Submission - Exercise [1] [Group_15]

Mobile Communication

[Mst. Mahfuja Akter , s6msakte@uni-bonn.de]
[Mahpara Hyder Chowdhury, s6machow@uni-bonn.de]
[M Shahzaib, s6mmshah@uni-bonn.de]

May 5, 2019

1 Free Space and Two-Ray Ground propagation

In lecture section 2.1 the *Free Space Propagation Model* (FSP) and the *Two-Ray Ground Propagation Model* (TRG) have been introduced. If we ignore the optional gain values, both models can be defined as follows:

$$FSP : P_r = \frac{P_t \cdot \lambda^2}{d^2 \cdot (4\pi)^2}$$

$$TRG : P_r = \frac{P_t \cdot h_t^2 \cdot h_r^2}{d^4}$$

Variable definitions: P_t – Transmitted power, P_r – Received power, λ – Wavelength, d – Distance, h_t – Transmitter antenna height, h_r – Receiver antenna height

1. In the study of wireless communication it is often desirable to express the attenuation independently of the absolute power levels. Instead we use the *Path loss*, which is defined as:

$$PL[dB] = 10 \cdot log_{10} \frac{P_t}{P_r}$$

Using the definitions of FSP and TRG given above, derive a formula to calculate the path loss PL_{FSP} and PL_{TRG} independently of a specific P_t and P_r value.

Answer: We know, by definition

$$FSP : P_r = \frac{P_t.\lambda^2}{d^2.(4\pi)^2}$$

$$TRG : P_r = \frac{P_t.h_t^2.h_r^2}{d^4}$$

Therefore,

$$PL_{FSP} = 10log_{10}\frac{P_t}{\frac{P_t.\lambda^2}{d^2.(4\pi)^2}} = 10log_{10}P_t\frac{d^2.(4\pi)^2}{P_t.\lambda^2} = 10log_{10}\frac{d^2.(4\pi)^2}{\lambda^2}$$

$$PL_{TRG} = 10log_{10}\frac{P_t}{\frac{P_t.h_t^2.h_r^2}{d^4}} = 10log_{10}P_t\frac{d^4}{P_t.h_t^2.h_r^2} = 10log_{10}\frac{d^4}{h_t^2.h_r^2}$$

Hereby, We come to know that path loss for both FSP and TRG are independently specific for P_t and P_r values.

2. TRG does not provide meaningful results for small distances. Therefore it is a common practice to define a *crossover distance* d_c and use FSP for distances $d \leq d_c$, TRG for $d > d_c$. For this exercise sheet, we assume:

$$d_c = \frac{4\pi \cdot h_t \cdot h_r}{\lambda}$$

Prove that there is a smooth transition between the two models at the crossover distance, i.e. prove that both models yield equal results at d_c.

Answer: From the definition of the FSP and TRG for Received power P_r we can write

$$\frac{P_t.\lambda^2}{d^2.(4\pi)^2} = \frac{P_t.h_t^2.h_r^2}{d^4} \Rightarrow \frac{P_t.\lambda^2}{P_t.h_t^2.h_r^2} = \frac{d^2.(4\pi)^2}{d^4} \Rightarrow \frac{\lambda^2}{h_t^2.h_r^2} = \frac{(4\pi)^2}{d^2}$$

Now considering crossover distance,

$$\frac{\lambda^2}{h_t^2.h_r^2} = \frac{(4\pi)^2}{\frac{4\pi.h_t.h_r}{\lambda}} \Rightarrow \frac{\lambda^2}{h_t^2.h_r^2} = \frac{(4\pi.)}{h_t.h_r} \Rightarrow \frac{\lambda}{h_t.h_r} = 4\pi \Rightarrow \frac{4\pi.h_t.h_r}{\lambda} = 0$$

2 Comparing two path loss models

The MoCo lecture also defines a path loss model based on a known reference loss and an environment-dependent path loss exponent. For this exercise sheet we use a specific variant of this model, the *Three Log Distance Model* (TLD), which defines three different distance fields and their corresponding exponents. Specifically, TLD is defined as:

$$PL_{TLD} = \begin{cases} 0 & d < d_0 \\ L_0 + 10 \cdot n_0 \log_{10}(\frac{d}{d_0}) & d_0 \leq d < d_1 \\ L_0 + 10 \cdot n_0 \log_{10}(\frac{d_1}{d_0}) + 10 \cdot n_1 \log_{10}(\frac{d}{d_1}) & d_1 \leq d < d_2 \\ L_0 + 10 \cdot n_0 \log_{10}(\frac{d_1}{d_0}) + 10 \cdot n_1 \log_{10}(\frac{d_2}{d_1}) + 10 \cdot n_2 \log_{10}(\frac{d}{d_2}) & d_2 \leq d \end{cases}$$

Variable definitions: d – Distance, d_0, d_1, d_2 – Three distance fields, n_0, n_1, n_2 –path loss distance exponent for each field, L_0 – path loss at reference distance d_0

1. Write a program that implements the path loss calculation using the combined FSP and TRG models with crossover distance dc. In addition, the program should be able to calculate the path loss according to the TLD model.
Answer:
According to our Path loss formula, we made some functions which calculate path loss for specific distance.
Here is our function implementation:

```python
# 2.1
import math
import pandas as pd
import matplotlib
import numpy as np
%matplotlib inline

def PL_FSP(lambda_, d):
    return 10 * mylog10((math.pow(d, 2) * math.pow(4 * math.pi, 2)) / math.pow(lambda_, 2))

def PL_TRG(ht, hr, d):
    return 10 * mylog10(math.pow(d, 4) / (math.pow(ht, 2) * math.pow(hr, 2)))

def dc(lambda_, ht, hr):
    return (4 * math.pi * hr * ht) / lambda_

def PL_TLD(d, d0, d1, d2, L0, n0, n1, n2):
    if d < d0:
        return 0
    elif d0 <= d and d < d1:
        return L0 + 10 * n0 * mylog10(d/d0)
    elif d1 <= d and d < d2:
        return L0 + 10 * n0 * mylog10(d1/d0) + 10 * n1 * mylog10(d/d1)
    elif d2 <= d:
        return L0 + 10 * n0 * mylog10(d1/d0) + 10 * n1 * mylog10(d2/d1) + 10 * n2 * mylog10(d/d2)

def mylog10(x):
    return  math.log10(max(0.00000000000000000000001,x))
```

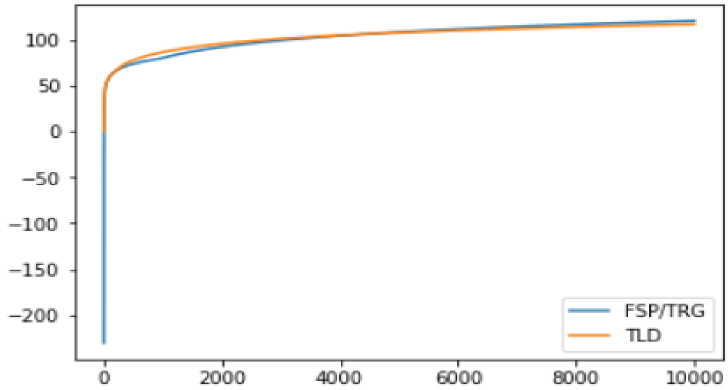
```python
def PL_FSP_func(d):
    lambda_ = 1.35
    ht = 50
    hr = 2
    d0, d1, d2 = 1, 200, 500
    n0, n1, n2 = 2.0, 3.0, 3.0
    L0 = 19.377
    dc_val = dc(lambda_, ht, hr)
    return PL_FSP(lambda_, d) if d <= dc_val else PL_TRG(ht, hr, d)


def PL_TLD_func(d):
    lambda_ = 1.35
    ht = 50
    hr = 2
    d0, d1, d2 = 1, 200, 500
    n0, n1, n2 = 2.0, 3.0, 3.0
    L0 = 19.377
    dc_val = dc(lambda_, ht, hr)
    return PL_TLD(d, d0, d1, d2, L0, n0, n1, n2)
```

2. Using your program, generate a plot of the calculated path loss of both FSP/TRG and TLD for the distance range from 0 meters up to 10000 meters. Use the following parameters for your calculations and plots:

   - $\lambda : 1.35m$
   - $h_t : 50m$
   - $h_r : 2m$
   - $d_0, d_1, d_2 : 1m, 200m, 500m$
   - $n_0, n_1, n_2 : 2.0, 3.0, 3.0$
   - $L_0 : 19.377dB$

Answer:
Using above functions, we measured path loss from 0 meter to 1000 meters and plot those data like below:



We have checked our implemented function with some specific distance also (e.g. 1000,5000,10000)
Here is our measured data:

```
# 2.3
out = []

for d in (1000,5000,10000):
    out.append([d,PL_FSP_func(d), PL_TLD_func(d)])

df = pd.DataFrame(out, columns=['Distance','FSP/TRG', 'TLD'])

print(df)
```

```
   Distance   FSP/TRG       TLD
0      1000   80.0000   86.3667
1      5000  107.9588  107.3358
2     10000  120.0000  116.3667
```

3. Give a brief description of your implementation and the resulting plot.
Answer:
From our measured data and plot, we can see that the combination of FSP

and TRG gives almost same path loss as TLD. Though FSP works with lower distance and TRG works with high distance, combination of these two gives better result if we can apply crossover distance value properly.
Besides of FSP and TRG, TLD works for all distance and gives similar path loss from other two. So, we can consider TLD as a complete method.

3 Reality check

On the MoCo web page we provide the file *ex01.csv* containing a set of real world measurements conducted inside of a moving car. Each line of the file contains a *timestamp*, the *receiver geo-coordinates* and the *relative signal strength* (a unitless value between 0 – no signal reception, and 1 – perfect signal reception). The transmitter is a fixed antenna located at $50°42'25.96"N, 7°5'49.13"E$.

1. Extend your program so that it can calculate the path loss based on two geographic coordinates.

Answer:
We have converted the geographic coordinates data into latitude, longitude and distance and pass those distance into path loss function.
Here is our implementation:

```
#3.1
data = pd.read_csv("ex1.csv")
data.columns = ['timestamp', 'x','y','rs']
data["start_x"], data["start_y"] =  50.4202596, 7.504913

# lats
data["a"] = (data.x - data.start_x) * 111320
# longs
data["b"] = (data.y - data.start_y) * 111320 * data["a"].apply(np.cos)


#distance calulation
data.a = data.a.apply(np.square) # dist in x dimention
data.b = data.b.apply(np.square) # dist in y dimention
data["d"] = data.a + data.b
data.d = data.d.apply(np.sqrt) # dist

#pathloss calculation
data["FSP"] = data.d.apply(PL_FSP_func) # path loss FSP calc.
data["TLD"] = data.d.apply(PL_TLD_func) # path loss TLD calc.
```

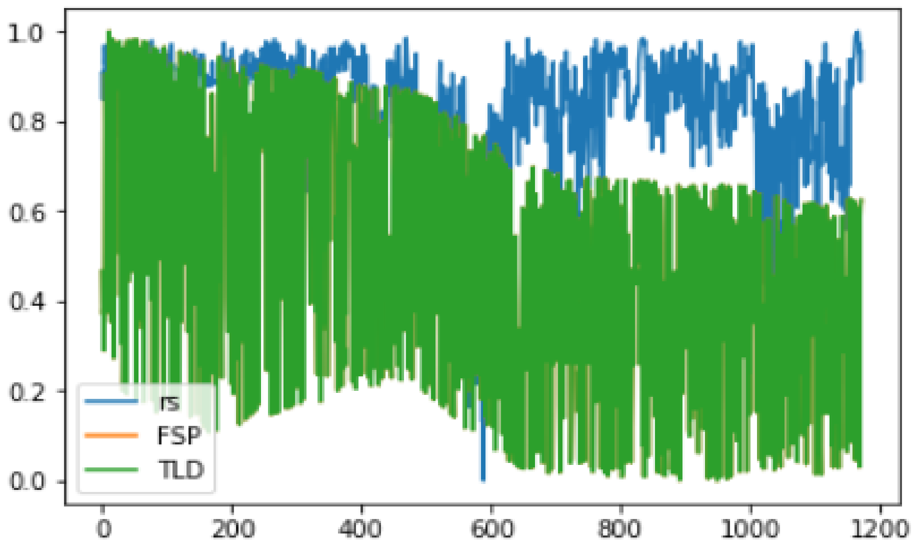
2. Compute the expected path loss for the moving car using both of the models defined before. Produce a plot which shows the expected path loss for both models in comparison to the conducted measurements.
Answer:
We have computed path loss from those distances and normalize those values from 0 to 1. And noticed that FSP/TRG and TLD almost overlaps with each other in normalized form. Besides relative signal works almost inversely with FSP/TRG and TLD. To see the better comparison with conducted relative signal we have inverted this signal and make a plot.
Here is our implementation:

```
#3.2
# normalization
data["FSP"] = (data["FSP"] - min(data["FSP"])) / (max(data["FSP"]) - min(data["FSP"]))
data["TLD"] = (data["TLD"] - min(data["TLD"])) / (max(data["TLD"]) - min(data["TLD"]))
data["rs"] = (data["rs"] - min(data["rs"])) / (max(data["rs"]) - min(data["rs"]))

# inversing relative strength to see the correlation clearly
data.rs = np.abs(data.rs - 1)

data[["rs", "FSP", "TLD"]].plot()
```

We found the plot like below:



3. Do the path loss models correctly represent the real world signal propagation? Give reasons why this might (not) be the case.
Answer:
No, this model does not represent the real world signal propagation. When relative strength is high, path loss supposed to be low. Here we inverse our relative strength data but till we see that the path loss data is not relative.