LAPACK (**l**inear **a**lgebra **pack**age) is a free library of Fortran (Fortran 90) with subroutines for solving the most commonly occurring problems in numerical linear algebra.

LAPACK uses BLAS (**b**asic **l**inear **a**lgebra **s**ubroutines) library – levels 1,2,3.

`http://www.netlib.org/blas`

`http://www.netlib.org/lapack`

The name of each LAPACK and BLAS routine has All driver and computational routines have names of the form `XYYZZZ`.

The first letter, `X`, indicates the data type as follows:

```
S  REAL
D  DOUBLE PRECISION
C  COMPLEX
Z  COMPLEX*16 or DOUBLE COMPLEX
```

The next two letters, `YY`, indicate the type of matrix (or of the most significant matrix).

```
BD  bidiagonal
DI  diagonal
GB  general band
GE  general (i.e., unsymmetric, in some cases rectangular)
GG  general matrices, generalized problem (i.e., a pair of general matrices)
GT  general tridiagonal
HB  (complex) Hermitian band
HE  (complex) Hermitian
HG  upper Hessenberg matrix, generalized problem (i.e a Hessenberg and a
   triangular matrix)
```

`HP` (complex) Hermitian, packed storage
`HS` upper Hessenberg
`OP` (real) orthogonal, packed storage
`OR` (real) orthogonal
`PB` symmetric or Hermitian positive definite band
`PO` symmetric or Hermitian positive definite
`PP` symmetric or Hermitian positive definite, packed storage
`PT` symmetric or Hermitian positive definite tridiagonal
`SB` (real) symmetric band
`SP` symmetric, packed storage
`ST` (real) symmetric tridiagonal
`SY` symmetric
`TB` triangular band
`TG` triangular matrices, generalized problem (i.e., a pair of triangular matrices)
`TP` triangular, packed storage
`TR` triangular (or in some cases quasi-triangular)
`TZ` trapezoidal
`UN` (complex) unitary
`UP` (complex) unitary, packed storage

The last three letters ZZZ indicate the computation performed, see,

`http://www.netlib.org/lapack/lug/node26.html#tabdrivelineq`

For example, `SGEBRD` is a `S`ingle precision routine that performs a bidiagonal reduction (`BRD`) of a real `GE`neral matrix.

Installation

1. Download the library, e.g., the file `lapack-3.8.0.tar.gz` and unpack it by

   ```
   gunzip lapack-3.8.0.tar.gz
   tar xf lapack-3.8.0.tar
   ```

   the directory with archive `lapack-3.8.0` appears.

2. create file `make.inc`, the simplest way is

   ```
   cd lapack-3.8.0
   cp make.inc.example make.inc
   ```

   and edit (if necessary) the resulting file.

3. run the command `make`, command `ls -l *.a` lists the libraries

   ```
   -rw-rw-r-- 1 dolejsi dolejsi 12385112 říj 25 19:48 liblapack.a
   -rw-rw-r-- 1 dolejsi dolejsi   631792 říj 25 19:48 librefblas.a
   ```

4. a little faster way is to replace the previous step by steps:

   (a) create the `BLAS` library by

   ```
   cd BLAS/SRC
   make
   cd ../../
   ```

   (b) translate LAPACK library by

   ```
   cd SRC
   make
   cd ../
   ```

Link of LAPACK with your own code

Example of my code (in file `lap_sub.f90`) using LAPACK subroutines `dgetri`, `dgetrf`:

```
 subroutine  MblockInverse(n, A)
  integer, intent(in) :: n
  real, dimension(1:n,1:n), intent(inout) :: A
  external:: dgetri, dgetrf          ! subroutines from LAPACK
  real, dimension(:), allocatable :: ident, work
  integer :: info, iwork

  iwork = 100 * 30
  allocate(ident(1:n), work(1:iwork) )
  ident(:) = 1.

  call DGETRF(n, n, A, n, ident, info )
  if(info /= 0 ) print*,'Problem 1 in MblockInverse in matrix.f90 ', info
  if(info /= 0 ) stop

  call DGETRI(n, A, n, ident, work,  iwork, info )
  if(info /= 0 ) print*,'Problem 2 in MblockInverse in matrix.f90 ', info
  if(info /= 0 ) stop

  deallocate(ident, work)

end subroutine MblockInverse
```

Example of my `Makefile`:

```
TARGETS= lap_sub.o geom.o integ.o f_mapping.o  main.o
FFLAGS=  -fPIC -fdefault-real-8 -O2 -w
LIBS= lapack-3.8.0/liblapack.a lapack-3.8.0/librefblas.a
FXX=gfortran

all: Adgfem

Adgfem: $(TARGETS)
        $(FXX) $(FFLAGS) -o Adgfem $^ $(LIBS)

clean:
        -rm -f Adgfem  *.o *.mod

%.o:%.f90
        $(FXX) $(FFLAGS) -c $?
```

For globally installed library using

```
sudo apt-get install libblas-dev liblapack-dev
```

change libraries to

```
  LIBS=-llapack  -lblas
```

in this case translator seeks files

```
  /usr/lib/libblas.a
  /usr/lib/liblapack.a
```

Write a code for the computation of the **inversion** of matrix $\mathbb{A} \in \mathbb{R}^{n\times n}$ given by

$$\mathbb{A} = \begin{pmatrix} 2(1+\frac{1}{n}) & \frac{1}{n}-1 & 0 & 0 & \dots & 0 \\ 0 & 2(1+\frac{2}{n}) & \frac{1}{n}-1 & 0 & \dots & 0 \\ 0 & 0 & 2(1+\frac{3}{n}) & \frac{1}{n}-1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \dots & \vdots \\ 0 & 0 & \dots & \dots & 2(1+\frac{n-1}{n}) & \frac{1}{n}-1 \\ 0 & 0 & 0 & \dots & \dots & 2(1+\frac{n}{n}) \end{pmatrix}$$

using the pre-prepared code and `LAPACK`.

1. From the link `msekce.karlin.mff.cuni.cz/~dolejsi/Vyuka/NS_source/LAPACK/index.html` download the file `LAPACK_test.tgz`
2. unzip it by the command

   ```
   tar xfz LAPACK_test.tgz
   ```

3. go to directory `LAPACK_test` and look through files `*.f90` and `Makefile` and translate the code, e.g.
   (a) #link to LAPACK

   ```
   ln -s ../lapack-3.8.0/ LAPACK
   ```

   (b) # translate matrix.f90

   ```
   make
   ```

4. run the code by `./matrix` and perform several experiments.

Turorial # 2

Modify code `matrix_sub.f90` using `TG` and not `GE`

Advanced tutorial

Write a code (by the modification of the code from **tutorial3**) for the multiplication of matrices $\mathbb{C} = \mathbb{A}\mathbb{B}$ using `BLAS` library. Compare the computational times (speed of computation) with three variants tested in **tutorial3**.

Hints:

1. use subroutine `DGEMM` in `BLAS`, file `dgemm.f` in directory `lapack-3.8.0/BLAS/SRC/`
2. study the parameters of the subroutine
3. modify the code `multi.f90` and `Makefile` from **tutorial3** by adding the `BLAS` (and `LAPACK`) library

One possible solution is on
`http://msekce.karlin.mff.cuni.cz/~dolejsi/Vyuka/NS_source/LAPACK/index.html`,
links `multi_lapack.f90` and `Makefile_lapack`
translation and running of the code in the same directory

```
make -f Makefile_lapack
```

```
./multi_lapack
```

Advanced tutorial # 2

Write a code for the solution of linear algebaric systems $\mathbb{A}\mathbf{x} = \mathbf{b}$ using `LAPACK` library.

Hints:

1. use subroutine `DGESV` in `LAPACK`, file `dgesv.f` in directory `lapack-3.8.0/SRC/`
2. study the parameters of the subroutine
3. use subroutine `DGESVX` in `LAPACK`, file `dgesvx.f` in directory `lapack-3.8.0/SRC/` which provides additional info, forward error, bakcward error, condition number

One possible solution is on
`http://msekce.karlin.mff.cuni.cz/~dolejsi/Vyuka/NS_source/LAPACK/index.html`,
links `solve_lapack.f90` and `Makefile_lapack`
translation and running of the code in the same directory

```
make -f Makefile_lapack

./solve_lapack
```