QUESTION 4

Let the carrier frequency be 100 MHz and let the symbol period be 1 microsecond. Consider a transmitted BPSK signal that uses the 25% excess bandwidth **Root Raised Cosine pulses** (you can use the definition in Wikipedia, where beta=0.25). Using MATLAB or your favorite programming language, plot the RF modulated BPSK signal,

$$x(t) = \left[\sum_{n=1}^{8} x_n p_{rrc}(t - nT_s)\right] \cos(2\pi f_c t)$$

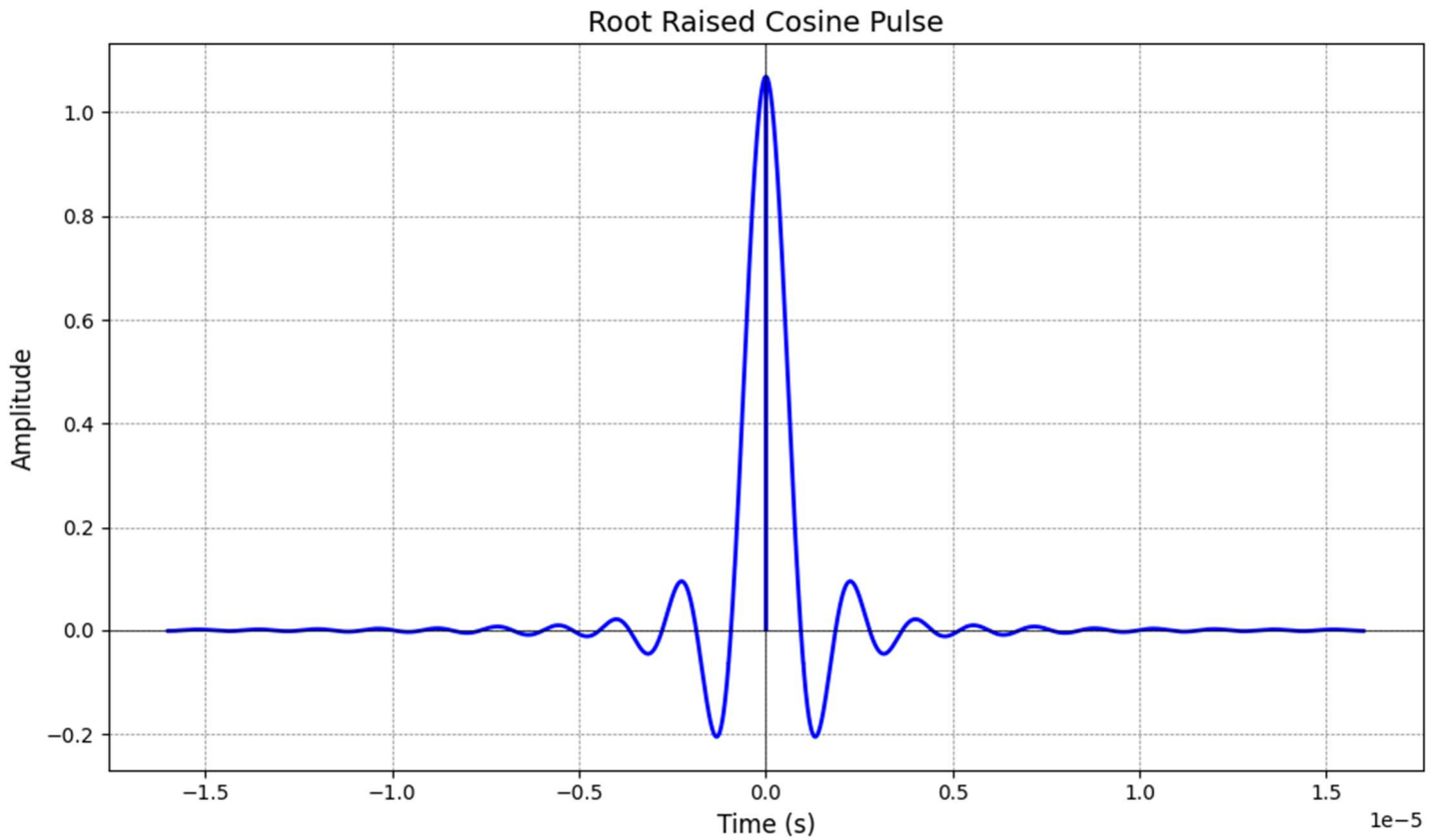
assuming the symbol

SOLUTION

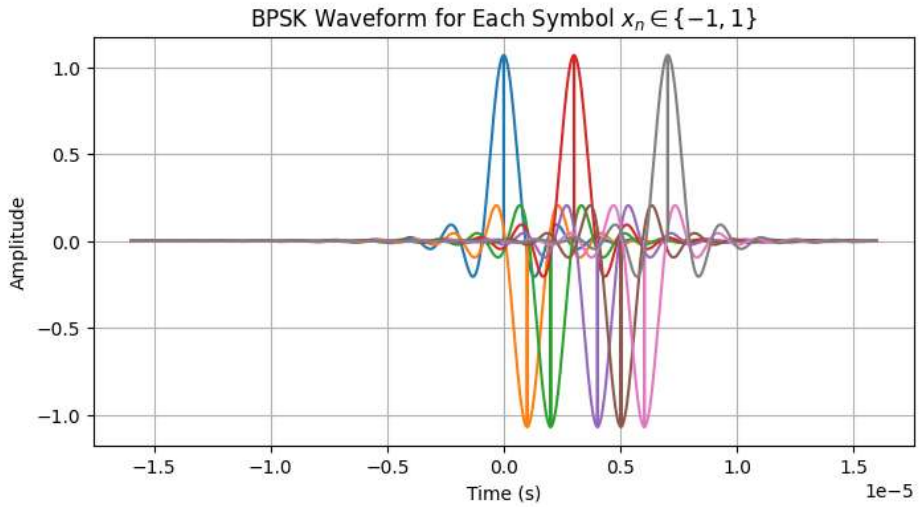
1.1 CODE:

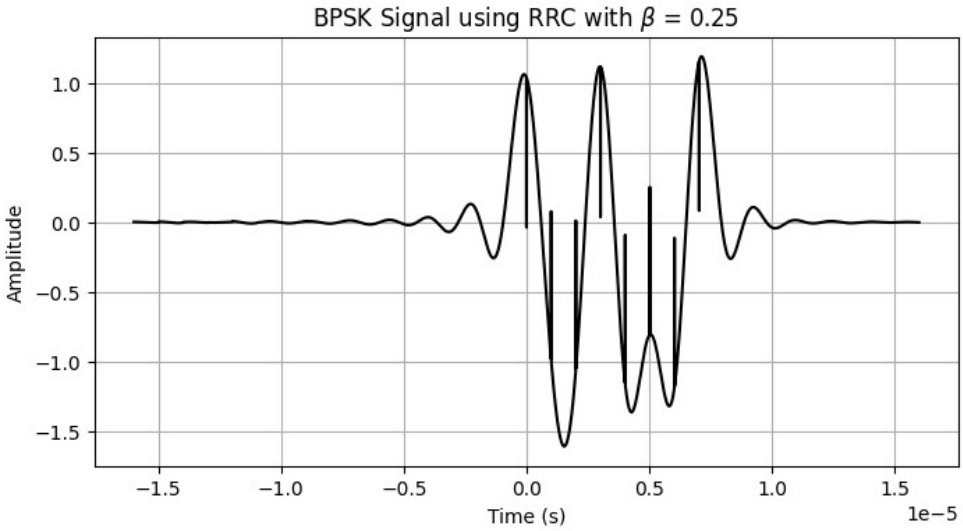
```
import numpy as np
import matplotlib.pyplot as plt

def root_raised_cosine(t, Ts, beta):
    """
    Generate a Root Raised Cosine (RRC) pulse shaping function.

    Parameters:
    - t (numpy.ndarray): Time vector.
    - Ts (float): Symbol period.
    - beta (float): Excess bandwidth parameter.

    Returns:
    numpy.ndarray: RRC pulse shape evaluated at each time point in t.
    """
    numerator = np.sin(np.pi * t / Ts * (1 - beta)) + 4 * beta * t / Ts *
np.cos(np.pi * t / Ts * (1 + beta))
    denominator = np.pi * t / Ts * (1 - (4 * beta * t / Ts) ** 2)
    return numerator / denominator

# Parameters
fc = 100e6  # Carrier Frequency
Ts = 1e-6   # Symbol Period
beta = 0.25 # Excess Bandwidth
N = 8       # Number of symbols

# Time vector
t = np.arange(-2 * N * Ts, 2 * N * Ts, 1 / (2 * fc))

# Zero crossing and shift for +-Ts/(4*beta)
```

```
zc = len(t) // 2
shift = len(np.arange(0, Ts / (4 * beta), 1 / (2 * fc))) + 1

# Root Raised Cosine pulse
p = root_raised_cosine(t, Ts, beta)
p[zc] = Ts * (1 + beta * (4 / np.pi - 1))
p[zc + shift] = (beta / np.sqrt(2)) * ((1 + 2 / np.pi) * np.sin(np.pi / (4 * beta)) +
(1 - 2 / np.pi) * np.cos(np.pi / (4 * beta)))
p[zc - shift] = p[zc + shift]

# Plot
plt.figure(figsize=(10, 6))
plt.plot(t, p, color='blue', linestyle='-', linewidth=2, label=r"Root Raised Cosine,
$\beta$ = 0.25")
plt.xlabel("Time (s)", fontsize=12)
plt.xlabel("Time (s)", fontsize=12)
plt.ylabel("Amplitude", fontsize=12)
plt.title("Root Raised Cosine Pulse", fontsize=14)
plt.axhline(0, color='black',linewidth=0.5)
plt.axvline(0, color='black',linewidth=0.5)
plt.grid(color = 'gray', linestyle = '--', linewidth = 0.5)
plt.tight_layout()
plt.show()
```

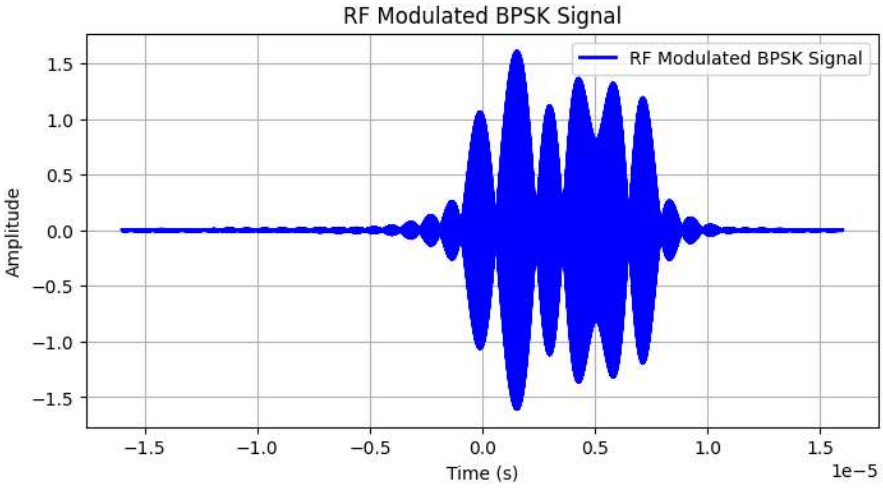


```
# Given BPSK sequence
x_n = np.array([1, -1, -1, 1, -1, -1, -1, 1])

# Initialize variables for shifted pulse, interval, and BPSK signal
shifted_p = np.zeros(len(t))
interval = len(np.arange(0, Ts, 1 / (2 * fc))) + 1
x_t = np.zeros(len(t))

# BPSK Waveform for Each Symbol
plt.figure(figsize=(8, 4))
plt.tight_layout()

# Iterate over each symbol in the BPSK sequence
for i in range(N):
    # Shift and update the pulse for each symbol
    shifted_p[interval * i : len(p)] = p[0 : len(p) - interval * i]

    # Accumulate the BPSK signal
    x_t += x_n[i] * shifted_p

    # Plot the BPSK waveform for each symbol
    plt.plot(t, x_n[i] * shifted_p, linewidth=1.5)

# Set plot labels, title, and grid
plt.xlabel("Time (s)")
plt.ylabel("Amplitude")
plt.title("BPSK Waveform for Each Symbol $x_n \in \{-1, 1\}$")
plt.grid()

#display the plot
plt.show()

# BPSK Signal using RRC with beta = 0.25
plt.figure(figsize=(8, 4))
plt.plot(t, x_t, "k-", linewidth=1.5)
plt.xlabel("Time (s)")
plt.ylabel("Amplitude")
plt.title("BPSK Signal using RRC with $\\beta$ = 0.25")
plt.grid()

# Save and display the plot
plt.show()
```






```
modulated_signal = x_t * np.cos(2 * np.pi * fc * t)
# Plot the RF Modulated BPSK Signal
plt.figure(figsize=(8, 4))
plt.plot(t, modulated_signal, color='blue', linestyle='-', linewidth=2, label='RF Modulated
BPSK Signal')
plt.xlabel("Time (s)")
plt.ylabel("Amplitude")
plt.title("RF Modulated BPSK Signal")
plt.grid(True)
plt.legend()
plt.show()
```



PROBLEM 5

Using the same parameters as in Problem 1, plot the RF modulated QPSK waveform

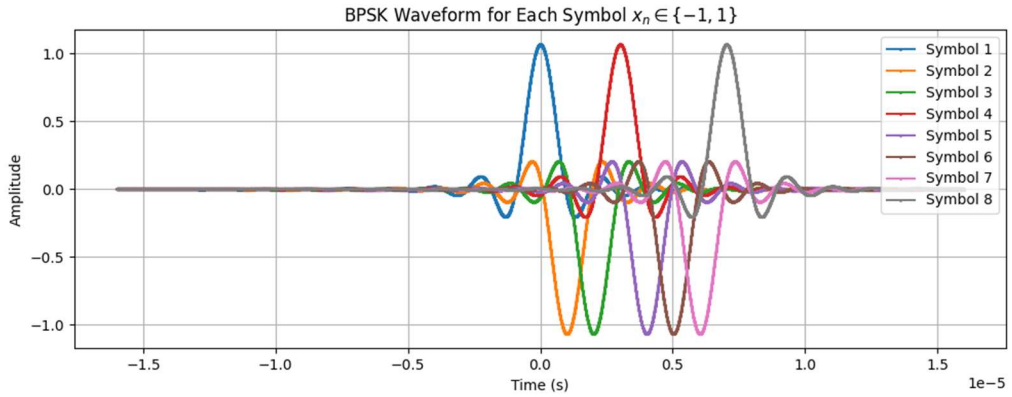
$$x(t) = [\sum_{n=1}^{8} x_n p_{rrc}(t - nT_s)] \cos(2\pi f_c t) - [\sum_{n=1}^{8} y_n p_{rrc}(t - nT_s)] \sin(2\pi f_c t)$$

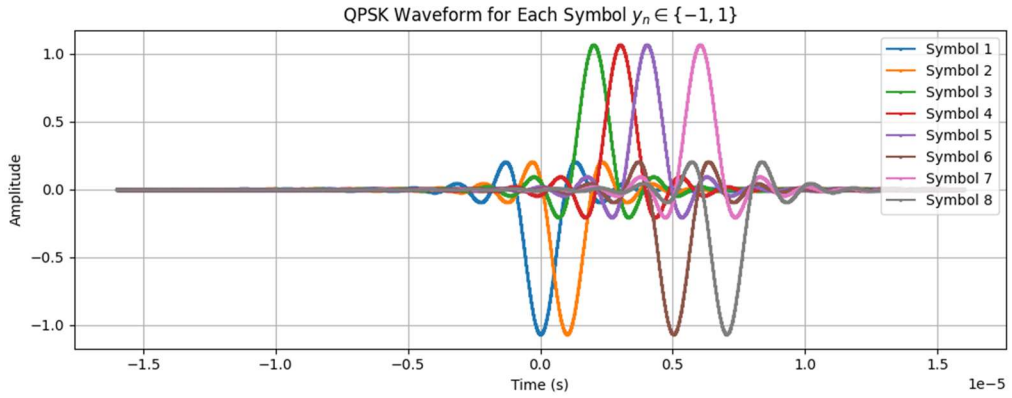
Use the same x_n sequence as in Problem 1, and let the quadrature symbols of the signal, y_n, be -1, -1, 1, 1, 1, -1, 1, -1.

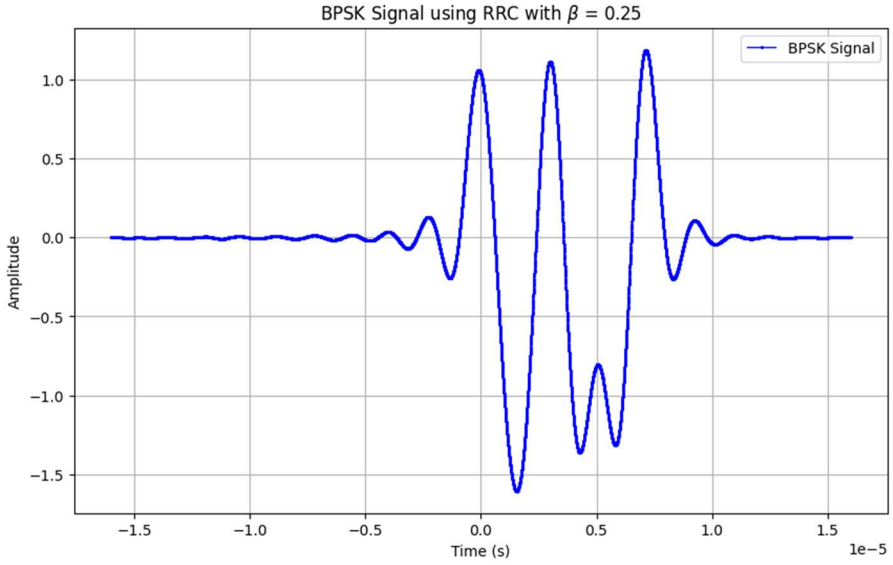
SOLUTION:

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

# QPSK symbols
y_n = np.array([-1, -1, 1, 1, 1, -1, 1, -1])

# Initialize arrays for BPSK and QPSK waveforms
shifted_p = np.zeros(len(t))
interval = len(np.arange(0, Ts, 1 / (2 * fc))) + 1
x_t = np.zeros(len(t))
y_t = np.zeros(len(t))

# Create subplots for BPSK and QPSK waveforms
fig, axes = plt.subplots(2, 1, figsize=(10, 8))

# Generate and plot BPSK and QPSK waveforms
for i in range(N):
    # Shift the pulse for each symbol
    shifted_p[interval * i : len(p)] = p[0 : len(p) - interval * i]

    # Accumulate BPSK and QPSK signals
```

```
    x_t += x_n[i] * shifted_p
    y_t += y_n[i] * shifted_p

    # Plot individual BPSK symbols with different colors and markers
    axes[0].plot(t, x_n[i] * shifted_p, linestyle='-', linewidth=1.5, marker='o',
markersize=1.2, label=f'Symbol {i + 1}')

    # Plot individual QPSK symbols with different colors and markers
    axes[1].plot(t, y_n[i] * shifted_p, linestyle='-', linewidth=1.5, marker='s',
markersize=1.2, label=f'Symbol {i + 1}')

# Set labels, titles, legends, and grids for BPSK subplot
axes[0].set_xlabel("Time (s)")
axes[0].set_ylabel("Amplitude")
axes[0].set_title(r"BPSK Waveform for Each Symbol $x_n \in \{-1, 1\}$")
axes[0].legend()
axes[0].grid()

# Set labels, titles, legends, and grids for QPSK subplot
axes[1].set_xlabel("Time (s)")
axes[1].set_ylabel("Amplitude")
axes[1].set_title(r"QPSK Waveform for Each Symbol $y_n \in \{-1, 1\}$")
axes[1].legend()
axes[1].grid()

# Adjust layout and save the combined plot as "p5a.png"
plt.tight_layout()

plt.show()

# Plot the overall BPSK signal with a different color and save it as "p5b.png"
plt.figure(figsize=(10, 6))
plt.plot(t, x_t, "b-", linewidth=1, label="BPSK Signal", marker='o', markersize=1.2)
plt.xlabel("Time (s)")
plt.ylabel("Amplitude")
plt.title("BPSK Signal using RRC with $\\beta$ = 0.25")
plt.legend()
plt.grid()
plt.show()
```








```
# Modulate BPSK and QPSK signals
modulated_i = x_t * np.cos(2 * np.pi * fc * t)  # In-phase component
modulated_q = y_t * np.sin(2 * np.pi * fc * t)  # Quadrature component
modulated_p5 = modulated_i - modulated_q  # Combine I and Q components

# Plot the RF-modulated QPSK signal
plt.figure(figsize=(10, 6))
plt.plot(t, modulated_p5, "b-", linewidth=2, label="RF Modulated QPSK Signal")
plt.xlabel("Time (s)")
```

```
plt.ylabel("Amplitude")
plt.title("RF Modulated QPSK Signal")
plt.legend()
plt.grid()
plt.show()
```