# MSSC 6040 - Homework 4

Instructor: Greg Ongie

Fall 2022

**Problem 1** (5 pts)**.** Let $S$ be any subspace of $\mathbb{C}^m$ and let $P \in \mathbb{C}^{m \times m}$ be the orthogonal projector onto $S$. Given any vector $x \in \mathbb{C}^m$, prove that $Px$ is the vector in $S$ closest to $x$ in Euclidean distance, i.e., prove that the minimizer of

$$\min_{y \in S} \|x - y\|_2^2$$

is $y = Px$. (Hint: Use the identity $x - y = P(x - y) + (I - P)(x - y)$, and recall the Pythagorean Theorem: if $u$ and $v$ are orthogonal vectors then $\|u + v\|_2^2 = \|u\|_2^2 + \|v\|_2^2$.)

**Problem 2** (5 pts)**.** T-B: 6.4

**Problem 3** (5 pts)**.** T-B: 7.1

**Problem 4** (5 pts, MATLAB)**.** Write two MATLAB functions, one that implements the Classical Gram-Schmidt algorithm (Algorithm 7.1) and one that implements the Modified Gram-Schmidt algorithm (Algorithm 8.1), to compute reduced QR decomposition $A = \hat{Q}\hat{R}$ of any $m \times n$ matrix $A$, with $m \geq n$. Run your functions on the matrix:

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1.0001 & 1 & 1 \\ 1.0001 & 1.0001 & 1 \end{bmatrix}.$$

Then run MATLAB's internal QR Factorization algorithm with the command `[Q, R] = qr(A)`. Compare the output of the three approaches, using the command `format long` to see more decimal places. In particular, for the $\hat{Q}$ obtained by each approach, check how close $\hat{Q}$ is to being unitary by displaying the entries of $\hat{Q}^*\hat{Q}$. Discuss what you observe. Include a printout/screenshot of your code and the requested output in your writeup.

**Problem 5** (5 pts, MATLAB). This problem extends "Experiment 1" on pg. 64 of T-B.

(a) First, read through the experiment and run the two indicated code blocks. Include the generated plot in your write-up.

(b) Approximate the function $f(x) = \cos(\pi x)$ on the interval $[-1, 1]$ as a linear combination of the first four Legendre polynomials by projecting the vector `y = cos(pi*x);` onto the range of `A` (Hint: the matrix `Q` defined in the first code block might be helpful for this). Plot both $f(x)$ and its approximation by Legendre polynomials on the same graph. Repeat this experiment for the first six Legendre polynomials, and again for the first eight Legendre polynomials.

(c) Redo part (b) for the "Heaviside function" given by

$$h(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

In MATLAB, you can use the built-in function `heaviside(x)`. What do you observe in terms of the ability of Legendre polynomials to approximate $h(x)$?