Cornell University
P4480/P7680/A7690

September 5, 2019
Due: September 12, 2019

Homework #1

Reminder: On every homework you turn in, be sure to include the computer code you used to generate your solutions. Put the answers first, then all the code next, in a single well-organized document.

Problem 1

Consider the system of equations

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b}\,,$$

where

$$\mathbf{A} = \begin{pmatrix} 0.7073725 & 0.5204556 \\ 0.8158208 & 0.6002474 \end{pmatrix}$$

and

$$\mathbf{b} = \begin{pmatrix} 0.1869169 \\ 0.2155734 \end{pmatrix}\,.$$

The exact solution is

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

Consider two approximate solutions

$$\mathbf{x}_1 = \begin{pmatrix} 0.9999999 \\ -1.0000001 \end{pmatrix}\,, \qquad \mathbf{x}_2 = \begin{pmatrix} 0.4073666 \\ -0.1945277 \end{pmatrix}\,.$$

- {**a**} Compute the residuals $\mathbf{r}_1$ and $\mathbf{r}_2$ corresponding to these two approximate solutions. (The residual is $\mathbf{r} = \mathbf{b} - \mathbf{A} \cdot \mathbf{x}$.) Does the more accurate solution have the smaller residual?
- {**b**} Use the LU decomposition of $\mathbf{A}$ to solve for $\mathbf{x}$. Compute the solution's error, $\mathbf{x}_{\text{numerical}} - \mathbf{x}$, and residual $\mathbf{r} = \mathbf{b} - \mathbf{A} \cdot \mathbf{x}_{\text{numerical}}$. Do you get the full machine accuracy in the solution? How does the residual's size compare to the solution error?
- {**c**} Compute the condition number of $\mathbf{A}$. How does this help you understand the results of (a) and (b)?

Problem 2

Numerical Recipes tells us that multiplication of a vector by a matrix inverse is inferior to alternative methods of solving a set of linear equations. Let's see if this is so, and also compare this approach to LU decomposition and iterative improvement.

Start by downloading from Blackboard the random 1600-by-1600 matrix, $\mathbf{A}$, random vector $\mathbf{c}$ and vector $\mathbf{b} = \mathbf{A} \cdot \mathbf{c}$ from the files `A.txt`, `b.txt`, and `c.txt`.

NOTE: Don't click on the links to the files to open them in your browser and then copy and paste the files to your local computer. The file `A.txt` is large and this process will be slow. Instead, right-click on the link for the file and select Save Link As (or whatever option your browser offers for saving a file). The right-hand side $\mathbf{b}$ has been computed with extended precision (i.e., greater than double) to reduce the accumulation of round-off errors. Thus, we expect $\mathbf{b}$ to carry full double precision accuracy despite resulting from many arithmetic operations.

{a} Find $\mathbf{A}^{-1}$, solve for $\mathbf{x}$ by $\mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b}$. Write down as a measure of the error

$$r = \sum_{j=1}^{1600} |\mathbf{x}(j) - \mathbf{c}(j)|$$

NOTE: Using *Numerical Recipes* code, find the inverse with `gaussj`. Using Python, find the inverse with `linalg.inv`.

{b} Solve $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ for $\mathbf{x}$ using LU decomposition and compute r.

{c} Use iterative improvement to improve your solution's accuracy. Please report your residual $\mathbf{r}$.

HINT: The *Numerical Recipes* code uses long double precision to represent $\mathbf{b}$, $\mathbf{A}$ and $\mathbf{x}$ during the refinement process. If you are not using *Numerical Recipes* code be sure to do the same! As an example, in Python, your code to compute the residual part might look something like

```
A_LF     = np.array(A,dtype=np.longdouble)
x_LF     = np.array(x,dtype=np.longdouble)
b_LF     = np.array(b,dtype=np.longdouble)
r_LF     = b_LF - np.dot(A_LF,x_LF)
```

{d} Is the difference in the error between LU and inverse methods roughly consistent with what you would expect based on the difference in the the number of floating-point operations required of each algorithm? Is the accuracy of the solution roughly consistent with what would expect given the conditioning of the matrix?