Short LAPACK User's Guide

12.01.2002

Bernhard Seiwald

Institut für Theoretische Physik
Abteilung Plasmaphysik
Technische Universität Graz
Petersgasse 16, A-8010 Graz, Austria
Tel.: +43(316)873-8194
e-mail: seiwald@itp.tu-graz.ac.at

Contents

1 Introduction

LAPACK (Linear Algebra PACKage) is a powerful tool for solving linear algebra problems.

2 Used System

In our case we use LAPACK Ver.3.x and BLAS Ver.3.x. The used system is a LINUX distribution RedHat 7.2 or newer, which uses glibc2.x. Further we use the NAG Fortran95 and the Lahey Fortran95 compiler.

3 Purpose of LAPACK

At the LAPACK homepage http://www.netlib.org/lapack/ we can read:

> LAPACK is written in Fortran77 and provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems. The associated matrix factorizations (LU, Cholesky, QR, SVD, Schur, generalized Schur) are also provided, as are related computations such as reordering of the Schur factorizations and estimating condition numbers. Dense and banded matrices are handled, but not general sparse matrices. In all areas, similar functionality is provided for real and complex matrices, in both single and double precision.

4 Literature

LAPACK homepage:
http://www.netlib.org/lapack/
User's guide:
LAPACK Users' Guide
List of routines of LAPACK:
LAPACK Quick Reference Guide to the Driver Routines or
local lapackqref.ps
Man pages for LAPACK and BLAS are installed.

5 Sample Session

A sample session is shown in this section. There are some difficulties in linking the library to own programs. In the following section 5.1 a simple source is shown. Section 5.2 shows a Makefile for compiling and linking, which is very important.

ATTENTION: Set the environment variable *F90* to your preferred Fortran compiler either *nag* or *lahey* for your session or in the *xterm* where you start the Fortran compiler. *bash* users (most of our users) use the command *export F90=nag* for using *NAG F95* compiler.

5.1 Fortran90/95 Source

How to solve an equation system $A\,\mathbf{x} = \mathbf{b}$ with the routine $DGESV$ is shown in the following example.

```
!------------------------------------------------------------------------
!
! test_lapack77.f90
!
! simple demo for using LAPACK77 routines in Fortran90
!
! Author: Bernhard Seiwald
! Date:   12.01.2002
!
!------------------------------------------------------------------------



PROGRAM test_lapack77

!------------------------------------------------------------------------
!------------------------------------------------------------------------
  IMPLICIT NONE
  INTEGER,  PARAMETER :: I4B  = SELECTED_INT_KIND(9)
  INTEGER,  PARAMETER :: DP   = KIND(1.0D0)

  INTEGER,  PARAMETER :: w_us = 6
  REAL(DP), PARAMETER :: EPS  = 1.0D-9

  CHARACTER(len=255) :: filename, msg
  INTEGER(I4B) :: iunit, i_alloc, istat
  INTEGER(I4B) :: dim, info
  INTEGER(I4B) :: n, nrhs, lda, ldb
  INTEGER(I4B) :: i, j
  REAL(DP)     :: erg
  INTEGER(I4B), DIMENSION(:),   ALLOCATABLE :: ipiv
  REAL(DP),     DIMENSION(:),   ALLOCATABLE :: inhom, loes
  REAL(DP),     DIMENSION(:,:), ALLOCATABLE :: mat, mat1
!-----------------------------------------------------------------------


!-----------------------------------------------------------------------
! open file containing info about equation system
!-----------------------------------------------------------------------
  WRITE(filename,'(A)') 'test_lapack_in.dat'
  iunit = 7
  OPEN(unit=iunit, file=TRIM(filename), status='unknown', form='formatted', &
       iostat=istat)
  IF (istat /= 0) THEN
     WRITE(w_us,*) 'test_lapack77: Error on opening file ', filename
     STOP
  END IF

!-----------------------------------------------------------------------
! read dimension of matrix and allocate arrays
!-----------------------------------------------------------------------
  READ(iunit,*) dim

  ALLOCATE( mat(dim,dim), mat1(dim,dim), inhom(dim), loes(dim), &
            stat = i_alloc )
  IF(i_alloc /= 0) STOP 'test_lapack77: Allocation for arrays1 failed!'
```

```
!-----------------------------------------------------------------------
! read matrix
!-----------------------------------------------------------------------
  DO i = 1, dim
     READ(iunit, *) (mat(i,j), j=1,dim)
  END DO
!-----------------------------------------------------------------------
! read right hand side
!-----------------------------------------------------------------------
  DO i = 1, dim
     READ(iunit,*) inhom(i)
  END DO

  CLOSE(unit=iunit)


  loes = inhom
  mat1 = mat
  n    = SIZE(loes,1)
  nrhs = 1
  lda  = MAX(1,SIZE(mat,1))
  ldb  = MAX(1,SIZE(loes,1))

  ALLOCATE( ipiv(n),  stat = i_alloc )
  IF(i_alloc /= 0) STOP 'test_lapack77: Allocation for arrays2 failed!'

!-----------------------------------------------------------------------
! solve system
!-----------------------------------------------------------------------
  CALL dgesv(n, nrhs, mat1, lda, ipiv, loes, ldb, info)
  ! error handling: messages are copies from man pages
  IF ( info < 0 ) THEN
     WRITE(msg,*) 'test_lapack77:  Lapack routine dgesv did not exit ', &
          'successful! The ', info, '-th argument had an illegal value'
     PRINT *, msg
  END IF
  IF ( info > 0 ) THEN
```

```
     WRITE(msg,*) 'test_lapack77:  Lapack routine dgesv did not exit ', &
          'successful! U(', info, ',' , info, ') is exactly zero. ',    &
          'The factorization has been completed, but the factor U ',    &
          'is exactly singular, so the solution could not be computed.'
     PRINT *, msg
  END IF

!-----------------------------------------------------------------------
! check result
!-----------------------------------------------------------------------
  DO i = 1, dim
     erg = SUM(mat(i,:)*loes)
     IF ( ABS(inhom(i) - erg) > EPS ) THEN
! report error in file 'FORTRAN_INT_ERROR'
        WRITE(filename,'(A)') 'FORTRAN_INT_ERROR'
        OPEN(unit=iunit, file=TRIM(filename), status='unknown', &
             form='formatted', iostat=istat)

        IF (istat /= 0) THEN
           WRITE(w_us,*) 'test_lapack77: Error on opening file ', filename
           STOP
        END IF
        WRITE(iunit,*) 'problem in ', i, erg
        CLOSE(unit=iunit)
! stop, when first error occurs
        STOP
     END IF
  END DO


!-----------------------------------------------------------------------
! deallocate arrays
!-----------------------------------------------------------------------
  DEALLOCATE( mat, mat1, inhom, loes,  stat = i_alloc )
  IF(i_alloc /= 0) STOP 'test_lapack77: Deallocation for arrays1 failed!'
  DEALLOCATE( ipiv,  stat = i_alloc )
  IF(i_alloc /= 0) STOP 'test_lapack77: Deallocation for arrays2 failed!'
```

```
END PROGRAM test_lapack77
```

5.2 Makefile

A working Makefile is shown in this section.

```
###
###  Makefile for some tests of fortran90/95 compilers
###

#
# notes:
# OSTYPE : defined by system
# F90    : defined by user in shell;  nag, lahey
# DEBUG 0/1
#

SRCDIR  = LAPACK77

SOURCES = test_lapack77.f90

INCLUDE_FILES =
OBJECTS       = $(SOURCES:.f90=.o)
MODULES       = *.mod
OUTPUT        = test_lapack77
MAKEFILE      = Makefile

RM = rm -f

# setting ostype
OSTYPE    = $(shell uname)


# define standard Fortran90 compiler
FC = f90
LD = $(FC)
# define libs
```

```
BLAS      = -lblas
LAPACK77  = -llapack
LIBS      = $(LAPACK77) $(BLAS) $(SPECIAL_LIBG2C)

ifeq ($(OSTYPE), Linux)
##### linux #####
  ifeq ($(F90), nag)
    ### NAGf95 ###
    ## disable warnings from license manager
    export NAG_LM_OPTS=nowarn
    S_BLAS    = libblas.a libblas.so
    S_LAPACK  = liblapack.a liblapack.so
    FC        = f95-nag
    LD       = $(FC)
    ifeq ($(DEBUG), 1)
      DEBUGFLAG = -C -g -g90 -gline
    else
      DEBUGFLAG =
    endif
    FFLAGS    = -v -u -O4 -nan
    LDFLAGS   =
    BLAS      = -lblas
    LAPACK77  = -llapack
    ifeq ($(DEBUG), 1)
      LIBS      = $(LAPACK77) $(BLAS) $(SPECIAL_LIBG2C) -lefence
    else
      LIBS      = $(LAPACK77) $(BLAS) $(SPECIAL_LIBG2C)
    endif
    CUT_ASM_WARN =
  endif

  ifeq ($(F90), lahey)
    ### Lahey ###
    S_BLAS    = libblas.a libblas.so
    S_LAPACK  = liblapack.a liblapack.so
    FC        = f95-lah
    LD        = $(FC)
    ifeq ($(DEBUG), 1)
```

```
      DEBUGFLAG = --chk -g --trace
    else
      DEBUGFLAG =
    endif
    FFLAGS    = --wo --warn --f95 -O --tpp --ap
    LDFLAGS   =
    BLAS      = -lblas
    LAPACK77  = -llapack
    ifeq ($(DEBUG), 1)
      LIBS      = $(LAPACK77) $(BLAS) $(SPECIAL_LIBG2C) -lefence
    else
      LIBS      = $(LAPACK77) $(BLAS) $(SPECIAL_LIBG2C)
    endif
    CUT_ASM_WARN = 2>&1 | grep -v "/tmp/asm"
  endif
endif


$(OUTPUT): $(OBJECTS)
$(LD) $(OBJECTS) -o $(OUTPUT) $(LDFLAGS) $(LIBS) $(DEBUGFLAG)

%.o : %.f90 $(MAKEFILE)
$(FC) $(@:.o=.f90) -c -o $@ $(DEBUGFLAG) $(FFLAGS) $(CUT_ASM_WARN)


install: $(OUTPUT)
cp $(OUTPUT) $(HOME)/bin/$(OUTPUT)


basic-clean:
$(RM) $(OBJECTS) $(MODULES)

clean: basic-clean
( $(RM) *.o *.mod *.g90 *~ core \#* $(OUTPUT) )

dist:
( cd ..; tar -zcvf $(SRCDIR)-‘date +"%Y-%m-%d"‘.tar.gz \
  $(SRCDIR)/*.f90 $(SRCDIR)/test_lapack_in.dat \
  $(SRCDIR)/Doc \
  $(SRCDIR)/Makefile $(SRCDIR)/README \
  $(SRCDIR)/no_arch -X $(SRCDIR)/no_arch )

distclean: clean dist
```

The meaning of important variables is as follows:

FC Fortran95 compiler

LD use $(FC) as loader

FFLAGS compiler flags

LAPACK77 -llapack: lapack77 library

BLAS -lblas: blas library

SPECIAL_LIBG2C special library which is needed (libg2c.a) in our case this is set by system

LIBS $(LAPACK77) $(BLAS) $(SPECIAL_LIBG2C): sequence is important

LDFLAGS flags for the linker

For further information for compiler flags and linker flags please read the documentation of the Fortran95 compilers.

The really **important** steps are:

- Set the environment variable *F90* either to *nag* or *lahey* to choose a compiler.
- Right sequence in linking libraries as shown in *LIBS* above.
- Use the library *libg2c*.