research!netlib
send index

The return mail shows a library *toeplitz* she is not familiar with, so she sends mail "*send index from toeplitz*" to see what is included. Curious to see a typical routine, she tries "*send only cslz from toeplitz*".

As typical examples of requests we give the following.

send dgeco from linpack

(Retrieves routine DGECO and all routines it calls from the LINPACK library.)

send only dgeco from linpack

(Retrieves just DGECO and not subsidiary routines.)

send dgeco but not dgefa from linpack

(Retrieves DGECO and subsidiaries, but excludes DGEFA and subsidiaries.)

send list of dgeco from linpack

(Retrieves just the file names rather than the contents; this can be helpful when one already has an entire library and just wants to know what pieces are needed in a particular application.)

find eigenvalue

(Retrieves the names of routines in the collection related to the keyword *eigenvalue*.)

whois golub

(Retrieves the address of Prof. Golub.)

whois france

(Retrieves all addresses of people in the database living in France.)

"Find" returns a one-line description of all routines in the collection that mention the keywords; this can be more convenient than checking the indexes for each sublibrary that might be relevant. "Whois" searches for address and telephone information in a database maintained by Gene Golub; this is soon to be supplemented by the membership files of SIAM.

Just how quickly these requests are answered depends on the speed of the network communications involved, but five or ten minutes is typical for Arpanet. CSNET or UNIX uucp may require anywhere from minutes to days to transmit a message from sender to recipient. The actual processing time is insignificant. One user wrote back enthusiastically that the system was so fast he preferred using it to hunting around on his own machine for the library software.

Netlib has been available since April 1985. To give a feel for the number of requests for software and information, we provide the following data.

And in March we received our first request from Japan!

MATERIAL AVAILABLE THROUGH NETLIB

Currently netlib offers a wide collection of public domain software as listed below:

Package	Description
BENCHMARK	Linpack and other timings
BIHAR	Bjorstad's biharmonic solver
BMP	Brent's multiple precision package
CORE	Machine constants, Basic Linear Algebra Subprograms and extensions
CALGO	Collected algorithms from ACM, published in Trans. Math. Soft.
CONFORMAL	Schwarz-Christoffel conformal mapping programs
DOMINO	a parallel programming environment from Univ. of Maryland
EISPACK	Solution of eigenvalue problems
ELEFUNT	Cody and Waite's tests for elementary functions
ERRATA	Corrections to numerical books
FISHPAK	Finite-difference approximation for elliptic BVP
FITPACK	Cline's splines under tension
FMM	Codes from book by Forsythe, Malcolm, and Moler
FNLIB	Fullerton's special-function library
FFTPACK	Swarztrauber's Fourier transforms
HARWELL	MA28 Sparse matrix routine from the Harwell library
HOMPACK	A continuation package
ITPACK	Iterative linear-systems solvers
LANCZOS	Cullum and Willoughby's Lanczos programs
LASO	Scott's Lanczos program for eigenvalues of sparse matrices
LINPACK	Solution of linear equations
LP/DATA	Linear programming test data
MACHINES	Short descriptions of various computers
MICROSCOPE	Alfeld and Harris' system for discontinuity checking
MINPACK	Nonlinear equations and least squares
MINPACK	Optimization routines
ODEPACK	Ordinary Differential Equations package
PARANOIA	Kahan's test of floating point
PCHIP	Hermite cubics by Fritsch and Carlson
PLTMG	Bank's multigrid code; too large for ordinary mail
PORT	The public subset of PORT library
PPPACK	Spline routines from de Boor
QUADPACK	Quadrature routines
SIAM	Typesetting macros for SIAM journal format

SLATEC	Machine constants and error handling package from the Slatec library
SPECFUN	Transportable special functions
TOEPLITZ	Solution of systems of equations where the matrix is toeplitz
Y12M	Package for sparse linear systems

In addition there are miscellaneous other items, such as Golub and Welsch's GAUSSQ, Cleveland's LOWESS scatterplot smoother, Bank and Smith's sparse matrix algorithm, Bjorstad's biharmonic solvers, Grosse's RAINBOW program for generating uniformly spaced colors, incomplete Cholesky factorization, Dongarra and Sorenson's TREEQR eigenvalue method, and particular Gay's nonlinear least squares package.

The various standard linear-algebra libraries are included for convenience, but the real heart of the collection lies in the recent research codes and the ‘‘golden oldies’’ that somehow never made it into standard libraries. Almost all of these programs are in Fortran. There is also a collection of errata for numerical books, descriptions and benchmark data for various computers, test data for linear programming collected by Gay, and the ‘‘na-list’’ electronic address book maintained by Gene Golub.

In addition, netlib itself—that is, this paper and the shell scripts and C codes that do the automatic processing of requests—is also available.

We do *not* send out entire libraries. A computer center setting up a comprehensive numerical library should get magnetic tapes through the usual channels.

THE NETLIB SERVER

The netlib server runs under the UNIX operating system (8th edition at Bell Labs and 4.2BSD at Argonne) and consists of a few shell scripts and C programs. The following discussion necessarily assumes some familiarity with UNIX commands.

When mail arrives for netlib, it is piped through a process that strips off punctuation, through a sort process that removes duplicates, and into a C program that parses the request, translates the given library names into a search list, and invokes the system loader with the given routine names as external symbols to be resolved. A requested routine may require that many routines be assembled, to resolve all references (perhaps across libraries). The resulting loader map is edited into a list of file names to satisfy the request. These files, along with a time stamp and disclaimer, are then mailed back to the requester. A logfile records the time, return address, number of characters sent, and requested routine and library names. When the incoming mail includes actual names as well as an electronic return address, the correspondence is also logged.

The programs can tolerate minor syntax deviations, since we do get requests like "*Please send me the index for port. Thank you.*" from people who don't realize they are talking to a program. Users sometimes submit a single request on the subject line of the mail message, so a "Subject:" prefix is also allowed. One user even sent "*send index 4 port*" so "4" is a synonym for "for" and "from." (This is not such an unreasonable mistake, since the instructions are often given orally.) However, we make no attempt to accept arbitrary English input.

We chose this mode of interaction via electronic mail, keeping the intelligence local to the central depository, because mail is at present the only ubiquitous data communication service. We considered putting an interactive program at remote sites, communicating by mail with the depository. That would allow a better dialogue (‘‘Do you want that in single or double?’’) but would be difficult to write in the necessary portable way.

COMPARISON WITH OTHER SERVICES

The netlib service provides its users with features not previously available:

- There are no administrative channels to go through.

- Since no human processes the request, it is possible to get software at any time, even in the middle of the night.

- The most up-to-date version is always available.

- Individual routines or pieces of a package can be obtained instead of a whole collection. (One of the problems with receiving a large package of software is the volume of material. Often only a few routines are required from a package, yet the material is distributed as a whole collection and cannot easily be stripped off.)

On the other hand, netlib is simply a clearinghouse for contributed software and therefore subject to various disadvantages that have plagued such projects in the past. The only documents, example programs, and implementation tests are those supplied by the code author or other users. Also, there may be multiple codes for the same task and no help in choosing which is best. We have made an effort not to stock duplicate copies of machine constants, but in general we have left submitted codes untouched.

In summary, we are not aware of any comparable software distribution service in existence. (A number of systems have been started based on netlib such as, the Archive Server tool on SIMTEL20 at White Sands Missile Range and the benchmarking effort at the National Bureau of Standards, Gaithersburg Maryland.) Our system has a different focus from, say, the Quantum Chemistry Exchange, and a more convenient distribution mechanism. Furthermore, we are more selective than many personal computer ‘‘public bulletin board’’ systems: we do not allow users to put their own software automatically in the collection. (We wish to avoid having our computer confiscated as a result of someone posting a stolen charge number.)

The main cost of running this service is for communications. If it becomes necessary, we will require uucp users to call the hosts to pick up their return mail so that such costs are distributed fairly. At an average of a few requests per day, the traffic has been small enough to impose a negligible load on the host systems. Disk costs are controlled by discarding files that the host administrators are not themselves interested in keeping. The current collection occupies 57 megabytes. Most important, the human costs for maintaining the collection are modest and consist mainly of collecting software. We do not see how we could run such a widely accessible and low-overhead operation if we had to charge for the service—and we are not interested in doing so. (See, however, [1] for a description of the Toolchest electronic ordering system. One problem mentioned there is that users want to see demonstrations of software before purchase.)

HOPES FOR THE FUTURE

There are several areas where we would like to see netlib expand:

- *Editors.* The coverage of netlib obviously will tend to reflect the interests of the collectors, so we would welcome ‘‘associate editors’’ to augment the collection.

- *Depositories.* At present, there are just two distribution sites. Mail delays would be reduced if machines on other networks or in other countries were willing to also serve as depositories. (On the other hand, it is difficult even to keep two locations in sync!)

- *New collections*. The software that netlib uses to reply to mail is itself available from netlib, so it would be fairly easy for someone to, say, announce a service for searching a bibliography that he has collected.

Netlib cannot replace commercial software firms. We provide no consulting, make no claims for the quality of the software distributed, and do not even guarantee the service will continue. In compensation, the quick response time and the lack of bureaucratic, legal, and financial impediments encourage researchers to send us their codes. They know that their work can quickly be available to a wide audience for testing and use. We hope netlib will promote the use of modern numerical techniques in general scientific computing.

Acknowledgments. We express our gratitude to the many authors and editors who have permitted their codes to be freely distributed and to Gene Golub for his encouragement and help in starting this project. We thank Greg Astfalk of AT&T Laboratories, Princeton for creating the keyword index, Bill Coughran of AT&T Bell Labs, Murray Hill for editing the ode library and David Gay, also of Bell Labs, for editing lp/data and paranoia. Dave Presotto, Bell Labs, provided invaluable advice for dealing with network mail systems. The trick of editing a loader map is taken from the GAMS system at the National Bureau of Standards. Finally, the managements of our organizations deserve thanks for sponsoring this public service.

REFERENCES

1. Brooks, C.A. Experiences with electronic software distribution. *USENIX Association 1985 Summer Conference Proceedings.* Portland, Oregon, 1985.

Authors' Present Address: J. Dongarra, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439-4844 (electronic mail: anl-mcs!dongarra or dongarra@anl-mcs); Eric Grosse, AT&T Bell Laboratories, Murray Hill, NJ 07974 (electronic mail: research!ehg or ehg@btl.csnet).