Math 465/565 **Homework 2** **Due Oct. 2, 2009**
Grady Wright

- Instructions: 465/565 students must answer all the questions unless noted otherwise. For example 465 students need to answer questions 1–6, and 565 students need to answer 2–8.
- Teamwork is encouraged! Please let me know if you are having trouble finding someone to team up with.

1. [465 only] Bradie, p.202, problem 7.
2. When deriving the LU decomposition algorithm we used the following result:

$$\begin{bmatrix} 1 & & & & & & & & \\ 0 & 1 & & & & & & & \\ 0 & 0 & \ddots & & & & & & \\ \vdots & \vdots & \ddots & 1 & & & & & \\ 0 & 0 & 0 & m_{i+1,i} & 1 & & & & \\ 0 & 0 & 0 & m_{i+2,i} & 0 & \ddots & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \\ 0 & 0 & 0 & m_{n,i} & 0 & \vdots & \vdots & \ddots & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & & & & & & & & \\ 0 & 1 & & & & & & & \\ 0 & 0 & \ddots & & & & & & \\ \vdots & \vdots & \ddots & 1 & & & & & \\ 0 & 0 & 0 & -m_{i+1,i} & 1 & & & & \\ 0 & 0 & 0 & -m_{i+2,i} & 0 & \ddots & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \\ 0 & 0 & 0 & -m_{n,i} & 0 & \vdots & \vdots & \ddots & 1 \end{bmatrix},$$

   for $i = 1, 2, \ldots, n$. Show that that it is correct.

3. Cholesky decomposition
   (a) Implement a Matlab function for computing the Cholesky decomposition of a symmetric positive definite (SPD) matrix A. The function should take as input a symmetric matrix A and output a lower triangular matrix L such that $A = LL^T$ if A is SPD. If the input matrix A is not SPD the function should report an error. The function declaration should be

   ```
   [L,err]=cholesky(A)
   ```

   where `err`=0 if the matrix is SPD and `err`=1 otherwise. For full credit, your function should avoid unnecessary computations.

   (b) Verify that your code works by testing it on the `minij` matrix from the matrix `gallery` function in Matlab , which can be obtained with the Matlab call

   ```
   >> A = gallery('minij',n)
   ```

   where `n` is the number of rows (and columns) of the matrix. The Cholesky decomposition of this matrix should be a lower triangular matrix where every value in the lower triangular matrix is 1. Report your results for a small value of n.

   (c) Use your `cholesky` function to determine which of the following family of matrices are SPD (the statements below are the exact calls you should use in Matlab to create given matrix):

```
M = magic(n)
H = hilb(n)
P = pascal(n)
I = eye(n)
D = -diag(ones(n-1,1),-1) + 2*diag(ones(n,1),0) - diag(ones(n,1),1)
```

Choose $n = 10$ and report SPD or not SPD for the above matrices.

(d) Implement a MATLAB function called `cholesky_solve` that computes the *solution* of a SPD system of equations $Ax = b$ after the Cholesky decomposition of A has been computed with your `cholesky` function above. This new function should take as input the lower triangular matrix L from your `cholesky` function, and the RHS vector b. It should return the solution vector x. A possible function declaration is

```
x = cholesky_solve(L,b)
```

Test `cholesky` and this function on the SPD linear system:

$$\begin{bmatrix} 4 & 1 & 1 & 1 \\ 1 & 4 & 1 & 1 \\ 1 & 1 & 4 & 1 \\ 1 & 1 & 1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3 \\ -3 \\ 3 \\ -3 \end{bmatrix}$$

Report your results.

(e) Determine separately the number of additions/subtractions, multiplications/divisions, and square roots required for computing the Cholesky decomposition an n-by-n SPD matrix A. Show your work.

4. Tridiagonal matrices appear suprisingly often in applications. Due to their simple form, it is possible to significantly speed up the LU decomposition of certain types of these matrices using Crout (or Thomas) factorization.

(a) Write a MATLAB program that uses Crout's algorithm for efficiently LU decomposing a tridiagonal matrix A, i.e.

$$\underbrace{\begin{bmatrix} a_1 & c_1 & & & & \\ b_2 & a_2 & c_2 & & & \\ & b_3 & a_3 & c_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & c_{n-1} \\ & & & & b_n & a_n \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} l_1 & & & & & \\ b_2 & l_2 & & & & \\ & b_3 & l_3 & & & \\ & & \ddots & \ddots & & \\ & & & \ddots & \ddots & \\ & & & & b_n & l_n \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} 1 & u_1 & & & & \\ & 1 & u_2 & & & \\ & & 1 & u_3 & & \\ & & & \ddots & \ddots & \\ & & & & \ddots & u_{n-1} \\ & & & & & 1 \end{bmatrix}}_{U}$$

Your function should take as input three *vectors* containing the entries of a, b, and c. It should output *vectors* l and u containing the entries from the LU decomposition. A possible function declaration in MATLAB is

```
function [l,u] = crout_lu(a,b,c)
```

Turn in a listing of your function.

(b) Consider the $n \times n$ tridiagonal matrix

$$A_n = \begin{bmatrix} 1 & -\frac{1}{2} & & & & \\ -\frac{2}{2} & 2 & -\frac{2}{2} & & & \\ & -\frac{3}{2} & 3 & -\frac{3}{2} & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & -\frac{n-1}{2} \\ & & & & -\frac{n}{2} & n \end{bmatrix}.$$

Compute the LU decomposition of A_{10} using your function from part a. Report the resulting values $l_1, l_2, \ldots, l_n$ and $u_1, u_2, \ldots, u_{n-1}$ in a table.

Extra Credit (5 points): Find a general expression for the entries of l and u in the LU decomposition of A_n given above and *prove* that it is correct.

(c) Determine the number of additions/subtractions as well as the number of multiplications/divisions required for LU decomposing an n-by-n tridiagonal matrix with Crout's method.

5. Assuming no pivoting is needed (to avoid breakdown or to ensure numerical stability), devise an efficient way to arrange the computations for solving an n-by-n linear system with non-zero entries in the coefficient matrix only in the first and last rows and columns and also in the two main diagonals. In the case of a 9×9 matrix, the non-zero entries would appear as follows:

$$\begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & & & & & & \bullet & \bullet \\ \bullet & & \bullet & & & & \bullet & & \bullet \\ \bullet & & & \bullet & & \bullet & & & \bullet \\ \bullet & & & & \bullet & & & & \bullet \\ \bullet & & & \bullet & & \bullet & & & \bullet \\ \bullet & & \bullet & & & & \bullet & & \bullet \\ \bullet & \bullet & & & & & & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \end{bmatrix} \begin{bmatrix} \\ \\ \\ \\ x \\ \\ \\ \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \\ b \\ \\ \\ \\ \\ \end{bmatrix},$$

where $\bullet$'s represent a non-zero entries and blanks represent zero entries. It is sufficient to illustrate the efficient method for this 9×9 case. However, for the general n-by-n case, determine the number of operations (additions/subtractions and multiplications/divisions) your algorithm requires (these should both be $O(n)$).

6. Consider the linear system of equation $Ax = b$, where

$$A = \begin{bmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \\ 1/6 & 1/7 & 1/8 & 1/9 & 1/10 \end{bmatrix}, \qquad b = \begin{bmatrix} 0.882 \\ 0.744 \\ 0.618 \\ 0.521 \\ 0.447 \end{bmatrix}.$$

Suppose we in some way have obtained the approximate solution vector

$$\overline{x} = \begin{bmatrix} -2.1333 \\ 0.6258 \\ 17.4552 \\ -11.8692 \\ -1.4994 \end{bmatrix}.$$

It is then easy to show that the residual becomes *exactly*

$$A\overline{x} - b = \begin{bmatrix} 0.00001 \\ -0.00001 \\ 0.00001 \\ -0.00001 \\ 0.00001 \end{bmatrix}.$$

(a) Does this imply that $\overline{x}$ is close to the exact solution x?

(b) Use MATLAB to obtain an accurate solution to the given system.

(c) Use MATLAB again to obtain a condition number for A. Use the appropriate result on perturbations of the right hand side (RHS) of a linear system to confirm that this very small residual indeed is big enough to allow for the solution to be as far away from the correct one as occurs in this example.

Hint: The A matrix can be constructed easily in MATLAB using the function `hankel` (use MATLAB `help` to see why). The following code constructs A:

```
A = 1./hankel(2:6,6:10).
```

The vector b can be entered into MATLAB manually:

```
b = [0.882 0.744 0.618 0.521 0.447]'.
```

The linear system $Ax = b$ can be solved in MATLAB using the `mldivide` command or simply the backslash operator "\":

```
x = A\b
```

The condition number (with respect to the two-norm) can be computed using the MATLAB function `cond`:

```
cond(A)
```

7. [565 only] Recall that Cholesky's method can be used to factor a symmetric positive definite (s.p.d.) matrix A such that $A = LL^T$, where L is a lower triangular matrix. The disadvantage of this method is that it requires n square roots. It was mentioned in class that these square roots could be avoided by factoring A into the form $A = \tilde{L}D\tilde{L}^T$, where D is a diagonal matrix and $\tilde{L}$ is a lower triangular matrix with 1s on the diagonal. Following the same steps we used to derive Cholesky's method in class, derive a procedure for computing the entries in $\tilde{L}$ and D for this new $A = \tilde{L}D\tilde{L}^T$ decomposition.

8. [565 only] Let A, B, and δA be n-by-n matrices, prove the following results:

   (a) If $\rho(A) < 1$, then the matrix $I - A$ is nonsingular.

   Hint: Use a contradiction agrument.

   (b) If A is nonsingular and
   $$\|A - B\| < \frac{1}{\|A^{-1}\|},$$
   then B is nonsingular.

   Hint: Use the fact that $B = A(I - A^{-1}(A - B))$ and the result from part (a).

   (c) If A is nonsingular and $\|\delta A\| < 1/\|A^{-1}\|$, then $A + \delta A$ is nonsingular.