

```
1 clc;clear;
2 %% Standardizing generator sub-transient reactances
3 s_new = 100; % MVA
4 s_old = [250; 100; 80; 50; 50;]; % MVA
5
6 x1 = 15/100;
7 x2 = 12/100;
8 x3 = 10/100;
9 x4 = 9/100;
10 x5 = 8/100;
11 x = [1 x1; 2 x2; 3 x3; 4 x4; 5 x5];
12
13 % converting xolds to xnews
14 x(:,2) = (x(:,2)./s_old)*s_new;
15 [r_x,~] = size(x);
16 %% Y bus calculation
17
18 % import data as table
19 linedata = readmatrix("line_data.xlsx");
20 for k=1:r_x
21     linedata(end+1,1) = x(k,1);
22     linedata(end,2) = x(k,1);
23     linedata(end,4) = x(k,2);
24 end
25 [r_data,~] = size(linedata);
26
27 % Preprocessing
28 number_of_buses = max(max(linedata(:,1:2))); % picking the highest numbered bus
29 y_bus = zeros(number_of_buses);
30 [length_y_bus,~] = size(y_bus);
31
32 % formatting the impedances
33 R = linedata(:,3);
34 X = linedata(:,4);
35 B = linedata(:,5);
36 impedances = [R X.*1i B*1i./2];
37
38 % calculating the diagonal elements
39 for m=1:length_y_bus % m is the index of the y bus (row)
40     z = zeros(1,3); % initialize 3 columns (R,X,B)
41     for n=1:r_data % n is the index of the linedata (row)
42         if linedata(n,1)==m||linedata(n,2)==m
43             z = [z; linedata(n,3:5)]; % collecting all the impedances
44         end
45     end
46     [r_z,~] = size(z);
47     for impedance=1:r_z
48         R = z(impedance,1);
49         X = z(impedance,2);
```

```

50     B = z(impedance,3);
51     if R~=0 || X~=0
52         y_bus(m,m) = y_bus(m,m) + 1./(R+X.*1i) + B.*1i/2;
53     end
54 end
55 end
56
57 % calculating off diagonal elements
58 for m=1:length_y_bus % first index of Y element
59     for n=1:length_y_bus % second index of Y element
60         current_nodes = [m n]
61         if m~=n
62             for k=1:r_data
63                 if linedata(k,1)==m && linedata(k,2)==n
64                     R = linedata(k,3);
65                     X = linedata(k,4);
66                     B = linedata(k,5);
67                     current_z = [R X B]
68                     y_bus(m,n) = -1/(R+1i*X)+B*1i/2;
69                     y_bus(n,m) = y_bus(m,n);
70                 end
71             end
72         end
73     end
74 end
75
76
77
78
79 %% z bus calculation
80 z_bus = y_bus^-1;
81
82
83 %% prefault voltages
84
85 v = readmatrix('voltages.xlsx'); % prefault voltages
86 v_magnitudes = v(:,1);
87 v_angles = deg2rad(v(:,2));
88
89 % converting to rectangular
90 vx = v_magnitudes.*cos(v_angles);
91 vy = v_magnitudes.*sin(v_angles);
92 v = vx+vy*1i;
93
94 %% q1: sags
95
96 %Calculate the voltage sag at bus 4 and bus 13 when a three-phase-fault occurs at
each bus
97 %in the system

```

```
98 sag_4_q1 = zeros(14,1);
99 sag_13_q1 = zeros(14,1);
100 for bus=1:14
101     %if bus~=4|bus~=13
102         sag_4_q1(bus) = (1-z_bus(4,bus)/z_bus(bus,bus))*v(bus);
103         sag_13_q1(bus) = (1-z_bus(13,bus)/z_bus(bus,bus))*v(bus);
104     %end
105 end
106
107 % sag magnitudes
108 sag_4_q1 = abs(sag_4_q1);
109 sag_13_q1 = abs(sag_13_q1);
110 %% q2a: sags (lines 4-5 and 6-13 are open)
111
112 % forming the y,z buses
113 [y_bus_q2, linedata_q2, z_bus_q2] = find_y_z_bus('line_data_q2.▼
xlsx','gen_reactances.xlsx');
114
115 %faults
116 sag_4_q2 = zeros(14,1);
117 sag_13_q2 = zeros(14,1);
118 for bus=1:14
119     %if bus~=4|bus~=13
120         sag_4_q2(bus) = (1-z_bus_q2(4,bus)/z_bus_q2(bus,bus))*v(bus);
121         sag_13_q2(bus) = (1-z_bus_q2(13,bus)/z_bus_q2(bus,bus))*v(bus);
122     %end
123 end
124 % sag magnitudes
125 sag_4_q2 = abs(sag_4_q2);
126 sag_13_q2 = abs(sag_13_q2);
127 %% q2b: sags (no x3 and x5)
128
129 [y_bus_q3, linedata_q3, z_bus_q3] = find_y_z_bus('line_data.▼
xlsx','gen_reactances_q2');
130
131 % faults
132 sag_4_q3 = zeros(14,1);
133 sag_13_q3 = zeros(14,1);
134 for bus=1:14
135     %if bus~=4|bus~=13
136         sag_4_q3(bus) = (1-z_bus_q3(4,bus)/z_bus_q3(bus,bus))*v(bus);
137         sag_13_q3(bus) = (1-z_bus_q3(13,bus)/z_bus_q3(bus,bus))*v(bus);
138     %end
139 end
140 % sag magnitudes
141 sag_4_q3 = abs(sag_4_q3);
142 sag_13_q3 = abs(sag_13_q3);
143 %% q4: no of sags
144
```

```
145 d = linedata(:,6); %line distance
146 frequency_100 = linedata(:,7); % faults/100km/yr
147
148 frequency_d = frequency_100.*d./100;
149 nodes = linedata(:,1:2);
150 [r,c] = size(nodes);
151 % calculating average sags
152 average_sags_4 = zeros(r,3); % will be of the same size as the nodes matrix
153 average_sags_4(:,1:2) = nodes;
154 average_sags_13 = zeros(r,3); % will be of the same size as the nodes matrix
155 average_sags_13(:,1:2) = nodes;
156
157 % sags at lines for bus 4
158 for k=1:r
159     from_node = linedata(k,1)
160     to_node = linedata(k,2)
161     sag_4_q1(linedata(k,1))
162     sag_4_q1(linedata(k,2))
163     sag1 = sag_4_q1(from_node);
164     sag2 = sag_4_q1(to_node);
165     average_sags_4(k,3) = 0.5*(sag1+sag2);
166 end
167
168 % sags at lines for bus 13
169 for k=1:r
170     sag1 = sag_13_q1(nodes(k,1));
171     sag2 = sag_13_q1(nodes(k,2));
172     average_sags_13(k,3) = 0.5*(sag1+sag2);
173 end
174
175 frequency_table = [nodes average_sags_4(:,3) average_sags_13(:,3) frequency_d];
176 f_sag_4 = 0;
177 f_sag_13 = 0;
178 % frequency of sag under 40%=0.4 pu
179 [r,~] = size(frequency_table);
180 for k=1:r
181     current_ave_sag_4 = frequency_table(k,3)
182     if current_ave_sag_4<0.4
183         frequency = frequency_table(k,5)
184         f_sag_4 = f_sag_4 + frequency
185     end
186     current_ave_sag_13 = frequency_table(k,4)
187     if current_ave_sag_13<0.4
188         frequency = frequency_table(k,5)
189         f_sag_13 = f_sag_13 + frequency_table(k,5)
190     end
191 end
192
193 %% q4: Bar chart
```

```
194 clc;
195
196 % creating chart
197 chart_intervals = 0:0.1:1; % each number represents an upper bound for an interval
198 chart_intervals = chart_intervals';
199 chart = [chart_intervals zeros(length(chart_intervals),2)];
200 [r,~] = size(chart);
201
202 % filling the chart
203 % sags at bus 4
204 for k=1:length(sag_4_q1)
205     for index=2:r
206         l_bound = chart(index-1,1)
207         u_bound = chart(index,1)
208         current_sag = sag_4_q1(k)
209         if l_bound<sag_4_q1(k) && sag_4_q1(k)<u_bound
210
211             chart(index,2) = chart(index,2)+1;
212         end
213     end
214 end
215 % sags at bus 13
216 for k=1:length(sag_13_q1)
217     for index=2:r
218         if chart(index-1,1)<sag_13_q1(k)&&sag_13_q1(k)<chart(index,1)
219             chart(index,3) = chart(index,3)+1;
220         end
221     end
222 end
223 clc;
224 chart = chart(2:end,:);
225
226 % barplots
227 figure(1)
228 bar(chart(:,1),chart(:,2))
229 alpha(.3)
230
231 hold on
232 bar(chart(:,1),chart(:,3))
233 alpha(.3)
234
235 legend('Bus 4 sags','Bus 13 sags')
236
237 %% q6 new generators
238 clc;
239 [y_bus_q6, linedata_q6, z_bus_q6] = find_y_z_bus('line_data.xlsx', 'gen_reactances_q6.xlsx');
240 % faults
```

```
241 sag_4_q6 = zeros(14,1);
242 sag_13_q6 = zeros(14,1);
243 for bus=1:14
244     %if bus~=4|bus~=13
245         sag_4_q6(bus) = (1-z_bus_q6(4,bus)/z_bus_q6(bus,bus))*v(bus);
246         sag_13_q6(bus) = (1-z_bus_q6(13,bus)/z_bus_q6(bus,bus))*v(bus);
247     %end
248 end
249 % sag magnitudes
250 sag_4_q6 = abs(sag_4_q6);
251 sag_13_q6 = abs(sag_13_q6);
252 %% Plots
253
254 % q1
255 figure(2)
256 bar(sag_4_q1)
257 hold on
258 bar(sag_13_q1)
259 alpha(0.4)
260 legend('Bus 4 sags','Bus 13 sags')
261
262 % q2a
263 figure(3)
264
265 bar(sag_4_q2)
266 hold on
267 bar(sag_13_q2)
268 alpha(0.4)
269 legend('Bus 4 sags','Bus 13 sags')
270 title('lines opened')
271
272 % q2b
273 figure(4)
274
275 bar(sag_4_q3)
276 hold on
277 bar(sag_13_q3)
278 alpha(0.4)
279 title('generators disconnected')
280 legend('Bus 4 sags','Bus 13 sags')
281
282 % q6
283 figure(5)
284 bar(sag_4_q6)
285 hold on
286 bar(sag_13_q6)
287 alpha(0.4)
288 title('Sags after addition of generators')
289 legend('Bus 4 sags','Bus 13 sags')
```

