数値計算演習
30.11.05
--LAPACKの使い方--

- 課題
  - 【課題1】LAPACKにある線形代数計算ルーチンdgesvをつかったプログラムを作れ．計算に要する時間を計測せよ．n=2000まで適当に間引いて計測し，そのn依存性をgnuplotで表示せよ．
  - 【課題2】このマシンのGFLOPS値を求めよ．
  - 【課題3(レポート：提出は来週以降でよい)】固有値を求めるdseyvをつかって同様の検証をおこなえ．Mapleでの検証やLatexでのレポート作成などいままでの課題を思い出して，完成したレポートを作成せよ．
- ssh address
  - グララボのマシンには，LAPACKやBLASがinstallされてないので，他のunix machineにリモートログインして使う．
- tar, gzip, sftp
  - ファイルの転送，圧縮・解凍のためのコマンド
  - sftp: file transfer protocol
  - tar, gzip, sftp put
  - sftp get, gunzip, tar -xvf
    - tar -cvf test.tar test （ファイルをまとめる）
      gzip test.tar （圧縮）
      sftp bob@192.168.1.32
          put test.tar.gz
    - sftp
          get test.tar.gz (cdなどで適切なdirectoryへ移動後)
      gunzip test.tar.gz (解凍)
      tar -tvf test.tar (中身の表示)
      tar -xvf test.tar (ファイルへの展開)
- LAPACK
  - 逆行列・固有値計算については，
    http://ist.ksc.kwansei.ac.jp/~nishitani/Lectures/NumRecipe/
    を参照．
  - http://www.waseda.jp/ocw/ComputerScience/17-4001CProgramming1Fall2003/StudyMaterials/lec2003.html
  - 大石進一著「Linux数値計算ツール」9章，LAPCKを使いこなそう
    参照
- 線形代数計算ルーチンdgesvを使ったプログラム．
  - ```
    #include <stdio.h>
    #include <stdlib.h>
    #include <math.h>
    #include <time.h>

    void printMatrix(double *a, double *b, int n);

    int main(void){
      long n, nrhs=1, lda,ldb, info;
      //  double A[LDA*LDA], B[LDB*NRHS];
      clock_t start, end;
      int i,j;
      double *a,*b;
      long *ipiv;

      scanf("%ld",&n);
      printf("%d ",n);

      lda=ldb=n;
      a=(double *)malloc(n*n*sizeof(double));
      b=(double *)malloc(n*sizeof(double));
      ipiv=(long *)malloc(n*sizeof(long));

      for (i=0;i<n;i++){
    ```
    ```

```
      for (j=0;j<n;j++){
        a[i*n+j]= 2*(double) random() / RAND_MAX - 1.0;
      }
    }

    for (i=0;i<n;i++){
      b[i]= 2*(double) random() / RAND_MAX - 1.0;
    }

    //  printMatrix(a,b,n);
    start = clock();
    dgesv_(&n, &nrhs, a, &lda, ipiv, b, &ldb, &info);
    //  MatrixInverse(a,b,n);
    //  printMatrix(a,b,n);
    end = clock();
    printf("%10.4f\n", (double)(end-start)/CLOCKS_PER_SEC);
    free(a);
    free(b);
    return 0;
  }

  void printMatrix(double *a, double *b, int n){
    int i,j;

    printf("\n");
    for (i=0;i<n;i++){
      for (j=0;j<n;j++){
        printf("%10.5f",a[i*n+j]);
      }
      printf(":%10.5f",b[i]);
      printf("\n");
    }
    printf("\n");
    return;
  }
```

- コンパイル
  - for Mac OSX
    - gcc -O3 -faltivec -bind_at_load -framework vecLib -o MatrixInverse2 MatrixInverse2.c
  - for linux
    - gcc MatrixInverse.c -llapack -lblas -lg2c
- MFLOPS
  - 浮動小数点演算が1秒間に何回行われたかを示す．CPU性能の基本値．
  - DGESVで1000x1000の演算をおこなうとtime x MFlops = 668.5 という定数になる．
- dgesv.cのhead
  - /* Subroutine */ int dgesv_(integer *n, integer *nrhs, doublereal *a, integer
    *lda, integer *ipiv, doublereal *b, integer *ldb, integer *info)
    {
    /* -- LAPACK driver routine (version 3.0) --
       Univ. of Tennessee, Univ. of California Berkeley, NAG Ltd.,
       Courant Institute, Argonne National Lab, and Rice University
       March 31, 1993

       Purpose
       =======

       DGESV computes the solution to a real system of linear equations
          A * X = B,
       where A is an N-by-N matrix and X and B are N-by-NRHS matrices.

       The LU decomposition with partial pivoting and row interchanges is

```
used to factor A as
   A = P * L * U,
where P is a permutation matrix, L is unit lower triangular, and U is
upper triangular.  The factored form of A is then used to solve the
system of equations A * X = B.

Arguments
=========

N       (input) INTEGER
        The number of linear equations, i.e., the order of the
        matrix A.  N >= 0.

NRHS    (input) INTEGER
        The number of right hand sides, i.e., the number of columns
        of the matrix B.  NRHS >= 0.

A       (input/output) DOUBLE PRECISION array, dimension (LDA,N)
        On entry, the N-by-N coefficient matrix A.
        On exit, the factors L and U from the factorization
        A = P*L*U; the unit diagonal elements of L are not stored.

LDA     (input) INTEGER
        The leading dimension of the array A.  LDA >= max(1,N).

IPIV    (output) INTEGER array, dimension (N)
        The pivot indices that define the permutation matrix P;
        row i of the matrix was interchanged with row IPIV(i).

B       (input/output) DOUBLE PRECISION array, dimension (LDB,NRHS)
        On entry, the N-by-NRHS matrix of right hand side matrix B.
        On exit, if INFO = 0, the N-by-NRHS solution matrix X.

LDB     (input) INTEGER
        The leading dimension of the array B.  LDB >= max(1,N).

INFO    (output) INTEGER
        = 0:  successful exit
        < 0:  if INFO = -i, the i-th argument had an illegal value
        > 0:  if INFO = i, U(i,i) is exactly zero.  The factorization
              has been completed, but the factor U is exactly
              singular, so the solution could not be computed.
```

▼ 固有値を求めるdseyv.cのhead

- ```
  #include "blaswrap.h"
  #include "f2c.h"

  /* Subroutine */ int dsyev_(char *jobz, char *uplo, integer *n, doublereal
  *a,
          integer *lda, doublereal *w, doublereal *work, integer *lwork,
         integer *info)
  {
  /*  -- LAPACK driver routine (version 3.0) --
         Univ. of Tennessee, Univ. of California Berkeley, NAG Ltd.,
         Courant Institute, Argonne National Lab, and Rice University
         June 30, 1999


      Purpose
      =======

      DSYEV computes all eigenvalues and, optionally, eigenvectors of a
      real symmetric matrix A.
  ```
  ```

```
          Arguments
          =========

          JOBZ    (input) CHARACTER*1
                  = 'N':  Compute eigenvalues only;
                  = 'V':  Compute eigenvalues and eigenvectors.

          UPLO    (input) CHARACTER*1
                  = 'U':  Upper triangle of A is stored;
                  = 'L':  Lower triangle of A is stored.

          N       (input) INTEGER
                  The order of the matrix A.  N >= 0.

          A       (input/output) DOUBLE PRECISION array, dimension (LDA, N)
                  On entry, the symmetric matrix A.  If UPLO = 'U', the
                  leading N-by-N upper triangular part of A contains the
                  upper triangular part of the matrix A.  If UPLO = 'L',
                  the leading N-by-N lower triangular part of A contains
                  the lower triangular part of the matrix A.
                  On exit, if JOBZ = 'V', then if INFO = 0, A contains the
                  orthonormal eigenvectors of the matrix A.
                              On exit, if JOBZ = 'V', then if INFO = 0, A
      contains the
                  orthonormal eigenvectors of the matrix A.
                  If JOBZ = 'N', then on exit the lower triangle (if UPLO='L')
                  or the upper triangle (if UPLO='U') of A, including the
                  diagonal, is destroyed.

          LDA     (input) INTEGER
                  The leading dimension of the array A.  LDA >= max(1,N).

          W       (output) DOUBLE PRECISION array, dimension (N)
                  If INFO = 0, the eigenvalues in ascending order.

          WORK    (workspace/output) DOUBLE PRECISION array, dimension (LWORK)
                  On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

          LWORK   (input) INTEGER
                  The length of the array WORK.  LWORK >= max(1,3*N-1).
                  For optimal efficiency, LWORK >= (NB+2)*N,
                  where NB is the blocksize for DSYTRD returned by ILAENV.

                  If LWORK = -1, then a workspace query is assumed; the routine
                  only calculates the optimal size of the WORK array, returns
                  this value as the first entry of the WORK array, and no error
                  message related to LWORK is issued by XERBLA.

          INFO    (output) INTEGER
                  = 0:  successful exit
                  < 0:  if INFO = -i, the i-th argument had an illegal value
                  > 0:  if INFO = i, the algorithm failed to converge; i
                        off-diagonal elements of an intermediate tridiagonal
                        form did not converge to zero.
```