MTH 451-551, Fall 2017, Assignment 1. Each problem is worth 5 points.

Instructions: *Please write neatly using proper mathematical notation, and show all the details necessary. As for code, you can show the snippets of the code, but please do not just print things from your MATLAB screen.*
Your paper should present what you have learned and concluded from the exercises, rather than a bunch of graphs stapled together.
You can discuss your problems with others, but all the solutions have to be written up individually. Please work independently on the computational part. 451 students can always solve the part for 551, if they so prefer.
Extra credit is turned in on separate paper.

1. Solve 3.3 (451 solve two, 551 solve all). In addition, apply Hölder inequality (Lecture 3) to estimate $|x^*y|$ using $\| x \|_{17}$, and knowing that $y = e_1 + e_3$.

2. Write out examples of at least five different 2×2 unitary matrices, of which at least two have nontrivial complex parts.
(451) What can you say about the eigenvalues for these 2×2 matrices?
(551) Answer the general problem formulated as Problem 2.4.

3. (Computational) (a) What is the largest vector that you can work with on your computer in MATLAB: What is the largest matrix ? (Try different n and observe for which n MATLAB starts screaming at you, or for which n you have to wait an inordinate amount of time.) You can follow these examples:
`n=10; x=rand(n,1); n=10; A = rand(n,n);`
(b) For which n is it feasible to solve $Ax = e_1$, obtain A^{-1}, and obtain all the eigenvalues of A? (Waiting an hour does not qualify as feasible in this class)
(c) Gather up the results in a table that shows how long it took to do (b) for different n (clearly you want to try more than two different n, preferably differing by an order of magnitude). (Time comparison is done with `tic;...toc`). Estimate the computational complexity of each of these operations. (*We will discuss the estimate of the cost in class*).

4. (Computational) Write code that calculates the dot product $x^T y$ using a loop, and code that calculates the SAXPY product $x + \alpha y$ using a loop. Compare its efficiency with the vectorized MATLAB code.

Extra: repeat 4 in PYTHON and NUMPY. Compare the timings to those in MATLAB. (Here you may want to use the actual wall clock time observed with your own watch.)