CSE 4351/5351 Notes 6: Numerical Problems (Systems of Linear Equations)

Approaches:

1. Elimination methods

    Gaussian

    LU

    Householder (available on web page, not on test)

2. Iteration

Gaussian Elimination

Solve Ax = b

$$\begin{array}{cc} & \begin{array}{cc} n & 1 \end{array} \\ n & \left[\begin{array}{c:c} A & b \end{array}\right] \end{array}$$

Subtract a multiple of the first row to zero the first position in all subsequent rows.

Subtract a multiple of the second row to zero the second position in all subsequent rows.

.
.
.

Subtract a multiple of (n-1)st row to zero (n-1)st position in last row.

$$\begin{array}{cc} & \begin{array}{cc} n & 1 \end{array} \\ n & \left[\begin{array}{l:c} \text{- - - - - - - -} & \\ 0\ \text{- - - - - - -} & \\ 0\ 0\ \text{- - - - - -} & \\ 0\ 0\ 0\ \text{- - - - -} & b' \\ \quad\ \ldots & \\ 0\ 0\ 0\ 0\ 0\ \text{- - -} & \\ 0\ 0\ 0\ 0\ 0\ 0\ \text{- -} & \\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ \text{-} & \end{array}\right] \end{array}$$

Now back substitution is used to compute x_n, x_{n-1}, x_{n-2}, ..., x_1.

To ensure finding a solution, if it exists, pivoting (partial or full) must be used.

1. Find maximum absolute value in column.

2. Exchange rows for pivot (or use pointers).

3. Elimination

Can parallelize the following (just interleaving and barriers on shared memory system - see `tsp6LUPT.c`):

1. Search for pivot.

2. Exchange rows.

3. Elimination.

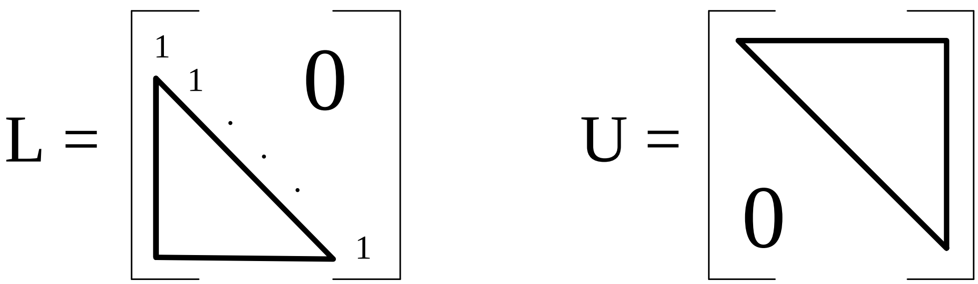
4. Back substitution - doesn't help much (don't solve each equation completely, substitute value of variable (as soon as it is known) into all equations).

Nearly linear speed-up when number of processors is small.

LU Decomposition

Especially useful when b (in Ax = b) may change, especially when b will take on the columns in the identity matrix for inverting A.

Principle is to generate two matrices L and U such that their product, LU, is a permutation of the rows in A. The form of these two matrices is:



i.e. L is unit lower-triangular and U is upper-triangular.

LU decomposition extends the usual Gaussian elimination by saving the "row multipliers" in L (U is unchanged). Due to the triangular format, a single n-by-n matrix may be used.

Example (without pivoting)

$$A = \begin{array}{cccc} 2 & 3 & 1 & 5 \\ 6 & 13 & 5 & 19 \\ 2 & 19 & 10 & 23 \\ 4 & 10 & 11 & 31 \end{array}$$

Entry [1,1] is divided into all other entries below it in column 1 to give the row multiplier. That value is multiplied by each entry in the first row and subtracted from the respective entry in the row of the row multiplier.

$$\begin{array}{rc|cccl} & 2 & 3 & 1 & 5 & \longleftarrow U \\ \hline & 3 & 4 & 2 & 4 & \\ & 1 & 16 & 9 & 18 & \\ L & 2 & 4 & 9 & 21 & \end{array}$$

Entry [2,2] is divided into all other entries below it in column 2 to give the row multiplier. Again, that is used to reduce each row using the second row.

$$\begin{array}{lccccl} & 2 & 3 & 1 & 5 & \longleftarrow U \\ & 3 & 4 & 2 & 4 & \\ & 1 & 4 & 1 & 2 & \\ L \nearrow & 2 & 1 & 7 & 17 & \end{array}$$

Similarly using entry [3,3]:

$$\begin{array}{lccccl} & 2 & 3 & 1 & 5 & \longleftarrow U \\ & 3 & 4 & 2 & 4 & \\ & 1 & 4 & 1 & 2 & \\ L \nearrow & 2 & 1 & 7 & 3 & \end{array}$$

To give the final LU decomposition:

$$L = \begin{array}{cccc} 1 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 2 & 1 & 7 & 1 \end{array} \qquad U = \begin{array}{cccc} 2 & 3 & 1 & 5 \\ 0 & 4 & 2 & 4 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 3 \end{array}$$

To solve Ax = b, we first solve Ly = b by forward substitution and then Ux = y by back substitution. Suppose that b =

$$\begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$$

y[1] = 1   y[2] = 2 - 3y[1] = -1   y[3] = 3 - 1y[1] - 4y[2] = 6   y[4] = 4 - 2y[1] - 1y[2] - 7y[3] = -39

x[4] = -39/3 = -13

x[3] = 6 - 2*x[4] = 32

x[2] = (-1 - 2x[3] - 4x[4])/4 = -3.25

x[1] = (1 - 3x[2] - x[3] - 5x[4])/2 = 21.875

Example with pivoting, Π indicates the row swaps used in pivoting

$$A = \begin{array}{cccc} 2 & 0 & 2 & .6 \\ 3 & 3 & 4 & -2 \\ 5 & 5 & 4 & 2 \\ -1 & -2 & 3.4 & -1 \end{array} \qquad \Pi = \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array}$$

The elimination occurs as before, except for choosing the largest absolute value in the column. To start, 5 is selected in column 1 and the first and third rows are swapped:

$$A = \begin{array}{cccc} 5 & 5 & 4 & 2 \\ 3 & 3 & 4 & -2 \\ 2 & 0 & 2 & .6 \\ -1 & -2 & 3.4 & -1 \end{array} \qquad \Pi = \begin{array}{c} 3 \\ 2 \\ 1 \\ 4 \end{array}$$

Now the reduction is performed:

$$\begin{array}{ll} U & \\ & A = \begin{array}{c|ccc} 5 & 5 & 4 & 2 \\ \hline .6 & 0 & 1.6 & -3.2 \\ .4 & -2 & .4 & -.2 \\ -.2 & -1 & 4.2 & -.6 \end{array} \\ L & \end{array} \qquad \Pi = \begin{array}{c} 3 \\ 2 \\ 1 \\ 4 \end{array}$$

Now the highest absolute value in {0, -2, -1} is chosen and row swapping.

$$\begin{array}{ll} U & \\ & A = \begin{array}{c|ccc} 5 & 5 & 4 & 2 \\ \hline .4 & -2 & .4 & -.2 \\ .6 & 0 & 1.6 & -3.2 \\ -.2 & -1 & 4.2 & -.6 \end{array} \\ L & \end{array} \qquad \Pi = \begin{array}{c} 3 \\ 1 \\ 2 \\ 4 \end{array}$$

Now the second column and last two rows are reduced:

$$\begin{array}{ll} U & \\ & A = \begin{array}{cc|cc} 5 & 5 & 4 & 2 \\ .4 & -2 & .4 & -.2 \\ .6 & 0 & 1.6 & -3.2 \\ -.2 & .5 & 4 & -.5 \end{array} \\ L & \end{array} \qquad \Pi = \begin{array}{c} 3 \\ 1 \\ 2 \\ 4 \end{array}$$

The pivot is now chosen from {1.6, 4}:

U

$$A = \begin{array}{cccc} 5 & 5 & 4 & 2 \\ .4 & -2 & .4 & -.2 \\ -.2 & .5 & 4 & -.5 \\ .6 & 0 & 1.6 & -3.2 \end{array} \qquad \Pi = \begin{array}{c} 3 \\ 1 \\ 4 \\ 2 \end{array}$$

L

And reduce for the last time:

U

$$A = \begin{array}{cccc} 5 & 5 & 4 & 2 \\ .4 & -2 & .4 & -.2 \\ -.2 & .5 & 4 & -.5 \\ .6 & 0 & .4 & -3 \end{array} \qquad \Pi = \begin{array}{c} 3 \\ 1 \\ 4 \\ 2 \end{array}$$

L

This gives L and U as the following matrices. If multiplied together, the result is the original matrix permuted according to Π.

$$L = \begin{array}{cccc} 1 & 0 & 0 & 0 \\ .4 & 1 & 0 & 0 \\ -.2 & .5 & 1 & 0 \\ .6 & 0 & .4 & 1 \end{array} \qquad U = \begin{array}{cccc} 5 & 5 & 4 & 2 \\ 0 & -2 & .4 & -.2 \\ 0 & 0 & 4 & -.5 \\ 0 & 0 & 0 & -3 \end{array}$$

When solving Ax = b, the forward substitution for Ly = b permutes the b vector according to Π. Back substitution is unchanged.

PARALLELIZING ELIMINATION METHODS ON A DISTRIBUTED MEMORY SYSTEM

Interleaved (cyclic) allocation of rows on processors - balances load (see diameter example in MPI notes), but mapping code is tedious.

All processors remain active nearly the entire computation

Pivot search involves local scan (of column in remaining rows) followed by combining (`MPI_Allreduce()` for maximum absolute value) to determine pivot

Pivot row must be broadcast to all processors

Substitution (if distributed) involves considerable broadcasting

`tsp6LUM.c` includes the following functions to handle the subscript mapping:

`globalSubscript2rank(x)` - Determines the rank of the process that stores element `x` of array (the ‘‘owner’’).

`globalSubscript2localSubscript(x,rank)` - Returns the smallest local subscript that has a global subscript ≥ `x`. > is needed for cases where it is necessary to scan the table after a particular global subscript.

`localSubscript2globalSubscript(x,rank)` - It is often convenient to code in terms of local subscripts, but then to use global subscripts when coordinating several processes.

Row swap is not necessarily critical

May use (simpler) contiguous allocation for rows - interleaved does not help

In place of pivot table, use table that indicates the elimination round when each row won the pivot ‘‘tournament’’ and was then broadcast for elimination

Depends on randomness of input for load balancing and to save slight swap delay.

ITERATIVE METHODS

1. Rewrite equations to get the i-th equation to isolate i-th variable.

2. Assign initial values.

3. Iterate, solving equations to get new values.

   a. Jacobi - Always use values from previous rounds

   b. Gauss-Seidel - Always use most recent value for variables, i.e. inherently sequential.

   c. Parallel iteration - ignore recency of values, i.e. no synchronization

   d. Variants - SOR - combines Jacobi & Gauss-Seidel

Convergence - How much does each variable change between iterations?

COMPARISON OF METHODS

System of 720 equations, 6 non-zero coefficients in each.

LU - sequential (tsp6LU.c)	34 seconds
LU - 1 thread (tsp6LUPT.c)	36 seconds
LU - 2 threads	23.5 seconds
LU - 1 MPI process (tsp6LUM.c)	34.3 seconds
LU - 2 MPI processes on 1 box	17.9 seconds
LU - 2 MPI processes on 2 boxes	20.5 seconds
LU - 4 MPI processes on 2 boxes	14 seconds
Householder - sequential (tsp6hh.c)	11.2 seconds
Householder - 2 threads (tsp6hhPT.c)	7.2 seconds
Householder - 4 MPI processes on two boxes (tsp6hhM.c)	10 seconds