# Mathematics and Computing Laboratory

**B. Tech.**

**4st Semester**


**Name** : Shobhitha.B

**Roll Number** : 21ETMC412031

**Department** : Mathematics and Statistics

## Faculty of Mathematical & Physical Sciences

**Ramaiah University of Applied Sciences**

## Laboratory Activity – 7

**1. Title of the Laboratory Exercise:** Gram – Schmidt orthogonalization process and QR factorisation

**2. Introduction and Purpose of Experiment**

In this laboratory exercise, students get familiar with orthonormal basis. They also learn to find an orthonormal basis given a set of ordinary vectors using Gram – Schmidt orthogonalization process and decompose the given set of vectors as product of upper triangular matrix and orthonormal matrix using QR factorisation

**3. Aim and Objectives**

Aim

- Obtaining an orthonormal basis from a given set of vectors.

Objectives

At the end of this lab, the student will be able to:

- Apply Gram – Schmidt orthogonalization process to obtain orthonormal basis

**4. Experimental Procedure**

- Students are expected to create a document. Also students are expected obtain an orthonormal basis from the given set of vectors in the script file.

**5. Calculations/Computations/Algorithms.**

Apply Gram–Schmidt orthogonalization process to obtain orthonormal basis and compute QR factorisation from the given set of vectors:

**a)** $\left\{\begin{bmatrix}2\\2\\1\end{bmatrix},\begin{bmatrix}-2\\1\\2\end{bmatrix},\begin{bmatrix}18\\0\\0\end{bmatrix}\right\}$

Program:

LUD.m | powermethod.m | GRM.m

/MATLAB Drive/GRM.m

```
function[]=GRM(A)
[m,n]=size(A);
Q=zeros(m,n);
R=zeros(n,n);
for j=1:n
    v=A(:,j);
    for i=1:j-1
        R(i,j)=Q(:,i)'*A(:,j);
        v=v-R(i,j)*Q(:,i);
    end
      R(j,j)=norm(v);
      Q(:,j)=v/R(j,j);
end
fprintf('Orthonormal set\n Q = \n')
disp(Q)
fprintf('Upper triangular matrix \n R = \n')
disp(R)
```

Command window:

```
Command Window
>> A = [2 2 1;-2 1 2;18 0 0];
>> GRM(A)
```

Results:

```
Orthonormal set
 Q = 
    0.1098    0.8901   -0.4423
   -0.1098    0.4531    0.8847
    0.9879   -0.0486    0.1474

Upper triangular matrix
 R = 
   18.2209    0.1098   -0.1098
         0    2.2334    1.7964
         0         0    1.3270
```

**b)** $\left\{\begin{bmatrix}1\\2\\0\end{bmatrix},\begin{bmatrix}8\\1\\-6\end{bmatrix},\begin{bmatrix}0\\0\\1\end{bmatrix}\right\}$

Command window:

```
>> A = [1 2 0; 8 1 -6; 0 0 1];
>> GRM(A)
```

Result:

```
Orthonormal set
 Q =
    0.1240    0.9923         0
    0.9923   -0.1240    0.0000
         0         0    1.0000

Upper triangular matrix
 R =
    8.0623    1.2403   -5.9537
         0    1.8605    0.7442
         0         0    1.0000
```

**c)** $\left\{\begin{bmatrix}1\\-1\\1\end{bmatrix},\begin{bmatrix}1\\0\\1\end{bmatrix},\begin{bmatrix}1\\1\\2\end{bmatrix}\right\}$

Command window:

```
>> A = [1 -1 1; 1 0 1; 1 1 2];
>> GRM(A)
```

Result:

```
Orthonormal set
 Q =
    0.5774   -0.7071    0.4082
    0.5774         0   -0.8165
    0.5774    0.7071    0.4082

Upper triangular matrix
 R =
    1.7321         0    2.3094
         0    1.4142    0.7071
         0         0    0.4082
```

**d)** $\left\{\begin{bmatrix}2\\2\\1\end{bmatrix},\begin{bmatrix}-2\\1\\2\end{bmatrix},\begin{bmatrix}18\\0\\0\end{bmatrix}\right\}$

Command window:

```
Command Window
>> A = [2 2 1;-2 1 2;18 0 0];
>> GRM(A)
```

Result:

```
Orthonormal set
 Q =
    0.1098    0.8901   -0.4423
   -0.1098    0.4531    0.8847
    0.9879   -0.0486    0.1474

Upper triangular matrix
 R =
   18.2209    0.1098   -0.1098
         0    2.2334    1.7964
         0         0    1.3270
```

**e)** $\left\{\begin{bmatrix} 2 \\ -5 \\ 1 \end{bmatrix}, \begin{bmatrix} 4 \\ -1 \\ 2 \end{bmatrix}\right\}$

Command window:

```
>> A = [2 -5 1; 4 -1 2];
>> GRM(A)
```

Result:

```
Orthonormal set
 Q =
    0.4472   -0.8944   -0.8944
    0.8944    0.4472    0.4472

Upper triangular matrix
 R =
    4.4721   -3.1305    2.2361
         0    4.0249   -0.0000
         0         0    0.0000
```

**f)** $\left\{\begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 3 \\ -1 \end{bmatrix}, \begin{bmatrix} -3 \\ 7 \\ 1 \\ 3 \end{bmatrix}\right\}$

Command window:

```
>> A = [1 -1 1 1; 1 1 3 -1 ; -3 7 1 3];
>> GRM(A)
```

Result:

```
Orthonormal set
 Q =
    0.3015    0.2752   -0.4472    0.9748
    0.3015    0.8808   -0.8944   -0.1393
   -0.9045    0.3853         0    0.1741

Upper triangular matrix
 R =
    3.3166   -6.3317    0.3015   -2.7136
         0    3.3029    3.3029    0.5505
         0         0    0.0000    0.4472
         0         0         0    1.9149
```

**g)** $\left\{\begin{bmatrix}1\\1\\1\\1\end{bmatrix}, \begin{bmatrix}-1\\4\\4\\-1\end{bmatrix}, \begin{bmatrix}4\\-2\\2\\0\end{bmatrix}\right\}$

Command window:

```
>> A = [1 1 1 1; -1 4 4 -1; 4 -2 2 0];
>> GRM(A)
```

Result:

```
Orthonormal set
 Q =
    0.2357    0.4264   -0.8733   -0.5494
   -0.2357    0.8969    0.3743   -0.1374
    0.9428    0.1176    0.3119   -0.8242

Upper triangular matrix
 R =
    4.2426   -2.5927    1.1785    0.4714
         0    3.7786    4.2491   -0.4705
         0         0    1.2476   -1.2476
         0         0         0    0.0000
```

## 6. Analysis and Discussions:

To apply the Gram-Schmidt orthogonalization process to obtain an orthonormal basis and compute the QR factorization:

- Define the input vectors.
- Perform Gram-Schmidt orthogonalization.
- Normalize the orthogonalized vectors to obtain an orthonormal basis.
- Construct the Q matrix from the orthonormal basis.
- Construct the R matrix from the Q matrix and the original vectors.

Gram-Schmidt orthogonalization is a mathematical process used to transform a set of linearly independent vectors into a set of orthogonal (perpendicular) or orthonormal (orthogonal and normalized to have a length of 1) vectors. This technique is particularly important in linear algebra and numerical computations.

**7. Conclusions:**

- Gram-Schmidt Orthogonalization: The Gram-Schmidt orthogonalization process is used to transform a set of vectors into an orthogonal or orthonormal basis.
- Orthogonalization is a mathematical process used to transform a set of vectors into a new set of vectors that are orthogonal (perpendicular) to each other.
- In linear algebra, orthogonal vectors have a dot product of zero, which means they are geometrically perpendicular in Euclidean space.
- Normalization: To obtain an orthonormal basis, we normalized the orthogonal vectors, resulting in the orthonormal vectors.
- QR Factorization: We constructed the Q matrix from the orthonormal basis vectors and computed the R matrix using the Q matrix and the original vectors.
- The Gram-Schmidt orthogonalization process was successfully applied to the given set of vectors, resulting in an orthonormal basis represented by the Q matrix and the upper triangular R matrix.

**8. Comments**

| **9. Particulars** | **Marks** | |
|---|---|---|
| | **Maximum** | **Actual** |
| **Results** | 12 | |
| **Procedures/Steps /Write up** | 05 | |
| **Viva** | 08 | |
| **Total** | **25** | |

**Signature of Staff in-charge**

## Laboratory Activity – 4

**Title of the Laboratory Exercise:** Consistency of system of equations

1. **Introduction and Purpose of Experiment:**

   In this laboratory exercise, students get familiar with different solution for different matrix. They also learn to solve the equation and find the consistency.

2. **Aim and Objectives:**

   <u>Aim</u>

   - Solving the system of equations to know the consistency of the matrix.

   <u>Objectives</u>

   At the end of this lab, the student will be able to:

   - To solve system of equations.

3. **Experimental Procedure:**

   - Students are expected to create a document. Also students are expected to solve the system of equations in the script file.

4. **Calculations/Computations/Algorithms:**

   Solve the following system of equations using Thomas algorithm:

**a)** $2x_1 - x_2 = 1;\ -x_1 + 2x_2 - x_3 = 0;\ -x_2 + 2x_3 - x_4 = 0;\ -x_3 + 2x_4 = 1$

<u>Program</u>:

/MATLAB Drive/syseq.m

```
function[]=syseq(A,B)
[~,n]=size(A);
rA=rank(A);
M=[A,B];
rM=rank(M);
if rA==rM
    fprintf('The system of eqns is consistent')
    if rA==n
        fprintf('\n The system has unique solution and soln is:\n')
        X=A\B;
        disp(X)
    else
        fprintf('\n The system has infinte solutions \n')
    end
else
    fprintf('\n The system of eqns is inconsistent \n')
end
end
```

<u>Command window</u>:

```
>> A = [2 -1 0 0;-1 2 -1 0 ; 0 -1 2 -1; 0 0 -1 2];
>> B = [1 ; 0 ; 0 ; 1];
>> syseq(A,B)
```

Result:

```
The system of eqns is consistent
 The system has unique solution and soln is:
    1.0000
    1.0000
    1.0000
    1.0000
```

**b)** $2.04x_1 - x_2 = 40.8;\ -x_1 + 2.04x_2 - x_3 = 0.8;\ -x_2 + 2.04x_3 - x_4 = 0.8;\ -x_3 + 2.04x_4 = 200.8$

Command window:

```
>> A = [2.04 -1 0 0;-1 2.04 -1 0; 0 -1 2.04 -1 ; 0 0 -1 2.04];
>> B = [40.8;0.8;0.8;200.8];
>> syseq(A,B)
```

Result:

```
The system of eqns is consistent
 The system has unique solution and soln is:
   65.9698
   93.7785
  124.5382
  159.4795
```

**c)** $2x_1 + x_2 = 1; 2x_1 + 3x_2 + x_3 = 2; x_2 + 4x_3 + 2x_4 = 3; x_3 + 3x_4 = 4$

Command window:

```
>> A = [2 1 0 0; 2 3 1 0; 0 2 4 2; 0 0 1 3];
>> B = [1;2;3;4];
>> syseq(A,B)
```

Result:

```
The system of eqns is consistent
 The system has unique solution and soln is:
    0.1786
    0.6429
   -0.2857
    1.4286
```

**d)** $4x + 8y = 8; 8x + 18y + 2z = 18; 2y + 5z + 1.5w = 0.5; 1.5z + 1.75w = -1.75$

Command window:

```
>> A = [4 8 0 0; 8 18 2 0; 0 2 5 1.5; 0 0 1.5 1.75];
>> B = [8;18;0.5;-1.75];
>> syseq(A,B)
```

Result:

```
The system of eqns is consistent
 The system has unique solution and soln is:
     0
     1
     0
    -1
```

**5. Analysis and Discussions:**

- We calculate the rank of matrix A (rank(A)) and the rank of the augmented matrix [A|b] (rank augmented). The rank of a matrix represents the number of linearly independent rows or columns.
- If rank(A) is equal to rank augmented and is equal to the number of unknowns (columns in A), then the system is consistent and has a unique solution.
- If rank(A) is equal to rank augmented but less than the number of unknowns, then the system is consistent but has infinitely many solutions (underdetermined).
- If rank(A) is not equal to rank augmented, the system is inconsistent and has no solution.

**6. Conclusions:**

By comparing the ranks of the coefficient matrix "A" and the augmented matrix [A|b], we can conclude whether the system has a unique solution, infinitely many solutions, or no solution at all. This is a valuable tool for solving and analyzing systems of linear equations in various mathematical and engineering applications.

**7. Comments:**

| **8. Particulars** | **Marks** | |
|---|---|---|
| | **Maximum** | **Actual** |
| **Results** | 12 | |
| **Procedures/Steps /Write up** | 05 | |
| **Viva** | 08 | |
| **Total** | **25** | |

**Signature of Staff in-charge**