LAB THREE: MICROSTRIP TRANSMISSION LINE

Electromagnetic Engineering ELG 3106A

Nick Cardamone
ncard024@uottawa.ca
300060019
Nov 13, 2020

Introduction

In this lab, we will be studying the linear microstrip line response to changes in physical dimensions. A microstrip line is a transmission line with a conductor on the top and a ground plane on the bottom, which is separated by an insulator, usually a PCB. A picture of the microstrip line can be seen below.

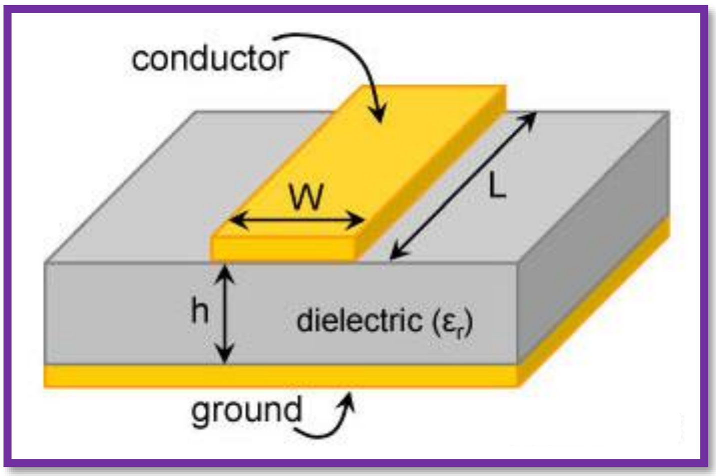

Figure 1: The Microstrip Transmission Line

This lab's objective is to evaluate a set of approximations for the microstrip line against a java app.

Theory

In order to describe the microstrip line, we will need to use some numerical approximations. For those approximations, we will need some additional parameters as follows.

$$x = 0.56\left(\frac{\varepsilon_r - 0.9}{\varepsilon_r + 3}\right)^{0.05}$$

$$y = 1 + 0.02\ln\left(\frac{s^4 + 0.00037s^2}{s^4 + 0.43}\right) + 0.05\ln\left(1 + 0.00017s^3\right)$$

$$t = \left(30.67/s\right)^{0.75}$$

Now that we have those three parameters, we can define our variables of interest. First, there is the width to thickness ratio s

$$s = \frac{w}{h}$$

Next, we have an effective permittivity ε_{eff}

$$u_p = \frac{c}{\sqrt{\varepsilon_{eff}}}$$

$$\varepsilon_{\text{eff}} = \frac{\varepsilon_r + 1}{2} + \left(\frac{\varepsilon_r - 1}{2}\right)\left(1 + \frac{10}{s}\right)^{-xy}$$

From there, we define the line impedance of the microstrip transmission line (Z_0) as

$$Z_0 = \frac{60}{\sqrt{\varepsilon_{\text{eff}}}} \ln\left(\frac{6 + (2\pi - 6)e^{-t}}{s} + \sqrt{1 + 4/s^2}\right)$$

It is important to note that all parameterizations are numerically determined. These approximations were first presented in a German paper about 100 years ago.

If we are instead given a target line impedance and need to design the width to thickness ratio. There is another set of approximations and parameterizations. They are as follows.

(a) For $Z_0 \leq (44 - 2\varepsilon_r)\,\Omega$:

$$s = \frac{w}{h} = \frac{2}{\pi}\left\{(q-1) - \ln(2q-1) + \frac{\varepsilon_r - 1}{2\varepsilon_r}\left[\ln(q-1) + 0.29 - \frac{0.52}{\varepsilon_r}\right]\right\} \text{ where } q = \frac{60\pi^2}{Z_0\sqrt{\varepsilon_r}};$$

(b) For $Z_0 \geq (44 - 2\varepsilon_r)\,\Omega$

$$s = \frac{w}{h} = \frac{8e^p}{e^{2p} - 2} \text{ where } p = \sqrt{\frac{\varepsilon_r + 1}{2}}\left(\frac{Z_0}{60}\right) + \left(\frac{\varepsilon_r - 1}{\varepsilon_r + 1}\right)\left(0.23 + \frac{0.12}{\varepsilon_r}\right).$$

It is important to note that approximations a and b are valid to within 2%

Simulation Results

Part 1

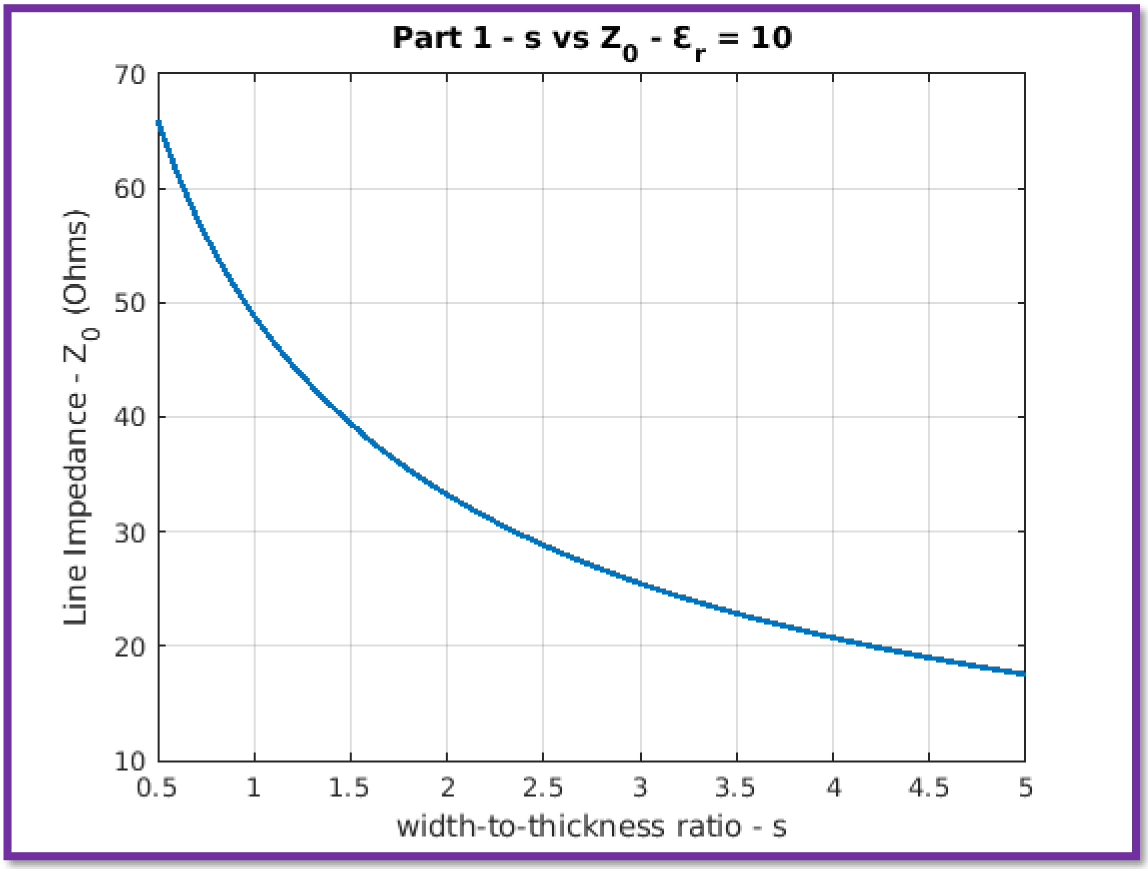

Figure 2: MATLAB, Width to Thickness Ratio vs Line Impedance

Part 2

Table 1: Java App Results

Width-to-thickness ratio (s)	Line Impedance (Z_0) (Ω)
0.5	65.0
1	48.2
1.2	44.2
1.5	39.2
2	33.2
3	25.4
4	20.7
5	17.5

Part 3

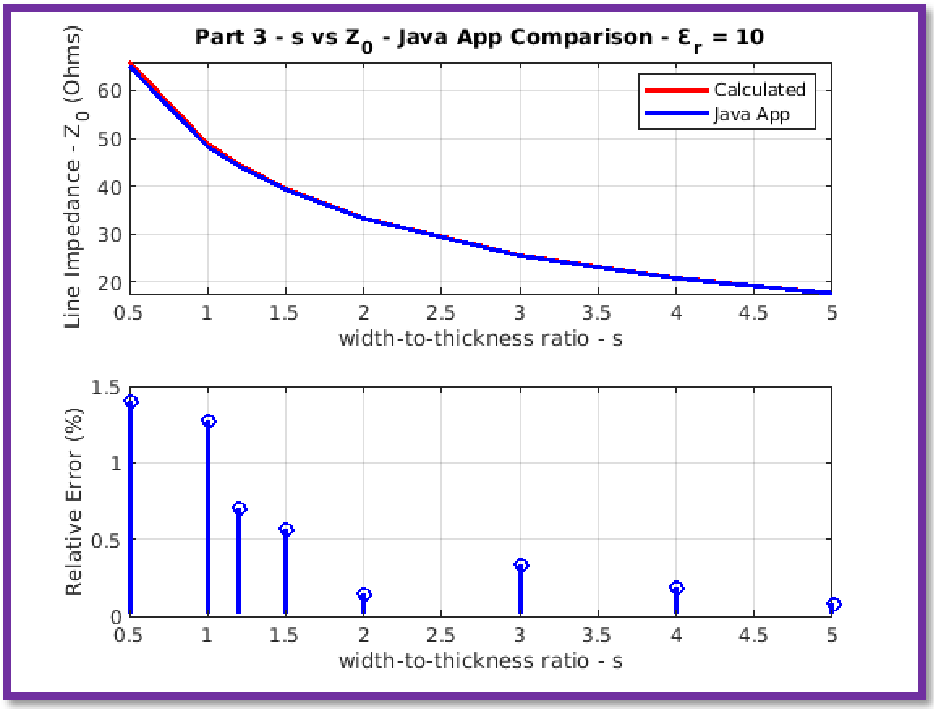

Figure 3: Java App and MATLAB Comparison

Part 4

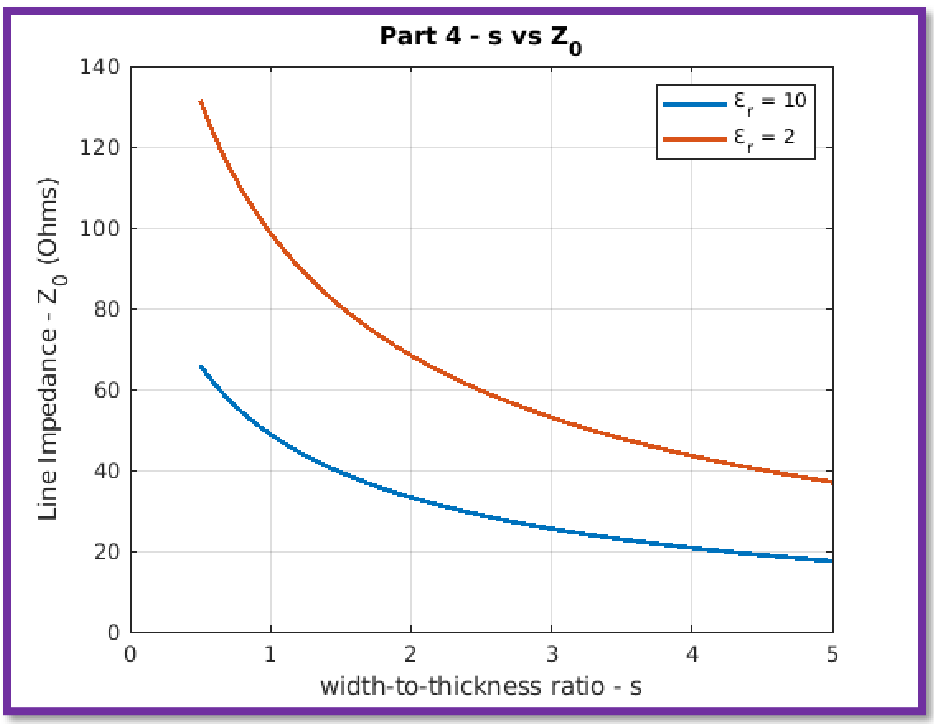

Figure 4: MATLAB, Width to Thickness Ratio vs Line Impedance, Varying ϵ_r

Part 5

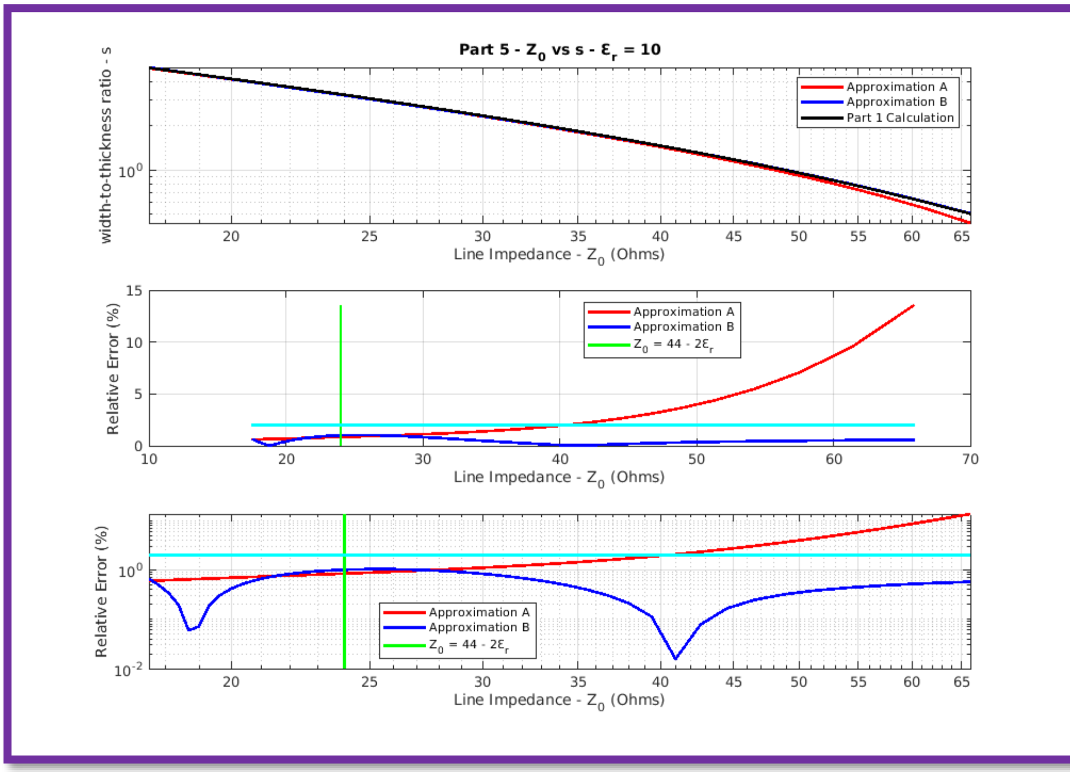

Figure 5: Comparing Approximations A and B, $\varepsilon_r = 10$

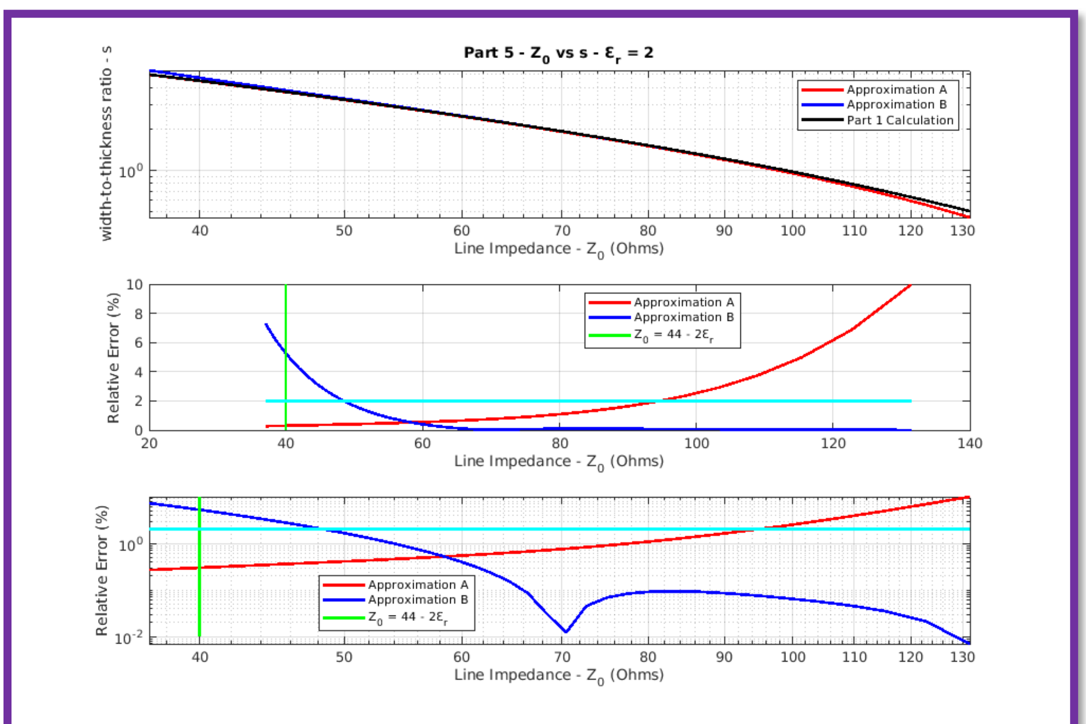

Figure 6: Comparing Approximations A and B, $\varepsilon_r = 2$

Part 6

Table 2: Interval where relative error of Z_0 is less than 2%

ε_r	Approximation A	Approximation B
10	[10, 40] (Ω)	[10, 66] (Ω)
2	[38.5, 92] (Ω)	[49, 130] (Ω)

Software Explanation

The MATLAB code for this lab is rather simple and self-explanatory. Hence, this explanation will be brief. The code starts by initializing an array for s. Then the parameterizations are calculated. Then the quantities of interest are calculated and plotted. The MATLAB code can be found in Appendix I.

Discussion

Part 1

Part one is relatively straight forward. No unexpected results occurred.

Part 2

In part two, the java app gives a frequency parameter. After varying the frequency, it was determined that the frequency does not affect the line impedance of the microstrip line for the java app. Although, this does not happen in reality, though. The characteristic line impedance is a function of frequency for any transmission line as there are parasitic capacitances and inductances. Some explanations for the java app ignoring the frequency dependence are: assuming a lossless linear line or that the height of the conductors are neglected

Part 3

In part three, the relative errors were all better than the stated 2%. Interestingly, the error got better as the width to thickness ratio increased. As to why that happens, I do not know. I do not have the source code for the java app. Nor do I speak German, as the paper introducing this microstrip theory is written in German.

Part 4

Part four is relatively straight forward. No unexpected results occurred.

Part 5 and 6

For parts five and six, something interesting happened, the approximations did not even come close to their stated validity conditions. In figures five and six, the green line is the condition where each approximation should have crossed the 2% relative error line. In addition to that, for approximation b with a dielectric constant of 10, it is valid for all values of line impedance calculated. Why does this happen? I have no clue, as I do not know where these equations come from. It is also important to note that both approximations a and b can be valid at the same time when using a 2% relative error as a benchmark for validity.

Conclusion

In conclusion, there were some surprising results from this lab. First, that frequency is neglected in the java app. Second, that the approximation conditions are not valid, finally, that both approximations a and b can b valid at the same time, using a 2% relative error as a benchmark. In these presented equations, frequency of operation does not appear. Since there are parasites in every line, there should be a non-flat frequency response. Either these equations are meant to be employed well below any poles/ zeros, or these equations are meant for the resonant frequency, where the capacitances and inductances cancel themselves out.

Note: I am not including a table with all the calculated values. There are 25 vectors; most of them have 46 elements. Inclusion of such a table is unnecessary madness. If the calculated values are truly desired, the MATLAB code is below.

Appendix I: MATLAB Code

```
%Lab3 EMAG


%Part 1


%Given
s = 0.5:0.1:5;
epsilon_r = 10


%Find


x = (epsilon_r-0.9)/(epsilon_r + 3);
x = x^0.05;
x = 0.56*x;


y = s.^4+0.00037*s.^2;
y = y./(s.^4 + 0.43);
y = 0.02*log(y);
y = y + 0.05*log(1+0.00017*s.^3);
y = y+1;


epsilon_eff = 1+10./s;
epsilon_eff = epsilon_eff.^(-x.*y);
epsilon_eff = epsilon_eff*0.5*(epsilon_r-1);
epsilon_eff = epsilon_eff + 0.5*(epsilon_r+1);


t = 30.67./s;
t = t.^0.75;
```

```
Z0 = sqrt(1+4./(s.^2));
Z0 = Z0 + (6+(2*pi-6).*exp(-t))./s;
Z0 = 60*log(Z0)./sqrt(epsilon_eff);


set(gcf, 'Visible', 'on')
plot(s, Z0, 'LineWidth', 2)
xlabel('width-to-thickness ratio - s')
ylabel('Line Impedance - Z_0 (Ohms)')
title('Part 1 - s vs Z_0 - Ɛ_r = 10')
grid on
```

```
%Part 2
sApp = [0.5 1 1.2 1.5 2 3 4 5];
Z0App = [65, 48.2, 44.2, 39.2, 33.2, 25.4, 20.7, 17.5];
```

```
%Part 3
yApp = sApp.^4+0.00037*sApp.^2;
yApp = yApp./(sApp.^4 + 0.43);
yApp = 0.02*log(yApp);
yApp = yApp + 0.05*log(1+0.00017*sApp.^3);
yApp = yApp+1;


epsilon_effApp = 1+10./sApp;
epsilon_effApp = epsilon_effApp.^(-x.*yApp);
epsilon_effApp = epsilon_effApp*0.5*(epsilon_r-1);
epsilon_effApp = epsilon_effApp + 0.5*(epsilon_r+1);


tApp = 30.67./sApp;
tApp = tApp.^0.75;


Z0AppC = sqrt(1+4./(sApp.^2));
Z0AppC = Z0AppC + (6+(2*pi-6).*exp(-tApp))./sApp;
Z0AppC = 60*log(Z0AppC)./sqrt(epsilon_effApp);
```

```
Z0Diff = (1-Z0App./Z0AppC)*100;
```

```
set(gcf, 'Visible', 'on')
subplot(2,1,1)
plot(sApp,Z0AppC,'r', 'Linewidth', 2)
hold on
plot(sApp, Z0App,'b', 'Linewidth', 2)
grid on
xlabel('width-to-thickness ratio - s')
ylabel('Line Impedance - Z_0 (Ohms)')
title('Part 3 - s vs Z_0 - Java App Comparison - Ɛ_r = 10')
legend('Calculated', 'Java App')


subplot(2,1,2)
stem(sApp, Z0Diff,'b', 'Linewidth', 2)
grid on
xlabel('width-to-thickness ratio - s')
ylabel('Relative Error (%)')
```

```
%Part 4


epsilon_r = 2;


%Find


x = (epsilon_r-0.9)/(epsilon_r + 3);
x = x^0.05;
x = 0.56*x;


y = s.^4+0.00037*s.^2;
y = y./(s.^4 + 0.43);
y = 0.02*log(y);
y = y + 0.05*log(1+0.00017*s.^3);
y = y+1;


epsilon_eff2 = 1+10./s;
epsilon_eff2 = epsilon_eff2.^(-x.*y);
epsilon_eff2 = epsilon_eff2*0.5*(epsilon_r-1);
epsilon_eff2 = epsilon_eff2 + 0.5*(epsilon_r+1);


t = 30.67./s;
t = t.^0.75;
```

```
Z04 = sqrt(1+4./(s.^2));
Z04 = Z04 + (6+(2*pi-6).*exp(-t))./s;
Z04 = 60*log(Z04)./sqrt(epsilon_eff2);


set(gcf, 'Visible', 'on')
plot(s, Z0, 'LineWidth', 2)
hold on
plot(s, Z04, 'LineWidth', 2)
xlabel('width-to-thickness ratio - s')
ylabel('Line Impedance - Z_0 (Ohms)')
title('Part 4 - s vs Z_0')
grid on
legend('Ɛ_r = 10', 'Ɛ_r = 2')
```

```
%Part 5


epsilon_r = 10
%Approx A


q = Z0*sqrt(epsilon_r);
q = 60*pi^2./q;


sA = log(q-1) + 0.29 - 0.52/epsilon_r;
sA = sA.*(epsilon_r-1)./(2*epsilon_r);
sA = q-1 -log(2*q-1)+sA;
sA = sA*2/pi;


%Approx B


p = 0.23 + 0.12./epsilon_r;
p = (epsilon_r-1)./(epsilon_r+1).*p;
p = sqrt(epsilon_r*0.5+0.5).*Z0/60+p;


sB = exp(2*p) - 2;
sB = 8*exp(p)./sB;




sADiff = 100*(1-sA./s);
sBDiff = 100*(1-sB./s);
```

```
xCondition = (44 - 2*epsilon_r)*ones(length(sADiff));
yCondition = linspace(0.01, max(sADiff), length(sADiff));
twoPercent = 2*ones(length(Z0));


hold off
set(gcf, 'Visible', 'on')
subplot(3,1,1)
loglog(Z0, sA, 'r', 'LineWidth', 2)
hold on
loglog(Z0, sB, 'b', 'LineWidth', 2)
loglog(Z0, s, 'k', 'LineWidth', 2)




grid on
xlabel('Line Impedance - Z_0 (Ohms)')
ylabel('width-to-thickness ratio - s')
title('Part 5 - Z_0 vs s - Ɛ_r = 10')
legend('Approximation A', 'Approximation B', 'Part 1 Calculation')


subplot(3,1,2)
plot(Z0, abs(sADiff), 'r', 'LineWidth', 2)
hold on
plot(Z0, abs(sBDiff), 'b', 'LineWidth', 2)
plot(xCondition, yCondition, 'g-', 'LineWidth', 2)
plot(Z0, twoPercent, 'c-', 'LineWidth', 2)
grid on
xlabel('Line Impedance - Z_0 (Ohms)')
ylabel('Relative Error (%)')
legend('Approximation A', 'Approximation B', 'Z_0 = 44 - 2Ɛ_r', 'Location',
'best')


subplot(3,1,3)
loglog(Z0, abs(sADiff), 'r', 'LineWidth', 2)
hold on
loglog(Z0, abs(sBDiff), 'b', 'LineWidth', 2)
loglog(xCondition, yCondition, 'g-', 'LineWidth', 2)
loglog(Z0, twoPercent, 'c-', 'LineWidth', 2)
grid on
xlabel('Line Impedance - Z_0 (Ohms)')
ylabel('Relative Error (%)')
legend('Approximation A', 'Approximation B', 'Z_0 = 44 - 2Ɛ_r', 'Location',
'best')
```

```
epsilon_r = 2;


%Approx A


q = Z04*sqrt(epsilon_r);
q = 60*pi^2./q;


sA = log(q-1) + 0.29 - 0.52/epsilon_r;
sA = sA.*(epsilon_r-1)./(2*epsilon_r);
sA = q-1 -log(2*q-1)+sA;
sA = sA*2/pi;


%Approx B


p = 0.23 + 0.12./epsilon_r;
p = (epsilon_r-1)./(epsilon_r+1).*p;
p = sqrt(epsilon_r*0.5+0.5).*Z04/60+p;


sB = exp(2*p) - 2;
sB = 8*exp(p)./sB;


sADiff = 100*(1-sA./s);
sBDiff = 100*(1-sB./s);


xCondition = (44 - 2*epsilon_r)*ones(length(sADiff));
yCondition = linspace(0.01, max(sADiff), length(sADiff));
twoPercent = 2*ones(length(Z04));


hold off
set(gcf, 'Visible', 'on')
subplot(3,1,1)
loglog(Z04, sA, 'r', 'LineWidth', 2)
hold on
loglog(Z04, sB, 'b', 'LineWidth', 2)
loglog(Z04, s, 'k', 'LineWidth', 2)




grid on
```

```
xlabel('Line Impedance - Z_0 (Ohms)')
ylabel('width-to-thickness ratio - s')
title('Part 5 - Z_0 vs s - Ɛ_r = 2')
legend('Approximation A', 'Approximation B', 'Part 1 Calculation')


subplot(3,1,2)
plot(Z04, abs(sADiff), 'r', 'LineWidth', 2)
hold on
plot(Z04, abs(sBDiff), 'b', 'LineWidth', 2)
plot(xCondition, yCondition, 'g-', 'LineWidth', 2)
plot(Z04, twoPercent, 'c-', 'LineWidth', 2)
grid on
xlabel('Line Impedance - Z_0 (Ohms)')
ylabel('Relative Error (%)')
legend('Approximation A', 'Approximation B', 'Z_0 = 44 - 2Ɛ_r', 'Location',
'best')


subplot(3,1,3)
loglog(Z04, abs(sADiff), 'r', 'LineWidth', 2)
hold on
loglog(Z04, abs(sBDiff), 'b', 'LineWidth', 2)
loglog(xCondition, yCondition, 'g-', 'LineWidth', 2)
loglog(Z04, twoPercent, 'c-', 'LineWidth', 2)
grid on
xlabel('Line Impedance - Z_0 (Ohms)')
ylabel('Relative Error (%)')
legend('Approximation A', 'Approximation B', 'Z_0 = 44 - 2Ɛ_r', 'Location',
'best')
```