Lab #6

Computational Physics I (**Phys381**)
R. Ouyed
Due April 8, 2014 (at the end of class)

[Total: 100 marks including 20 marks for the report]

- The latexed report is worth **20%** of the total mark. Only well documented and neatly presented reports will be worth that much! It is not sufficient to give only numerical results and show plots, you should also discuss your results. Include complete figure captions, introduction and conclusion sections.
- Your report must be in a *two-column format.*
- Your report should include your *Fortran code* and *Gnuplot scripts* in an Appendix using the *verbatim* command.
- If applicable, animations should be shown to the teacher and/or TA before handing in the lab report. The animation should be well documented and contains necessary information (student name, assignment number, run time etc ...). Basically, the information should be included in each frame before they are put together.
- You must name your report using the names (last names only): *student1-student2-phys381-lab#.pdf.*
- **Procedure for Handing in your lab report (see instructions on phys381 wbsite):**

  1) Set permission to your PDF report as 644. It means:
  chmod 644 *student1-student2-phys381-lab#.pdf*
  2) cp -a *student1-student2-phys381-lab#.pdf* **/home/ambrish/phys381/labs/lab#**
  3) Copy a second time to ensure that your exam copied correctly. If you are prompted as to whether or not you would like to replace the existing file, then you report has been successfully submitted.

- **You must check with your TAs (Ambrish or Zach) that your report was received and is readable BEFORE you leave the lab.**

→ I encourage you to make use of the routines in section 2.3 in the **Numerical Recipes (NR)** textbook. These are the most robust (and best tested) routines in the market.

→ The **Numerical Recipes (NR)** can be accessed via the Phys381 website or directly via: *http://apps.nrbook.com/fortran/index.html*.

→ The **Numerical Recipes** (see pages 38-41) provide you with the following routines to deal with any $Ax = b$ system where A is an $N \times N$ matrix:

(i) An LU decomposition code using the Crout algorithm which takes A as input and outputs L and U.

(ii) A forward substitution and a backward substitution code which takes L, U and b as input and outputs the solution x of the $Ax = b$ system.

(iii) A matrix Inversion code using LU decomposition which takes A as input and outputs its inverse A^{-1}.

(iv) A code that calculates the determinant of a matrix A using LU decomposition.

1 Using LU-decomposition to find the Inverse of a Matrix [30 marks]

Let us define $[B]_{N\times N}$ as the inverse matrix of an $N \times N$ square matrix $[A]_{N\times N}$. It means $[B]_{N\times N} = [A^{-1}]_{N\times N}$.

- *[10 marks]* Starting with $[A][B] = [I]$ where I is the identity matrix, write down an algorithm (it means a scheme) that solves for the coefficients b_{ij} of $[B]$. *Hint: Start with* $[A][B] = [I] \rightarrow [L][U][B] = [I] \rightarrow [L][Z] = [I]$ *with* $[U][B] = [Z]$.

- *[10 marks]* Write your own Fortran 90 code (or use the one provided in **NR**) that implements your algorithm above.

- *[10 marks]* Apply your code to an 5×5 matrix with entries defined by

$$A(i,j) = \frac{1}{i+j-1} \ , \qquad (1)$$

and check that the corresponding inverse matrix is:

$$A^{-1} = \begin{pmatrix} 25 & -300 & 1050 & -1400 & 630 \\ -300 & 4800 & -18900 & 26880 & -12600 \\ 1050 & -18900 & 79380 & -117600 & 56700 \\ -1400 & 26880 & -117600 & 179200 & -88200 \\ 630 & -12600 & 56700 & -88200 & 44100 \end{pmatrix} \quad \text{With } A = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{pmatrix}$$

2 LU Decomposition and solutions of $Ax = b$ systems [50 marks]

Consider the linear system $Ax = b$ where[1]

$$A = \begin{pmatrix} 10^{-2k} & 1 \\ 1 & 1 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} 1 + 10^{-2k} \\ 2 \end{pmatrix} \tag{2}$$

The solution of this system is

$$x = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \tag{3}$$

- *[5 marks]* What is the LU factorization of matrix A=LU without pivoting? Meaning, give L and U matrices in terms of k (you do not need the code for this part).
- *[20 marks]* Write your own Fortran 90 code (or use the one provided in **NR**) that solves for x using the LU decomposition. Your code should have 3 parts:

  a) The first performs the **Clouts algorithm** (see section 2.7.3 in Ouyed&Dobler)[2]

  b) The second performs a forward substitution;

  c) The third part performs the backward substitution.
- *[5 marks]* Solve for x for different values of k; $k = 0, 1, 2, ..., 10$. How does the accuracy of the computed solution behave as the values of k increases?

2.1 Iterative refinement [20 marks]

Iterative refinement is often necessary to improve accuracy of your solution. If $\tilde{x}$ is the computed solution from the LU algorithm (with no pivoting) and x the exact solution, the error is defined as $e = (x - \tilde{x})$. Refinement consists in reducing the error to a desired precision ϵ (i.e. iterate until $e \leq \epsilon$).

i) *[10 marks]* Following **Sections 2.4 and 2.7.2** in Ouyed&Dobler textbook add an iterative refinement part to your code.

ii) *[5 marks]* Plot a 3D figure showing $e = (x - \tilde{x})$ versus k and N (the number of iterations) for a desired precision ϵ. Comment on the figure and discuss how the iterative refinement procedure performs as k increases.

iii) *[5 marks]* What does this laboratory tell you about the effectiveness of iterative refinement in LU decomposition?

[1]**Hint:** See section 2.7.4 in Ouyed&Dobler on how to write matrices in Latex. You will need to call the "\usepackage{amsmath}" package.

[2]There is a typo in the last equation in section 2.7.3. One should divide by U_{jj} instead of multiplying by U_{jj}.