DEPARTMENT OF ELECTRICAL ENGINEERING

LAB MANUAL -2020

LAB NAME: POWER SYSTEM-II LAB (6EE4-21)

PREPERED BY: Mr. K.D. KANSAL

EXPERIMENT LIST

6EE4-21: POWER SYSTEM-II LAB

1. Fault analysis (for 3 to 6 bus) and verify the results using MATLAB or any available software for the cases:

(i) LG Fault (ii) LLG Fault (iii) LL Fault and (iv) 3-Phase Fault.

2. Load flow analysis for a given system (for 3 to 6 bus) using (i) Gauss Seidal (ii) Newton Raphson (iii) Fast Decoupled Method and verify results using MATLAB or any available software.

3. Three phase short circuit analysis in a synchronous machine (symmetrical fault analysis)

4. Study of voltage security analysis.

5. Study of overload security analysis and obtain results for the given problem using MATLAB or any software.
6. Study of economic load dispatch problem with different methods.
7. Study of transient stability analysis using MATLAB/ETAP Software. 8. Power flow analysis of a slack bus connected to different loads

Experiment NO:- 1

AIM:- Fault analysis (for 3 to 6 bus) and verify the results using MATLAB or any available software for the cases: (i) LG Fault (ii) LLG Fault (iii) LL Fault and (iv) 3-Phase Fault.

THEORY

Symmetrical Fault

Three phase fault

From the thevenin's equivalent circuit

$$\text{Fault current}, I_f'' = {}^{V}{}_{Z^{t}{}_{t}{}^{h}{}_{h}}$$

Where Vth = Thevenin's Voltage

Z th = Thevenin's Impedance

Unsymmetrical Fault

Single line to ground fault

Fault current, I_f = Ia = $3I_{a1}$

$$I_{a1} = \left[\frac{E}{Z_1 + Z_2 + Z_0}\right]$$

Line to line fault

Fault current, $I_f = I_{a1}(a^2 - a)$

$$I_{a1} = \frac{E}{Z_1 + aZ_2}$$

Double Line to ground fault

Fault current, $I_f = 2I_{a0} + (I_{a1} + I_{a2})(a^2 + a)$

$$\boldsymbol{I_{a1}} = \frac{\boldsymbol{Z_2 E(Z_0 + 3Z_f)}}{\boldsymbol{Z_1 + Z_2 + Z_0 + 3Z_f}}$$

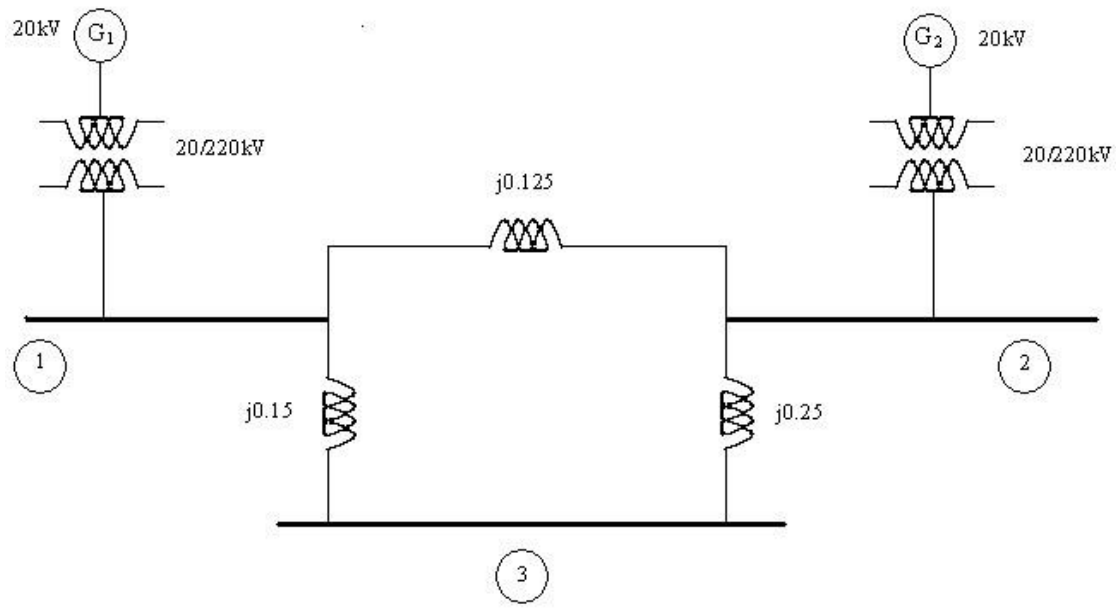
Fault MVA = 3 * I_f * V_{pu} where, I_{a1}, I_{a2} and I_{a0} are positive, negative and zero phase sequence currents.

Z_1, Z_2 and Z_o are positive, negative and zero phase sequence impedances.

PROCEDURE

1. Enter the command window of the MATLAB.
2. Create a new M – file by selecting File - New – M – File
3. Type and save the program.
4. Execute the program by pressing Tools – Run.
5. View the results.

EXERCISE



The one line diagram of a simple power system is shown in figure. The neutral of each generator is grounded through a current limiting reactor of 0.25/3 per unit on a 100MVA base. The system data expressed in per unit on a common 100 MVA base is tabulated below. The generators are running on no load at their rated voltage and rated frequency with their emf in phase.

Determine the fault current for the following faults.

(a) A balanced three phase fault at bus 3 through a fault impedance, Z_f = j0.1 per unit.

(b) A single line to ground fault at bus3 through a fault impedance, Z_f = j0.1 per unit.

(c) A line to line fault at bus3 through a fault impedance, Z_f = j0.1 per unit.

(d) A double line to ground fault at bus3 through a fault impedance, Z_f = j0.1 per unit.

Item	Base MVA	Voltage Rating KV	X_1	X_2	X_0
G_1	100	20	0.15	0.15	0.05
G_2	100	20	0.15	0.15	0.05
T_1	100	20/200	0.10	0.10	0.10
L_{12}	100	20/200	0.10	0.10	0.10
L_{13}	100	220	0.125	0.125	0.30
L_{23}	100	220	0.15	0.15	0.35
	100	220	0.25	0.25	0.7125

Verify the result using MATLAB program.

RESULT:-

(a) Symmetrical three-phase fault

```
Warning: Invalid escape sequence appears in format string. See help sprintf for valid
escape sequences.  > In Symfault at 34
  In Chp10ex7 at 17
Enter Faulted Bus No. -> 1

Enter Fault Impedance Zf = R + j*X in complex form (for bolted fault enter 0). Zf = .2

Balanced three-phase fault at bus No. 1
Total fault current =   4.0481 per unit

Bus Voltages during fault in per unit

    Bus     Voltage       Angle
    No.     Magnitude     degrees
1       0.8096     -35.9421
2       0.8256     -24.6322
3       0.8119     -31.6530

Line currents for fault at bus No.  1

    From      To     Current     Angle
    Bus       Bus    Magnitude   degrees
     G         1      2.3479    -35.9421
     1         F      4.0481    -35.9421
     G         2      1.7002    -35.9421
     2         1      1.2954    -35.9421
2         3      0.4048    -35.9421
3         1      0.4048    -35.9421
Another fault location? Enter 'y' or 'n' within single quote -> 'n'

(b) Line-to-ground fault

Line-to-ground fault analysis
Enter Faulted Bus No. -> 2

Enter Fault Impedance Zf = R + j*X in complex form (for bolted fault enter 0). Zf = 0

Single line to-ground fault at bus No. 2
Total fault current =    7.9708 per unit

Bus Voltages during the fault in per unit

    Bus    -------Voltage Magnitude-------
    No.    Phase a     Phase b     Phase c
```

```
        1      0.2972      0.9401      0.9401
        2      0.0000      0.9319      0.9319
        3      0.1896      0.9355      0.9355

Line currents for fault at bus No.  2

    From      To      -----Line Current Magnitude----
    Bus       Bus     Phase a     Phase b     Phase c
     1         2      1.9827      0.5679      0.5679
        1         3      0.6111      0.1860      0.1860
        2         F      7.9708      0.0000      0.0000
        3         2      0.6111      0.1860      0.1860
Another fault location? Enter 'y' or 'n' within single quote -> 'n'

(c) Line-to-line fault

Line-to-line fault analysis
Enter Faulted Bus No. -> 1

Enter Fault Impedance Zf = R + j*X in complex form (for bolted fault enter 0). Zf = 0

Line-to-line fault at bus No. 1
Total fault current =    5.9726 per unit

Bus Voltages during the fault in per unit

    Bus           -------Voltage   Magnitude-------
No.    Phase a     Phase b     Phase c
        1      1.0000      0.5000      0.5000
        2      1.0000      0.5541      0.5541
        3      1.0000      0.5080      0.5080

Line currents for fault at bus No.  1

    From     To     -----Line Current Magnitude----       Bus
Bus     Phase a     Phase b     Phase c
        1         F      0.0000      5.9726      5.9726
        2         1      0.0000      1.9112      1.9112
        2         3      0.0000      0.5973      0.5973
        3         1      0.0000      0.5973      0.5973
Another fault location? Enter 'y' or 'n' within single quote -> 'n'

(d) double line-to-ground fault
```

Double line-to-ground fault analysis
Enter Faulted Bus No. -> 1

Enter Fault Impedance Zf = R + j*X in complex form (for bolted fault enter 0). Zf = 0

Double line-to-ground fault at bus No. 1
Total fault current = 5.8939 per unit

Bus Voltages during the fault in per unit

Bus	-------Voltage Magnitude-------		
No.	Phase a	Phase b	Phase c
1	1.0727	0.0000	0.0000
2	0.9008	0.3756	0.3756
3	1.0196	0.1322	0.1322

Line currents for fault at bus No. 1

From	To	-----Line Current Magnitude----		Bus
Bus	Phase a	Phase b	Phase c	
1	F	0.0000	6.6601	6.6601
2	1	0.2063	2.2302	2.2302
2	3	0.0393	0.6843	0.6843
3	1	0.0393	0.6843	0.6843

Experiment NO:- 2

AIM:- Load flow analysis for a given system (for 3 to 6 bus) using Newton Raphson and Fast Decoupled Method and verify results using MATLAB or any available software.

THEORY:- The Newton Raphson method of load flow analysis is an iterative method which Approximates the set of non-linear simultaneous equations to a set of linear simultaneous equations using Taylor's series expansion and the terms are limited to first order approximation. The load flow equations for Newton Raphson method are non-linear equations in terms of real and imaginary part of bus voltages.

$$P_p = \sum_{nq=1}\{e_p(e_q G_{pq} + f_q B_{pq}) + f_p(f_q G_{pq} - e_q B_{pq})\}$$

$$Q_p = \sum_{nq=1}\{f_p(e_q G_{pq} + f_q B_{pq}) - e_p(f_q G_{pq} - e_q B_{pq})\}$$

Where, ep = Real part of Vp fp

= Imaginary part of Vp

Gpq, Bpq = Conductance and Susceptance of admittance Ypq respectively.

EXERCISE:- for IEEE 30 Bus system find the load flow using Newton raphson method

DATA

basemva = 100; accuracy = 0.001; accel = 1.8; maxiter = 100;

```
%       IEEE 30-BUS TEST SYSTEM (American Electric Power)
%       Bus Bus  Voltage Angle   ---Load---- -------Generator----- Static Mvar
%       No  code Mag.    Degree  MW    Mvar  MW  Mvar Qmin Qmax    +Qc/-Ql

busdata=[1   1    1.06    0.0     0.0   0.0    0.0  0.0   0   0       0
2    2    1.043   0.0   21.70  12.7   40.0  0.0 -40  50       0
3    0    1.0     0.0     2.4   1.2    0.0  0.0   0   0       0
4    0    1.06    0.0     7.6   1.6    0.0  0.0   0   0       0
```

```
5    2    1.01    0.0    94.2  19.0    0.0  0.0 -40  40       0
6    0    1.0     0.0     0.0   0.0    0.0  0.0   0   0       0
7    0    1.0     0.0    22.8  10.9    0.0  0.0   0   0       0
8    2    1.01    0.0    30.0  30.0    0.0  0.0 -30  40       0
9    0    1.0     0.0     0.0   0.0    0.0  0.0   0   0       0
10   0    1.0     0.0     5.8   2.0    0.0  0.0  -6  24      19
11   2    1.082   0.0     0.0   0.0    0.0  0.0   0   0       0
12   0    1.0     0       11.2  7.5    0    0     0   0       0
13   2    1.071   0       0     0.0    0    0    -6  24       0
14   0    1       0       6.2   1.6    0    0     0   0       0
15   0    1       0       8.2   2.5    0    0     0   0       0
16   0    1       0       3.5   1.8    0    0     0   0       0
17   0    1       0       9.0   5.8    0    0     0   0       0
18   0    1       0       3.2   0.9    0    0     0   0       0
19   0    1       0       9.5   3.4    0    0     0   0       0
20   0    1       0       2.2   0.7    0    0     0   0       0
21   0    1       0       17.5  11.2   0    0     0   0       0
22   0    1       0       0     0.0    0    0     0   0       0
23   0    1       0       3.2   1.6    0    0     0   0       0
24   0    1       0       8.7   6.7    0    0     0   0       4.3
25   0    1       0       0     0.0    0    0     0   0       0
26   0    1       0       3.5   2.3    0    0     0   0       0
27   0    1       0       0     0.0    0    0     0   0       0
28   0    1       0       0     0.0    0    0     0   0       0
29   0    1       0       2.4   0.9    0    0     0   0       0
30   0    1       0       10.6  1.9    0    0     0   0       0];

%      Line code
%       Bus bus   R       X      1/2 B   = 1 for lines %      nl
nr  p.u.   p.u.   p.u.    > 1 or < 1 tr. tap at bus nl linedata=[1
2   0.0192   0.0575   0.02640    1            1   3   0.0452
0.1852   0.02040    1
```

2	4	0.0570	0.1737	0.01840	1
3	4	0.0132	0.0379	0.00420	1
2	5	0.0472	0.1983	0.02090	1
2	6	0.0581	0.1763	0.01870	1
4	6	0.0119	0.0414	0.00450	1
5	7	0.0460	0.1160	0.01020	1
6	7	0.0267	0.0820	0.00850	1
6	8	0.0120	0.0420	0.00450	1
6	9	0.0	0.2080	0.0	0.978
6	10	0	.5560	0	0.969
9	11	0	.2080	0	1
9	10	0	.1100	0	1
4	12	0	.2560	0	0.932
12	13	0	.1400	0	1
12	14	.1231	.2559	0	1
12	15	.0662	.1304	0	1
12	16	.0945	.1987	0	1
14	15	.2210	.1997	0	1
16	17	.0824	.1923	0	1
15	18	.1073	.2185	0	1
18	19	.0639	.1292	0	1
19	20	.0340	.0680	0	1
10	20	.0936	.2090	0	1
10	17	.0324	.0845	0	1
10	21	.0348	.0749	0	1
10	22	.0727	.1499	0	1
21	22	.0116	.0236	0	1
15	23	.1000	.2020	0	1
22	24	.1150	.1790	0	1
23	24	.1320	.2700	0	1
24	25	.1885	.3292	0	1
25	26	.2544	.3800	0	1

```
25  27    .1093    .2087   0         1
28  27     0       .3960   0      0.968
27  29    .2198    .4153   0         1
27  30    .3202    .6027   0         1
29  30    .2399    .4533   0         1
 8  28    .0636    .2000   0.0214     1
 6  28    .0169    .0599   0.065      1];
```

lfybus % form the bus admittance matrix lfnewton % Load flow solution by Newton-Raphson method

RESULT:- Line Flow and Losses

```
    --Line--  Power at bus & line flow   --Line loss--  Transformer     from
to    MW      Mvar     MVA       MW      Mvar      tap

1       260.950  -17.010  261.504
2       177.743  -22.140  179.117    5.461   10.517
3       83.197    5.125   83.354    2.807    7.079

2        18.300   36.126   40.497
         1  -172.282   32.657  175.350    5.461   10.517
         4   45.702    2.720   45.783    1.106   -0.519
5   82.990    1.704   83.008    2.995    8.178
         6   61.905   -0.966   61.913    2.047    2.263

3        -2.400   -1.200    2.683
         1  -80.390    1.954   80.414    2.807    7.079
4        78.034   -3.087   78.095    0.771    1.345

4        -7.600   -1.600    7.767
         2  -44.596   -3.239   44.713    1.106   -0.519
         3  -77.263    4.432   77.390    0.771    1.345
         6   70.132  -17.624   72.313    0.605    1.181
        12   44.131   14.627   46.492    0.000    4.686    0.932

5        -94.200   16.995   95.721
         2  -79.995    6.474   80.256    2.995    8.178
         7  -14.210   10.467   17.649    0.151   -1.687
```

6		0.000	0.000	0.000		
	2	-59.858	3.229	59.945	2.047	2.263
	4	-69.527	18.805	72.026	0.605	1.181
7		37.537	-1.915	37.586	0.368	-0.598
8		29.534	-3.712	29.766	0.103	-0.558
9		27.687	-7.318	28.638	0.000	1.593	0.978
10		15.828	0.656	15.842	-0.000	1.279	0.969
	28	18.840	-9.575	21.134	0.060	-13.085
7		-22.800	-10.900	25.272		
	5	14.361	-12.154	18.814	0.151	-1.687
	6	-37.170	1.317	37.193	0.368	-0.598
8		-30.000	0.759	30.010		
	6	-29.431	3.154	29.599	0.103	-0.558
	28	-0.570	-2.366	2.433	0.000	-4.368
9		0.000	0.000	0.000		
	6	-27.687	8.911	29.086	0.000	1.593
	11	0.003	-15.653	15.653	0.000	0.461
10		27.731	6.747	28.540	-0.000	0.811
10		-5.800	17.000	17.962		
	6	-15.828	0.623	15.840	-0.000	1.279
	9	-27.731	-5.936	28.359	-0.000	0.811
	20	9.018	3.569	9.698	0.081	0.180
	17	5.347	4.393	6.920	0.014	0.037
	21	15.723	9.846	18.551	0.110	0.236
	22	7.582	4.487	8.811	0.052	0.107
11		0.000	16.113	16.113		
	9	-0.003	16.114	16.114	0.000	0.461
12		-11.200	-7.500	13.479		
	4	-44.131	-9.941	45.237	0.000	4.686
13		-0.021	-10.274	10.274	0.000	0.132
14		7.852	2.428	8.219	0.074	0.155
15		17.852	6.968	19.164	0.217	0.428
16		7.206	3.370	7.955	0.053	0.112
13		0.000	10.406	10.406		
	12	0.021	10.406	10.406	0.000	0.132
14		-6.200	-1.600	6.403		

12 -7.778 -2.273 8.103 0.074 0.155
15 1.592 0.708 1.742 0.006 0.006

15 -8.200 -2.500 8.573
12 -17.634 -6.540 18.808 0.217 0.428
14 -1.586 -0.702 1.734 0.006 0.006
18 6.009 1.741 6.256 0.039 0.079
23 5.004 2.963 5.815 0.031 0.063

16 -3.500 -1.800 3.936
12 -7.152 -3.257 7.859 0.053 0.112
17 3.658 1.440 3.931 0.012 0.027

17 -9.000 -5.800 10.707
16 -3.646 -1.413 3.910 0.012 0.027
10 -5.332 -4.355 6.885 0.014 0.037

18 -3.200 -0.900 3.324
15 -5.970 -1.661 6.197 0.039 0.079
19 2.779 0.787 2.888 0.005 0.010

19 -9.500 -3.400 10.090
18 -2.774 -0.777 2.881 0.005 0.010
20 -6.703 -2.675 7.217 0.017 0.034

20 -2.200 -0.700 2.309
19 6.720 2.709 7.245 0.017 0.034
10 -8.937 -3.389 9.558 0.081 0.180 21 -17.500 -11.200 20.777
10 -15.613 -9.609 18.333 0.110 0.236
22 -1.849 -1.627 2.463 0.001 0.001

22 0.000 0.000 0.000
10 -7.531 -4.380 8.712 0.052 0.107
21 1.850 1.628 2.464 0.001 0.001
24 5.643 2.795 6.297 0.043 0.067

23 -3.200 -1.600 3.578
15 -4.972 -2.900 5.756 0.031 0.063
24 1.771 1.282 2.186 0.006 0.012

24 -8.700 -2.400 9.025
22 -5.601 -2.728 6.230 0.043 0.067
23 -1.765 -1.270 2.174 0.006 0.012
25 -1.322 1.604 2.079 0.008 0.014

25	0.000	0.000	0.000		
24	1.330	-1.590	2.073	0.008	0.014
26	3.520	2.372	4.244	0.044	0.066
27	-4.866	-0.786	4.929	0.026	0.049

26	-3.500	-2.300	4.188		
25	-3.476	-2.306	4.171	0.044	0.066

27	0.000	0.000	0.000		
25	4.892	0.835	4.963	0.026	0.049
28	-18.192	-4.152	18.660	-0.000	1.310
29	6.178	1.675	6.401	0.086	0.162
30	7.093	1.663	7.286	0.162	0.304

28	0.000	0.000	0.000			
27	18.192	5.463	18.994	-0.000	1.310	0.968
8	0.570	-2.003	2.082	0.000	-4.368	
6	-18.780	-3.510	19.106	0.060	-13.085	

29	-2.400	-0.900	2.563		
27	-6.093	-1.513	6.278	0.086	0.162
30	3.716	0.601	3.764	0.034	0.063

30	-10.600	-1.900	10.769		
27	-6.932	-1.359	7.064	0.162	0.304
29	-3.683	-0.537	3.722	0.034	0.063

Total loss	17.594	22.233

Experiment NO:- 3

AIM:-. Load flow analysis for a given system (for 30 bus) using Gauss Seidal.

THEORY

The GAUSS – SEIDEL method is an iterative algorithm for solving a set of non-linear load flow equations. The non-linear load flow equation is given by

$$1 \quad P \qquad p-1 \qquad n$$

$$V_{pk+1} = Y_{pp} \left[\frac{P_p - V_{pk} jQ_p - \sum_{q=1}^{p-1} Y_{pq} V_{qk+1} - \sum_{q=p+1} V_{pk} q}{} \right]$$

The reactive power of bus-p is given by

$$Q_{pk+1} = (-1) x Im \left[(V_{pk}) \{ \sum_{q=1}^{p-1} Y_{pq} V_{qk+1} + \sum_{q=p}^{n} V_{qk} Y_{pq} \} \right]$$

PROCEDURE

1. Enter the command window of the MATLAB.
2. Create a new M – file by selecting File - New – M – File.
3. Type and save the program in the editor Window.
4. Execute the program by pressing Tools – Run.
5. View the results.

EXERCISE

For IEEE 30 bus system find the load flow solution using Gauss seidel Method Data is already given in previous Experiment

RESULT

Line Flow and Losses

--Line-- Power at bus & line flow --Line loss-- Transformer from to MW Mvar MVA MW Mvar tap

1 260.998 -17.021 261.553

2 177.778 -22.148 179.152 5.464 10.524

3 83.221 5.127 83.378 2.808 7.085

2 18.300 36.122 40.493

1 -172.314 32.671 175.384 5.464 10.524

4 45.712 2.705 45.792 1.106 -0.517

5 82.990 1.703 83.008 2.995 8.178

6 61.912 -0.958 61.920 2.048 2.264

3 -2.400 -1.200 2.683

1 -80.412 1.958 80.436 2.808 7.085

4 78.012 -3.158 78.076 0.771 1.344

4 -7.600 -1.600 7.767

2 -44.605 -3.222 44.722 1.106 -0.517

3 -77.242 4.503 77.373 0.771 1.344

6 70.126 -17.526 72.282 0.604 1.179

12 44.121 14.646 46.489 0.000 4.685 0.932

5 -94.200 16.975 95.717

2 -79.995 6.475 80.257 2.995 8.178

7 -14.205 10.500 17.664 0.151 -1.687

6 0.000 0.000 0.000

2 -59.864 3.222 59.951 2.048 2.264

4 -69.521 18.705 71.994 0.604 1.179

7 37.523 -1.885 37.570 0.367 -0.598

8 29.528 -3.754 29.766 0.103 -0.558

9 27.693 -7.322 28.644 0.000 1.594
0.978

10 15.823 0.653 15.836 0.000 1.278
0.969

28 18.819 -9.618 21.134 0.060 -13.086

-

7 -22.800 -10.900 25.272

5 14.356 12.187 18.831 0.151 -1.687

6 -37.156 1.287 37.178 0.367 -0.598

8 -30.000 0.826 30.011

6 -29.425 3.196 29.598 0.103 -0.558

28 -0.575 -2.370 2.438 0.000 -4.368

9 0.000 0.000 0.000

6 -27.693 8.916 29.093 0.000 1.594

11 0.000 -15.657 15.657 0.000 0.462

10 27.693 6.741 28.501 0.000 0.809

10 -5.800 17.000 17.962

6 -15.823 0.626 15.835 0.000 1.278

9 -27.693 -5.932 28.321 0.000 0.809

20 9.027 3.560 9.704 0.081 0.180

17 5.372 4.414 6.953 0.014 0.037

21 15.733 9.842 18.558 0.110 0.237

22 7.583 4.490 8.813 0.052 0.107

11 0.000 16.119 16.119

9 -0.000 16.119 16.119 0.000 0.462

12 -11.200 -7.500 13.479

4 -44.121 -9.961 45.232 0.000 4.685

13 0.000 -10.291 10.291 0.000 0.133

14 7.856 2.442 8.227 0.075 0.155

15 17.857 6.947 19.161 0.217 0.428

16 7.208 3.363 7.954 0.053 0.112

13 0.000 10.423 10.423

12 -0.000 10.424 10.424 0.000 0.133

-

14 -6.200 -1.600 6.403

12 -7.782 2.287 8.111 0.075 0.155

15 1.582 0.687 1.724 0.006 0.005

15 -8.200 -2.500 8.573

12 -17.640 -6.519 18.806 0.217 0.428

14 -1.576 -0.681 1.717 0.006 0.005

18 6.014 1.744 6.262 0.039 0.080

23 5.001 2.956 5.810 0.031 0.063

16 -3.500 -1.800 3.936

12 -7.154 -3.251 7.858 0.053 0.112

17 3.654 1.451 3.932 0.012 0.027

17 -9.000 -5.800 10.707

16 -3.643 -1.424 3.911 0.012 0.027

10 -5.357 -4.376 6.918 0.014 0.037

18 -3.200 -0.900 3.324

15 -5.975 -1.665 6.203 0.039 0.080

19 2.775 0.765 2.879 0.005 0.010

19 -9.500 -3.400 10.090

18 -2.770 -0.755 2.871 0.005 0.010

20 -6.730 -2.645 7.231 0.017 0.034

20 -2.200 -0.700 2.309

19 6.747 2.679 7.259 0.017 0.034

10 -8.947 -3.379 9.564 0.081 0.180

21 -17.500 -11.200 20.777

10 -15.623 -9.606 18.340 0.110 0.237

-

22	-1.877	-1.594	2.462	0.001	0.001
22		0.000	0.000	0.000	
10	-7.531	4.384	8.714	0.052	0.107

-

21 1.877 1.596 2.464 0.001 0.001
24 5.654 2.788 6.304 0.043 0.067

23 -3.200 -1.600 3.578
15 -4.970 -2.893 5.751 0.031 0.063
24 1.770 1.293 2.192 0.006 0.012

24 -8.700 -2.400 9.025
22 -5.611 -2.721 6.236 0.043 0.067
23 -1.764 -1.280 2.180 0.006 0.012
25 -1.325 1.602 2.079 0.008 0.014

25 0.000 0.000 0.000
24 1.333 -1.588 2.073 0.008 0.014
26 3.545 2.366 4.262 0.045 0.066
27 -4.877 -0.778 4.939 0.026 0.049

26 -3.500 -2.300 4.188
25 -3.500 -2.300 4.188 0.045 0.066

27 0.000 0.000 0.000
25 4.903 0.827 4.972 0.026 0.049
28 -18.184 -4.157 18.653 0.000 1.309
29 6.189 1.668 6.410 0.086 0.162
30 7.091 1.661 7.283 0.161 0.304

28 0.000 0.000 0.000
27 18.184 5.466 18.987 0.000 1.309 0.968
8 0.575 -1.999 2.080 0.000 -4.368
6 -18.759 -3.467 19.077 0.060 -13.086

29 -2.400 -0.900 2.563
27 -6.104 -1.506 6.286 0.086 0.162
30 3.704 0.606 3.753 0.033 0.063

30 -10.600 -1.900 10.769

27 -6.930 -1.358 7.062 0.161 0.304

29 -3.670 -0.542 3.710 0.033 0.063 Total loss 17.599 22.244

Experiment NO:- 4

AIM:-Study of voltage security analysis.

THEORY:- Reliable operation of large scale electric power networks requires that system voltages and currents stay within design limits. Operation beyond those limits can lead to equipment failures and blackouts. Security margins measure the amount by which system loads or power transfers can change before a security violation, such as an overloaded transmission line, is encountered. This thesis shows how to efficiently compute security margins defined by limiting events and instabilities, and the sensitivity of those margins with respect to assumptions, system parameters, operating policy, and transactions. Security margins to voltage collapse blackouts, oscillatory instability, generator limits, voltage constraints and line overloads are considered. The usefulness of computing the sensitivities of these margins with respect to interarea transfers, loading parameters, generator dispatch, transmission line parameters, and VAR support is established for networks as large as 1500 buses. The sensitivity formulas presented apply to a range of power system models.

Conventional sensitivity formulas such as line distribution factors, outage distribution factors, participation factors and penalty factors are shown to be special cases of the general sensitivity formulas derived in this thesis. The sensitivity formulas readily accommodate sparse matrix techniques. Margin sensitivity methods are shown to work effectively for avoiding voltage collapse blackouts caused by either saddle node bifurcation of equilibria or immediate instability due to generator reactive power limits. Extremely fast contingency analysis for voltage collapse can be implemented with margin sensitivity based rankings. Interarea transfer can be limited by voltage limits, line limits, or voltage stability. The sensitivity formulas presented in this thesis apply to security margins defined by any limit criteria. A method to compute transfer margins by directly locating intermediate events reduces the total number of loadflow iterations required by each margin computation and provides sensitivity information at minimal additional cost. Estimates of the effect of simultaneous transfers on the transfer margins agree well with the exact computations for a network model derived from a portion of the U.S grid. The accuracy of the estimates over a useful range of conditions and the ease of obtaining the estimates suggest that the sensitivity computations will be of practical value.

Voltage stability margin is a measure of the available transfer capacity, net transfer capacity, or total transfer capacity. The margin is the difference (or a ratio) between operation and voltage collapse points based on the KSP (loading, line flow, etc.) and accounts for a pattern of load increase or generation loss. As a concept for system operators, margin is a straightforward and easily understood index, and thus, widely accepted. There are a number of advantages of the stability margin as a collapse index.

- The margin is not based on a particular power system model and can be used with static or dynamic models independent of the details of the power system dynamics.
- It is an accurate index that takes full account of the power system non-linearity and device limits as loading is increased.
- Sensitivity analysis may be applied easily and quickly to study the effects of power system parameters and controls.
- The margin accounts for patterns of load increase.

The primary disadvantage is that it may oversimplify the view of the stability problem and may not account for the variety of ways in which instabilities can arise. In theory, the computation of the stability margin should be performed for all contingencies. This would be an excessively time-consuming process but is generally not necessary in practice. Instead, the margin is determined based on the most critical contingency from a relatively short list of known severe contingencies. Key to the analysis is the degree of experience that allows one to identify a more manageable list of disturbances. Still, the precise computation of the margin is timeconsuming, thus limiting application for on-line use. The most common methods to estimate the proximity of the voltage collapse point are the minimum singular value, point of collapse method, continuation power flow, and optimization methods. Some other methods are sensitivity analysis, second order performance index, and the energy function method.

RESULT:-We have study about voltage security analysis.

Experiment NO:- 5

AIM : Three phase short circuit analysis in a synchronous machine (symmetrical fault

analysis).

PROGRAM REQUIRED: MATLAB 5.3

THEORY :

Symmetrical Fault :

Three phase fault :

From the thevenin's equivalent circuit

$$\text{Fault current } I_f'' = \frac{V_{th}}{Z_{th}}$$

Where V_{th} = Thevenin's Voltage

Z_{th} = Thevenin's Impedance

Unsymmetrical Fault :

Single line to ground fault :

Fault current $I_f = I_a = 3I_{a1}$

$$I_{a1} = E_a \left[\frac{}{Z_1+Z_2+Z_0} \right]$$

Line to line fault:

Fault current $I_f = I_{a1}(a^2 - a)$

$$I_{a1} = \frac{E_a}{Z_1+Z_2}$$

Double Line to ground fault :

Fault current $I_f = 2\, I_{a0} + (I_{a1} + I_{a2})\,(a^2 + a)$

$$I_{a1} = \frac{E_a}{\left[Z_1 + \frac{Z_0 Z_2}{Z_0+Z_2} \right]}$$

$$I_{a2} = (-I_{a1}) * \frac{Z_0}{}$$

Z_0+Z_2

$$I_{a0} = -(I_{a1} - I_{a2})$$

Fault MVA = $\sqrt{3} * I_f * V_{pu}$

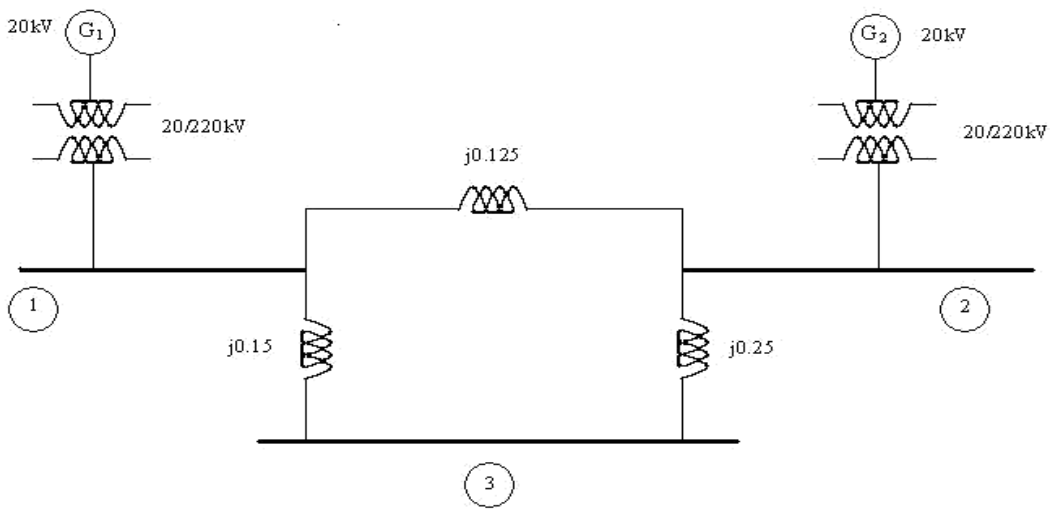
where, I_{a1}, I_{a2} and I_{a0} are positive, negative and zero phase sequence currents

Z_1 ,Z_2 and Z_o are positive, negative and zero phase sequence impedances

PROCEDURE:

1. Enter the command window of the MATLAB.
2. Create a new M – file by selecting File - New – M – File
3. Type and save the program.
4. Execute the program by either pressing Tools – Run. View the results.

EXERCISE :



The one line diagram of a simple power system is shown in figure. The neutral of each generator is grounded through a current limiting reactor of 0.25/3 per unit on a 100MVA base. The system data expressed in per unit on a common 100 MVA base is tabulated below. The generators are running on no load at their rated voltage and rated frequency with their emfs in phase.

Determine the fault current for the following faults.

(a) A balanced three phase fault at bus 3 through a fault impedance Z_f = j0.1 per unit.

(b) A single line to ground fault at bus3 through a fault impedance Z_f = j0.1 per unit.

(c) A line to line fault at bus3 through a fault impedance Z_f = j0.1 per unit.

(d) A double line to ground fault at bus3 through a fault impedance Z_f = j0.1 per unit.

Item	Base MVA	Voltage Rating kV	X_1	X_2	X_0
G_1	100	20	0.15	0.15	0.05
G_2	100	20	0.15	0.15	0.05
T_1	100	20/220	0.10	0.10	0.10
T_2	100	20/220	0.10	0.10	0.10
L_{12}	100	220	0.125	0.125	0.30
L_{13}	100	220	0.15	0.15	0.35
L_{23}	100	220	0.25	0.25	0.7125

Verify the result using MATLAB program.

PROGRAM:

```
zdata1 = [0 1 0 0.25
          0 2 0 0.25
          1 2 0 0.125
1 3 0 0.15
2 3 0 0.25];
zdata0 = [0 1 0 0.40
0 2 0 0.10
1 2 0 0.30
1 3 0 0.35
2 3 0 0.7125];
zdata2 = zdata1;
Zbus1 = zbuild(zdata1)
```

```
Zbus0 = zbuild(zdata0)
Zbus2     =     Zbus1;
symfault(zdata1,Zbus1)
lgfault(zdata0, Zbus0, zdata1, Zbus1, zdata2, Zbus2)
llfault(zdata1, Zbus1, zdata2, Zbus2) dlgfault(zdata0,
Zbus0, zdata1, Zbus1, zdata2, Zbus2)
```

Experiment NO:- 6

AIM:-. Study of economic load dispatch problem with different methods

THEORY:-Mathematical Model for Economic Dispatch of Thermal Units Without Transmission Loss Statement of Economic Dispatch Problem

In a power system, with negligible transmission loss and with N number of spinning thermal generating units the total system load PD at a particular interval can be met by different sets of generation schedules

$\{PG_{1(k)}, PG_{2(k)}, \ldots\ldots\ldots\ldots\ldots\ldots PG_{N(K)}\}$; k = 1,2,……..NS

Out of these NS set of generation schedules, the system operator has to choose the set of schedules, which minimize the system operating cost, which is essentially the sum of the production cost of all the generating units. This economic dispatch problem is mathematically stated as an optimization problem.

Given : The number of available generating units N, their production cost functions, their operating limits and the system load PD, **To determine :** The set of generation schedules,

PG_i ; i = 1,2……N

Which minimize the total production cost,

Min ; $F_T = \sum F_i (PG_i)$

and satisfies the power balance constraint

$PG_i - PD = 0$ and the operating limits

$PG_{i,min} \leq PG_i \leq PG_{i,,max}$

The units production cost function is usually approximated by quadratic function

$F_i (PG_i) = a_i PG_{2i} + b_i PG_i + c_i$; i = 1,2,…….N where

a_i , b_i and c_i are constants

Necessary conditions for the existence of solution to ED problem

The ED problem given by the equations (1) to (4). By omitting the inequality constraints (4) tentatively, the reduce ED problem (1),(2) and (3) may be restated as an unconstrained

optimization problem by augmenting the objective function (1) with the constraint multiplied by LaGrange multiplier, to obtained the LaGrange function, L as

Min : L (PG_1 ……..PG_N ,) =□ $F_i(PG_i)$ -□ [Σ PG_i – PD]

The necessary conditions for the existence of solution to (6) are given by

$$\frac{\partial L}{\partial PG_i} = 0 = \frac{d}{dPG_I} F_i(PG_i) - \lambda;$$

$$\partial L / \partial \lambda = 0 = \sum_{i=1}^{N} PG_i - PD$$

The solution to ED problem can be obtained by solving simultaneously the necessary conditions (7) and (8) which state that the economic generation schedules not only satisfy the system power balance equation (8) but also demand that the incremental cost rates of all the units be equal be equal to□ which can be interpreted as "incremental cost of received power".

When the inequality constraints (4) are included in the ED problem the necessary condition (7) gets modified as dFi (PGi) / dPGi = l for PGi,min £ PGi £ PGi, ,max

£ l for PGi = PGi, ,max

` l for PGi = PGi, ,min

Economic Schedule

PGi = (l -bi)/ 2ai ; i=1,2…….N λ

Incremental fuel cost

$$\lambda = [PD + \sum_{i=1}^{N}(b_i/2a_i)] / \sum_{i=1}^{N} 1/2a_i$$

PROCEDURE

1. Enter the command window of the MATLAB.
2. Create a new M – file by selecting File - New – M – File
3. Type and save the program.
4. Execute the program by either pressing Tools – Run.
5. View the results

EXERCISE

1. The fuel cost functions for three thermal plants in $/h are given by

C1 = 500 + 5.3 P1 + 0.004 P1

2 ; P1 in MW

C2 = 400 + 5.5 P2 + 0.006 P2

2 ; P2 in MW

C3 = 200 +5.8 P3 + 0.009 P3

2 ; P3 in MW

The total load, PD is 800MW.Neglecting line losses and generator limits, find the optimal dispatch and the total cost in $/h by analytical method. Verify the result using MATLAB program.

RESULT:

Total generation cost = 8236.25 $/h

Incremental cost of delivered power (system lambda) = 8.500000 $/MWh Optimal Dispatch of Generation:

400.0000

250.0000

150.0000

Total system loss = 0 MW

Total generation cost = 6682.50 $/h

Experiment NO:- 7

AIM:- Study of transient stability analysis using MATLAB/ETAP Software.

THEORY

Stability: Stability problem is concerned with the behavior of power system when it is subjected to disturbance and is classified into small signal stability problem if the disturbances are small and transient stability problem when the disturbances are large.

Transient stability: When a power system is under steady state, the load plus transmission loss equals to the generation in the system. The generating units run at synchronous speed and system frequency, voltage, current and power flows are steady. When a large disturbance such as three phase fault, loss of load, loss of generation etc., occurs the power balance is upset and the generating units rotors experience either acceleration or deceleration. The system may come back to a steady state condition maintaining synchronism or it may break into subsystems or one or more machines may pull out of synchronism. In the former case the system is said to be stable and in the later case it is said to be unstable.

Small signal stability: When a power system is under steady state, normal operating condition, the system may be subjected to small disturbances such as variation in load and generation, change in field voltage, change in mechanical toque etc., the nature of system response to small disturbance depends on the operating conditions, the transmission system strength, types of controllers etc. Instability that may result from small disturbance may be of two forms, (iii) Steady increase in rotor angle due to lack of synchronizing torque. (iv) Rotor oscillations of increasing magnitude due to lack of sufficient damping torque.

FORMULA

Reactive power Qe = sin(cos-1(p.f))

Reactive power $Q_e = sin\left(cos^{-1}(p.f)\right)$

Stator Current $I_t = \frac{S}{E_{*t}}$

$$= \frac{P_e - E_t jQ_*}{e}$$

Voltage behind transient condition $E' = E_t + jX'_d I_t$

Voltage of infinite bus $E_B = E_t - (X_3 + X_{tr})I_t$

Where, $X_3 = \frac{X}{X_{11} + XX_{22}}$

Angular separation between E_B and E'

$$\delta_0 = E' - E_B \text{ Pre}$$

fault operation:

$$X = jX_d' + jX_{tr} + \frac{X_{11}}{X} + \frac{X_{22}}{X}$$

Power $P_e = \frac{E'}{X} E_B \sin \delta_0$

$$where, \delta_0 = sin^{-1}\left[\frac{e}{*}\right]_{EP' * EX_B}$$

During fault condition:

$$\delta_{max} = \pi - \delta_0$$

$$P_e = \frac{P_{\delta m}}{\sin_{max}}$$

Critical clearing angle:

$$Cos\ \delta_{cr} = \frac{P_m(\delta_{max} - \delta_0)}{+PP_{3max}\delta_{max} - CosP_{2\ max}\delta_{max} - P_{2max}Cos\ \delta_0}$$

Critical clearing time:

$$t_{cr} = \sqrt{\frac{2H\pi(\delta f_{cr0}P - m\delta_0)}{}} \text{ Secs}$$

PROCEDURE

1. Enter the command window of the MATLAB.
2. Create a new M – file by selecting File - New – M – File
3. Type and save the program.

4. Execute the program by pressing Tools – Run

5. View the results.

EXERCISE

1. Transient stability analysis of a 9-bus, 3-machine, 60 Hz power system with the following system modeling requirements:

i. Classical model for all synchronous machines, models for excitation and speed governing systems not included.

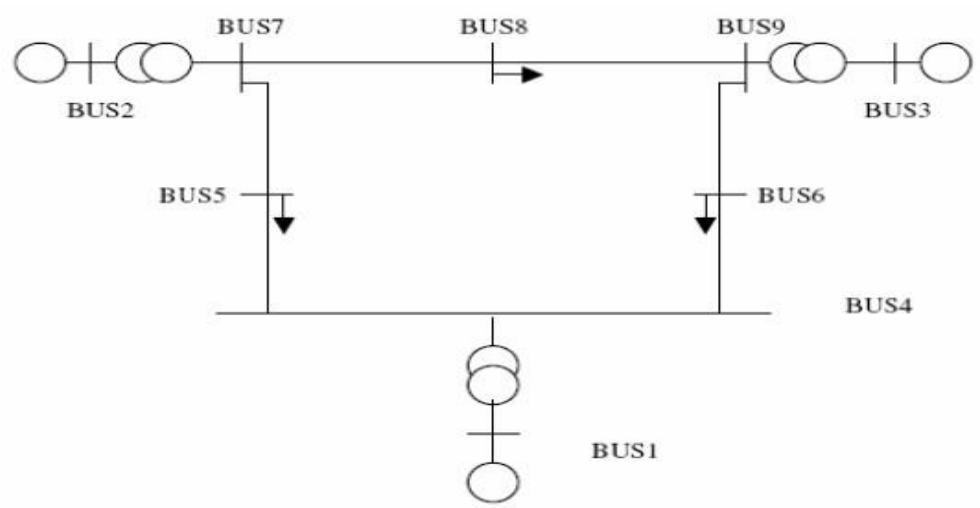
(a) Simulate a three-phase fault at the end of the line from bus 5 to bus 7 near bus 7 at time = 0.0 sec. Assume that the fault is cleared successfully by opening the line 5-7 after 5 cycles (0.083 sec)

. Observe the system for 2.0 seconds

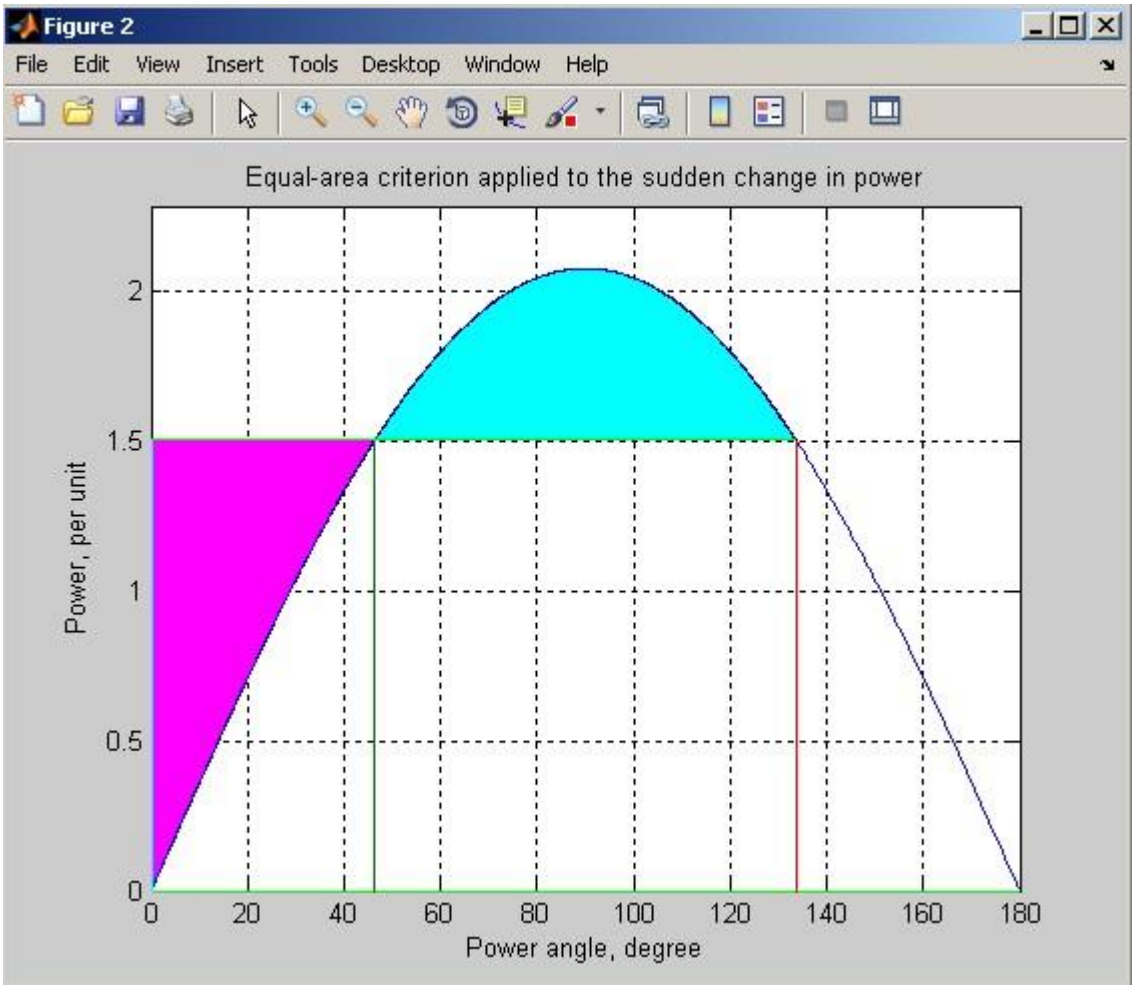
(b) Obtain the following time domain plots:

- Relative angles of machines 2 and 3 with respect to machine 1
- Angular speed deviations of machines 1, 2 and 3 from synchronous speed - Active power variation of machines 1, 2 and 3.

(c) Determine the critical clearing time by progressively increasing the fault clearing time.



RESULT



Experiment NO:- 8

Aim: To perform power flow analysis of a slack bus connected to different loads.

Apparatus: MATLAB-PSAT

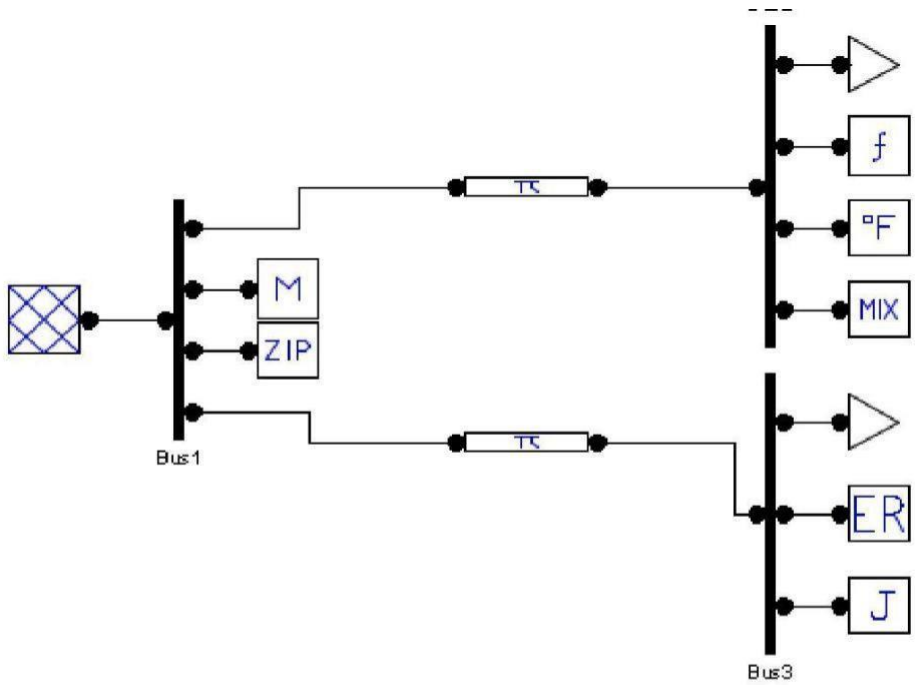
Theory:

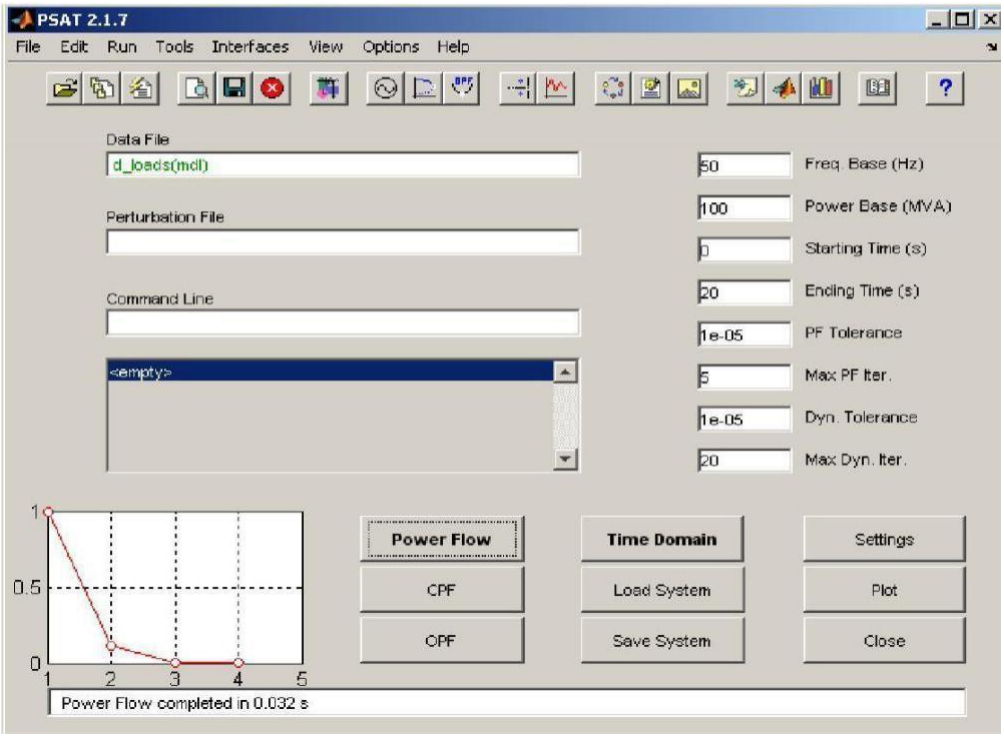
Slack Bus: To calculate the angles θi (as discussed above), a reference angle (θi = 0) needs to be specified so that all the other bus voltage angles are calculated with respect to this reference angle. Moreover, physically, total power supplied by all the generation must be equal to the sum of total load in the system and system power loss. However, as the system loss cannot be computed before the load flow problem is solved, the real power output of all the generators in the system cannot be pre-specified. There should be at least one generator in the system which would supply the loss (plus its share of the loads) and thus for this generator, the real power output can't be pre-specified. However, because of the exciter action, Vi for this generator can still be specified. Hence for this generator, Vi and θi(= 0) are specified and the quantities Pi and Qi are calculated. This generator bus is designated as the slack bus. Usually, the largest generator in the system is designated as the slack bus.

In the General Load Flow Problem the Bus with largest generating capacity is taken as the Slack Bus or Swing Bus. The Slack Bus Voltage is taken to be 1.0 + j 0 P.U. and should be capable of supplying total Losses in the System. But usually the generator bus are only having station auxiliary which may be only up to 3% of total generation . If the Generation at Slack Bus is more it can take more load connected to the slack bus.

A slack bus is usually a generator bus with a large real and reactive power output. It is assumed that its real and reactive power outputs are big enough that they can be adjusted as required in order to balance the power in the whole system so that the power flow can be solved. A slack bus can have load on it because in real systems it is actually the bus of a power plant, which can have its own load. It also takes care of the Line losses

Circuit Diagram:

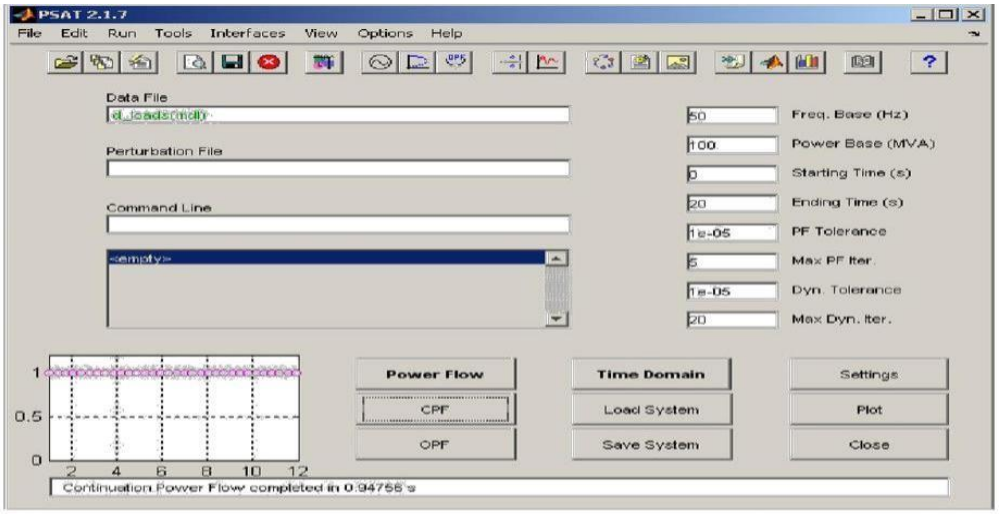


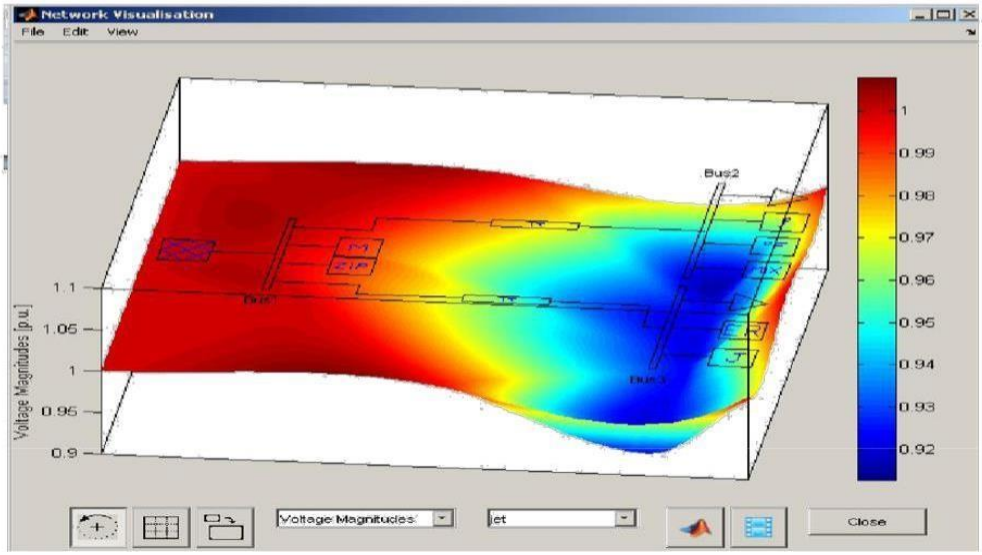


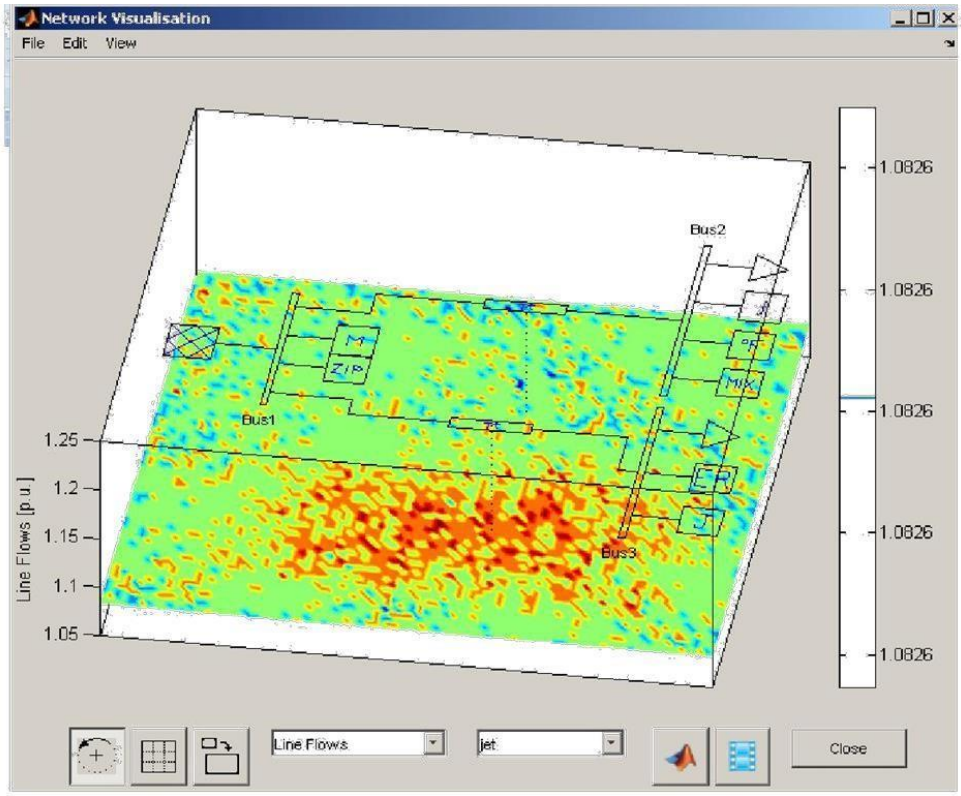
Procedure :

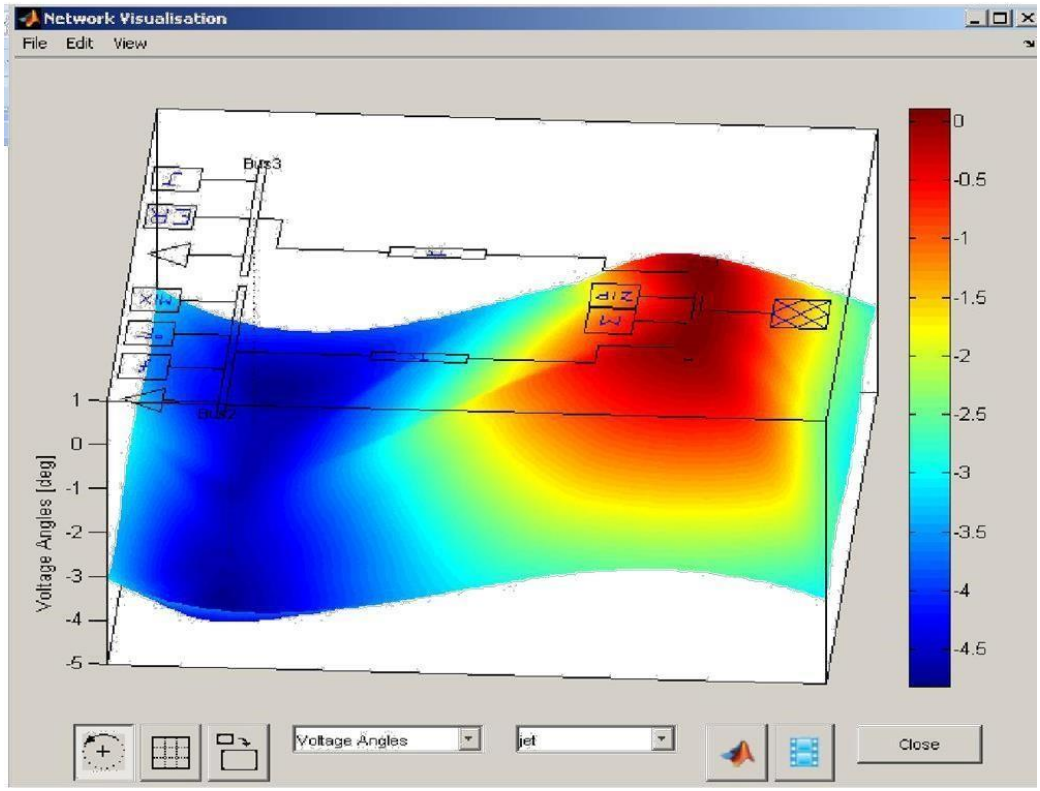
- Create a new file in MATLAB-PSAT
- Browse the components in the library and build the bus system.

- Save the file and upload the data file in PSAT main window
- Execute the program and run power flow
- Get the network visualization.






Network Visualisation
File Edit View
Bus1
Bus2
Bus3
Line Flows [p.u.]
1.25
1.2
1.15
1.1
1.05
1.0826
1.0826
1.0826
1.0826
1.0826
Line Flows
jet
Close


Network Visualisation
File Edit View
Bus3
Voltage Angles [deg]
1
0
-1
-2
-3
-4
-5
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
-4
-4.5
Voltage Angles
jet
Close

APPENDIX

PROGRAM 1

```
alpha = [500; 400; 200];
beta = [5.3; 5.5; 5.8]; gamma = [0.004; 0.006; 0.009];
PD = 800; DelP = 10; lamda = input('Enter
estimated value of Lamda = ');
fprintf(' ') disp(['Lamda P1
P2 P3 DP'... ' grad
Delamda']) iter = 0; while
abs(DelP) >= 0.001 iter =
iter + 1;
P = (lamda - beta)./(2*gamma);
DelP = PD - sum(P);
J = sum(ones(length(gamma),1)./(2*gamma));
Delamda = DelP/J;
disp([lamda,P(1),P(2),P(3),DelP,J,Delamda])
lamda = lamda + Delamda; end totalcost =
sum(alpha + beta.*P + gamma.*P.^2)
```

PROGRAM: 2

```
Pm = 0.8; E = 1.17; V = 1.0; X1
= 0.65; X2 = inf; X3 = 0.65;
eacfault (Pm, E, V, X1, X2, X3)
For b)
Pm = 0.8; E = 1.17; V = 1.0; X1
= 0.65; X2 = 1.8; X3 = 0.8;
eacfault (Pm, E, V, X1, X2, X3)
```

PROGRAM 3

```
function eacpower(P0, E, V, X)

if exist('P0')~=1
P0 = input('Generator initial power in p.u. P0 = '); else, end if
exist('E')~=1
```

```
E = input('Generator e.m.f. in p.u. E = '); else, end if
exist('V')~=1
V = input('Infinite bus-bar voltage in p.u. V = '); else, end if
exist('X')~=1
X = input('Reactance between internal emf and infinite bus in p.u. X = '); else, end
Pemax= E*V/X; if
P0 >= Pemax
  fprintf('\nP0 must be less than the peak electrical power Pemax = %5.3f p.u. \n', Pemax)
fprintf('Try again. \n\n') return, end
d0=asin(P0/Pemax);
delta = 0:.01:pi; Pe =
Pemax*sin(delta);
dmax=pi; Ddmax=1;
while abs(Ddmax) > 0.00001
Df = cos(d0) - (sin(dmax)*(dmax-d0)+cos(dmax));
J=cos(dmax)*(dmax-d0);
Ddmax=Df/J;
dmax=dmax+Ddmax;
end dc=pi-dmax;
Pm2=Pemax*sin(dc);
Pmx =[0  pi-d0]*180/pi; Pmy=[P0 P0]; Pm2x=[0
dmax]*180/pi; Pm2y=[Pm2 Pm2];
x0=[d0 d0]*180/pi; y0=[0 Pm2]; xc=[dc dc]*180/pi; yc=[0 Pemax*sin(dc)];
xm=[dmax dmax]*180/pi; ym=[0 Pemax*sin(dmax)];
d0=d0*180/pi; dmax=dmax*180/pi; dc=dc*180/pi;
x=(d0:.1:dc); y=Pemax*sin(x*pi/180);
%y1=Pe2max*sin(d0*pi/180);
%y2=Pe2max*sin(dc*pi/180);
x=[d0  x  dc];  y=[Pm2  y
Pm2];      xx=dc:.1:dmax;
h=Pemax*sin(xx*pi/180)
;   xx=[dc    xx       dmax];
hh=[Pm2     h          Pm2];
delta=delta*180/pi;
%clc
fprintf('\nInitial power                    =%7.3f p.u.\n', P0)
fprintf('Initial power angle               =%7.3f degrees \n', d0)
fprintf('Sudden additional power           =%7.3f p.u.\n', Pm2-P0)
fprintf('Total power for critical stability =%7.3f p.u.\n', Pm2)
fprintf('Maximum angle swing               =%7.3f degrees \n', dmax)
fprintf('New operating angle               =%7.3f degrees \n\n\n', dc)
fill(x,y,'m') hold; fill(xx,hh,'c')
plot(delta, Pe,'-', Pmx, Pmy,'g', Pm2x,Pm2y,'g', x0,y0,'c', xc,yc, xm,ym,'r'), grid
Title('Equal-area criterion applied to the sudden change in power') xlabel('Power
angle, degree'), ylabel(' Power, per unit')
```

```
axis([0 180  0 1.1*Pemax])
hold off;
```



```
1 2 0 0.125
1 3 0 0.15
2 3 0 0.25];
zdata0 = [0 1 0 0.40
0 2 0 0.10
1 2 0 0.30
1 3 0 0.35
2 3 0 0.7125]; zdata2 =
  zdata1; Zbus1 =
  zbuild(zdata1)
Zbus0 = zbuild(zdata0) Zbus2 = Zbus1;
symfault(zdata1,Zbus1) lgfault(zdata0, Zbus0,
zdata1, Zbus1, zdata2, Zbus2) llfault(zdata1, Zbus1,
zdata2, Zbus2) dlgfault(zdata0, Zbus0, zdata1, Zbus1,
zdata2, Zbus2)
```

Double Line to Ground Fault Program

PROGRAM 5

```
function dlgfault(zdata0, Zbus0, zdata1, Zbus1, zdata2, Zbus2, V)

if exist('zdata2') ~= 1
zdata2=zdata1; else,
end if exist('Zbus2')
~= 1
Zbus2=Zbus1;
else, end
```

```
nl = zdata1(:,1); nr = zdata1(:,2); nl0 = zdata0(:,1); nr0
= zdata0(:,2); nbr=length(zdata1(:,1)); nbus =
max(max(nl), max(nr)); nbr0=length(zdata0(:,1)); R0 =
zdata0(:,3); X0 = zdata0(:,4);
R1 = zdata1(:,3); X1 = zdata1(:,4);
R2 = zdata2(:,3); X2 = zdata2(:,4);

for k = 1:nbr0     if R0(k) == inf | X0(k)
== inf     R0(k) = 99999999; X0(k) =
999999999;     else, end end
ZB1 = R1 + j*X1;  ZB0 = R0 + j*X0;
ZB2 = R2 + j*X2;

if exist('V') == 1     if length(V) == nbus
V0 = V;     else, end else,  V0 = ones(nbus,
1) + j*zeros(nbus, 1); end

fprintf('\nDouble line-to-ground fault analysis \n')
ff = 999; while ff > 0 nf = input('Enter Faulted
Bus No. -> '); rtn=isempty(nf);      if rtn==1; nf=-
1; end
       while nf <= 0 | nf > nbus      fprintf('Faulted bus No.
must be between 1 & %g \n', nbus)      nf = input('Enter
Faulted Bus No. -> ');      rtn=isempty(nf);      if rtn==1; nf=-
1; end      end rtz=1;      while rtz==1
   fprintf('\nEnter Fault Impedance Zf = R + j*X in ')      Zf =
input('complex form (for bolted fault enter 0). Zf = ');
rtz=isempty(Zf);      end fprintf('  \n') fprintf('Double line-to-
ground fault at bus No. %g\n', nf) a
=cos(2*pi/3)+j*sin(2*pi/3); sctm = [1   1   1; 1 a^2  a; 1 a  a^2];

Z11 = Zbus2(nf, nf)*(Zbus0(nf, nf)+ 3*Zf)/(Zbus2(nf, nf)+Zbus0(nf, nf)+3*Zf);
Ia1 = V0(nf)/(Zbus1(nf,nf)+Z11);
```

```
Ia2 =-(V0(nf) - Zbus1(nf, nf)*Ia1)/Zbus2(nf,nf);
Ia0 =-(V0(nf) - Zbus1(nf, nf)*Ia1)/(Zbus0(nf,nf)+3*Zf);
I012=[Ia0; Ia1; Ia2];
Ifabc = sctm*I012; Ifabcm=abs(Ifabc);
Ift = Ifabc(2)+Ifabc(3);
Iftm = abs(Ift);

fprintf('Total fault current = %9.4f per unit\n\n', Iftm)
fprintf('Bus Voltages during the fault in per unit \n\n')
fprintf('     Bus    -------Voltage Magnitude-------  \n') fprintf('
No.    Phase a     Phase b     Phase c  \n')

for n = 1:nbus
Vf0(n)= 0 - Zbus0(n, nf)*Ia0;
Vf1(n)= V0(n) - Zbus1(n, nf)*Ia1;
Vf2(n)= 0 - Zbus2(n, nf)*Ia2;
Vabc = sctm*[Vf0(n); Vf1(n); Vf2(n)]; Va(n)=Vabc(1);
Vb(n)=Vabc(2); Vc(n)=Vabc(3); fprintf(' %5g',n) fprintf('
%11.4f', abs(Va(n))),fprintf(' %11.4f', abs(Vb(n))) fprintf('
%11.4f\n', abs(Vc(n))) end
fprintf('  \n') fprintf('Line currents for fault at bus No.  %g\n\n',
nf) fprintf('     From       To       -----Line Current Magnitude----
\n') fprintf('     Bus       Bus     Phase a     Phase b     Phase c  \n')
for n= 1:nbus    for I = 1:nbr       if nl(I) == n | nr(I) == n          if
nl(I) ==n       k = nr(I);          elseif nr(I) == n  k = nl(I);          end
if k ~= 0
          Ink1(n, k) = (Vf1(n) - Vf1(k))/ZB1(I);
Ink2(n, k) =  (Vf2(n) - Vf2(k))/ZB2(I);
else, end       else, end    end    for I = 1:nbr0
if nl0(I) == n | nr0(I) == n          if nl0(I) ==n
k = nr0(I);          elseif nr0(I) == n  k = nl0(I);
end
          if k ~= 0
          Ink0(n, k) =  (Vf0(n) - Vf0(k))/ZB0(I);
else, end       else, end    end    for I = 1:nbr
```

```
if nl(I) == n | nr(I) == n          if nl(I) ==n
k = nr(I);          elseif nr(I) == n  k = nl(I);
end             if k ~= 0
            Inkabc = sctm*[Ink0(n, k); Ink1(n, k); Ink2(n, k)];            Inkabcm
= abs(Inkabc); th=angle(Inkabc);                if real(Inkabc(2)) < 0
fprintf('%7g', n), fprintf('%10g', k),                fprintf(' %11.4f',
abs(Inkabc(1))),fprintf(' %11.4f', abs(Inkabc(2)))                fprintf('
%11.4f\n', abs(Inkabc(3)))                elseif real(Inkabc(2)) ==0 &
imag(Inkabc(2)) > 0                fprintf('%7g', n), fprintf('%10g', k),
fprintf(' %11.4f', abs(Inkabc(1))),fprintf(' %11.4f', abs(Inkabc(2)))
fprintf(' %11.4f\n', abs(Inkabc(3)))                else, end             else, end
else, end    end    if n==nf    fprintf('%7g',n), fprintf('         F'),    fprintf('
%11.4f', Ifabcm(1)),fprintf(' %11.4f', Ifabcm(2))    fprintf(' %11.4f\n',
Ifabcm(3))    else, end end
resp=0;    while strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp, 'y')~=1 & strcmp(resp,
'Y')~=1    resp = input('Another fault location? Enter "y" or "n" within single quote -> ');    if
strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp, 'y')~=1 & strcmp(resp, 'Y')~=1
fprintf('\n Incorrect reply, try again \n\n'), end    end    if resp == 'y' | resp == 'Y'    nf = 999;    else
ff = 0; end end   % end for while
```

Line to Ground Fault

PROGRAM 6

```
function lgfault(zdata0, Zbus0, zdata1, Zbus1, zdata2, Zbus2, V)
if exist('zdata2') ~= 1 zdata2=zdata1; else, end
if exist('Zbus2') ~= 1
Zbus2=Zbus1; else, end nl =
zdata1(:,1); nr = zdata1(:,2); nl0 =
zdata0(:,1); nr0 = zdata0(:,2);
nbr=length(zdata1(:,1)); nbus = max(max(nl), max(nr));
nbr0=length(zdata0(:,1));
R0 = zdata0(:,3); X0 = zdata0(:,4);
R1 = zdata1(:,3); X1 = zdata1(:,4);
R2 = zdata1(:,3); X2 = zdata1(:,4);

for k=1:nbr0
   if R0(k)==inf | X0(k) ==inf
```

```
   R0(k) = 99999999;  X0(k) = 99999999;
else, end end
ZB1 = R1 + j*X1;  ZB0 = R0 + j*X0;
ZB2 = R2 + j*X2;

if exist('V') == 1
   if length(V) == nbus
V0 = V;     else, end
else,  V0 = ones(nbus, 1) + j*zeros(nbus, 1); end
fprintf('\nLine-to-ground fault analysis \n')
ff = 999; while ff > 0
nf = input('Enter Faulted Bus No. -> ');
rtn=isempty(nf);      if rtn==1; nf=-1;
end         while nf <= 0 | nf > nbus
     fprintf('Faulted bus No. must be between 1 & %g \n', nbus)
nf = input('Enter Faulted Bus No. -> ');      rtn=isempty(nf);
if rtn==1; nf=-1; end
     end rtz=1;
while rtz==1
   fprintf('\nEnter Fault Impedance Zf = R + j*X in ')      Zf =
input('complex form (for bolted fault enter 0). Zf = ');
rtz=isempty(Zf);
   end fprintf('
\n')
fprintf('Single line to-ground fault at bus No. %g\n', nf)
a =cos(2*pi/3)+j*sin(2*pi/3); sctm = [1   1   1; 1 a^2
a; 1 a  a^2];
Ia0 = V0(nf)/(Zbus1(nf,nf)+Zbus2(nf, nf)+ Zbus0(nf, nf)+3*Zf); Ia1=Ia0; Ia2=Ia0;
I012=[Ia0; Ia1; Ia2];
Ifabc = sctm*I012; Ifabcm = abs(Ifabc); fprintf('Total fault
current = %9.4f per unit\n\n', Ifabcm(1)) fprintf('Bus
Voltages during the fault in per unit \n\n') fprintf('     Bus
-------Voltage Magnitude-------  \n') fprintf('     No.    Phase
a     Phase b     Phase c  \n')

for n = 1:nbus
Vf0(n)= 0 - Zbus0(n, nf)*Ia0;
Vf1(n)= V0(n) - Zbus1(n, nf)*Ia1;
Vf2(n)= 0 - Zbus2(n, nf)*Ia2;
Vabc = sctm*[Vf0(n); Vf1(n); Vf2(n)];
Va(n)=Vabc(1); Vb(n)=Vabc(2); Vc(n)=Vabc(3); fprintf('
%5g',n)
fprintf(' %11.4f', abs(Va(n))),fprintf(' %11.4f', abs(Vb(n)))
fprintf(' %11.4f\n', abs(Vc(n))) end
fprintf('  \n')
fprintf('Line currents for fault at bus No.  %g\n\n', nf) fprintf('
From      To      -----Line Current Magnitude----  \n')
```

```
fprintf('     Bus       Bus     Phase a     Phase b     Phase c  \n')
for n= 1:nbus     for I = 1:nbr        if nl(I) == n | nr(I) == n
if nl(I) ==n       k = nr(I);          elseif nr(I) == n  k = nl(I);
        end
          if k ~= 0
          Ink1(n, k) = (Vf1(n) - Vf1(k))/ZB1(I);
Ink2(n, k) =  (Vf2(n) - Vf2(k))/ZB2(I);
else, end       else, end    end    for I = 1:nbr0
if nl0(I) == n | nr0(I) == n          if nl0(I) ==n
k = nr0(I);          elseif nr0(I) == n  k = nl0(I);
        end
          if k ~= 0
          Ink0(n, k) =  (Vf0(n) - Vf0(k))/ZB0(I);
else, end       else, end    end    for I = 1:nbr
if nl(I) == n | nr(I) == n          if nl(I) ==n
k = nr(I);          elseif nr(I) == n  k = nl(I);
        end
if k ~= 0
          Inkabc = sctm*[Ink0(n, k); Ink1(n, k); Ink2(n, k)];
Inkabcm = abs(Inkabc); th=angle(Inkabc);                if
real(Inkabc(1)) > 0                fprintf('%7g', n),
fprintf('%10g', k),
             fprintf(' %11.4f', abs(Inkabc(1))),fprintf(' %11.4f', abs(Inkabc(2)))
fprintf(' %11.4f\n', abs(Inkabc(3)))                elseif real(Inkabc(1)) ==0 &
imag(Inkabc(1)) < 0
             fprintf('%7g', n), fprintf('%10g', k),
             fprintf(' %11.4f', abs(Inkabc(1))),fprintf(' %11.4f', abs(Inkabc(2)))
fprintf(' %11.4f\n', abs(Inkabc(3)))
             else, end
else, end       else,
end    end
  if n==nf
  fprintf('%7g',n), fprintf('         F'),
  fprintf(' %11.4f', Ifabcm(1)),fprintf(' %11.4f', Ifabcm(2))
fprintf(' %11.4f\n', Ifabcm(3))
  else, end
end resp=0;
  while strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp, 'y')~=1 & strcmp(resp, 'Y')~=1
resp = input('Another fault location? Enter "y" or "n" within single quote -> ');
  if strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp, 'y')~=1 & strcmp(resp, 'Y')~=1
fprintf('\n Incorrect reply, try again \n\n'), end    end
  if resp == 'y' | resp == 'Y'
nf = 999;    else ff = 0; end
end   % end for while
%Ink0
%Ink1
%Ink2
```

<u>Line To Line Fault</u>

PROGRAM 7

```
function llfault(zdata1, Zbus1, zdata2, Zbus2, V)
if exist('zdata2') ~= 1 zdata2=zdata1; else, end
if exist('Zbus2') ~= 1
Zbus2=Zbus1; else,
end

nl = zdata1(:,1); nr = zdata1(:,2); R1
= zdata1(:,3); X1 = zdata1(:,4);
R2 = zdata2(:,3); X2 = zdata2(:,4); ZB1
= R1 + j*X1;  ZB2 = R2 + j*X2;
nbr=length(zdata1(:,1)); nbus = max(max(nl), max(nr));
if exist('V') == 1     if length(V) == nbus     V0 = V;
else, end
else,  V0 = ones(nbus, 1) + j*zeros(nbus, 1); end
fprintf('\nLine-to-line fault analysis \n')
ff = 999; while ff > 0
nf = input('Enter Faulted Bus No. -> ');
rtn=isempty(nf);      if rtn==1; nf=-1;
end         while nf <= 0 | nf > nbus
    fprintf('Faulted bus No. must be between 1 & %g \n', nbus)
nf = input('Enter Faulted Bus No. -> ');      rtn=isempty(nf);
if rtn==1; nf=-1; end
    end rtz=1;      while rtz==1    fprintf('\nEnter Fault
Impedance Zf = R + j*X in ')      Zf = input('complex form
(for bolted fault enter 0). Zf = ');    rtz=isempty(Zf);    end
fprintf('  \n')
fprintf('Line-to-line fault at bus No. %g\n', nf)
a =cos(2*pi/3)+j*sin(2*pi/3); sctm
= [1   1   1; 1 a^2  a; 1 a  a^2];
Ia0=0;
Ia1 = V0(nf)/(Zbus1(nf,nf)+Zbus2(nf, nf)+Zf); Ia2=-Ia1;
I012=[Ia0; Ia1; Ia2];
Ifabc = sctm*I012; Ifabcm = abs(Ifabc); fprintf('Total fault
current = %9.4f per unit\n\n', Ifabcm(2)) fprintf('Bus
Voltages during the fault in per unit \n\n') fprintf('     Bus
-------Voltage Magnitude-------  \n')
fprintf('     No.    Phase a     Phase b     Phase c  \n')

for n = 1:nbus
Vf0(n)= 0;
Vf1(n)= V0(n) - Zbus1(n, nf)*Ia1;
Vf2(n)= 0 - Zbus2(n, nf)*Ia2;
Vabc = sctm*[Vf0(n); Vf1(n); Vf2(n)]; Va(n)=Vabc(1);
Vb(n)=Vabc(2); Vc(n)=Vabc(3); fprintf(' %5g',n)
fprintf(' %11.4f', abs(Va(n))),fprintf(' %11.4f', abs(Vb(n))) fprintf('
%11.4f\n', abs(Vc(n)))
```

```
end
fprintf('  \n')
fprintf('Line currents for fault at bus No.  %g\n\n', nf) fprintf('
From      To      -----Line Current Magnitude----  \n') fprintf('
Bus       Bus     Phase a     Phase b     Phase c  \n')

for n= 1:nbus
for I = 1:nbr
if nl(I) == n |
nr(I) == n
if nl(I) ==n       k
= nr(I);
elseif nr(I) == n
k = nl(I);
        end             if k
~= 0            Ink0(n,
k) = 0;
           Ink1(n, k) = (Vf1(n) - Vf1(k))/ZB1(I);
           Ink2(n, k) =  (Vf2(n) - Vf2(k))/ZB2(I);

           Inkabc = sctm*[Ink0(n, k); Ink1(n, k); Ink2(n, k)];
Inkabcm = abs(Inkabc); th=angle(Inkabc);                  if
real(Inkabc(2)) < 0

              fprintf('%7g', n), fprintf('%10g', k),
              fprintf(' %11.4f', abs(Inkabc(1))),fprintf(' %11.4f', abs(Inkabc(2)))
fprintf(' %11.4f\n', abs(Inkabc(3)))
              elseif real(Inkabc(2)) ==0 & imag(Inkabc(2)) > 0
              fprintf('%7g', n), fprintf('%10g', k),
              fprintf(' %11.4f', abs(Inkabc(1))),fprintf(' %11.4f', abs(Inkabc(2)))
fprintf(' %11.4f\n', abs(Inkabc(3)))                 else, end              else, end
else, end    end    if n==nf
   fprintf('%7g',n), fprintf('         F'),
   fprintf(' %11.4f', Ifabcm(1)),fprintf(' %11.4f', Ifabcm(2))
fprintf(' %11.4f\n', Ifabcm(3))
   else, end
end resp=0;
   while strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp, 'y')~=1 & strcmp(resp, 'Y')~=1
resp = input('Another fault location? Enter "y" or "n" within single quote -> ');
   if strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp, 'y')~=1 & strcmp(resp, 'Y')~=1
fprintf('\n Incorrect reply, try again \n\n'), end    end
   if resp == 'y' | resp == 'Y'
nf = 999;    else ff = 0; end
end   % end for while
```

Three Phase Symmetrical Fault PROGRAM 8

```
function symfaul(zdata, Zbus, V)

nl = zdata(:,1); nr = zdata(:,2); R = zdata(:,3);
X = zdata(:,4); nc = length(zdata(1,:));   if nc
> 4   BC = zdata(:,5);
  elseif nc ==4, BC = zeros(length(zdata(:,1)), 1);
end
ZB = R + j*X;
nbr=length(zdata(:,1)); nbus = max(max(nl), max(nr));
if exist('V') == 1     if
length(V) == nbus
V0 = V;     else, end
else,  V0 = ones(nbus, 1) + j*zeros(nbus, 1); end
fprintf('\Three-phase balanced fault analysis \n')
ff = 999; while ff > 0
nf = input('Enter Faulted Bus No. -> ');
        rtn=isempty(nf);      if
rtn==1; nf=-1; end
while nf <= 0 | nf > nbus
     fprintf('Faulted bus No. must be between 1 & %g \n', nbus)
nf = input('Enter Faulted Bus No. -> ');      rtn=isempty(nf);
if rtn==1; nf=-1; end
     end rtz=1;      while rtz==1    fprintf('\nEnter Fault
Impedance Zf = R + j*X in ')      Zf = input('complex form
(for bolted fault enter 0). Zf = ');    rtz=isempty(Zf);    end
fprintf('  \n')
fprintf('Balanced three-phase fault at bus No. %g\n', nf)

If = V0(nf)/(Zf + Zbus(nf, nf)); Ifm = abs(If);
Ifmang=angle(If)*180/pi; fprintf('Total fault current
= %8.4f per unit \n\n', Ifm)
%fprintf(' p.u. \n\n', Ifm)
fprintf('Bus Voltages during fault in per unit \n\n')
fprintf('     Bus     Voltage       Angle\n')
fprintf('     No.     Magnitude     degrees\n')

for n = 1:nbus
if n==nf
    Vf(nf) = V0(nf)*Zf/(Zf + Zbus(nf,nf)); Vfm = abs(Vf(nf)); angv=angle(Vf(nf))*180/pi;
else, Vf(n) = V0(n) - V0(n)*Zbus(n,nf)/(Zf + Zbus(nf,nf));
    Vfm = abs(Vf(n)); angv=angle(Vf(n))*180/pi;
end
    fprintf('   %4g',  n), fprintf('%13.4f', Vfm),fprintf('%13.4f\n', angv)

end
```

```
fprintf('  \n')

fprintf('Line currents for fault at bus No.  %g\n\n', nf)
fprintf('     From      To     Current     Angle\n') fprintf('
Bus       Bus    Magnitude   degrees\n')

for n= 1:nbus    %Ign=0;     for
I = 1:nbr       if nl(I) == n | nr(I)
== n          if nl(I) ==n       k =
nr(I);          elseif nr(I) == n  k
= nl(I);
        end
if k==0
           Ink = (V0(n) - Vf(n))/ZB(I);
           Inkm = abs(Ink); th=angle(Ink);
              %if th <= 0
if real(Ink) > 0
              fprintf('      G   '), fprintf('%7g',n), fprintf('%12.4f', Inkm)
fprintf('%12.4f\n', th*180/pi)                    elseif real(Ink) ==0 &
imag(Ink) < 0
              fprintf('      G   '), fprintf('%7g',n), fprintf('%12.4f', Inkm)
fprintf('%12.4f\n', th*180/pi)
             else, end
Ign=Ink;
elseif k ~= 0
           Ink = (Vf(n) - Vf(k))/ZB(I)+BC(I)*Vf(n);
           %Ink = (Vf(n) - Vf(k))/ZB(I);
           Inkm = abs(Ink); th=angle(Ink);
           %Ign=Ign+Ink;
%if th <= 0             if
real(Ink) > 0
                fprintf('%7g', n), fprintf('%10g', k),
                fprintf('%12.4f', Inkm), fprintf('%12.4f\n', th*180/pi)
elseif real(Ink) ==0 & imag(Ink) < 0                  fprintf('%7g',
n), fprintf('%10g', k),
                fprintf('%12.4f', Inkm), fprintf('%12.4f\n', th*180/pi)
else, end              else, end       else, end    end

 if n==nf
 fprintf('%7g',n), fprintf('         F'),  fprintf('%12.4f', Ifm)
 fprintf('%12.4f\n', Ifmang)
else, end end    resp=0;
   while strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp, 'y')~=1 & strcmp(resp, 'Y')~=1
resp = input('Another fault location? Enter "y" or "n" within single quote -> ');
   if strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp, 'y')~=1 & strcmp(resp, 'Y')~=1
fprintf('\n Incorrect reply, try again \n\n'), end    end
   if resp == 'y' | resp == 'Y'
nf = 999;    else ff = 0; end
end   % end for while
```

PROGRAM 9

```
%NEWTON RAPHSON METHOD
clc; gbus = [1 2.0 1.0
0.0 0.0
2 0.0 0.0 0.5 1.0
3 1.5 0.6 0.0 0.0]; ybus = [5.882-j*23.528 -2.941+j*11.764 -
   2.941+j*11.764 -2.941+j*11.764 5.882-j*23.528 -2.941+j*11.764 -
   2.941+j*11.764 -2.941+j*11.764 5.882-j*23.528]; t= 0.001
   v1=1.04+j*0; v2=1+j*0; v3=1.04+j*0; del3=angle(v3); del1=angle(v1);
   del2=angle(v2); %abs(ybus(2,1)) %abs(v2) for i=1:10
   p2=(abs(v2)*abs(v1)*abs(ybus(2,1))*cos((angle(ybus(2,1)))+del1-
   del2))
+abs(v2)*
abs(v2)*abs(ybus(2,2))*cos((angle(ybus(2,2))))+(abs(v2)*abs(v3)*
abs(ybus(2,3))*cos((angle(ybus(2,3))+del3-del2));
q2=-(abs(v2)*abs(v1)*abs(ybus(2,1))*sin((angle(ybus(2,1)))+del1-del2))-
abs(v2)*abs(v2)*abs(ybus(2,2))*sin((angle(ybus(2,2))))-(abs(v2)*abs(v3)*
abs(ybus(2,3))*sin((angle(ybus(2,3)))+del3-del2));
```

```
p3=(abs(v3)*abs(v1)*abs(ybus(3,1))*cos((angle(ybus(3,1)))+del1-
del3))+abs(v3)*abs(v3)*abs(ybus(3,3))*cos((angle(ybus(3,3))))+
(abs(v2)*abs(v3)* abs(ybus(3,2))*cos((angle(ybus(3,2)))+del2-del3));
delp20=gbus(2,4)-gbus(2,2)-p2; delp30=gbus(3,4)-gbus(3,2)-p3;
delq20=gbus(2,5)-gbus(2,3)-q2;
J(1,1)=(abs(v2)*abs(v1)*abs(ybus(2,1))*sin((angle(ybus(2,1)))+del1-
del2))+(abs(v2)*abs(v3)*abs(ybus(2,3))*sin((angle(ybus(2,3)))+del3-
del2));
J(1,2)=-(abs(v2)*abs(v3)*abs(ybus(2,3))*sin((angle(ybus(2,3)))+del3-
del2));
J(1,3)=(abs(v1)*abs(ybus(2,1))*cos((angle(ybus(2,1)))+del1-
del2))+2*(abs(v2)*abs(ybus(2,2))*cos((angle(ybus(2,2))))+
(abs(v3)*abs(ybus(2,3))* cos((angle(ybus(2,3)))+del3-del2));
J(2,1)=-(abs(v3)*abs(v2)*abs(ybus(3,2))*sin((angle(ybus(3,2)))+del2-
del3));
J(2,2)=(abs(v3)*abs(v1)*abs(ybus(3,1))*sin((angdel3))+(abs(v3)*abs(v2)*abs(ybus(3,2))*sin((angle(yb
us(3,2)))+del2- del3));
J(2,3)=(abs(v3)*abs(ybus(3,2))*cos((angle(ybus(3,2)))+del2-del3));
J(3,1)=(abs(v2)*abs(v1)*abs(ybus(2,1))*cos((angle(ybus(2,1)))+del1-
del2))-(abs(v2)*abs(v3)*abs(ybus(2,3))*cos((angle(ybus(2,3)))+del2-
del3));
J(3,2)=(abs(v2)*abs(v3)*abs(ybus(2,3))*cos((angle(ybus(2,3)))+del2-
del3));
J(3,3)=-(abs(v2)*abs(ybus(2,1))*sin((angle(ybus(2,1)))+del1- del2))-
2*(abs(v2)*abs(ybus(2,2))*sin((angle(ybus(2,2))))-
(abs(v3)*abs(ybus(2,3))*
sin((angle(ybus(2,3)))+del3-del2)); end
J inv(J);
A=[del2;del3;abs(v2)]; delA0=[delp20;delp30;delq20];
delA1=inv(J)*delA0; delA1; b0=abs(v2);
A1=[del2;del3;b0]+delA1; A1-delA0; if((A1-
delA0)<=t) break; del2=A1(1,1); del3=A1(2,1);
abs(v2)=A1(3,1); end A1
```

PROGRAM 10

```
%Gauss Sedial
clc; data=[1 1 2 10-j*20 2 1 3 10-j*30 3
2 3 16-j*32] elements=max(data(:,1));
bus=max(max(data(:,2)),max(data(:,3)))
; y=zeros(bus,bus); for p=1:bus, for
q=1:elements,
if(data(q,2)==p|data(q,3)==p)
y(p,p)=y(p,p)+data(q,4); end end end for
p=1:bus,
for q=1:bus, if (p~=q) for r=1:elements
if((data(r,2)==p&data(r,3)==q)|(data(r,2)==q&data(r,3)==p))
y(p,q)=-(data(r,4)); end end end end end a1=input('enter p2
in MW:'); b1=input('enter q2 in MVAR:'); a2=input('enter p3
in MW:'); b2=input('enter q3 in MVAR'); pu=input('enter the
base value in MVA'); p2=(a1/pu); q2=(b1/pu); p3=(a2/pu);
q3=(b2/pu); dx1=1+j*0; dx2=1+j*0; v1=1.05; v2=1+j*0;
v3=1+j*0; iter=0; disp('iter v2 v3');
while(abs(dx1)&abs(dx2)>=0.00001)&iter<7; iter=iter+1;
g1=(((p2-j*q2)/conj(v2))+(-y(1,2)*v1)+(-y(2,3)*v3))/y(2,2);
g2=(((p3-j*q3)/conj(v3))+(-y(1,3)*v1)+(-y(2,3)*g1))/y(3,3);
dx1=g1-v2;
dx2=g2-v3; v2=v2+dx1;
v3=v3+dx2;
fprintf ('%g',iter),disp([v2,v3]); end
```

PROGRAM:-11

Economic load dispatch

```
cost = [500    5.3     0.004
400   5.5    0.006        200
5.8    0.009];  limits=[200
450
        150  350
```

100 225];
Pdt = 800;
dispatch gencost