HW3_EE_5601

October 9, 2024

[1]:
```
import skrf as rf
import matplotlib.pyplot as plt
import numpy as np
import cmath as cm
import math
import sympy as sp
from sympy.solvers import solve
pi = math.pi

#Prints out numbers without "np.flat64" displaying
np.set_printoptions(legacy='1.25')
```

[21]:
```
# Constants
epsilon_0 = 8.854*10**-12 #permitivity of free space
mu_0 = 4*pi*10**-7 #permeability of free space
c = 2.998*10**8 #speed of light
```

[46]:
```
# Generic Equations

def freq_to_omega(freq) : #get angular frequency (rad/s)
    return 2*pi*freq

def Np_to_dB(Np) : # converts Nepers to Decibels.
    return Np*20/(np.log(10))
```

0.0.1 Stripline Equations

[29]:
```
#mu_r, epsilon_r = relative properties of medium
#freq = frequency of propagation
def calc_wavenumber(mu_r, epsilon_r, freq) : # wavenumber, k
    return freq_to_omega(freq)*np.sqrt(mu_r*mu_0*epsilon_r*epsilon_0)

# k = wavenumber
def calc_SL_guide_wavelength(k) : # lambda_g
    return 2*pi/k

#frequency = frequency of propagation
#conductance = property of conductor, sigma
```

```
def calc_surface_resistance(frequency,conductance) :
    return np.sqrt(freq_to_omega(frequency)*mu_0/(2*conductance))

#Rs = surface resistance.
#b = dialectric thickness
#t = internal conductor thickness
#W = internal conductor width
def calc_SL_conductor_loss(Zo, epsilon_r, Rs, b, t, W) :
    if (np.sqrt(epsilon_r)*Zo) <= 120 :
        A = 1 - (2*W)/(b-t) + (1/pi)*(b+t)*(np.log((2*b-t)/t))/(b-t)
        return (2.7*(10**-3)*Rs*epsilon_r*Zo)/(30*pi*(b-t))*A
    else :
        B = 1 + (b)*(0.5 + (0.414*t/W) + (1/(2*pi))*np.log(4*pi*W/t))/(0.5*W +
 ↪0.7*t)
        return (0.16*Rs/(Zo*b))*B

#tandelta = dialectric property
#k = wavenumber
def calc_SL_dialectric_loss(k,tandelta) :
    return (k*tandelta/2)

#Zo = characteristic impedence
#epsilon_r = relative permitivity of dialectric
def calc_SL_width(Zo,epsilon_r, b) :
    x = (30*pi)/(np.sqrt(epsilon_r)*Zo) - 0.441
    print(x)
    if (np.sqrt(epsilon_r)*Zo) <= 120 :
        return x*b
    else :
        return (0.85 - np.sqrt(0.6-x))*b

#Zo = characteristic impedence
#epsilon_r = relative permitivity of dialectric
#mu_r = relative permeability of dialectric
#freq = frequency of line
#b = dialectric thickness
#t = thickness of internal conductor.
#tandelta = property of dialectric
#conductance = conductance of conductor.
def calc_SL_total_loss(Zo, epsilon_r, mu_r, freq, b, t, tandelta, conductance) :
    W = calc_SL_width(Zo,epsilon_r, b)
    k = calc_wavenumber(mu_r, epsilon_r, freq)
    Rs = calc_surface_resistance(freq,conductance)
    cond_loss = Np_to_dB(calc_SL_conductor_loss(Zo, epsilon_r, Rs, b, t, W))
    print(f'Conductor Loss = {cond_loss} dB/m')
    dialectric_loss = Np_to_dB(calc_SL_dialectric_loss(k, tandelta))
    print(f'Dialectric Loss = {dialectric_loss} dB/m')
```

```
    return cond_loss + dialectric_loss
```

0.0.2 Microstrip Equations

[84]:
```
#epsilon_r = relative permitivity of dialectric
#Zo = desired characteristic impedence of line
#d = substrate thickness
def calc_MS_width(epsilon_r, Zo,d) :
    A = (Zo/60)*(np.sqrt((epsilon_r+1)/2)) + ((epsilon_r-1)/(epsilon_r+1))*(0.
 ↪23 + (0.11)/epsilon_r)
    B = 377*pi*np.sqrt(epsilon_r)/(2*Zo)
    #print(f'A = {A}')
    #print(f'B = {B}')
    W = (8*np.exp(A)/(np.exp(2*A)-2))*d
    if (W/d) <= 2 :
        print(f'W/d <= 2')
        return W
    else :
        return ((2/pi)*(B - 1 - np.log(2*B-1) + ((epsilon_r-1)/(2*epsilon_r)) *␣
 ↪(np.log(B-1) + 0.39 - (0.61/epsilon_r))))*d

#epsilon_r = relative permitivity of dialectric
#W = width of copper microstrip (not groundplane)
#d = substrate thickness
def calc_MS_epsilon_effective(epsilon_r, W, d) :
    return ((epsilon_r+1)/(2) + ((epsilon_r-1)/(2))*(1/(np.sqrt(1 + 12*(d/W)))))

#epsilon_r = relative permitivity of dialectric
#tandelta = property of dialectric
#W = width of copper microstrip (not groundplane)
#d = substrate thickness
def calc_MS_dialectric_loss(epsilon_r, tandelta, W, d) :
    k0 = (2*pi*freq/c)
    epsilon_e = calc_MS_epsilon_effective(epsilon_r, W, d)
    return ((k0*epsilon_r*(epsilon_e-1)*tandelta)/(2*np.
 ↪sqrt(epsilon_e)*(epsilon_r-1)))

#Rs = surface resistance of conductor
#Zo = characteristic impedence of line.
#W = width of copper microstrip (not groundplane)
def calc_MS_conductor_loss(Rs, Zo, W) :
    return (Rs/(Zo*W))

#Zo = characteristic impedence of line.
#epsilon_r = relative permitivity of dialectric
#freq = frequency of line.
#d = substrate thickness
```

```
#tandelta = property of dialectric
#conductance = conductance of conductor.
def calc_MS_total_loss(Zo, epsilon_r, freq, d, tandelta, conductance) :
    Rs = calc_surface_resistance(freq,conductance)
    W = calc_MS_width(epsilon_r, Zo, d)
    print(f'W = {W}')
    cond_loss = Np_to_dB(calc_MS_conductor_loss(Rs, Zo, W))
    dialectric_loss = Np_to_dB(calc_MS_dialectric_loss(epsilon_r, tandelta, W,␣
 ↪d))
    print(f'Conductor Loss = {cond_loss} dB/m')
    print(f'Dialectric Loss = {dialectric_loss} dB/m')
    return cond_loss + dialectric_loss
```

1 Problems

1.1 Problem 3.19

[50]:
```
#Assume t = 0.01 mm.
Zo = 100 #characteristic impedence
epsilon_r = 2.2 #relative dialectric permitivvity
mu_r = 1 #conductor relative permeability (assumed to be non-magnetic)
frequency = (5*10**9) # given
b = (1.02*10**-3) #dialectric thickness / separation between ground plates
t = (0.01*10**-3) #assuming thickness to be 0.01 mm
tandelta = 0.001 #dialectric property, given.
conductance = (5.813*10**7) #conductance of copper.

print(f'Total Loss = {calc_SL_total_loss(Zo, epsilon_r, mu_r, frequency, b, t,␣
 ↪tandelta, conductance)} dB/m')
```

```
0.19441840048973097
Conductor Loss = 3.376159797161932 dB
Dialectric Loss = 0.6750255560119459 dB
Total Loss = 4.051185353173878 dB/m
```

1.2 Problem 3.20

[85]:
```
Zo = 100 # given
epsilon_r = 2.2 # given
d = 0.51*10**-3 # given
tandelta = 0.001 # given
freq = 5*10**9 # given
conductance = 5.813*10**7 # conductance of copper



calc_MS_total_loss(Zo, epsilon_r, freq, d, tandelta, conductance)
```

```
W/d <= 2
```

```
W = 0.00045708684830194999
Conductor Loss = 3.5017109194273104 dB/m
Dialectric Loss = 0.4770695991861882 dB/m
```

[85]:
```
3.9787805186134984
```

1.3 Problem 3.22

[86]:
```python
#General Parameters
Zo = 50 # Ohms
f = 5*10**9 # 5 GHz
conductance_copper = 5.813*10**7

#Microstrip Parameters
epsilon_r = 2.20
d = 0.16*10**-2 # 0.16 cm
tandelta = 0.001

W_MS = calc_MS_width(epsilon_r, Zo,d)
epsilon_effective = calc_MS_epsilon_effective(epsilon_r, W_MS, d)
length_MS = c/(np.sqrt(epsilon_effective)*f) #m

MS_Loss_per_meter = calc_MS_total_loss(Zo, epsilon_r, f, d, tandelta,␣
 ↪conductance_copper) # dB/m

Microstrip_Loss = MS_Loss_per_meter*length_MS # dB

#Stripline Parameters
epsilon_r = 2.20
b = 0.32*10**-2 #0.32 cm
tandelta = 0.001
t = 0.01*10**-3 # 0.01 mm
mu_r = 1

length_LS = c/(np.sqrt(epsilon_r)*f)
SL_Loss_per_meter = calc_SL_total_loss(Zo, epsilon_r, mu_r, f, b, t, tandelta,␣
 ↪conductance_copper)

Stripline_Loss = SL_Loss_per_meter*length_LS # dB

print('')
print(f'Microstrip Loss = {Microstrip_Loss} dB on {length_MS} meters of line')
print(f'Stripline Loss = {Stripline_Loss} dB on {length_LS} meters of line')
```

```
W = 0.014090483807364458
Conductor Loss = 0.22718680631661273 dB/m
Dialectric Loss = 0.5856902029696914 dB/m
0.8298368009794619
```

```
Conductor Loss = 0.22203596971226652 dB
Dialectric Loss = 0.6750255560119459 dB

Microstrip Loss = 0.03454790696972203 dB on 0.04250078003811999 meters of line
Stripline Loss = 0.03626373348556924 dB on 0.04042502375329599 meters of line
```