psk-modulation-demodulation

March 29, 2024

[1]:
```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import simps
```

[3]:
```python
x = np.random.randint(0, 2, 10)
x
```

[3]:
```
array([1, 1, 1, 1, 0, 1, 0, 1, 1, 0])
```

[4]:
```python
bp = 0.000001
```

[5]:
```python
bit = np.array([])
for n in range(len(x)):
    if x[n] == 1:
        se = np.ones(100)
    else:
        se = np.zeros(100)
    bit = np.concatenate((bit, se))
len(bit)
```

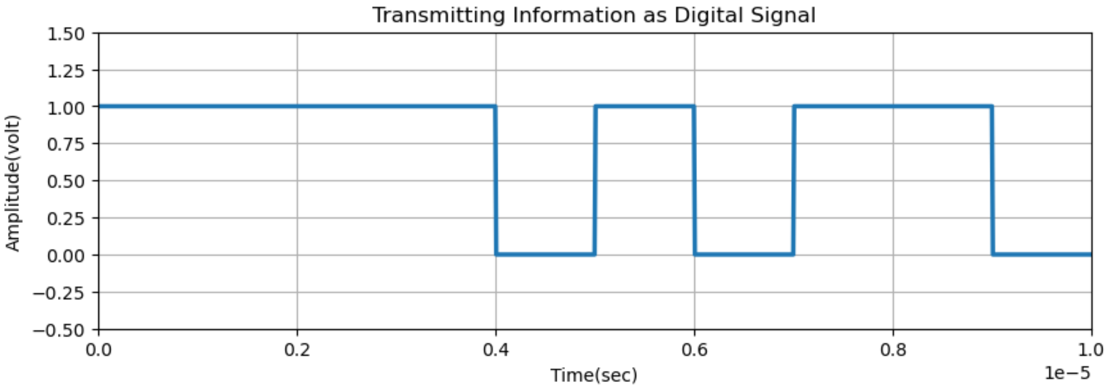
[5]:
```
1000
```

[12]:
```python
t1 = np.arange(bp/100, 100*len(x)*(bp/100) + bp/100, bp/100)
t1 = t1[:len(bit)]
len(t1)
```

[12]:
```
1000
```

[13]:
```python
plt.figure(figsize=(10,3))
plt.plot(t1, bit, linewidth=2.5)
plt.grid(True)
plt.axis([0, bp*len(x), -0.5, 1.5])
plt.ylabel('Amplitude(volt)')
plt.xlabel('Time(sec)')
plt.title('Transmitting Information as Digital Signal')
```

[13]:
```
Text(0.5, 1.0, 'Transmitting Information as Digital Signal')
```



Binary PSK Modulation

[31]:
```
A = 5
br = 1/bp
f = br*1
t2 = np.arange(bp/99, bp + bp/99, bp/99)
ss = len(t2)
ss
```

[31]:
```
99
```

[32]:
```
m = np.array([])
for i in range(len(x)):
    if x[i] == 1:
        y = A * np.cos(2*np.pi*f*t2)
    else:
        y = A * np.cos(2*np.pi*f*t2 + np.pi)
    m = np.concatenate((m, y))
len(m)
```

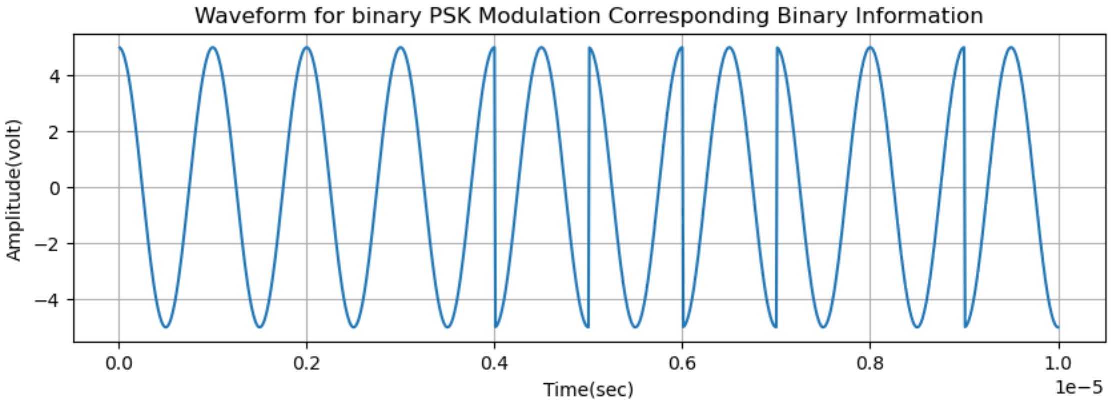
[32]:
```
990
```

[33]:
```
t3 = np.arange(bp/99, bp*len(x) + bp/99, bp/99)
len(t3)
```

[33]:
```
990
```

[34]:
```
plt.figure(figsize=(10, 3))
plt.plot(t3, m)
plt.grid(True)
plt.xlabel('Time(sec)')
plt.ylabel('Amplitude(volt)')
plt.title('Waveform for binary PSK Modulation Corresponding Binary Information')
```

[34]: Text(0.5, 1.0, 'Waveform for binary PSK Modulation Corresponding Binary Information')



Binary PSK Demodulation

[37]:
```
mn = np.array([])

for n in range(ss-1, len(m), ss):
    t = np.arange(bp/99, bp + bp/99, bp/99)
    y = np.cos(2*np.pi*f*t)
    mm = y * m[n-ss+1:n+1]
    z = simps(mm, t)
    zz = round((2*z/bp))
    if zz > 0:
        a = 1
    else:
        a = 0
    mn = np.append(mn, a)
len(mn)
```

[37]: 10

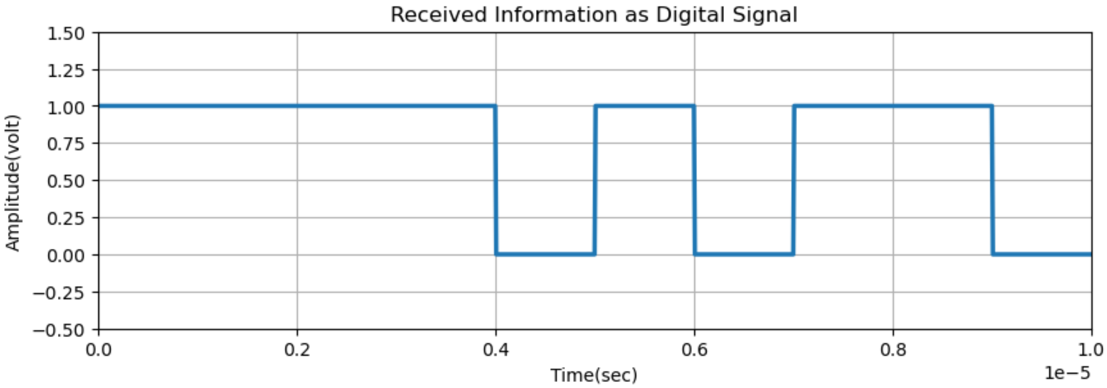
[38]:
```
bit = np.array([])
for n in range(len(mn)):
    if mn[n] == 1:
        se = np.ones(100)
    else:
        se = np.zeros(100)
    bit = np.concatenate((bit, se))
```

[40]:
```
t4 = np.arange(bp/100, 100*len(mn)*(bp/100) + bp/100, bp/100)
t4 = t4[:len(bit)]
len(t4)
```

```
[40]: 1000
```

```
[41]: plt.figure(figsize=(10,3))
      plt.plot(t4, bit, linewidth=2.5)
      plt.grid(True)
      plt.axis([0, bp*len(mn), -0.5, 1.5])
      plt.ylabel('Amplitude(volt)')
      plt.xlabel('Time(sec)')
      plt.title('Received Information as Digital Signal')
```

```
[41]: Text(0.5, 1.0, 'Received Information as Digital Signal')
```



```
[ ]:
```