I was not quite satisfied with one example, so I've split and merged two examples into one.

1. This problem is the analysis of the pole-splitting and right-half-plane zero cancellation of a standard two stage structure. The beginning of the mathematical derivation is shown on page 2. To ensure correctness, the following transfer function was obtained from an online source:

\[
\frac{V_{out}(s)}{V_{in}(s)} = \frac{a\{1 - s[(C_c/g_{mnl}) - R_cC_c]\}}{1 + bs + cs^2 + ds^3}
\]

where:
- \(a = g_{mnl}R_cR_H\)
- \(b = (C_i + C_c)R_H + (C_i + C_c)R_k + g_{mnl}R_HR_cC_c + R_cC_c\)
- \(c = [R_HR_k(C_iC_H + C_iC_f + C_fC_H) + R_cC_c(R_fC_f + R_fC_H)]\)
- \(d = R_HR_kR_cC_fC_H\)

Pole location was observed as \(R_c\) and \(C_c\) were changed. As the value of the compensation capacitor increases, pole-splitting occurs, and the two poles move further apart. This is shown in Figure 1 with the directions of the arrows. The poles closer to zero are difficult to see in this image because they are grouped very close to zero with respect to the scale of the real axis. The zero also moves in the imaginary axis as the value of \(C_c\) increases.

![Pole-Zero Map](image)

*Figure 1: Pole-Zero Map for sweep of \(C_c\)*

With a fixed value of \(C_c\), the value of the nulling resistor \(R_c\) was swept. This is shown in Figure 2. As the value of \(R_c\) approaches \(1/gm2\) (50), the zero moves to the right. At 50, the zero goes to infinity. After 50, the zero switches to the left half plane and moves in towards zero.

---

\[ Y_1 = \frac{1}{R_1} + sC_1 \quad Y_2 = \frac{1}{R_2} + sC_2 \quad z = R_2 + \frac{1}{sC_C} \]

**KCL @ node 0:**
\[ g_{m1}v_{in} + v_1 Y_1 = \frac{(v_1 - v_{out})}{2} = \frac{R_2 \cdot sC_C + 1}{sC_C} \]
\[ g_{m1}v_{in} = -v_1 Y_1 + \frac{1}{2}v_1 - \frac{1}{2}v_{out} \]
\[ g_{m1}v_{in} = v_1 \left( \frac{1}{2} - Y_1 \right) - \frac{1}{2}v_{out} \]

**KCL @ node 2:**
\[ \frac{v_1 - v_{out}}{2} = g_{m2}v_1 + Y_2 v_{out} \]
\[ v_1 \left( \frac{1}{2} - g_{m2} \right) = v_{out} \left( Y_2 + \frac{1}{2} \right) \]
\[ v_1 = \frac{Y_2 + \frac{1}{2}}{\frac{1}{2} - g_{m2}} v_{out} \]

\[ g_{m1}v_{in} = \left( \frac{Y_2 + \frac{1}{2}}{\frac{1}{2} - g_{m2}} \right) \left( \frac{1}{2} - Y_1 \right) v_{out} + - \frac{1}{2} v_{out} \]
\[ \frac{v_{out}}{v_{in}} = \frac{g_{m1}}{\left( \frac{Y_2 + \frac{1}{2}}{\frac{1}{2} - g_{m2}} \right) \left( \frac{1}{2} - Y_1 \right) - \frac{1}{4}} = \frac{g_{m1}}{\frac{1}{2} - g_{m2}} \]
\[ \frac{v_{out}}{v_{in}} = \left( \frac{1}{R_2 + sC_2} + \frac{sC_C}{R_2 \cdot sC_C + 1} \right) sC_C = \frac{sC_C}{R_2 \cdot sC_C + 1} \]
% Problem 1 Code
clear all; clc;

% Define circuit parameters
gm1 = 0.01;
gm2 = 0.02;
R1 = 100e3;
R2 = 5e3;
C1 = 0.5e-12;
C2 = 2e-12;

% Set Rc=0 and sweep Cc
Rc = 0;
figure;
hold on;
a = gm1*gm2*R1*R2;
for Cc = logspace(-12,-6)
b = (C2+Cc)*R2+(C1+Cc)*R1+gm2*R1*R2*Cc+Rc*Cc;
c = (R1*R2*(C1*C2+Cc*C1+Cc*C2)+Rc*Cc*(R1*C1+R2*C2));
d = R1*R2*Rc*C1*C2*Cc;
A = tf([-a*(Cc/gm2-Rc*Cc) a],[d c b 1]);
pzmap(A);
end

% Set Cc=2e-12 and sweep Rc
Cc = 2e-12;
figure;
hold on;
a = gm1*gm2*R1*R2;
for Rc = 41:2:59
b = (C2+Cc)*R2+(C1+Cc)*R1+gm2*R1*R2*Cc+Rc*Cc;
c = (R1*R2*(C1*C2+Cc*C1+Cc*C2)+Rc*Cc*(R1*C1+R2*C2));
d = R1*R2*Rc*C1*C2*Cc;
A = tf([-a*(Cc/gm2-Rc*Cc) a],[d c b 1]);
pzmap(A);
end

Figure 2: Pole-Zero Map for sweep of Rc
As the feedback is increased two poles approach one another before rapidly moving away.

Problem_2.m

iteration_count = 5000;
B = linspace(0,1,iteration_count);

%Constants
a0 = 1e3;
p1 = 1e3;
p2 = 1e5;
p3 = 1e6;

s = tf('s');

%Create a figure and begin adding data. Retain working plot.
figure;
hold on;
closed_loop = (a0) / ((1 + s/p1) * (1 + s/p2) * (1 + s/p3));

rlocus(closed_loop, B);

hold off;
The “%Overshoot” column has extra 100% (the step), thus the real %Overshoot would be 100 subtracted from that number.

Similarly, “dB Magnitude” is just showing linear magnitude. “1.0” would be no peaking, and the number greater than 1 should be converted to dB peaking.

In short, despite the flaws in the tables, I like the plots.

<table>
<thead>
<tr>
<th>Phase Margin</th>
<th>( P_2 / (a_0 P_1) ) [e-12]</th>
<th>%Overshoot</th>
<th>dB Magnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>39.9293</td>
<td>0.5365</td>
<td>129.04</td>
<td>1.4672</td>
</tr>
<tr>
<td>45.0837</td>
<td>0.7150</td>
<td>123.09</td>
<td>1.3050</td>
</tr>
<tr>
<td>50.0127</td>
<td>0.9134</td>
<td>118.1</td>
<td>1.1911</td>
</tr>
<tr>
<td>55.0267</td>
<td>1.1713</td>
<td>113.24</td>
<td>1.0984</td>
</tr>
<tr>
<td>60.1005</td>
<td>1.5086</td>
<td>108.68</td>
<td>1.0309</td>
</tr>
<tr>
<td>65.0150</td>
<td>1.9451</td>
<td>104.7</td>
<td>1.0004</td>
</tr>
<tr>
<td>69.986</td>
<td>2.5800</td>
<td>101.45</td>
<td>1.0</td>
</tr>
<tr>
<td>75.0252</td>
<td>3.6116</td>
<td>99.82</td>
<td>1.0</td>
</tr>
<tr>
<td>79.9841</td>
<td>5.5758</td>
<td>99.56</td>
<td>1.0</td>
</tr>
</tbody>
</table>
In instructor office hours I was having problems getting P4 to work. It turns out that in problem 3, I had set p2 to be in the e-6 range, and not e6. This would generate the correct phase margins but when gain was lowered and p1 shifted, problem 4 would not work.

Adjusting Problem 3’s pole 2 to the correct scale in e6 took a bit of time and as a result I was unable to adjust the exact p2's into this problem.

He recommended leaving a small paragraph about this as I would not be able to update all the information in time.

In short - P3 was solved twice. This cost a lot of time and was not able to update the pole locations to the exact ones used. There is a small variation between each pole, but roughly the same phase.
Overshoot and peaking is slightly smaller in a closed loop response.
Problem 3.m

Problem 4 uses the same process with changed a0 and p1.

close all;
phase_margin_appx_count = 500; %How many test points to gather initial
% phase information VS p2

p2_appx = linspace( 1e5 , 1e7,phase_margin_appx_count);

% Specific phase margins we are looking for.
deg_step = 5;
phase_margins_for_test = linspace(40, 80, (80-40)/deg_step + 1);
legendCell = cellstr(num2str(phase_margins_for_test', 'N-%d'));
% Constants
a0 = 1e6;
p1 = 1;
p2 = 1; % To be varied

s = tf('s'); % Activates transfer function

for i=1:length(p2_appx)
    p2 = p2_appx(i);
    open_loop = ( a0 ) / ( ( 1 + s/p1 ) * ( 1 + s/p2 ) );
    [Gm,Pm] = margin(open_loop);
    phase_margins_calc(i) = Pm;
end

% We now extract best index values for each case.
for i=1:length(phase_margins_for_test)
    [~, idx] = min( abs(phase_margins_calc - phase_margins_for_test(i)) );
    closest_x = p2_appx( idx );
    p2_for_phase(i) = closest_x;
    phase_exact_for_p2(i) = phase_margins_calc(idx);
end

% Debug output to ensure valid phases
phase_exact_for_p2
p2_for_phase

hold on;
for i=1:length(p2_for_phase)
    p2 = p2_for_phase(i);
    open_loop = ( a0 ) / ( ( 1 + s/p1 ) * ( 1 + s/p2 ) );
    b = 1; % Feedback factor
    closed_loop = feedback(open_loop, b);
    %- P2 / (a0*p1) -- TABLE INFO
    p2_a0p1(i) = (p2) / (a0 * p1);
%-- Max overshoot --%
figure(1);
step(closed_loop); % Display step response
[y, t] = step(closed_loop);
p2_step_response_max(i) = max(y);

%-- Freq Response --%
figure(2);
hold on;
margin(closed_loop);
[y, t] = bode(closed_loop);
p2_peak_db(i) = max(y);

%-- Open loop bode --%
figure(3);
hold on;
margin(open_loop);

end

figure(1);
legend(legendCell);
title('Step Response');
figure(2);
legend(legendCell);
title('Closed Bode Plot');
figure(3);
legend(legendCell);
title('Open Bode Plot');
hold off;

phase_exact_for_p2

p2_a0p1

p2_step_response_max

p2_peak_db