

This assignment is due at the **start** of the class on 7 December 2011.

For the questions that require you to write a MatLab program, hand in the program and its output as well as any written answers requested in the question. Your program should conform to the usual CS standards for comments, good programming style, etc. Try to format your output from your program so that it is easy for your TA to read your results.

1. [15 marks; 5 marks for each part]

Consider the linear system  $Ax = b$  where

$$A = \begin{pmatrix} \gamma & 1 \\ 1 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 2 - \gamma \\ 1 \end{pmatrix}$$

The solution of this system is

$$x = \begin{pmatrix} -1 \\ 2 \end{pmatrix}.$$

This question shows that, if  $|\gamma|$  is small, you need to pivot when solving such systems. It also shows that, even if the LU factorization is inaccurate, you can use this inaccurate LU factorization together with *iterative refinement* (see part (c) below) to compute an accurate solution to the linear system  $Ax = b$ .

- (a) If you don't pivot, the LU factorization of  $A$  is

$$L_1 = \begin{pmatrix} 1 & 0 \\ 1/\gamma & 1 \end{pmatrix} \quad U_1 = \begin{pmatrix} \gamma & 1 \\ 0 & 1 - 1/\gamma \end{pmatrix}$$

That is,  $A = L_1 U_1$ .

For  $\gamma = 10^{-2k}$ ,  $k = 1, 2, \dots, 10$ , use MatLab and this LU factorization to compute the solution of  $Ax = b$ . That is, use the MatLab backslash operator  $\backslash$  first to solve  $L_1 y = b$  for  $y$  in MatLab by  $y = L_1 \backslash b$  and then to solve  $U_1 x = y$  for  $x$  in MatLab by  $x = U_1 \backslash y$ .

How does the accuracy of the computed solution behave as the value of  $\gamma$  decreases?

- (b) If you do pivot, the LU factorization of  $A$  is

$$P_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad L_2 = \begin{pmatrix} 1 & 0 \\ \gamma & 1 \end{pmatrix} \quad U_2 = \begin{pmatrix} 1 & 1 \\ 0 & 1 - \gamma \end{pmatrix}$$

That is,  $P_2 A = L_2 U_2$ .

For  $\gamma = 10^{-2k}$ ,  $k = 1, 2, \dots, 10$ , use MatLab and this LU factorization to compute the solution of  $Ax = b$ . That is, first compute  $\tilde{b} = P_2 b$  in MatLab, next solve  $L_2 y = \tilde{b}$  for  $y$  in MatLab by  $y = L_2 \backslash \tilde{b}$  and then solve  $U_2 x = y$  for  $x$  in MatLab by  $x = U_2 \backslash y$ .

How does the accuracy of the computed solution behave as the value of  $\gamma$  decreases?

What do the solutions of parts (a) and (b) tell you about the importance of pivoting in solving linear systems?

- (c) Read section 2.4.10 of your textbook on “Improving Accuracy” using *iterative refinement*. Repeat part (a) using the LU factorization  $A = L_1U_1$  computed without pivoting, but this time use one iteration of iterative refinement to improve the solution. Compute the residual in MatLab as  $r = b - A\hat{x}$ , where  $\hat{x}$  is the computed solution from part (a).

In your implementation of iterative refinement, you must solve  $Ae = r$  for an approximation  $e$  to the error in your computed solution. Use MatLab and the inaccurate LU factorization  $A = L_1U_1$  to solve this system. That is, first solve  $L_1z = r$  for  $z$  in MatLab by  $z = L_1 \setminus r$  and then solve  $U_1e = z$  for  $e$  in MatLab by  $e = U_1 \setminus z$ . Then compute in MatLab the “better” approximate solution  $\tilde{x} = \hat{x} + e$ , where  $\hat{x}$  is the computed solution from part (a).

How does the accuracy of the “better” approximate solution  $\tilde{x}$  behave as the value of  $\gamma$  decreases?

What do the solutions of parts (a) and (c) tell you about the effectiveness of iterative refinement in this context?

2. [15 marks; 5 marks for each part]

Consider the matrix

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{pmatrix}$$

This is a famous example that shows that, in some cases, complete pivoting can be much more effective than row partial-pivoting in reducing the growth of elements during LU factorization.

- (a) Calculate by hand the matrices  $P$ ,  $L$  and  $U$  that you obtain if you compute the LU factorization of  $A$  with row partial-pivoting. (This is the factorization  $PA = LU$  that we’ve talked about extensively over the last few weeks.) In doing this factorization, you should never have to pivot. Hence, you should find that  $P = I$  in this case. So, the  $L$  and  $U$  that you calculate should satisfy  $A = LU$ .

Show all your calculations.

- (b) Calculate by hand the matrices  $P$ ,  $Q$ ,  $L$  and  $U$  that you obtain if you apply LU factorization with complete pivoting to the matrix  $A$  above. (This is the factorization  $PAQ = LU$  discussed on page 75 of your textbook.) In doing this factorization, at each stage  $k$ , just before you zero the elements below the main diagonal in column  $k$ , you should interchange columns  $k$  and 5. (Do this interchange for column 1 also, even though it is not really required there.) You should never interchange any rows. Therefore, the  $P$ ,  $Q$ ,  $L$  and  $U$  that you calculate should satisfy  $P = I$  and  $AQ = LU$ .

Show all your calculations.

(Note: the largest element in  $U$  should be smaller in this case than in part (a) above.)

- (c) To see the ill effects of the element growth that can occur when you perform an LU factorization with row partial-pivoting, we need to consider a larger example than the one above to allow the elements to grow more.

To this end, let  $n = 60$  and construct an  $n \times n$  matrix similar to  $A$  above with the MatLab commands

```

A = ones(n,n);
A = A - triu(A);
A = eye(n) - A;
A = A + [ones(n-1,1); 0] * [zeros(1,n-1),1];

```

and an  $n \times n$  matrix similar to  $Q$  from part (b) with the MatLab commands

```

Q = diag(ones(n-1,1),1);
Q(n,1) = 1;

```

You can compute the LU factorization with row partial-pivoting of this  $n \times n$  matrix  $A$  with the MatLab command

```
[L1, U1, P1] = lu(A)
```

(Read “help lu” in MatLab.) Print  $U1(n,n)$  and verify that  $U1(n,n) = 2^{n-1}$ . Thus we see an exponential growth in the elements of  $U1$  with respect to  $n$ , the size of the matrix, even though all the multipliers used in the LU factorization process are of magnitude 1 or less.

You can compute the LU factorization with complete pivoting for the same  $n \times n$  matrix  $A$  with the MatLab command

```
[L2, U2] = lu(A*Q)
```

(Read “help lu” in MatLab.) You can compute the largest element of  $U2$  with the MatLab command  $\max(\max(\text{abs}(U2)))$ . Verify that this value is 2, just as it was for the smaller version of the matrix  $A$  in part (b) above. Thus, the LU factorization with complete pivoting does not suffer from the exponential growth of elements that we saw above in the LU factorization with row partial-pivoting.

To see the ill effect of this exponential growth in the elements of  $U1$ , let  $x = \text{ones}(n,1)$  and  $b = Ax$ . Solve the system  $Ax = b$  using the matrices computed by the lu factorization with row partial-pivoting by executing the MatLab commands

```

y = L1 \ b
x1 = U1 \ y

```

(You don’t have to use  $P$  above because  $P = I$ .) Compute and print  $\text{norm}(x - x1, \text{inf})$ , where  $x$  is the exact solution of  $Ax = b$ .

Note that  $x1$  is a very poor approximation to  $x$ .

Also, solve the system  $Ax = b$  using the matrices computed by the lu factorization with complete pivoting by executing the MatLab commands

```

y = L2 \ b
z = U2 \ y
x2 = Q * z

```

(You don’t need a  $P$  matrix in this case because  $P = I$ .) Compute and print  $\text{norm}(x - x2, \text{inf})$ , where  $x$  is the exact solution of  $Ax = b$ .

Note that  $x2$  is a good approximation to  $x$ .

3. [20 marks: 5 marks for each part]

Consider the matrix

$$A = \begin{pmatrix} 1 & 1 & -1 \\ 4 & 0 & -4 \\ 2 & -2 & 0 \end{pmatrix}$$

- (a) Use partial pivoting to compute the LU factorization of  $A$ . That is, compute the  $3 \times 3$  permutation matrix  $P$ , the  $3 \times 3$  unit-lower-triangular matrix  $L$  with  $|L_{ij}| \leq 1$  for  $i > j$ , and the  $3 \times 3$  upper-triangular matrix  $U$  such that  $PA = LU$ .

Show all your calculations.

- (b) Use the LU factorization of  $A$  computed in part (a) to solve the linear system  $Ax = b$ , where

$$b = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix}$$

Show all your calculations.

- (c) Suppose we change the (3,1) element of  $A$  from 2 to 1 to yield a new matrix

$$\hat{A} = \begin{pmatrix} 1 & 1 & -1 \\ 4 & 0 & -4 \\ 1 & -2 & 0 \end{pmatrix}$$

Note that all the elements of  $A$  and  $\hat{A}$  are equal except for the (3,1) element.

Find two vectors  $u$  and  $v$  such the  $\hat{A} = A - uv^T$ . Thus,  $A$  and  $\hat{A}$  differ by a rank 1 update.

- (d) Read section 2.4.9 of your textbook and then use the Sherman-Morrison formula

$$(A - uv^T)^{-1} = A^{-1} + \frac{A^{-1}uv^T A^{-1}}{1 - v^T A^{-1}u} \quad (1)$$

to solve  $\hat{A}\hat{x} = b$ , where  $\hat{A}$  is the matrix in part (c) and  $b$  is the vector in part (b).

Do not compute any inverses in the Sherman-Morrison formula (1) explicitly. Instead, use the LU factorization from part (a) whenever you need to solve a linear system with the matrix  $A$ .

In addition, organize your computation so that there are no matrix-matrix multiplies or solves. That is, if the vectors were  $n$ -vectors and the matrices were  $n \times n$  matrices, you should be able to solve the system  $\hat{A}\hat{x} = b$  in time proportional to  $n^2$ , not time proportional to  $n^3$ .

Show all your calculations.

4. [10 marks: 5 marks for each part]

- (a) Do question 2.3 on page 100 of your textbook. Use the MatLab backslash operator  $\backslash$  to solve the linear system of equations  $Ax = b$  that arises in this problem.

Hint: read “help mldivide” in MatLab.

Suggestion: you might find it easiest to hard-code the matrix and right side vector into your MatLab program. If you choose this option, you may want to start by using the MatLab function “zeros” to construct a matrix and a vector with all elements equal to zero and then change the appropriate elements to their nonzero values.

- (b) Let  $\hat{x}$  be the computed solution to the linear system  $Ax = b$  from part (a). Use the error bound at the top of page 60 of your textbook to bound the relative error,  $\|\hat{x} - x\|/\|x\|$ , associated with the computed solution  $\hat{x}$  in terms of the condition number of the matrix  $A$  associated with the system  $Ax = b$  and the relative residual,  $\|r\|/\|b\|$ , where  $r = b - A\hat{x}$  is the residual associated with the computed solution  $\hat{x}$ .

Hint: read “help cond” in MatLab and make sure that the norm associated with the condition number that you use agrees with the vector norm that you use for the relative error and the relative residual.

Hand in your program and its output.

5. [10 marks]

Do question 2.6 on page 101 of your textbook.

The book says to use a library routine for Gaussian elimination to solve the problem. Instead you should use MatLab to solve this problem. In particular, use the MatLab backslash operator  $\backslash$  to solve the linear system of equations  $Hx = b$ , where  $H$  is the Hilbert matrix.

Read “help mldivide” and “help hilb” in MatLab.

Use the MatLab function “cond” for the condition number estimator.

Read “help cond” in MatLab.

Since you are asked to use the  $\infty$ -norm for the vectors in this question, make sure that you use the condition number estimator associated with the  $\infty$ -norm also.

The book asks, “How large can you take  $n$  before the error is 100 percent?” This is not very well-defined. So, instead, determine how large can you take  $n$  before the relative error in the solution,  $\|x - \hat{x}\|/\|x\|$ , is greater than or equal to 1?

Hand in your program and its output.