1) Given $A \in \mathbb{C}^{m\times n}$ with $m \geq n$, show that A^*A is nonsingular if and only if A has full rank.

Recall that A is full rank if $\text{rank}(A) = \min\{m, n\} = n$. Additionally, note that $\text{rank}(A^*A) = \text{rank}(A)$.

Let us prove the forward direction first. Suppose that A^*A is a nonsingular matrix, meaning that its eigenvalues are nonzero (otherwise its determinant would be zero). Hence, A has n nonzero singular values, or equivalently Σ has n nonzero diagonal entries, which implies that A has rank n since $\text{rank}(A) = \text{rank}(U\Sigma V^{\mathrm{T}}) = \text{rank}(\Sigma) = n$, where we used the fact that U and V are invertible matrices.

Now, suppose that A is full rank. Then, since A has a singular value decomposition

$$A^*A = V\Sigma^2V^*, \tag{1}$$

which implies that

$$\det(A^*A) = \det\left(\Sigma^2\right) = \prod_{i=1}^{n} \sigma_i^2 \neq 0, \tag{2}$$

where the last inequality comes from the fact that $\text{rank}(\Sigma) = \text{rank}(A) = n$, implying that Σ has n nonzero singular values. Thus, we have shown that A^*A is nonsingular.

2) For the matrix

$$A = \begin{pmatrix} 3 & 4 \\ 0 & 1 \\ 4 & 0 \end{pmatrix} \tag{3}$$

a) Find the Householder matrix H

We calculate the Householder matrix as follows. Denote $A = \begin{pmatrix} a_1 a_2 \end{pmatrix}$, where $a_1 = \begin{pmatrix} 3 & 0 & 4 \end{pmatrix}^{\mathrm{T}}$ and $a_2 = \begin{pmatrix} 4 & 1 & 0 \end{pmatrix}^{\mathrm{T}}$. Our first reflection matrix is constructed as follows such that $a_1 \to ||a_1||_2 e_1$under H_1. Observe that $||a_1|| = 5$ and let

$$u_1 = a_1 + 5e_1 = \begin{pmatrix} 8 \\ 0 \\ 4 \end{pmatrix}. \tag{4}$$

Then

$$H_1 = \mathbb{1}_{3\times 3} - \frac{2}{||u||^2}uu^{\mathrm{T}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - \frac{2}{5}\begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}\begin{pmatrix} 2 & 0 & 1 \end{pmatrix} = \begin{pmatrix} -3/5 & 0 & -4/5 \\ 0 & 1 & 0 \\ -4/5 & 0 & 3/5 \end{pmatrix}, \tag{5}$$

and

$$A \to H_1 A = \begin{pmatrix} -5 & -12/5 \\ 0 & 1 \\ 0 & -16/5 \end{pmatrix}. \tag{6}$$

Now, we repeat a similar process for the second column. We let $\tilde{a}_2 = \begin{pmatrix} 1 & -16/5 \end{pmatrix}^{\mathrm{T}}$, and

$$u = \tilde{a}_2 + \operatorname{sign}(1) e_1 = \begin{pmatrix} 1 + \sqrt{281}/5 \\ -16/5 \end{pmatrix}. \tag{7}$$

Hence,

$$H_2 = \begin{pmatrix} 1 & 0 \\ 0 & \mathbb{1} - 2\hat{u}\hat{u}^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -5/\sqrt{281} & 16/\sqrt{281} \\ 0 & 16/\sqrt{281} & 5/\sqrt{281} \end{pmatrix}. \tag{8}$$

Finally, we have

$$H_1 A \to H_2 H_1 A = \begin{pmatrix} -5 & -12/5 \\ 0 & -\sqrt{281}/5 \\ 0 & 0 \end{pmatrix}. \tag{9}$$

The full reflection matrix $H = H_2 H_1$ is as follows:

$$H = \begin{pmatrix} -3/5 & 0 & -4/5 \\ -64\sqrt{281}/1405 & -5\sqrt{281}/281 & 48\sqrt{281}/1405 \\ -4\sqrt{281}/281 & 16\sqrt{281}/281 & 3\sqrt{281}/281 \end{pmatrix}. \tag{10}$$

b) Use (a) to calculate QR decomposition

With the above work, we can calculate the QR decomposition simply. The matrix R is just the upper triangular matrix

$$R = \begin{pmatrix} -5 & -12/5 \\ 0 & -\sqrt{281}/5 \end{pmatrix}, \tag{11}$$

and

$$Q = (H_2 H_1)^{-1} = (H_2 H_1)^{\mathrm{T}} = H_1^{\mathrm{T}} H_2^{\mathrm{T}} = \begin{pmatrix} -3/5 & -64\sqrt{281}/1405 & -4\sqrt{281}/281 \\ 0 & -5\sqrt{281}/281 & 16\sqrt{281}/281 \\ -4/5 & 48\sqrt{281}/1405 & 3\sqrt{281}/281 \end{pmatrix}. \tag{12}$$

Note that we remove the last column such that Q has the same shape as A:

$$Q = \begin{pmatrix} -3/5 & -64\sqrt{281}/1405 \\ 0 & -5\sqrt{281}/281 \\ -4/5 & 48\sqrt{281}/1405 \end{pmatrix}. \tag{13}$$

3) Let $A = U\Sigma V^{\mathrm{T}} \in \mathbb{R}^{m\times n}$ $(m \geq n)$ with singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n$. Prove $||A||_2 = \sigma_1$. Moreover, if A is square and nonsingular, then $||A^{-1}||_2^{-1} = \sigma_n$ and $\kappa_2(A) = \sigma_1/\sigma_n$.

Recall that $||A||_2 = \sqrt{\rho(A^{\mathrm{T}}A)}$, where $\rho(A^{\mathrm{T}}A)$ is the spectral radius of A. The eigenvalues of $A^{\mathrm{T}}A$ just the square of the singular values of A, so by definition

$$||A||_2 = \sqrt{\rho(A^{\mathrm{T}}A)} = \sqrt{\sigma_1^2} = \sigma_1. \tag{14}$$

If A is $n \times n$ and nonsingular, then all of its singular values are nonzero (i.e. Σ is invertible). This can be observed as follows. Since A is nonsingular, then $A^{\mathrm{T}}A$ is nonsingular since $\det\big(A^{\mathrm{T}}A\big) = \det^2(A) \neq 0$. Hence, this means that all the eigenvalues of $A^{\mathrm{T}}A$ are nonzero (otherwise its determinant would be zero), and therefore the singular values of A are nonzero. Thus,

$$A^{-1} = V\Sigma^{-1}U^{\mathrm{T}}, \tag{15}$$

where $\Sigma_{ii}^{-1} = 1/\sigma_i$ for $i = 1, 2, \ldots, n$ and $\Sigma_{ij}^{-1} = 0$ if $i \neq j$. Notice that the ordering of the singular values of A implies that $1/\sigma_n \geq \ldots \geq 1/\sigma_2 \geq 1/\sigma_1$, implying that the largest singular value of A^{-1} is $1/\sigma_n$. By the same work as above then

$$||A^{-1}||_2 = \frac{1}{\sigma_n} \Leftrightarrow ||A^{-1}||_2^{-1} = \sigma_n. \tag{16}$$

Finally, we observe that for a square nonsingular matrix

$$\kappa_2(A) = ||A^{-1}||_2||A||_2 = \frac{\sigma_1}{\sigma_n}. \tag{17}$$

4) Assume A is singular, and x is the solution to

$$\min_x ||Ax - b||_2. \tag{18}$$

Let $A = U\Sigma V^{\mathrm{T}}$ have rank $r < n$ and its SVD can be written as

$$A = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^{\mathrm{T}} \\ V_2^{\mathrm{T}} \end{bmatrix}. \tag{19}$$

where Σ_1 is $r \times r$ nonsingular, and U_1, V_1 have r columns. Prove all solutions x can be written as $x = V_1\Sigma_1^{-1}U_1^{\mathrm{T}}b + V_2 z$, with z is an arbitrary vector.

We proved in class that the vector x which minimizes this norm is $x = V\Sigma^{-1}U^{\mathrm{T}}b$. Written in terms of $U_{1,2}$, $V_{1,2}$, and Σ_1, we find that

$$x = \begin{pmatrix} V_1 & V_2 \end{pmatrix} \begin{pmatrix} \Sigma_1^{-1} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} U_1^{\mathrm{T}} \\ U_2^{\mathrm{T}} \end{pmatrix} b = V_1\Sigma_1^{-1}U_1^{\mathrm{T}}b. \tag{20}$$

However, note that since A has rank r, the kernel (or nullspace) of A has dimension $n - r$, where $A(V_2 z) = 0$ for any vector z. This can be observed by noting that V is an orthogonal matrix, and since $A = U_1 \Sigma_1 V_1^{\mathrm{T}}$, then it follows that $AV_2 = 0$ since $V_1^{\mathrm{T}} V_2 = 0$. Hence, the vector $x = V_1 \Sigma_1^{-1} U_1^{\mathrm{T}} b + V_2 z$ is also a vector which minimizes the norm above.

5) Consider

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 1 \\ 2 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 12 \\ 6 \\ 18 \end{pmatrix}. \tag{21}$$

Write codes to solve the linear least square problem $Ax = b$ using (a) normal equation methods; (b) QR decomposition with modified Gram-Schmidt orthogonalization; (c) SVD decomposition.

For each method, the script is displayed along with a screenshot of the output:

a)

```
#!/usr/bin/env python3

import numpy as np

def LU_factorize(A):
    n = np.shape(A)[0]

    L = np.zeros(np.shape(A))
    U = np.copy(A)

    for j in range(n):
        L[j,j] = 1
        for i in range(j+1,n):
            L[i,j] = U[i,j]/U[j,j]
        for l in range(j+1,n):
            for m in range(j,n):
                U[l,m] = U[l,m] - U[j,m]*L[l,j]
    return L,U

def solve_x(A,b):
    n = np.shape(A)[0]

    L,U = LU_factorize(A)

    y = np.zeros(n)
    y[0] = b[0]/L[0,0]
    for i in range(1,n):
        temp = np.array([y[k]*L[i,k] for k in range(i)])
        y[i] = (b[i] - np.sum(temp))/L[i,i]

    x = np.zeros(n)
    x[-1] = y[-1]/U[-1,-1]
    for i in range(n-2,-1,-1):
```

```
        temp = np.array([x[k]*U[i,k] for k in range(i+1,n)])
        x[i] = (y[i] - np.sum(temp))/U[i,i]

    return x

def least_squares_normal(A,b):
    return solve_x(A.T@A,A.T@b)

if __name__ == '__main__':

    A = np.array([
        [2.0,1.0],
        [1.0,1.0],
        [2.0,1.0]
        ],)
    b = np.array([12.0,6.0,18.0])
    x = least_squares_normal(A,b)

    np.set_printoptions(precision=3)
    print('\n  x = {}'.format(x))
    print('  residual: {:.3f}\n'.format(np.linalg.norm(A@x-b)))
```

```
x = [ 9. -3.]
residual: 4.243
```

b)

```
#!usr/bin/env python3

import numpy as np

def QR_modifiedGS(A):
    m,n = np.shape(A)
    R = np.zeros([n,n])
    Q = np.copy(A)
    for i in range(n):
        v = Q[:,i]
        for j in range(i):
            R[j,i] = v.T @Q[:,j]
            Q[:,i] -= R[j,i]*Q[:,j]
        R[i,i] = np.linalg.norm(Q[:,i])
        Q[:,i] = v/R[i,i]

    return Q,R

def least_squares_QR(A,b):
    m,n = np.shape(A)
    Q,R = QR_modifiedGS(A)

    y = Q.T @ b
```

```
    x = np.zeros(n)
    x[-1] = y[-1]/R[-1,-1]
    for i in range(n-2,-1,-1):
        temp = np.array([x[k]*R[i,k] for k in range(i+1,n)])
        x[i] = (y[i] - np.sum(temp))/R[i,i]

    return x


if __name__ == '__main__':

    A = np.array([
        [2.0,1.0],
        [1.0,1.0],
        [2.0,1.0]
        ])
    b = np.array([12.0,6.0,18.0])

    x = least_squares_QR(A,b)

    np.set_printoptions(precision=3)
    print('\n  x = {}'.format(x))
    print('  residual: {:.3f}\n'.format(np.linalg.norm(A@x-b)))
```

```
x = [ 9. -3.]
residual: 4.243
```

c)

```
#!usr/bin/env python3

import numpy as np

def least_squares_SVD(A,b):
    U,S,VT = np.linalg.svd(A,full_matrices=False)

    y = U.T @ b
    w = np.array([y[i]/S[i] for i in range(len(y))])
    x = VT.T @ w

    return x

if __name__ == '__main__':

    A = np.array([
        [2.0,1.0],
        [1.0,1.0],
        [2.0,1.0]
        ])
    b = np.array([12.0,6.0,18.0])
```

```python
x = least_squares_SVD(A,b)

np.set_printoptions(precision=3)
print('\n  x = {}'.format(x))
print('  residual: {:.3f}\n'.format(np.linalg.norm(A@x-b)))
```

```
x = [ 9. -3.]
residual: 4.243
```