* **Average CT marks calculations**

**clc;** %Clears previous data from command window

**clear all;** %Removes all variables from the current workspace

**cd('F:\Study material\Lab\3-2\Power System I');** %Changes file directory

**x = xlsread('Exp01')** %Imports data from excel file

**fprintf('\n Roll No \tCTI CT2 CT3 CT4 CT5 CT6\n')** %Display the text

**disp(x)** %Display the data inside variable

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

**y = x(:,2:n);** %Isolates the data to be averaged from the roll

**fprintf('Marks:\n')** %Display the text

**disp(y)** %Display the data inside variable

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

**fprintf('\nSorting descending order rawwise: \n')** %Display the text

**disp(w)** %display the data inside variable

**z = w(:,1:3)** %Takes the first three columns containing highest three marks

**fprintf('\nBest three marks: \n')** %Display the text

**disp(z)** %Display the data inside variable

**m = mean(z,2)** %Calculates mean of the highest three marks. 2 is for doing the action rawwise

**fprintf('\nAverage marks: \n')** %Display the text

**disp(m)** %Display the data inside variable

**Output = round(m)** %Round the calculated data

**fprintf('Rounding the average marks: \n')** %Display the text

**disp(Output)** %display the value inside variable

**Roll=x(:,1)** %Taking the column of Roll

**Y=[Roll Output]** %Forming a matrix of column Roll and Attained data as marks

**fprintf(' Roll No Attained Marks \n')** %Display the text

**disp(Y)** %Display the marks inside variable

* **Ascending and descending**

**clc;** %Clears previous data from command window

**clear all;** %Removes all variables from the current workspace

**cd('F:\Study material\Lab\3-2\Power System I');** %Changes file directory

**Matrix=xlsread('Exp01p02');** %Reads from excel file

**fprintf('Matrix:');** %Prints the data

**disp(Matrices)** %Shows the output

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

%Ascending

**for j=1:n** %Campare first elements

**for k=j+1:n** %Campare second elements

**if Matrix(j)>=Matrix(k)** %Compare greater or not

**m=Matrix(j);** %Store the greater number in a variable

**Matrix(j)=Matrix(k);** %Replace the greater number by the smaller one

**Matrix(k)=m;** %Replace the smaller number with greater number

**end**

**end**

**end**

**fprintf('Ascending: ');** %Print the data in desired order

%Descending

**disp(Matrix)** %Show the output

**Output=xlsread('Exp01p02');** %Read from excel file

**n=length(Output);** %Read the number of elements

**for j=1:n** %Campare first elements

**for k=j+1:n** %Campare second elements

**if Output(j)<=Output(k)** %Compare samller or not

**m=Output(j);**%Store the smaller number in a variable

**Output(j)=Output(k);**%Replace the smaller number by the smaller one

**Output(k)=m;** %Replace the greater number with smaller number

**end**

**end**

**end**

**fprintf('Descending: ');** %Printing the data

**disp(Output)** %Show the output

* **Generate Y-bus matrix**

**clc;** %Clears previous data from command window

**clear all;** %Removes all variables from the current workspace

**cd('F:\Study material\Lab\3-2\Power System I');** %change the file directory

**A = xlsread('Exp02');** %Read the excel file

**n = length(A);** %Determine the length of the excel file

% Applying symmetric condition

**for w=1:n**

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

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

**end**

**m = length(Z)** %Determine the length of the new matrix

**for j=1:m**

**for k=1:m**

**if Z(j,k) == 0**

**Z(j,k) = inf;**

**end**

**end**

**end**

**fprintf(' Z matrix is \n')** %Display the text

**disp(Z)** %Display the output

**y = 1./Z** %Taking inverse impedance matrix

**p = sum(y,2)** %Taking symmetric summation

%Apply looping condition to determine value of the matrix element

**for u=1:m**

**for x=1:m**

**if u~=x**

**Y(u,x)= -y(u,x);** %For diagonal element

**else**

**Y(u,x)= p(u);** %For non-diagonal element

**end**

**end**

**end**

**fprintf(' Y- bus matrix is \n')** %Display the text

**disp(Y)** %Display the output

* **Reduced Y-bus**

**clc;** %Clears previous data from command window

**clear all;** %Removes all variables from the current workspace

**cd('F:\Study material\Lab\3-2\Power System I');** %change the file directory

**A = xlsread('Exp02');** %Read the excel file

**n = length(A);** %Determine the length of the excel file

% Applying symmetric condition

**for w=1:n**

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

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

**end**

**m = length(Z)** %Determine the length of the new matrix

**for j=1:m**

**for k=1:m**

**if Z(j,k) == 0**

**Z(j,k) = inf;**

**end**

**end**

**end**

**fprintf(' Z matrix is \n')** %Display the text

**disp(Z)** %Display the output

**y = 1./Z** %Taking inverse impedance matrix

**p = sum(y,2)** %Taking symmetric summation

%Apply looping condition to determine value of the matrix element

**for u=1:m**

**for x=1:m**

**if u~=x**

**Y(u,x)= -y(u,x);** %For diagonal element

**else**

**Y(u,x)= p(u);** %For non-diagonal element

**end**

**end**

**end**

**fprintf(' Y- bus matrix is \n')** %Display the text

**disp(Y)** %Display the output

**d = length(Y);**

**f = input('Total no. of buses: ');**

**g = input('No. of reduction: ');**

**for h=1:g** %No. of reduction

**F = zeros(d-1);** %Define a new matrix

**for t=1:(d-1)** %Access all Y matrix element

**for r=1:(d-1) %**

**F(t,r) = Y(t,r)-((Y(t,d)\*Y(d,r))/Y(d,d));**

**end**

**end**

**Y = F;**

**d = d-1;**

**end**

**fprintf(' Z matrix is \n')** %Display the text

**disp(Y)**

* **Gaus-Seidal**

**clc;** %Clears previous data from command window

**clear all;** %Removes all variables from the current workspace

**cd('F:\Study material\Lab\3-2\Power System I');** %change the file directory

**A = xlsread('EXp02p02');** %Read the excel file

**n = length(A);** %Determine the length of the excel file

% Applying symmetric condition

**for w=1:n**

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

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

**end**

**m = length(Z);** %Determine the length of the new matrix

**for j=1:m**

**for k=1:m**

**if Z(j,k) == 0**

**Z(j,k) = inf;**

**end**

**end**

**end**

**fprintf(' Z matrix is \n')** %Display the text

**disp(Z)** %Display the output

**y = 1./Z** %Taking inverse impedance matrix

**p = sum(y,2)** %Taking symmetric summation

%Apply looping condition to determine value of the matrix element

**for u=1:m**

**for x=1:m**

**if u~=x**

**Y(u,x)= -y(u,x);** %For diagonal element

**else**

**Y(u,x)= p(x);** %For non-diagonal element

**end**

**end**

**end**

**fprintf(' Y- bus matrix is \n')** %Display the text

**disp(Y)** %Display the output

**cd('F:\Study material\Lab\3-2\Power System I');** %change the file directory

**B = xlsread('Exp05');** %Read the excel file

**j = 3;**

**V = B(:,2);**

**V0=B(:,2);**

%to get the value of real power

**P=B(:,3)-B(:,5);**

%to get the value of reactive power

**Q=B(:,4)-B(:,6);**

%to get the angle

**ang=B(:,7);**

**V1=V;**

% to get the value of generator bus

**Pg=B(:,3);**

**for w=1:100**

**z=V;**

**for k=2:j**

**yv1=0;**

**yv2=0;**

**for h=1:j**

**yv2=yv2+Y(k,h)\*V(h);** %to find the product of Y bus and voltages

**if h~=k**

**yv1=yv1+Y(k,h)\*V(h);** %to find the product of Y bus and voltages

**end**

**end**

**if Pg(k)~=0**

**g(k)=imag(V(k)\*(conj(yv2)));** %to get the imaginary value

**S(k)=P(k)+1i\*g(k);** %to calculate the apparent power

**else S(k)=P(k)+1i\*Q(k);**

**end**

**V(k)=(1/Y(k,k))\*((conj(S(k))/conj(V(k)))-yv1);** %to get the value of node voltages

**ang1(k)=angle(V(k));** %to get the angles

**ang2(k)=rad2deg(ang1(k));** %to convert the radian values to degrees

**if Pg(k)~=0**

**V(k)=V0(k)\*exp(1i\*ang1(k));**

**end**

**end**

**V1=abs(V);**

**ang2=rad2deg(ang1);**

**E=abs((V-z)/V);**

**if E<=10e-4**

**break;** %to break the for loop

**end**

**Vlt\_1(w)=V1(1);**

**Vlt\_2(w)=V1(2);**

**Vlt\_3(w)=V1(3);**

**ang\_1(w)=ang2(1);**

**ang\_2(w)=ang2(2);**

**ang\_3(w)=ang2(3);**

**end**

% to show the value column wise

**Vlt\_1=Vlt\_1';**

**ang\_1=ang\_1';**

**Vlt\_2=Vlt\_2';**

**ang\_2=ang\_2';**

**Vlt\_3=Vlt\_3';**

**ang\_3=ang\_3';**

**iteration=(1:w-1)';**

% to show the values in a table

**table(iteration,Vlt\_1,ang\_1,Vlt\_2,ang\_2,Vlt\_3,ang\_3)**

* **Zero-positive-negative sequence**

**clc;** %Clears previous data from command window

**clear all;** %Removes all variables from the current workspace

% magnitude and phase of unbalanced voltage V1

**M1 = input('Enter magnitude of V1:');**

**P1 = input('Enter phase of V1:');**

**V1 = M1.\*exp(j\*deg2rad(P1))**

% magnitude and phase of unbalanced voltage V2

**M2 = input('Enter magnitude of V2:');**

**P2 = input('Enter phase of V2:');**

**V2 = M2.\*exp(j\*deg2rad(P2))**

% magnitude and phase of unbalanced voltage V3

**M3 = input('Enter magnitude of V3:');**

**P3 = input('Enter phase of V3:');**

**V3 = M3.\*exp(j\*deg2rad(P3))**

% calculate a

**a = 1.\*exp(j\*2\*pi/3);**

**anew = a.\*a;**

% finding zero sequence

**Va0 = (1/3)\*(V1+V2+V3);**

**Vb0 = Va0;**

**Vc0 = Va0;**

% finding positive sequence

**Va1 = (1/3)\*(V1+a\*V2+anew\*V3);**

**Vb1 = anew\*Va1;**

**Vc1 = a\*Va1;**

% finding negative sequence

**Va2 = (1/3)\*(V1+anew\*V2+a\*V3);**

**Vb2 = a\*Va2;**

**Vc2 = anew\*Va2;**

% finding original sequence

**Va = (Va0+Va1+Va2); Vb = (Vb0+Vb1+Vb2); Vc = (Vc0+Vc1+Vc2);**

% plotting unbalanced sequence

**subplot(2,3,1); compass([V1,V2,V3]); title('Unbalanced Sequence');**

% plotting zero sequence

**subplot(2,3,2); compass([Va0,Vb0,Vc0]); title('Zero Sequence');**

% finding positive sequence

**subplot(2,3,3); compass([Va1,Vb1,Vc1]); title('Positive Sequence');**

% finding negative sequence

**subplot(2,3,4); compass([Va2,Vb2,Vc2]); title('Negative Sequence');**

% finding original sequence

**subplot(2,3,6); compass([Va,Vb,Vc]); title('Original Sequence');**

* **Transient response analysis**

**clc;** %Clears previous data from command window

**clear all;** %Removes all variables from the current workspace

% Declaring function

**syms i(t)**

% Assigning values to the variable

**R = 50;** % Resistance

**a = pi/3;** % Phase angle

**L = 300e-3;** % Inductance

**f = 100;** % Frequency

**w = 2\*pi\*f;**

**Vm = 100;** % Voltage

% Formula

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

% Plotting function

**ezplot(p)**

**grid on**

% Labeling plot

**xlabel('Time(sec)')**

**ylabel('Current(amp)')**

**title('Transient response')**