ANL-80-105 ANL-80-105

LINPACK WORKING NOTE #13: IMPLEMENTATION GUIDE FOR LINPACK

by

J. J. Dongarra and C. B. Moler

ARGONNE NATIONAL LABORATORY

ARGONNE NATIONAL LABORATORY, ARGONNE, ILLINOIS

Prepared for the U. S. DEPARTMENT OF ENERGY
under Contract W-31-109-Eng-38

Printed in the United States of America
Available from
National Technical Information Service
U. S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

NTIS price codes
Printed copy: A02
Microfiche copy: A01

Distribution Category:
Mathematics and Computers
(UC-32)

ANL-80-105

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

LINPACK WORKING NOTE #13:
IMPLEMENTATION GUIDE FOR LINPACK

by

J. J. Dongarra and C. B. Moler

October 1980

TABLE OF CONTENTS

LINPACK WORKING NOTE #13

Implementation Guide for LINPACK

by

J. J. Dongarra and C. B. Moler

ABSTRACT

This working note is intended to help a person install and test LINPACK. The instructions are designed for a person whose responsibility is the maintenance of a mathematical software library. We assume the reader has a working knowledge of the system job control language and some experience with numerical calculations. The installation process involves reading a magnetic tape, creating a library from the Fortran source, then running and examining the output of the test aids.

1. Introduction

This working note is intended to help a person install and test LINPACK. The instructions are designed for a person whose responsibility is the maintenance of a mathematical software library. We assume the reader has a working knowledge of the system job control language and some experience with numerical calculations. The installation process involves reading a magnetic tape, creating a library from the Fortran source, then running and examining the output of the test aids.

LINPACK may be obtained from either:

National Energy Software Center
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439
(Phone: 312-972-7250)
Cost: Determined for various categories of requestors by NESC policy.

or

International Mathematics and Statistical Libraries, Inc.
Sixth Floor, GNB Building
7500 Bellaire Blvd.
Houston, TX 77036
(Phone: 713-772-1927
Cost: $75.00 (Tape included)

The complete documentation for the package can be found in:

J. J. Dongarra, J. R. Bunch, C. B. Moler, G. W. Stewart, LINPACK Users' Guide, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.

2. The Format

LINPACK is distributed in the form of a magnetic tape which contains the Fortran source for LINPACK, the Basic Linear Algebra Subprograms (BLAS) needed by LINPACK, the testing aids and the initial program comments.

The tape contains 54 files. The usual format is 9 track EBCDIC, unlabeled, 80 characters per logical record, 40 logical records per block. A 7 track tape is available upon special request.

Card counts for the 54 files are given in section 11.

3. Overview of Tape Contents

There are four versions of each routine in LINPACK: real, double precision, complex and complex*16. The first three versions (real, double and complex) are written in standard Fortran and are completely portable. The complex*16 version is provided for those compilers which allow this data type.

4. LINPACK

The first four files on the tape are:

File 1	Real	LINPACK,	40	subroutines
2	Double precision	" ,	40	"
3	Complex	" ,	48	"
4	Complex*16	" ,	48	"

These four files contain the routines we call LINPACK. The remaining 50 files contain auxiliary material.

5. Support Routines

LINPACK employs the Basic Linear Algebra Subprograms (BLAS) to carry out basic vector operations. This package was designed by C. Lawson, R. Hanson, D. Kincaid and F. Krogh. Versions of the BLAS used in LINPACK are included in files 5-8 on the tape. The BLAS provided are written in portable Fortran and were designed to be as efficient as possible under these conditions. Assembly language versions of the BLAS for many machines will be available Fall 1979 from ACM Algorithms Distribution Service, c/o International Mathematical and Statistical Libraries, Inc., and we recommend they be used. The assembly language versions may have lower overhead associated with them, and thereby increase the overall efficiency of LINPACK.

In addition to the BLAS in files 5-8 there are auxiliary routines which are used by the package or the testing routines. Subprograms CSROT, ZDROT, and DABS1 are used by LINPACK in addition to the BLAS. Functions SMACH, DMACH, CMACH, and ZMACH are used by the test drivers to calculate machine dependent parameters in the testing programs. These functions are used in generating test matrices only.

File 5	Real ,	11	support	subprograms
6	Double precision,	11	"	"
7	Complex ,	14	"	"
8	Complex*16 ,	15	"	"

To initially install, say, the real version of LINPACK, you will need File 1 and File 5 from the tape. Each support file of a given precision contains all the routines needed to install the LINPACK routine of that same version.

The machine parameter functions SMACH, DMACH, CMACH, and ZMACH should be removed from the library after testing has been completed.

6. Test Aids

There is a collection of testing aids which should help in checking out LINPACK after it has been installed.

Files 9-18	Test	drivers	for	single	precision	routines
19-28	"	"	"	double	"	"
29-39	"	"	"	complex		"
40-50	"	"	"	complex*16		"

7. Initial Comments

In order to give the user some sort of online documentation for LINPACK, we have prepared four files which contain initial program comments from the four versions of the code. These comments describe, in general terms, what the routines do and the meaning of the calling sequence parameters. This is <u>not</u> intended to be a complete discussion of what the routines can do or how to use every facet of them. For this information, see the <u>LINPACK Users' Guide</u>.

File	51	Single	precision	routines	initial	comments
	52	Double	"	"	"	"
	53	Complex		"	"	"
	54	Complex*16		"	"	"

8. Creating a Library

For the creation of the library, first you must decide what version or versions of the package you intend to install at your site.

To install	single precision you will need file	1 and	5
	double	2	6
	complex	3	7
	complex*16	4	8

We suggest you compile and create a load module library from the appropriate files.

The method of creating a library will vary from machine to machine and from system to system. Outlined below are two methods which may help you in creating the library.

Some systems allow one to take many Fortran source routines, compile them at one time and send the output directly to the linkage editor for library creation. At some IBM sites there is a user written utility called ADDLINK which will do this in one step. ADDLINK essentially calls the Fortran compiler, inserts the appropriate linkage editor cards into the generated object decks and invokes the link editor. This utility is not available on all machines, but is quite handy.

If your system does not have the ADDLINK capability, you may have to take each subroutine, compile it and have it processed by the linkage editor on an individual basis. You will need to separate the routines in order to do this. We have not explicitly provided separator cards, but each subprogram starts with sequence number "00000010" in columns 73-80 and ends with "   END" in columns 1-9.

Separator cards can be inserted between each subroutine in files 1-4 quite easily. The name of the subroutine can be found in columns 18 through 22; the subroutine names are always 5 characters long. The names of the subprograms in files 5-8 are harder to extract since both functions and subroutines are provided there.

Once separator cards have been inserted, a utility like IEBUPDTE on IBM machines can be used to create a library of source routines. This is called a partitioned dataset on IBM machines. Once the library of source is created, each member can then be compiled and link edited separately to create the load library. A complete listing of the subprograms in files 1-8 is given in section 10.

9. Testing

After the library is created the testing aids should be run. The testing aids are in files 9-50 of the tape and are organized as follows:

File	9	SG	File	19	DG	File	29	CG	File	40	ZG
	10	SP		20	DP		30	CP		41	ZP
	11	SS		21	DS		31	CS		42	ZS
	12	ST		22	DT		32	CH		43	ZH
	13	SGT		23	DGT		33	CT		44	ZT
	14	SCH		24	DCH		34	CGT		45	ZGT
	15	SUD		25	DUD		35	CCH		46	ZCH
	16	SEX		26	DEX		36	CUD		47	ZUD
	17	SQR		27	DQR		37	CEX		48	ZEX
	18	SSV		28	DSV		38	CQR		49	ZQR
							39	CSV		50	ZSV

Each of the above test drivers test the following routines:

```
*G
     *GECO,FA,SL,DI
     *GBCO,FA,SL,DI
```

```
*P
     *POCO,FA,SL,DI
     *PPCO,FA,SL,DI
     *PBCO,FA,SL,DI
*S
     *SICO,FA,SL,DI
     *SPCO,FA,SL,DI
*H                                 where * is S, D, C, or Z
     *HICO,FA,SL,DI
     *HPCO,FA,SL,DI
*T
     *TRCO,SL,DI
*GT
     *GTSL
     *PTSL
*CH
     *CHDC
*EX
     *CHEX
*UD
     *CHUD,DD
*QR
     *QRDC,SL
*SV
     *SVDC
```

Each test file contains a main program and several subroutines which generate test matrices, call LINPACK subroutines and their support routines from the library, calculate measures of performance and produce output. The output unit number is set to 6 by a single assignment near the beginning of the main program. There are no READ statements. An additional system dependent support routine called TRAPS must also be supplied.

The routines in LINPACK are designed not to overflow, cause a divide by zero, or allow underflows when they would alter the results. Underflows which are non-destructive are allowed. Some test cases are generated that involve arithmetic near the underflow and overflow limits of the machine. In order to generate these test cases, underflows are necessary. On some systems this may be a problem. It is necessary to provide a subroutine TRAPS which sets the number of arithmetic exceptions permitted. TRAPS is taken from the WATFIV system and has the following arguments:

```
TRAPS(I1,I2,I3,I4,I5)
```

```
where  I1 is the number of fixed point overflows permitted.
       I2 is the number of floating point overflows permitted.
       I3 is the number of floating point underflows permitted.
       I4 is the number of fixed point divide by zero permitted.
       I5 is the number of floating point divide by zero permitted.
```

S and D test routines have

```
          CALL TRAPS(0,0,5001,0,0)
```

and C and Z test routines have

```
          CALL TRAPS(0,0,5001,0,3).
```

The subroutine TRAPS on IBM systems would include:

```
          SUBROUTINE TRAPS(I1,I2,I3,I4,I5)
          CALL ERRSET(208,I3,-1,0)
          CALL ERRSET(209,I5,-1,0)
```

Our routine CMACH used by the complex test routines, generates a large number for the machine on which it is running. It does this by taking the reciprocal of a small number. This may result in a division by zero if the compiler is not performing complex division correctly. Consequently the complex tests may generate one or more divisions by zero.

All the routines in the package have passed through the Bell Laboratories' PFORT Checker, which verifies the codes conform to a Fortran ANSI subset. The routines have also been run on the new WATFIV compiler, but will not run on the old WATFIV because of the way dummy arrays are declared.

Each of the different test routines produces output which summarizes how well the routines handled the problem. Figures of merit in the form of ratios are printed out and the suspicious ones are flagged.

For test paths SG, SP, SS, ST, and SGT (and their double precision and complex counterparts), the measures are:

$\frac{\text{COND}}{\text{ACTUAL}}$	is the ratio of the estimate of the condition number to the actual condition number, which is calculated explicitly. If this number is > 1, then something is wrong.
$\frac{\text{ACTUAL}}{\text{COND}}$	is the reciprocal of the previous measure. This is almost always < 10, but there is no known upper bound on its possible value.
$\frac{\text{ERROR}}{\text{E*COND*X}}$	is $\frac{\|x-\bar{x}\|}{\varepsilon \, \text{COND} \|x\|}$, where x is the true solution, $\bar{x}$ is the computed solution, ε is the machine rounding unit, and COND is the estimate of the condition number of the matrix.
$\frac{\text{ERROR-T}}{\text{E*COND*X}}$	is the measure as above for solving $A^T x = b$.

$\frac{\text{RESID}}{\text{E*A*X}}$ is $\frac{\|A\bar{x}-b\|}{\varepsilon \|A\| \|\bar{x}\|}$, where A is the original matrix, $\bar{x}$ is the computed solution, b is the right hand side, and ε is the machine rounding unit.

$\frac{\text{RESID-T}}{\text{E*A*X}}$ is the measure as above for solving $A^T x = b$.

$\frac{\text{A-LU}}{\text{E*A}}$ is $\frac{\|A-LU\|}{\varepsilon \|A\|}$, where A is the original matrix, L and U are produced by the decomposition, and ε is the machine rounding unit.

$\frac{\text{A*AI-I}}{\text{E*COND}}$ is $\frac{\|AA^{-1}-I\|}{\varepsilon\, \text{COND}}$, where A is the original matrix, A^{-1} is the computed inverse of A, I is the identity matrix, ε is the machine rounding unit, and COND is the estimate of the condition of A.

If any of the last six ratios are > N, the order of the matrix, then they are considered suspicious and are flagged as such.

For the test path SSV, the measures are:

U*SIGMA*VH is $\frac{\|U\Sigma V^H-X\|}{\varepsilon \|X\|}$, where X is the original matrix, U, Σ, and V are produced by the decomposition, and ε is the machine rounding unit.

UHU is $\frac{\|U^H U-I\|}{\varepsilon}$, where U is a matrix produced by the decomposition, I is the identity, and ε is the machine rounding unit.

VHV is $\frac{\|V^H V-I\|}{\varepsilon}$, where V is a matrix produced by the decomposition, I is the identity matrix, and ε is the machine rounding unit.

If any of these ratios are > 100, then they are considered suspicious and are flagged as such.

For the test path, SQR, the measures are:

FORWARD MULTIPLICATION is $\frac{\|R-Q^T X\|}{\varepsilon \|R\|}$, where R and Q are produced by the decomposition, X is the original matrix, and ε is the machine rounding unit.

BACKWARD MULTIPLICATION is $\frac{\|X-QR\|}{\varepsilon \|X\|}$, where X is the original matrix, Q and R are produced by the decomposition, and ε is the machine rounding unit.

If any of these ratios are > 100, then they are considered suspicious and are flagged as such.

The three test paths SCH, SEX, SUD produce similar ratios, but no attempt is made to flag particular values as suspicious.

The machine rounding unit, which is the approximate distance from 1.0 to the next floating point number, is calculated by subroutines SMACH, DMACH, CMACH, and ZMACH. These routines also compute a small and large number for the machine on which they are running. The small number is roughly the underflow limit divided by the machine rounding unit and the large number is the reciprocal of that.

These tests are fairly stringent and so the appearance of a few "slightly" suspicious ratios should not be cause for alarm. Improper installation or errors in compilers and operating systems cause the test runs to terminate abnormally or produce many very large ratios.

Each test driver produces at most 600 lines of output. The time to run each test, including both compilation and execution, is given by the table below.

IBM 370/195, CDC 7600	15 seconds
IBM 370/168, CDC 6600	1 minute
IBM 360/75, UNIVAC 1110, Honeywell 6020	4 minutes

These times are only rough estimates and have not been generated from actual runs but from extrapolations.

10. List of Subprograms

LINPACK Subroutines

File 1	File 2	File 3	File 4
SGECO	DGECO	CGECO	ZGECO
SGEFA	DGEFA	CGEFA	ZGEFA
SGESL	DGESL	CGESL	ZGESL
SGEDI	DGEDI	CGEDI	ZGEDI
SGBCO	DGBCO	CGBCO	ZGBCO
SGBFA	DGBFA	CGBFA	ZGBFA
SGBSL	DGBSL	CGBSL	ZGBSL
SGBDI	DGBDI	CGBDI	ZGBDI
SPOCO	DPOCO	CPOCO	ZPOCO
SPOFA	DPOFA	CPOFA	ZPOFA
SPOSL	DPOSL	CPOSL	ZPOSL
SPODI	DPODI	CPODI	ZPODI
SPPCO	DPPCO	CPPCO	ZPPCO
SPPFA	DPPFA	CPPFA	ZPPFA
SPPSL	DPPSL	CPPSL	ZPPSL
SPPDI	DPPDI	CPPDI	ZPPDI
SPBCO	DPBCO	CPBCO	ZPBCO
SPBFA	DPBFA	CPBFA	ZPBFA
SPBSL	DPBSL	CPBSL	ZPBSL
SPBDI	DPBDI	CPBDI	ZPBDI
SSICO	DSICO	CSICO	ZSICO
SSIFA	DSIFA	CSIFA	ZSIFA
SSISL	DSISL	CSISL	ZSISL
SSIDI	DSIDI	CSIDI	ZSIDI
SSPCO	DSPCO	CSPCO	ZSPCO
SSPFA	DSPFA	CSPFA	ZSPFA
SSPSL	DSPSL	CSPSL	ZSPSL
SSPDI	DSPDI	CSPDI	ZSPDI
STRCO	DTRCO	CHICO	ZHICO
STRSL	DTRSL	CHIFA	ZHIFA
STRDI	DTRDI	CHISL	ZHISL
SGTSL	DGTSL	CHIDI	ZHIDI
SPTSL	DPTSL	CHPCO	ZHPCO
SCHDC	DCHDC	CHPFA	ZHPFA
SCHUD	DCHUD	CHPSL	ZHPSL
SCHDD	DCHDD	CHPDI	ZHPDI
SCHEX	DCHEX	CTRCO	ZTRCO
SQRDC	DQRDC	CTRSL	ZTRSL
SQRSL	DQRSL	CTRDI	ZTRDI
SSVDC	DSVDC	CGTSL	ZGTSL
		CPTSL	ZPTSL
		CCHDC	ZCHDC
		CCHUD	ZCHUD
		CCHDD	ZCHDD
		CCHEX	ZCHEX
		CQRDC	ZQRDC
		CQRSL	ZQRSL
		CSVDC	ZSVDC

Support Subprograms

File 5	File 6	File 7	File 8
ISAMAX	DASUM	CAXPY	DABS1
SASUM	DAXPY	CCOPY	DROTG
SAXPY	DCOPY	CDOTC	DZASUM
SCOPY	DDOT	CDOTU	DZNRM2
SDOT	DMACH	CMACH	ZDROT
SMACH	DNRM2	CROTG	IZAMAX
SNRM2	DROT	CSCAL	ZAXPY
SROT	DROTG	CSROT	ZCOPY
SROTG	DSCAL	CSSCAL	ZDOTC
SSCAL	DSWAP	CSWAP	ZDOTU
SSWAP	IDAMAX	ICAMAX	ZDSCAL
		SCASUM	ZMACH
		SCNRM2	ZROTG
		SROTG	ZSCAL
			ZSWAP

Note: Routine SROTG appears in both file 5 and file 7 and routine DROTG appears in both file 6 and file 8.

Alphabetical List of Subprograms in Files 1 through 8

CAXPY	CSIFA	DPTSL	SNRM2	ZGBSL
CCHDC	CSISL	DQRDC	SPBCO	ZGECO
CCHDD	CSPCO	DQRSL	SPBDI	ZGEDI
CCHEX	CSPDI	DROT	SPBFA	ZGEFA
CCHUD	CSPFA	DROTG	SPBSL	ZGESL
CCOPY	CSPSL	DSCAL	SPOCO	ZGTSL
CDOTC	CSROT	DSICO	SPODI	ZHICO
CDOTU	CSSCAL	DSIDI	SPOFA	ZHIDI
CGBCO	CSVDC	DSIFA	SPOSL	ZHIFA
CGBDI	CSWAP	DSISL	SPPCO	ZHISL
CGBFA	CTRCO	DSPCO	SPPDI	ZHPCO
CGBSL	CTRDI	DSPDI	SPPFA	ZHPDI
CGECO	CTRSL	DSPFA	SPPSL	ZHPFA
CGEDI	DABS1	DSPSL	SPTSL	ZHPSL
CGEFA	DASUM	DSVDC	SQRDC	ZMACH
CGESL	DAXPY	DSWAP	SQRSL	ZPBCO
CGTSL	DCHDC	DTRCO	SROT	ZPBDI
CHICO	DCHDD	DTRDI	SROTG	ZPBFA
CHIDI	DCHEX	DTRSL	SSCAL	ZPBSL
CHIFA	DCHUD	DZASUM	SSICO	ZPOCO
CHISL	DCOPY	DZNRM2	SSIDI	ZPODI
CHPCO	DDOT	ICAMAX	SSIFA	ZPOFA
CHPDI	DGBCO	IDAMAX	SSISL	ZPOSL
CHPFA	DGBDI	ISAMAX	SSPCO	ZPPCO
CHPSL	DGBFA	IZAMAX	SSPDI	ZPPDI
CMACH	DGBSL	SASUM	SSPFA	ZPPFA
CPBCO	DGECO	SAXPY	SSPSL	ZPPSL
CPBDI	DGEDI	SCASUM	SSVDC	ZPTSL
CPBFA	DGEFA	SCHDC	SSWAP	ZQRDC
CPBSL	DGESL	SCHDD	STRCO	ZQRSL
CPOCO	DGTSL	SCHEX	STRDI	ZROTG
CPODI	DMACH	SCHUD	STRSL	ZSCAL
CPOFA	DNRM2	SCNRM2	ZAXPY	ZSICO
CPOSL	DPBCO	SCOPY	ZCHDC	ZSIDI
CPPCO	DPBDI	SDOT	ZCHDD	ZSIFA
CPPDI	DPBFA	SGBCO	ZCHEX	ZSISL
CPPFA	DPBSL	SGBDI	ZCHUD	ZSPCO
CPPSL	DPOCO	SGBFA	ZCOPY	ZSPDI
CPTSL	DPODI	SGBSL	ZDOTC	ZSPFA
CQRDC	DPOFA	SGECO	ZDOTU	ZSPSL
CQRSL	DPOSL	SGEDI	ZDROT	ZSVDC
CROTG	DPPCO	SGEFA	ZDSCAL	ZSWAP
CSCAL	DPPDI	SGESL	ZGBCO	ZTRCO
CSICO	DPPFA	SGTSL	ZGBDI	ZTRDI
CSIDI	DPPSL	SMACH	ZGBFA	ZTRSL

11. Card Counts for Files 1-54

File 1	contains	6,861	cards
2		6,867	
3		8,771	
4		8,935	
5-50		less than 700 cards each	
51		2,708	
52		2,714	
53		3,189	
54		3,189	

Distribution for ANL-80-105

Internal:

J. J. Dongarra (20)	G. W. Pieper
G. T. Garvey	R. J. Royston
A. B. Krisciunas	National Energy Software Center (100)
P. C. Messina	ANL Contract File
D. M. Pahis	ANL Libraries
L. M. Phebus (2)	TIS Files (6)

External:

DOE-TIC, for distribution per UC-32 (183)
Manager, Chicago Operations and Regional Office, DOE-CORO
Chief, Office of Patent Counsel, DOE-CORO
President, Argonne Universities Association
Applied Mathematics Division Review Committee:
  G. Estrin, U. California, Los Angeles
  W. M. Gentleman, U. Waterloo
  J. M. Ortega, U. Virginia
  E. N. Pinson, Bell Telephone Labs
  S. Rosen, Purdue U.
  M. F. Wheeler, Rice U.
  D. M. Young, Jr., U. Texas at Austin
International Mathematical and Statistical Libraries (100)