**Experiment-10**

**Aim:** To implement concept code for the fading effect for cellular mobile communication.

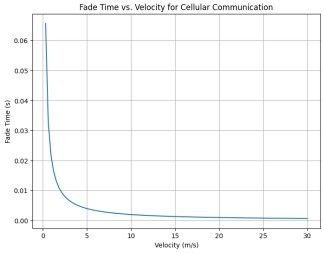
**Activities:**

1. To Develop and implement concept code for analytical calculation of fade time (duration) using fadetime equations.
2. To plot the graph for fadetime against the velocity for analytical analysis.

# Theory:

* Cellular Communication fade time typically refers to the duration for which the received signal strength experience significant fluctuations or fade due to various factors like multipath propagation, shadowing and interference. Fade time can be described in different ways based on the context.
* **Fast Fading:** This refers to rapid and short-term fluctuations in signal strength that occur due to multipath propagation. Fast fading is often caused by the signal taking multiple paths to reach the receiver, resulting in constructive and destructive interference. The fade time in this context is very short, typically on the order of milliseconds to microseconds.
* **Slow Fading:** Slow fading, on the other hand, refers to longer-term variations in signal strength caused by factors like changes in the distance between the transmitter and receiver, terrain, and large obstacles. In this case, the fade time is relatively longer, ranging from seconds to minutes or more.

# Expected Output:



# Answer:

**For activity 1:**

Code:

import numpy as np import matplotlib.pyplot as plt

# Simulation parameters

num\_samples = 1000 # Number of time samples fade\_duration = 100 # Duration of fading in milliseconds velocity = 30 # Velocity of the mobile terminal (m/s)

# Generate Rayleigh fading amplitudes

fade\_amplitudes = np.random.exponential(scale=fade\_duration, size=num\_samples)

# Create a time vector

time = np.linspace(0, fade\_duration, num\_samples)

# Set the threshold for fade detection (adjust as needed) fade\_threshold = 0.1

# Initialize variables for fade time calculation fade\_time = 0 # Total fade time

in\_fade = False # Flag to track whether we are in a fade event fade\_event\_start = 0 # Start time of the current fade event

# Analyze fade events for i in range(num\_samples): if fade\_amplitudes[i] < fade\_threshold: if not in\_fade: in\_fade = True fade\_event\_start = time[i]

elif in\_fade:

in\_fade = False

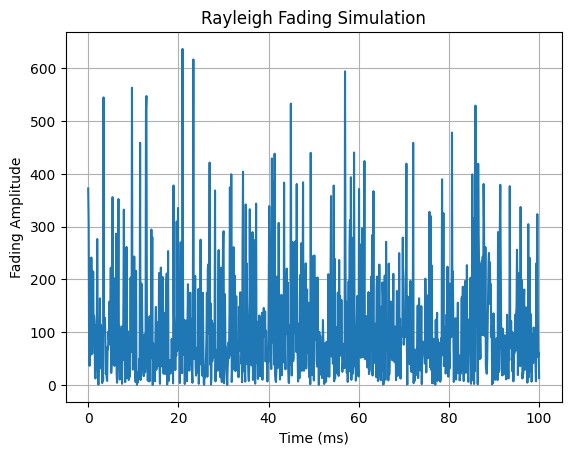
fade\_time += time[i] - fade\_event\_start

# Plot the fade amplitudes plt.plot(time, fade\_amplitudes) plt.xlabel('Time (ms)') plt.ylabel('Fading Amplitude') plt.title('Rayleigh Fading Simulation') plt.grid(True) plt.show()

# Calculate fade time duration

print(f'Estimated Fade Time Duration: {fade\_time:.2f} ms')

# Output:



Estimated Fade Time Duration: 0.00 ms

**For activity 2:**

Code:

import numpy as np import matplotlib.pyplot as plt

# Simulation parameters

num\_samples = 1000 # Number of time samples fade\_duration = 100 # Duration of fading in milliseconds

# Define the carrier frequency (in Hz) carrier\_frequency = 900e6

# Generate velocities (e.g., in m/s) for your scenario

velocity\_range = np.linspace(0, 100, num\_samples) # Adjust the range as needed

# Initialize lists to store fade times for different velocities fade\_times = []

# Simulate fading for different velocities for velocity in velocity\_range:

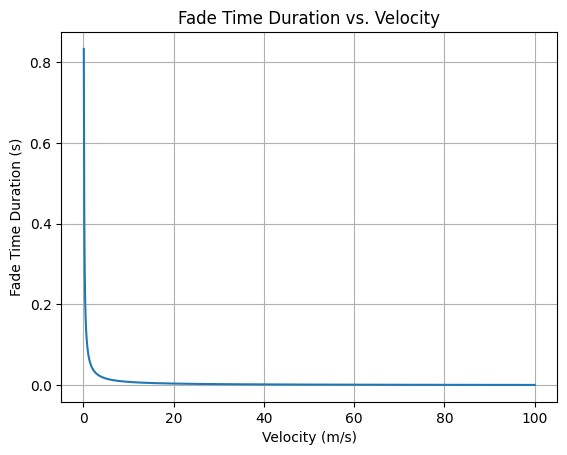
# Calculate the Doppler spread using the Doppler shift formula doppler\_spread = (velocity / 3e8) \* carrier\_frequency

# Use the Doppler spread to estimate the fade duration (example formula) fade\_time = 1 / (4 \* doppler\_spread) # Example formula, adjust as needed

fade\_times.append(fade\_time)

# Plot fade time duration against velocity plt.plot(velocity\_range, fade\_times) plt.xlabel('Velocity (m/s)') plt.ylabel('Fade Time Duration (s)') plt.title('Fade Time Duration vs. Velocity') plt.grid(True) plt.show()

# Output:



**Conclusion*:***

In this experiment, we were able to learn fading effect for cellular mobile communication.