MTH 308: Numerical Analysis & Scientific Computing I
Department of Mathematics and Statistics
Indian Institute of Technology - Kanpur January 26, 2017

Problem Set 2

The following problem is due **Friday, February 10** at **5:00 PM**.
Steps you should follow :

1. While submitting, all of your programs should be arrange as follows: main directory should be called HW2-ID where ID stands for your roll number. For example, if your roll number is 12345, then the directory name will be HW2-12345; if your roll number is 123456, then the directory name will be HW2-123456.

2. The directory should contain five subdirectories named Problem1, Problem2, Problem3, Problem4 and Problem5. The codes corresponding to Problem J should go in the subdirectory ProblemJ. If Problem J does not require any programs to be written, then the corresponding subdirectory ProblemJ remains empty.

3. Upon completion of the work, once you are ready for submission, go to the directory that contains the directory HW2-ID and compress it using following command at the xterm/terminal command prompt

```
$ tar cvfz HW2-ID.tgz HW2-ID
```

   which will create a file called HW2-ID.tgz .

4. Attach this file in an email with subject HW2-ID and send it to akasha@iitk.ac.in by 5.00 pm on February 10. The supporting writeup should also be submitted by 5:00 pm in hard-copy in Akash Anand's mail-box in the Department of Mathematics and Statistics. Late submission will not get any credit.

Full credit will be given only to the correct solution which is described clearly. Convoluted and obtuse descriptions might receive low marks, even when they are correct. Aim for concise solution and focus on the key idea of the problem.

I. The solution to $Ax = b$ may be written $b = A^{-1}x$. This can be a good way to analyze algorithms involving linear systems. But we try to avoid forming A^{-1} explicitly in computations because it is more than twice as expensive as solving the linear equations. A good way to form $B = A^{-1}$ is to solve the matrix equation $AB = I$. Gauss elimination applied to A gives $A = LU$, where the entries of L are the pivots used in elimination.

(a) Show that about $n^3/3$ work reduces $AB = I$ to $UB = L^{-1}$, where the entries of U and L^{-1} are known.

(b) Show that computing the entries of B from $UB = L^{-1}$ takes about $n^3/2$ work. Hint: It takes one floating point operation (flop) per element for each of the n elements of the bottom row of B, then two flops per element of the $n-1$ row of B, and so on to the top. The total is $n(1 + 2 + \cdots + n)$.

(c) Use this to verify the claim that computing A^{-1} is more than twice as expensive as solving $Ax = b$.

II. Write a program for estimating the condition number of a matrix A. You may use either the 1-norm or the ∞-norm (or try both and compare the results). You will need to compute $\|A\|$, which is easy, and estimate $\|A^{-1}\|$, which is more challenging. From the properties of the norms, we know that if z is the solution to $Az = y$, then

$$\|z\| = \|A^{-1}y\| \leq \|A^{-1}\|\|y\|,$$

so that

$$\frac{\|z\|}{\|y\|} \leq \|A^{-1}\|,$$

and this bound is achieved for some optimally chosen y. Thus, if we choose a y such that the ratio $\|z\|/\|y\|$ is as large as possible, then we will have a reasonable estimate for $\|A^{-1}\|$. Try two different approaches to choosing y:

(a) Choose y as the solution to the system $A^T y = c$ where c is a vector each of whose components is ± 1, with the sign of each component chosen by the following heuristic. Using the factorization $A = LU$, the system $A^T y = c$ is solved in two stages, successively solving $U^T v = c$ and $L^T y = v$. At each step of the first triangular solution, choose the corresponding component of c to be 1 or -1 depending on which will make the resulting component of v larger in magnitude. Then solve the second triangular system in the usual way for y.

(b) Choose a small number, say five, different vectors y randomly and use the one producing the largest ratio $\|z\|/\|y\|$.

Use both of the approaches on each of the following matrices:

$$A_1 = \begin{bmatrix} 10 & -7 & 0 \\ -3 & 2 & 6 \\ 5 & -1 & 5 \end{bmatrix}, \qquad A_2 = \begin{bmatrix} -73 & 78 & 24 \\ 92 & 66 & 25 \\ -80 & 37 & 10 \end{bmatrix}.$$

How do the results using these two methods compare? To check the quality of your estimates, compute A^{-1} explicitly to to determine its true norm (this computation can also make use of the LU factorization already computed). How does your condition numbers (the that uses estimated $\|A^{-1}\|$ as well as the one with exact $\|A^{-1}\|$) compares with the condition number given by Matlab.

III. Consider the linear system

$$\begin{bmatrix} 1 & 1+\epsilon \\ 1-\epsilon & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1+(1+\epsilon)\epsilon \\ 1 \end{bmatrix}$$

where ϵ is a small parameter to be specified. The exact solution is obviously

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ \epsilon \end{bmatrix}$$

for any value of ϵ.

Use Gaussian elimination to solve the linear system. For $\epsilon = 1.0 \times 10^{-n}, n = 0, 2, 4, 6, 8$, compute an estimate of the condition number of the matrix (using one or both methods given in Problem II) and the relative error in each component of the solution. How accurately is each component computed? How does the accuracy attained for each component compare with expectations based on the condition number of the matrix. What conclusions can you draw from this experiment?

IV. An $n \times n$ Hilbert matrix H has entries $h_{ij} = 1/(i+j-1)$ so it has the form

$$\begin{bmatrix} 1 & 1/2 & 1/3 & \cdots \\ 1/2 & 1/3 & 1/4 & \cdots \\ 1/3 & 1/4 & 1/5 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}.$$

Fon $n = 2, 3, \ldots$, generate the Hilbert matrix of order n and also generate the n-vector $b = Hx$, where x is the n-vector with all its components equal to 1. Use Gaussian elimination to solve the resulting linear system $Hx = b$, obtaining the approximate solution $\hat{x}$. Compute the ∞-norm of the residual $r = b - H\hat{x}$ and of the error $\Delta x = \hat{x} - x$, where x is the vector of all ones.

(a) How large can you take n before the error is 100%.

(b) Use a condition number estimator to obtain cond(H) for each value of n.

(c) As n varies, how does the number of correct digits in the components of the computed solution relate to the condition number of the matrix?

V. The determinant of a triangular matrix is equal to the product of its diagonal entries. Use this fact to develop a program for computing the determinant of an arbitrary $n \times n$ matrix by using its LU factorization. How can you determine the proper sign for the determinant? To avoid the risk of underflow and overflow, you may wish to consider computing the logarithm of the determinant instead of the actual value of the determinant.