PYTHON CODE

3D WHEELER'S CURVE

ALOK T J
Guided By - Dr. S. Raghavan
Department of Electronics and Communication
National Institute of Technology, Trichy

ABSTRACT

This presentation consist of programming code and plot of Wheeler's Curve for various standard Dielectric Substrates. The Wheeler's Curve is a discontinuous graph at (w/h)=1. The Wheeler's curve is plotted by taking the two curves from either side of the point of discontinuity and then combined plotted on one graph for complete range of (w/h) values. From the Wheeler's Curve Plot, corresponding values of Impedance and (w/h) on the curve is taken with uniform linear separation and made into a tabular column for each Standard Dielectric Substrates. All the values obtained from the Wheeler's curve is Consolidated in a single chart for Microstrip line designers.

INTRODUCTION

Early attempts to characterize the performance of microstrip line were based on the quasi-TEM model. Various electrostatic approximations such as conformal mapping, relaxation methods, variation techniques, the method of Green's function and the moment method are generally used. During the last few years, a number of papers using a hybrid-mode model of microstrip line have appeared, but these involve extensive computations.

Closed-form expressions are absolutely necessary for optimization and computer-aided design of a microstrip circuit. The closed form expressions for Z_0 *and* k_a have been reported by Wheeler, Schneider and Hammerstad. Wheeler and Hammerstad have also given an expression for w/h in terms of Z_0 *and* k

For a practical range of microstrip lines ($0.05 \leq w/h \leq 20$ and $k \leq 16$) Hammerstad reported that his expressions are more accurate than earlier work, and fall within ±1 per cent of Wheeler's numerical results. His expressions, which are based on the work of Wheeler and Schneider, include useful relationships defining both characteristic impedance and effective dielectric constant:

For $w/h \leq 1$

$$Z_0 = \frac{60}{\sqrt{k_a}} \ln\left(\frac{8h}{w} + \frac{w}{4h}\right)$$

where

$$k_a = \frac{k+1}{2} + \frac{k-1}{2}\left[\left(1 + 12\left(\frac{h}{w}\right)^{-\frac{1}{2}} + 0.04\left(1 - \left(\frac{w}{h}\right)\right)^2\right)\right]$$

For w/h ≥ 1

$$Z_0 = \frac{120\pi}{\sqrt{k_a}\left(\frac{w}{h} + 1.383 + 0.667\ln\left(\frac{w}{h} + 1.444\right)\right)}$$

where

$$k_a = \frac{k+1}{2} + \frac{k-1}{2}\left(1 + 12\left(\frac{h}{w}\right)\right)^{-\frac{1}{2}}$$

Python Program for Wheeler's Curve

It is important to note that the Wheeler's Curve equation is discontinuous at w/h=1. Therefore the program required to plot Wheeler's curve must contain provision for plot of two curves which apparently looks continuous at w/h=1.

Parameters invoked in the program

Z_0 = Characteristic Wave Impedance
k = Dielectric Constant
k_a = Effective Dielectric Constant
w = Width of Substrate
h = Height of Substrate

Python Code for 2D Wheeler's Curve

Libraries Imported

```
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
import math
```

Parameters Initialization

```
zc=377
k=[1.3,2,2.1,2.2,2.3,2.33,4.3,8.9,9.3]
pi=math.pi
```

Splitting Alogrithm

```
x=np.linspace(.5,10,50)
x1=[]
x2=[]
for i in x:
  if i<1:
    x1.append(i)
  else:
    x2.append(i)
```

Calculation of Impedance for Cotton

```
z1=[]
for i in x1:
  ka=((k[0]+1)/2)+(((k[0]-1)/2)*(((1+(12/i))**-0.5)+0.04*(1-i)**2))
  d=(zc/(2*pi*(ka**0.5)))*np.log((8/i)+(i/4))
  z1.append(d)
for i in x2:
  ka=((k[0]+1)/2)+(((k[0]-1)/2)*((1+(12/i))**-0.5))
  d=zc/((ka**0.5)*(1.393+i+((2/3)*np.log(i+1.4444))))
  z1.append(d)
```

Calculation of Impedance for PTFE

```
z2=[]
for i in x1:
  ka=((k[1]+1)/2)+(((k[1]-1)/2)*(((1+(12/i))**-0.5)+0.04*(1-i)**2))
  d=(zc/(2*pi*(ka**0.5)))*np.log((8/i)+(i/4))
  z2.append(d)
for i in x2:
  ka=((k[1]+1)/2)+(((k[1]-1)/2)*((1+(12/i))**-0.5))
  d=zc/((ka**0.5)*(1.393+i+((2/3)*np.log(i+1.4444))))
  z2.append(d)
```

Calculation of Impedance for Teflon

```
z3=[]
for i in x1:
  ka=((k[2]+1)/2)+(((k[2]-1)/2)*(((1+(12/i))**-0.5)+0.04*(1-i)**2))
  d=(zc/(2*pi*(ka**0.5)))*np.log((8/i)+(i/4))
  z3.append(d)
for i in x2:
  ka=((k[2]+1)/2)+(((k[2]-1)/2)*((1+(12/i))**-0.5))
  d=zc/((ka**0.5)*(1.393+i+((2/3)*np.log(i+1.4444))))
  z3.append(d)
```

Calculation of Impedance for RT/duroid 5880

```
z4=[]
for i in x1:
  ka=((k[3]+1)/2)+(((k[3]-1)/2)*(((1+(12/i))**-0.5)+0.04*(1-i)**2))
  d=(zc/(2*pi*(ka**0.5)))*np.log((8/i)+(i/4))
  z4.append(d)
for i in x2:
  ka=((k[3]+1)/2)+(((k[3]-1)/2)*((1+(12/i))**-0.5))
  d=zc/((ka**0.5)*(1.393+i+((2/3)*np.log(i+1.4444))))
  z4.append(d)
```

Calculation of Impedance for Paper

```
z5=[]
for i in x1:
  ka=((k[4]+1)/2)+(((k[4]-1)/2)*(((1+(12/i))**-0.5)+0.04*(1-i)**2))
  d=(zc/(2*pi*(ka**0.5)))*np.log((8/i)+(i/4))
  z5.append(d)
for i in x2:
  ka=((k[4]+1)/2)+(((k[4]-1)/2)*((1+(12/i))**-0.5))
  d=zc/((ka**0.5)*(1.393+i+((2/3)*np.log(i+1.4444))))
  z5.append(d)
```

Calculation of Impedance for RT/duroid 5870

```
z6=[]
for i in x1:
  ka=((k[5]+1)/2)+(((k[5]-1)/2)*(((1+(12/i))**-0.5)+0.04*(1-i)**2))
  d=(zc/(2*pi*(ka**0.5)))*np.log((8/i)+(i/4))
  z6.append(d)
for i in x2:
  ka=((k[5]+1)/2)+(((k[5]-1)/2)*((1+(12/i))**-0.5))
  d=zc/((ka**0.5)*(1.393+i+((2/3)*np.log(i+1.4444))))
  z6.append(d)
```

Calculation of Impedance for FR-4

```
z7=[]
for i in x1:
  ka=((k[6]+1)/2)+(((k[6]-1)/2)*(((1+(12/i))**-0.5)+0.04*(1-i)**2))
  d=(zc/(2*pi*(ka**0.5)))*np.log((8/i)+(i/4))
  z7.append(d)
for i in x2:
  ka=((k[6]+1)/2)+(((k[6]-1)/2)*((1+(12/i))**-0.5))
  d=zc/((ka**0.5)*(1.393+i+((2/3)*np.log(i+1.4444))))
  z7.append(d)
```

Calculation of Impedance for Sapphire

```
z8=[]
for i in x1:
  ka=((k[7]+1)/2)+(((k[7]-1)/2)*(((1+(12/i))**-0.5)+0.04*(1-i)**2))
  d=(zc/(2*pi*(ka**0.5)))*np.log((8/i)+(i/4))
  z8.append(d)
for i in x2:
  ka=((k[7]+1)/2)+(((k[7]-1)/2)*((1+(12/i))**-0.5))
  d=zc/((ka**0.5)*(1.393+i+((2/3)*np.log(i+1.4444))))
  z8.append(d)
```

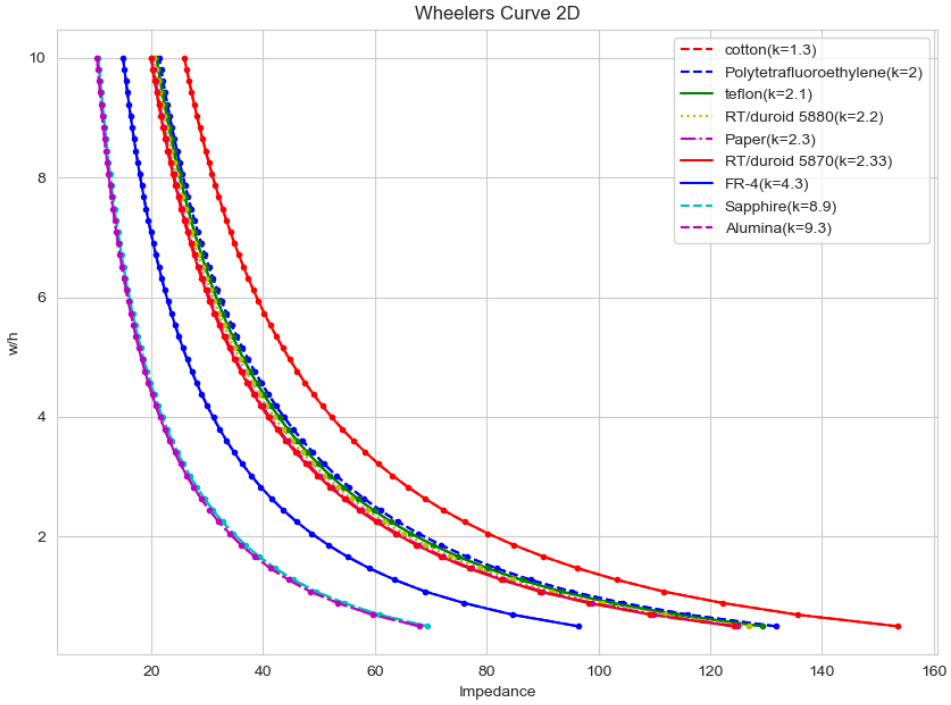
Calculation of Impedance for Alumina

```
z9=[]
for i in x1:
  ka=((k[8]+1)/2)+(((k[8]-1)/2)*(((1+(12/i))**-0.5)+0.04*(1-i)**2))
  d=(zc/(2*pi*(ka**0.5)))*np.log((8/i)+(i/4))
  z9.append(d)
for i in x2:
  ka=((k[8]+1)/2)+(((k[8]-1)/2)*((1+(12/i))**-0.5))
  d=zc/((ka**0.5)*(1.393+i+((2/3)*np.log(i+1.4444))))
  z9.append(d)
```

Plotting 2D Wheelers Curve

```
sns.set_style("whitegrid")
plt.figure(figsize=(10,8))
plt.plot(z1,x,'--r');
plt.xlabel("Impedance")
plt.ylabel("w/h")
plt.title('Wheelers Curve');
plt.plot(z2,x,'--b');
plt.plot(z3,x,'-g');
plt.plot(z4,x,":y");
plt.plot(z5,x,"-.m");
plt.plot(z6,x,"-r");
plt.plot(z7,x,"-b");
plt.plot(z8,x,"--c");
plt.plot(z9,x,"--m");
plt.legend(['cotton(k=1.3)','Polytetrafluoroethylene(k=2)','teflon(k=2.1)',
'RT/duroid 5880(k=2.2)','Paper(k=2.3)','RT/duroid 5870(k=2.33)','FR-4(k=4.3)',
'Sapphire(k=8.9)','Alumina(k=9.3)']);
```

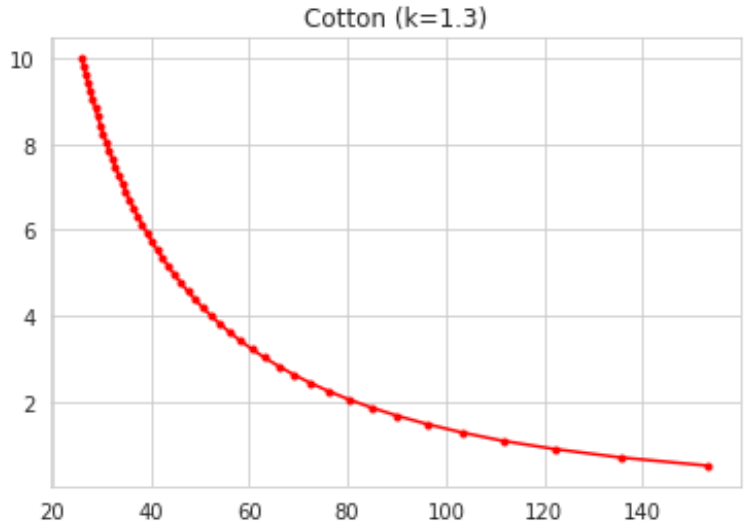
3D Wheeler's Curve



2D Graph of Wheeler's Curve for some of the standard Dielectric Substrates

Extract Data From Curve of Cotton

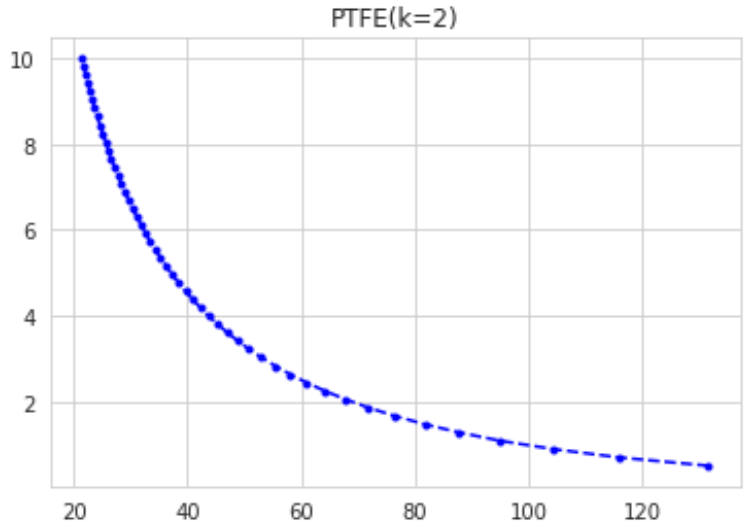
```
line2d1 = plt.plot(z1,x,'.-r')
plt.title("Cotton (k=1.3)")
xvalues1 = line2d1[0].get_xdata()
yvalues1 = line2d1[0].get_ydata()
```



These are the points that are extracted from the graph

Extract Data From Curve of PTFE

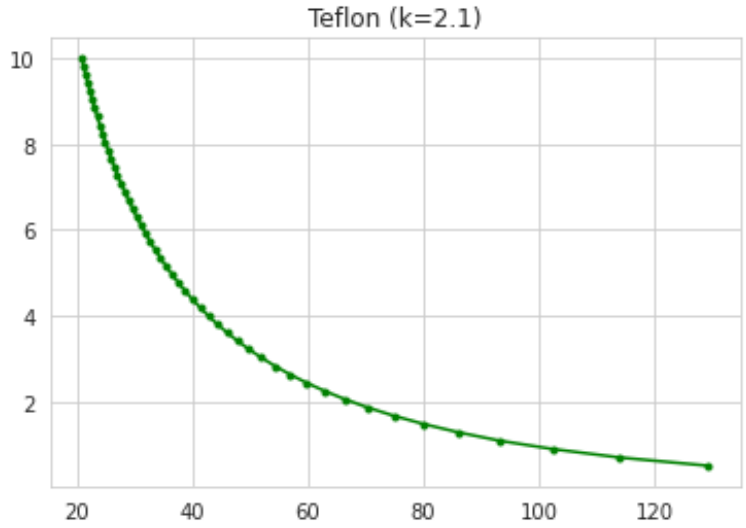
```
line2d2 = plt.plot(z2,x,'.--b')
plt.title("PTFE(k=2)")
xvalues2 = line2d2[0].get_xdata()
yvalues2 = line2d2[0].get_ydata()
```



These are the points that are extracted from the graph

Extract Data From Curve of Teflon

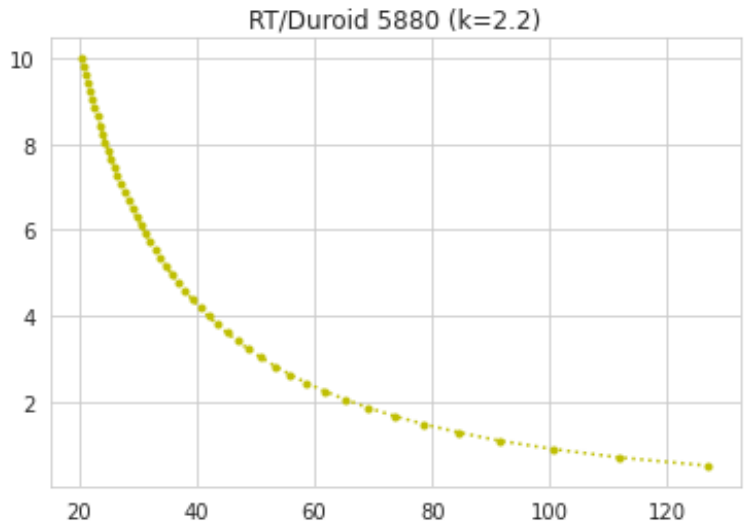
```
line2d3 = plt.plot(z3,x,'.-g')
plt.title("Teflon (k=2.1)")
xvalues3 = line2d3[0].get_xdata()
yvalues3 = line2d3[0].get_ydata()
```



These are the points that are extracted from the graph

Extract Data From Curve of RT/duroid 5880

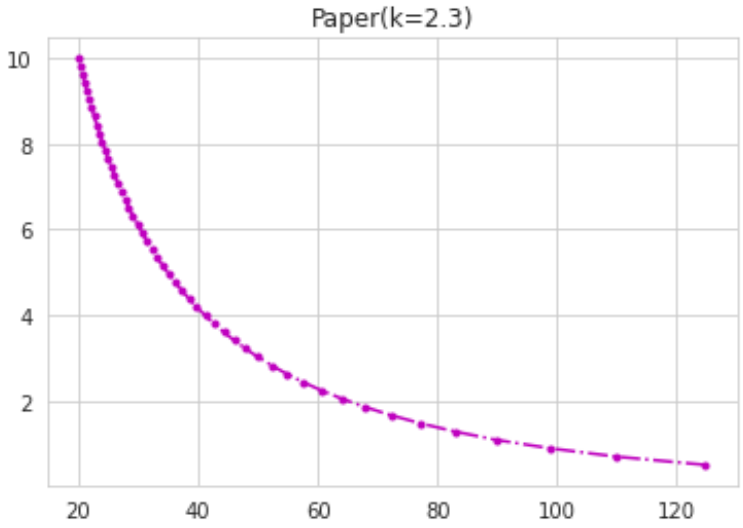
```
line2d4 = plt.plot(z4,x,".:y")
plt.title("RT/Duroid 5880 (k=2.2)")
xvalues4 = line2d4[0].get_xdata()
yvalues4 = line2d4[0].get_ydata()
```



These are the points that are extracted from the graph

Extract Data From Curve of Paper

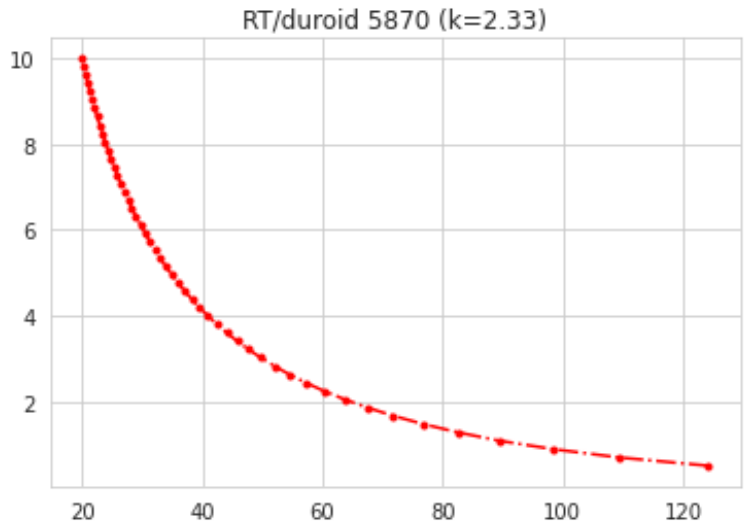
```
line2d5 = plt.plot(z5,x,".-.m")
plt.title("Paper(k=2.3)")
xvalues5 = line2d5[0].get_xdata()
yvalues5 = line2d5[0].get_ydata()
```



These are the points that are extracted from the graph

Extract Data From Curve of RT/duroid 5870

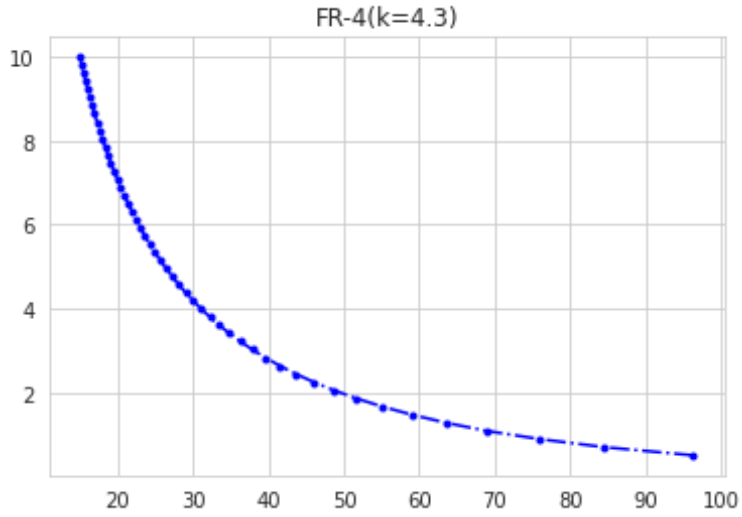
```
line2d6 = plt.plot(z6,x,".-.r")
plt.title("RT/duroid 5870 (k=2.33)")
xvalues6 = line2d6[0].get_xdata()
yvalues6 = line2d6[0].get_ydata()
```



These are the points that are extracted from the graph

Extract Data From Curve of FR-4

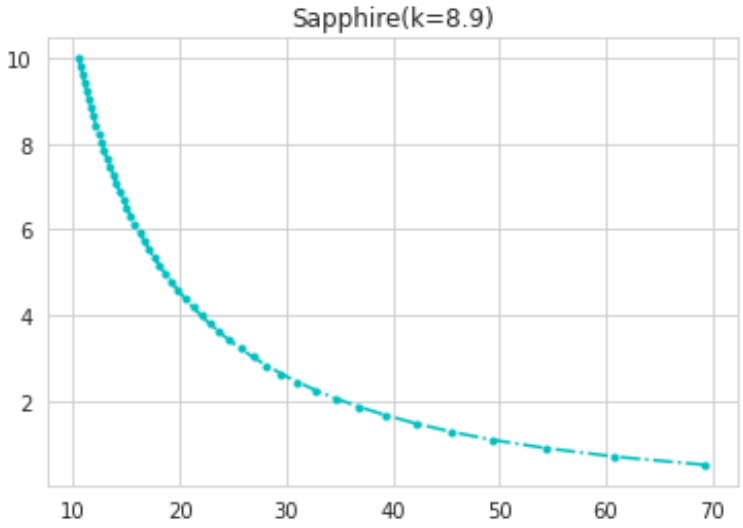
```
line2d7 = plt.plot(z7,x,".-.b")
plt.title("FR-4(k=4.3)")
xvalues7 = line2d7[0].get_xdata()
yvalues7 = line2d7[0].get_ydata()
```



These are the points that are extracted from the graph

Extract Data From Curve of Sapphir

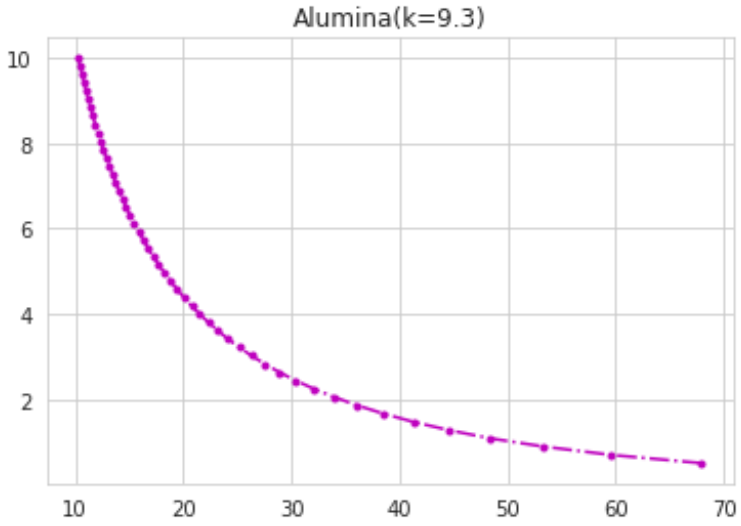
```
line2d8 = plt.plot(z8,x,".-.c")
plt.title("Sapphire(k=8.9)")
xvalues8 = line2d8[0].get_xdata()
yvalues8 = line2d8[0].get_ydata()
```



These are the points that are extracted from the graph

Extract Data From Curve of Alumina

```
line2d9 = plt.plot(z9,x,".-.m")
plt.title("Alumina(k=9.3)")
xvalues9 = line2d9[0].get_xdata()
yvalues9 = line2d9[0].get_ydata()
```



These are the points that are extracted from the graph

Print the Extracted Data for Cotton

```
print("The tabular column for Cotton  ")
print("  w/h         Impedance")
for i in range(50):
  print(("{a:1.5f}     : {b:1.5f}").format(a=yvalues1[i],b=xvalues1[i]))
```

Table 1: Output of above print code

The tabular column for Cotton			
w/h	Impedance	w/h	Impedance
0.50000	153.47843	5.34694	42.34449
0.69388	135.54780	5.54082	41.23321
0.88776	122.29139	5.73469	40.18094
1.08163	111.60959	5.92857	39.18299
1.27551	103.32729	6.12245	38.23518
1.46939	96.27510	6.31633	37.33373
1.66327	90.18933	6.51020	36.47525
1.85714	84.87785	6.70408	35.65668
2.05102	80.19709	6.89796	34.87524
2.24490	76.03751	7.09184	34.12841
2.43878	72.31398	7.28571	33.41390
2.63265	68.95925	7.47959	32.72961
2.82653	65.91948	7.67347	32.07363
3.02041	63.15097	7.86735	31.44420
3.21429	60.61793	8.06122	30.83972
3.40816	58.29066	8.25510	30.25870
3.60204	56.14438	8.44898	29.69978
3.79592	54.15817	8.64286	29.16170
3.98980	52.31429	8.83673	28.64330
4.18367	50.59756	9.03061	28.14350
4.37755	48.99492	9.22449	27.66129
4.57143	47.49508	9.41837	27.19576
4.76531	46.08821	9.61224	26.74603
4.95918	44.76570	9.80612	26.31130
5.15306	43.52002	10.00000	25.89082

Print the Extracted Data for PTFE

```
print("The tabular column for PTFE  ")
print("  w/h         Impedance")
for i in range(50):
  print(("{a:1.5f}     : {b:1.5f}").format(a=yvalues2[i],b=xvalues2[i]))
```

Table 2: Output of above print code

The tabular column for PTFE			
w/h	Impedance	w/h	Impedance
0.50000	131.68208	5.34694	35.27045
0.69388	116.00437	5.54082	34.32610
0.88776	104.41839	5.73469	33.43251
1.08163	95.08780	5.92857	32.58563
1.27551	87.85644	6.12245	31.78183
1.46939	81.71390	6.31633	31.01785
1.66327	76.42435	6.51020	30.29074
1.85714	71.81654	6.70408	29.59785
2.05102	67.76293	6.89796	28.93678
2.24490	64.16641	7.09184	28.30536
2.43878	60.95168	7.28571	27.70160
2.63265	58.05938	7.47959	27.12369
2.82653	55.44203	7.67347	26.56999
3.02041	53.06118	7.86735	26.03898
3.21429	50.88537	8.06122	25.52927
3.40816	48.88853	8.25510	25.03959
3.60204	47.04893	8.44898	24.56876
3.79592	45.34826	8.64286	24.11571
3.98980	43.77098	8.83673	23.67942
4.18367	42.30384	9.03061	23.25897
4.37755	40.93544	9.22449	22.85351
4.57143	39.65591	9.41837	22.46224
4.76531	38.45669	9.61224	22.08441
4.95918	37.33029	9.80612	21.71933
5.15306	36.27014	10.00000	21.36637

Print the Extracted Data for Teflon

```
print("The tabular column for Teflon  ")
print("  w/h        Impedance")
for i in range(50):
  print(("{a:1.5f}    : {b:1.5f}").format(a=yvalues3[i],b=xvalues3[i]))
```

Table 3: Output of above print code

The tabular column for Teflon			
w/h	Impedance	w/h	Impedance
0.50000	129.26826	5.34694	34.52344
0.69388	113.84884	5.54082	33.59734
0.88776	102.45441	5.73469	32.72109
1.08163	93.27874	5.92857	31.89071
1.27551	86.16788	6.12245	31.10261
1.46939	80.12919	6.31633	30.35359
1.66327	74.93019	6.51020	29.64077
1.85714	70.40215	6.70408	28.96154
2.05102	66.41942	6.89796	28.31354
2.24490	62.88637	7.09184	27.69464
2.43878	59.72884	7.28571	27.10288
2.63265	56.88839	7.47959	26.53649
2.82653	54.31831	7.67347	25.99384
3.02041	51.98075	7.86735	25.47347
3.21429	49.84474	8.06122	24.97399
3.40816	47.88466	8.25510	24.49416
3.60204	46.07911	8.44898	24.03283
3.79592	44.41008	8.64286	23.58893
3.98980	42.86231	8.83673	23.16148
4.18367	41.42274	9.03061	22.74957
4.37755	40.08017	9.22449	22.35236
4.57143	38.82490	9.41837	21.96906
4.76531	37.64852	9.61224	21.59894
4.95918	36.54366	9.80612	21.24133
5.15306	35.50386	10.00000	20.89560

Print the Extracted Data for RT/duroid 5880

```
print("The tabular column for RT/duroid 5880  ")
print("  w/h        Impedance")
for i in range(50):
  print(("{a:1.5f}    : {b:1.5f}").format(a=yvalues4[i],b=xvalues4[i]))
```

Table 4: Output of above print code

The tabular column for RT/duroid 5880			
w/h	Impedance	w/h	Impedance
0.50000	126.98248	5.34694	33.82196
0.69388	111.80918	5.54082	32.91311
0.88776	100.59724	5.73469	32.05324
1.08163	91.56915	5.92857	31.23842
1.27551	84.57308	6.12245	30.46515
1.46939	78.63324	6.31633	29.73026
1.66327	73.52038	6.51020	29.03093
1.85714	69.06817	6.70408	28.36458
2.05102	65.15278	6.89796	27.72891
2.24490	61.67999	7.09184	27.12181
2.43878	58.57675	7.28571	26.54137
2.63265	55.78551	7.47959	25.98584
2.82653	53.26026	7.67347	25.45362
3.02041	50.96374	7.86735	24.94327
3.21429	48.86546	8.06122	24.45344
3.40816	46.94019	8.25510	23.98290
3.60204	45.16689	8.44898	23.53051
3.79592	43.52782	8.64286	23.09524
3.98980	42.00797	8.83673	22.67611
4.18367	40.59449	9.03061	22.27224
4.37755	39.27636	9.22449	21.88279
4.57143	38.04404	9.41837	21.50700
4.76531	36.88925	9.61224	21.14414
4.95918	35.80475	9.80612	20.79356
5.15306	34.78419	10.00000	20.45464

Print the Extracted Data for Paper

```
print("The tabular column for Paper ")
print("  w/h        Impedance")
for i in range(50):
  print(("{a:1.5f}    : {b:1.5f}").format(a=yvalues5[i],b=xvalues5[i]))
```

Table 5: Output of above print code

The tabular column for Paper			
w/h	Impedance	w/h	Impedance
0.50000	124.81383	5.34694	33.16157
0.69388	109.87535	5.54082	32.26905
0.88776	98.83754	5.73469	31.42468
1.08163	89.95025	5.92857	30.62460
1.27551	83.06367	6.12245	29.86534
1.46939	77.21805	6.31633	29.14382
1.66327	72.18726	6.51020	28.45724
1.85714	67.80725	6.70408	27.80308
2.05102	63.95595	6.89796	27.17906
2.24490	60.54048	7.09184	26.58312
2.43878	57.48886	7.28571	26.01338
2.63265	54.74439	7.47959	25.46810
2.82653	52.26172	7.67347	24.94575
3.02041	50.00417	7.86735	24.44486
3.21429	47.94171	8.06122	23.96414
3.40816	46.04949	8.25510	23.50236
3.60204	44.30678	8.44898	23.05843
3.79592	42.69614	8.64286	22.63130
3.98980	41.20276	8.83673	22.22003
4.18367	39.81401	9.03061	21.82374
4.37755	38.51905	9.22449	21.44162
4.57143	37.30848	9.41837	21.07291
4.76531	36.17414	9.61224	20.71691
4.95918	35.10892	9.80612	20.37297
5.15306	34.10657	10.00000	20.04047

Print the Extracted Data for RT/duroid 5870

```
print("The tabular column for RT/duroid 5870 ")
print("  w/h        Impedance")
for i in range(50):
  print(("{a:1.5f}    : {b:1.5f}").format(a=yvalues6[i],b=xvalues6[i]))
```

Table 6: Output of above print code

The tabular column for RT/duroid 5870			
w/h	Impedance	w/h	Impedance
0.50000	124.18459	5.34694	32.97088
0.69388	109.31450	5.54082	32.08309
0.88776	98.32738	5.73469	31.24321
1.08163	89.48108	5.92857	30.44739
1.27551	82.62637	6.12245	29.69220
1.46939	76.80818	6.31633	28.97454
1.66327	71.80126	6.51020	28.29166
1.85714	67.44224	6.70408	27.64103
2.05102	63.60957	6.89796	27.02038
2.24490	60.21076	7.09184	26.42767
2.43878	57.17414	7.28571	25.86102
2.63265	54.44324	7.47959	25.31872
2.82653	51.97294	7.67347	24.79921
3.02041	49.72671	7.86735	24.30107
3.21429	47.67464	8.06122	23.82298
3.40816	45.79202	8.25510	23.36374
3.60204	44.05818	8.44898	22.92225
3.79592	42.45578	8.64286	22.49748
3.98980	40.97008	8.83673	22.08848
4.18367	39.58851	9.03061	21.69439
4.37755	38.30026	9.22449	21.31439
4.57143	37.09599	9.41837	20.94773
4.76531	35.96759	9.61224	20.59371
4.95918	34.90796	9.80612	20.25169
5.15306	33.91088	10.00000	19.92104

Print the Extracted Data for FR-4

```
print("The tabular column for FR-4 ")
print("  w/h         Impedance")
for i in range(50):
  print(("{a:1.5f}     : {b:1.5f}").format(a=yvalues7[i],b=xvalues7[i]))
```

Table 7: Output of above print code

The tabular column for FR-4			
w/h	Impedance	w/h	Impedance
0.50000	96.37338	5.34694	24.90193
0.69388	84.62352	5.54082	24.22018
0.88776	75.94843	5.73469	23.57563
1.08163	68.96978	5.92857	22.96527
1.27551	63.56639	6.12245	22.38641
1.46939	58.99131	6.31633	21.83665
1.66327	55.06261	6.51020	21.31380
1.85714	51.64882	6.70408	20.81592
2.05102	48.65243	6.89796	20.34124
2.24490	45.99943	7.09184	19.88815
2.43878	43.63259	7.28571	19.45519
2.63265	41.50693	7.47959	19.04104
2.82653	39.58654	7.67347	18.64448
3.02041	37.84240	7.86735	18.26440
3.21429	36.25081	8.06122	17.89978
3.40816	34.79218	8.25510	17.54969
3.60204	33.45019	8.44898	17.21326
3.79592	32.21110	8.64286	16.88971
3.98980	31.06330	8.83673	16.57830
4.18367	29.99687	9.03061	16.27836
4.37755	29.00330	9.22449	15.98925
4.57143	28.07524	9.41837	15.71039
4.76531	27.20631	9.61224	15.44124
4.95918	26.39094	9.80612	15.18130
5.15306	25.62425	10.00000	14.93011

Print the Extracted Data for Sapphire

```
print("The tabular column for Sapphire ")
print("  w/h        Impedance")
for i in range(50):
  print(("{a:1.5f}    : {b:1.5f}").format(a=yvalues8[i],b=xvalues8[i]))
```

Table 8: Output of above print code

The tabular column for Sapphire			
w/h	Impedance	w/h	Impedance
0.50000	69.39355	5.34694	17.59491
0.69388	60.82357	5.54082	17.10795
0.88776	54.50067	5.73469	16.64775
1.08163	49.41816	5.92857	16.21214
1.27551	45.48562	6.12245	15.79918
1.46939	42.16195	6.31633	15.40713
1.66327	39.31236	6.51020	15.03442
1.85714	36.83972	6.70408	14.67964
2.05102	34.67214	6.89796	14.34150
2.24490	32.75516	7.09184	14.01886
2.43878	31.04676	7.28571	13.71066
2.63265	29.51392	7.47959	13.41593
2.82653	28.13037	7.67347	13.13382
3.02041	26.87486	7.86735	12.86351
3.21429	25.73007	8.06122	12.60427
3.40816	24.68170	8.25510	12.35544
3.60204	23.71784	8.44898	12.11638
3.79592	22.82849	8.64286	11.88654
3.98980	22.00519	8.83673	11.66538
4.18367	21.24071	9.03061	11.45242
4.37755	20.52888	9.22449	11.24720
4.57143	19.86435	9.41837	11.04930
4.76531	19.24248	9.61224	10.85835
4.95918	18.65925	9.80612	10.67397
5.15306	18.11110	10.00000	10.49583

Print the Extracted Data for Alumina

```
print("The tabular column for Alumina ")
print("  w/h         Impedance")
for i in range(50):
  print(("{a:1.5f}     : {b:1.5f}").format(a=yvalues9[i],b=xvalues9[i]))
```

Table 9: Output of above print code

The tabular column for Alumina			
w/h	Impedance	w/h	Impedance
0.50000	67.98481	5.34694	17.22391
0.69388	59.58418	5.54082	16.74699
0.88776	53.38642	5.73469	16.29630
1.08163	48.40469	5.92857	15.86971
1.27551	44.55025	6.12245	15.46530
1.46939	41.29285	6.31633	15.08137
1.66327	38.50026	6.51020	14.71638
1.85714	36.07723	6.70408	14.36896
2.05102	33.95325	6.89796	14.03784
2.24490	32.07493	7.09184	13.72190
2.43878	30.40105	7.28571	13.42010
2.63265	28.89926	7.47959	13.13151
2.82653	27.54377	7.67347	12.85527
3.02041	26.31378	7.86735	12.59059
3.21429	25.19230	8.06122	12.33675
3.40816	24.16531	8.25510	12.09310
3.60204	23.22113	8.44898	11.85904
3.79592	22.34996	8.64286	11.63399
3.98980	21.54351	8.83673	11.41745
4.18367	20.79470	9.03061	11.20893
4.37755	20.09748	9.22449	11.00800
4.57143	19.44660	9.41837	10.81425
4.76531	18.83753	9.61224	10.62729
4.95918	18.26630	9.80612	10.44677
5.15306	17.72944	10.00000	10.27236

Python Code for 3D Wheeler's Curve

Libraries Imported

```
import matplotlib.pyplot as plt
import numpy as np
import math
```

Parameter Initialization

```
zc=377
pi=math.pi
#k=[1.3,2,2.1,2.2,2.3,2.33,4.3,6,6.15,6.76,7,7.9,8.6,8.9,9.3,10,12.94]
n=int(input("Enter the number dielectric constants :"))
k=[]
```

Input for Dielectric Constants

```
for i in range(n):
    c=float(input('Enter the dielectric constant:'))
    k.append(c)
```

Set Title and Axis Label

```
ax = plt.axes( projection='3d')
ax.set_ylabel('Er  (Y-axis)')
ax.set_xlabel('Impedance(ohms) (X-axis)')
ax.set_zlabel('w/h Ratio (Z-axis)')
```

Plot of Wheeler's curve for w/h >1

```
for i in range(n):
    x=np.linspace(1,10,50)
    a=np.full(50,k[i])
    v='C'+str(i)
    ka=((a+1)/2)+(((a-1)/2)*((1+(12/x))**-0.5))
    d=zc/((ka**0.5)*(1.393+x+((2/3)*np.log(x+1.4444))))
    ax.plot3D(d, a, x, label='Er = {0:.3f}'.format(k[i]))
    ax.legend()
    plt.show()
    plt.title('3D-Wheelers Curve');
```

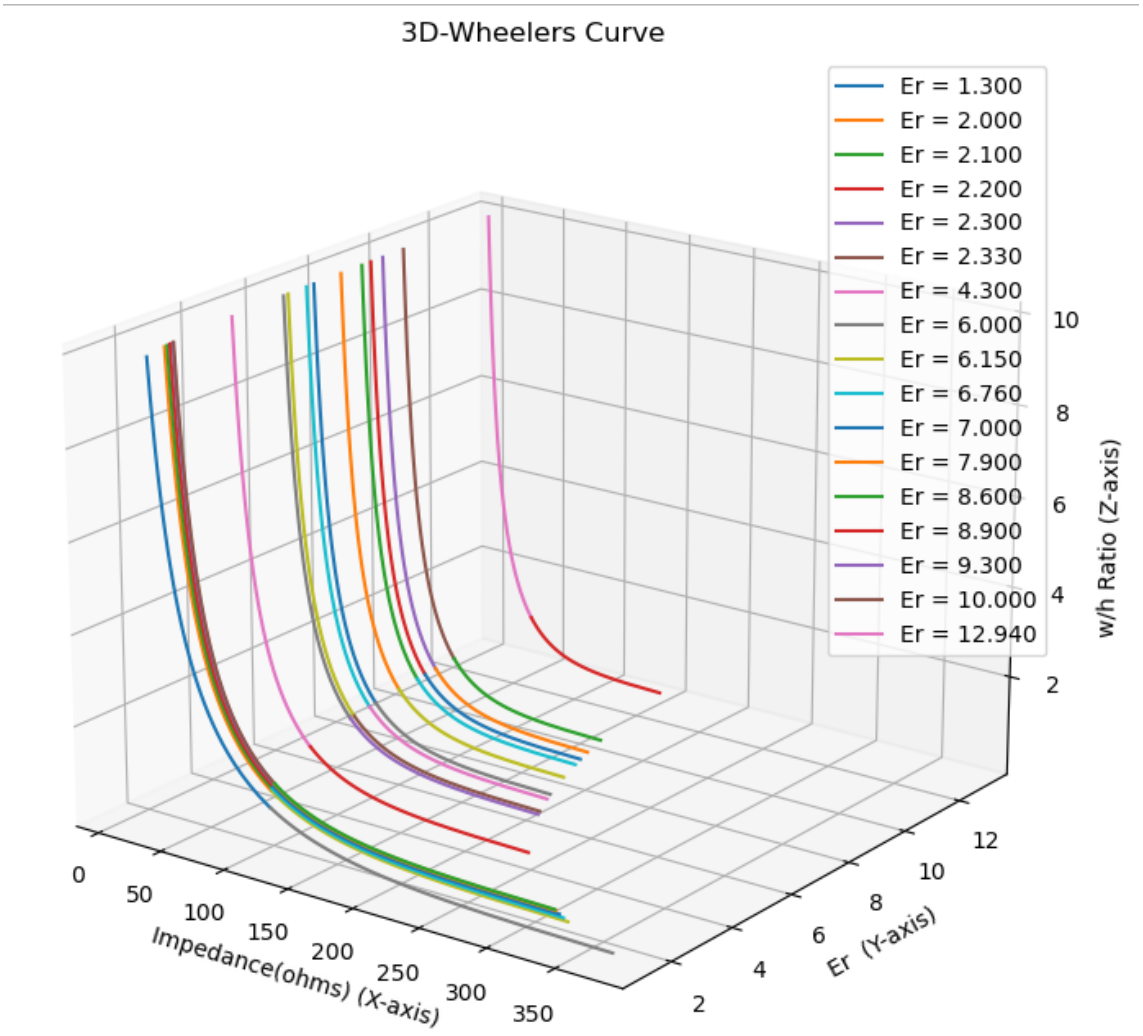
Plot of Wheeler's curve for w/h <1

```
for i in range(n):
    y=np.linspace(0.01,1,50)
    a=np.full(50,k[i])
    #v='C'+str(i)
    kb=((a+1)/2)+(((a-1)/2)*(((1+(12/y))**-0.5)+0.04*(1-y)**2))
    u=(zc/(2*pi*(kb**0.5)))*np.log((8/y)+(y/4))
    ax.plot3D(u, a, y)
```

Input Console

```
>>Enter the number dielectric constants : 17
>>Enter the dielectric constant:1.3
>>Enter the dielectric constant:2
>>Enter the dielectric constant:2.1
>>Enter the dielectric constant:2.2
>>Enter the dielectric constant:2.3
>>Enter the dielectric constant:2.33
>>Enter the dielectric constant:4.3
>>Enter the dielectric constant:6
>>Enter the dielectric constant:6.15
>>Enter the dielectric constant:6.76
>>Enter the dielectric constant:7
>>Enter the dielectric constant:7.9
>>Enter the dielectric constant:8.6
>>Enter the dielectric constant:8.9
>>Enter the dielectric constant:9.3
>>Enter the dielectric constant:10
>>Enter the dielectric constant:12.94
```

3D Wheeler's Curve

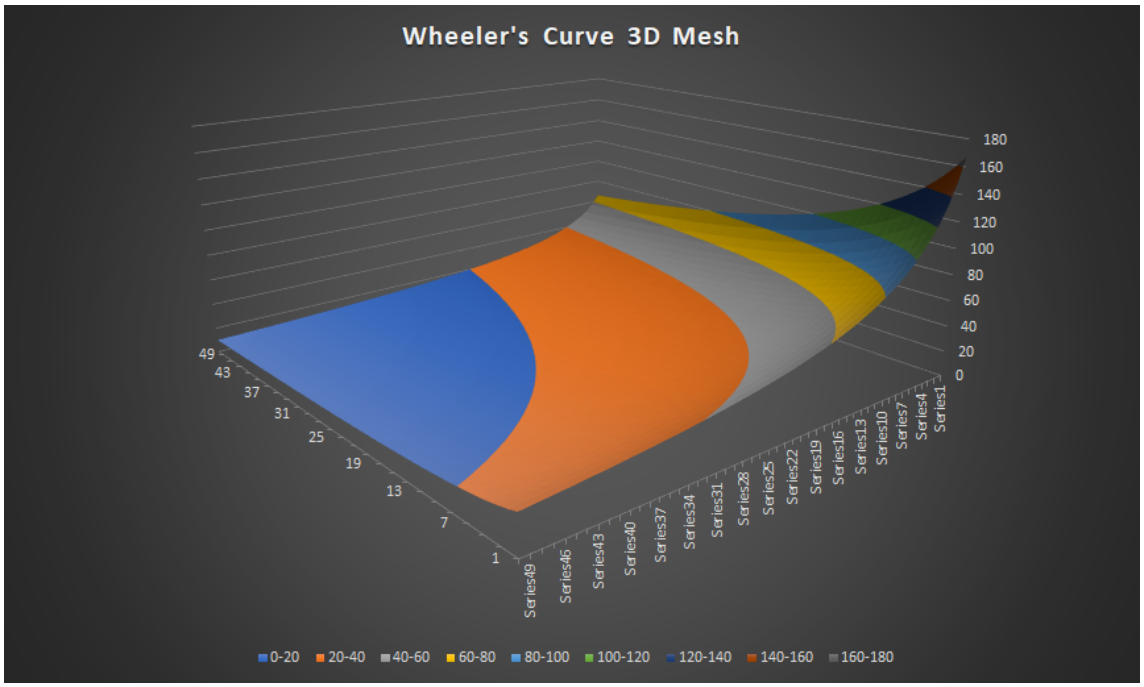


3D Wheeler's Curve is plotted for the user's choice of substrate dielectric constant values

Dielectric Constant Values of Materials Used as Substrate

Material Dielectric Constant values			
Name	Dielec. const.	Name	Dielec. const.
Cotton	1.3	Beryllium Oxide	6.76
PTFE	2	Preperm L700HF	7
Teflon	2.1	Mica	7.9
RT/duroid 5880	2.2	Aluminium Nitride	8.6
Paper	2.3	Sapphire	8.9
RT/duroid 5870	2.33	Alumina	9.3
FR-4	4.3	Preperm 1000	10
Porcelain	6	Gallium Arsenide	12.94
Taconic	6.15		

3D Wheelers Mesh using Excel Surface Chart



Wheelers Curve 3D Mesh using Excel Surface Chart

REFERENCES

1. I.J. Bahl and D. K. Trivedi, "A designer's guide to microstrip line," Microwaves, pp. 90- 96, 1977

2. H. A. Wheeler, "Transmission-Line Properties of Parallel Strips Separated by a Dielectric Sheet," in IEEE Transactions on Microwave Theory and Techniques, vol. 13, no. 2, pp. 172-185, March 1965, doi: 10.1109/TMTT.1965.1125

3. E. Hammerstad and O. Jensen, "Accurate Models for Microstrip Computer-Aided Design," 1980 IEEE MTT-S International Microwave symposium Digest, Washington, DC, USA, 1980, pp. 407-409, doi: 10.1109/MWSYM.1980.1124303.

4. H. A. Wheeler, "Transmission-Line Properties of a Strip on a Dielectric Sheet on a Plane," in IEEE Transactions on Microwave Theory and Techniques, vol. 25, no. 8, pp. 631-647, Aug. 1977, doi: 10.1109/TMTT.1977.1129179.

	Impedance								
	Cotton	PTFE	Teflon	RT/duroid 5	Paper	RT/duroid	FR-4	Sapphire	Alumina
w/h	k=1.3	k=2	k=2.1	k=2.2	k=2.3	k=2.33	k=4.3	k=8.9	k=9.3
0.5	153.4784	131.6821	129.2683	126.9825	124.8138	124.1846	96.3734	69.3936	67.9848
0.69388	135.5478	116.0044	113.8488	111.8092	109.8754	109.3145	84.6235	60.8236	59.5842
0.88776	122.2914	104.4184	102.4544	100.5972	98.83754	98.32738	75.9484	54.5007	53.3864
1.08163	111.60959	95.0878	93.27874	91.56915	89.95025	89.48108	68.9698	49.4182	48.4047
1.27551	103.32729	87.85644	86.16788	84.57308	83.06367	82.62637	63.5664	45.4856	44.5503
1.46939	96.2751	81.7139	80.12919	78.63324	77.21805	76.80818	58.9913	42.162	41.2929
1.66327	90.18933	76.42435	74.93019	73.52038	72.18726	71.80126	55.0626	39.3124	38.5003
1.85714	84.87785	71.81654	70.40215	69.06817	67.80725	67.44224	51.6488	36.8397	36.0772
2.05102	80.19709	67.76293	66.41942	65.15278	63.95595	63.60957	48.6524	34.6721	33.9533
2.2449	76.03751	64.16641	62.88637	61.67999	60.54048	60.21076	45.9994	32.7552	32.07493
2.43878	72.31398	60.95168	59.72884	58.57675	57.48886	57.17414	43.6326	31.0468	30.4011
2.63265	68.95925	58.05938	56.88839	55.78551	54.74439	54.44324	41.5069	29.5139	28.8993
2.82653	65.91948	55.44203	54.31831	53.26026	52.26172	51.97294	39.5865	28.1304	27.5438
3.02041	63.15097	53.06118	51.98075	50.96374	50.00417	49.72671	37.8424	26.8749	26.3138
3.21429	60.61793	50.88537	49.84474	48.86546	47.94171	47.67464	36.2508	25.7301	25.1923
3.40816	58.29066	48.88853	47.88466	46.94019	46.04949	45.79202	34.7922	24.6817	24.1653
3.60204	56.14438	47.04893	46.07911	45.16689	44.30678	44.05818	33.4502	23.7178	23.2211
3.79592	54.15817	45.34826	44.41008	43.52782	42.69614	42.45578	32.2111	22.8285	22.35
3.9898	52.31429	43.77098	42.86231	42.00797	41.20276	40.97008	31.0633	22.0052	21.5435
4.18367	50.59756	42.30384	41.42274	40.59449	39.81401	39.58851	29.9969	21.2407	20.7947
4.37755	48.99492	40.93544	40.08017	39.27636	38.51905	38.30026	29.0033	20.5289	20.0975
4.57143	47.49508	39.65591	38.8249	38.04404	37.30848	37.09599	28.0752	19.8644	19.4466
4.76531	46.08821	38.45669	37.64852	36.88925	36.17414	35.96759	27.2063	19.2425	18.8375
4.95918	44.7657	37.33029	36.54366	35.80475	35.10892	34.90796	26.3909	18.6593	18.2663
5.15306	43.52002	36.27014	35.50386	34.78419	34.10657	33.91088	25.6243	18.1111	17.7294
5.34694	42.34449	35.27045	34.52344	33.82196	33.16157	32.97088	24.9019	17.5949	17.2239
5.54082	41.23321	34.3261	33.59734	32.91311	32.26905	32.08309	24.2202	17.108	16.747
5.73469	40.18094	33.43251	32.72109	32.05324	31.42468	31.24321	23.5756	16.6478	16.2963
5.92857	39.18299	32.58563	31.89071	31.23842	30.6246	30.44739	22.9653	16.2121	15.8697
6.12245	38.23518	31.78183	31.10261	30.46515	29.86534	29.6922	22.3864	15.7992	15.4653
6.31633	37.33373	31.01785	30.35359	29.73026	29.14382	28.97454	21.8367	15.4071	15.0814
6.5102	36.47525	30.29074	29.64077	29.03093	28.45724	28.29166	21.3138	15.0344	14.7164
6.70408	35.65668	29.59785	28.96154	28.36458	27.80308	27.64103	20.8159	14.6796	14.369
6.89796	34.87524	28.93678	28.31354	27.72891	27.17906	27.02038	20.3412	14.3415	14.0378
7.09184	34.12841	28.30536	27.69464	27.12181	26.58312	26.42767	19.8882	14.0189	13.7219
7.28571	33.4139	27.7016	27.10288	26.54137	26.01338	25.86102	19.4552	13.7107	13.4201
7.47959	32.72961	27.12369	26.53649	25.98584	25.4681	25.31872	19.041	13.4159	13.1315
7.67347	32.07363	26.56999	25.99384	25.45362	24.94575	24.79921	18.6445	13.1338	12.8553
7.86735	31.4442	26.03898	25.47347	24.94327	24.44486	24.30107	18.2644	12.8635	12.5906
8.06122	30.83972	25.52927	24.97399	24.45344	23.96414	23.82298	17.8998	12.6043	12.3368
8.2551	30.2587	25.03959	24.49416	23.9829	23.50236	23.36374	17.5497	12.3554	12.0931
8.44898	29.69978	24.56876	24.03283	23.53051	23.05843	22.92225	17.2133	12.1164	11.859
8.64286	29.1617	24.11571	23.58893	23.09524	22.6313	22.49748	16.8897	11.8865	11.634
8.83673	28.6433	23.67942	23.16148	22.67611	22.22003	22.08848	16.5783	11.6654	11.41745
9.03061	28.1435	23.25897	22.74957	22.27224	21.82374	21.69439	16.2784	11.4524	11.2089
9.22449	27.66129	22.85351	22.35236	21.88279	21.44162	21.31439	15.9893	11.2472	11.008
9.41837	27.19576	22.46224	21.96906	21.507	21.07291	20.94773	15.7104	11.0493	10.8143
9.61224	26.74603	22.08441	21.59894	21.14414	20.71691	20.59371	15.4412	10.8584	10.6273
9.80612	26.3113	21.71933	21.24133	20.79356	20.37297	20.25169	15.1813	10.674	10.4468
10	25.89082	21.36637	20.8956	0.45464	20.04047	19.92104	14.9301	10.4958	10.2724