Assignment 1

Math Fundamentals for Robotics 16-811, Fall 2025

DUE: Thursday, September 11, 2025, by 11:59pm

1. Implement the $PA = LDU$ decomposition algorithm discussed in class. Do so yourself (in other words, do not merely use predefined Gaussian elimination code in MatLab or Python).

   Simplifications: (i) You may assume that the matrix A is square and invertible.
   (ii) Do not worry about column interchanges, just row interchanges.

   Demonstrate (in your pdf) that your implementation works properly, on some examples.

2. Compute the $PA = LDU$ decomposition and the SVD decomposition for each of the following matrices: (In fact, here it is enough to let P be an identity matrix, so $A = LDU$.)

$$A_1 = \begin{pmatrix} 10 & -10 & 0 \\ 0 & -4 & 2 \\ 2 & 0 & -5 \end{pmatrix} \qquad A_2 = \begin{pmatrix} 5 & -5 & 0 & 0 \\ 5 & 5 & 5 & 0 \\ 0 & -1 & 4 & 1 \\ 0 & 4 & -1 & 2 \\ 0 & 0 & 2 & 1 \end{pmatrix} \qquad A_3 = \begin{pmatrix} 1 & 1 & 1 \\ 10 & 2 & 9 \\ 8 & 0 & 7 \end{pmatrix}.$$

   You may use any method or tools you wish to solve this problem, including pre-defined routines from MatLab or Python, your code, and/or hand calculations. (Hand calculations may be easiest for computing the $PA = LDU$ decompositions of these examples. We recommend using pre-defined code to compute the SVD decompositions.)

   Show how you obtained your solutions, that is, show your work, including intermediate steps.

3. Solve the systems of equations $Ax = b$ for the values of A and b given below.

   For each system, specify whether the system has zero, one, or many exact solutions. If a system has zero exact solutions, give "the SVD solution" (as defined in class) and explain what this solution means. If a system has a unique exact solution, compute that solution. If a system has more than one exact solution, specify both "the SVD solution" and all solutions, using properties of the SVD decomposition of the matrix A, as discussed in class.

   Show your work, including verifying that your answers are correct.

$$\text{(a)} \qquad A = \begin{pmatrix} 10 & -10 & 0 \\ 0 & -4 & 2 \\ 2 & 0 & -5 \end{pmatrix} \qquad b = \begin{pmatrix} 10 \\ 2 \\ 13 \end{pmatrix}$$

$$\text{(b)} \qquad A = \begin{pmatrix} 1 & 1 & 1 \\ 10 & 2 & 9 \\ 8 & 0 & 7 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix}$$

$$\text{(c)} \qquad A = \begin{pmatrix} 1 & 1 & 1 \\ 10 & 2 & 9 \\ 8 & 0 & 7 \end{pmatrix} \qquad b = \begin{pmatrix} 3 \\ 2 \\ 2 \end{pmatrix}$$

   Parts (b) and (c) have the same A matrix but different b vectors. You will see that "the SVD solution" is the same for both parts. Explain why that makes sense. Your explanation should be based on the meaning of the columns of U that arose in A's SVD decomposition.

4. Suppose that u is an n-dimensional column vector of unit length in $\Re^n$, and let u^T be its transpose. Then uu^T is a matrix. Consider the $n \times n$ matrix $A = I - uu^T$.
   (a) Describe the action of the matrix A geometrically.
   (b) Give the eigenvalues of A.
   (c) Describe the null space of A.
   (d) What is A^2?

   (As always, show your work.)

5. The following problem arises in a large number of robotics and vision problems: Suppose $\mathbf{p}_1, \ldots, \mathbf{p}_n$ are the 3D coordinates of n points located on a rigid body in three-space. Suppose further that $\mathbf{q}_1, \ldots, \mathbf{q}_n$ are the 3D coordinates of these same points after the body has been translated and rotated by some unknown amount. Derive an algorithm in which SVD plays a central role for inferring the body's translation and rotation. (You may assume that the coordinate values are precise not noisy, but see comment and caution below.)

   Show (in your pdf) that your algorithm works correctly by running it on some examples.

   Comment: This problem requires some thought. There are different approaches. Although you can find a solution on the web or in a vision text book, try to solve the problem yourself before looking at any such sources. Spend some time on the problem. It is good practice to develop your analytic skills. Feel free to discuss among yourselves. (As always, cite any sources, including discussions with others.)

   Requirement: Your algorithm should make use of all the information available. True, in principle you only need three pairs of points – but if you use more points your solution will be more robust, something that might come in handy some day when you need to do this for real with noisy data.

   Caution: A common mistake is to derive an algorithm that finds the best affine transformation, rather than the best rigid body transformation. Even though you may assume precise coordinate values, imagine how your algorithm would behave with noise. Your algorithm should still produce a rigid body transformation.

   Hint: Suppose for a moment that both sets of points have the origin as centroid. Assemble all the points $\{\mathbf{p}_i\}$ into a matrix P and all the points $\{\mathbf{q}_i\}$ into another matrix Q. Now think about the relationship between P and Q. You may wish to find a rigid body transformation that minimizes the sum of squared distances between the points $\{\mathbf{q}_i\}$ and the result of applying the rigid body transformation to the points $\{\mathbf{p}_i\}$.

   You may find the following facts useful (assuming the dimensions are sensible):

$$\|x\|^2 = x^T x, \qquad x^T R^T y \;=\; \mathrm{Tr}(Rxy^T).$$

   [Here x and y are column vectors (e.g., 3D vectors) and R is a matrix (e.g., a 3×3 rotation matrix). The superscript T means *transpose*, so $x^T x$ is a number and xy^T is a matrix. Also, Tr is the *trace* operator that adds up the diagonal elements of its square matrix argument.]

   You will have more complicated expressions for x and y, involving the points $\{\mathbf{p}_i\}$ and $\{\mathbf{q}_i\}$.

2. Compute the $PA = LDU$ decomposition and the SVD decomposition for each of the following matrices: (In fact, here it is enough to let P be an identity matrix, so $A = LDU$.)

$$A_1 = \begin{pmatrix} 10 & -10 & 0 \\ 0 & -4 & 2 \\ 2 & 0 & -5 \end{pmatrix} \qquad A_2 = \begin{pmatrix} 5 & -5 & 0 & 0 \\ 5 & 5 & 5 & 0 \\ 0 & -1 & 4 & 1 \\ 0 & 4 & -1 & 2 \\ 0 & 0 & 2 & 1 \end{pmatrix} \qquad A_3 = \begin{pmatrix} 1 & 1 & 1 \\ 10 & 2 & 9 \\ 8 & 0 & 7 \end{pmatrix}.$$

You may use any method or tools you wish to solve this problem, including pre-defined routines from MatLab or Python, your code, and/or hand calculations. (Hand calculations may be easiest for computing the $PA = LDU$ decompositions of these examples. We recommend using pre-defined code to compute the SVD decompositions.)

Show how you obtained your solutions, that is, show your work, including intermediate steps.

$A_1 = \begin{pmatrix} 10 & -10 & 0 \\ 0 & -4 & 2 \\ 2 & 0 & -5 \end{pmatrix}$

$\begin{matrix} 2 & 0 & -5 \\ -2 & +2 & 0 \\ 0 & -2 & 1 \end{matrix}$

L

$\begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1/5 & -1/2 & 1 \end{matrix}$

A'

$\begin{matrix} 10 & -10 & 0 \\ 0 & -4 & 2 \\ 0 & 0 & -4 \end{matrix}$

$LA' = A$

$D = \begin{pmatrix} 10 & 0 & 0 \\ 0 & -4 & 0 \\ 0 & 0 & -4 \end{pmatrix}$

$U = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & -0.5 \\ 0 & 0 & 1 \end{pmatrix}$

$$A_2 = \begin{pmatrix} 5 & -5 & 0 & 0 \\ 5 & 5 & 5 & 0 \\ 0 & -1 & 4 & 1 \\ 0 & 4 & -1 & 2 \\ 0 & 0 & 2 & 1 \end{pmatrix} \begin{matrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{matrix} \qquad P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

L

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & -\frac{1}{10} & 1 & 0 & 0 \\ 0 & \frac{4}{10} & -\frac{2}{3} & 1 & 0 \\ 0 & 0 & \frac{4}{9} & \frac{5}{24} & 1 \end{pmatrix}$$

A' $\quad \frac{4}{10} \times 5$

$$\begin{pmatrix} 5 & -5 & 0 & 0 & 0 \\ 0 & 10 & 5 & 0 & 0 \\ 0 & 0 & 4.5 & 1 & 0 \\ 0 & 0 & 0 & \frac{8}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$\text{row}2 - \text{row}1$

$r3 - (-\frac{1}{10} r2)$

$r4 - (-\frac{2}{3} r3)$

$-\frac{4}{10} r2$

$-1 - \frac{4}{2} - \frac{9}{2}x = 0$

$-\frac{9}{2}x = \frac{6}{2}$

$x = -\frac{6}{9}$

$\text{row}3 - (-\frac{1}{10}\text{row}2) = -1 - (-\frac{1}{10} \times 10) \quad 4 - (-\frac{1}{10} \times 5) \quad 1 - 0$

$= 0 \qquad 4.5 \qquad 1$

$\text{row}4 - (\frac{2}{3} r2) - \frac{4}{10} r2 = 4 - \frac{4}{10}$

$-1 - \frac{6}{2} - (\frac{9}{2} \times -\frac{1}{3}) \quad 2 - (-\frac{2}{3} \times 1)$

$$\Rightarrow L = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & -\frac{1}{10} & 1 & 0 & 0 \\ 0 & \frac{4}{10} & -\frac{2}{3} & 1 & 0 \\ 0 & 0 & \frac{4}{9} & \frac{5}{24} & 1 \end{bmatrix}$$

$$A' = \begin{bmatrix} 5 & -5 & 0 & 0 & 0 \\ 0 & 10 & 5 & 0 & 0 \\ 0 & 0 & \frac{9}{2} & 1 & 0 \\ 0 & 0 & 0 & \frac{8}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & \frac{9}{2} & 0 \\ 0 & 0 & 0 & \frac{8}{3} \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad U = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & \frac{1}{2} & 0 \\ 0 & 0 & 1 & \frac{2}{9} \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 1 & 1 & 1 \\ 10 & 2 & 9 \\ 8 & 0 & 7 \end{bmatrix}$$

L

$$\begin{matrix} 1 & 0 & 0 \\ 10 & 1 & 0 \\ 8 & 1 & 1 \end{matrix}$$

A'

$$\begin{matrix} 1 & 1 & 1 \\ 0 & -8 & -1 \\ 0 & 0 & 0 \end{matrix}$$

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 10 & 1 & 0 \\ 8 & 0 & 1 \end{bmatrix} \qquad D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -8 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$U = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1/8 \\ 0 & 0 & 1 \end{bmatrix}$$

3. Solve the systems of equations $Ax = b$ for the values of A and b given below.

For each system, specify whether the system has zero, one, or many exact solutions. If a system has zero exact solutions, give "the SVD solution" (as defined in class) and explain what this solution means. If a system has a unique exact solution, compute that solution. If a system has more than one exact solution, specify both "the SVD solution" and all solutions, using properties of the SVD decomposition of the matrix A, as discussed in class.

Show your work, including verifying that your answers are correct.

(a) $A = \begin{pmatrix} 10 & -10 & 0 \\ 0 & -4 & 2 \\ 2 & 0 & -5 \end{pmatrix} \quad b = \begin{pmatrix} 10 \\ 2 \\ 13 \end{pmatrix}$

(b) $A = \begin{pmatrix} 1 & 1 & 1 \\ 10 & 2 & 9 \\ 8 & 0 & 7 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix}$

(c) $A = \begin{pmatrix} 1 & 1 & 1 \\ 10 & 2 & 9 \\ 8 & 0 & 7 \end{pmatrix} \quad b = \begin{pmatrix} 3 \\ 2 \\ 2 \end{pmatrix}$

Parts (b) and (c) have the same A matrix but different b vectors. You will see that "the SVD solution" is the same for both parts. Explain why that makes sense. Your explanation should be based on the meaning of the columns of U that arose in A's SVD decomposition.

a) $A = \begin{pmatrix} 10 & -10 & 0 \\ 0 & -4 & 2 \\ 2 & 0 & -5 \end{pmatrix} \quad b = \begin{pmatrix} 10 \\ 2 \\ 13 \end{pmatrix}$

$\det A = 160 \neq 0.$

A is square & $\det(A) \neq 0 \Rightarrow A$ is invertable.

$\Rightarrow Ax = b$ has a unique solution that can be solved by SVD solution.

$\vec{x} = V \cdot \frac{1}{S} \cdot U^T \cdot b$

(b) $A = \begin{pmatrix} 1 & 1 & 1 \\ 10 & 2 & 9 \\ 8 & 0 & 7 \end{pmatrix}$ $b = \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix}$

$$\det(A) = 1\begin{vmatrix} 2 & 9 \\ 0 & 7 \end{vmatrix} - 1\begin{vmatrix} 10 & 9 \\ 8 & 7 \end{vmatrix} + 1\begin{vmatrix} 8 & 2 \\ 8 & 0 \end{vmatrix}$$

$$= 14 - (70 - 72) - 16$$

$$= 0$$

$\Rightarrow$ Rank $(A) \leq 2$. $\Rightarrow$ A is square but not invertable.

$R_2 = 2 \times R_1 + R_3$.

$\therefore$ row space of this matrix is $(1\ 1\ 1)$ & $(8\ 0\ 7)$.

$C_3 = \frac{7}{8} C_1 + \frac{1}{8} C_2$.

$\therefore$ Column space of A is $(1\ 10\ 8)^T$ & $(1\ 2\ 0)^T$

$$x_1 + x_2 + x_3 = 1$$

$$10x_1 + 2x_2 + 9x_3 = 3$$

$$8x_1 + 7x_3 = 1 \Rightarrow x_1 = \frac{1 - 7x_3}{8}$$

$\Downarrow$ to ①

$$\frac{1 - 7x_3}{8} + x_2 + x_3 = 1$$

$$x_2 = 1 - \frac{1 + x_3}{8}$$

$$x_2 = \frac{7 - x_3}{8}$$

$$\begin{cases} x_1 = \dfrac{1-7x_3}{8} \\ x_2 = \dfrac{7-x_3}{8} \\ x_3 = x_3 \end{cases} \Rightarrow x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \dfrac{1-7x_3}{8} \\ \dfrac{7-x_3}{8} \\ x_3 \end{bmatrix}$$

$$x = x_p + x_n = \begin{bmatrix} \frac{1}{8} \\ \frac{7}{8} \\ 0 \end{bmatrix} + \begin{bmatrix} -\frac{7}{8} \\ -\frac{1}{8} \\ 1 \end{bmatrix} x \,, \; x \in \mathbb{R}$$

(c) $A = \begin{pmatrix} 1 & 1 & 1 \\ 10 & 2 & 9 \\ 8 & 0 & 7 \end{pmatrix}$ $b = \begin{pmatrix} 3 \\ 2 \\ 2 \end{pmatrix}$

Same as (b). $\det(A) = 0$

we have $\mathcal{U}(A) = \text{span}\left(\begin{bmatrix} -7/8 \\ -1/8 \\ 1 \end{bmatrix}\right)$.

Double check:

$$x_1 + x_2 + x_3 = 3$$
$$10x_1 + 2x_2 + 9x_3 = 2$$
$$8x_1 + 7x_3 = 2 \Rightarrow x_1 = \frac{2-7x_3}{8}$$

$x_1 = \frac{2-7x_3}{8} \Rightarrow ① \Rightarrow \frac{2-7x_3}{8} + x_2 + x_3 = 3$

$$x_2 = 3 - \frac{2+x_3}{8}$$
$$= \frac{22-x_3}{8}$$

$$10\left(\frac{2-7x_3}{8}\right) + 2\left(\frac{22-x_3}{8}\right) + 9x_3 = 2$$

$$20 - 70x_3 + 44 - 2x_3 + 72x_3 = 16$$

$$64 \neq 16$$

which means A_3 has no unique solution

Base on matlab. code.

$$\vec{x} = V \cdot \frac{1}{S} \cdot U^T \cdot b_3 = \begin{bmatrix} 0.0175 \\ 0.8596 \\ 0.1228 \end{bmatrix}$$

4. Suppose that u is an n-dimensional column vector of unit length in $\Re^n$, and let u^T be its transpose. Then uu^T is a matrix. Consider the $n \times n$ matrix $A = I - uu^T$.
   (a) Describe the action of the matrix A geometrically.
   (b) Give the eigenvalues of A.
   (c) Describe the null space of A.
   (d) What is A^2?

   (As always, show your work.)

a). $A = I - uu^T$ $\quad u$: column.

uu^T is the orthogonal projector onto span $\{u\}$.

So $A = I - uu^T$ is a orthogonal projector onto $u^\perp$.

$A = I - uu^T$ $\quad uu^T x = (u^T x)\, u$

$$Ax = x - uu^T x$$
$$= x - (u^T x)\, u$$
$$= u^\perp$$

$\mathbb{R}^n$ is consist of $\{u\}$ with $\{u\}^\perp$.

$\{u\}^\perp$ is hyperplane that perpendicular to $\{u\}$.

$x = a + b \quad a \in \{u\} \quad b \in \{u\}^\perp$.

Assume $a = ku \quad k \in \mathbb{R}$.

$$Aa = (I - uu^T)\, a = ku - uu^T ku = ku - ku = 0$$

$$Ab = (I - uu^T)\, b = b - uu^T b = b$$

$\Rightarrow A(a+b) = Ax = b$

$\Rightarrow$ A is remove the component along u and keep the rest that orthogonal to u.

b) eigenvalue of A.

$$\det(A - \lambda I) = \det\left((1-\lambda)I - uu^T\right) = 0$$

$$= (1-\lambda)^{n-1} \det\left((1-\lambda)I - u^T u\right)$$

$$= (1-\lambda)^{n-1} \det(1-\lambda-1)$$

$$= (1-\lambda)^{n-1} \lambda$$

$\Rightarrow$ eigenvalues of A is $(n-1) \times 1$ and 0.

c) Null space of A. is $\{u\}$. spanned by vector u.

d) $A^2 = (I - uu^T)(I - uu^T)$

$$= I - 2uu^T + uu^T uu^T$$

$$= I - 2uu^T + uu^T$$

$$= I - uu^T = A.$$

5. $P = p_1, p_2, \dots p_n \Rightarrow \{p_i\}_{i=1}^{n}$

$Q = q_1, q_2, \dots q_n \Rightarrow \{q_i\}_{i=1}^{n}$

Assume $p_i, q_i \in \mathbb{R}^3$

and $q_i = Rp_i + t$

$$E(R,t) = \sum_{i=1}^{n} \| q_i - (Rp_i + t) \|^2$$

$\bar{p} = \frac{1}{n}\sum p_i$ & $\bar{q} = \frac{1}{n}\sum q_i$

$p_i' = p_i - \bar{p}$ $\quad q_i' = q_i - \bar{q}$ $\quad t = \bar{q} - R\bar{p}$

$$E(R) = \sum_{i=1}^{n} \| q_i' - Rp_i' \|^2$$

$$= \sum_{i=1}^{n} \|q_i'\|^2 - \sum_{i=1}^{n} \|p_i'\|^2 - 2\sum_{i=1}^{n} q_i'^{T} R p_i'$$

set $H = P'Q'^{T} = \sum_{i=1}^{n} p_i' q_i^{T}$

$$\sum_{i=1}^{n} q_i'^{T} R p_i' = Tr\left(R \sum_i p_i' q_i^{T}\right) = Tr(RH)$$

$SVD(H) = U \cdot S \cdot V \Rightarrow Tr(RH) = Tr(RU\Sigma V^{T})$

$= Tr(V^{T}RU\Sigma)$

let set $M = V^{T}RU$

$$Tr(RH) = Tr(M\Sigma)$$

$$= \sum_{i=1}^{3} M_{ii}\sigma_i$$

Since $M_{ii} = I = I$

$$V^T R U = I \qquad R = VU^T$$

HW1

Question1

Assumption: matrix A is square and invertible.

```
A = [2 1 1;
    4 -6 0;
    -2 7 2];

[P, L, D, U] = LDU_Decomposition(A);

disp('P * A =');
```

```
P * A =
```

```
disp(P*A);
```

```
     4    -6     0
     2     1     1
    -2     7     2
```

```
disp('L * D * U =');
```

```
L * D * U =
```

```
disp(L*D*U);
```

```
     4    -6     0
     2     1     1
    -2     7     2
```

```
err = norm(P*A - L*D*U);
fprintf('Verification error = %.2e\n', err);
```

```
Verification error = 0.00e+00
```

```

```

Test 2:

```
A2 = [5, -2, 3;
      1,  0, 2;
      4,  1, 1];

 [P, L, D, U] = LDU_Decomposition(A2);

disp('P * A =');
```

```
P * A =
```

```
disp(P*A2);
```

```
     5    -2     3
     4     1     1
     1     0     2
```

```
disp('L * D * U =');
```

```
L * D * U =
```

```
disp(L*D*U);
```

```
    5.0000   -2.0000    3.0000
    4.0000    1.0000    1.0000
    1.0000         0    2.0000
```

```
err = norm(P*A2 - L*D*U);
fprintf('Verification error = %.2e\n', err);
```

```
Verification error = 4.44e-16
```

Question 2

```
A1 = [10, -10,  0;
       0,  -4,  2;
       2,   0, -5];
[U, S, V] = SVD(A1);

disp('Matrix A1:');
```

```
Matrix A1:
```

```
disp(A1);
```

```
    10   -10     0
     0    -4     2
     2     0    -5
```

```
disp('U matrix:');
```

```
U matrix:
```

```
disp(U);
```

```
   -0.9750   -0.0193    0.2213
   -0.2000    0.5095   -0.8369
   -0.0966   -0.8603   -0.5006
```

```
disp('S matrix:');
```

```
S matrix:
```

```
disp(S);
```

```
   14.4978         0         0
         0    5.9473         0
         0         0    1.8556
```

```
disp('V matrix:');
```

```
V matrix:
```

```
disp(V);
```

```
   -0.6859   -0.3217    0.6528
    0.7277   -0.3102    0.6117
    0.0057    0.8946    0.4469
```

```
disp('Reconstructed A1 from U*S*V'':');
```

```
Reconstructed A1 from U*S*V':
```

```
disp(U*S*V');
```

```
   10.0000  -10.0000    0.0000
    0.0000   -4.0000    2.0000
    2.0000   -0.0000   -5.0000
```

```
disp('Difference between original and reconstructed A1:');
```

```
Difference between original and reconstructed A1:
```

```
disp(norm(A1 - U*S*V'));
```

```
   4.3381e-15
```

A2

```
A2 = [5, -5,  0, 0;
      5,  5,  5, 0;
      0, -1,  4, 1;
      0,  4, -1, 2;
      0,  0,  2, 1];

[U, S, V] = SVD(A2);

% Display results
disp('Matrix A2:');
```

```
Matrix A2:
```

```
disp(A2);
```

```
     5    -5     0     0
     5     5     5     0
     0    -1     4     1
```

```
     0     4    -1     2
     0     0     2     1
```

```
disp('U matrix:');
```

```
U matrix:
```

```
disp(U);
```

```
   -0.1126    0.8671    0.3748   -0.3075   -0.0212
    0.9322    0.1523    0.1626    0.2846    0.0212
    0.2020    0.2258   -0.7497   -0.3169   -0.4957
    0.2399   -0.4123    0.3833   -0.7709   -0.1770
    0.1417    0.0637   -0.3522   -0.3603    0.8497
```

```
disp('S matrix:');
```

```
S matrix:
```

```
disp(S);
```

```
    9.1449         0         0         0
         0    7.7981         0         0
         0         0    4.4207         0
         0         0         0    2.2398
         0         0         0         0
```

```
disp('V matrix:');
```

```
V matrix:
```

```
disp(V);
```

```
    0.4481    0.6536    0.6078   -0.0511
    0.6541   -0.6988    0.2765    0.0867
    0.6028    0.2826   -0.7405    0.0919
    0.0900   -0.0686   -0.0758   -0.9907
```

```
% Verify correctness
disp('Reconstructed A2 from U*S*V'':');
```

```
Reconstructed A2 from U*S*V':
```

```
disp(U*S*V');
```

```
    5.0000   -5.0000   -0.0000    0.0000
    5.0000    5.0000    5.0000   -0.0000
   -0.0000   -1.0000    4.0000    1.0000
    0.0000    4.0000   -1.0000    2.0000
   -0.0000   -0.0000    2.0000    1.0000
```

```
disp('Difference between original and reconstructed A2:');
```

```
Difference between original and reconstructed A2:
```

```
disp(norm(A2 - U*S*V'));
```

```
   8.3107e-15
```

A3

```
A3 = [1, 1, 1, 1;
     10, 2, 9, 0;
      8, 0, 0, 7];


[U, S, V] = SVD(A3);

% Display results
disp('Matrix A3:');
```

```
Matrix A3:
```

```
disp(A3);
```

```
     1     1     1     1
    10     2     9     0
     8     0     0     7
```

```

disp('U matrix:');
```

```
U matrix:
```

```
disp(U);
```

```
    0.1088   -0.0195    0.9939
    0.8344    0.5453   -0.0806
    0.5404   -0.8380   -0.0756
```

```

disp('S matrix:');
```

```
S matrix:
```

```
disp(S);
```

```
   15.4773         0         0         0
         0    7.8292         0         0
         0         0    1.0748         0
```

```

disp('V matrix:');
```

```
V matrix:
```

```
disp(V);
```

```
    0.8254   -0.1623   -0.3883    0.3762
    0.1148    0.1368    0.7747    0.6066
    0.4922    0.6243    0.2496   -0.5528
    0.2514   -0.7518    0.4321   -0.4300
```

```
% Verify correctness
disp('Reconstructed A from U*S*V'':');
```

```
Reconstructed A from U*S*V':
```

```
disp(U*S*V');
```

```
    1.0000    1.0000    1.0000    1.0000
   10.0000    2.0000    9.0000    0.0000
    8.0000    0.0000    0.0000    7.0000
```

```
disp('Difference between original and reconstructed A:');
```

```
Difference between original and reconstructed A:
```

```
disp(norm(A3 - U*S*V'));
```

```
   7.2975e-15
```

Question 3

A1

```
A1 = [10 -10 0;
      0  -4  2;
      2   0  -5];
b1 = [10; 2; 13];

det_A1 = det(A1)
```

```
det_A1 =
160
```

```
[U1, S1, V1] = SVD(A1);

disp('U1 matrix:');
```

```
U1 matrix:
```

```
disp(U1);
```

```
   -0.9750   -0.0193    0.2213
   -0.2000    0.5095   -0.8369
   -0.0966   -0.8603   -0.5006
```

```
disp('S1 matrix:');
```

```
S1 matrix:
```

```
disp(S1);
```

```
   14.4978         0         0
         0    5.9473         0
         0         0    1.8556
```

```
disp('V1 matrix:');
```

```
V1 matrix:
```

```
disp(V1);
```

```
   -0.6859   -0.3217    0.6528
    0.7277   -0.3102    0.6117
    0.0057    0.8946    0.4469
```

```
X1 = V1 * (S1 \ ( U1' * b1))
```

```
X1 = 3×1
   -1.0000
   -2.0000
   -3.0000
```

A2

```
A2 = [1 1 1;
      10 2 9;
      8 0 7];
b2 = [1; 3; 1];

det_A2 = det(A2)
```

```
det_A2 =
1.7764e-15
```

```
[U2, S2, V2] = SVD(A2);

disp('U2 matrix:');
```

```
U2 matrix:
```

```
disp(U2);
```

```
    0.0869    0.5708    0.8165
    0.7859    0.4644   -0.4082
    0.6122   -0.6772    0.4082
```

```
disp('S2 matrix:');
```

```
S2 matrix:
```

```
disp(S2);
```

```
   17.2832         0         0
         0    1.5132         0
         0         0         0
```

```
disp('V2 matrix:');
```

V2 matrix:

```
disp(V2);
```

```
    0.7431   -0.1339    0.6556
    0.0960    0.9910    0.0937
    0.6622    0.0067   -0.7493
```

```
rank2 = rank(A2)
```

```
rank2 =
2
```

```
s2   = diag(S2);
tol = max(size(A)) * eps(max(s2));
idx = (s2 > tol);

x2 = V2(:, idx) * ((U2(:, idx)' * b2) ./ s2(idx))
```

```
x2 = 3×1
    0.0175
    0.8596
    0.1228
```

A3

```
A3 = [1 1 1;
      10 2 9;
      8 0 7];
b3 = [3; 2; 2];

[U3, S3, V3] = SVD(A3);

disp('U3 matrix:');
```

U3 matrix:

```
disp(U3);
```

```
    0.0869    0.5708    0.8165
    0.7859    0.4644   -0.4082
    0.6122   -0.6772    0.4082
```

```
disp('S3 matrix:');
```

S3 matrix:

```
disp(S3);
```

```
   17.2832         0         0
         0    1.5132         0
         0         0         0
```

```
disp('V3 matrix:');
```

```
V3 matrix:
```

```
disp(V3);
```

```
    0.7431   -0.1339    0.6556
    0.0960    0.9910    0.0937
    0.6622    0.0067   -0.7493
```

```
rank3 = rank(A3)
```

```
rank3 =
2
```

```
s3   = diag(S3);
tol = max(size(A)) * eps(max(s3));
idx = (s3 > tol);

x3 = V3(:, idx) * ((U3(:, idx)' * b3) ./ s3(idx))
```

```
x3 = 3×1
    0.0175
    0.8596
    0.1228
```

Question 5

```
rng(0);
P = randn(3,3);

theta = deg2rad(30);
Rz = [cos(theta) -sin(theta) 0;
     sin(theta) cos(theta)   0;
     0               0       1];
t0 = [0.3; -0.4; 0.8];

Q = Rz*P + t0;

[A,t] = transform(P,Q);
disp(A), disp(t), disp(det(A))
```

```
    0.8660   -0.5000    0.0000
    0.5000    0.8660   -0.0000
   -0.0000    0.0000    1.0000

    0.3000
   -0.4000
    0.8000

     1
```

```
r_error = norm(A - Rz)
```

```
r_error =
9.9987e-16
```

```
t_error = norm(t - t0)
```

```
t_error =
6.8664e-16
```

```
function [P,L,D,U] = LDU_Decomposition(A)

    [m, n] = size(A);
    if m ~= n
        error('A must be a square matrix');
    end

    P = eye(m);
    L = eye(m);
    U = A;

    for k = 1:m-1
        [~, idx] = max(abs(U(k:m, k)));
        pivot_row = k + idx - 1;

        if U(pivot_row, k) == 0
            error('Matrix is singular at column %d.', k);
        end

        if pivot_row ~= k
            [P, U, L] = row_swap(U, P, L, pivot_row, k);
        end

        for i = k+1:m
            L(i,k) = U(i,k) / U(k,k);
            U(i,:) = U(i,:) - L(i,k) * U(k,:);
        end
    end

    D = diag(diag(U));
    U = D \ U;
end

function [P_new, U_new, L_new] = row_swap(U, P, L, r2, r1)
    U_new = U;
    P_new = P;
    L_new = L;

    U_new([r1 r2], :) = U_new([r2 r1], :);
    P_new([r1 r2], :) = P_new([r2 r1], :);

    if r1 > 1
        L_new([r1 r2], 1:r1-1) = L_new([r2 r1], 1:r1-1);
    end
end

```

```
function [U, S, V] = SVD(A)

    [m, n] = size(A);
    S = zeros(m, n);

    [U, D] = eig(A * A');
    [~, index] = sort(diag(D), 'descend');
    U = U(:, index);
    D = D(index, index);


    [V, Dv] = eig(A' * A);
    [~, index] = sort(diag(Dv), 'descend');
    V = V(:, index);
    Dv = Dv(index, index);

    r = rank(D);

    S(1:r, 1:r) = sqrt(D(1:r, 1:r));

    V(:, 1:r) = A' * U(:, 1:r) / S(1:r, 1:r);
end

```

```
function [A, t] = transform(P, Q)

    p_avg = mean(P, 2);
    q_avg = mean(Q, 2);

    Pp = P - p_avg;
    Qp = Q - q_avg;

    B = Qp * Pp.';

    [U, ~, V] = SVD(B);


    C = diag([1, 1, sign(det(U*V'))]);
    A = U * C * V.';

    t = q_avg - A * p_avg;
end

```