Software Distribution Using Xnetlib

JACK DONGARRA
University of Tennessee and Oak Ridge National Laboratory

TOM ROWAN
Oak Ridge National Laboratory
and
REED WADE
University of Tennessee

Xnetlib is a new tool for software distribution. Whereas its predecessor netlib uses email as the user interface to its large collection of public-domain mathematical software, xnetlib uses an X Window interface and socket-based communication. Xnetlib makes it easy to search through a large distributed collection of software and to retrieve requested software in seconds.

Categories and Subject Descriptors: C.2.3 [**Computer-Communication Networks**]: Network Operations—*public networks*; D.2.2 [**Software Engineering**]: Tools and Techniques—*software libraries*; *user interfaces*; D.2.7 [**Software Engineering**]: Distribution and Maintenance—*documentation*; *portability*; G.1.0 [**Numerical Analysis**]: General—*numerical algorithms*; G.4 [**Mathematics of Computing**]: Mathematical Software—*portabılity*; H.3.0 [**Information Systems Applications**]: Communications Applications; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*search process*; *selection process*; H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*databank sharing*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*windowing systems*; K.6.3 [**Management of Computing and Information Systems**]: Software Management—*software development*; *software maintenance*; *software selection*

General Terms: Algorithms, Documentation, Management

Additional Key Words and Phrases: Netlib, software repositories

1. INTRODUCTION

Xnetlib is a new software distribution tool recently developed at the University of Tennessee and Oak Ridge National Laboratory. The goal in developing

This work was supported in part by ARPA and ARO under contract DAA 03-91-C-0047, and in part by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U.S. Department of Energy, under contract DE-AC05-84OR21400.

Authors' addresses: J. Dongarra and R. Wade, Department of Computer Science, University of Tennessee, Knoxville, TN 37996-1301; T. Rowan, Oak Ridge National Laboratory, Oak Ridge, TN 37831-6367.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.
© 1995 ACM 0098-3500/95/0300-0079$03.50

xnetlib was to provide Internet users faster and easier access to netlib's large collection of software, data, and documents. Unlike netlib, which uses email to process requests for software, xnetlib uses an X Window interface and socket-based communication between the user's machine and the xnetlib server[1] machine to process software requests. This enables users to search through a large distributed collection of software easily and to retrieve requested software in seconds.

2. BACKGROUND

Xnetlib's predecessor, netlib, grew from a need to have a quick and easy method for distributing small pieces of mathematical software. Netlib services began in 1985 at two sites, Argonne National Laboratory and AT & T Bell Laboratories, and distributed software from about 30 libraries. For additional information about netlib's operation and use, see the introductory paper by Dongarra and Grosse [1987].

One of the changes since netlib's introduction has been the transfer of netlib services from Argonne National Laboratory to Oak Ridge National Laboratory. Also, the availability on netlib of the netlib program itself has enabled many other sites to set up their own software repositories.[2] The netlib software collection has now grown to over 150 libraries. The number of software requests sent to netlib also has grown dramatically. The most heavily used netlib server, at Oak Ridge National Laboratory, processed over 147,000 requests in 1993.

Xnetlib use has grown even more rapidly. In 1993, its third year of operation, xnetlib processed over 233,000 requests.

3. OVERVIEW

We based xnetlib's design on the following requirements:

Speed. Retrieving software should take seconds, not minutes as typically required by email.

Usability. The user interface should make searching through a large collection of software and documents easy.

Organization. The system's repository should be a moderated collection, with up-to-date indexes and a database organized to facilitate searching and ease of retrieval. The repository may be distributed over several sites.

Record Keeping. The system should have the capability of logging requests so updates and corrections can be reported to users.

Security. The system should be secure from accidental or intentional misuse.

[1] Throughout this paper *server* refers to the process-handling software requests and not to the X display server.

[2] Send the message **send sites from netlib** to netlib@ornl.gov to receive a list of **netlib** sites.

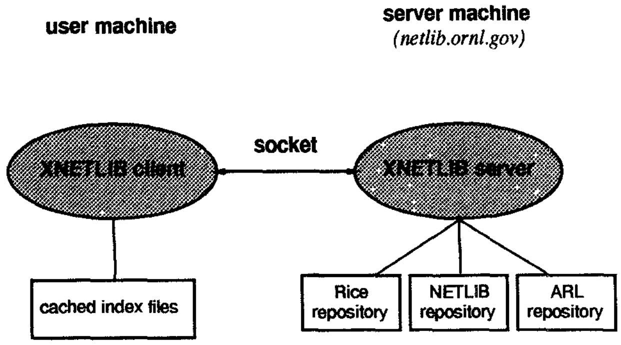


Fig. 1. Xnetlib configuration.

Portability. The system's implementation should be as portable as possible.

Accessibility. The system should be accessible to a large number of users.

There already exists a variety of tools for Internet browsing and for software and document retrieval [Krol 1992; Studt 1993]. Archie, ftp, gopher, mosaic, WAIS, and WorldWideWeb are among the most widely used.

In contrast to these other tools, xnetlib operates on a limited, moderated collection that contains the most up-to-date version of software packages. This approach of retaining control over the size and contents of xnetlib's repositories simplifies greatly any organization, record keeping, and security. Xnetlib uses socket-based communication to provide the desired speed and a client design based on X Windows for portability and usability. The accessibility requirement is satisfied by making the client software freely available over the Internet.

Figure 1 shows the basic configuration of the xnetlib system. The system consists of an xnetlib server process running on a machine at a repository site, an xnetlib X client process running on a user's local machine, and TCP/IP socket-based communication links between the client and server.

Xnetlib's server process runs continually at an xnetlib repository site, listening for incoming requests from xnetlib client processes. Typically, the xnetlib server runs on the same machine and accesses the same software repository as the netlib server. Upon receiving a request, the xnetlib server determines the nature of the request and responds by transferring the appropriate file from the repository to the xnetlib client process.

The xnetlib client running on the user's local machine provides an X Window interface to the xnetlib repository. It is programmed in C, using the Athena widget libraries. This interface makes searching through the software and document collection easy. For example, an xnetlib user can view the contents of any library simply by clicking a button. Other commands, such as key word searching or requesting software, also require just a few button clicks. To avoid unnecessary communication between the client and server, requested indexes are cached locally. Frequently requested information can therefore be quickly retrieved from local cache instead of repeatedly retrieved from the remote server. Other requests are passed to the server via sockets. Section 4 describes the use of the interface in more detail.

4. FEATURES

Xnetlib features and capabilities include

- —access to a distributed repository,
- —searching by a software libraries list,
- —searching by software classifications,
- —searching by key word,
- —software and document retrieval,
- —access to a computer performance database,
- —access to a conferences database,
- —access to the NA-NET White-pages, and
- —online help.

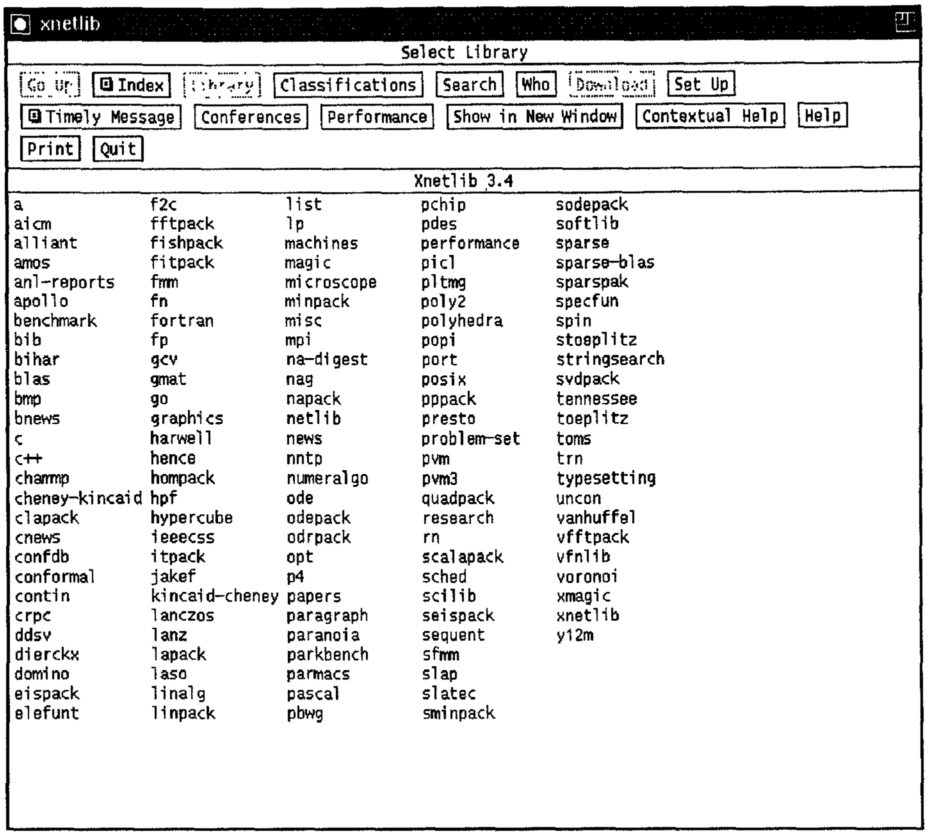
Many Internet sites have sizable collections of documents or software. It is both unnecessary and undesirable to require that these collections reside at a single site. Xnetlib gives users access to a distributed repository of software and documents by establishing socket-based links with the xnetlib repository sites. Users have access to any or all of these repositories through a single interface.

Xnetlib users control which sites are linked into the distributed repository using the **set up** button. Current repository sites include netlib.ornl.gov, spark.brl.mil, and softlib.rice.edu. The **timely message** button displays news about individual repository sites, and the **index** button displays their general indexes.

The **library** button shows the libraries that are available through xnetlib. The collection includes major packages such as LAPACK, LINPACK, EISPACK, algorithms published in the *ACM Transactions on Mathematical Software*, and a large variety of less well-established packages. Figure 2 shows a unified list of software and documents available at the repositories of Oak Ridge National Laboratory, Rice University, and the U.S. Army Research Laboratory.

Clicking on a library name lists the contents of that library. For example, clicking on **lapack** displays a partial listing of LAPACK's contpnts (Figure 3). The complete contents list of LAPACK is too large to fit in the window, but the provided scrollbar allows the user to scroll through the rest of the list. Clicking on the box adjacent to a routine name selects that routine for future downloading. In Figure 3 the user has selected **sgetrf** and **sgetrs** from the LAPACK library.

The **classifications** feature allows a user to narrow a search. The classification of the xnetlib software libraries is based on the highest level of the GAMS [Boisvert et al. 1985] classification system, augmented to include classifications other than mathematical software. Selecting **linear algebra**



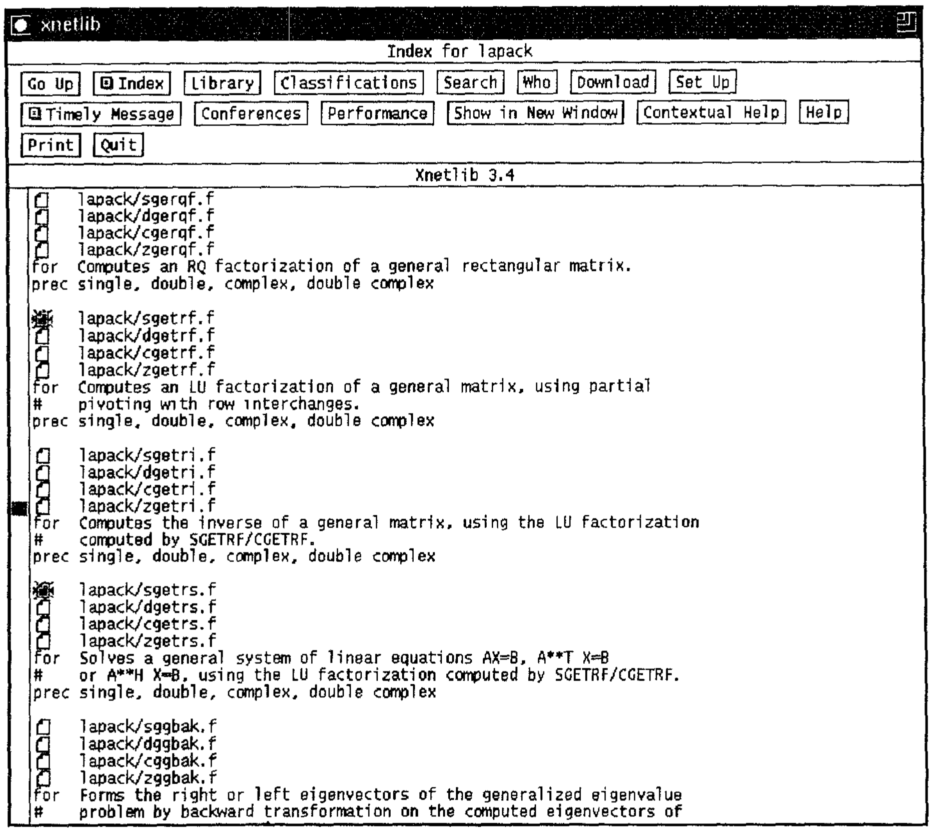
```
a              f2c            list        pchip        sodepack
aicm           fftpack        lp          pdes         softlib
alliant        fishpack       machines    performance  sparse
amos           fitpack        magic       picl         sparse-blas
anl-reports    fmm            microscope  pltmg        sparspak
apollo         fn             minpack     poly2        specfun
benchmark      fortran        misc        polyhedra    spin
bib            fp             mpi         popi         stoeplitz
bihar          gcv            na-digest   port         stringsearch
blas           gmat           nag         posix        svdpack
bmp            go             napack      pppack       tennessee
bnews          graphics       netlib      presto       toeplitz
c              harwell        news        problem-set  toms
c++            hence          nntp        pvm          trn
chammp         hompack        numeralgo   pvm3         typesetting
cheney-kincaid hpf            ode         quadpack     uncon
clapack        hypercube      odepack     research     vanhuffel
cnews          ieeecss        odrpack     rn           vfftpack
confdb         itpack         opt         scalapack    vfnlib
conformal      jakef          p4          sched        voronoi
contin         kincaid-cheney papers      scilib       xmagic
crpc           lanczos        paragraph   seispack     xnetlib
ddsv           lanz           paranoia    sequent      y12m
dierckx        lapack         parkbench   sfmm
domino         laso           parmacs     slap
eispack        linalg         pascal      slatec
elefunt        linpack        pbwg        sminpack
```

Fig. 2. Library menu.

causes the names of the libraries with linear algebra software to be displayed (Figure 4).

A user may also wish to **search** by key word instead of viewing the contents of a particular library. In this mode, index files containing lists of files and their descriptions are searched by a key word string the user provides. The key word search can be a search on the intersection or union of the words in a search string, a literal search for an exact string (with or without case sensitivity), or a fuzzy search based on the latent semantic indexing technique [Deerwester et al. 1990]. Latent semantic indexing uses statistical analysis to find useful matches that may not be uncovered by other types of searches. In contrast, the fuzzy-search capability in WAIS is based on a heuristic rather than on a statistical approach.

Clicking on the **download** button causes xnetlib to display a list of selected software and documents. The **download path** button allows a user to change the directory to which files will be downloaded. The **dependency checking** button is a modal switch. If **dependency checking** is off, xnetlib will send only the selected routines. If it is on, xnetlib will send the selected routines



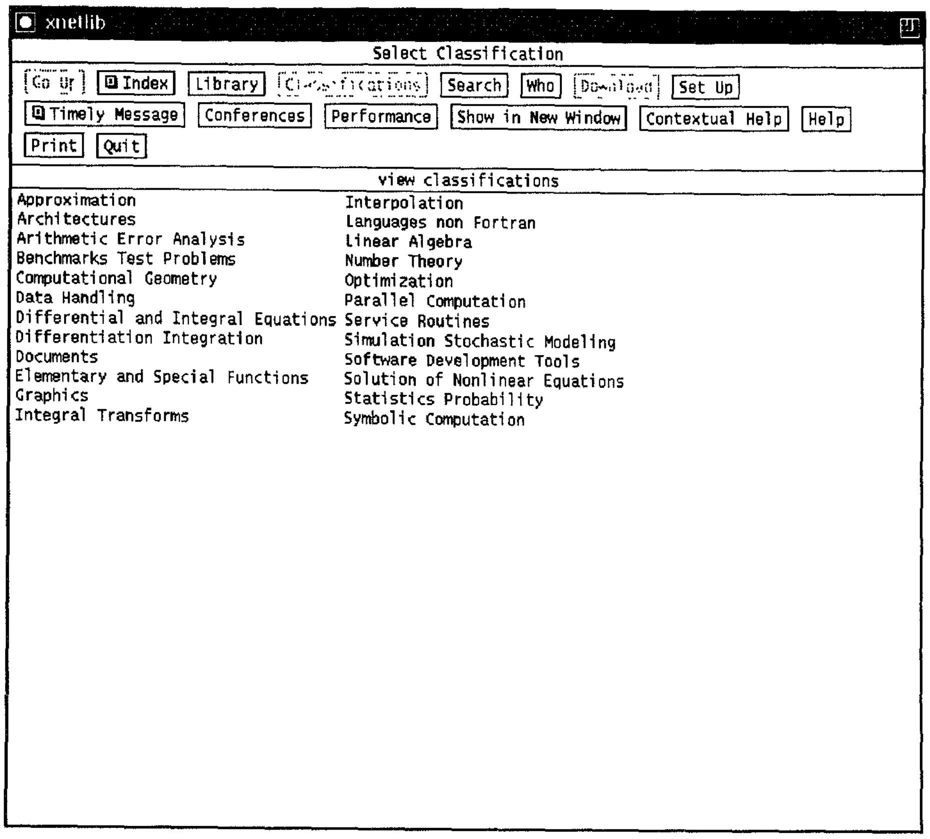
```
     lapack/sgerqf.f
     lapack/dgerqf.f
     lapack/cgerqf.f
     lapack/zgerqf.f
for  Computes an RQ factorization of a general rectangular matrix.
prec single, double, complex, double complex

     lapack/sgetrf.f
     lapack/dgetrf.f
     lapack/cgetrf.f
     lapack/zgetrf.f
for  Computes an LU factorization of a general matrix, using partial
#    pivoting with row interchanges.
prec single, double, complex, double complex

     lapack/sgetri.f
     lapack/dgetri.f
     lapack/cgetri.f
     lapack/zgetri.f
for  Computes the inverse of a general matrix, using the LU factorization
#    computed by SGETRF/CGETRF.
prec single, double, complex, double complex

     lapack/sgetrs.f
     lapack/dgetrs.f
     lapack/cgetrs.f
     lapack/zgetrs.f
for  Solves a general system of linear equations AX=B, A**T X=B
#    or A**H X=B, using the LU factorization computed by SGETRF/CGETRF.
prec single, double, complex, double complex

     lapack/sggbak.f
     lapack/dggbak.f
     lapack/cggbak.f
     lapack/zggbak.f
for  Forms the right or left eigenvectors of the generalized eigenvalue
#    problem by backward transformation on the computed eigenvectors of
```

Fig. 3. LAPACK index.

and any routines that they call. If the user is satisfied with the selection list and the target directory, the user should click on **get files now** to initiate the transfer. The displayed "gas gauge" shows the progress of file transfers. Figure 5 shows the downloading of selections made from the LAPACK, SOFTLIB, and RN libraries. These three libraries reside in repositories at separate sites. The selected files will appear in the specified directory, usually within a few seconds.

The **who** button allows a user to search the NA-NET White-pages [Dongarra and Rosener 1991; Kent 1988], a database containing information about individuals interested in numerical analysis and other disciplines. For simple searches the user need only enter an individual's last name. The **modify search** feature can be used to specify which fields the search applies to, and the **modify listing** feature can be used to control the form of the output.

The **conferences** database contains announcements of conferences and meetings. Users can search announcements by title, dates, location, or description. Users can also submit conference announcements to the database directly through xnetlib.



```
Approximation                         Interpolation
Architectures                         Languages non Fortran
Arithmetic Error Analysis             Linear Algebra
Benchmarks Test Problems              Number Theory
Computational Geometry                Optimization
Data Handling                         Parallel Computation
Differential and Integral Equations   Service Routines
Differentiation Integration           Simulation Stochastic Modeling
Documents                             Software Development Tools
Elementary and Special Functions      Solution of Nonlinear Equations
Graphics                              Statistics Probability
Integral Transforms                   Symbolic Computation
```

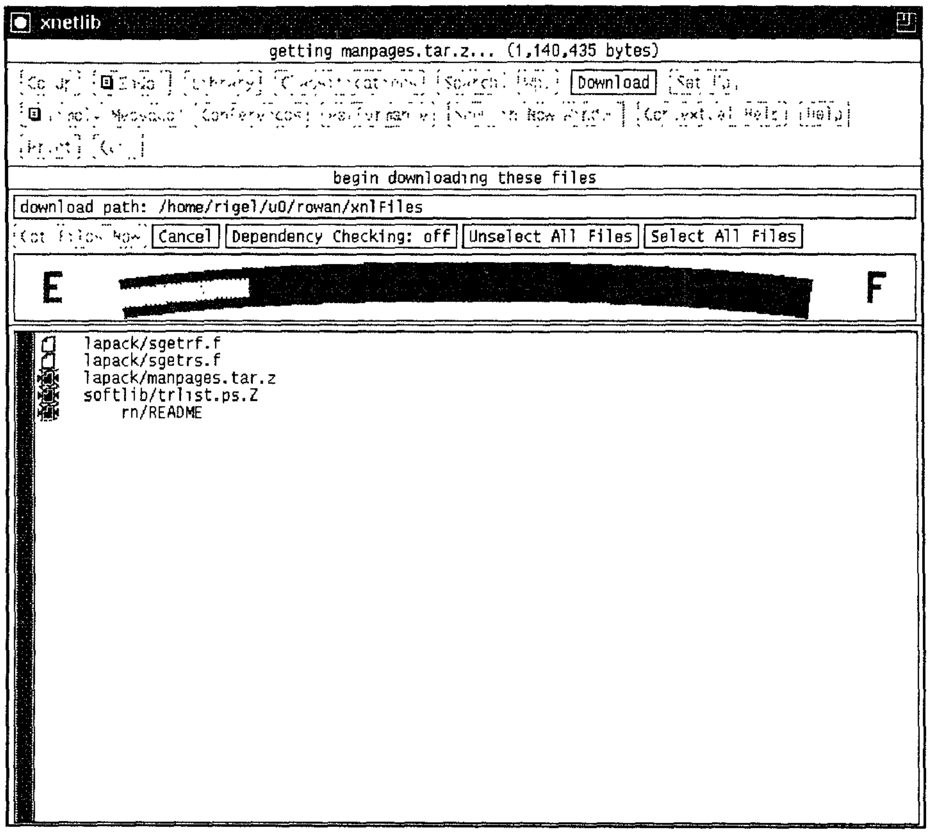
Fig. 4. Classifications menu.

The **performance** database [Berry et al. 1994] is a repository of computer performance data for a wide variety of machines. Users can make selections from a menu of computer vendors and from a menu of benchmarks to generate a formatted display of benchmark results.

5. GETTING STARTED

Xnetlib requires release 4 (or later) of X11, and the Athena widget libraries as supplied by MIT. The executable for the xnetlib client requires approximately 800 KB on a Sun SPARCstation2. The locally cached indexes can occupy up to 1 MB. One common customization at multiuser sites is to have a single cache of index files so that indexing information can be shared by all local users.

The first step in installing xnetlib is to obtain the source code for the xnetlib client. The software for the xnetlib client is itself available from netlib, so a simple way to obtain the source is to send the message **send xnetlib.shar from xnetlib** to netlib@ornl.gov. Netlib will respond by sending a shar file containing all necessary source code and documentation. Xnetlib is also available by anonymous ftp from netlib2.cs.utk.edu in the xnetlib directory. The



getting manpages.tar.z... (1,140,435 bytes)

begin downloading these files

download path: /home/rigel/u0/rowan/xnlFiles

Dependency Checking: off

```
lapack/sgetrf.f
lapack/sgetrs.f
lapack/manpages.tar.z
softlib/trlist.ps.Z
    rn/README
```

Fig. 5. Downloading the selected routines.

xnetlib distribution includes an Imakefile, so installation is normally trivial if the X Window System has been configured properly on the client machine.

6. PLANS

Plans are already under way to expand xnetlib. One major addition will be the capability of remote execution. Many useful utility programs are large, making distribution tedious, or are more expensive to build than to execute. In such cases, allowing remote execution may be a better use of resources from both the distributor's and user's point of views. Toolpack [Pollicini 1989], a large collection of Fortran software tools, and f2c [Feldman et al. 1990], a Fortran-to-C translator, are logical candidates for this remote execution service. Users could, instead of downloading, installing, and executing these programs, submit suitable input and have a machine at a remote server site execute the programs and return the output.

Another proposed capability will allow users to add their own servers to the collection of servers globally available through xnetlib. Any xnetlib user could then access this contributed software by adding the appropriate server to his or her active server list. This feature will greatly expand the amount of software available through xnetlib.

We also plan to expand the scope of the xnetlib system. In the future, xnetlib's distributed repository will include more nonmathematical software and more reports. Additional software and document collections will be linked into xnetlib's existing collection.

Starting with LAPACK, xnetlib began distributing entire libraries. This service will be extended to other libraries.

Xnetlib already provides fast and easy access to a large collection of mathematical software. In the future xnetlib will provide greatly expanded capabilities to satisfy more of the computational needs of users.

7. SUMMARY

Software distributed by netlib comes with the disclaimer that "anything free comes with no guarantee." In contrast to commercial vendors like NAG and IMSL, netlib offers no support beyond whatever documentation contributing authors choose to provide with their code. These caveats also apply to xnetlib.

On the other hand, both netlib and xnetlib provide free, easy access to a large body of high-quality code, and the phenomenal growth of netlib and xnetlib attests to the value of this service. We hope that xnetlib, by making high-quality code even more accessible, will encourage software developers to make their codes freely available and will make good programming easier for the scientific computing community.

ACKNOWLEDGMENTS

Xnetlib is the product of the efforts of many people. Ken Bateman, David Bolt, Shirley Browne, Jennifer Finger, Tracy Gangwer, Stan Green, Brian LaRose, Todd Letsche, Sharon Lewis, Keith Moore, Andrew Pearson, Jon Richardson, Bill Rosener, and Andrea Van Hull all made valuable contributions to the design, development, and testing of various versions of xnetlib. We also thank the many users of early versions of xnetlib for their constructive comments and for their patience.

REFERENCES

BERRY, M., DONGARRA, J., AND LAROSE, B. 1994. PDS: A performance database server. *Sci. Comput.* To be published.

BOISVERT, R. F., HOWE, S. E., AND KAHANER, D. K. 1985. GAMS—A framework for the management of scientific software. *ACM Trans. Math. Softw. 11*, 4 (Dec.), 313–355.

DEERWESTER, S., DUMAIS, S., FURNAS, G., LANDAUER, T., AND HARSHAMN, R. 1990. Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci. 41*, 6 (Sept.), 391–407.

DONGARRA, J., AND GROSSE, E. 1987. Distribution of mathematical software via electronic mail. *Commun. ACM 30*, 5 (May), 403–407.

DONGARRA, J., AND ROSENER, B. 1991. NA-NET numerical analysis net. Tech. Rep. ORNL/TM-11986, Oak Ridge National Laboratory, Oak Ridge, Tenn., Dec.

FELDMAN, S., GAY, D., MAIMONE, M., AND SCHRYER, N. 1990. A Fortran-to-C converter. Comput. Sci. Tech. Rep. 149, AT & T Bell Laboratories, Murray Hill, N.J., May.

KENT, M. 1988. The numerical analysis net. Tech. Rep. 85, Eidgenössische Technische Hochschle (ETH), Zurich, Jan.

KROL, E. 1992. *The Whole Internet User's Guide and Catalog*. O'Reilly & Associates, Inc., Newton, Mass.

POLLICINI, A. A., (ED.). 1989. *Using Toolpack Software Tools*. Kluwer Academic, Deventer, The Netherlands.

STUDT, T. 1993. Access information anywhere on the internet. *Res. Devel. Mag.* (Aug), 71.

Received April 1993; revised January 1994; accepted January 1994