References

[1] Statistical Digest, Ceylon Electricity Board, 2013

[2] Annual report, Ceylon Electricity Board, 2012

[3] Sri Lanka Energy Sector Development Plan for a Knowledge-based Economy 2015-2025. [Online], Available:http://powermin.gov.lk/sinhala/wp-content/uploads/2015/03/energy_empowered_nation_2015_2025.pdf

[4] Distribution code of Sri Lanka, Public Utilities Commission of Sri Lanka, 2012

[5] W. H. Kersting, Distribution System Modeling and Analysis, (CRC Press, 2012)

[6] M.A. Paul, Analysis of Faulted Power systems, New York:IEEE Press Power Systems Engineering Series,1995,pp.71-83 - ISBN 0-7803-1145-0

[7] J.R. Carson, “Wave propagation in overhead wires with ground return”, Bell System Technical Journal, Vol. 5, New York, 1926.

[8] Thomas Allen Short-Electric power distribution handbook-CRC Press(2004) - ISBN 0-8493-1791-6

[9] Horton, R., W. G. Sunderman, R. F. Arritt, and R. C. Dugan. "Effect of line modeling methods on neutral-to-earth voltage analysis of multi-grounded distribution feeders." In Power Systems Conference and Exposition (PSCE), 2011 IEEE/PES, pp. 1-6. IEEE, 2011.

[10] Kersting, W. H. "A three-phase unbalanced line model with grounded neutrals through a resistance." In Power and Energy Society General Meeting-Conversion and Delivery of Electrical Energy in the 21st Century, 2008 IEEE, pp. 1-2. IEEE, 2008.

[11] Urquhart, A.J. and Thomson, M., 2013. Assumptions and approximations typically applied in modelling LV networks with high penetrations of low carbon technologies. Solar Integration Workshop 2013, London, October 21st-22nd 2013, 6pp

[12] A.J. Urquhart," Accuracy of Low Voltage Electricity Distribution Network Modeling", Doctoral dissertation, Loughborough University, 2016

[13] Ciric, R. M., Feltrin, A. P., & Ochoa, L. F. (2003). Power flow in four-wire distribution networks-general approach. IEEE Transactions on Power Systems, 18(4), 1283-1290.

[14] Guang-Xiang Luo and Adam Semlyen.Efficient load flow for large weakly meshed networks.Power Sytems,IEEE Transactions on 5(4): 1309-1316,1990

[15] Carol S.Cheng and D.Shirmohammadi,"A three phase power flow method for real time distribution system analysis,"IEEE Transactions on Power System,Vol. 10,No. 2,pp.671-679,May 1995.

[16] Kersting, W. H., & Green, R. K. The application of Carson's equation to the steady-state analysis of distribution feeders. In Power Systems Conference and Exposition (PSCE), 2011 IEEE/PES (pp. 1-6). IEEE.,2011

[17] Afolabi, Olukayode A., Warsame H. Ali, Penrose Cofie, John Fuller, Pamela Obiomon, and Emmanuel S. Kolawole. "Analysis of the Load Flow Problem in Power System Planning Studies." Energy and Power Engineering 7, no. 10, 509, 2015.

[18] J.A.Michline Rupa,S.Ganesh "Power Flow Analysis for Radial Distribution System using Backward/Forward Sweep Method" World Academy of Science, Engineering and Technology International Journal of Electrical ,Computer ,Energetic, Electronic and Communication Engineering Vol:8,No:10,2014

[19] Kersting, W. H. "The computation of neutral and dirt currents and power losses." In Power Systems Conference and Exposition, 2004. IEEE PES, pp. 213-218. IEEE, 2004.

Appendix A - Arial Bundle Conductor Data and Impedance Calculations

Cable data

Table A.01 shows the cable data of 50 ABC**.**

Conductor	Formation	A(mm2)	R(Ω/km)	D1 (mm)	D2(mm)	t (mm)
Phase	Hard drawn Aluminum	50	0.61	8.2	11.5	1.65
Neutral	Aluminum Alloy	35	0.98	7.2	11	1.9

Table A.01: Cable data

A-Nominal cross section area

R-DC conductor resistance measured at 30C

D1-Nominal diameter Without Insulation

D2- Nominal diameter With Insulation

t-Insulating sheath thickness

Insulation Material- XLPE - Cross Link Poly Ethylene

Span Length between Nodes

Table A.02 shows the span length details of considered feeder**..**

	Node 1-2	Node 2-3	Node 3-4	Node 4-5	Node 5-6	Node6-7	Node7-8
Length(m)	29	32	20	57	38	45	34

Table A.02:- Span length

Calculation of Primitive impedance matrix for considered feeder

Primitive impedance matrix can be calculated using following equations.

$$\hat{z}_{ii} = r_i + \frac{\mu_0 \omega}{8} + j\omega \frac{\mu_0}{2\pi} * ln\left(\frac{658.9}{D_{ii}\sqrt{f/\rho}}\right)$$

$$\hat{z}_{ij} = \frac{\mu_0 \omega}{8} + j\omega \frac{\mu_0}{2\pi} * ln\left(\frac{658.9}{D_{ij}\sqrt{f/\rho}}\right)$$

Self-impedance calculation for Phase conductors (Z_{aa}) was done using Matlab 7.6 (R2008a) including ground effect. Following is the code used for calculation.

```
format short g
Ra=0.61;
f=50;
p=100;
R_Ground=9.869*10^-4*50*10^-3;
% constant k, assuming permiability of free space in H/m %
u=4*pi*10^-7;
k = 2*10^-7;
w=2*pi*f;
ZRa=(Ra/1000)+R_Ground;
Daa=0.758*4.1*10^-3;
ZXa=k*w*log((658.9)/(Daa*sqrt(f/p)));
Zaa=complex(ZRa,ZXa);
```

Z_{bb} and Z_{cc} can be calculated similar manner.

Self-impedance calculation for Neutral conductors (Z_{nn}) was done using Matlab 7.6 (R2008a) including ground effect. Following is the code used for calculation.

```
format short g
Rn=0.98;
f=50;
p=100;
R_Ground=9.869*10^-4*50*10^-3;
```

```
% constant k, assuming permiability of free space in H/m
k = 2*10^-7   ;
w=2*pi*f;
ZRn=(Rn/1000)+R_Ground;
Dnn=0.758*3.6*10^-3;
ZXn=k*w*log((658.9)/(Dnn*sqrt(f/p)));
Znn=complex(ZRn,ZXn);
```

Mutual-impedance calculation for Phase conductors (Zab) was done using Matlab 7.6 (R2008a) including ground effect. Following is the code used for calculation.

```
Dab=0.019485572
f=50;
p=100;
R_Ground=9.869*10^-4*50*10^-3;
% constant k, assuming permiability of free space in H/m
k = 2*10^-7   ;
w=2*pi*f;
ZRab =R_Ground;
ZXab =k*w*log((658.9)/(Dab*sqrt(f/p)));
Zab=complex(ZRab,ZXab);
```

Other phase to phase mutual impedance components can be calculated similar manner.

Mutual-impedance calculation for Phase to neutral (Z_{an}) was done using Matlab 7.6 (R2008a) including ground effect. Following is the code used for calculation.

```
GMDn=0.01125;
f=50;
p=100;
R_Ground=9.869*10^-4*50*10^-3;
% constant k, assuming permiability of free space in H/m
k = 2*10^-7   ;
w=2*pi*f;
ZRpn=R_Ground;
%Dij=((11.5+11)/2)*10^-3;
ZXpn=k*w*log((658.9)/(GMDn*sqrt(f/p)));
Zan=complex(ZRpn,ZXpn);
```

Other phase to neutral mutual impedance components can be calculated similar manner. Following is the calculated impedance matrix (Ω/m) for considered conductors.

$$\begin{bmatrix} 0.00065934 + 0.00079237i & 4.9345e-005 + 0.00067703i & 4.9345e-005 + 0.00067703i & 4.9345e-005 + 0.00071154i \\ 4.9345e-005 + 0.00067703i & 0.00065934 + 0.00079237i & 4.9345e-005 + 0.00067703i & 4.9345e-005 + 0.00071154i \\ 4.9345e-005 + 0.00067703i & 4.9345e-005 + 0.00067703i & 0.00065934 + 0.00079237i & 4.9345e-005 + 0.00071154i \\ 4.9345e-005 + 0.00071154i & 4.9345e-005 + 0.00071154i & 4.9345e-005 + 0.00071154i & 0.0010293 + 0.00080054i \end{bmatrix}$$

Calculation of self-reactance of phase "a" for considered feeder

Self-reactance calculation for Phase conductors (Z_{aa}) was done using Matlab 7.6 (R2008a). Following is the code used for calculation.

```
k = 2*10^-7;
w=2*pi*f;
Daa=0.758*4.1*10^-3;
Xaa=k*w*log(1/(Daa));        (Ω/m)
```

Calculation of total reactance calculation for term $\bar{z}_{aa} + \bar{z}_{gg} - \bar{z}_{ga} - \bar{z}_{ag}$

$$(\bar{z}_{aa} + \bar{z}_{gg} - \bar{z}_{ga} - \bar{z}_{ag}) = \omega \frac{\mu_0}{2\pi} * ln\left(\frac{658.9}{D_{aa}\sqrt{f/\rho}}\right)$$

Total-reactance calculation for above considered terms was done using Matlab 7.6 (R2008a). Following is the code used for calculation.

```
f=50;
p=100;
k = 2*10^-7;
w=2*pi*f;
Daa=0.758*4.1*10^-3;
Xa=k*w*log((658.9)/(Daa*sqrt(f/p)));
```

Appendix B - Load Flow Calculation

Power flow analysis was implemented based on the customized forward-backward iterative technique using Matlab 7.6 (R2008a). Following is the code used for implementation. Measured data are stored in the excel tables and retrieve during program execution.

- Apparent power for each node relevant to corresponding phase for measured time intervals are stored in the ‘Data input node power 2016.12.18.xls’.
- Voltage values for each node relevant to corresponding phase for measured time intervals are stored in the Actual voltage profile 2016.12.18.xls’.
- Measured current of starting span for corresponding phase for measured time intervals are stored in the ‘Actual currents span 2016.12.18.xls'.
- Current for each node relevant to corresponding phase for measured time intervals are stored in the ‘Actual node current 2016.12.18.xls.

All above data are included in the Compact Disc provided with dissertation.

Load Flow Program

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Input parameters for radial feeder without spurs-Number of poles and span
lengths
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Node connected measured apparent power extraction-extraction from excel
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for Load_sheet=2;
exl = actxserver('excel.application');
exlWkbk = exl.Workbooks;
exlFile = exlWkbk.Open([docroot '/techdoc/matlab_external/examples/Data
input  node power 2016.12.18.xls']);
exlSheet1 = exlFile.Sheets.Item(Load_sheet);
robj = exlSheet1.Columns.End(4);       % Find the end of the column
numrows = robj.row;                    % And determine what row it is
dat_range = ['A2:H' num2str(numrows)]; % Read to the last row
rngObj = exlSheet1.Range(dat_range);
exlData = rngObj.Value;
L_V=str2double(exlData);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Source & Node measured Voltage extraction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for Voltage_sheet=1;
exl = actxserver('excel.application');
exlWkbk = exl.Workbooks;
exlFile = exlWkbk.Open([docroot '/techdoc/matlab_external/examples/Actual
voltage profile  2016.12.18.xls']);
exlSheet1 = exlFile.Sheets.Item(Voltage_sheet);
robj = exlSheet1.Columns.End(4);       % Find the end of the column
numrows = robj.row;                    % And determine what row it is
dat_range = ['A2:R' num2str(numrows)]; % Read to the last row
rngObj = exlSheet1.Range(dat_range);
exlData = rngObj.Value;
V_V=cell2mat(exlData);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Measured span current  extraction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for span_current_sheet=1;
exl = actxserver('excel.application');
exlWkbk = exl.Workbooks;
exlFile = exlWkbk.Open([docroot '/techdoc/matlab_external/examples/Actual
currents span 2016.12.18.xls']);
exlSheet1 = exlFile.Sheets.Item(span_current_sheet);
robj = exlSheet1.Columns.End(4);       % Find the end of the column
numrows = robj.row;                    % And determine what row it is
dat_range = ['A2:D' num2str(numrows)]; % Read to the last row
rngObj = exlSheet1.Range(dat_range);
exlData = rngObj.Value;
SC_X=cell2mat(exlData);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Measured node current  extraction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
for node_current_sheet=1;
exl = actxserver('excel.application');
exlWkbk = exl.Workbooks;
exlFile = exlWkbk.Open([docroot '/techdoc/matlab_external/examples/Actual
node current 2016.12.18.xls']);
exlSheet1 = exlFile.Sheets.Item(node_current_sheet);
robj = exlSheet1.Columns.End(4);       % Find the end of the column
numrows = robj.row;                    % And determine what row it is
dat_range = ['A2:N' num2str(numrows)]; % Read to the last row
rngObj = exlSheet1.Range(dat_range);
exlData = rngObj.Value;
NC_X=cell2mat(exlData);

end

for Loop=1:96;

x=8;
Spanmatrix=[29 32 20 57 38 45 34];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Node connected measured apparent power extraction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Load_row_index=Loop;

Load_matrix(1,1)=0;
Load_matrix(2,1)=0;
Load_matrix(3,1)=0;

Load_matrix(1,2)=L_V(Load_row_index,2);
Load_matrix(2,2)=0;
Load_matrix(3,2)=0;

Load_matrix(1,3)=0;
Load_matrix(2,3)=L_V(Load_row_index,3);
Load_matrix(3,3)=0;

Load_matrix(1,4)=0;
Load_matrix(2,4)=L_V(Load_row_index,4);
Load_matrix(3,4)=0;

Load_matrix(1,5)=L_V(Load_row_index,5);
Load_matrix(2,5)=0;
Load_matrix(3,5)=0;

Load_matrix(1,6)=0;
Load_matrix(2,6)=0;
Load_matrix(3,6)=0;

Load_matrix(1,7)=L_V(Load_row_index,7);
Load_matrix(2,7)=0;
Load_matrix(3,7)=0;

Load_matrix(1,8)=0;
Load_matrix(2,8)=0;
Load_matrix(3,8)=0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Source voltage definition
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Voltage_row_index=Loop;
```

```
Node_voltage_matrix=zeros(3,8);

[B,Y] =(pol2cart(0,V_V(Voltage_row_index,1)));
Node_voltage_matrix(1,1)=complex(B,Y);

[K,L] =(pol2cart(-2*pi/3,V_V(Voltage_row_index,2)));
Node_voltage_matrix(2,1)=complex(K,L);

[M,N] =(pol2cart(2*pi/3,V_V(Voltage_row_index,3)));
Node_voltage_matrix(3,1)=complex(M,N);
Node_voltage_matrix(4,1)=complex(0,0);

%%%%%%%%Actual voltage extraction for one minute instant

Voltage_row_index=Loop;

Act_Node_voltage_matrix=zeros(3,8);

Act_Node_voltage_matrix(1,1)=V_V(Voltage_row_index,1);
Act_Node_voltage_matrix(2,1)=V_V(Voltage_row_index,2);
Act_Node_voltage_matrix(3,1)=V_V(Voltage_row_index,3);

Act_Node_voltage_matrix(1,2)=V_V(Voltage_row_index,4);
Act_Node_voltage_matrix(2,2)=0;
Act_Node_voltage_matrix(3,2)=0;

Act_Node_voltage_matrix(1,3)=0;
Act_Node_voltage_matrix(2,3)=V_V(Voltage_row_index,6);
Act_Node_voltage_matrix(3,3)=0;

Act_Node_voltage_matrix(1,4)=0;
Act_Node_voltage_matrix(2,4)=V_V(Voltage_row_index,8);
Act_Node_voltage_matrix(3,4)=0;

Act_Node_voltage_matrix(1,5)=V_V(Voltage_row_index,10);
Act_Node_voltage_matrix(2,5)=0;
Act_Node_voltage_matrix(3,5)=0;

Act_Node_voltage_matrix(1,6)=V_V(Voltage_row_index,12);
Act_Node_voltage_matrix(2,6)=V_V(Voltage_row_index,13);
Act_Node_voltage_matrix(3,6)=V_V(Voltage_row_index,14);

Act_Node_voltage_matrix(1,7)=V_V(Voltage_row_index,15);
Act_Node_voltage_matrix(2,7)=0;
Act_Node_voltage_matrix(3,7)=0;

Act_Node_voltage_matrix(1,8)=V_V(Voltage_row_index,16);
Act_Node_voltage_matrix(2,8)=V_V(Voltage_row_index,17);
Act_Node_voltage_matrix(3,8)=V_V(Voltage_row_index,18);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Measured span current  extraction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
sp_current_row_index=Loop;

Ac_span_current_matrix(1,1)=SC_X(sp_current_row_index,1);
Ac_span_current_matrix(2,1)=SC_X(sp_current_row_index,2);
Ac_span_current_matrix(3,1)=SC_X(sp_current_row_index,3);
Ac_span_current_matrix(4,1)=SC_X(sp_current_row_index,4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Measured node current  extraction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
node_current_row_index=Loop;

Act_Node_current_matrix=zeros(3,8);

Act_Node_current_matrix(1,1)=0;
Act_Node_current_matrix(2,1)=0;
Act_Node_current_matrix(3,1)=0;

Act_Node_current_matrix(1,2)=NC_X(node_current_row_index,3);
Act_Node_current_matrix(2,2)=0;
Act_Node_current_matrix(3,2)=0;

Act_Node_current_matrix(1,3)=0;
Act_Node_current_matrix(2,3)=NC_X(node_current_row_index,6);
Act_Node_current_matrix(3,3)=0;

Act_Node_current_matrix(1,4)=0;
Act_Node_current_matrix(2,4)=NC_X(node_current_row_index,9);
Act_Node_current_matrix(3,4)=0;

Act_Node_current_matrix(1,5)=NC_X(node_current_row_index,12);
Act_Node_current_matrix(2,5)=0;
Act_Node_current_matrix(3,5)=0;

Act_Node_current_matrix(1,6)=0;
Act_Node_current_matrix(2,6)=0;
Act_Node_current_matrix(3,6)=0;

Act_Node_current_matrix(1,7)=NC_X(node_current_row_index,14);
Act_Node_current_matrix(2,7)=0;
Act_Node_current_matrix(3,7)=0;

Act_Node_current_matrix(1,8)=0;
Act_Node_current_matrix(2,8)=0;
Act_Node_current_matrix(3,8)=0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Impedence
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Grounding resistance at trnasfomer neutral bushing and feeder end.
Z_Grounding=63;

%earth resistance per m ,Rd  = 9.869*10^-4*f*10^-3
Z_Ground=9.869*10^-4*50*10^-3;

% PER METER IMPEDENCE AT MIDDLE SPAN
%MImpedence=[0.000641+0.000087i 0 0 0;0 0.000641+0.000087i 0 0;0 0
0.000641+0.000087i 0;0 0 0 0.000641+0.000087i];
MImpedence=[0.00065934 + 0.00079237i 4.9345e-005 + 0.00067703i 4.9345e-005 +
0.00067703i  4.9345e-005 + 0.00071154i;4.9345e-005 + 0.00067703i
0.00065934 + 0.00079237i  4.9345e-005 + 0.00067703i  4.9345e-005 +
0.00071154i;4.9345e-005 + 0.00067703i 4.9345e-005 + 0.00067703i  0.00065934
+ 0.00079237i  4.9345e-005 + 0.00071154i;4.9345e-005 + 0.00071154i 4.9345e-
005 + 0.00071154i 4.9345e-005 + 0.00071154i 0.0010293 + 0.00080054i];

% PER METER IMPEDENCE AT END SPAN(GROUNDED NEUTRAL END)
%EImpedence=[0.000641+0.000087i 0 0 0;0 0.000641+0.000087i 0 0;0 0
0.000641+0.000087i 0;0 0 0 0.000641+0.000087i];
EImpedence=[0.00065934 + 0.00079237i 4.9345e-005 + 0.00067703i 4.9345e-005 +
0.00067703i  4.9345e-005 + 0.00071154i;4.9345e-005 + 0.00067703i
0.00065934 + 0.00079237i  4.9345e-005 + 0.00067703i  4.9345e-005 +
0.00071154i;4.9345e-005 + 0.00067703i 4.9345e-005 + 0.00067703i  0.00065934
```

```
+ 0.00079237i  4.9345e-005 + 0.00071154i;4.9345e-005 + 0.00071154i 4.9345e-
005 + 0.00071154i 4.9345e-005 + 0.00071154i 0.0010293 + 0.00080054i];
Espan=Spanmatrix(1,x-1)*EImpedence;

% PER METER IMPEDENCE AT START SPAN(GROUNDED NEUTRAL END)
%SImpedence=[0.000641+0.000087i 0 0 0;0 0.000641+0.000087i 0 0;0 0
0.000641+0.000087i 0;0 0 0 0.000641+0.000087i];
SImpedence=[0.00065934 + 0.00079237i 4.9345e-005 + 0.00067703i 4.9345e-005 +
0.00067703i  4.9345e-005 + 0.00071154i;4.9345e-005 + 0.00067703i
0.00065934 + 0.00079237i  4.9345e-005 + 0.00067703i  4.9345e-005 +
0.00071154i;4.9345e-005 + 0.00067703i 4.9345e-005 + 0.00067703i  0.00065934
+ 0.00079237i  4.9345e-005 + 0.00071154i;4.9345e-005 + 0.00071154i 4.9345e-
005 + 0.00071154i 4.9345e-005 + 0.00071154i 0.0010293 + 0.00080054i];
Sspan=Spanmatrix(1,1)*SImpedence;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Source voltage based intial Current calculation   for entire nodes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=1:x

% Phase current calculation

Node_current_a=zeros(1,1);
Node_current_b=zeros(1,1);
Node_current_c=zeros(1,1);
Node_current_a=conj(Load_matrix(1,i)/Node_voltage_matrix(1,1));
Node_current_b=conj(Load_matrix(2,i)/Node_voltage_matrix(2,1));
Node_current_c=conj(Load_matrix(3,i)/Node_voltage_matrix(3,1));
Node_current_unbalance=(Node_current_a+Node_current_b+Node_current_c);

Node_current_matrix(1,i)=Node_current_a;
Node_current_matrix(2,i)=Node_current_b;
Node_current_matrix(3,i)=Node_current_c;
Node_current_matrix(4,i)=Node_current_unbalance;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Node based  Impedence calculation for unbalance current division
calculation-Node to source through ground path/neutral conductor
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Node to source through ground path %%%%%%%%%%

if i==x
Z_back_i=zeros(1,1);
Z_back_i=0;
Z_back_i=(2*Z_Grounding) + Z_Ground;

elseif i==x-1

Z_back_i=zeros(1,1);
Z_back_i=0;
Z_back_i=Espan(4,4) + (2*Z_Grounding) + Z_Ground;

else

Z_back=zeros(1,1);
zT_back_i=zeros(1,1);
Z_back_i=zeros(1,1);
Z_back=0;
zT_back_i=0;
Z_back_i=0;
for j=i:x-2
zT_back_i=Spanmatrix(1,j)*MImpedence(4,4);
Z_back=Z_back+zT_back_i;
```

```
end

Z_back_i=Z_back+Espan(4,4) + (2*Z_Grounding) + Z_Ground;
end

%%%% Node to source through neutral conductor %%%%%%%%%%

if i==1
z_front_i=0;

elseif  i==x
z_front=zeros(1,1);
zT_front_i=zeros(1,1);
z_front_i=zeros(1,1);
z_front=0;
zT_front_i=0;
z_front_i=0;
for k=1:i-2
zT_front_i=Spanmatrix(1,k)*MImpedence(4,4);
z_front=z_front+zT_front_i;
end
z_front_i=z_front+Espan(4,4);

else

z_front=zeros(1,1);
zT_front_i=zeros(1,1);
z_front_i=zeros(1,1);
z_front=0;
zT_front_i=0;
z_front_i=0;
for k=1:i-1
zT_front_i=Spanmatrix(1,k)*MImpedence(4,4);
z_front=z_front+zT_front_i;
end
z_front_i=z_front;
end

Return_path_impedence_matrix(1,i)=Z_back_i;
Return_path_impedence_matrix(2,i)=z_front_i;

%%%% Impedence ratio calculation between two paths %%%%%%%%%%

Node_current_n=Node_current_unbalance*(Z_back_i/(z_front_i+Z_back_i));
Node_current_g=Node_current_unbalance*(z_front_i/(z_front_i+Z_back_i));
Z_back_i=0;
z_front_i=0;

Node_current_matrix(5,i)=Node_current_n;
Node_current_matrix(6,i)=Node_current_g;

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% span current matrix  -for Ground and Neutral seperatly considering
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=1:x
spani_jNCR_l=zeros(1,1);
spani_jGCR_l=zeros(1,1);
span_current_matrix(5,1)=0;   %%no ground current flow between first
node(source)  ,only neutral current
span_current_matrix(5,2)=0;   %%no ground current flow between  second node
node(source),only neutral current
```

```
for k=i:x
spani_jNCR_l=spani_jNCR_l+Node_current_matrix(5,k);
end
span_current_matrix(4,i)=spani_jNCR_l;

if (i>1)
for k=2:i
spani_jGCR_l=spani_jGCR_l+Node_current_matrix(6,k);
end

span_current_matrix(5,i+1)=spani_jGCR_l;
end

spani_jNCR_l=0;
spani_jGCR_l=0;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% span current matrix  -for phase currents
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:x

spani_jacT=zeros(1,1);
spani_jacT=0;
for k=i:x
spani_jacT=spani_jacT+Node_current_matrix(1,k);
end
span_current_matrix(1,i)=spani_jacT;

spani_jbcT=zeros(1,1);
spani_jbcT=0;
for k=i:x
spani_jbcT=spani_jbcT+Node_current_matrix(2,k);
end
span_current_matrix(2,i)=spani_jbcT;

spani_jccT=zeros(1,1);
spani_jccT=0;
for k=i:x
spani_jccT=spani_jccT+Node_current_matrix(3,k);
end
span_current_matrix(3,i)=spani_jccT;

end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% span current matrix  -for Resultant Neutral current
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for k=1:x
span_current_matrix(6,k)=span_current_matrix(4,k)-span_current_matrix(5,k);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Node VOLTAGE  MATRIX
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=2:x-1
SPAN_i_IMPEDENCE=zeros(4,4);
SPAN_i_IMPEDENCE=Spanmatrix(1,i-1)*MImpedence;
Direct_span_current_matrix=(-1)*span_current_matrix(6,i);
Span_drop_matrix(:,i)=SPAN_i_IMPEDENCE*[span_current_matrix(1,i);span_curren
t_matrix(2,i);span_current_matrix(3,i);Direct_span_current_matrix];
```

```
Phase_drop_matrix(:,i)=[Span_drop_matrix(1,i)-
Span_drop_matrix(4,i);Span_drop_matrix(2,i)-
Span_drop_matrix(4,i);Span_drop_matrix(3,i)-Span_drop_matrix(4,i)];

Node_voltage_matrix(4,i)=Node_voltage_matrix(4,i-1)-Span_drop_matrix(4,i);
Node_voltage_matrix(1,i)=Node_voltage_matrix(1,i-1)-Phase_drop_matrix(1,i);
Node_voltage_matrix(2,i)=Node_voltage_matrix(2,i-1)-Phase_drop_matrix(2,i);
Node_voltage_matrix(3,i)=Node_voltage_matrix(3,i-1)-Phase_drop_matrix(3,i);
end

SPAN_E_IMPEDENCE=Spanmatrix(1,x-1)*EImpedence;
Direct_span_current_matrix_x=(-1)*span_current_matrix(6,x);
Span_drop_matrix(:,x)=SPAN_E_IMPEDENCE*[span_current_matrix(1,x);span_curren
t_matrix(2,x);span_current_matrix(3,x);Direct_span_current_matrix_x];

Phase_drop_matrix(:,x)=[Span_drop_matrix(1,x)-
Span_drop_matrix(4,x);Span_drop_matrix(2,x)-
Span_drop_matrix(4,x);Span_drop_matrix(3,x)-Span_drop_matrix(4,x)];

Node_voltage_matrix(4,x)=Span_drop_matrix(4,x-1)-Span_drop_matrix(4,x);
Node_voltage_matrix(1,x)=Node_voltage_matrix(1,x-1)-Phase_drop_matrix(1,x);
Node_voltage_matrix(2,x)=Node_voltage_matrix(2,x-1)-Phase_drop_matrix(2,x);
Node_voltage_matrix(3,x)=Node_voltage_matrix(3,x-1)-Phase_drop_matrix(3,x);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calulated_Node_Power_matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%P=VI*

A=conj([Node_current_matrix(1,:);Node_current_matrix(2,:);Node_current_matri
x(3,:)]);
B=[Node_voltage_matrix(1,:);Node_voltage_matrix(2,:);Node_voltage_matrix(3,:
)];

for i=1:3
for j=1:x
Calulated_Node_Power_matrix(i,j)=B(i,j)*A(i,j);
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Node_Power_mismatch_matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Node_Power_mismatch_matrix=Load_matrix-
[Calulated_Node_Power_matrix(1,:);Calulated_Node_Power_matrix(2,:);Calulated
_Node_Power_matrix(3,:)];

max(abs(Node_Power_mismatch_matrix(:)));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Iteration process
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

while(max(abs(Node_Power_mismatch_matrix(:)))>0.0001)

for i=1:x

% Phase current calculation

Node_current_a=zeros(1,1);
Node_current_b=zeros(1,1);
```

```
Node_current_c=zeros(1,1);
Node_current_a=conj(Load_matrix(1,i)/Node_voltage_matrix(1,i));
Node_current_b=conj(Load_matrix(2,i)/Node_voltage_matrix(2,i));
Node_current_c=conj(Load_matrix(3,i)/Node_voltage_matrix(3,i));
Node_current_unbalance=(Node_current_a+Node_current_b+Node_current_c);

Node_current_matrix(1,i)=Node_current_a;
Node_current_matrix(2,i)=Node_current_b;
Node_current_matrix(3,i)=Node_current_c;
Node_current_matrix(4,i)=Node_current_unbalance;

%%%% Impedence ratio calculation %%%%%%%%%%

if i==x
Z_back_i=zeros(1,1);
Z_back_i=0;
Z_back_i=(2*Z_Grounding) + Z_Ground;

elseif i==x-1

Z_back_i=zeros(1,1);
Z_back_i=0;
Z_back_i=Espan(4,4) + (2*Z_Grounding) + Z_Ground;

else

Z_back=zeros(1,1);
zT_back_i=zeros(1,1);
Z_back_i=zeros(1,1);
Z_back=0;
zT_back_i=0;
Z_back_i=0;
for j=i:x-2
zT_back_i=Spanmatrix(1,j)*MImpedence(4,4);
Z_back=Z_back+zT_back_i;
end

Z_back_i=Z_back+Espan(4,4) + (2*Z_Grounding) + Z_Ground;
end

if i==1
z_front_i=0;

elseif  i==x
z_front=zeros(1,1);
zT_front_i=zeros(1,1);
z_front_i=zeros(1,1);
z_front=0;
zT_front_i=0;
z_front_i=0;
for k=1:i-2
zT_front_i=Spanmatrix(1,k)*MImpedence(4,4);
z_front=z_front+zT_front_i;
end
z_front_i=z_front+Espan(4,4);

else

z_front=zeros(1,1);
zT_front_i=zeros(1,1);
z_front_i=zeros(1,1);
z_front=0;
zT_front_i=0;
z_front_i=0;
```

```
for k=1:i-1
zT_front_i=Spanmatrix(1,k)*MImpedence(4,4);
z_front=z_front+zT_front_i;
end
z_front_i=z_front;
end

Node_current_n=Node_current_unbalance*(Z_back_i/(z_front_i+Z_back_i));
Node_current_g=Node_current_unbalance*(z_front_i/(z_front_i+Z_back_i));
Z_back_i=0;
z_front_i=0;

Node_current_matrix(5,i)=Node_current_n;
Node_current_matrix(6,i)=Node_current_g;

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% span current matrix  -for Ground and Neutral seperatly considering
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:x
spani_jNCR_l=zeros(1,1);
spani_jGCR_l=zeros(1,1);
span_current_matrix(5,1)=0;   %%no ground current flow between source and
first node ,only neutral current
span_current_matrix(5,2)=0;   %%no ground current flow between  first node
and second node ,only neutral current
for k=i:x
spani_jNCR_l=spani_jNCR_l+Node_current_matrix(5,k);
end
span_current_matrix(4,i)=spani_jNCR_l;

if (i>1)
for k=2:i
spani_jGCR_l=spani_jGCR_l+Node_current_matrix(6,k);
end

span_current_matrix(5,i+1)=spani_jGCR_l;
end

spani_jNCR_l=0;
spani_jGCR_l=0;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% span current matrix  -for phase currents
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=1:x

spani_jacT=zeros(1,1);
spani_jacT=0;
for k=i:x
spani_jacT=spani_jacT+Node_current_matrix(1,k);
end
span_current_matrix(1,i)=spani_jacT;

spani_jbcT=zeros(1,1);
spani_jbcT=0;
for k=i:x
spani_jbcT=spani_jbcT+Node_current_matrix(2,k);
end
span_current_matrix(2,i)=spani_jbcT;
```

```
spani_jccT=zeros(1,1);
spani_jccT=0;
for k=i:x
spani_jccT=spani_jccT+Node_current_matrix(3,k);
end
span_current_matrix(3,i)=spani_jccT;

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% span current matrix  -for Resultant Neutral current
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for k=1:x
span_current_matrix(6,k)=span_current_matrix(4,k)-span_current_matrix(5,k);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Node VOLTAGE  MATRIX
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=2:x-1
SPAN_i_IMPEDENCE=zeros(4,4);
SPAN_i_IMPEDENCE=Spanmatrix(1,i-1)*MImpedence;
Direct_span_current_matrix=(-1)*span_current_matrix(6,i);
Span_drop_matrix(:,i)=SPAN_i_IMPEDENCE*[span_current_matrix(1,i);span_curren
t_matrix(2,i);span_current_matrix(3,i);Direct_span_current_matrix];

Phase_drop_matrix(:,i)=[Span_drop_matrix(1,i)-
Span_drop_matrix(4,i);Span_drop_matrix(2,i)-
Span_drop_matrix(4,i);Span_drop_matrix(3,i)-Span_drop_matrix(4,i)];

Node_voltage_matrix(4,i)=Node_voltage_matrix(4,i-1)-Span_drop_matrix(4,i);
Node_voltage_matrix(1,i)=Node_voltage_matrix(1,i-1)-Phase_drop_matrix(1,i);
Node_voltage_matrix(2,i)=Node_voltage_matrix(2,i-1)-Phase_drop_matrix(2,i);
Node_voltage_matrix(3,i)=Node_voltage_matrix(3,i-1)-Phase_drop_matrix(3,i);
end

SPAN_E_IMPEDENCE=Spanmatrix(1,x-1)*EImpedence;
Direct_span_current_matrix_x=(-1)*span_current_matrix(6,x);
Span_drop_matrix(:,x)=SPAN_E_IMPEDENCE*[span_current_matrix(1,x);span_curren
t_matrix(2,x);span_current_matrix(3,x);Direct_span_current_matrix_x];

Phase_drop_matrix(:,x)=[Span_drop_matrix(1,x)-
Span_drop_matrix(4,x);Span_drop_matrix(2,x)-
Span_drop_matrix(4,x);Span_drop_matrix(3,x)-Span_drop_matrix(4,x)];

Node_voltage_matrix(4,x)=Span_drop_matrix(4,x-1)-Span_drop_matrix(4,x);
Node_voltage_matrix(1,x)=Node_voltage_matrix(1,x-1)-Phase_drop_matrix(1,x);
Node_voltage_matrix(2,x)=Node_voltage_matrix(2,x-1)-Phase_drop_matrix(2,x);
Node_voltage_matrix(3,x)=Node_voltage_matrix(3,x-1)-Phase_drop_matrix(3,x);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calulated_Node_Power_matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%P=VI*

A=conj([Node_current_matrix(1,:);Node_current_matrix(2,:);Node_current_matri
x(3,:)]);
B=[Node_voltage_matrix(1,:);Node_voltage_matrix(2,:);Node_voltage_matrix(3,:
)];

for i=1:3
for j=1:x
Calulated_Node_Power_matrix(i,j)=B(i,j)*A(i,j);
```

```
end
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Node_Power_mismatch_matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Node_Power_mismatch_matrix=Load_matrix-
[Calulated_Node_Power_matrix(1,:);Calulated_Node_Power_matrix(2,:);Calulated
_Node_Power_matrix(3,:)];

max(abs(Node_Power_mismatch_matrix(:)));

end

%Load_matrix;
%Node_current_matrix;
%span_current_matrix;
%Node_voltage_matrix;
%Calulated_Node_Power_matrix;
%
%Ac_span_current_matrix;

Validation_Table(Loop,1)=abs(Node_voltage_matrix(1,2));
Validation_Table(Loop,2)=abs(Node_voltage_matrix(1,5));
Validation_Table(Loop,3)=abs(Node_voltage_matrix(1,7));
Validation_Table(Loop,4)=abs(Node_voltage_matrix(1,8));
Validation_Table(Loop,5)=abs(Node_voltage_matrix(2,3));
Validation_Table(Loop,6)=abs(Node_voltage_matrix(2,4));
Validation_Table(Loop,7)=abs(Node_voltage_matrix(2,8));
Validation_Table(Loop,8)=abs(Node_voltage_matrix(3,8));

Validation_Table(Loop,9)=abs(Node_current_matrix(1,2));
Validation_Table(Loop,10)=abs(Node_current_matrix(1,5));
Validation_Table(Loop,11)=abs(Node_current_matrix(1,7));
Validation_Table(Loop,12)=abs(Node_current_matrix(2,3));
Validation_Table(Loop,13)=abs(Node_current_matrix(2,4));

Validation_Table(Loop,14)=abs(Ac_span_current_matrix(1,1));
Validation_Table(Loop,15)=abs(Ac_span_current_matrix(2,1));
Validation_Table(Loop,16)=abs(Ac_span_current_matrix(3,1));
Validation_Table(Loop,17)=abs(Ac_span_current_matrix(4,1));

end

%%% node Voltage comparison

Node_2VRME=[V_V(:,4)     Validation_Table(:,1)  V_V(:,5)];
Node_3VYME=[V_V(:,6)     Validation_Table(:,5)  V_V(:,7)];
Node_4VYME=[V_V(:,8)     Validation_Table(:,6)  V_V(:,9)];
Node_5VRME=[V_V(:,10)    Validation_Table(:,2)  V_V(:,11)];
Node_7VRME=[V_V(:,15)    Validation_Table(:,3)];
Node_8VRME=[V_V(:,16)    Validation_Table(:,4)];
Node_8VYME=[V_V(:,17)    Validation_Table(:,7)];
Node_8VBME=[V_V(:,18)    Validation_Table(:,8)];

%%% node current comparison

Node_2CRME=[NC_X(:,3)     Validation_Table(:,9)];
Node_3CYME=[NC_X(:,6)      Validation_Table(:,12)];
Node_4CYME=[NC_X(:,9)     Validation_Table(:,13)];
Node_5CRME=[NC_X(:,12)    Validation_Table(:,10)];
Node_7CRME=[NC_X(:,14)    Validation_Table(:,11)];
```

```
Node_Power_mismatch_matrix;

Voltage_Errors=[(Node_2VRME(:,2)-Node_2VRME(:,1))      (Node_3VYME(:,2)-
Node_3VYME(:,1))    (Node_4VYME(:,2)-Node_4VYME(:,1))  (Node_5VRME(:,2)-
Node_5VRME(:,1))  (Node_7VRME(:,2)-Node_7VRME(:,1))    (Node_8VRME(:,2)-
Node_8VRME(:,1))  (Node_8VYME(:,2)-Node_8VYME(:,1)) (Node_8VBME(:,2)-
Node_8VBME(:,1))];
numel(Voltage_Errors)
flatA = Voltage_Errors(:);
count_v_errors=sum(flatA > -0.5 & flatA < 0.5);
Percentage_confidence_V=(count_v_errors/numel(Voltage_Errors))*100;

Voltage_Errors=[(Node_2VRME(:,2)-Node_2VRME(:,1))      (Node_3VYME(:,2)-
Node_3VYME(:,1))    (Node_4VYME(:,2)-Node_4VYME(:,1))  (Node_5VRME(:,2)-
Node_5VRME(:,1))  (Node_7VRME(:,2)-Node_7VRME(:,1))    (Node_8VRME(:,2)-
Node_8VRME(:,1))  (Node_8VYME(:,2)-Node_8VYME(:,1)) (Node_8VBME(:,2)-
Node_8VBME(:,1))];
numel(Voltage_Errors)
flatA = Voltage_Errors(:);
count_v_errors=sum(flatA < -0.5 );
Percentage_confidence_V=(count_v_errors/numel(Voltage_Errors))*100;

Voltage_Errors=[(Node_2VRME(:,2)-Node_2VRME(:,1))      (Node_3VYME(:,2)-
Node_3VYME(:,1))    (Node_4VYME(:,2)-Node_4VYME(:,1))  (Node_5VRME(:,2)-
Node_5VRME(:,1))  (Node_7VRME(:,2)-Node_7VRME(:,1))    (Node_8VRME(:,2)-
Node_8VRME(:,1))  (Node_8VYME(:,2)-Node_8VYME(:,1)) (Node_8VBME(:,2)-
Node_8VBME(:,1))];
numel(Voltage_Errors)
flatA = Voltage_Errors(:);
count_v_errors=sum(flatA > +0.5 );
Percentage_confidence_V=(count_v_errors/numel(Voltage_Errors))*100;

Current_Errors=[(Node_2CRME(:,2)-Node_2CRME(:,1))    (Node_3CYME(:,2)-
Node_3CYME(:,1))   (Node_4CYME(:,2)-Node_4CYME(:,1))   (Node_5CRME(:,2)-
Node_5CRME(:,1)) (Node_7CRME(:,2)-Node_7CRME(:,1))];
numel(Current_Errors);
flatB = Current_Errors(:);
count_c_errors=sum(flatB  > -0.2 & flatB  < 0.2);
Percentage_confidence_c=(count_c_errors/numel(Current_Errors))*100;
```

Appendix C - Consumer Connection Pattern Generation

Consumer connection pattern generation algorithm.

```
syms C1 C2 Ph_c_L Ph_b_L Ph_a_L Acum_Perm_2
data_2= [C1, C2];
if (0<(rem(numel(data_2),3)))
for i=numel(data_2)+1:(numel(data_2)+3-rem(numel(data_2),3));
data_2(1,i)=0;
end
end
%%%%%% creating permutation matrix (All possible arrangements of load among
phases %%%%%%%
P_2=perms(data_2);
%%%%%% NUmber of load elements in permutation matrix %%%%%%
numel(P_2);
%%%%%% size of  permutation matrix %%%%%%
size(P_2);
for z=1:size(P_2,1);
%%%%%Accumilation of loads related to "phase a" in  permutaion matrix at zth
permutation %%%%
Ph_a_L=zeros(1,1);
k=1;
while (k<(numel(data_2)))
Ph_a_L=Ph_a_L+P_2(z,k);
k=k+3;
end
%%%%%Accumilation of loads related to "phase a" in  permutaion matrix at zth
permutation %%%%
Ph_b_L=zeros(1,1);
j=2;
while (j<(numel(data_2)))
Ph_b_L=Ph_b_L+P_2(z,j);
j=j+3;
end
%%%%%Accumilation of loads related to "phase c" in  permutaion matrix at zth
permutation %%%%
Ph_c_L=zeros(1,1);
l=3;
while (l<=(numel(data_2)))
Ph_c_L=Ph_c_L+P_2(z,l);
l=l+3;
end
Acum_Perm_2(z,1)=Ph_a_L;
Acum_Perm_2(z,2)=Ph_b_L;
Acum_Perm_2(z,3)=Ph_c_L;
end
syms C3 C4 Ph_c_L Ph_b_L Ph_a_L Acum_Perm_3
data_3=[C3,C4];
if (0<(rem(numel(data_3),3)))
for i=numel(data_3)+1:(numel(data_3)+3-rem(numel(data_3),3));
data_3(1,i)=0;
end
end
%%%%%% creating permutation matrix (All possible arrangements of load among
phases %%%%%%%
P_3=perms(data_3);
%%%%%% NUmber of load elements in permutation matrix %%%%%%
```

```
numel(P_3);
%%%%%% size of  permutation matrix %%%%%%
size(P_3);
for z=1:size(P_3,1);
%%%%%Accumilation of loads related to "phase a" in  permutaion matrix at zth
permutation %%%%
Ph_a_L=zeros(1,1);
k=1;
while (k<(numel(data_3)))
Ph_a_L=Ph_a_L+P_3(z,k);
k=k+3;
end
%%%%%Accumilation of loads related to "phase a" in  permutaion matrix at zth
permutation %%%%
Ph_b_L=zeros(1,1);
j=2;
while (j<(numel(data_3)))
Ph_b_L=Ph_b_L+P_3(z,j);
j=j+3;
end
%%%%%Accumilation of loads related to "phase c" in  permutaion matrix at zth
permutation %%%%
Ph_c_L=zeros(1,1);
l=3;
while (l<=(numel(data_3)))
Ph_c_L=Ph_c_L+P_3(z,l);
l=l+3;
end
Acum_Perm_3(z,1)=Ph_a_L;
Acum_Perm_3(z,2)=Ph_b_L;
Acum_Perm_3(z,3)=Ph_c_L;
end
syms C5 C6 Ph_c_L Ph_b_L Ph_a_L Acum_Perm_4
data_4=[C5 C6];
if (0<(rem(numel(data_4),3)))
for i=numel(data_4)+1:(numel(data_4)+3-rem(numel(data_4),3));
data_4(1,i)=0;
end
end
%%%%%% creating permutation matrix (All possible arrangements of load among
phases %%%%%%%
P_4=perms(data_4);
%%%%%% NUmber of load elements in permutation matrix %%%%%%
numel(P_4);
%%%%%% size of  permutation matrix %%%%%%
size(P_4);
for z=1:size(P_4,1);
%%%%%Accumilation of loads related to "phase a" in  permutaion matrix at zth
permutation %%%%
Ph_a_L=zeros(1,1);
k=1;
while (k<(numel(data_4)))
Ph_a_L=Ph_a_L+P_4(z,k);
k=k+3;
end
%%%%%Accumilation of loads related to "phase B" in  permutaion matrix at zth
permutation %%%%
Ph_b_L=zeros(1,1);
j=2;
while (j<(numel(data_4)))
Ph_b_L=Ph_b_L+P_4(z,j);
j=j+3;
end
```

```
%%%%%Accumilation of loads related to "phase c" in  permutaion matrix at zth
permutation %%%%
Ph_c_L=zeros(1,1);
l=3;
while (l<=(numel(data_4)))
Ph_c_L=Ph_c_L+P_4(z,l);
l=l+3;
end
Acum_Perm_4(z,1)=Ph_a_L;
Acum_Perm_4(z,2)=Ph_b_L;
Acum_Perm_4(z,3)=Ph_c_L;
end
syms C7 C8 Ph_c_L Ph_b_L Ph_a_L Acum_Perm_5
data_5=[C7 C8];
if (0<(rem(numel(data_5),3)))
for i=numel(data_5)+1:(numel(data_5)+3-rem(numel(data_5),3));
data_5(1,i)=0;
end
end
%%%%%% creating permutation matrix (All possible arrangements of load among
phases %%%%%%%
P_5=perms(data_5);
%%%%%% NUmber of load elements in permutation matrix %%%%%%
numel(P_5);
%%%%%% size of  permutation matrix %%%%%%
size(P_5);
for z=1:size(P_5,1);
%%%%%Accumilation of loads related to "phase a" in  permutaion matrix at zth
permutation %%%%
Ph_a_L=zeros(1,1);
k=1;
while (k<(numel(data_5)))
Ph_a_L=Ph_a_L+P_5(z,k);
k=k+3;
end
%%%%%Accumilation of loads related to "phase B" in  permutaion matrix at zth
permutation %%%%
Ph_b_L=zeros(1,1);
j=2;
while (j<(numel(data_5)))
Ph_b_L=Ph_b_L+P_5(z,j);
j=j+3;
end
%%%%%Accumilation of loads related to "phase c" in  permutaion matrix at zth
permutation %%%%
Ph_c_L=zeros(1,1);
l=3;
while (l<=(numel(data_5)))
Ph_c_L=Ph_c_L+P_5(z,l);
l=l+3;
end
Acum_Perm_5(z,1)=Ph_a_L;
Acum_Perm_5(z,2)=Ph_b_L;
Acum_Perm_5(z,3)=Ph_c_L;
end
syms C9  Ph_c_L Ph_b_L Ph_a_L Acum_Perm_7
data_7=[C9];
if (0<(rem(numel(data_7),3)))
for i=numel(data_7)+1:(numel(data_7)+3-rem(numel(data_7),3));
data_7(1,i)=0;
end
end
%%%%%% creating permutation matrix (All possible arrangements of load among
phases %%%%%%%
```

```
P_7=perms(data_7);
%%%%%% NUmber of load elements in permutation matrix %%%%%%
numel(P_7);
%%%%%% size of  permutation matrix %%%%%%
size(P_7);
for z=1:size(P_7,1);
%%%%%Accumilation of loads related to "phase a" in  permutaion matrix at zth
permutation %%%%
Ph_a_L=zeros(1,1);
k=1;
while (k<(numel(data_7)))
Ph_a_L=Ph_a_L+P_7(z,k);
k=k+3;
end
%%%%%Accumilation of loads related to "phase B" in  permutaion matrix at zth
permutation %%%%
Ph_b_L=zeros(1,1);
j=2;
while (j<(numel(data_7)))
Ph_b_L=Ph_b_L+P_7(z,j);
j=j+3;
end
%%%%%Accumilation of loads related to "phase c" in  permutaion matrix at zth
permutation %%%%
Ph_c_L=zeros(1,1);
l=3;
while (l<=(numel(data_7)))
Ph_c_L=Ph_c_L+P_7(z,l);
l=l+3;
end
Acum_Perm_7(z,1)=Ph_a_L;
Acum_Perm_7(z,2)=Ph_b_L;
Acum_Perm_7(z,3)=Ph_c_L;
end
S2=Acum_Perm_2;
S3=Acum_Perm_3;
S4=Acum_Perm_4;
S5=Acum_Perm_5;
S7=Acum_Perm_7;
syms C Load_matrix
[p7,p5,p4,p3,p2] =
ndgrid(1:size(S7,1),1:size(S5,1),1:size(S4,1),1:size(S3,1),1:size(S2,1));
Combination= [S2(p2,:),S3(p3,:),S4(p4,:),S5(p5,:),S7(p7,:)];”
```

Appendix D - Characteristic Curve Generation

Following code used to generate characteristic curves for nine consumers in the considered node.Input data stored in the excel workbooks and retrieve during program execution.

- Total individual consumptions of consumers for the period of characteristic curve generation are stored in the ‘Individual consumption CC generation period.xlsx'.

- Active and reactive power consumptions of consumers for the period of characteristic curve generation are stored in the ‘Consumers data.xlsx'.

Program for Characteristic Curve Generation

```
%%%%%%%%%%%%%%%%Individual actual monthly consumption extraction %%%%%%%%%%
format short g
exl = actxserver('excel.application');
exlWkbk = exl.Workbooks;
exlFile = exlWkbk.Open([docroot
'/techdoc/matlab_external/examples/Individual consumption CC generation
period.xlsx']);
exlSheet1 = exlFile.Sheets.Item(1);
robj = exlSheet1.Columns.End(4);       % Find the end of the column
numrows = robj.row;                    % And determine what row it is
rngObj = exlSheet1.Range('B2:B11');
exlData_C = rngObj.Value;
Individual_consumption=cell2mat(exlData_C);

%%%%%%%%%%%%%%%%curve generation %%%%%%%%%%%%%%%%

for f=1:9;
exl = actxserver('excel.application');
exlWkbk = exl.Workbooks;
exlFile = exlWkbk.Open([docroot '/techdoc/matlab_external/examples/Consumers
data.xlsx']);
exlSheet1 = exlFile.Sheets.Item(f);
robj = exlSheet1.Columns.End(4);       % Find the end of the column
numrows = robj.row;                    % And determine what row it is
rngObj = exlSheet1.Range('A2885:U2980');
exlData = rngObj.Value;
W_Active=cell2mat(exlData);

rngObj = exlSheet1.Range('A2987:I3082');
exlData = rngObj.Value;
H_Active=cell2mat(exlData);

rngObj = exlSheet1.Range('A3088:U3183');
exlData = rngObj.Value;
W_RActive=cell2mat(exlData);

rngObj = exlSheet1.Range('A3190:I3285');
exlData = rngObj.Value;
H_RActive=cell2mat(exlData);
```

```
size(W_Active);
size(H_Active);
size(W_RActive);
size(H_RActive);


%%%%  characteristic curve generation for working day active power
x=0;
Number_working_days=0;
sort_Raw_Data =0;

x=size(W_Active);
Number_working_days=x (1,2);
for T_i=1:96
sort_Raw_Data = sortrows(transpose(W_Active(T_i,:)));


for h=1:Number_working_days
while (abs((max(sort_Raw_Data)-
median(sort_Raw_Data)))>25*iqr(sort_Raw_Data))
sort_Raw_Data(sort_Raw_Data == max(sort_Raw_Data)) = [];
end
while (abs((min(sort_Raw_Data)-
median(sort_Raw_Data)))>25*iqr(sort_Raw_Data))
sort_Raw_Data(sort_Raw_Data == min(sort_Raw_Data)) = [];
end
h=h+1;
end
[idx,C] = kmeans(sort_Raw_Data,1);
Characteristic_point_active(T_i,1)=C;

T_i=T_i+1;

end

%%%%  characteristic curve generation for working day  reactive power
y=0;
Number_working_days=0;
sort_Raw_Data =0;

y=size(W_RActive);
Number_working_days=y (1,2);
for T_i=1:96
sort_Raw_Data = sortrows(transpose(W_RActive(T_i,:)));


for h=1:Number_working_days

while (abs((max(sort_Raw_Data)-
median(sort_Raw_Data)))>25*iqr(sort_Raw_Data))
sort_Raw_Data(sort_Raw_Data == max(sort_Raw_Data)) = [];
end
while (abs((min(sort_Raw_Data)-
median(sort_Raw_Data)))>25*iqr(sort_Raw_Data))
sort_Raw_Data(sort_Raw_Data == min(sort_Raw_Data)) = [];
end

h=h+1;
end
[idx,C] = kmeans(sort_Raw_Data,1);
```

```
Characteristic_point_Ractive(T_i,1)=C;

T_i=T_i+1;

end

%%%%complex characteristic curve for working day


complex_characteristic_w=complex(Characteristic_point_active,Characteristic_
point_Ractive);


%%%%  characteristic curve generation for Holiday day active power
x=0;
Number_working_days=0;
sort_Raw_Data =0;

x=size(H_Active);
Number_working_days=x (1,2);
for T_i=1:96
sort_Raw_Data = sortrows(transpose(H_Active(T_i,:)));


for h=1:Number_working_days

while (abs((max(sort_Raw_Data)-
median(sort_Raw_Data)))>25*iqr(sort_Raw_Data))
sort_Raw_Data(sort_Raw_Data == max(sort_Raw_Data)) = [];
end
while (abs((min(sort_Raw_Data)-
median(sort_Raw_Data)))>25*iqr(sort_Raw_Data))
sort_Raw_Data(sort_Raw_Data == min(sort_Raw_Data)) = [];
end
h=h+1;
end
[idx,C] = kmeans(sort_Raw_Data,1);
Characteristic_point_H_active(T_i,1)=C;

T_i=T_i+1;

end

%%%%  characteristic curve generation for working day  reactive power
y=0;
Number_working_days=0;
sort_Raw_Data = 0;

y=size(H_RActive);
Number_working_days=y (1,2);
for T_i=1:96
sort_Raw_Data = sortrows(transpose(H_RActive(T_i,:)));


for h=1:Number_working_days

while (abs((max(sort_Raw_Data)-
median(sort_Raw_Data)))>25*iqr(sort_Raw_Data))
sort_Raw_Data(sort_Raw_Data == max(sort_Raw_Data)) = [];
end
while (abs((min(sort_Raw_Data)-
median(sort_Raw_Data)))>25*iqr(sort_Raw_Data))
sort_Raw_Data(sort_Raw_Data == min(sort_Raw_Data)) = [];
end
```

```
h=h+1;
end
[idx,C] = kmeans(sort_Raw_Data,1);
Characteristic_point_H_Ractive(T_i,1)=C;

T_i=T_i+1;

end
complex_characteristic_H=complex(Characteristic_point_H_active,Characteristi
c_point_H_Ractive);


%%%%complex characteristic curve


Total_Charactristic_curve_Wmatrix(:,f)=[complex_characteristic_w];
Total_Charactristic_curve_Hmatrix(:,f)=[complex_characteristic_H];
end


%%%%%%%%%End of charactristic curve generation %%%%%%%%%%%%%%%%%


%%%%%%%%%kWh calcualtion for 21- working day for nine consumers using
characteristic curves%%%%%%%%%%%%%%%%%%

Total_Charactristic_curve_Wmatrix_active=real(Total_Charactristic_curve_Wmat
rix);
Calculated_units_W=((sum(Total_Charactristic_curve_Wmatrix_active)*0.25)/100
0)*21;

Total_Charactristic_curve_Wmatrix_Ractive=imag(Total_Charactristic_curve_Wma
trix);

%%%%%%%%%kWh calcualtion for Holiday day for nine consumers using
characteristic curves%%%%%%%%%%%%%%%%%%


Total_Charactristic_curve_Hmatrix_active=real(Total_Charactristic_curve_Hmat
rix);
Calculated_units_H=((sum(Total_Charactristic_curve_Hmatrix_active)*0.25)/100
0)*9;

Total_Charactristic_curve_Hmatrix_Ractive=imag(Total_Charactristic_curve_Hma
trix);

%%%%%%%%%Total kWh calcualtion for Holiday day for nine consumers using
characteristic curves%%%%%%%%%%%%%%%%%%


Calculated_units_consumer=Calculated_units_W+Calculated_units_H;

Calculated_units_consumer(1,10)=sum(Calculated_units_consumer(:));


Unit_difference=Individual_consumption-transpose(Calculated_units_consumer);
```