Aerospace Computing

Assignment 2

Completion Goal: 1/26/24

Do a google search to look for a Python timing function. One possibility comes with the basic Python distribution in a module called time. The call time.perf_counter() sets a start point and is used with a second time.perf_counter() call. If set to a variable you can take the difference and determine how long a particular section of code takes to execute.

1. Use the text's code for gaussPivot and LUPivot to write Python scripts for each. Make them functions that you can apply later. Include them in a single module for ease of use later.
2. Use the notes description of Cramer's rule to write a Python function.
3. For text Problem Set 2.2, problem 11, with the righthand side ($\vec{b}$) multiplied by 2.
    a. Solve with all 3 direct methods.
    b. Verify solution accuracy. (Compute the residual vector, $\vec{r} = A\vec{x} - \vec{b}$ – it should be zero or near zero.)
    c. Use the timing function to compare the performance of each method (make a table and insert it into your Jupyter Notebook)
4. Set up nonsingular 3-, 7-, 9-, and 11-equation systems
    a. Verify the accuracy of each method (see b above).
    b. Assess the performance of each method by plotting time versus matrix size using matplotlib.
5. Using LUPivot, create a code that computes the inverse of matrix [A].
    a. Test your code using both matrices from text problem 9 from Problem Set 2.3.
    b. Use numpy's dot(A,B) function to verify that $[A][A]^{-1}=[I]$.