Linear **A**lgebra **PACK**age

(LAPACK)
лекция 4

Типы матриц

- Общие
- Ленточные
- Симметричные
- Треугольные
- Трехдиагональные
- Пр.

Операции

- Решение систем линейных уравнений.
- Оценка числа обусловленности.
- Уточнение решений систем линейных уравнений и вычисление ошибок.
- Разложение матриц (факторизация).
- Метод наименьших квадратов.
- Поиск собственных чисел.

Варианты использования

Для решения СЛАУ (обычная матрица)

- ?getrf (LU факторизация),
- ?getrs (поиск решения)
- ?gerfs (уточнение решения)

Или (driver function)

- ?gesvx

Формат функций

- ?yyzzz или ?yyzz
- ? - s, c, d, z
- yy — тип матрицы и способ упаковки
- ge — обычная
- gb — ленточная
- gt — трехдиагональная
- tr — треугольная
- пр.

Формат функций (продолжение)

- yyy - операция
- trs — решение СЛАУ с факторизованой матрицей
- rfs — уточнение решения
- con — оценка числа неопределенности
- пр.
- Функции драйвера могут заканчиваться на -sv, -svx, -svxx

Разложение матриц

- LU-разложение
  - $A=LU$
- Разложение Холесского
  - $A=LL^T$, где A - это симметричная положительно определенная матрица, L - верхнетреугольная матрица, L^T - ее транспонированый вариант
- Разложение Банч-Кауфмана (Bunch-Kaufman)
  - $A=LDL^T$

Вычисление обратной матрицы

- $Ax = b$
- $x = A^{-1}b$

Не пытайтесь через обратную матрицу решить СЛАУ. Вызов специальных функций более эффективен и точен.

Метод наименьших квадратов

Дана матрица A и вектор b. Найти такой вектор x, для которого $||Ax-b||_2$ минимальна.

- ?geqrf - A = QR
- ?ormqr - $C = Q^H B$
- ?trsm - для решения RX = C.
- или (функция драйвера)
- ?gels

Вспомогательные функции

Более 150 функций от проверки числа на NaN, до масштабирования матриц.

Практика: использование LAPACK

Решение уравнения Ax=B

gesv(n, nrhs, a, lda, ipiv, b, ldb, info)

n – порядок матрицы A

nrhs – количество столбцов матрицы B

a, b – матрицы, на выходе b - решение

lda, ldb – строчные размерности матриц a, b (lda≥ max(1,n) и ldb≥ max(1, n))

ipiv – вектор перестановки строк

info – содержит результат выполнения функции(0 – успешно)

Практика: использование LAPACK

Норма вектора (Евклидова)

nrm2(n, x, incx)

n – размерность

x – вектор

incx - приращение для элементов из x

Задание

$$\frac{du}{dt} = \frac{d^2u}{dx^2}$$

$$\frac{u_i^{n+1} - u_i^n}{\tau} = \frac{u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}}{h_x^2}$$

$$H = 1, H / h_x = 64, \tau = 0.01$$

Граничные условия: $u^0 = 0, u_0^t = 5t, u_H^t = 0$

- Построить график функции U(x) в момент времени T=1, 3, 5.

Вариант решения

```
for (i=0; i<N; i++){
      u[i] = 0;  // решение
                  u_old[i] = 0;
                  pr[i] = 0; // правая часть
}
// сборка матрицы

// нижняя поддиагональ
values[0] = 0;
for (i=1; i<N; i++)                values[i] = (-1.0/(hx*hx));

// главная диагональ
for (i=0; i<N; i++)                values[i+N] = (2.0/(hx*hx)) + 1.0/dt;

// верхняя поддиагональ
for (i=0; i<N-1; i++)              values[i+2*N] = (-1.0/(hx*hx));
values[3*N-1] = 0;
```

```
r = new double[N];
ipiv = new int[N];
int n =N;
int info;
// учет краевых условий
    values[N] = 1;
    values[2*N] = 0;


dgttrf(&n, &values[1], &values[n], &values[n*2], r, ipiv, &info );
time = time0; ntime = 0;
do {
    // сборка вектора
    for (i=0; i<N; i++)  pr[i] = u[i]/dt;
    // учет краевых условий
    pr[0] = 5.0*time;
    dgttrs(&trans, &n, &nrhs, &values[1], &values[n], &values[n*2],
                                        r, ipiv, pr, &ldb, &info);
    for (i=0; i<N; i++)  u[i] = pr[i];
    // запись результата
    ntime ++; time = time0 + ntime*dt;
}while(time <= 5);
```

ScaLAPACK

- **ScaLAPACK = Scalable LAPACK, BLAS + LAPACK + BLACS**
- распределенные вычисления (поверх **MPI**)

ScaLAPACK

- Параллельное расширение LAPACK для компьютеров с распределённой памятью (кластеров).
- Обеспечивает параллельную версию LAPACK функций (обычно добавлением p: zgetrf -> pzgetrf).
- Использует PBLAS (Parallel BLAS) и BLAS, так же как и LAPACK для локальных операций.
- LAPACK уже содержит параллелизацию для машин с общей памятью (многоядерных)!

ScaLAPACK

Подпрограммы библиотеки ScaLAPACK:

- Драйверные (решает законченную задачу, например, решение СЛАУ).
- Вычислительные (отдельные подзадачи, например, LU-разложение матрицы).
- Служебные (внутренние вспомогательные действия).

Векторы и матрицы, являющиеся параметрами подпрограмм, должны быть предварительно распределены по процессам

ScaLAPACK: использование

- Инициализировать сетку процессов.

blacs_pinfo(mypnum, nprocs)
mypnum – число от 0 до (nprocs – 1), идентифицирующий каждый процесс
nprocs – число доступных процессов

- Распределить данные на сетку процессов.
- Вызвать вычислительную подпрограмму.
- Функции из библиотеки.
- Освободить сетку процессов.

blacs_exit(continue)

continue = 0 – не использовать процессы

continue != 0 – продолжить использование процессов

Сборка программы (C\C++)

```
/opt/mpich/bin/mpicc <user files to link> \
   -L$MKLPATH \
   -lmkl_scalapack_core \
   -lmkl_blacs_intelmpi \
   -lmkl_lapack \
   -lmkl_intel -lmkl_intel_thread \
   -lmkl_lapack -lmkl_core \
   -liomp5 -lpthread
```

Сборка программы (Фортран)

```
/opt/intel/mpi/3.0/bin/mpiifort \
   <user files to link> \
   -L$MKLPATH \
   -lmkl_scalapack_lp64 \
   -lmkl_blacs_intelmpi_lp64 \
   -lmkl_lapack \
   -lmkl_intel_lp64 -lmkl_intel_thread \
   -lmkl_lapack -lmkl_core \
   -liomp5 -lpthread
```