Math 6601 – Programming assignment 1

Assigned on 09/14/2018, Due: 09/28/2018

Name:

You could use MATLAB, C/C++, FORTRAN or any other programming language to write your code. Turn in your results and a copy of your code in the class. Also, submit a copy of your code (in a zipped file) through Carmen.

1: (20 points) Implement both the classical and modified Gram-Schmidt procedures. Use each to generate an orthogonal matrix Q whose columns form an orthonormal basis for the column space of the Hilbert matrix $H \in \mathcal{R}^{n\times n}$,for $n = 2, \cdots, 12$. The Hilbert matrix has entries $h_{ij} = 1/(i + j - 1)$. For example, a 2×2 Hilbert matrix has entries

$$\begin{bmatrix} 1 & 1/2 \\ 1/2 & 1/3 \end{bmatrix}.$$

In addition, try to apply the CGS procedure twice (i.e., apply your CGS routine to its own output Q to obtain a new Q), and treat this as the third method. As a measure of the quality of the results, specifically, the potential loss of orthogonality, please plot the quantity $-\log_{10}(\|I - Q^T Q\|_F)$, which can be interpreted as "digits of accuracy", for each of the three methods as a function of n. How do the three methods compare in speed, storage, and accuracy?

2: (20 points) (a): Implement the Householder QR factorization (Algorithm 10.1), the implicit calculation of product $\hat{Q}^* b$ (by modifying Algorithm 10.2), and back substitution (Algorithm 17.1) by writing your own codes.

(b) Use these algorithms as building blocks for Algorithm 11.2 to solve the following least squares problem arising from polynomial fitting: fitting a polynomial of degree $n - 1$,

$$p_{n-1}(t) = x_0 + x_1 t + x_2 t^2 + x_3 t^3 + \cdots + x_{n-1} t^{n-1},$$

to m data points (t_i, s_i), $m > n$. Let $t_i = (i - 1)/(m - 1)$, $i = 1, \cdots, m$, so that the data points are equally spaced on the interval $[0, 1]$. The corresponding values s_i can be generated by first fixing values for the x_j, for example we can pick $x_j = 1$, $j = 0, \cdots, n-1$, and then evaluating the resulting polynomial to obtain $s_i = p_{n-1}(t_i)$, $i = 1, \cdots, m$. First, reformulate this problem as a least square problem for $Ax = b$ by introducing A and b.

Our objective is to see whether we can recover the x_j that are used to generate s_i, and measure the error as the difference between the computed x_j and the exact x_j. Choose $n = 4, 6, 8, \cdots, 24$, and $m = 2n$, plot the error of $x = (x_0, \cdots, x_{n-1})$ in 2-norm. What do you observe?