第 13 回報告書　　2021 年 12 月 1 日

分布定数線路の周波数特性

愛媛大学工学部
8531037m
祖父江匠真

1　はじめに

分布定数線路の周波数特性を調べた.

2　分布定数線路の周波数特性

図 1 の同軸ケーブルの仕様表より, 3D-2V ケーブルの減衰特性の折れ線グラフを図 2 に示す.

・ 同軸ケーブルの仕様（参考：藤倉電線資料）

引用ＵＲＬ
■ http://www.orixrentec.co.jp/tmsite/know/know_doujiku53.html
■ http://home3.highway.ne.jp/welcome/tv/cable/syurui.htm

名称	外径 (mm)	特性インピーダンス (Ω)	静電容量 (nF/km)	波長短縮率 (%)	減衰特性 (dB/km)		
					1MHz	10MHz	200MHz
1.5C-2V	2.9	75±3	67	67	27	82	390
3C-2V	5.6	75±3	67	67	12	40	195
5C-2V	7.5	75±3	67	67	7.6	25	125
1.5D-2V	2.9	50±2	100	67	27	85	420
3D-2V	5.5	50±2	100	67	13	44	220
5D-2V	7.5	50±2	100	67	7.3	26	125
8D-2V	11.5	50±2	100	67	4.8	17	85
RG58/U	5.0	53.5	94	67	13	42	200
RG58A/U	5.0	50	102	67	14	48	230

図 1: 同軸ケーブルの仕様

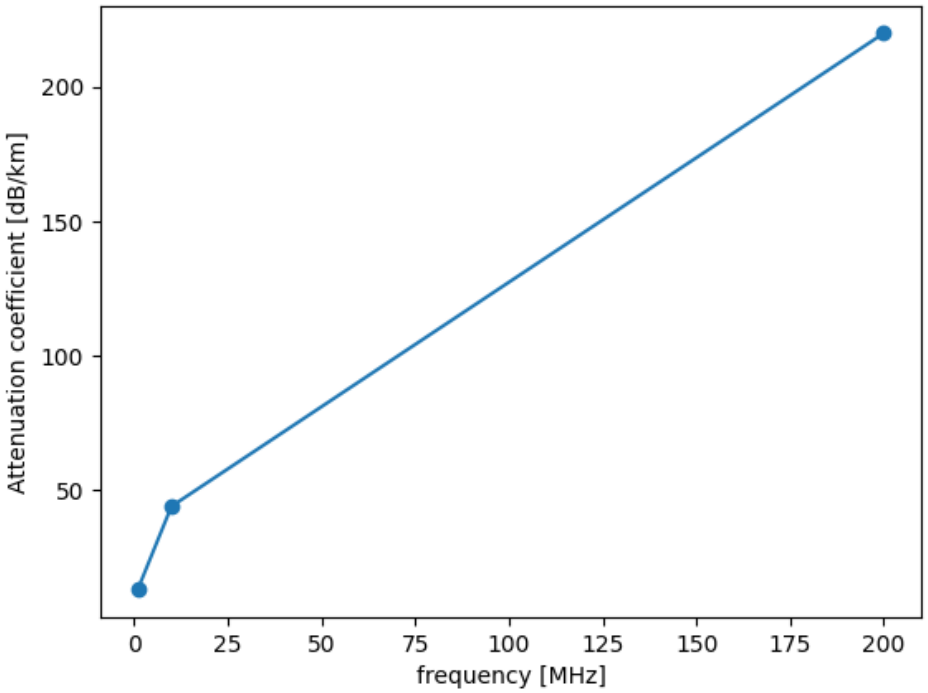

図 2: 減衰特性の折れ線グラフ

伝搬定数 γ ($= \alpha + j\beta$) における減衰定数 α については, 図 2 の周波数に対応した値を計算に用いた.

図 3 の回路図について, 周波数特性を調べる.

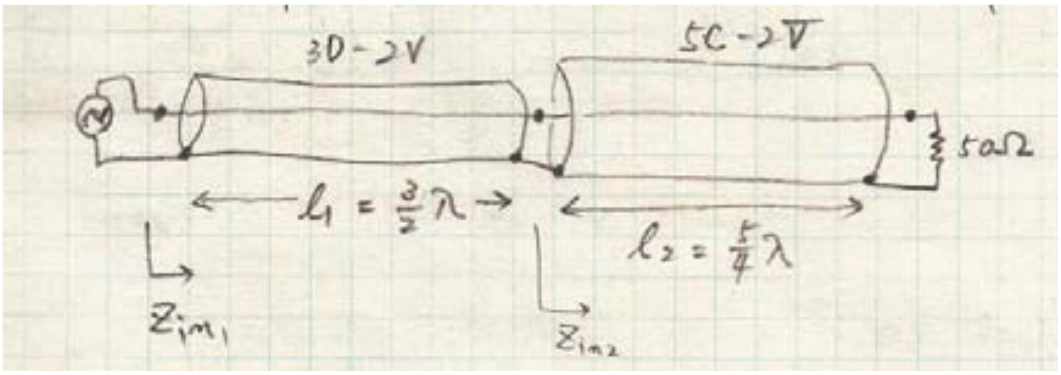

図 3: 回路図

Python で伝達関数の計算, 周波数特性のグラフを出力するプログラムをソースコード 1 に示す.

ソースコード 1: 周波数特性

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


def calculateTheta(cableLength, frequency, alphas):
    """
    伝搬定数γと同軸ケーブルの長さ l の積を求める

    Parameters
    ----------
    cableLength : float
        同軸ケーブルの長さ
    frequency : float
        周波数
    alphas : float
        減衰定数αの離散値のリスト
    """
    alpha = calculateAlpha(frequency, alphas)
    # 無損失の場合のみ成り立つ計算の仕方
    FRACTIONAL_SHORTENING = 0.67 # 波長短縮率(同軸ケーブルは一律で 0.67)
    SPEED_OF_LIGHT = 3 * 10 ** 8 # 光速
    beta = 2 * np.pi * frequency / (SPEED_OF_LIGHT * FRACTIONAL_SHORTENING)
    gamma = alpha + beta * 1j
    return gamma * cableLength


def createFMatrixForDcc(Z0, theta):
    """
    分布定数回路の F 行列を求める

    Parameters
    ----------
    Z0 : float
        同軸ケーブルの入力インピーダンス
    theta : float
        伝搬定数γと同軸ケーブルの長さ l の積
    """
    return np.array(
        [
            [np.cosh(theta), Z0 * np.sinh(theta)],
            [np.sinh(theta) / Z0, np.cosh(theta)],
        ]
    )


# f からαを求める
def calculateAlpha(frequency, alphas):
    """
    周波数に対応した減衰定数を取得する

    Parameters
    ----------
```

```
    frequency : float
        周波数
    alphas : float
        減衰定数αの離散値のリスト
    """
    if frequency < 10 == True:
        coef = (alphas[1] - alphas[0]) / (10 - 1)
        intercept = alphas[0] - coef * 1
        return coef * frequency + intercept
    else:
        coef = (alphas[2] - alphas[1]) / (200 - 10)
        intercept = alphas[1] - coef * 10
        return coef * frequency + intercept


def calculateInputImpedanceByFMatrix(Zr, Z0, cableLength, frequency, alphas):
    """
    受電端に抵抗を接続した分布定数回路の入力インピーダンスを求める
    与えられた周波数から入力インピーダンスを求める

    Parameters
    ----------
    Zr : float
        受電端のインピーダンス
    Z0 : float
        同軸ケーブルの入力インピーダンス
    cableLength : float
        同軸ケーブルのケーブル長
    frequency : float
        周波数
    alphas : float
        減衰定数αの離散値のリスト
    """
    # γlを求める
    theta = calculateTheta(cableLength, frequency, alphas)
    # 分布定数回路のF行列
    f_matrix_dcc = createFMatrixForDcc(Z0, theta)
    # 受電端のZrのF行列
    f_matrix_Zr = np.array(
        [
            [1, 0],
            [1 / Zr, 1],
        ]
    )

    # 受信端にZrを接続した場合のf行列
    f_matrix = np.dot(f_matrix_dcc, f_matrix_Zr)

    return abs(f_matrix[0, 0] / f_matrix[1, 0])


def createTransferFunction(Zr, Z0, cableLength, frequency, alphas):
    """
    受電端に抵抗を接続した分布定数回路の伝達関数を求める

    Parameters
    ----------
```

```
111         Zr : float
112             受電端のインピーダンス
113         Z0 : float
114             同軸ケーブルの入力インピーダンス
115         cableLength : float
116             同軸ケーブルのケーブル長
117         frequency : float
118             周波数
119         alphas : float
120             減衰定数αの離散値のリスト
121         """
122         theta = calculateTheta(cableLength, frequency, alphas)
123 
124         R1 = 0 # 入力側の抵抗は0で考える
125         R2 = Zr
126 
127         f_matrix_dcc = createFMatrixForDcc(Z0, theta)
128         A = f_matrix_dcc[0][0]
129         B = f_matrix_dcc[0][1]
130         C = f_matrix_dcc[1][0]
131         D = f_matrix_dcc[1][1]
132 
133         return 1 / (A + B / R2 + R1 * C + (R1 / R2) * D)
134 
135 
136 # 5C-2V
137 l2 = 5 / 4
138 Z02 = 75 # 同軸ケーブルのインピーダンス
139 
140 alpha2_1mhz = 7.6
141 alpha2_10mhz = 25
142 alpha2_200mhz = 125
143 alphas2 = [alpha2_1mhz, alpha2_10mhz, alpha2_200mhz]
144 
145 Zr = 50 # 受端のインピーダンス
146 
147 
148 # 3D-2V
149 l1 = 3 / 2
150 Z01 = 50 # 同軸ケーブルのインピーダンス
151 
152 alpha1_1mhz = 13
153 alpha1_10mhz = 44
154 alpha1_200mhz = 220
155 alphas1 = [alpha1_1mhz, alpha1_10mhz, alpha1_200mhz]
156 
157 # 単位はMHz (= 1 x 10^6 Hz)
158 frequencies = range(1, 201)
159 # frequencies = range(1, 11)
160 
161 transferFunctions1 = []
162 # 周波数ごとに伝達関数を求める
163 for frequency in frequencies:
164     # 5C-2V + Zr の回路の入力インピーダンスを求める
165     inputImpedance2 = calculateInputImpedanceByFMatrix(Zr, Z02, l2, frequency,
            alphas2)
166 
```

```
        # 回路全体の伝達関数を求める
        # transferFunctions2 = createTransferFunction(Zr, Z02, l2, frequency, alphas2)
        transferFunction1 = createTransferFunction(
            inputImpedance2, Z01, l1, frequency, alphas1
        ) # 5C−2V + Zr の回路の入力インピーダンスを受電端側の抵抗 Zr とする

        transferFunctions1.append(transferFunction1)

# 周波数特性
fig, ax = plt.subplots()
ax.plot(frequencies, list(map(abs, transferFunctions1)), label="Gain")
ax.set_xlabel("frequency [MHz]")
ax.set_ylabel("Gain [dB]")
plt.show()
```

ソースコード 1 によって得られた周波数特性を図 4 に示す.

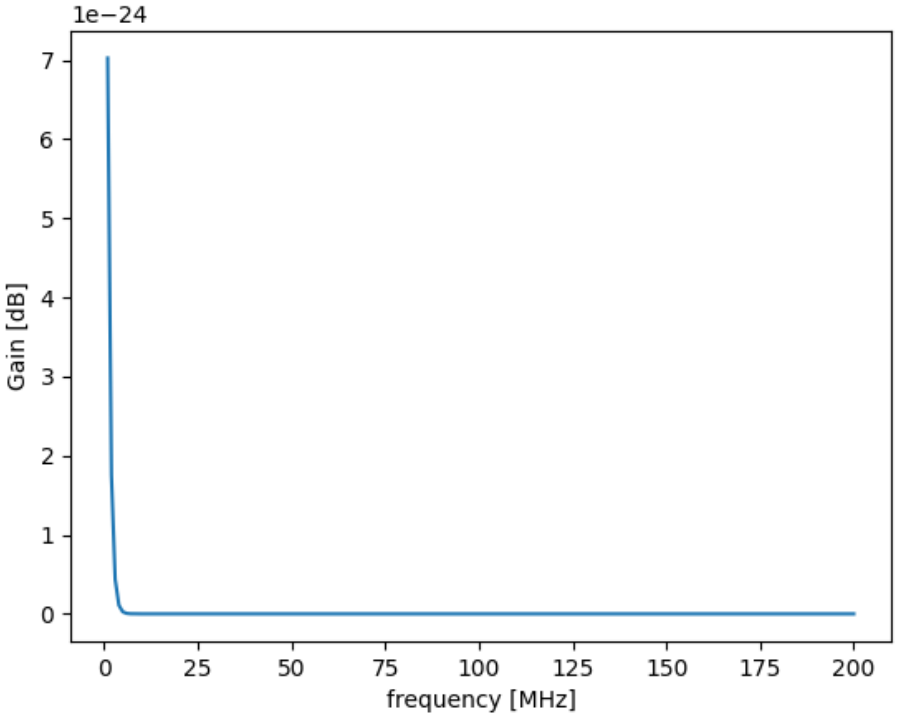

図 4: 周波数特性 (1MHz ˜200MHz)

図 4 より, 周波数が 10MHz を超えた辺りからゲインが 0 になっているのでプロットする周波数の値域を 1MHz ˜10MHz に変更した物を図 5 に示す.

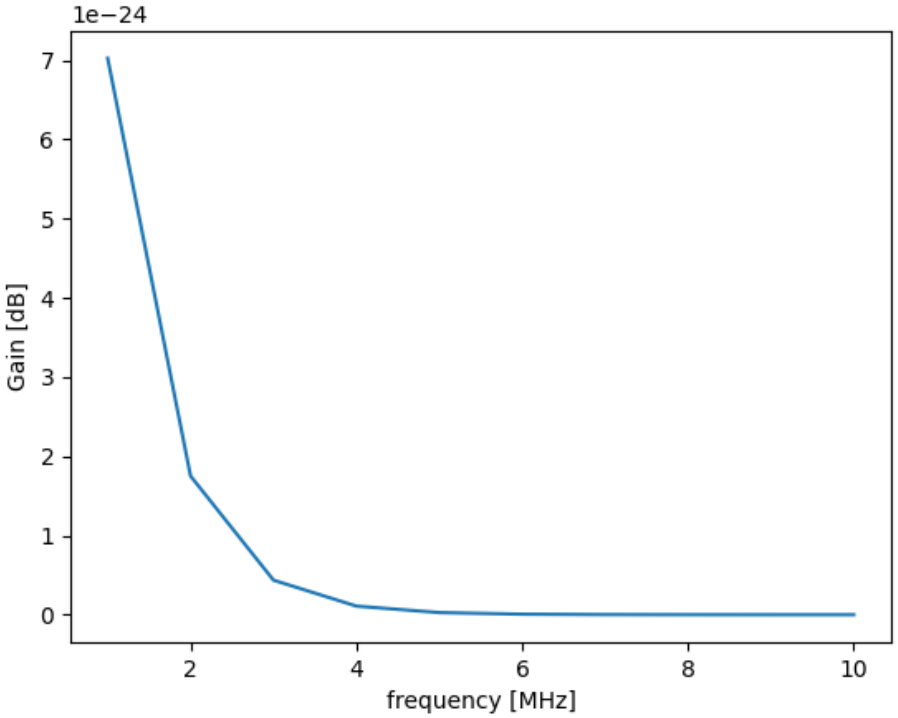

図 5: 周波数特性 (1MHz ˜10MHz)

3　おわりに

今回は分布定数線路の周波数特性を調べたが, ローパスフィルターの周波数特性のようにはならなかった.

参考文献

[1] 都築, ”2020Q4-応用通信工学 II-都築 ”, moodle 内, 参照 October 20,2021.