DS284: Numerical Linear Algebra
Assignment 4

October 25, 2024

Sandeep Kumar
24022

---

1. Let matrix:

$$A = \begin{bmatrix} 0.70000 & 0.70711 \\ 0.70001 & 0.70711 \end{bmatrix}$$

   (a) Consider a computer that rounds all computed results to five digits of relative accuracy. Using CGS or MGS, what will be the matrix **Q** associated with the QR decomposition of **A** assuming that you are working on such a computer.

   (b) Apply Householder's method to compute the QR factorization of **A** using the same 5-digit arithmetic.

   Compare the **Q** matrices obtained in (a) and (b) and comment on the orthogonal nature of the **Q** matrix.

   Solution: Below is a comparison of the Q and R matrices for different methods, along with the loss of orthogonality:

```
Classical Gram-Schmidt:          |  Householder Triangularization:
Q [[ 0.7071  1.    ]             |  Q [[ -0.70711  0.70712 ]
   [ 0.70711  0.   ]]            |      [ -0.70712 -0.7071  ]]
                                 |
R [[ 9.8996e-01  1.0000e+00 ]    |   R [[ -9.89970e-01 -1.00002e+00 ]
   [ 0.0000e+00  1.0000e-05 ]]   |       [  1.00000e-05  1.00000e-05 ]]
                                 |
Loss of orthogonality: 0.99999 | Loss of orthogonality: 2.68941e-05
```

```
Modified Gram-Schmidt:
Q [[ 0.7071  1.    ]]                R [[ 9.8996e-01  1.0000e+00 ]
   [ 0.70711  0.   ]]                   [ 0.0000e+00  1.0000e-05 ]]

Loss of orthogonality: 0.99999
```

The condition number of the matrix A is given by:

$$\kappa(A) = \frac{\sigma_{\max}}{\sigma_{\min}}$$

$$\sigma_{\max} = 1.40713294, \quad \sigma_{\min} = 5.02518263 \times 10^{-6}, \quad \epsilon_{\mathrm{M}} = 5 \times 10^{-6}$$

$$\boxed{\kappa(A) = 280016.2781}$$

Loss of orthogonality for CGS = $\mathcal{O}\left(\kappa(A)^2 \cdot \epsilon_M\right)$

Loss of orthogonality for Householder Triangularization = $\mathcal{O}\left(\epsilon_M\right)$

Loss of orthogonality for CGS = $\mathcal{O}\left(\kappa(A) \cdot \epsilon_M\right)$

Conclusion for Problem 3:

We have compared the orthogonality loss for **Classical Gram-Schmidt (CGS)**, **Modified Gram-Schmidt (MGS)**, and **Householder Triangularization**. Given that matrix A is close to rank deficiency with a condition number of 280,016.2781, this affects the stability of the different algorithms. The conclusions are as follows:

Classical Gram-Schmidt (CGS):

- **Loss of orthogonality observed:** 0.99999.
- **Theoretical bound:** $\mathcal{O}\left(\kappa(A)^2 \cdot \epsilon_M\right)$, where $\kappa(A)$ is the condition number and ϵ_M is machine precision.
- Since $\kappa(A) = 280,016.2781$, the matrix is highly ill-conditioned, resulting in a significant orthogonality loss. This shows that CGS is not numerically stable for matrices close to rank deficiency.

Modified Gram-Schmidt (MGS):

- **Loss of orthogonality observed:** 0.99999 (similar to CGS in this case).
- **Theoretical bound:** $\mathcal{O}\left(\kappa(A) \cdot \epsilon_M\right)$.
- Although MGS generally provides better numerical stability than CGS, the large condition number of A leads to similar performance here. The orthogonality loss is amplified due to the high $\kappa(A)$, but MGS is expected to perform better than CGS when A is less ill-conditioned.

Householder Triangularization:

- **Loss of orthogonality observed:** 2.68941e-05.
- **Theoretical bound:** $\mathcal{O}\left(\epsilon_M\right)$.

- Householder triangularization is highly stable, as its orthogonality loss is independent of the condition number $\kappa(A)$. This method maintains excellent orthogonality even for ill-conditioned matrices like the one in this case.

2. In this problem you will test different algorithms for the least squares problem to approximate the function $f(t) = \sin(10t)$ for $t \in [0, 1]$ using a polynomial fit. To this end, first generate $m = 100$ data points using the above function which forms your given data i.e. $(t_i, f(t_i))$ for $i = 1, 2, \ldots, m$. Using this data, we would like to construct a 14th degree least squares polynomial fit to $f(t)$. Determine its least squares using the following methods:

   (a) Use QR Factorization with your implementation of Modified Gram Schmidt. You should write your own back substitution code for solving the resulting triangular system.

   (b) Use QR Factorization with your implementation of Householder factorization.

   (c) Using SVD (Computed with any inbuilt libraries in MATLAB/Python/Octave)

   (d) Using normal equations, you can use the backslash command in MATLAB to solve this system.

   Accept the MATLAB/Octave/Python least squares solution (given by backslash 'in MATLAB) as the truth. Display and plot the approximation given by this "true" solution and compare it with $f(t)$. Compare with the solution given by the four methods described above. Explain the results.

   Solution: Let's assume a 14th-order polynomial to fit the data of 100 points

$$\mathcal{P}(x) = c_0 + c_1 x^1 + c_2 x^2 + \cdots + c_{14} x^{14} \tag{1}$$

   When we put the data of 100 points in the equation, we will get an overdefined problem as we have a 14th order polynomial and 15 knowns. In matrix form, this problem will look like as follows:

$$Ac = b$$

   where $A_{100\times 15}$ is a matrix, $c_{15\times 1}$, and $b_{100\times 1}$ are vectors.

   We solve this problem using different methods, and the coefficients of the x's are given in the table below.

Table 1: Comparison of Solutions: c_MGS, c_house, c_SVD, c_NE, and c_lstsq

c_MGS	**c_householder**	**c_SVD**	**c_Normal Equations**	**c_in built**
-2.21943e-07	-1.60396e-07	-1.60395e-07	1.93349e-05	-1.60395e-07
1.00001e+01	1.00001e+01	1.00001e+01	9.99457e+00	1.00001e+01
-4.61558e-03	-3.14923e-03	-3.14923e-03	2.53612e-01	-3.14923e-03
-1.66575e+02	-1.66611e+02	-1.66611e+02	-1.71380e+02	-1.66611e+02
-7.44972e-01	-2.95950e-01	-2.95949e-01	4.51810e+01	-2.95949e-01
8.33924e+02	8.30457e+02	8.30457e+02	5.82784e+02	8.30457e+02
3.77339e+01	5.52238e+01	5.52239e+01	8.41821e+02	5.52239e+01
-2.33181e+03	-2.39204e+03	-2.39204e+03	-3.67736e+03	-2.39204e+03
1.65591e+03	1.80111e+03	1.80111e+03	1.81227e+03	1.80111e+03
-2.26144e+03	-2.50883e+03	-2.50883e+03	2.23056e+03	-2.50883e+03
1.02336e+04	1.05301e+04	1.05301e+04	-3.35875e+02	1.05301e+04
-1.66295e+04	-1.68741e+04	-1.68741e+04	-4.00635e+03	-1.68741e+04
1.26552e+04	1.27873e+04	1.27873e+04	3.90087e+03	1.27873e+04
-4.75836e+03	-4.80045e+03	-4.80045e+03	-1.40255e+03	-4.80045e+03
7.21568e+02	7.27561e+02	7.27561e+02	1.69233e+02	7.27561e+02

solution of Householder, SVD, and python in built(true value) function are close to each other, while the solution of MGS and Normal equations are not close to the true value. when we plot the polynomial for the same 100 points using the MGS, SVD, Householder and Normal equation the results looks indistinguishable (as shown in fig1).

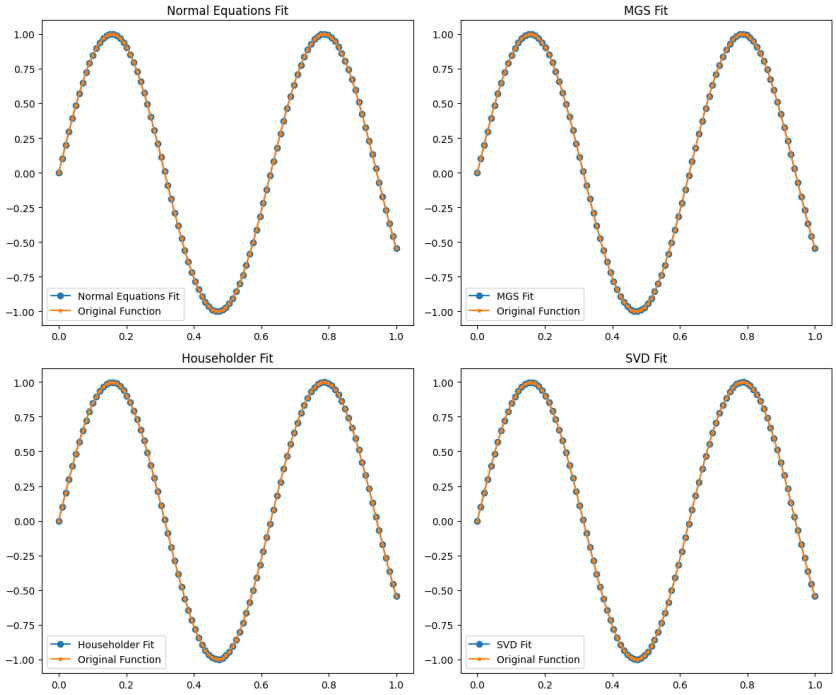

Figure 1:

The least squares errors for different methods are as follows:

- Least Square Error for MGS: 3.0478×10^{-7}
- Least Square Error for Householder: 1.2100×10^{-11}
- Least Square Error for SVD: 6.8180×10^{-12}
- Least Square Error for NE: 0.0001084

The condition number for A is $\kappa(A) = 2.2717772809977962 \times 10^{10}$ and for A^TA is $\kappa(A^TA) = 1.1020061528999347 \times 10^{18}$. The machine epsilon is $\epsilon_M = 1 \times 10^{-16}$.

Least square error for different methods are coming in less than or equal order as theoritical error order.

Order of Error:

MGS: The error bound is given by $\kappa(A) \cdot \epsilon_M = 10^{-16} \times 10^{10} = 10^{-6}$.

Householder: The error bound is approximately $\epsilon_M = 10^{-16}$.

SVD: The error bound is also approximately $\epsilon_M = 10^{-16}$.

Normal Equations: the error bound is approximately $\kappa(A^TA) \cdot \epsilon_M = 10^{18} \times 10^{-16} = 10^2$.

Modified Gram-Schmidt (MGS): For MGS, the orthogonality error is estimated by $O(\kappa(A) \cdot \epsilon_M)$, where $\kappa(A)$ is the condition number and ϵ_M is the machine epsilon.

As $\kappa(A)$ increases, MGS becomes more susceptible to error amplification, leading to greater numerical instability. This sensitivity explains the larger observed errors in MGS for high-condition-number matrices.

Householder and SVD Methods: Householder and Singular Value Decomposition (SVD) methods have error bounds closer to $O(\epsilon_M)$, meaning they are largely unaffected by high condition numbers. This stability results in significantly lower errors for both methods, even with ill-conditioned matrices, as observed.

Normal Equations: For the Normal Equations method, the condition number is effectively squared in the system $A^T A \cdot x = A^T b$, resulting in a condition number of $\kappa(A)^2$. This higher condition number makes the method highly sensitive to round-off errors, causing a greater deviation from the true solution. This amplified sensitivity, especially in matrices with high condition numbers, accounts for the larger error observed with the Normal Equations method.

Problem 3:

common data? $A \in \mathbb{C}^{m \times m}$

(a) $Ax = \lambda x$; $\mu \in \mathbb{C}$, then $(A - \mu I)x = (\lambda - \mu)x$??

$(A - \mu I)x = (\lambda - \mu)x$

$Ax - \mu x = \lambda x - \mu x$

$Ax = \lambda x$ — true

(b) A is real and $Ax = \lambda x$; (then $-\lambda$. ??)

let's say $A = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$

is diagonal matrix where $\lambda_1 \neq \lambda_2 \neq \lambda_3$, then eigen values of A will be λ_1, λ_2 and λ_3. $-\lambda_1, -\lambda_2, -\lambda_3$ will not be eigen values of A. hence **false**

(c) $A \in \mathbb{R}^{m \times m}$, $Ax = \lambda x$, then λ^*. ($*$ → complex conjugate operation.

$(Ax)^* = (\lambda x)^* \Rightarrow$ $A x^* = \lambda^* x^*$

if λ is an eigen value of A, then λ^* will also be an eigenvalue of A and eigenvector Associated with λ^* will x^*.

If λ is real, then $Ax = \lambda x$ — true

(d) $Ax = \lambda x$, $\det(A) \neq 0$, $A^{-1}y = \frac{1}{\lambda} y$

$Ax = \lambda x$

multiply both sides with A^{-1}

$A^{-1}Ax = \lambda A^{-1}x$

$x = \lambda A^{-1}x$

$A^{-1}x = \frac{x}{\lambda}$ $\Rightarrow$ eigenvalue of A^{-1} will be $\frac{1}{\lambda}$ and eigenvector will be same as eigenvector of A associated with eigenvalue λ. **true**

(e) all eigenvalues of A are zero, then $A = 0$. ??

All eigenvalues of A equal to zero $\Rightarrow$ $A = 0$

let's assume $A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \Rightarrow A \neq 0$

$(A - \lambda I) = \begin{bmatrix} 0-\lambda & 0 & 1 \\ 0 & 0-\lambda & 0 \\ 0 & 0 & 0-\lambda \end{bmatrix} = -\lambda(\lambda^2) - 0(0) + 1(0-0)$

$= \lambda^3 = 0$

$\lambda_{1,2,3} = 0$

false.

(f) A is diagonalizable

all eigenvalues are equal $= \lambda$

$\Lambda = \lambda I$

$A = X^{-1} \Lambda X$

$A = \lambda X^{-1} X$ $(X^{-1}X = I)$

$A = \lambda I$

true.