This report is to includes answers for the questions that are asked in Assignment2.mlx

Q1

(a) This proof is based on lecture to show $\mathrm{Cond}(A^TA) = \mathrm{Cond}(A)^2$

First, we know $\mathrm{Cond}(A) = \|A\|\|A^{-1}\| = \frac{\sigma_1}{\sigma_n}$

where σ_1 is the largest sigular value, and σ_n is the smallest sigular value.

Next, $A = U\Sigma V^t$

where U and V are both orthonormal, and Σ is $\begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix}$

Therefore, $A^T = V\Sigma U^t$

$A^tA = U\Sigma(V^tV)\Sigma U^t = U\Sigma^2 U^t$

where U is still unitary, and Σ^2 is $\begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_n^2 \end{bmatrix}$

So, $\mathrm{Cond}(A^tA) = \frac{\sigma_1^2}{\sigma_n^2}$

$= \left(\frac{\sigma_1}{\sigma_n}\right)^2$

$= \mathrm{Cond}(A)^2$

(b)

Since U and Q are both orthonormal, therefore:

$$U^TU = I \quad \text{and} \quad Q^TQ = I$$

So, $(UQ)^T(UQ) = (Q^TU^T)(UQ)$

$= Q^T(U^TU)Q$

$= Q^TQ$

$= I$

Therefore, the product of U and Q is still orthonormal.

(c) $A = U\Sigma V^t$ and $\Sigma = \begin{bmatrix} \sigma_1 & \sigma_2 & \cdots & \sigma_n \end{bmatrix}$ and $k(A) = \frac{\sigma_1}{\sigma_n}$

Therefore, $QA = Q(A) = Q(U\Sigma V^t)$

$$= (QU)\Sigma V^t$$

So, we know $\frac{\sigma_1}{\sigma_n}$ is not changed for QA

$\Rightarrow$ QA and A will have the same $\frac{\sigma_1}{\sigma_n}$, so $cond(A) = cond(QA)$.

Q3

(b)

In this part, we basically want to fit in the function $y(x) = ax+b$ to all the 300 data points such that the error is minimized. As we can roughly see, in Figure 1, the distance between the fitting line and the data points is minimized. In our algorithm, we have found a and b such that the error becomes smallest possible by solving the following matrix with cholesky and QR decomposition.

x	x_0	x_1	$\cdots$	x_{300}
y	y_0	y_1	$\cdots$	y_{300}

$$\underbrace{\begin{bmatrix} 1 & x_0 \\ 1 & x_1 \\ \vdots & \vdots \\ 1 & x_{300} \end{bmatrix}}_{Mp} \underbrace{\begin{bmatrix} b \\ a \end{bmatrix}}_{\substack{x \\ \uparrow \\ \text{unknown}}} = \underbrace{\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{350} \end{bmatrix}}_{y}$$

$$x = (Mp^T Mp)^{-1} Mp^{-1} y$$

$$\downarrow$$

$$\begin{bmatrix} b \\ a \end{bmatrix}$$

In Figure 1, the line is a straight line because degree is 1 and $y(x) = ax+b$. After a and b are found, the straight line is placed in where $r = Ax-b$ is minimized.

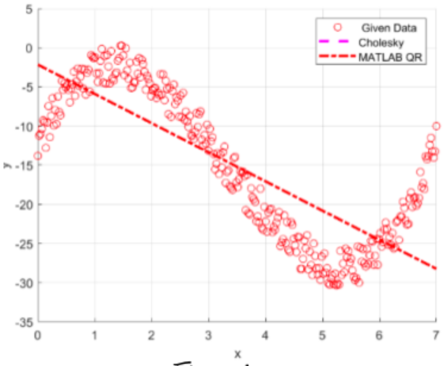


Figure 1

(c)

Similar to (b), but the degree is 14 now so the coefficients increase from 2 to 15. Since 15 is still far less than 300 data points, the line will not be overfitting to pass all the points. It is still trying to minimized the error (residual $r = Ax - b$). Since the degree is 15, the line can relatively better represent the data set than degree of 1 does.

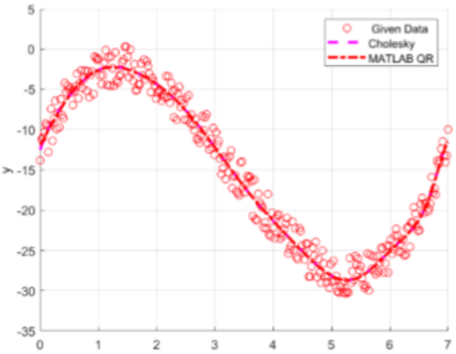


Figure 2.

$$\underbrace{\begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^{14} \\ 1 & x_1 & x_1^2 & \cdots & x_1^{14} \\ \vdots & & & & \\ 1 & x_{300} & x_{300}^2 & \cdots & x_{300}^{14} \end{bmatrix}}_{M_p} \underbrace{\begin{bmatrix} a \\ b \\ \vdots \end{bmatrix}_{14\times 1}}_{x} = \underbrace{\begin{bmatrix} y_0 \\ \vdots \\ y_{300} \end{bmatrix}}_{y}$$

(d)

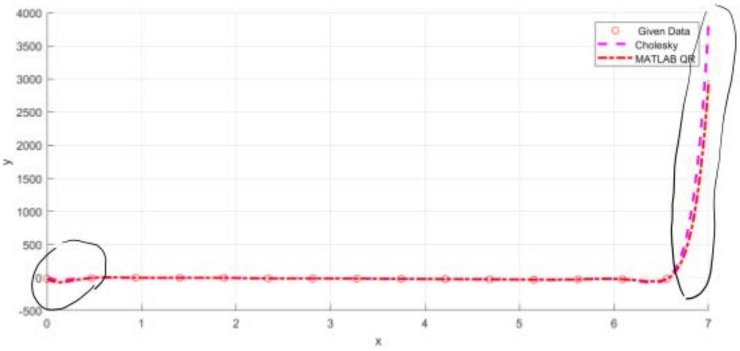


Figure 3

Cholesky is more accurate. We can see QR tends to be less smooth at the beginning and the end (circled). So, cholesky can better represent the trend of the 15 data points. So, cholesky is better because it does not curve as much as QR does.

When we do QR decomposition, the algorithm will have more rounding errors. The generalization of QR decomposition is:

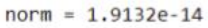
$r_{ij} = q_i^T a_j \ (i \neq j)$ $\qquad$ $r_{jj} = \|v_j\|$

$v_j = a_j - \sum_{i=1}^{j-1} r_{ij} q_i$ ← $\qquad$ $q_j = \frac{v_j}{r_{jj}}$

So, when we are computing v, we would have rounding error when substracting and adding $r_{ij} q_i$. This will generate errors in QR decomposition. Therefore, we see cholesky passes data points exactly, but QR is abit off.

Q4.

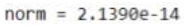
(a) After running the test code for my Gram-schmidt algorithm, The result I get is:



```
norm = 1.9132e-14
```

It's considerably small, so the algorithm is pretty accurate.

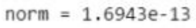
(b) After ruming the test code for my modified Gram-Schmidt algorithm, the result I get is:



```
norm = 2.1390e-14
```

After rerunning several times, the error is still this small. The error is very small, so the algorithm is successfully implemented.

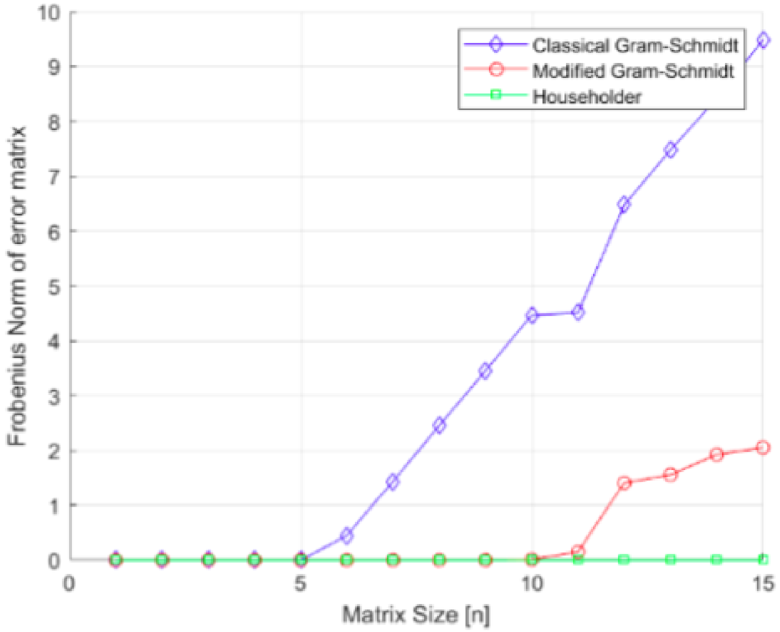
(c) After running the test code for householder, the result I get is:



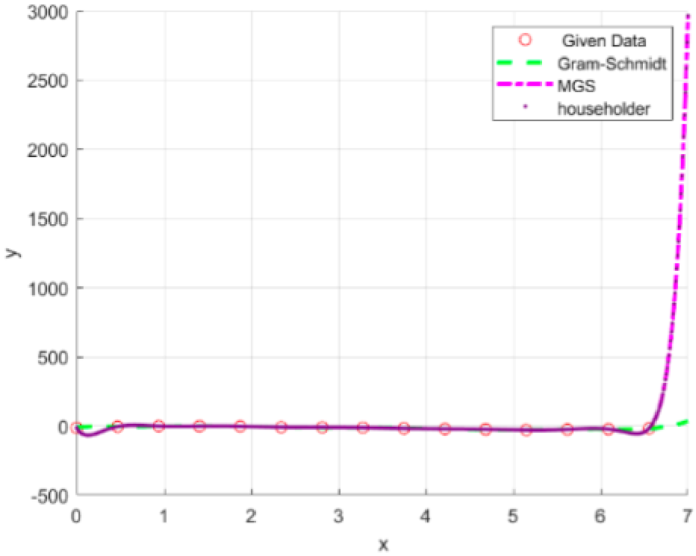
```
norm = 1.6943e-13
```

After rerunning several times, the error is still very small. So, the algorithm is successfully implemented.

(d) As we can see from the graph obtained, classical Gram-Schmidt algorithm starts to have increasing Frobenius norm of error matrix at matrix size of 6. The modified Gram-Schmidt algorithm starts to have increasing errors at 11, while the householder has error of 0 across all matrix sizes from 0 to 15. This indicates that the numerical stability of Gram-Schmidt is the worst of the three and house holder is the most error-free one. Gram-Schmidt has loss of orthogonality due to the round-off errors as discussed in previous questions, so it makes sense that it is the lease accurate one. Modified Gram-Schmidt has changed the order of computation to reduce errors, therefore it would has less numerical errors than the classical one.



(e) As we can see from the graph. Modified Gram-Schmidt algorithm and householder algorithm are overlapping with each other and all the 15 data points. However, classical Gram-Schmidt algorithm is also covering all the data points, but deviate from the other two at some data points. This is also due to the round-off errors when computing QR decomposition. Also, as shown in he graph and indicated in (d), the error will keep increasing as the number of data points increases.


Given Data
Gram-Schmidt
MGS
householder
y
x