DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING,
FACULTY OF ECE,
Rajshahi University of Engineering & Technology, Bangladesh

EEE - 3212– Power System I Sessional

Sessional Report

Student Lab Report

Submitted by

Ashraf Al- Khalique
Roll: 1801171
Session: 2018-2019
Dept. of Electrical & Electronic Engineering,
Rajshahi University of Engineering and Technology.

Table of Contents

Experiment No. 01

1.1 Experiment Name

Introduction to MATLAB programming

1.2 Objectives

- To become acquainted with the MATLAB functions and necessary parameters
- To learn how to implement MATLAB code to a system using computational methods

1.3 Apparatus

- MATLAB

1.4 Problem

(i)

Roll	CT1	CT2	CT3	CT4	CT5	CT6
1801170	10	13	14	12	16	15
1801171	11	14	15	18	15	14
1801172	14	10	15	18	17	14
1801173	16	12	15	20	17	14
1801174	12	11	14	18	16	14

(ii)

A	5	0	-10	27	1	15

1.5 MATLAB Code

1.5.1 For Problem (i)

```
clc; %Clears previous data from command window
clear all; %Removes all variables from the current workspace

cd('F:\Study material\Lab\3-2\Power System I'); %Changes file directory
x = xlsread('Exp01') %Imports data from excel file
fprintf('\n Roll No        \tCTI       CT2       CT3        CT4       CT5       CT6\n')
%Display the text
disp(x) %Display the data inside variable

n=length(x) ; %Determines the number of column

y = x(:,2:n) ; %Isolates the data to be averaged from the roll
fprintf('Marks:\n') %Display the text
disp(y) %Display the data inside variable

w = sort(y,2,'descend'); %Rearranging the columns in descending order. 2 is for descending
rawwise
```

```
fprintf('\nSorting descending order rawwise: \n') %Display the text
disp(w) %display the data inside variable

z = w(:,1:3) %Takes the first three columns containing highest three marks
fprintf('\nBest three marks: \n') %Display the text
disp(z) %Display the data inside variable

m = mean(z,2) %Calculates mean of the highest three marks. 2 is for doing the action
rawwise
fprintf('\nAverage marks: \n') %Display the text
disp(m) %Display the data inside variable

Output = round(m) %Round the calculated data
fprintf('Rounding the average marks: \n') %Display the text
disp(Output) %display the value inside variable

Roll=x(:,1) %Taking the column of Roll
Y=[Roll Output] %Forming a matrix of column Roll and Attained data as marks
fprintf(' Roll No Attained Marks \n') %Display the text
disp(Y) %Display the marks inside variable
```

1.5.2 For Problem (ii)

```
clc; %Clears previous data from command window
clear all; %Removes all variables from the current workspace

cd('F:\Study material\Lab\3-2\Power System I'); %Changes file directory
Matrix=xlsread('Exp01p02'); %Reads from excel file
fprintf('Matrix:'); %Prints the data
disp(Matrices) %Shows the output

n=length(Matrix); %Determines the number of elements

%Ascending
for j=1:n %Campare first elements
   for k=j+1:n %Campare second elements
      if Matrix(j)>=Matrix(k) %Compare greater or not
         m=Matrix(j); %Store the greater number in a variable
         Matrix(j)=Matrix(k); %Replace the greater number by the smaller one
         Matrix(k)=m; %Replace the smaller number with greater number
      end
   end
end
fprintf('Ascending: '); %Print the data in desired order

%Descending
disp(Matrix) %Show the output
Output=xlsread('Exp01p02'); %Read from excel file
n=length(Output); %Read the number of elements
for j=1:n %Campare first elements
   for k=j+1:n %Campare second elements
      if Output(j)<=Output(k) %Compare samller or not
```

```
            m=Output(j);%Store the smaller number in a variable
            Output(j)=Output(k);%Replace the smaller number by the smaller one
            Output(k)=m; %Replace the greater number with smaller number
        end
    end
end
fprintf('Descending: '); %Printing the data
disp(Output) %Show the output
```

1.6 Output

1.6.1 For Problem (i)

```
x =

1801170      10      13      14      12      16      15
1801171      11      14      15      18      15      14
1801172      14      10      15      18      17      14
1801173      16      12      15      20      17      14
1801174      12      11      14      18      16      14
1801175      15      10      14      17      19      14


Roll No      CTI      CT2     CT3      CT4     CT5     CT6
1801170      10      13      14      12      16      15
1801171      11      14      15      18      15      14
1801172      14      10      15      18      17      14
1801173      16      12      15      20      17      14
1801174      12      11      14      18      16      14
1801175      15      10      14      17      19      14

Marks:
10   13   14   12   16   15
11   14   15   18   15   14
14   10   15   18   17   14
16   12   15   20   17   14
12   11   14   18   16   14
15   10   14   17   19   14

Sorting descending order rawwise:
16   15   14   13   12   10
18   15   15   14   14   11
18   17   15   14   14   10
20   17   16   15   14   12
18   16   14   14   12   11
19   17   15   14   14   10

z =

16   15   14
```

```
18    15    15
18    17    15
20    17    16
18    16    14
19    17    15

Best three marks:
16    15    14
18    15    15
18    17    15
20    17    16
18    16    14
19    17    15

m =

15.0000
16.0000
16.6667
17.6667
16.0000
17.0000

Average marks:
15.0000
16.0000
16.6667
17.6667
16.0000
17.0000

Output =

15
16
17
18
16
17

Rounding the average marks:
15
16
17
18
16
17

Roll =
1801170
1801171
1801172
1801173
1801174
1801175

Y =

1801170         15
1801171         16
1801172         17
1801173         18
1801174         16
1801175         17

Roll No Attained Marks
1801170         15
1801171         16
1801172         17
1801173         18
1801174         16
1801175         17
```

1.6.2 For Problem (ii)

```
Matrix:     5     0   -10    27     1    15    30

Ascending:    -10     0     1     5    15    27    30

Descending:     30    27    15     5     1     0   -10
```

1.7 Discussion & Conclusion

We used MATLAB code to solve the problem in this experiment. In the first problem, which is quite practical, we find the average class test score for six different students. In this case, we calculated the output for each student by taking the best three marks and averaging them. The elements of a 7x1 matrix were sorted in both ascending and descending order in the second problem.

Through this experiment, we become acquainted with the implementation and scope of MATLAB in computational methods and systems by solving these problems. Thus the objective of the experiment was achieved.

Experiment No. 02

2.1 Experiment Name

Basic concept development on the idea of Simulink using MATLAB

2.2 Objectives

- To become acquainted with the Simulink platform and its' libraries
- To learn how to implement Simulink libraries to design a system using MATLAB Simulink
- To get familiar with the procedure of designing a power system in Simulink

2.3 Apparatus

- Simulink

2.4 Theory

2.4.1 Simulink

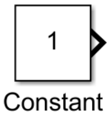
Simulink is an application that allows you to simulate signals and dynamic systems. Simulink contains toolboxes for developing, simulating, and analyzing communication systems. In addition, source coding, channel coding, interleaving, analog and digital modulation, equalization, synchronization, and channel modeling are all possible with Simulink.

2.4.2 Simulink Library

The Simulink Library Browser is the library where you can locate all the Simulink blocks. Simulink software contains a large library of functions that are often used in system modeling. For this experiment following libraries were used,

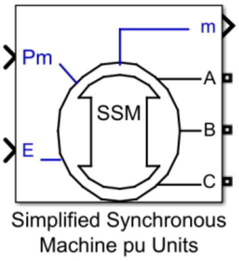
Constant

It helps output the constant specified by the 'Constant value' parameter.



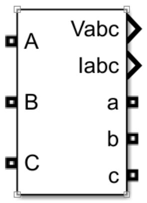
Simplified Synchronous Machine pu Units

Implements a 3-phase simplified synchronous machine. Machine is modeled as an internal voltage behind a R-L impedance. Stator windings are connected in wye to an internal neutral point.



Three-Phase V-I Measurement

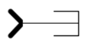
It's an ideal three-phase voltage and current measurements block. The block can output the voltages and currents in per unit values or in volts and amperes.



Three-Phase
V-I Measurement

Terminator

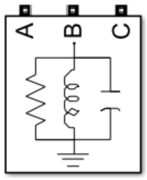
This block is used to "terminate" output signals & prevent warnings about unconnected output ports.



Terminator

Three-Phase Parallel RLC Load

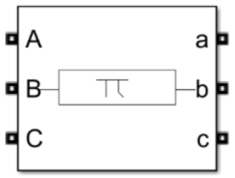
This block implements a three-phase parallel RLC load.



Three-Phase
Parallel RLC Load

Three-Phase PI Section Line

This block models a three-phase transmission line with a single PI section. The model consists of one set of RL series elements connected between input and output terminals and two sets of shunt capacitances lumped at both ends of the line.



Three-Phase
PI Section Line

Powergui

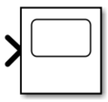
This block help set simulation type, simulation parameters, and preferences.

Continuous

powergui

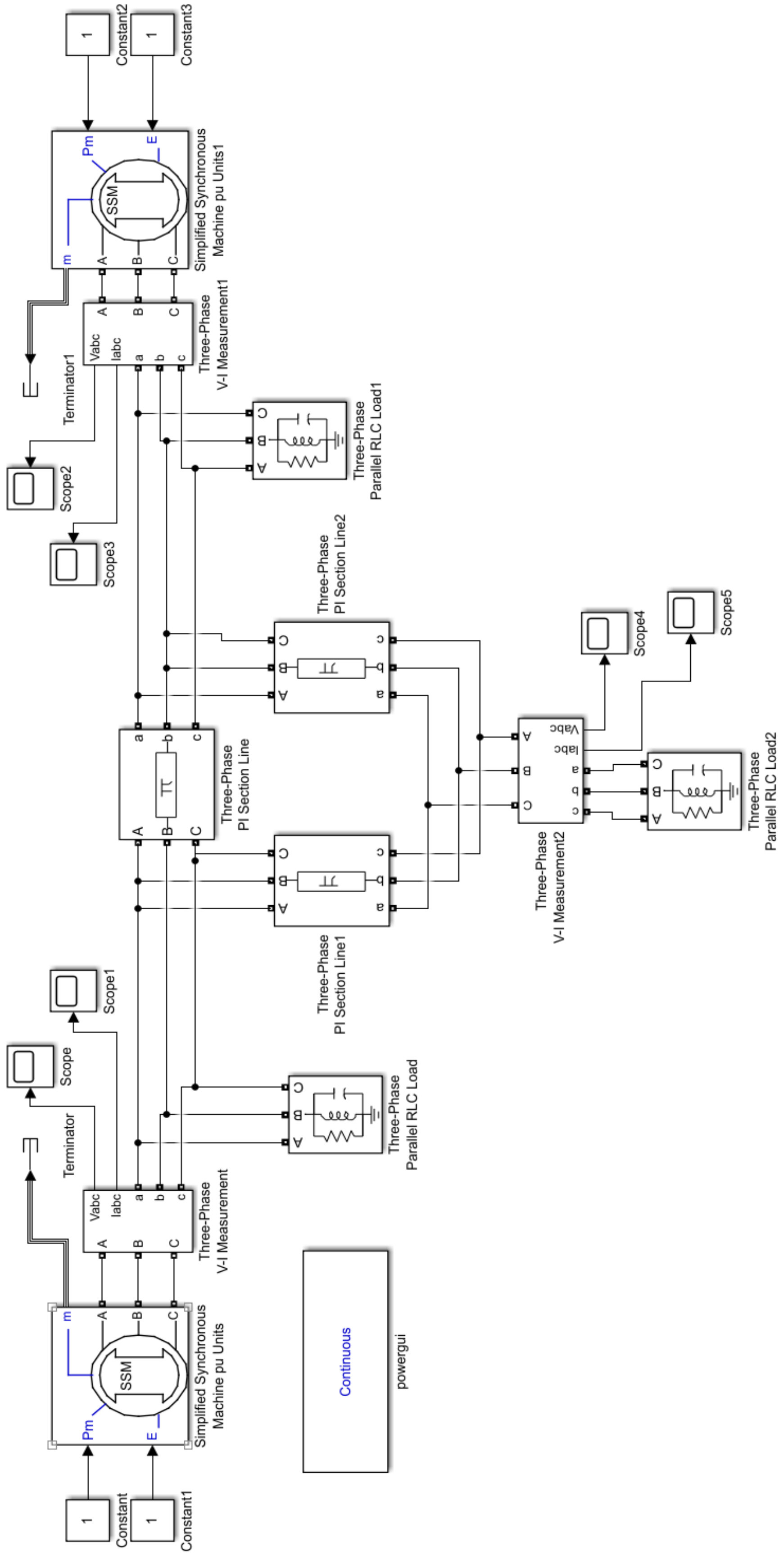
Scope

Its display’s output waveform.



Scope

2.5 Block Diagram



Constant2
Constant3
1
Simplified Synchronous Machine pu Units1
SSM
Pm
E
m
A
B
C
Terminator1
Three-Phase V-I Measurement1
Vabc
Iabc
a
b
c
Scope2
Scope3
Three-Phase Parallel RLC Load1
Three-Phase PI Section Line2
Three-Phase PI Section Line
Three-Phase PI Section Line1
Scope4
Scope5
Three-Phase V-I Measurement2
Three-Phase Parallel RLC Load2
Scope1
Scope
Terminator
Three-Phase V-I Measurement
Three-Phase Parallel RLC Load
Simplified Synchronous Machine pu Units
Constant
Constant1
Continuous
powergui

2.6 Output

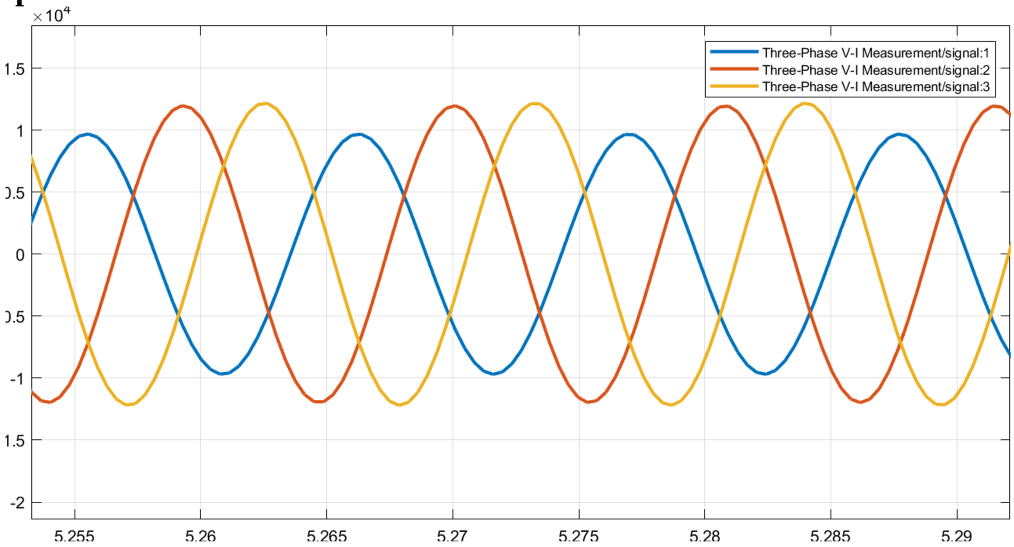


Fig 2.1: Three- Phase Generator no.1 voltage signals

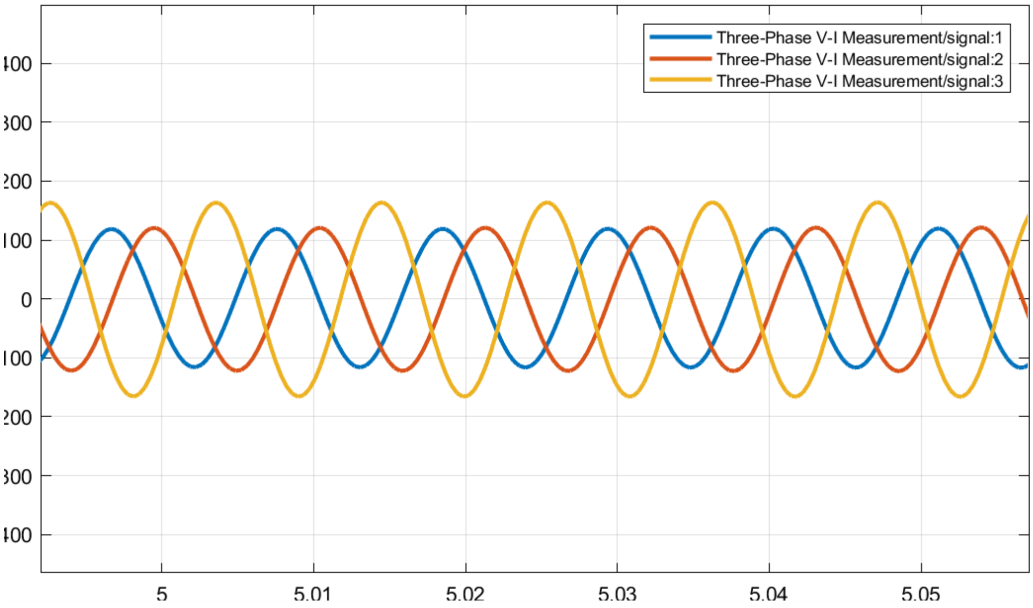


Fig 2.2: Three- Phase Generator no.1 current signals

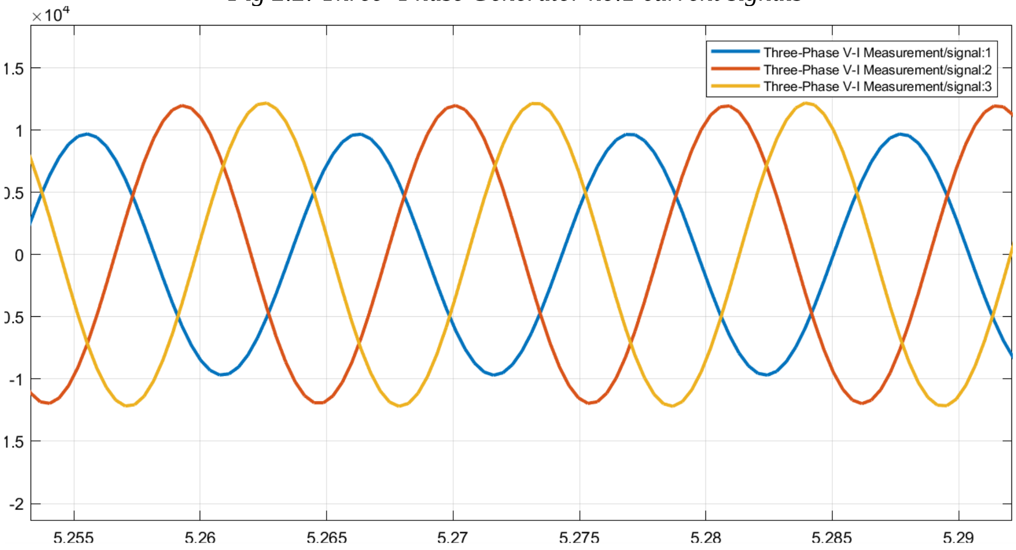


Fig 2.3: Three- Phase Generator no.2 voltage signals

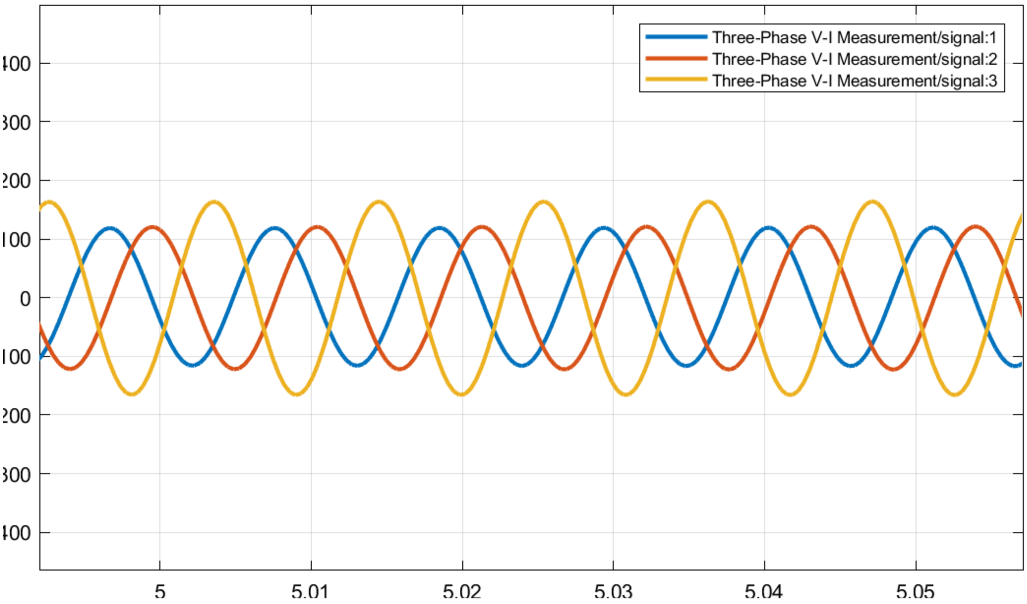


Fig 2.4: Three- Phase Generator no.2 current signals

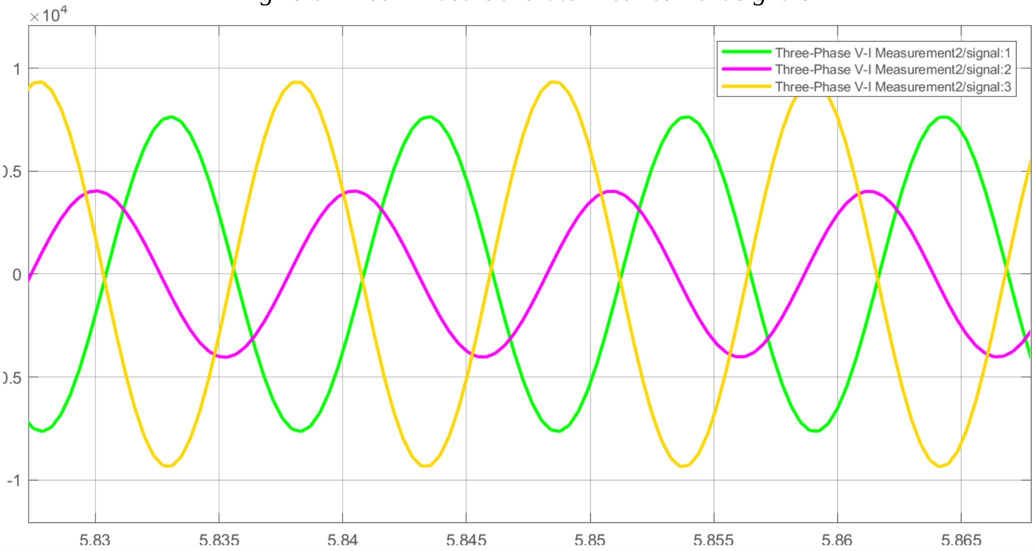


Fig 2.5: Three- Phase parallel RLC voltage signals

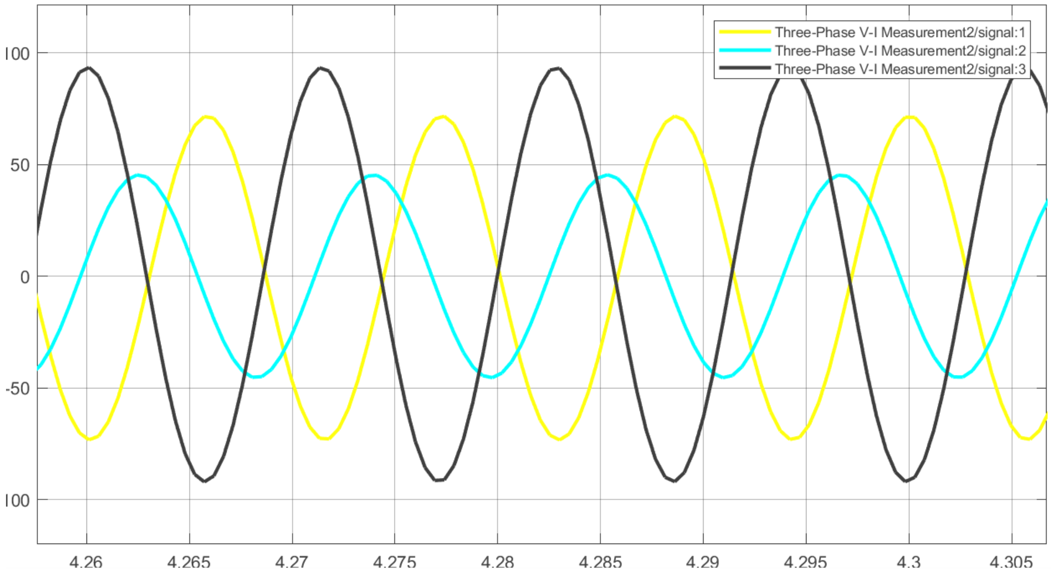


Fig 2.6: Three- Phase parallel RLC voltage signals

2.7 Discussion & Conclusion

In this experiment, we used the Simulink platform to design a three-bus power system. Various libraries and functions were implemented throughout the process. As a result, we become acquainted with the implementation and scope of the Simulink platform, as well as its libraries and functionalities. We also comprehended the system design process. As a result, the experiment's goal was achieved.

Experiment No. 03

3.1 Experiment Name

Generate an algorithm and write a program to calculate the Y-bus matrix of a given power system

3.2 Objectives

- To become acquainted with the Y-bus matrix of a given power system
- To learn how to generate a MATLAB code for a Y-bus matrix of a given power system
- To get familiar with the procedure of designing and analyzing a power system in MATLAB

3.3 Apparatus

- MATLAB

3.4 Theory

The Y-bus matrix is a N × N matrix used in power engineering to describe a power system with N buses. It reflects the buses' nodal admittance in a power system. The Y matrix is relatively sparse in realistic systems with thousands of buses.

In a practical power system, each bus is often connected to only a few other buses via transmission lines. The Y Matrix is also one of the data requirements needed to formulate a power flow study.

One of the strategies developed to study power systems is the Node voltage method. In terms of load currents, the equations in the nodal admittance form result in a simultaneous complex algebraic equation. The voltages and currents of the buses are obtained by solving these equations. The power and load flow on all buses in the system may then be determined. That is why the construction of the admittance bus matrix Y-bus is very important.

If I_{bus} is the bus currents in matrix form and Y_{bus} is the admittance matrix and V bus is the bus voltages in matrix form then they can be related as I_{bus}=$Y_{bus}V_{bus}$

The general mathematical expressions are

$$Yij = -yij \text{ ; if } i \neq j$$

$$Yij = \sum_{k=1}^{n} y_{ik} \text{ ; if } i=j$$

The nodal admittance matrix form

$$Y = \begin{bmatrix} y_{11} & \cdots & y_{1n} \\ \cdots & \cdots & \cdots \\ y_{1n} & \cdots & y_{nn} \end{bmatrix}$$

Before constructing the equations that comprise the Y Matrix, three major steps must be taken beginning with a single line diagram of a power system. The single line diagram is first transformed into an impedance diagram. Following that, all voltage sources are translated to their corresponding current sources. The impedance diagram is then transformed into an admittance diagram.

The bus impedances can be recorded in an excel file for better management. The excel file can then be put into MATLAB to create the admittance matrix.

3.5 Block diagram

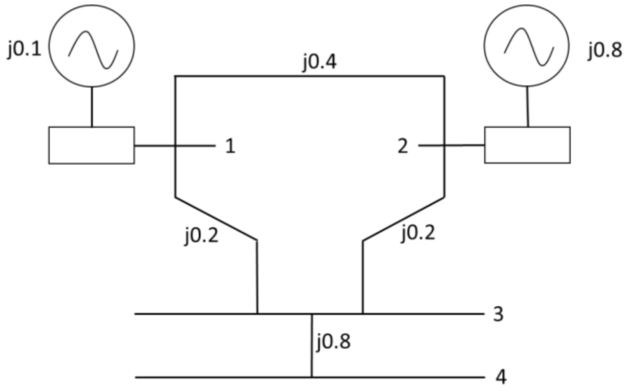


Fig 3.1: Diagram of Y- bus no.1

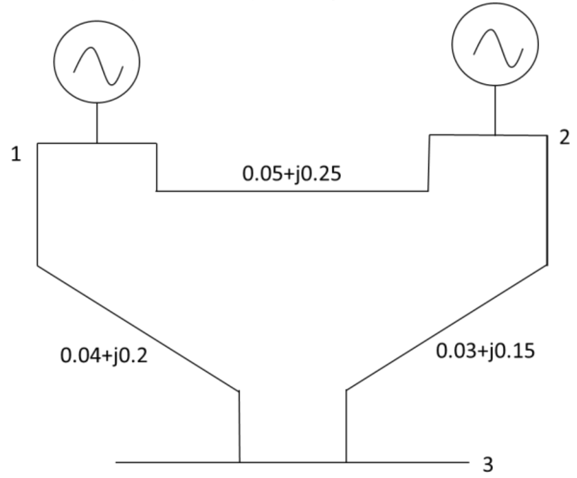


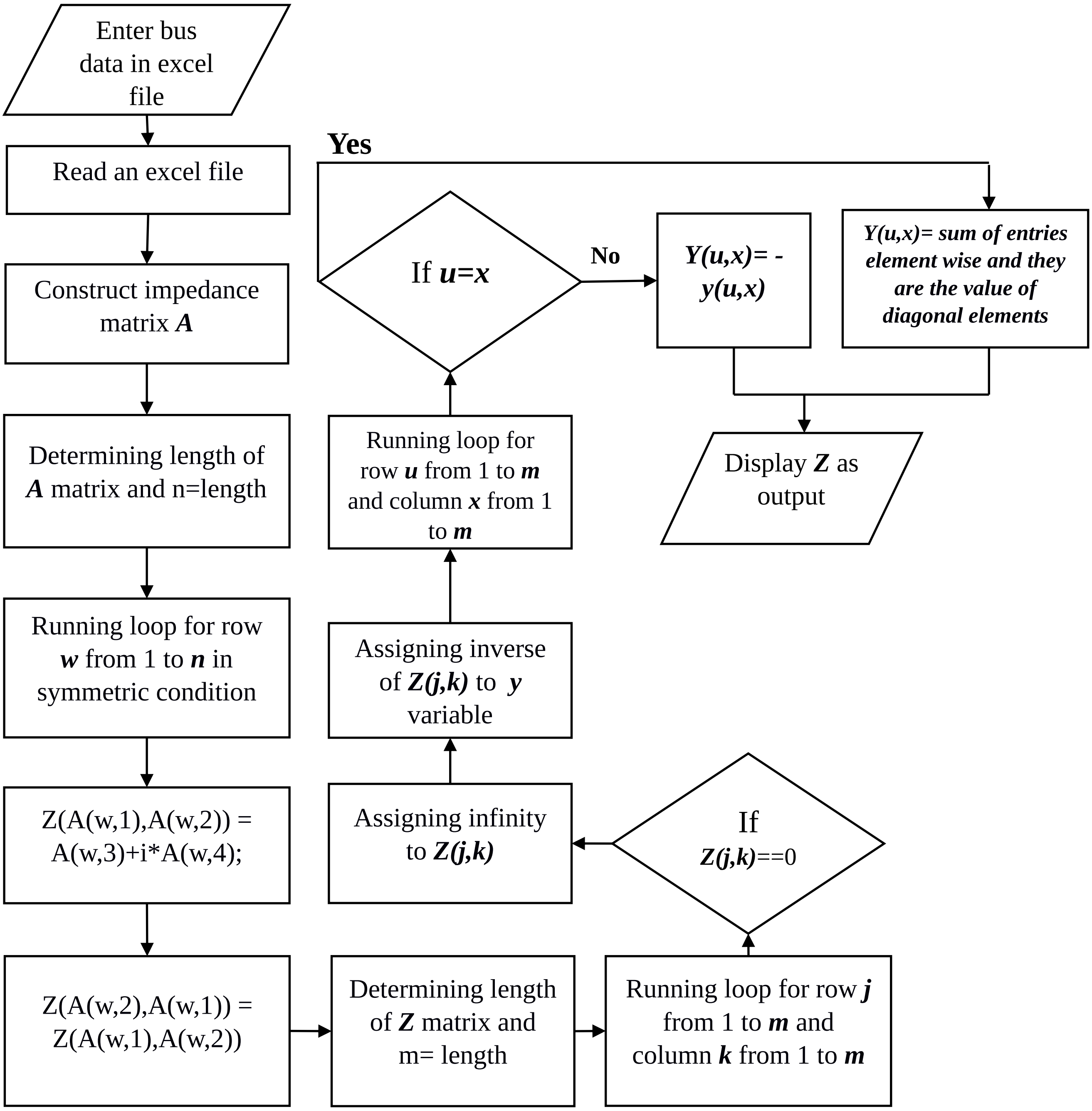
Fig 3.1: Diagram of Y- bus no.2

3.6 Algorithm

1. Start
2. Read an excel file that has bus numbers ***i*** and ***j*** in the first two columns and resistance and admittance value in the third and fourth columns which represent impedance between the buses
3. Construct a matrix ***A*** whose element $a_{i,j}$ denotes the impedance between ***i*** and ***j*** buses
4. Determine the length of matrix ***A***
5. Perform symmetric condition and construct a new matrix ***Z*** where 3rd and 4th column element of the excel file is equivalent to ***a+bj*** and it can be assigned to $Z_{i,j}$ later applying symmetric condition $Z_{i,j} = Z_{j,i}$
6. Determine the length of new matrix ***Z***
7. Use looping condition for row j from 1 to length of Z and for column k from1 to length Z

8. If value of ***j*** row and ***k*** column is equal to zero, assign the element value as infinity
9. Determine inverse matrix of ***Z*** and assigned to ***Z***
10. Calculate the summation of ***Y*** matrix element row wise
11. Use looping condition for row u from1 to length of ***Z*** and for column x from1 to length ***Z***
12. Use formula if ***u≠x*** $Y_{ux}= -y_{ux}$, else $Y_{ux} = \sum_{k=1}^{n} y_{ux}$
13. Display ***Y*** as output
14. End

3.7 Flow chart


Enter bus data in excel file
Read an excel file
Construct impedance matrix A
Determining length of A matrix and n=length
Running loop for row w from 1 to n in symmetric condition
Z(A(w,1),A(w,2)) = A(w,3)+i*A(w,4);
Z(A(w,2),A(w,1)) = Z(A(w,1),A(w,2))
Determining length of Z matrix and m= length
Running loop for row j from 1 to m and column k from 1 to m
If Z(j,k)==0
Assigning infinity to Z(j,k)
Assigning inverse of Z(j,k) to y variable
Running loop for row u from 1 to m and column x from 1 to m
If u=x
Yes
No
Y(u,x)= -y(u,x)
Y(u,x)= sum of entries element wise and they are the value of diagonal elements
Display Z as output

3.8 MATLAB Code & Output

- **Generalized MATLAB Code for Y bus no.1**

```
clc;  %Clears previous data from command window
clear all; %Removes all variables from the current workspace
cd('F:\Study material\Lab\3-2\Power System I'); %change the
file directory
A = xlsread('Exp02'); %Read the excel file
n = length(A); %Determine the length of the excel file

% Applying symmetric condition
for w=1:n
     Z(A(w,1),A(w,2)) = A(w,3)+i*A(w,4);
     Z(A(w,2),A(w,1)) = A(w,3)+i*A(w,4);
end

m = length(Z) %Determine the length of the new matrix
for j=1:m
     for k=1:m
         if Z(j,k) == 0
            Z(j,k) = inf;
         end
     end
end
fprintf(' Z matrix is \n') %Display the text
disp(Z) %Display the output
y = 1./Z %Taking inverse impedance matrix
p = sum(y,2) %Taking symmetric summation

%Apply looping condition to determine value of the matrix
element
for u=1:m
     for x=1:m
         if u~=x
            Y(u,x)= -y(u,x); %For diagonal element
         else
            Y(u,x)= p(u); %For non-diagonal element
         end
     end
end
fprintf(' Y- bus matrix is \n') %Display the text
disp(Y) %Display the output
```

- **Output**

```
m =

   4
```

```
 Z matrix is

   0.0000 + 0.1000i   0.0000 + 0.4000i   0.0000 + 0.2000i      Inf + 0.0000i

   0.0000 + 0.4000i   0.0000 + 0.8000i   0.0000 + 0.2000i      Inf + 0.0000i

   0.0000 + 0.2000i   0.0000 + 0.2000i      Inf + 0.0000i   0.0000 + 0.0800i

      Inf + 0.0000i      Inf + 0.0000i   0.0000 + 0.0800i      Inf + 0.0000i

y =

   0.0000 -10.0000i   0.0000 - 2.5000i   0.0000 - 5.0000i   0.0000 + 0.0000i

   0.0000 - 2.5000i   0.0000 - 1.2500i   0.0000 - 5.0000i   0.0000 + 0.0000i

   0.0000 - 5.0000i   0.0000 - 5.0000i   0.0000 + 0.0000i   0.0000 -12.5000i

   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 -12.5000i   0.0000 + 0.0000i

p =

   0.0000 -17.5000i

   0.0000 - 8.7500i

   0.0000 -22.5000i

   0.0000 -12.5000i

 Y- bus matrix is

   0.0000 -17.5000i   0.0000 + 2.5000i   0.0000 + 5.0000i   0.0000 + 0.0000i

   0.0000 + 2.5000i   0.0000 - 8.7500i   0.0000 + 5.0000i   0.0000 + 0.0000i

   0.0000 + 5.0000i   0.0000 + 5.0000i   0.0000 -22.5000i   0.0000 +12.5000i

   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +12.5000i   0.0000 -12.5000i
```

- **Generalized MATLAB Code for Y bus no.2**

```
clc;  %Clears previous data from command window
clear all; %Removes all variables from the current workspace
cd('F:\Study material\Lab\3-2\Power System I'); %change the
file directory
A = xlsread('Exp02'); %Read the excel file
n = length(A); %Determine the length of the excel file

% Applying symmetric condition
for w=1:n
```

```
    Z(A(w,1),A(w,2)) = A(w,3)+i*A(w,4);
    Z(A(w,2),A(w,1)) = A(w,3)+i*A(w,4);
end

m = length(Z) %Determine the length of the new matrix
for j=1:m
    for k=1:m
        if Z(j,k) == 0
            Z(j,k) = inf;
        end
    end
end
fprintf(' Z matrix is \n') %Display the text
disp(Z) %Display the output
y = 1./Z %Taking inverse impedance matrix
p = sum(y,2) %Taking symmetric summation

%Apply looping condition to determine value of the matrix
element
for u=1:m
    for x=1:m
        if u~=x
            Y(u,x)= -y(u,x); %For diagonal element
        else
            Y(u,x)= p(u); %For non-diagonal element
        end
    end
end
fprintf(' Y- bus matrix is \n') %Display the text
disp(Y) %Display the output
```

- **Output**

```
m =

   3

 Z matrix is

  0.0000 + 1.0000i   0.0500 + 0.2500i   0.0400 + 0.0200i

  0.0500 + 0.2500i   0.0000 + 1.0000i   0.0300 + 0.1500i

  0.0400 + 0.0200i   0.0300 + 0.1500i      Inf + 0.0000i




y =

  0.0000 - 1.0000i   0.7692 - 3.8462i  20.0000 -10.0000i
```

```
   0.7692 - 3.8462i   0.0000 - 1.0000i   1.2821 - 6.4103i

  20.0000 -10.0000i   1.2821 - 6.4103i   0.0000 + 0.0000i

p =

  20.7692 -14.8462i

   2.0513 -11.2564i

  21.2821 -16.4103i

 Y- bus matrix is

  20.7692 -14.8462i  -0.7692 + 3.8462i -20.0000 +10.0000i

  -0.7692 + 3.8462i   2.0513 -11.2564i  -1.2821 + 6.4103i

 -20.0000 +10.0000i  -1.2821 + 6.4103i  21.2821 -16.4103i
```

3.9 Discussion & Conclusion

In this experiment, we designed an algorithm, flow chart, and programmed a generalized code for two different Y- bus system. In each case, we extracted the values from an excel file and formulated necessary condition to assign values to variables. Through this generalized coding format, we easily design and calculate Y bus matrix for any given power system.

The only adjustment to the code we may need is changing the directory of the file to work with and the given data saved inside the file. The bus numbers and the resistance and reactance values must also be given in the order defined for the code to work and give accurate result.

Experiment No. 04

4.1 Experiment Name

Generate an algorithm and write a program to calculate the Y-bus reduction matrix of a given power system

4.2 Objectives

- To become acquainted with the Y-bus and its reduction matrix of a given power system
- To learn how to generate a MATLAB code for a Y-bus reduction matrix of a given power system
- To get familiar with the procedure of designing and analyzing a power system in MATLAB

4.3 Apparatus

- MATLAB

4.4 Theory

The Y-bus matrix is a N × N matrix used in power engineering to describe a power system with N buses. In a practical power system, each bus is often connected to only a few other buses via transmission lines.

In terms of load currents, the voltages and currents of the buses are obtained by solving these equations. The power and load flow on all buses in the system may then be determined. If $\boldsymbol{I_{bus}}$ is the bus currents in matrix form and $\boldsymbol{Y_{bus}}$ is the admittance matrix and $\boldsymbol{V_{bus}}$ is the bus voltages in matrix form then they can be related as $\boldsymbol{I_{bus}}$=$\boldsymbol{Y_{bus}V_{bus}}$

The current and voltage of specific branches under different situations are sometimes more interesting than others. In that instance, the unimportant buses can be decreased to make the system easier to understand. Buses from bus admittance matrix can be reduced using the following equation:

$$\boldsymbol{Y_{i,j}}(new) = \boldsymbol{Y_{i,j}}(old) - \frac{\boldsymbol{Y_{i,n}} * \boldsymbol{Y_{n,j}}}{\boldsymbol{Y_{n,n}}}$$

Here $\boldsymbol{i}$, $\boldsymbol{j}$=1,2,3, ...(n-1) and $\boldsymbol{Y}$ is the bus admittance matrix. By using the reduced bus, the admittance between the buses directly not connected can be found and using that admittance power flow can be determined.

The bus impedances can be recorded in an excel file for better management. The excel file can then be put into MATLAB to create the admittance matrix.

4.5 Required apparatus

- MATLAB

4.6 Block diagram

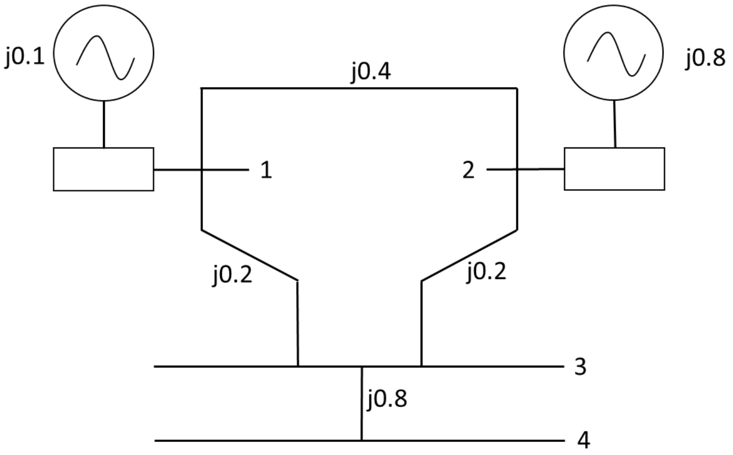


Fig. 4.1: Diagram of Y- bus no.1

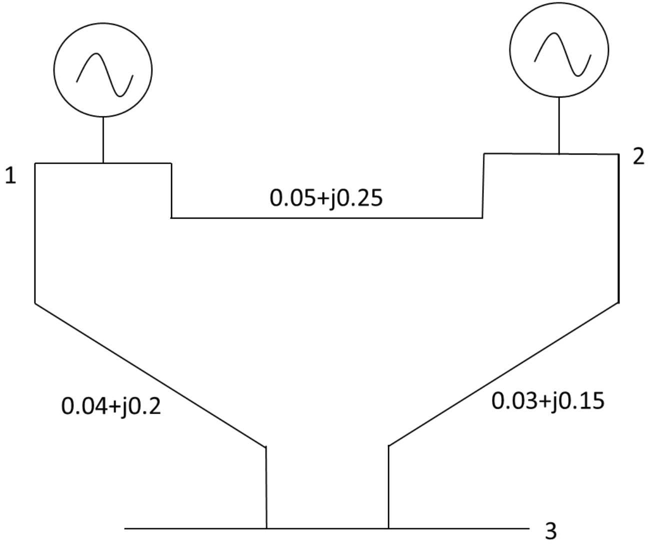


Fig. 4.2: Diagram of Y- bus no.2

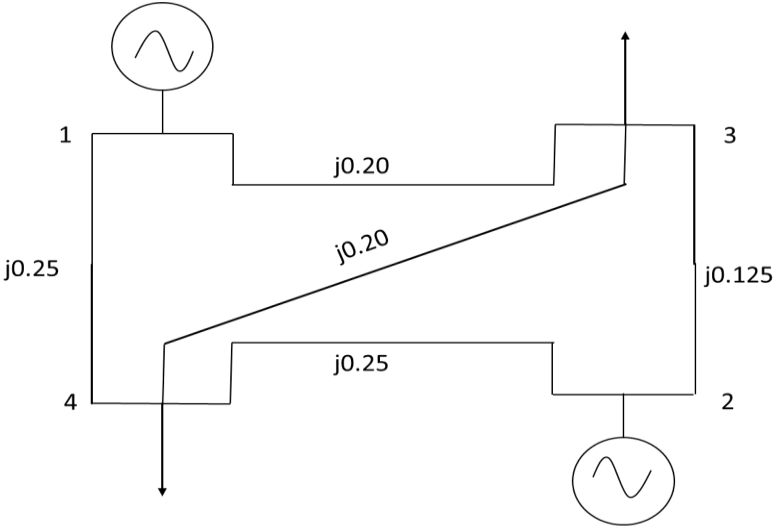


Fig. 4.3: Diagram of Y- bus no.3

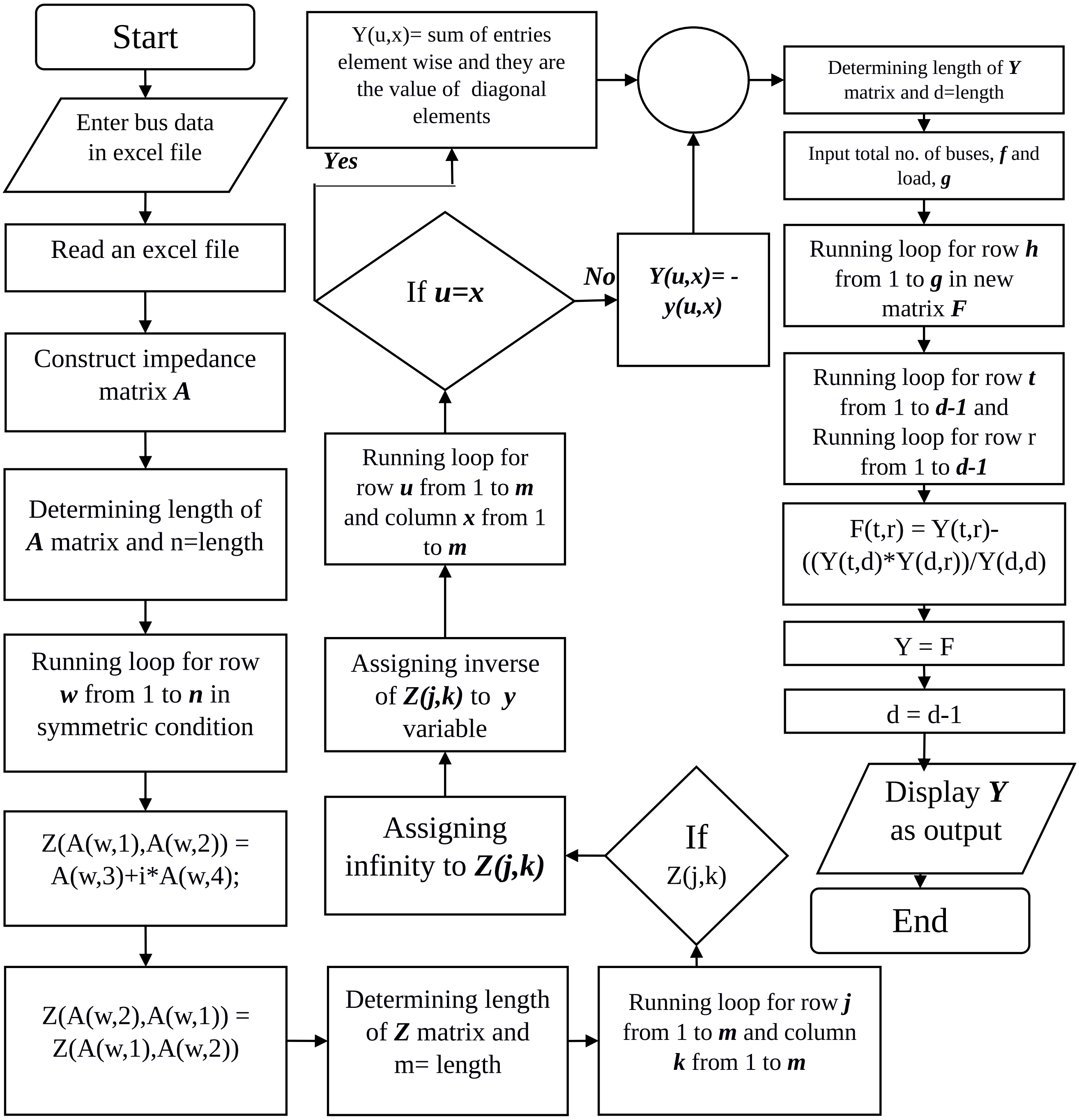
4.7 Algorithm

1. Start
2. Read an excel file that has bus numbers ***i*** and ***j*** in the first two columns and resistance and admittance value in the third and fourth columns which represent impedance between the buses
3. Construct a matrix ***A*** whose element $\boldsymbol{a_{i,j}}$ denotes the impedance between ***i*** and ***j*** buses
4. Determine the length of matrix ***A***
5. Perform symmetric condition and construct a new matrix ***Z*** where 3rd and 4th column element of the excel file is equivalent to ***a+bj*** and it can be assigned to $\boldsymbol{Z_{i,j}}$ later applying symmetric condition $\boldsymbol{Z_{i,j}} = \boldsymbol{Z_{j,i}}$
6. Determine the length of new matrix ***Z***
7. Use looping condition for row j from 1 to length of Z and for column k from1 to length Z
8. If value of ***j*** row and ***k*** column is equal to zero, assign the element value as infinity
9. Determine inverse matrix of ***Z*** and assigned to ***Z***
10. Calculate the summation of ***Y*** matrix element row wise
11. Use looping condition for row ***u*** from 1 to length of ***Z*** and for column ***x*** from 1 to length ***Z***
12. Use formula if $\boldsymbol{u \neq x}$ $Y_{\boldsymbol{ux}} = -\boldsymbol{y_{ux}}$, else $Y_{\boldsymbol{ux}} = \sum_{k=1}^{n} y_{ux}$
13. Display ***Y*** as output
14. Determine the length of ***Y*** bus matrix
15. Provide the total no. of buses and load to be reduced as input
16. Use looping condition for ***h*** from 1 to length of load to be reduced and define new matrix, ***F*** inside the loop
17. Now use the formula inside a loop condition where for row ***t*** from 1 to (total no. Of buses-1) and for column ***r*** from 1 to (total no. of buses-1) to obtain the reduced bus matrix,

$$\boldsymbol{Y_{i,j}}(new) = \boldsymbol{Y_{i,j}}(old) - \frac{\boldsymbol{Y_{i,n}} * \boldsymbol{Y_{n,j}}}{\boldsymbol{Y_{n,n}}}$$

18. End

4.8 Flow chart


Start
Enter bus data in excel file
Read an excel file
Construct impedance matrix *A*
Determining length of *A* matrix and n=length
Running loop for row *w* from 1 to *n* in symmetric condition
Z(A(w,1),A(w,2)) = A(w,3)+i*A(w,4);
Z(A(w,2),A(w,1)) = Z(A(w,1),A(w,2))
Determining length of *Z* matrix and m= length
Running loop for row *j* from 1 to *m* and column *k* from 1 to *m*
If Z(j,k)
Assigning infinity to *Z(j,k)*
Assigning inverse of *Z(j,k)* to *y* variable
Running loop for row *u* from 1 to *m* and column *x* from 1 to *m*
If *u=x*
Yes
Y(u,x)= sum of entries element wise and they are the value of diagonal elements
No
Y(u,x)= - *y(u,x)*
Determining length of *Y* matrix and d=length
Input total no. of buses, *f* and load, *g*
Running loop for row *h* from 1 to *g* in new matrix *F*
Running loop for row *t* from 1 to *d-1* and Running loop for row r from 1 to *d-1*
F(t,r) = Y(t,r)-((Y(t,d)*Y(d,r))/Y(d,d)
Y = F
d = d-1
Display *Y* as output
End

4.9 MATLAB Code & Output

- **Generalized MATLAB Code for Y bus reduction for all the problems**

```
clc;  %Clears previous data from command window
clear all; %Removes all variables from the current workspace
cd('F:\Study material\Lab\3-2\Power System I'); %change the file
directory
A = xlsread('Exp02'); %Read the excel file
n = length(A); %Determine the length of the excel file

% Applying symmetric condition
for w=1:n
     Z(A(w,1),A(w,2)) = A(w,3)+i*A(w,4);
     Z(A(w,2),A(w,1)) = A(w,3)+i*A(w,4);
end

m = length(Z) %Determine the length of the new matrix
for j=1:m
     for k=1:m
         if Z(j,k) == 0
            Z(j,k) = inf;
         end
     end
end
fprintf(' Z matrix is \n') %Display the text
disp(Z) %Display the output
y = 1./Z %Taking inverse impedance matrix
p = sum(y,2) %Taking symmetric summation

%Apply looping condition to determine value of the matrix element
for u=1:m
     for x=1:m
         if u~=x
            Y(u,x)= -y(u,x); %For diagonal element
         else
            Y(u,x)= p(u); %For non-diagonal element
         end
     end
end
fprintf(' Y- bus matrix is \n') %Display the text
disp(Y) %Display the output
d = length(Y);
f = input('Total no. of buses: ');
g = input('No. of reduction: ');
for h=1:g %No. of reduction
    F = zeros(d-1); %Define a new matrix
    for t=1:(d-1) %Access all Y matrix element
        for r=1:(d-1) %
        F(t,r) = Y(t,r)-((Y(t,d)*Y(d,r))/Y(d,d));
        end
    end
    Y = F;
   d = d-1;
```

```
end
fprintf(' Z matrix is \n') %Display the text
disp(Y)
```

- **Output for Y- bus no. 1**

```
Z matrix is
   0.0000 + 0.1000i   0.0000 + 0.4000i   0.0000 + 0.2000i      Inf + 0.0000i
   0.0000 + 0.4000i   0.0000 + 0.8000i   0.0000 + 0.2000i      Inf + 0.0000i
   0.0000 + 0.2000i   0.0000 + 0.2000i      Inf + 0.0000i   0.0000 + 0.0800i
      Inf + 0.0000i      Inf + 0.0000i   0.0000 + 0.0800i      Inf + 0.0000i


y =
   0.0000 -10.0000i   0.0000 - 2.5000i   0.0000 - 5.0000i   0.0000 + 0.0000i
   0.0000 - 2.5000i   0.0000 - 1.2500i   0.0000 - 5.0000i   0.0000 + 0.0000i
   0.0000 - 5.0000i   0.0000 - 5.0000i   0.0000 + 0.0000i   0.0000 -12.5000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 -12.5000i   0.0000 + 0.0000i


p =
   0.0000 -17.5000i
   0.0000 - 8.7500i
   0.0000 -22.5000i
   0.0000 -12.5000i


 Y- bus matrix is
   0.0000 -17.5000i   0.0000 + 2.5000i   0.0000 + 5.0000i   0.0000 + 0.0000i
   0.0000 + 2.5000i   0.0000 - 8.7500i   0.0000 + 5.0000i   0.0000 + 0.0000i
   0.0000 + 5.0000i   0.0000 + 5.0000i   0.0000 -22.5000i   0.0000 +12.5000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +12.5000i   0.0000 -12.5000i


Total no. of buses: 4
No. of reduction: 2
 Reduced Y- bus matrix is
   0.0000 -15.0000i   0.0000 + 5.0000i
   0.0000 + 5.0000i   0.0000 - 6.2500i
```

- **Output for Y- bus no. 2**

```
Z matrix is
   0.0000 + 1.0000i   0.0500 + 0.2500i   0.0400 + 0.0200i
   0.0500 + 0.2500i   0.0000 + 1.0000i   0.0300 + 0.1500i
   0.0400 + 0.0200i   0.0300 + 0.1500i      Inf + 0.0000i


y =
   0.0000 - 1.0000i   0.7692 - 3.8462i  20.0000 -10.0000i
   0.7692 - 3.8462i   0.0000 - 1.0000i   1.2821 - 6.4103i
  20.0000 -10.0000i   1.2821 - 6.4103i   0.0000 + 0.0000i


p =
  20.7692 -14.8462i
   2.0513 -11.2564i
  21.2821 -16.4103i


 Y- bus matrix is
  20.7692 -14.8462i  -0.7692 + 3.8462i -20.0000 +10.0000i
  -0.7692 + 3.8462i   2.0513 -11.2564i  -1.2821 + 6.4103i
 -20.0000 +10.0000i  -1.2821 + 6.4103i  21.2821 -16.4103i


Total no. of buses: 3
No. of reduction: 1
 Reduced Y- bus matrix is
   2.8402 - 9.8757i  -2.8402 + 8.8757i
  -2.8402 + 8.8757i   2.8402 - 9.8757i
```

- **Output for Y- bus no. 3**

```
Z matrix is
      Inf + 0.0000i      Inf + 0.0000i   0.0000 + 0.2000i   0.0000 + 0.2500i
      Inf + 0.0000i      Inf + 0.0000i   0.0000 + 0.1250i   0.0000 + 0.2500i
   0.0000 + 0.2000i   0.0000 + 0.1250i      Inf + 0.0000i   0.0000 + 0.2000i
   0.0000 + 0.2500i   0.0000 + 0.2500i   0.0000 + 0.2000i      Inf + 0.0000i
```

```
y =
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 - 5.0000i   0.0000 - 4.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 - 8.0000i   0.0000 - 4.0000i
   0.0000 - 5.0000i   0.0000 - 8.0000i   0.0000 + 0.0000i   0.0000 - 5.0000i
   0.0000 - 4.0000i   0.0000 - 4.0000i   0.0000 - 5.0000i   0.0000 + 0.0000i

p =
   0.0000 - 9.0000i
   0.0000 -12.0000i
   0.0000 -18.0000i
   0.0000 -13.0000i

 Y- bus matrix is
   0.0000 - 9.0000i   0.0000 + 0.0000i   0.0000 + 5.0000i   0.0000 + 4.0000i
   0.0000 + 0.0000i   0.0000 -12.0000i   0.0000 + 8.0000i   0.0000 + 4.0000i
   0.0000 + 5.0000i   0.0000 + 8.0000i   0.0000 -18.0000i   0.0000 + 5.0000i
   0.0000 + 4.0000i   0.0000 + 4.0000i   0.0000 + 5.0000i   0.0000 -13.0000i

Total no. of buses: 4
No. of reduction: 2
Reduced Y- bus matrix is
   0.0000 - 5.1100i   0.0000 + 5.1100i
   0.0000 + 5.1100i   0.0000 - 5.1100i
```

4.10 Discussion & Conclusion

In this experiment, we designed an algorithm, flow chart, and programmed a generalized code for three different Y- bus system. In each case, we extracted the values from an excel file and formulated necessary condition to assign values to variables. Through this generalized coding format, we easily design and calculate reduced Y bus matrix for any given power system.

The only adjustment to the code we may need is changing the directory of the file to work with and the given data saved inside the file. The bus numbers and the resistance and reactance values must also be given in the order defined for the code to work and give accurate result.

Experiment No. 05

5.1 Experiment Name

Generate an algorithm and write a program on load flow study of a given power system using Gauss-Seidel method

5.2 Objectives

- To become acquainted with the load flow study of a given power system
- To learn how to generate a MATLAB code for numerical analysis using Gauss-Seidel method
- To get familiar with the procedure of designing and analyzing a power system in MATLAB

5.3 Theory

A load flow study is a numerical analysis of the flow of electric power in any electrical system. Its goal is to determine the flow, current, voltage, real power, and reactive power in a system under any load conditions.

A single-phase model is used to solve a load flow problem because the system is assumed to be operating under balanced conditions. Each bus is associated with four quantities. These are voltage magnitudes $|\boldsymbol{V}|$, phase angle $\boldsymbol{\delta}$, real power $\boldsymbol{P}$ and reactive power $\boldsymbol{Q}$.

For a power system with n nodes the network equation can be given by the matrix equation

$$\boldsymbol{I = YV}$$

The current injected into the $\boldsymbol{i^{th}}$ node can be obtained as $\boldsymbol{I_i = \sum_{k=1}^{n} Y_{ik}V_{ik}}$

The power injected into the $\boldsymbol{i^{th}}$ node can be written as

$$\boldsymbol{S_i = P_i + jQ_i = \sum_{K=1}^{n}|V_i|\,|V_k||V_{ik}|\angle\delta_i - \theta_{ik} - \delta_k}$$

Hence, the real power, $\boldsymbol{P_i}$ = **Re**($\boldsymbol{S_i}$) = $\boldsymbol{\sum_{K=1}^{n}|V_i|\,|V_k||V_{ik}|\cos\delta_i - \theta_{ik} - \delta_k}$

The reactive power, $\boldsymbol{Q_i}$ = **Im**($\boldsymbol{S_i}$) = $\boldsymbol{-\sum_{K=1}^{n}|V_i|\,|V_k||V_{ik}|\sin\delta_i - \theta_{ik} - \delta_k}$

The node voltage of a system can be found using the following equation

$$\boldsymbol{V_i = \frac{1}{Y_{ii}}\left[\frac{S_i}{V_i} - \sum_{\substack{K=1\\K\neq i}}^{n} Y_{ik}^{*}\,V_k^{*}\right]} \qquad [\text{for } \boldsymbol{i}= 1,2,3\ldots..\boldsymbol{n}]$$

5.4 Required apparatus

- MATLAB

5.5 Block diagram

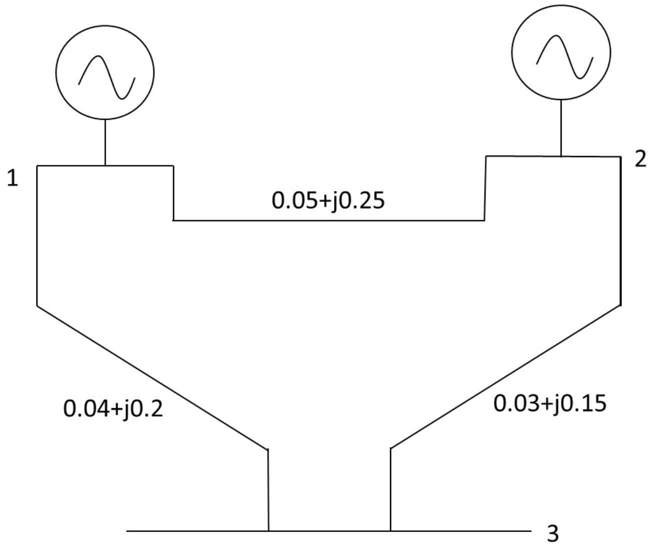


Fig. 5.1: Diagram of a three- bus system

5.6 Data table

Bus	Bus	R	X
1	1	0	1
1	2	0.05	0.25
1	3	0.04	0.02
2	2	0	1
2	3	0.03	0.15

Fig. 5.1 Excel file of the Impedance data

Bus	|v|	Pg	Qg	Pl	Ql	Angle
1	1.03	0	0	0	0	0
2	1.02	0.8	0	0.4	0.3	0
3	1	0	0	1	0.8	0

Fig. 5.2 Excel file of the Load data

5.7 Algorithm

1. Start
2. Read an excel file for taking input data and locate it
3. Generate a Y-bus matrix
4. Read node data. For P-V nodes take the starting values of the voltages as

$$V_k^0 = V_k^0 \angle 0^0 \text{ for k=1,2,3......m.}$$

5. For load nodes take the starting values as $V_k^0 = 1.0\angle 0^0$ for k= m+1, m+2......
6. Start iteration count $i = 0$ and node number $k = 2$
7. If $k > m$, proceed to step 10, else move on to 9
8. As this is a P-V bus bar, compute the reactive power injection as follows

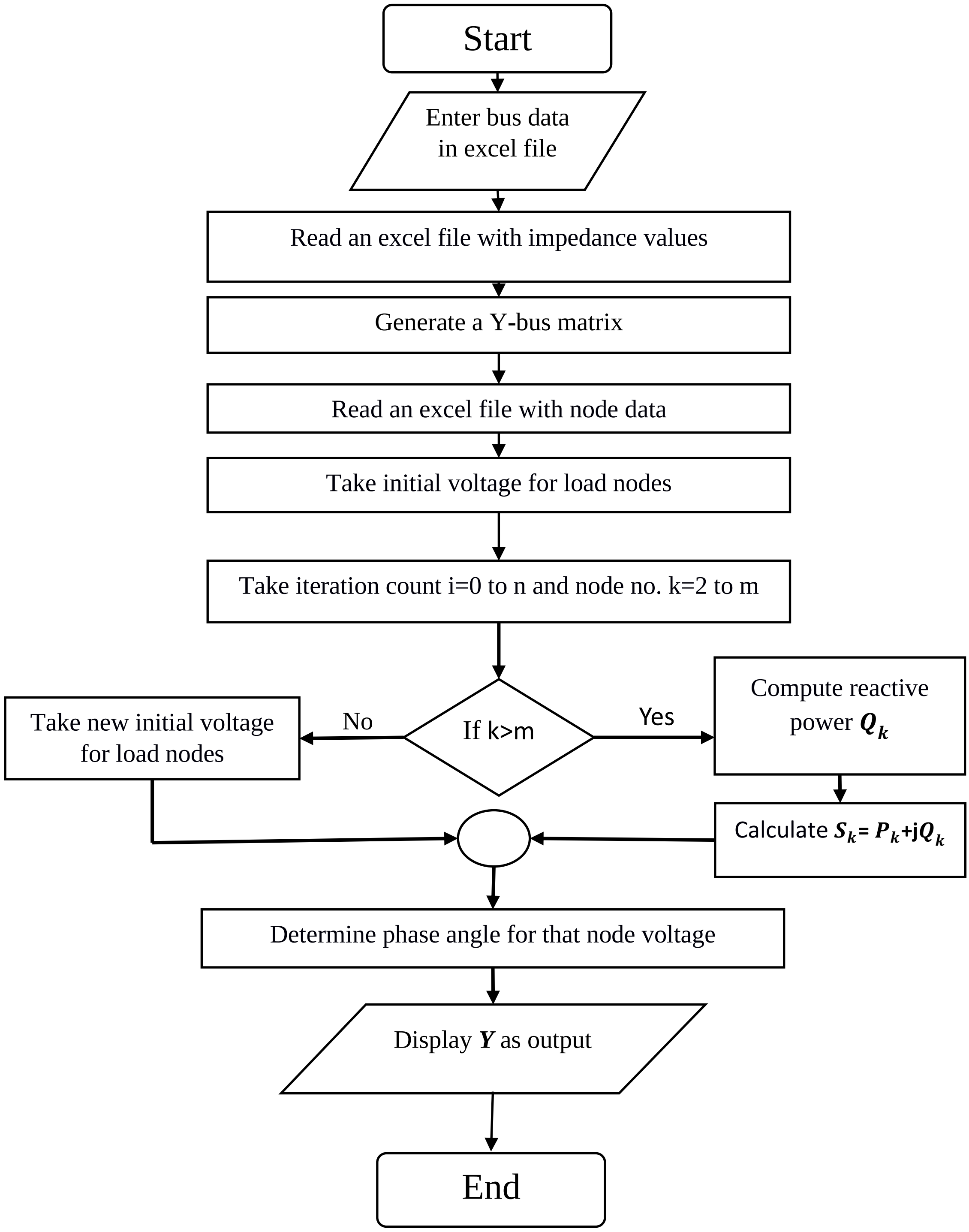
$$Q_k = \mathrm{Im}(S_k) = \mathrm{Im}\ [\sum_{l=1}^{k-1} V_k^i\, Y_{kl}^*(V_k^{i+1})^* + \sum_{l=k}^{n} V_k^i\, Y_{kl}^*(V_k^i)^*]$$

9. Calculate $S_k = P_k + jQ_k$
10. Calculate the new value of voltage as follows

$$V_k^{i+1} = \frac{1}{Y_{kk}}\ [\frac{S_k^*}{(V_k^i)^*} - \sum_{l=1}^{k-1} Y_{kl}\, V_l^{i+1} - \sum_{l=k+1}^{n} Y_{kl}\, V_l^i]$$

11. If $k > m$, proceed to step 14, else move to step 13
12. For P-V node and $|V_k^\delta|$ has already calculated, determine the phase angle from the value of V_k obtained in the above equation $\delta_k = \tan^{-1}[ImV_k^{i+1}/Re\ (V_k^{i+1})$ and compute the voltage at this node as $V_k^{i+1} = |V_k^\delta| \angle \delta_k$
13. If $k < n$ go to step 15 otherwise go to step 16
14. Take $k = K+1$ and go to step 8
15. If $|V_k^{i+1} - V_k^i|$ for all k = 2,3,4…. n is within tolerance limit, proceed to step 18, else go to step 17.
16. Take $i = i+1$ and go to step 7.
17. If $Pg\sim \leftarrow 0, V_k \leftarrow V_{o_k} e^{ia_{ng}}$ and $E <\leftarrow 10^{-3}$ break
18. Display output
19. End

5.8 Flow chart


Start
Enter bus data in excel file
Read an excel file with impedance values
Generate a Y-bus matrix
Read an excel file with node data
Take initial voltage for load nodes
Take iteration count i=0 to n and node no. k=2 to m
If k>m
No
Yes
Take new initial voltage for load nodes
Compute reactive power Q_k
Calculate S_k= P_k+jQ_k
Determine phase angle for that node voltage
Display Y as output
End

5.9 MATLAB Code & Output

```
clc;  %Clears previous data from command window
clear all; %Removes all variables from the current workspace
cd('F:\Study material\Lab\3-2\Power System I'); %change the file directory
A = xlsread('EXp02p02'); %Read the excel file
n = length(A); %Determine the length of the excel file

% Applying symmetric condition
for w=1:n
    Z(A(w,1),A(w,2)) = A(w,3)+i*A(w,4);
    Z(A(w,2),A(w,1)) = A(w,3)+i*A(w,4);
end

m = length(Z); %Determine the length of the new matrix
for j=1:m
    for k=1:m
        if Z(j,k) == 0
            Z(j,k) = inf;
        end
    end
end
fprintf(' Z matrix is \n') %Display the text
disp(Z) %Display the output
y = 1./Z %Taking inverse impedance matrix
p = sum(y,2) %Taking symmetric summation

%Apply looping condition to determine value of the matrix element
for u=1:m
    for x=1:m
        if u~=x
            Y(u,x)= -y(u,x); %For diagonal element
        else
            Y(u,x)= p(x); %For non-diagonal element
        end
    end
end
fprintf(' Y- bus matrix is \n') %Display the text
disp(Y) %Display the output

cd('F:\Study material\Lab\3-2\Power System I'); %change the file directory
B = xlsread('Exp05'); %Read the excel file
j = 3;
V = B(:,2);
V0=B(:,2);

%to get the value of real power
P=B(:,3)-B(:,5);

%to get the value of reactive power
Q=B(:,4)-B(:,6);

%to get the angle
ang=B(:,7);
V1=V;

% to get the value of generator bus
Pg=B(:,3);
for w=1:100
    z=V;
    for k=2:j
```

```
        yv1=0;
        yv2=0;
        for h=1:j
            yv2=yv2+Y(k,h)*V(h); %to find the product of Y bus and voltages
            if h~=k
                yv1=yv1+Y(k,h)*V(h); %to find the product of Y bus and
voltages
            end
        end
        if Pg(k)~=0
            g(k)=imag(V(k)*(conj(yv2))); %to get the imaginary value
            S(k)=P(k)+1i*g(k); %to calculate the apparent power
        else S(k)=P(k)+1i*Q(k);
        end
        V(k)=(1/Y(k,k))*((conj(S(k))/conj(V(k)))-yv1); %to get the value of
node voltages
        ang1(k)=angle(V(k));    %to get the angles
        ang2(k)=rad2deg(ang1(k)); %to convert the radian values to degrees
        if Pg(k)~=0
            V(k)=V0(k)*exp(1i*ang1(k));
        end
    end
        V1=abs(V);
         ang2=rad2deg(ang1);
    E=abs((V-z)/V);
    if E<=10e-4
        break;   %to break the for loop
    end
    Vlt_1(w)=V1(1);
    Vlt_2(w)=V1(2);
    Vlt_3(w)=V1(3);
    ang_1(w)=ang2(1);
    ang_2(w)=ang2(2);
    ang_3(w)=ang2(3);
end

% to show the value column wise
Vlt_1=Vlt_1';
ang_1=ang_1';
Vlt_2=Vlt_2';
ang_2=ang_2';
Vlt_3=Vlt_3';
ang_3=ang_3';
iteration=(1:w-1)';

% to show the values in a table
table(iteration,Vlt_1,ang_1,Vlt_2,ang_2,Vlt_3,ang_3)
```

Output

```
Z matrix is

  0.0000 + 1.0000i   0.0500 + 0.2500i   0.0400 + 0.0200i

  0.0500 + 0.2500i   0.0000 + 1.0000i   0.0300 + 0.1500i

  0.0400 + 0.0200i   0.0300 + 0.1500i      Inf + 0.0000i

y =

  0.0000 - 1.0000i   0.7692 - 3.8462i  20.0000 -10.0000i
```

```
   0.7692 - 3.8462i   0.0000 - 1.0000i   1.2821 - 6.4103i
  20.0000 -10.0000i   1.2821 - 6.4103i   0.0000 + 0.0000i


p =
  20.7692 -14.8462i
   2.0513 -11.2564i
  21.2821 -16.4103i


 Y- bus matrix is
  20.7692 -14.8462i  -0.7692 + 3.8462i -20.0000 +10.0000i
  -0.7692 + 3.8462i   2.0513 -11.2564i  -1.2821 + 6.4103i
 -20.0000 +10.0000i  -1.2821 + 6.4103i  21.2821 -16.4103i


ans =
  3×7 table
    iteration    Vlt_1    ang_1    Vlt_2    ang_2      Vlt_3      ang_3
    _________    _____    _____    _____    ______    _______    _______
        1        1.03       0      1.02     1.7965    0.98557    0.48806
        2        1.03       0      1.02     2.2139    0.98602    0.54621
        3        1.03       0      1.02     2.3017    0.98628    0.56041
```

5.10 Discussion & Conclusion

In this experiment, we designed an algorithm, flow chart, and programmed a generalized code for load flow study of a given power system. The sinusoidal steady state of the entire system is provided by the load flow.

It is critical for evaluating the best operating of the existing system and planning future system expansion. It aids in the design of a new power system network, the reduction of system loss, and the selection of transformer taps for efficient operation.

The Gauss-Seidel method is utilized for doing load flow analysis because of these consequences. Though this method takes longer to converge than others, its advantage is its simplicity and ease of performance.

The only adjustment to the code we may need is changing the directory of the file to work with and the given data saved inside the file. The bus numbers and the resistance and reactance values must also be given in the order defined for the code to work and give accurate result.

Experiment No. 06

6.1 Experiment Name

Generate an algorithm and write a program on load flow study of a given power system using Newton Raphson method

6.2 Objectives

- To become acquainted with the load flow study of a given power system
- To learn how to generate a MATLAB code for numerical analysis using Newton Raphson method
- To get familiar with the procedure of designing and analyzing a power system in MATLAB

6.3 Theory

Power system is a very large interconnected electrical network. So different techniques have been developed to analyze power systems. Node voltage method is one of those techniques.

The equations in the nodal admittance form result in a simultaneous complex algebraic equation in terms of load currents. Solving these equations gives the voltages and currents of the buses.

The biggest challenge in solving these equations is finding $\boldsymbol{n}$ unknown quantities from $\boldsymbol{n}$ nonlinear equations. Iterative methods are suitable for solving nonlinear systems of equations.

For this reason, iterative methods are suitable for solving node voltage equations. Net injected bus current has the relation. For a power system with $\boldsymbol{n}$ nodes the network equation where the current injected into the $\boldsymbol{i^{th}}$ node can be obtained as $\boldsymbol{I_i = \sum_{k=1}^{n} Y_{ik}V_{ik}}$

The power injected into the $\boldsymbol{i^{th}}$ node can be written as

$$\boldsymbol{S_i = P_i + jQ_i = \sum_{K=1}^{n} |V_i|\ |V_k||V_{ik}|\angle\delta_i - \theta_{ik} - \delta_k}$$

6.4 Required apparatus

- MATLAB

6.5 Block diagram

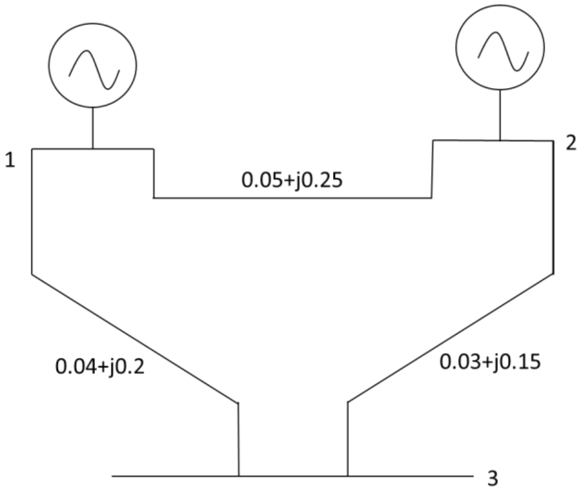


Fig. 6.1: Diagram of a three- bus system

6.6 Data table

Bus	|v|	Pg	Qg	Pl	Ql	Angle
1	1.03	0	0	0	0	0
2	1.02	0.8	0	0.4	0.3	0
3	1	0	0	1	0.8	0

Fig. 6.1 Excel file of the Load data

Bus	Bus	R	X
1	1	0	1
1	2	0.05	0.25
1	3	0.04	0.02
2	2	0	1
2	3	0.03	0.15

Fig. 6.2 Excel file of the Impedance data

6.7 Algorithm

1 Start

2 Read an excel file that has bus numbers in the first two columns and resistance and admittance value in the third and fourth columns which represent impedance between the buses

3 Construct a matrix $\boldsymbol{M_{imp}}$ whose element $\boldsymbol{a_{i,j}}$ denotes the impedance between i and j buses

4 Perform element wise inversion of $\boldsymbol{M_{imp}}$ matrix and keep it in y matrix

5 Use the formulas to construct the $\boldsymbol{Y_{bus}}$ matrix $\boldsymbol{Y_{i,j}|_{i=j} = \sum_0^j \frac{1}{y_{(i,j)}}}$ and $\boldsymbol{y_{i,j}|_{i\neq j} = \frac{-1}{y_{(i,j)}}}$

6 Read the bus parameters from an excel file where P and V are defined for generator buses and P and Q are defined for load buses and P and δ are defined

7 Assume V = 1 and δ = 0° for nodes if not defined

8 Calculate the complex power as follows:

9 $\boldsymbol{S_i^j = (P_{g_i} - P_{l_i}) + i \times Im(V_i^{j-1} \times (\sum_{k=1}^n Y_{i,k} \times V_k^{mostrecent})^*)}$

10 Calculate the jacobian $\boldsymbol{J} = \begin{bmatrix} \boldsymbol{H} & \boldsymbol{L} \\ \boldsymbol{M} & \boldsymbol{N} \end{bmatrix}$ as follows:

$\boldsymbol{H = \frac{\partial P}{\partial \delta}}$; $\boldsymbol{L = \frac{\partial P}{\partial V}}$; $\boldsymbol{M = \frac{\partial Q}{\partial \delta}}$; $\boldsymbol{H = \frac{\partial Q}{\partial V}}$

11 where bus real power, $\boldsymbol{P_i = \sum_{j=1}^n |V_i||Y_{i,j}||V_j|\cos(\theta_i - \delta_i + \delta_j)}$ and bus reactive power, $\boldsymbol{Q_i = -\sum_{j=1}^n |V_i||Y_{i,j}||V_j|\sin(\theta_i - \delta_i + \delta_j)}$

12 Calculate the power mismatches as follows: $\boldsymbol{\Delta P = P_g - P_l - \mathrm{Re}(S)}$ and $\boldsymbol{\Delta Q = -Q_l - \mathrm{Im}(S)}$

13 Calculate the phase and amplitude differences of buses using $\boldsymbol{\Delta(\delta, V) = J^{-1} \times \Delta(P, Q)}$

14 Correct the bus voltages using $\boldsymbol{V^{i+1} = V^i + \Delta V}$ and $\boldsymbol{\delta^{i+1} = \delta^i + \Delta\delta}$

15 Calculate error as: - $\boldsymbol{\varepsilon = |V_i^{j-1} - V_i^j|}$

16 If ε < tolerance value goes to next step otherwise go to step 8

17 End

6.8 MATLAB Code & Output

```
clc;  %Clears previous data from command window
clear all; %Removes all variables from the current workspace
cd('F:\Study material\Lab\3-2\Power System I'); %change the file directory
A = xlsread('EXp02p02'); %Read the excel file
```

```
n = length(A); %Determine the length of the excel file
% Applying symmetric condition
for w=1:n
    Z(A(w,1),A(w,2)) = A(w,3)+i*A(w,4);
    Z(A(w,2),A(w,1)) = A(w,3)+i*A(w,4);
end
m = length(Z); %Determine the length of the new matrix
for j=1:m
    for k=1:m
        if Z(j,k) == 0
            Z(j,k) = inf;
        end
    end
end
fprintf(' Z matrix is \n') %Display the text
disp(Z) %Display the output
y = 1./Z %Taking inverse impedance matrix
p = sum(y,2) %Taking symmetric summation
%Apply looping condition to determine value of the matrix element
for u=1:m
    for x=1:m
        if u~=x
            Y(u,x)= -y(u,x); %For diagonal element
        else
            Y(u,x)= p(x); %For non-diagonal element
        end
    end
end
fprintf(' Y- bus matrix is \n') %Display the text
disp(Y) %Display the output
cd('F:\Study material\Lab\3-2\Power System I'); %change the file directory
LFS = xlsread('Exp05'); %Read the excel file
busnum=LFS(:,1); bustype=LFS(:,1); nbus=length(busnum); voltage=LFS(:,2);
Angle=LFS(:,7); Pg=LFS(:,3); Pl=LFS(:,5); Qg=LFS(:,4); Ql=LFS(:,6);
P=LFS(:,3)-LFS(:,5); Q=LFS(:,3)-LFS(:,5); Psp=P; Qsp=P;
for j=1:nbus
    v(j)=voltage(j)*exp(1i*Angle(j));
end
for i=1:nbus
    bustype(1)=busnum(1);
    if Pg(i)~=0&&Qg(i)==0&&v(i)~=1
        bustype(i)=2;
    else if P(i)~=0&&Q(i)~=0&&v(i)==1
            bustype(i)=3;
        end
    end
end
w=real(v); u=imag(v); [del V]=cart2pol(w,u); vnew=V; w=real(Y); u=imag(Y);
[theta Y]=cart2pol(w,u); pv=find(bustype==2); pq=find(bustype==3);
npv=length(pv); npq=length(pq); delv=1; count=0; tol=0.1
v1=0;
v2=0;
v3=0;
V=vnew;
P=zeros(nbus,1);
Q=zeros(nbus,1);
for i=1:nbus
        for k=1:nbus
            P(i)=P(i)+(V(i)*Y(i,k)*V(k)*cos(theta(i,k)-del(i)+del(k)));
            Q(i)=Q(i)-(V(i)*Y(i,k)*V(k)*sin(theta(i,k)-del(i)+del(k)));
        end
    end
    dPp=Psp-P;
    dQp=Qsp-Q;
    k=1;
    dQ=zeros(npq,1);
    for i=1:npq
        if bustype(i)==2
            dQ(k)=dQp(i);
        end
    end
    dP=dPp(2:nbus);
    M=[dP;dQ];
    J1=zeros(nbus-1,nbus-1);
    for i=1:nbus-1
        m=i+1;
        for k=1:nbus-1
            n=k+1;
            if m==n
                for n=1:nbus
                    J1(i,k)=J1(i,k)+(V(m)*Y(m,n)*V(n)*sin(theta(m,n)-
del(m)+del(n)));
                end
                J1(i,k)=J1(i,k)-(V(m)*Y(m,n)*V(m)*sin(theta(m,m)));  %hh
            else
            J1(i,k)=-(V(m)*Y(m,n)*V(n)*sin(theta(m,n)-del(m)+del(n)));
            end
        end
    end
    J2=zeros(nbus-1,npq);
    for i=1:nbus-1
        m=i+1;
         for k=1:npq
            n=pq(k);
            if m==n
                for n=1:nbus
                    J2(i,k)=J2(i,k)+(V(m)*Y(m,n)*V(n)*cos(theta(m,n)-
del(m)+del(n)));
                end
                J2(i,k)=J2(i,k)+(2*V(m)*Y(m,n)*cos(theta(m,m)));
            else
            J2(i,k)=(V(m)*Y(m,n)*sin(theta(m,n)-del(m)+del(n)));
            end
        end
    end
         J3=zeros(npq,nbus-1);
     for i=1:npq
        m=pq(i);
         for k=1:nbus-1
            n=k+1;
            if m==n
                for n=1:nbus
                    J3(i,k)=J3(i,k)+(V(m)*Y(m,n)*V(n)*cos(theta(m,n)-
del(m)+del(n)));
                end
                J3(i,k)=J3(i,k)-(V(m)*Y(m,m)*V(m)*cos(theta(m,m)));  %h
            else
            J3(i,k)=-(V(m)*Y(m,n)*V(n)*cos(theta(m,n)-del(m)+del(n)));
            end
        end
     end
          J4=zeros(npq,npq);
     for i=1:npq
        m=pq(i);
         for k=1:npq
            n=pq(k);
            if m==n
                for n=1:nbus
                    J4(i,k)=J4(i,k)-(V(n)*Y(m,n)*sin(theta(m,n)-
del(m)+del(n)));
                end
                J4(i,k)=J4(i,k)-(2*V(m)*Y(m,m)*sin(theta(m,m)));  %h
            else
            J4(i,k)=-(V(m)*Y(m,n)*sin(theta(m,n)-del(m)+del(n)));
            end
        end
     end
J=[J1 J2;J3 J4]
X=inv(J)*M;
dth=X(1:nbus-npq);
dV=X(npv+1:nbus);
for i=2:nbus
    del(i)=del(i)+dth(i-1);
end
for i=1:nbus
    k=1;
    if(bustype(i)==3)
        vnew(i)=V(i)+dV(k);
    end
    k=k+1;
end
for i=1:nbus
   I=[vnew(i) del(i)];
end
%disp
count=count+1; delv=abs(vnew-V); I=[count vnew del delv];
disp('Iteration Magnitude Angle Error'); disp(I); Bus=busnum;
Magnitude=vnew'; ang=radtodeg(del); Angle=ang';
T=table(Bus,Magnitude,Angle);
disp(T);
```

Output

Z matrix is

```
   0.0000 + 1.0000i   0.0500 + 0.2500i   0.0400 + 0.0200i
   0.0500 + 0.2500i   0.0000 + 1.0000i   0.0300 + 0.1500i
   0.0400 + 0.0200i   0.0300 + 0.1500i      Inf + 0.0000i
y =

   0.0000 - 1.0000i   0.7692 - 3.8462i  20.0000 -10.0000i
   0.7692 - 3.8462i   0.0000 - 1.0000i   1.2821 - 6.4103i
  20.0000 -10.0000i   1.2821 - 6.4103i   0.0000 + 0.0000i
p =

  20.7692 -14.8462i
  2.0513 -11.2564i
 21.2821 -16.4103i
 Y- bus matrix is

 20.7692 -14.8462i  -0.7692 + 3.8462i -20.0000 +10.0000i
 -0.7692 + 3.8462i   2.0513 -11.2564i  -1.2821 + 6.4103i
-20.0000 +10.0000i  -1.2821 + 6.4103i  21.2821 -16.4103i
J =

   5.5592   -6.5385    6.5385
  -6.5385   16.8385   41.9385
   1.3077  -21.9077   32.3923
Iteration Magnitude Angle Error

   1.0000    1.0300    1.0200    1.0052         0    0.0742    0.0052         0         0    0.0052

   Bus    Magnitude     Angle
   ___    _________    _______
    1        1.03           0
    2        1.02       4.2497
    3      1.0052      0.29937
```

6.9 Discussion & Conclusion

In this experiment, we designed an algorithm, flow chart, and programmed a generalized code for load flow study of a given power system. The sinusoidal steady state of the entire system is provided by the load flow. The code was generated to read the bus parameters from another excel file, solve the bus voltages, and then calculate the admittance matrix using impedances in an excel file.

The only adjustment to the code we may need is changing the directory of the file to work with and the given data saved inside the file. The bus numbers and the resistance and reactance values must also be given in the order defined for the code to work and give accurate result.

Experiment No. 07

7.1 Experiment Name

Write a program to draw the zero, positive, and negative sequence components of a given unbalanced system

7.2 Objectives

- To become acquainted with the balance and unbalance system
- To understand the algorithm and generate a MATLAB code for zero, positive, and negative sequence components of a given unbalanced system
- To get familiar with the procedure of designing and analyzing a power system in MATLAB

7.3 Theory

An unbalanced system of n related phasors can be resolved into n systems of balanced phasors called the symmetrical components of the original phasors. In a three- phase system which is normally balanced, unbalanced fault conditions generally cause unbalanced currents and voltages to exist in each of the phases.

Three unbalanced phasors of a three-phase system can be resolved into three balanced systems of phasors. The balanced sets of components are:

- **Positive-sequence** components consisting of three phasors equal in magnitude, displaced from each other by 120^o in phase, and having the same phase sequence as the original phasors
- **Negative-sequence** components consisting of three phasors equal in magnitude, displaced from each other by 120^o in phase, and having the phase sequence opposite to that of the original phasors, and
- **Zero-sequence** components consisting of three phasors equal in magnitude and with zero phase displacement from each other.

Since each of the original unbalanced phasors is the sum of its components, the original phasors expressed in terms of their components are,

$$\boldsymbol{V_a = V_a^{(0)} + V_a^{(1)} + V_a^{(2)}}$$

$$\boldsymbol{V_b = V_b^{(0)} + V_b^{(1)} + V_b^{(2)} = V_a^{(0)} + a^2V_a^{(1)} + aV_a^{(2)}}$$

$$\boldsymbol{V_c = V_c^{(0)} + V_c^{(1)} + V_c^{(2)} = V_a^{(0)} + aV_a^{(1)} + a^2V_a^{(2)}}$$

Where,

$$\boldsymbol{V_a^{(0)} = \frac{1}{3}(V_a + V_b + V_c)} \qquad \boldsymbol{V_a^{(1)} = \frac{1}{3}(V_a + aV_b + a^2V_c)} \qquad \boldsymbol{V_a^{(2)} = \frac{1}{3}(V_a + a^2V_b + aV_c)}$$

For this experiment, we will consider a system having following values draw their zero, positive, and negative sequence respectively.

$$\boldsymbol{V_a = 10\angle 90^o} \qquad \boldsymbol{V_b = 5\angle 30^o} \qquad \boldsymbol{V_c = 15\angle 210^o}$$

7.4 Required apparatus

- MATLAB

7.5 Algorithm

1. Start
2. Provide magnitude and phase of three unbalanced voltage
3. Provide the value of operator $\boldsymbol{a}$
4. Calculate the values of zero sequence components

$$\boldsymbol{V_a^{(0)}} = \frac{\mathbf{1}}{\mathbf{3}}(\boldsymbol{V_a} + \boldsymbol{V_b} + \boldsymbol{V_c}),\ \boldsymbol{V_a^{(0)}} = \boldsymbol{V_b^{(0)}} = \boldsymbol{V_c^{(0)}}$$

5. Calculate the values of positive sequence components

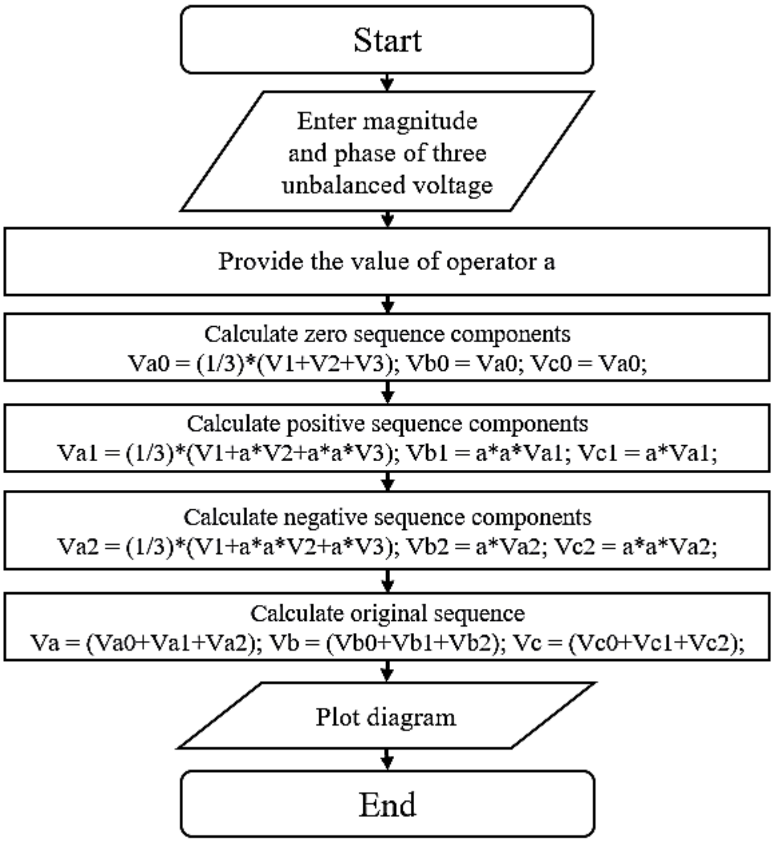
$$\boldsymbol{V_a^{(1)}} = \frac{\mathbf{1}}{\mathbf{3}}\left(\boldsymbol{V_a} + \boldsymbol{aV_b} + \boldsymbol{a^2V_c}\right), \qquad \boldsymbol{V_b^{(1)}} = \boldsymbol{a^2V_a^{(1)}}, \qquad \boldsymbol{V_c^{(1)}} = \boldsymbol{aV_a^{(1)}}$$

6. Calculate the values of negative sequence components

$$\boldsymbol{V_a^{(2)}} = \frac{\mathbf{1}}{\mathbf{3}}\left(\boldsymbol{V_a} + \boldsymbol{a^2V_b} + \boldsymbol{aV_c}\right), \qquad \boldsymbol{V_b^{(2)}} = \boldsymbol{aV_a^{(2)}}, \qquad \boldsymbol{V_c^{(2)}} = \boldsymbol{a^2V_a^{(2)}}$$

7. Calculate original sequence
8. Plot and display the diagram
9. End

7.6 Flow chart


Start
Enter magnitude and phase of three unbalanced voltage
Provide the value of operator a
Calculate zero sequence components
Va0 = (1/3)*(V1+V2+V3); Vb0 = Va0; Vc0 = Va0;
Calculate positive sequence components
Va1 = (1/3)*(V1+a*V2+a*a*V3); Vb1 = a*a*Va1; Vc1 = a*Va1;
Calculate negative sequence components
Va2 = (1/3)*(V1+a*a*V2+a*V3); Vb2 = a*Va2; Vc2 = a*a*Va2;
Calculate original sequence
Va = (Va0+Va1+Va2); Vb = (Vb0+Vb1+Vb2); Vc = (Vc0+Vc1+Vc2);
Plot diagram
End

7.7 MATLAB Code & Output

```
clc; %Clears previous data from command window
clear all; %Removes all variables from the current workspace

% magnitude and phase of unbalanced voltage V1
M1 = input('Enter magnitude of V1:');
P1 = input('Enter phase of V1:');
V1 = M1.*exp(j*deg2rad(P1))
% magnitude and phase of unbalanced voltage V2
M2 = input('Enter magnitude of V2:');
P2 = input('Enter phase of V2:');
V2 = M2.*exp(j*deg2rad(P2))
% magnitude and phase of unbalanced voltage V3
M3 = input('Enter magnitude of V3:');
P3 = input('Enter phase of V3:');
V3 = M3.*exp(j*deg2rad(P3))

% calculate a
a = 1.*exp(j*2*pi/3);
anew = a.*a;

% finding zero sequence
Va0 = (1/3)*(V1+V2+V3);
Vb0 = Va0;
Vc0 = Va0;
% finding positive sequence
Va1 = (1/3)*(V1+a*V2+anew*V3);
Vb1 = anew*Va1;
Vc1 = a*Va1;
% finding negative sequence
Va2 = (1/3)*(V1+anew*V2+a*V3);
Vb2 = a*Va2;
Vc2 = anew*Va2;
% finding original sequence
Va = (Va0+Va1+Va2);
Vb = (Vb0+Vb1+Vb2);
Vc = (Vc0+Vc1+Vc2);

% plotting unbalanced sequence
subplot(2,3,1);
compass([V1,V2,V3]);
title('Unbalanced Sequence');
% plotting zero sequence
subplot(2,3,2);
compass([Va0,Vb0,Vc0]);
title('Zero Sequence');
% finding positive sequence
subplot(2,3,3);
compass([Va1,Vb1,Vc1]);
title('Positive Sequence');
% finding negative sequence
subplot(2,3,4);
compass([Va2,Vb2,Vc2]);
title('Negative Sequence');
```

```
% finding original sequence
subplot(2,3,6);
compass([Va,Vb,Vc]);
title('Original Sequence');
```

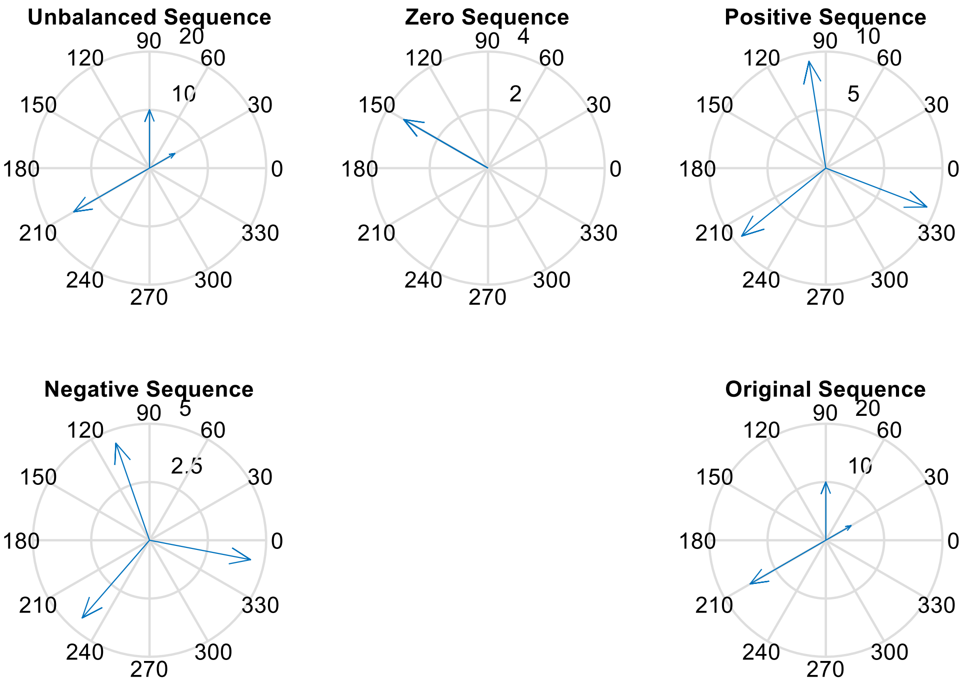
Output

```
Enter magnitude of V1:10
Enter phase of V1:90

Enter magnitude of V2:5
Enter phase of V2:30

Enter magnitude of V3:15
Enter phase of V3:210
```

Diagram


Unbalanced Sequence

Zero Sequence

Positive Sequence

Negative Sequence

Original Sequence

7.8 Discussion & Conclusion

In this experiment, we designed an algorithm, flow chart, and programmed a generalized code for zero, positive, and negative sequence components of a given unbalanced system. Here, we provided the values from and formulated necessary condition to assign values to variables. The only adjustment to the code we may need is changing the input values of the file to work with. Through this generalized coding format, we easily designed our desired output.

Experiment No. 08

8.1 Experiment Name

Transient response analysis of a series RL circuit to imitate the terminal fault of an unloaded alternator

8.2 Objectives

- To become acquainted with the transient response analysis of a series RL circuit
- To understand the algorithm and generate a MATLAB code for transient response analysis of a series RL circuit
- To get familiar with the procedure of designing and analyzing a power system in MATLAB

8.3 Theory

A first-order RL circuit is constructed of one resistor and one inductor that are either connected in series by a voltage source or connected in parallel by a current source. It is one of the most basic analogue infinite impulse response electrical filters available.

Thus, the differential equation is, $\boldsymbol{V_{max}\sin(\omega t+\ \alpha) =\ iR + L\frac{di}{dt}}$

By solving this equation, we obtained, $\boldsymbol{i =\ \frac{V_{max}}{|z|}}[\boldsymbol{\sin(\omega t+\ \alpha-\theta)-\ e^{-\frac{Rt}{L}}\sin(\alpha-\theta)}]$

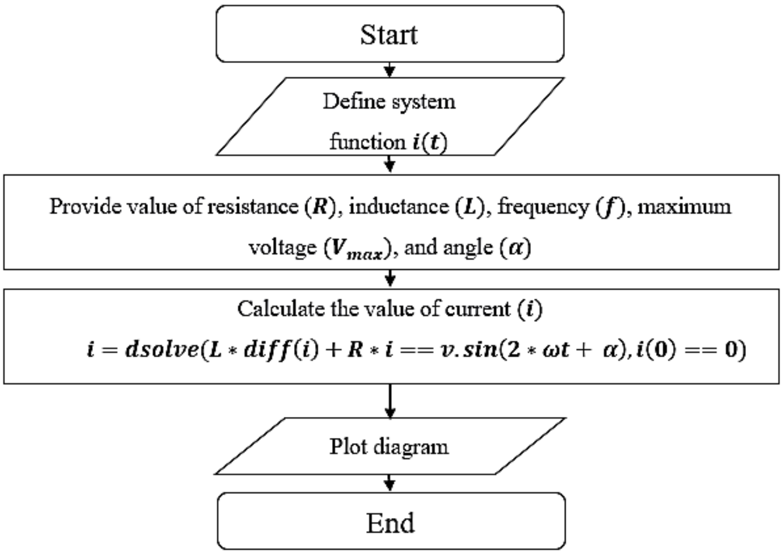
8.4 Required apparatus

- MATLAB

8.5 Algorithm

1. Start
2. Define system function $\boldsymbol{i(t)}$
3. Provide value of resistance, inductance, frequency, maximum voltage, and angle
4. Calculate the value of current ($\boldsymbol{i}$)
   $$\boldsymbol{i = dsolve(L * diff(i) + R * i == v.sin(2 * \omega t +\ \alpha), i(0) == 0)}$$
5. Plot and display the diagram
6. End

8.6 Flow chart


Start
Define system function $\boldsymbol{i(t)}$
Provide value of resistance ($\boldsymbol{R}$), inductance ($\boldsymbol{L}$), frequency ($\boldsymbol{f}$), maximum voltage ($\boldsymbol{V_{max}}$), and angle ($\boldsymbol{\alpha}$)
Calculate the value of current ($\boldsymbol{i}$)
$\boldsymbol{i = dsolve(L * diff(i) + R * i == v.sin(2 * \omega t +\ \alpha), i(0) == 0)}$
Plot diagram
End

8.7 MATLAB Code & Output

```
clc; %Clears previous data from command window
clear all; %Removes all variables from the current workspace

% Declaring function
syms i(t)
% Assigning values to the variable
R = 50;      % Resistance
a = pi/3;    % Phase angle
L = 300e-3;  % Inductance
f = 100;     % Frequency
w = 2*pi*f;
Vm = 100;    % Voltage

% Formula
p = dsolve(L*diff(i)+i/R==Vm*sin(w*t+a),i(0)==0)

% Plotting function
ezplot(p)
grid on
% Labeling plot
xlabel('Time(sec)')
ylabel('Current(amp)')
title('Transient response')
```

Diagram

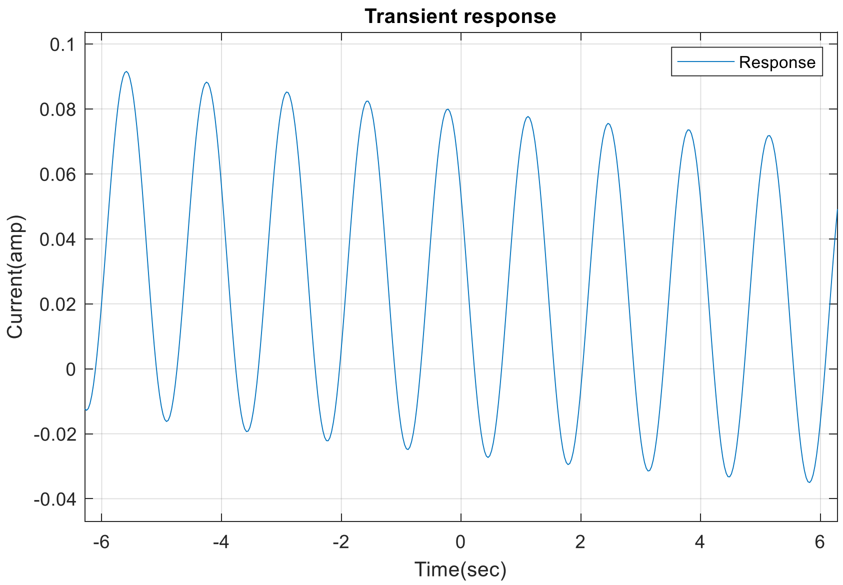


Fig. 8.1: Transient response analysis of a series RL circuit

8.8 Discussion & Conclusion

In this experiment, we designed an algorithm, flow chart, and programmed a generalized code for given transient response analysis of a series RL circuit. Here, we provided the values from and formulated necessary condition to assign values to variables. The only adjustment to the code we may need is changing the input values of the file to work with. Through this generalized coding format, we easily designed our desired output.

Experiment No. 09

9.1 Experiment Name

Load flow study using MATLAB Simulink platform

9.2 Objectives

- To become acquainted with the load flow study
- To understand the algorithm and generate a MATLAB code for load flow study
- To get familiar with the procedure of designing and analyzing a power system in MATLAB

9.3 Theory

Hundreds of buses and branches with impedances specified per unit on a common MVA base make up the power system network. Power flow studies, also known as load flow, are critical for power system analysis and design. Load flow assessments are required for planning, economic operation, scheduling, and power exchange between utilities.

Load Flow analysis in the Simulink platform is simple thanks to the basic functions of the SimPowerSystems^{TM} toolbox.

9.4 Required apparatus

- MATLAB
- Simulink

9.5 Block diagram

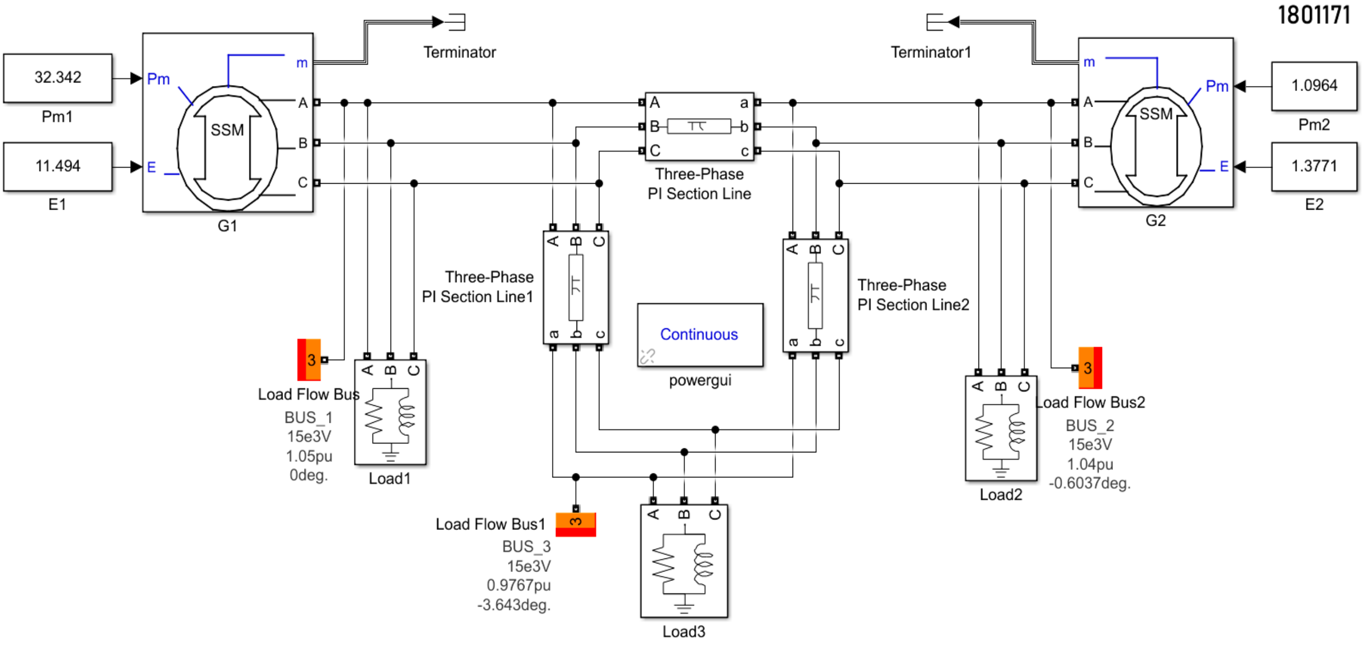
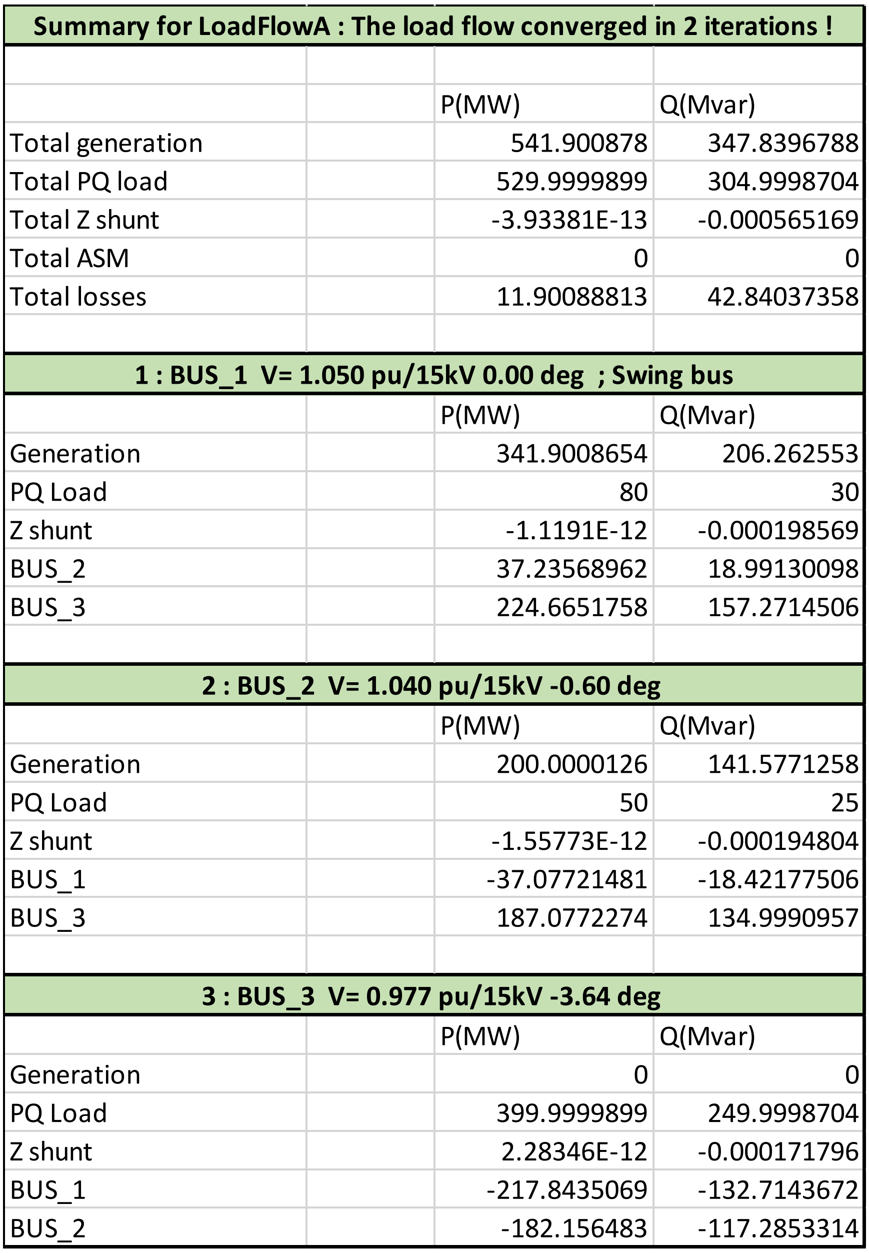


Fig. 9.1: Circuit diagram for load flow study using MATLAB Simulink platform

9.6 Load flow analysis report



Summary for LoadFlowA : The load flow converged in 2 iterations !			
		P(MW)	Q(Mvar)
Total generation		541.900878	347.8396788
Total PQ load		529.9999899	304.9998704
Total Z shunt		-3.93381E-13	-0.000565169
Total ASM		0	0
Total losses		11.90088813	42.84037358
1 : BUS_1 V= 1.050 pu/15kV 0.00 deg ; Swing bus			
		P(MW)	Q(Mvar)
Generation		341.9008654	206.262553
PQ Load		80	30
Z shunt		-1.1191E-12	-0.000198569
BUS_2		37.23568962	18.99130098
BUS_3		224.6651758	157.2714506
2 : BUS_2 V= 1.040 pu/15kV -0.60 deg			
		P(MW)	Q(Mvar)
Generation		200.0000126	141.5771258
PQ Load		50	25
Z shunt		-1.55773E-12	-0.000194804
BUS_1		-37.07721481	-18.42177506
BUS_3		187.0772274	134.9990957
3 : BUS_3 V= 0.977 pu/15kV -3.64 deg			
		P(MW)	Q(Mvar)
Generation		0	0
PQ Load		399.9999899	249.9998704
Z shunt		2.28346E-12	-0.000171796
BUS_1		-217.8435069	-132.7143672
BUS_2		-182.156483	-117.2853314

9.7 Discussion & Conclusion

The exercise taught us how to create a three-bus system in the Simulink platform using the SimPowerSystems^{TM} toolbox. We tweaked the parameters of the generators, loads, and transmission lines after developing it.

In the load flow analysis phase, we used PowerShell to evaluate the system. We studied the system and replaced the system's values with new values. The system's objectives were met with success.

Experiment No. 10

10.1 Experiment Name

Study of the effect of over-current relay and observation of the response for 3LG and 1LG fault in transmission line using MATLAB Simulink

10.2 Objectives

- To become acquainted with the relay and circuit breaker operation of system
- To understand the operation and simulate observation of the response for 3LG and 1LG fault in transmission line
- To get familiar with the procedure of designing and analyzing a power system in MATLAB

10.3 Theory

The fault analysis of a power system is required in order to provide information for the selection of switchgear, setting of relays and stability of system operation. Faults usually occur in a power system due to either insulation failure, flashover, physical damage or human error. These faults, may either be three phases in nature involving all three phases in a symmetrical manner, or may be asymmetrical where usually only one or two phases may be involved. The common types of asymmetrical faults occurring in a Power System are single line to ground faults and line to line faults, with and without fault impedance.

10.4 Required apparatus

- Simulink

10.5 Block diagram

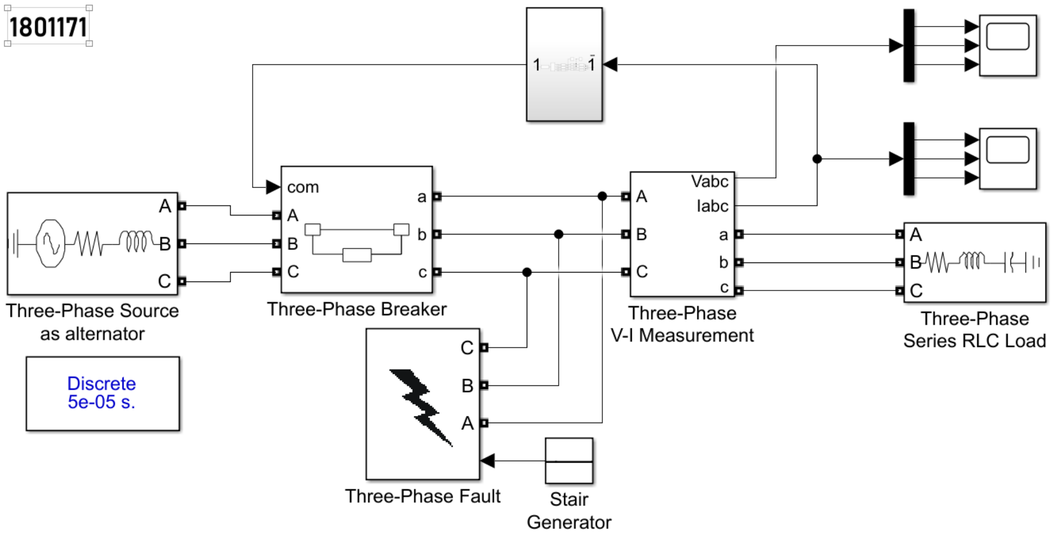


Fig. 10.1: Circuit diagram for 3LG and 1LG fault in transmission line

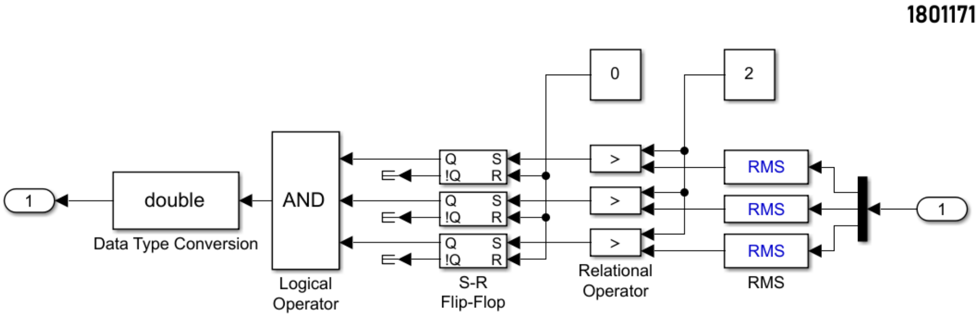


Fig. 10.2: Circuit diagram for logical subsystem

10.6 Waveform

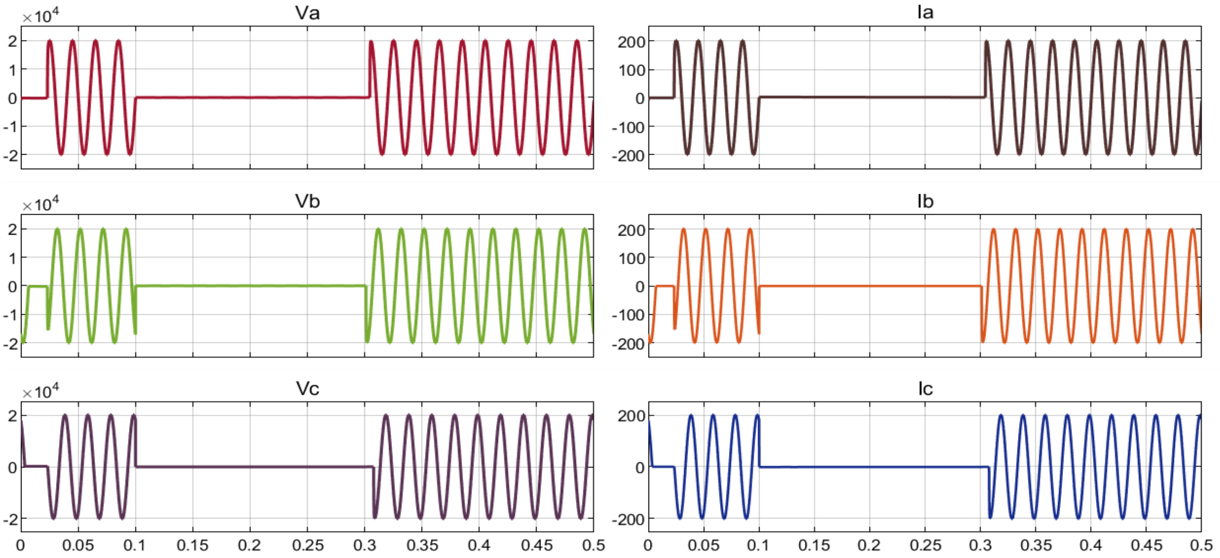


Fig. 10.3: Voltage and current waveform for 3LG fault in the transmission line

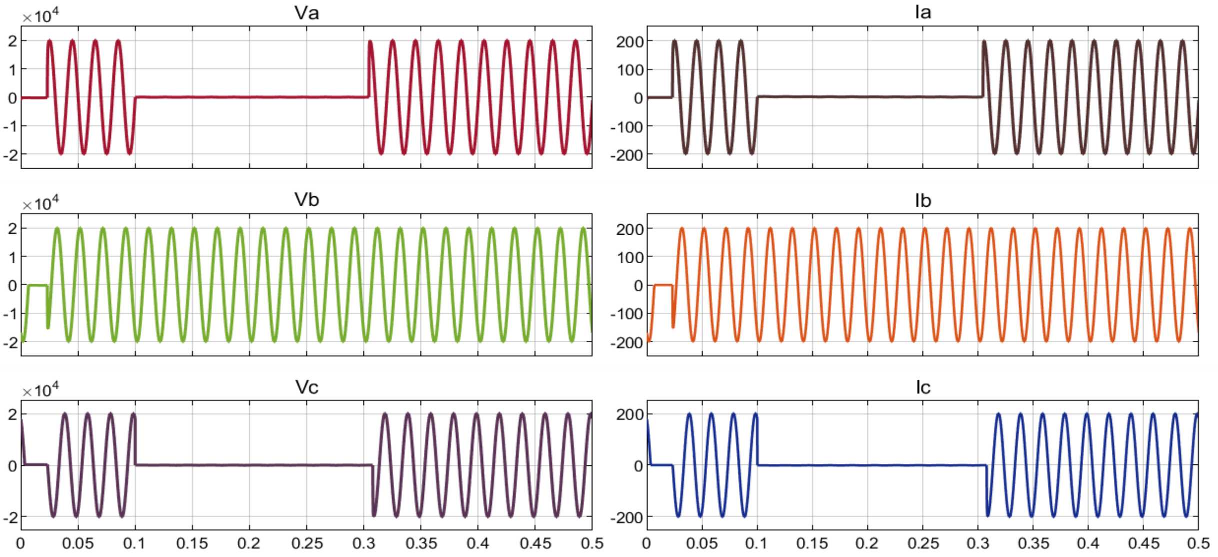


Fig. 10.4: Voltage and current waveform for LLG fault in the transmission line

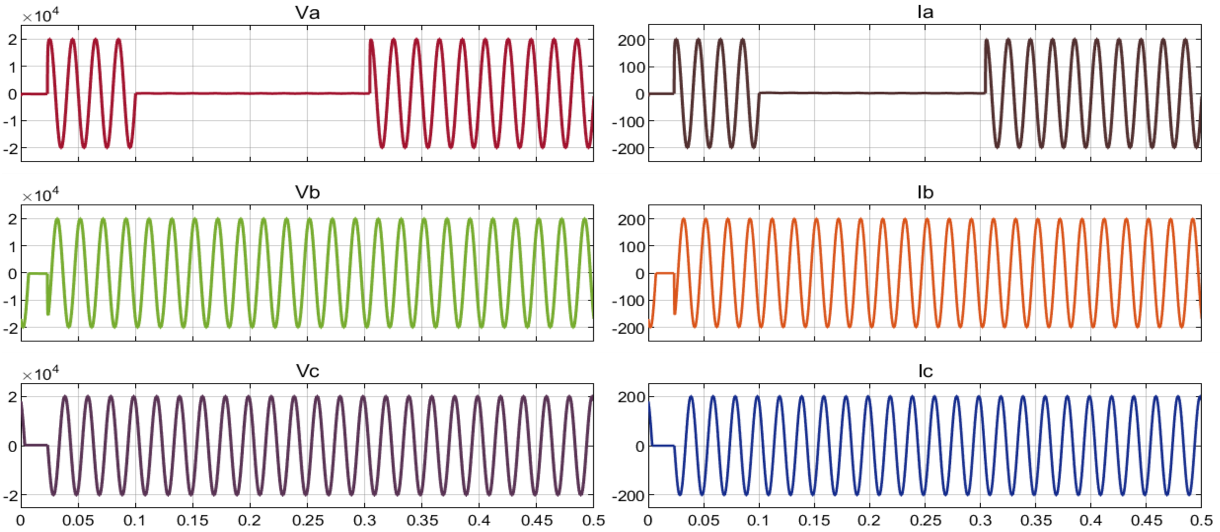


Fig. 10.5: Voltage and current waveform for 1LG fault in the transmission line

10.7 Discussion & Conclusion

The experiment taught us the fault analysis of power system in the Simulink platform. We tweaked the parameters of the generators, loads, and transmission lines after developing it.

Here, we also observed the response for 3LG and 1LG fault in transmission line to evaluate the system. We studied the system and replaced the system's values with new values. The system's objectives were met with success.

Experiment No. 11

11.1 Experiment Name

Study of the effect of the circuit breaker and observation of the response for 3LG and 1LG fault in transmission line using MATLAB Simulink

11.2 Objectives

- To become acquainted with the circuit breaker operation of system
- To understand the operation and simulate observation of the response for 3LG and 1LG fault in transmission line
- To get familiar with the procedure of designing and analyzing a power system in MATLAB

11.3 Theory

The fault analysis of a power system is required in order to provide information for the selection of switchgear, setting of relays and stability of system operation. Faults usually occur in a power system due to either insulation failure, flashover, physical damage or human error. These faults, may either be three phases in nature involving all three phases in a symmetrical manner, or may be asymmetrical where usually only one or two phases may be involved. The common types of asymmetrical faults occurring in a Power System are single line to ground faults and line to line faults, with and without fault impedance.

11.4 Required apparatus

- Simulink

11.5 Block diagram

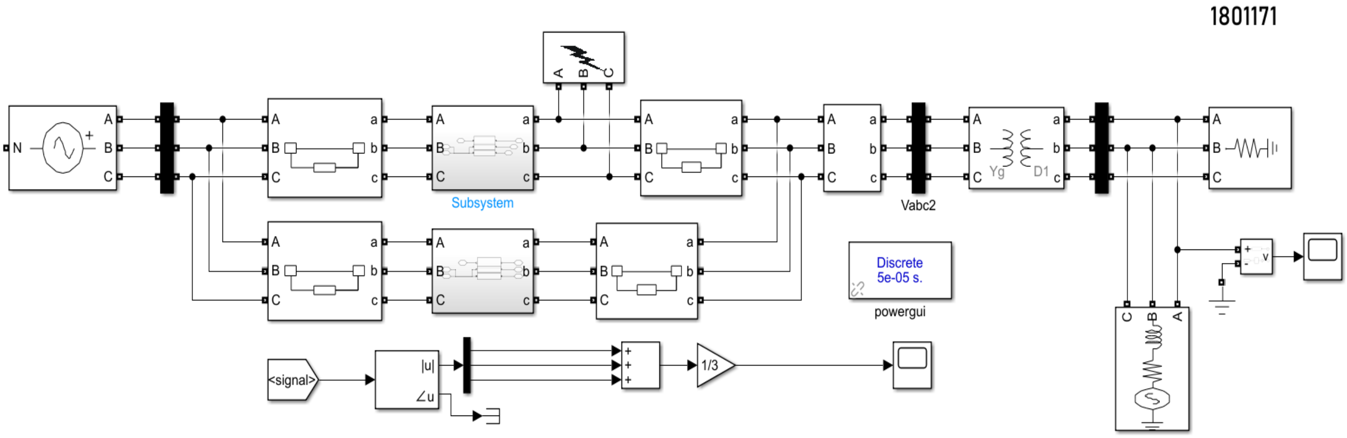


Fig. 10.1: Circuit diagram for 3LG and 1LG fault in transmission line

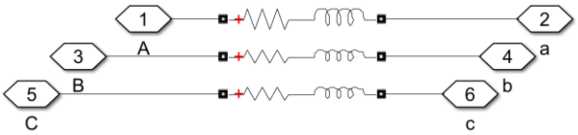


Fig. 10.2: Circuit diagram for logical subsystem

11.6 Waveform

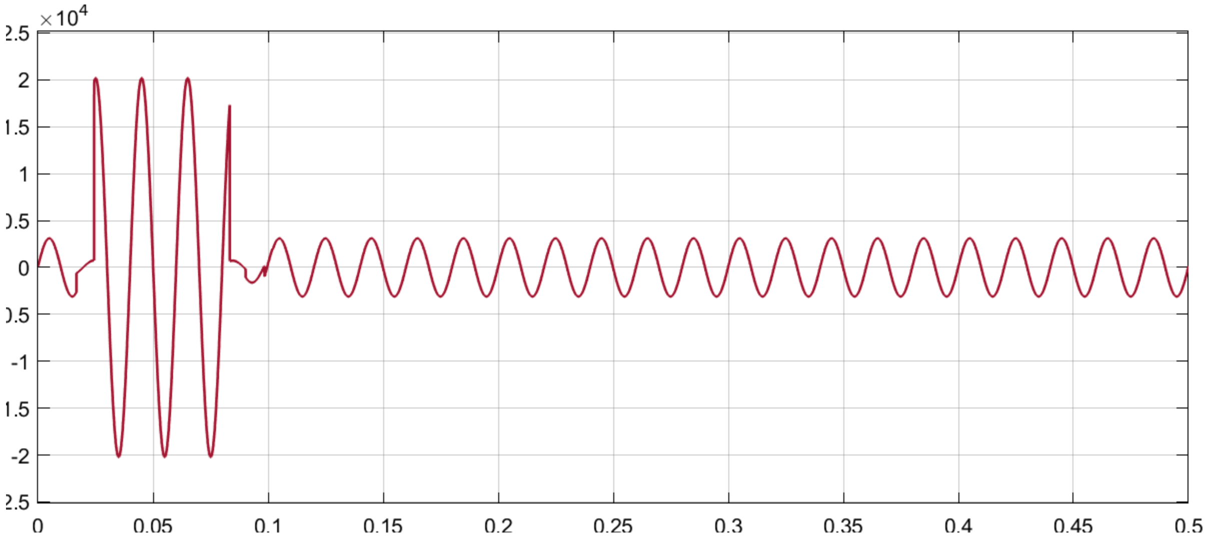


Fig. 10.3: Voltage waveform for 3LG fault in the transmission line

11.7 Discussion & Conclusion

The experiment taught us the fault analysis of power system in the Simulink platform. We tweaked the parameters of the generators, loads, and transmission lines after developing it.

Here, we also observed the response for 3LG and 1LG fault in transmission line to evaluate the system. We studied the system and replaced the system's values with new values. The system's objectives were met with success.