Due Date: Monday, 21 January, 2019

- Please follow the *Guidelines for Computing Assignments* found on Canvas (under FAQ).
- The computing report grading scheme is also on the FAQ.
- Your one-page report is to be submitted via Crowdmark by 11:59pm on Monday, 21 January.
- Acknowledge any and all collaborations and assitance from classmates/TAs/instructor.

Finite-Precision Errors in GE for Large Linear Systems

Begin by downloading the in-class demo *w02w_GEerr.m* from the Canvas page. Recall, from lecture, that this demo produced the distribution of errors resulting from the Gaussian Elimination (GE) algorithm with finite-precision arithmetic. The goal of this exercise is to understand the statistics of the growth of truncation error as matrix size N increases.

As presented in the notes, a brief summary of the experimental method is:

- produce a random matrix $[A]$,
- construct the right-side vector $\vec{b} = [A]\vec{x}_0$, where $\vec{x}_0$ is the N-component vector of all 1's (using the Matlab *ones* command),
- use Matlab's backslash to get the GE solution $\vec{x}_1 = [A] \backslash \vec{b}$,
- compute the finite-precision solution error, $\text{sol\_err} = \text{rms}(\vec{x}_1 - \vec{x}_0)$,
- compute the finite-precision *residual error*, $\text{res\_err} = \text{rms}([A]\vec{x}_1 - \vec{b}\,)$,
- gather data for statistical analysis.

The *rms* command is Matlab is the *Root-Mean-Squared* (RMS) function, which converts the vector of errors to a non-negative scalar number

$$\text{rms}(\vec{x}) = \frac{|\vec{x}|}{N}$$

where $|\vec{x}|$ is the vector magnitude, which is then divided by the number of components of $\vec{x}$ — so it gives a measure of the error per unknown (so we don't grow the error when comparing to ever larger N values.) Furthermore, by considering the $\log_{10}$ of the RMS error, it is basically converted into a number that roughly indicates the first corrupted decimal place (as in its power of 10).

In exact arithmetic, we should expect the errors to be 0. However as seem in lecture, in the world of floating-point arithmetic, even small matrix-size results in measurable non-zero errors. You will learn in this exercise that, as N increases, so do the GE errors — and the goal of this assignment is **to produce a rough estimate for the matrix size N^* at which the error resulting from GE becomes as large as the solution itself**.

Of course, it is not possible to do computations for matrices of size N^*. Therefore, we will have to do computations for a number of smaller N values, and use these to obtain a rough, but logical, estimate for the value of N^*.

Since the error computed for each GE depends on the random matrix A, we can determine what the 'typical' error for matrix size N is by averaging over many experiments (different random matrices). If we run N_{ex} number of experiments, and call $\text{res\_err}^{(k)}$ the errors obtained from the k^{th} experiment, we can then define the average $\log_{10}$ error as

$$\varepsilon_{res}(N) = \frac{1}{N_{ex}} \sum_{k=1}^{N_{ex}} \log_{10}(\text{res\_err}^{(k)}) = \text{mean}_{k=1\to N_{ex}} \left\{ \log_{10}(\text{res\_err}^{(k)}) \right\} .$$

We will take this error to be representative of the typical GE error for matrices of size N. Once you have several of these $\varepsilon_{res}(N)$ values, show them in a plot which you then base your estimate for N^*.

Your one-page computing report should include/address the following:

(a) Obtain values $\varepsilon_{res}(N)$ for several different values of N. But to do this, YOU will need to decide on the values for N and N_{exp} to be used. All of these choices will affect the quality of your estimate. Clearly indicate which values you selected and discuss how you made your choices.

(b) Include a plot of the points $(\log_{10} N, \varepsilon_{res}(N))$.

(c) Use your plot from (b) to argue for your value of an estimated value N^* where $\varepsilon_{res}(N^*) \approx 0$. You only need to present a rough order of magnitude here, and you should indicate how achievable, or not, your value of N^* is.