

# Non-Destructive Testing Analysis Using Oersted

## ABSTRACT

The purpose of this report is to compare the results of IES's Oersted software with the analytical results of the impedance of a coil above a two-conductor plane, as given in a publication by Dodd, Deeds and Luquire [1]. Dodd et. al. derives an equation for the impedance based on the vector potential of the coil configuration. Oersted calculates a full-wave solution to the configuration using the boundary element method (BEM). The BEM solves for equivalent sources based on the geometry and the boundary conditions, and then calculates the fields due to these sources. Impedance values are determined from these sources. There is excellent agreement between the results produced by the analytic solution and Oersted.

### Integrated Engineering Software - Website Links

[Home](#)[Products](#)[Support](#)[Technical Papers](#)

*"Page Down" or use scroll bars to read the article*

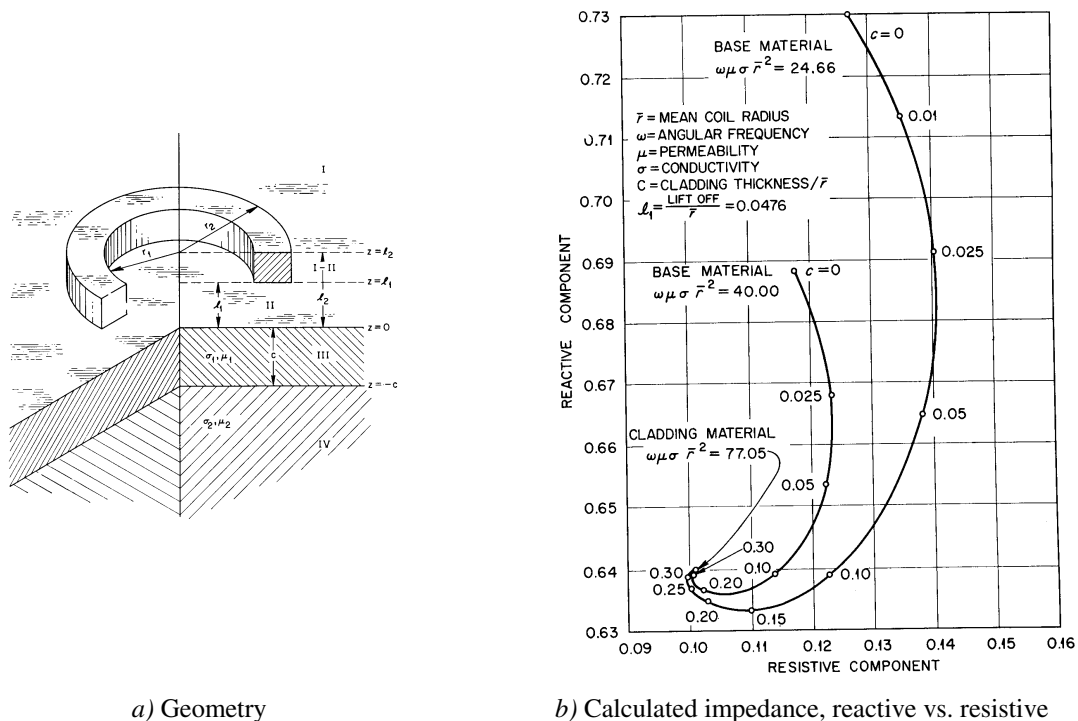
# Non-Destructive Testing Analysis Using Oersted

By James Dietrich  
 For Integrated Engineering Software  
 January 2001

**Abstract** – The purpose of this report is to compare the results of IES’s Oersted<sup>®</sup> software with the analytical results of the impedance of a coil above a two-conductor plane, as given in a publication by Dodd, Deeds & Luquire [1]. Dodd *et. al.* derives an equation for the impedance based on the vector potential of the coil configuration. Oersted<sup>®</sup> calculates a full-wave solution to the configuration using the boundary element method (BEM). The BEM solves for equivalent sources based on the geometry and the boundary conditions, and then calculates the fields due to these sources. Impedance values are determined from these sources. There is excellent agreement between the results produced by the analytic solution and Oersted<sup>®</sup>.

## Overview

The Dodd paper considers six geometries of a coil with rectangular cross-section in the presence of multiple conductors. This report is concerned with the case where the coil is above a two-conductor surface, that is, a base conductor has been clad (coated) with another conducting material. The cladding is  $c$  units thick, the coil is  $l_1$  units above the cladding surface, and the coil has an inner and outer radius of  $r_1$  &  $r_2$  and extends in height from  $l_1$  to  $l_2$ , as shown in Figure 1.



The formulations of the Dodd paper are based on the vector potential for a single loop of wire. Such vector potentials are superimposed to obtain the vector potential of a coil of rectangular cross-section. From the vector potential for a coil, the equations for various phenomena are derived. For the geometry shown in Figure 1a, the impedance of the coil is derived in equation (3.11) in the paper and is normalized by the mean coil radius. This impedance is normalized by the impedance of the coil in air (not in the proximity of conducting surfaces), and is plotted as reactance vs. resistance in Figure 1b.

The difficulty in reproducing the plot of Figure 1b, is not in programming the equations, but rather is due to the fact that we are not given the values for  $r_1$  &  $r_2$  and  $l_1$  &  $l_2$ , which have a dramatic impact on the resulting plot, as will be shown. First, however, let's look at the agreement between the analytic results and Oersted<sup>®</sup>.

Because the dimensions were not given, the values for  $r_1$  &  $r_2$  and  $l_1$  &  $l_2$ , were taken from the Oersted<sup>®</sup> database provided by a participant in this study, and are assumed to have been chosen arbitrarily or by the design requirements of a specific project. These dimensions are given in Table 1.

**Table 1**

Dimension	length, $\mu\text{m}$
$r_1$	381
$r_2$	$762 = 2 * r_1$
$l_1$	27.2034
$l_2$	327.2034

In order to reproduce the curves of Figure 1b, we must observe the following constraints, as listed on the Dodd graph, and shown in Table 2, where it is assumed that  $\mu = \mu_o$  for all materials.

**Table 2**

Property	Description	Value
1	Base 1, $\omega \mu \sigma_1 \bar{r}^2$	24.66
2	Base 2, $\omega \mu \sigma_2 \bar{r}^2$	40.00
3	Cladding, $\omega \mu \sigma_c \bar{r}^2$	77.05
4	$l_1 = \frac{\text{lift off}}{\bar{r}}$	0.0476
5	$c = \frac{\text{cladding thickness}}{\bar{r}}$	0 to 0.30

Given that frequency, permeability and mean radius are constant, one deduces immediately that the layer of cladding has a higher conductivity than either base material, with a ratio of  $\sigma_c/\sigma_b = 77.05/24.66$  for base 1 material and  $\sigma_c/\sigma_b = 77.05/40.00$  for base 2 material. Once one chooses values for  $r_1$  &  $r_2$ ,  $l_1$  is chosen by property 4 of Table 2 and  $l_2$  is chosen arbitrarily. One chooses either the operating frequency,  $\omega$ , or the conductivities (observing the  $\sigma_c/\sigma_b$  ratios) and properties 1 – 3 govern ones choice for the other variable.

Table 3 shows the comparison of results. The values labeled Dodd Curve are taken from the curves published in the paper. The calculated values are the results of putting the dimensions used in Oersted<sup>®</sup> into the equations given in the Dodd paper. Note the excellent agreement between the calculated and Oersted<sup>®</sup> results, which are also shown in Figure 2.

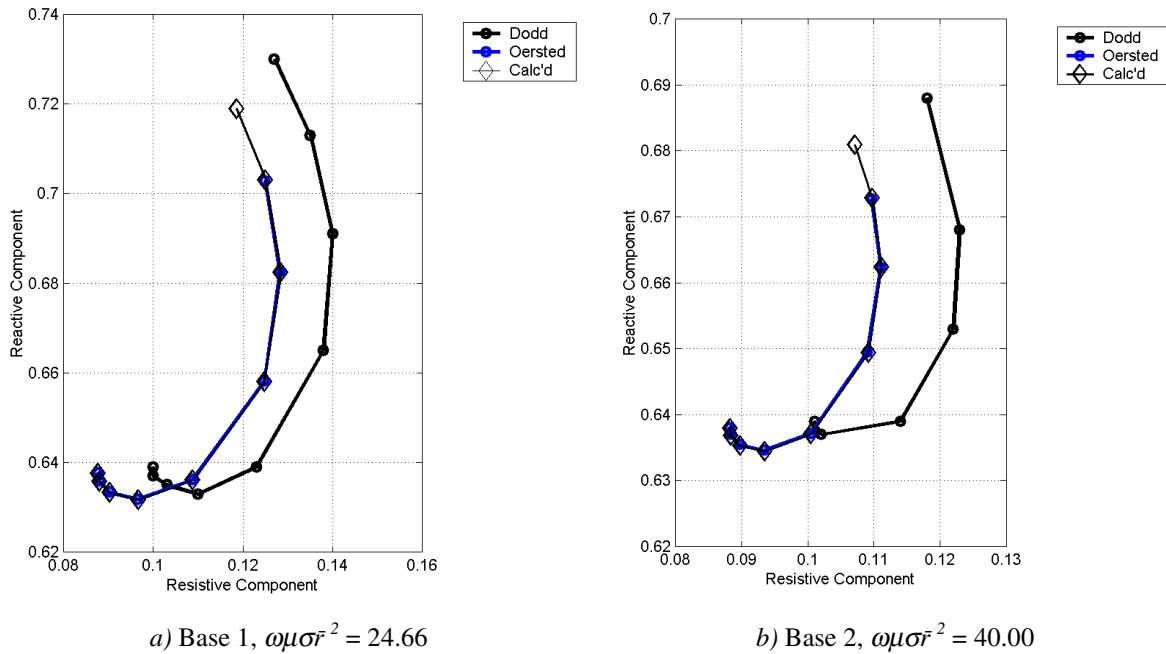
**Table 3** – Comparison of results.

Base 1, $\omega\mu\sigma\bar{r}^2 = 24.66$						
Dodd Curve			Oersted <sup>®</sup>		Calc'd Values	
c	R	X	R	X	R	X
0.000	0.127	0.730			0.119	0.720
0.010	0.135	0.713	0.125	0.703	0.125	0.703
0.025	0.140	0.691	0.128	0.682	0.128	0.682
0.050	0.138	0.665	0.125	0.658	0.125	0.658
0.100	0.123	0.639	0.109	0.636	0.109	0.636
0.150	0.110	0.633	0.097	0.632	0.097	0.632
0.200	0.103	0.635	0.090	0.633	0.090	0.633
0.250	0.100	0.637	0.088	0.636	0.088	0.636
0.300	0.100	0.639	0.088	0.638	0.088	0.638

Base 2, $\omega\mu\sigma\bar{r}^2 = 40.00$						
Dodd Curve			Oersted <sup>®</sup>		Calc'd Values	
c	R	X	R	X	R	X
0.000	0.118	0.688			0.107	0.681
0.010			0.110	0.673	0.110	0.673
0.025	0.123	0.668	0.111	0.662	0.111	0.662
0.050	0.122	0.653	0.109	0.650	0.109	0.649
0.100	0.114	0.639	0.100	0.637	0.101	0.637
0.150			0.093	0.635	0.094	0.634
0.200	0.102	0.637	0.090	0.635	0.090	0.635
0.250			0.088	0.637	0.088	0.637
0.300	0.101	0.639	0.088	0.638	0.088	0.638

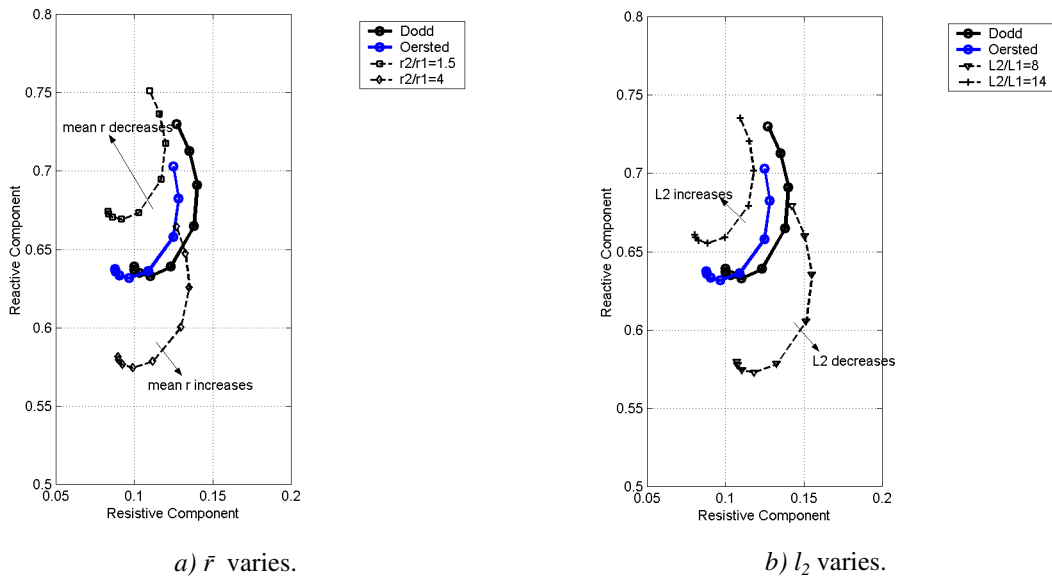
It turns out that the curves generated by the Dodd equations are not only dependent on  $\omega\mu\sigma\bar{r}^2$  and the lift-off,  $l_1$ , but also on the height of the coil,  $l_2-l_1$ . Also, changes in the value of the mean radius,  $\bar{r}$ , will displace the curve even though the product  $\omega\mu\sigma\bar{r}^2$  and the ratio  $l_1/\bar{r}$  are kept constant.

It should be pointed out that although the integral in the Dodd equation goes from zero to infinity, it was sufficient for the numeric integration to go from 1e-9 to 20. Zero produces a singularity, and a plot of the function showed that for  $\alpha > 10$ , the function is practically zero.



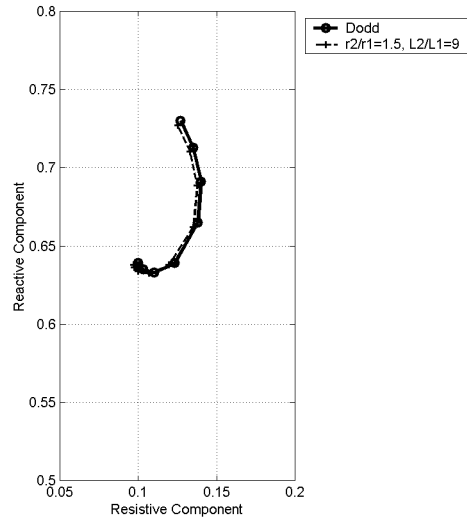
**Figure 2** – Comparison of Dodd’s curves with Oersted® results and results calculated using Dodd’s equations and the dimensions used in the Oersted® simulation.

The behavior of the impedance curves as  $\bar{r}$  varies or as  $l_2$  varies, is shown in Figure 3.



**Figure 3** – Movement of impedance curve as  $\bar{r}$  and  $l_2$  are varied.

One notices that for both  $\bar{r}$  decreasing ( $r_2/r_1$  decreases) and  $l_2$  increasing, the curve moves to the upper left, that is, resistance decreases and reactance increases. Also note that it scales smaller than the original curve. Likewise, the curve scales larger and moves to the lower right when either  $\bar{r}$  increases or  $l_2$  decreases. The curve scales less and does not move as dramatically to the right or left as a function of  $\bar{r}$ . In fact, as  $\bar{r}$  increases, the curve practically moves vertically downward. In order to get the curve produced by Oersted®, or the equations, to overlap the curve published by Dodd, the curve needs to be moved to the right and up slightly. This implies that  $l_2$  should decrease to translate the curve to the (lower) right, and then decrease  $\bar{r}$  to lift the curve vertically. These results are shown in Figure 4.



**Figure 4** –Location of curve for  $r_2/r_1 = 1.5$  and  $l_2/l_1 = 9$ .

It actually proves to be a difficult task to find the exact values to match the two curves. This is because the numeric integration over  $rJ_1(\alpha r)$  from  $r_1$  to  $r_2$  is not well-behaved, resulting in recursion limits being reached and warnings of singularities issued. However, the results of Figure 4 are quite convincing.

The intention of this part of the report has been to show the agreement between the results of Oersted<sup>®</sup> and the equations developed in the paper by Dodd *et. al.* Judging from the results in Figure 2 to Figure 4, the agreement is excellent when the same dimensions are used in both cases.

The next section will briefly cover the Matlab<sup>®</sup> scripts used to represent the equations.

## Matlab Files

The analytic solutions to Dodd's equations were produced using Matlab<sup>®</sup>. The m-files are listed in the appendix, and should also accompany this report on a disk with the Oersted<sup>®</sup> database. The m-files are listed in Table 4 with a description. A brief discussion on running the Matlab scripts is given.

Table 4

Filename	Description
ies1.m	Main program to calculate equation with Oersted <sup>®</sup> dimensions.
ies2.m	Main program to calculate equation while varying dimensions.
xBessJ1xa.m	The function $rJ_1(\alpha r)$
Jr2r1.m	Performs the integral of $1/\alpha^3 rJ_1(\alpha r)$ from $r_1$ to $r_2$
Zintgrnd.m	The function for the integrand in Dodd's eq. (3.11)
ZintgrndAIR.m	The function for the integrand of the coil in air in Dodd's eq. (3.11)
plotZ.m	Creates the impedance scatter plot for Base 1, $\omega\mu\sigma\bar{r}^2 = 24.66$
plotZ40.m	Creates the impedance scatter plot for Base 1, $\omega\mu\sigma\bar{r}^2 = 40.00$
findbest.m	To attempt to find the best $\bar{r}$ and $l_2$ to fit the original Dodd curve.
excise.m	removes NaNs from a column matrix (Dodd40.00 has NaNs for $c = 0.01, 0.15, \& 0.25$ .)
DataCmp.mat	Mat file with the Dodd & Oersted <sup>®</sup> results.

Below is an excerpt from ies1.m:

Script Line	Description
<code>if ~exist('c', 'var')</code>	If not already open,
<code>close all</code>	open a plot for
<code>load DataCmp</code>	comparison.
<code>plotZ,hold on</code>	
<code>end</code>	
	<b>constants in free space</b>
<code>c0 = 2.997924574E+08;</code>	speed of light
<code>u0 = 1.256637061E-06;</code>	permeability
<code>e0 = 8.854187853E-12;</code>	permittivity
<code>n = 1; % number of turns</code>	
<code>r1 = 3e-3; %381e-6;</code>	inner coil radius
<code>r2 = 2*r1;</code>	outer coil radius
<code>rbar = (r1+r2)/2; %5.715e-4;</code>	computes mean
<code>w = 2*pi*4e6;</code>	angular frequency
<code>%w = 24.66/(u0*cond1*rbar^2);</code>	
<code>cond1 = 77.05/(u0*w*rbar^2);</code>	conductivity of cladding
<code>cond2 = 24.66/(u0*w*rbar^2);</code>	conductivity of base
	forcing $\omega\mu\sigma\bar{r}^2$
<code>l1 = 0.0476*rbar;</code>	matching the lift-off height.

<code>l2 = 327.2034e-6; %l1*l2</code>	Arb. chosen l2
<code>%% normalize lengths</code>	normalize lengths
<code>r1 = r1/rbar;</code>	
<code>r2 = r2/rbar;</code>	
<code>l1 = l1/rbar;</code>	
<code>l2 = l2/rbar;</code>	
<code>Le = (l2-l1);</code>	
<code>%% -----</code>	
<code>alimlo = 1e-9; %lower limit of itegration</code>	set limits of integration
<code>alimhi = 20; %upper limit of itegration</code>	
<code>c = [0 .01 .025 .05 .1 .15 .2 .25 .3];</code>	normalized cladding thickness
<code>disp(' ')</code>	
<code>disp(' ')</code>	
<code>disp(['w*u0*cond1*rbar^2 =', num2str(w*u0*cond1*rbar^2)])</code>	display results showing that $\omega\mu\sigma\bar{r}^2$ is correct.
<code>disp(['w*u0*cond2*rbar^2 =', num2str(w*u0*cond2*rbar^2)])</code>	
<code>disp(' ')</code>	
<code>disp(' ')</code>	

To run this script, one simply types **ies1** at the Matlab prompt. The impedance results are returned in a vector called Zaccum, with one row for each  $c$  value. The results are plotted on the graph that is opened up in the first few lines. The user can easily change any parameters to see how the resulting impedance is affected.

The script ies2.m differs mainly that the conductivities of the base and cladding are chosen and fixed (chosen to agree with the ratios of the Dodd graph). Then, in order to preserve the  $\omega\mu\sigma\bar{r}^2$  constant, the angular frequency is adjusted. As long as  $r_2/r_1 = 2$  and  $l_2/l_1 = 327.2034/27.2034$ , the results of Figure 2 will be reproduced.

## Summary

Extremely good agreement has been shown between the analytic results for the impedance of a coil and the results returned from Oersted<sup>®</sup> BEM field-solving software. The flexibility of Matlab's software allows the user to easily corroborate the results of Dodd's equations. Limitations of the numeric solutions to the analytic equations are primarily the representation of an infinite integral with finite limits, and the ability of the numeric integration routines to converge over singularities. The well-corroborated results attest that these limitations are minimal.



## References

- [1] Dodd, C.V., Deeds, W.E., Luquire, J.W., "Integral Solutions to some Eddy Current Problems", *International Journal of Nondestructive Testing*, 1969, Vol. 1, pp. 29-90.

## Appendix – M-file listings

### ies1.m

```
if ~exist('c','var')
    close all
    load DataCmp
    plotZ,hold on
end

%cond1 = 1.493896e7;
%cond2 = 4.7812e6;
c0 = 2.997924574E+08;
u0 = 1.256637061E-06;
e0 = 8.854187853E-12;
n = 1; % number of turns
r1 = 3e-3; %381e-6;
r2 = 2*r1;
rbar = (r1+r2)/2; %5.715e-4;
w = 2*pi*4e6;
%w = 24.66/(u0*cond1*rbar^2);
cond1 = 77.05/(u0*w*rbar^2);
cond2 = 24.66/(u0*w*rbar^2);

l1 = 0.0476*rbar;
l2 = l1*12.0280333; %327.2034e-6;

%% normalize lengths
r1 = r1/rbar;
r2 = r2/rbar;
l1 = l1/rbar;
l2 = l2/rbar;
Le = (l2-l1);
%% -----
alimlo = 1e-9;
alimhi = 20;
c = [0 .01 .025 .05 .1 .15 .2 .25 .3];
disp(' ')
disp(' ')
disp(['w*u0*cond1*rbar^2 = ',num2str(w*u0*cond1*rbar^2)])
disp(['w*u0*cond2*rbar^2 = ',num2str(w*u0*cond2*rbar^2)])
disp(' ')
```

```

disp(' ')

Zaccum = [];
for cnt = 1:9 % cladding thickness normalized
    Zc = j*w*pi*u0*n^2*rbar/(Le^2*(r2-r1)^2)*...
        quad8('Zintgrnd',alimlo,alimhi,[],[],u0,e0,w,cond1,cond2,Le,r1,r2,c(cnt),rbar,l1,l2);

    Zair = 2*pi*w*u0*n^2*rbar/(Le^2*(r2-r1)^2)*...
        quad8('ZintgrndAIR',alimlo,alimhi,[],[],u0,e0,w,0,0,Le,r1,r2,c(cnt),rbar,l1,l2);

    Znorm = Zc./abs(Zair);
    Zaccum = [Zaccum;Znorm];
end

Zb2466 = Zaccum;
plot(real(Zaccum),imag(Zaccum),'kx','markersize',5)
figure(gcf)

```

## ies2.m

```

%% ies2 is to determine what the effect of keeping conductivities
%% constant is, while varying the frequency

if ~exist('c','var')
    close all
    load DataCmp
    plotZ,hold on
    plot(real(Oerst2466),imag(Oerst2466),'-go','linewidth',1.5)
end

BaseMat = 24.66;
cond1 = 5.813e7; %1.493896e7;
cond2 = cond1*BaseMat/77.05; %4.7812e6;
c0 = 2.997924574E+08;
u0 = 1.256637061E-06;
e0 = 8.854187853E-12;
n = 1; % number of turns
r1 = 381e-6;
r2 = 1.5*r1;
rbar = (r1+r2)/2; %5.715e-4;
%w = 2*pi*4e6;
w = BaseMat/(u0*cond2*rbar^2);

l1 = 0.0476*rbar;
l2 = l1*9; %12.0280333; %327.2034e-6;

%% normalize lengths
r1 = r1/rbar;

```

```

r2 = r2/rbar;
l1 = l1/rbar;
l2 = l2/rbar;
Le = (l2-l1);
%% -----
alimlo = 1e-9;
alimhi = 20;
c = [0 .01 .025 .05 .1 .15 .2 .25 .3];

Zaccum = [];
for cnt = 1:9 % cladding thickness normalized
    Zc = j*w*pi*u0*n^2*rbar/(Le^2*(r2-r1)^2)*...
        quad8('Zintgrnd',alimlo,alimhi,[],[],u0,e0,w,cond1,cond2,Le,r1,r2,c(cnt),rbar,l1,l2);

    Zair = 2*pi*w*u0*n^2*rbar/(Le^2*(r2-r1)^2)*...
        quad8('ZintgrndAIR',alimlo,alimhi,[],[],u0,e0,w,0,0,Le,r1,r2,c(cnt),rbar,l1,l2);

    Znorm = Zc./abs(Zair);
    Zaccum = [Zaccum;Znorm];
end

Zb2466 = Zaccum;
plot(real(Zaccum),imag(Zaccum),'--k+','markersize',5,'linewidth',1.5)
figure(gcf)
wavplay(laser)
disp(' ')
disp(' ')
disp(['w*u0*cond1*rbar^2 = ',num2str(w*u0*cond1*rbar^2)])
disp(['w*u0*cond2*rbar^2 = ',num2str(w*u0*cond2*rbar^2)])
disp(' ')
disp(' ')

```

## xBessJ1xa.m

```

function y = xBessJ1xa(x,a)
% function y = xBessJ1xa(x,a)
% y = x.*besselj(1,a*x);

y = x.*besselj(1,a*x);

```

## Jr2r1.m

```

function y = Jr1r2(a,r1,r2)

% function y = Jr1r2(a,r1,r2)
if size(a,1)==1 & size(a,2)>1

```

```

    a=a';
end
y=[];
for cnt = 1:length(a)
    y(cnt,1) = a(cnt)^2*quad('xBessJ1xa',r1,r2,[],[],a(cnt));
end

```

## Zintgrnd.m

```

function Zout = Zintgrnd(aa,u0,e0,w,cond1,cond2,Le,r1,r2,c,rbar,l1,l2)

if size(aa,1)==1 & size(aa,2)>1
    aa=aa';
end
a0 = alpha(0,aa,rbar,1,1,0,w,u0,e0);
a1 = alpha(1,aa,rbar,1,1,cond1,w,u0,e0);
B1 = beta(1,aa,rbar,1,1,cond1,w,u0,e0);
B2 = beta(1,aa,rbar,1,1,cond2,w,u0,e0);

Zout = 1./(a0.*aa).^3 .* Jr2r1(aa,r1,r2).^2 .* (2*a0*Le + 2*exp(-a0*Le) - 2 + ...
    (exp(-2*a0*l2) + exp(-2*a0*l1) - 2*exp(-a0*(l1+l2))) .*...
    ((a0+B1).*(B1-B2)+(a0-B1).*(B1+B2).*exp(2*a1*c))./((a0-B1).* ...
    (B1-B2)+(a0+B1).*(B1+B2).*exp(2*a1*c))));

function alf = alpha(n,a,rbar,ur,er,sigma,w,u0,e0)
% n is a dummy argument so you can see which alpha it is.
if n == 0
    alf = sqrt(a.^2 - rbar^2*w^2*ur*u0*er*e0);
else
    alf = sqrt(a.^2 - rbar^2*w^2*ur*u0*er*e0 + j*rbar^2*w*ur*u0*sigma);
end

function bet = beta(n,a,rbar,ur,er,sigma,w,u0,e0)
bet = alpha(n,a,rbar,ur,er,sigma,w,u0,e0)/ur;

```

## ZintgrndAIR.m

```

function Zout = ZintgrndAIR(aa,u0,e0,w,cond1,cond2,Le,r1,r2,c,rbar,l1,l2)

if size(aa,1)==1 & size(aa,2)>1
    aa=aa';
end
a0 = alpha(0,aa,rbar,1,1,0,w,u0,e0);
Zout = 1./(a0.*aa).^3 .* Jr2r1(aa,r1,r2).^2 .* (a0*Le +exp(-a0*Le) - 1);

function alf = alpha(n,a,rbar,ur,er,sigma,w,u0,e0)

```

```

% n is a dummy argument so you can see which alpha it is.
if n == 0
    alf = sqrt(a.^2 - rbar^2*w^2*ur*u0*er*e0);
else
    alf = sqrt(a.^2 - rbar^2*w^2*ur*u0*er*e0 + j*rbar^2*w*ur*u0*sigma);
end

function bet = beta(n,a,rbar,ur,er,sigma,w,u0,e0)
bet = alpha(n,a,rbar,ur,er,sigma,w,u0,e0)/ur;

```

## plotZ.m

```

plot(real(Dodd2466),imag(Dodd2466),'-ko','linewidth',2.5)
grid on,hold on
plot(real(Oerst2466),imag(Oerst2466),'-bo','linewidth',2.5)
axis image
set(gca,'xlim',[0.08 0.16],'ylim',[0.62 0.74])
%set(gca,'xlim',[0.05 0.25],'ylim',[0.5 0.8])
%set(gca,'ylim',[.5 .8])
%set(gca,'xlim',[.05 .2])
xlabel('Resistive Component')
ylabel('Reactive Component')

legend('Dodd','Oersted',1)

```

## plotZ40.m

```

plot(excise(real(Dodd4000)),excise(imag(Dodd4000)),'-ko','linewidth',2.5)
grid on,hold on
plot(real(Oerst4000),imag(Oerst4000),'-bo','linewidth',2.5)
axis image
set(gca,'xlim',[0.08 0.16],'ylim',[0.62 0.74])
%set(gca,'xlim',[0.05 0.25],'ylim',[0.5 0.8])
%set(gca,'ylim',[.5 .8])
%set(gca,'xlim',[.05 .2])
xlabel('Resistive Component')
ylabel('Reactive Component')

legend('Dodd','Oersted',1)

```

## findbest.m

```

%% findbest is to match teh 0.05 c-point as well as possible
%% checking r2/r1 = 1.5:.1:1.6 and l2/l1 = 9:.1:10

c = 0.05;
BaseMat = 24.66;

```

```

cond1 = 5.813e7; %1.493896e7;
cond2 = cond1*BaseMat/77.05; %4.7812e6;
c0 = 2.997924574E+08;
u0 = 1.256637061E-06;
e0 = 8.854187853E-12;
n = 1; % number of turns
r1 = 381e-6;
rfac = 1.5:.01:1.6;
lfac=(9:.1:10)';
Zaccum = zeros(length(lfac),length(rfac));

for cnt1 = 1:length(rfac)
    r2 = r1*rfac(cnt1);
    rbar = (r1+r2)/2; %5.715e-4;
    l1 = 0.0476*rbar;
    w = BaseMat/(u0*cond2*rbar^2);
    for cnt2 = 1:length(lfac)
        l2 = l1*lfac(cnt2);
        %% normalize lengths
        r1 = r1/rbar;
        r2 = r2/rbar;
        l1 = l1/rbar;
        l2 = l2/rbar;
        Le = (l2-l1);
        %% -----
        alimlo = 1e-9;
        alimhi = 20;
        Zc = j*pi*u0*n^2*rbar/(Le^2*(r2-r1)^2)*...
            quad8('Zintgrnd',alimlo,alimhi,[],[],u0,e0,w,cond1,cond2,Le,r1,r2,c,rbar,l1,l2);
        Zair = 2*pi*w*u0*n^2*rbar/(Le^2*(r2-r1)^2)*...
            quad8('ZintgrndAIR',alimlo,alimhi,[],[],u0,e0,w,0,0,Le,r1,r2,c,rbar,l1,l2);

        Znorm = Zc./abs(Zair);
        Zaccum(cnt2,cnt1) = Znorm;
    end
end
Dodd05 = 0.1380 + 0.6650*i;

clear cnt1 cnt2

```

## excise.m

```

function X = excise(X)

%% removes NaNs from a column-oriented data matrix,
%% the entire row is removed.
%% Or, removes NaNs from a vector.
%% See page 2-36 of the User's Guide.

```

```
[p,q] = size(X);

if q > 1 & p >1
    X(any(isnan(X)',:)) = [];
else
    X(isnan(X)) = [];
end
```