HW4_EE_5601_code

October 21, 2024

[9]:
```
import numpy as np
import math
pi = math.pi

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

0.1 Problem 4.14

a) is network lossless?

[10]:
```
def pol2comp(mag,ang) :
    a = mag*math.cos(math.radians(ang))
    b = mag*math.sin(math.radians(ang))
    if a < 10**-15 :
        a = 0
    if b < 10**-15 :
        b = 0
    output = complex(a,b)
    return output

S11 = pol2comp(0.178,90)

S = np.matrix([[pol2comp(0.178,90),pol2comp(0.6,45),pol2comp(0.
 ↪4,45),0],[pol2comp(0.6,45),0,0,pol2comp(0.3,-45)],[pol2comp(0.
 ↪4,45),0,0,pol2comp(0.5,-45)],[0,pol2comp(0.3,-45),pol2comp(0.5,-45),0]])
S_conj = S.conjugate()
S_squared =  np.matmul(S,S_conj)

print_matrix = S
print('S = ')

for i in range(0,len(print_matrix)) :
    string = '|'
    for j in range(0,len(print_matrix)) :
        val = 'S' + str(i+1) + str(j+1)
        string = string + val
        if j != len(print_matrix)-1 : string = string + ','
```

```
        if len(val) < 15 :
            l = len(val)
            while l<15 :
                string = string + ' '
                l+=1
        if j == len(print_matrix)-1 : string = string + '|'
    print(string)

print(' = ')

for i in range(0,len(print_matrix)) :
    string = '|'
    for j in range(0,len(print_matrix)) :
        val = str(np.round(print_matrix[i,j],3))
        string = string + val
        if j != len(print_matrix)-1 : string = string + ','
        if len(val) < 15 :
            l = len(val)
            while l<15 :
                string = string + ' '
                l+=1
        if j == len(print_matrix)-1 : string = string + '|'
    print(string)
```

```
S =
|S11,            S12,            S13,            S14            |
|S21,            S22,            S23,            S24            |
|S31,            S32,            S33,            S34            |
|S41,            S42,            S43,            S44            |
 =
|0.178j,         (0.424+0.424j), (0.283+0.283j), 0j             |
|(0.424+0.424j), 0j,             0j,             (0.212+0j)     |
|(0.283+0.283j), 0j,             0j,             (0.354+0j)     |
|0j,            (0.212+0j),     (0.354+0j),     0j             |
```

S is not identiy matrix, therefore this network is not lossless.

b) Network is reciprocal.

c) Return Loss?

[11]:
```
S11 = 0.178
RL = -20*math.log10(S11)
print(f'c) Return Loss = {RL}')
```

```
c) Return Loss = 14.99159995382212
```

d) Insertion Loss and

[12]:
```
S42 = 0.3
IL = -20*math.log10(S42)
print(f'Insertion Loss = {np.round(IL,2)} dB')
print('Phase Delay = 45 degrees')
```

```
Insertion Loss = 10.46 dB
Phase Delay = 45 degrees
```

e) Reflection coefficient @P1, with short @P3, and all other ports terminated with matched loads.

[13]:
```
S11 = S[0,0]
S13 = S[0,2]
S31 = S[2,0]

Ref = S11 - S13*S31
print(f'Reflection Coefficient = {np.round(Ref,3)}')
```

```
Reflection Coefficient = 0.018j
```

0.2 Problem 4.20

[25]:
```
ZL = 100
Zo = 70.7
Zg = 100
Vo= 30
Bl = math.radians(90)

Zin = np.round(Zo*(complex(ZL,(Zo*math.tan(Bl))))/(complex(Zo,(ZL*math.
 ↪tan(Bl)))),1)

a = Vo*np.sqrt(Zin)/(Zin+Zg)
PL = 0.5*(abs(a)**2)
print(f'a = {np.round(a,3)}')
print(f'PL = {np.round(PL,3)}')
```

```
a = (1.414+0j)
PL = 1.0
```

0.3 Problem 4.24

[17]:
```
#Source Parameters
R = 50

A1 = 1
B1 = R
C1 = 0
D1 = 1
ABCD1 = np.matrix([[A1, B1],[C1,D1]])
```

```
print('Source Resistor ABCD parameters')
print(f'{A1} {B1}')
print(f'{C1} {D1}')
print('')


#Transformer Parameters
N = 1/2

A2 = N
B2 = 0
C2 = 0
D2 = 1/N
ABCD2 = np.matrix([[A2,B2],[C2,D2]])

print('Transformer ABCD parameters')
print(f'{A2} {B2}')
print(f'{C2} {D2}')
print('')


#Tline parameters
Zo3 = 50 #ohms
Bl3 = 90 #degrees

Yo3 = 1/Zo3
A3 = complex(round(math.cos(math.radians(Bl3)),4),0)
B3 = complex(0,round(Zo3*math.sin(math.radians(Bl3)),4))
C3 = complex(0,round(Yo3*math.sin(math.radians(Bl3)),4))
D3 = complex(round(math.cos(math.radians(Bl3)),4),0)
ABCD3 = np.matrix([[A3, B3],[C3,D3]])

print('T-line ABCD parameters')
print(f'{A3} {B3}')
print(f'{C3} {D3}')
print('')


#Load Resistor
R_L = 25

A4 = 1
B4 = 0
C4 = 1/R_L
D4 = 1
ABCD4 = np.matrix([[A4,B4],[C4,D4]])
```

```
print('Load ABCD parameters')
print(f'{A4} {B4}')
print(f'{C4} {D4}')
print('')


#Multiply all ABCD matrices together in order.
ABCD = np.matmul(ABCD1,ABCD2)
ABCD = np.matmul(ABCD,ABCD3)
ABCD = np.matmul(ABCD,ABCD4)

print('Final System Parameters')
print(ABCD)
```

```
Source Resistor ABCD parameters
1 50
0 1

Transformer ABCD parameters
0.5 0
0 2.0

T-line ABCD parameters
0j 50j
0.02j 0j

Load ABCD parameters
1 0
0.04 1

Final System Parameters
[[0. +3.j   0.+25.j  ]
 [0. +0.04j 0. +0.j  ]]
```