Frequency Modulation.

4.3 The message signal *m(t)* is given by

$$m(t) = \begin{cases} t & 0 \le t < 1 \\ -t + 2 & 1 \le t < 2 \\ 0 & otherwise \end{cases}$$

Frequency modulates the carrier signal $c(t) = \cos(2\pi f_c t)$, where f_c = 1000Hz and the frequency deviation constant is k_f = 25.

1. Plot the message signal and its integral on two separate graphs.

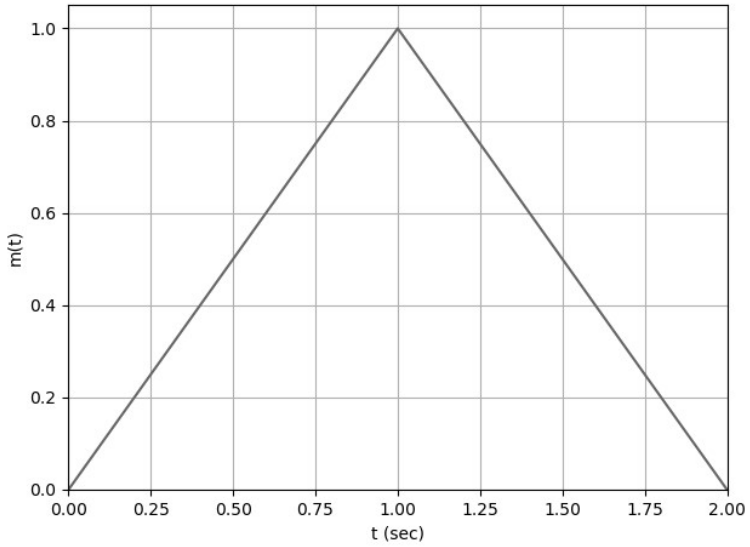

Fig 1: *m(t)*

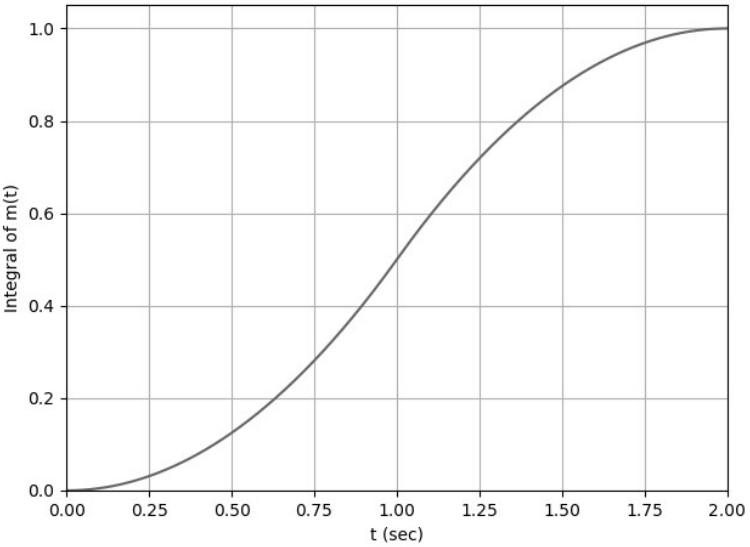

Fig 2: *Integral of m(t)*

CODE snippet:

```
# mathematical function of message signal def
message(t):
	if 0 <= t < 1: return
		t
	elif 1 <= t < 2:
		return -t + 2
	else:
		return 0


m = [message(T) for T in t]                          # message signal
m_inti = [ing.quad(message, 0, T)[0] for T in t]     # integration of
                                                       message signal
```

2. Plot the FM signal u(t)

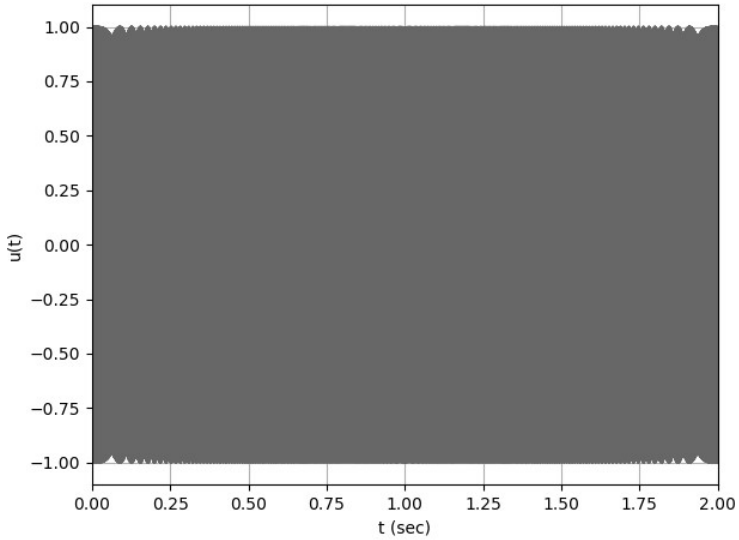

Fig 3: *u(t)*

CODE snippet:

```
fc = 1000           # carrier frequency
ts = 0.0001         # sampling time
kf = 25             # freq. deviation const.
 u = [cos(2*pi*fc*t[i] + 2*pi*kf*m_inti[i]) for i in
range(len(t))]
```

3. Compute and plot the spectra of m(t) and u(t)

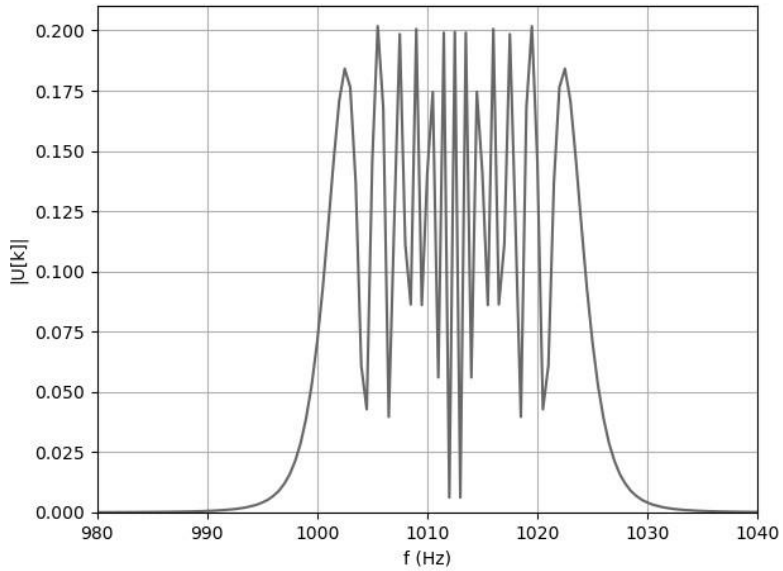

Fig 4: |U[k]|

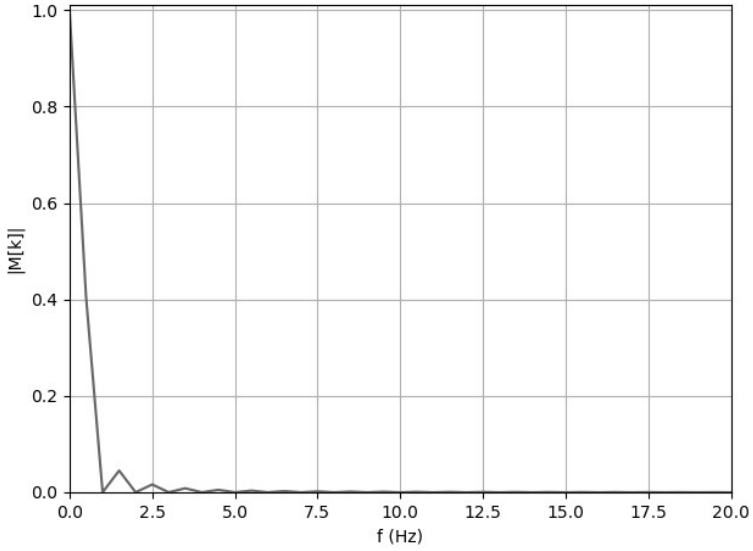

Fig 5: |M[k]|

CODE snippet:

```
t = np.arange(0, 2, ts)
N = int(len(t) / 2)
f = [i / (ts * len(t)) for i in np.arange(N)]

U = np.abs(np.fft.fft(u)[0:N]) / N
M = np.abs(np.fft.fft(m)[0:N]) / N
```

4. Find
   a. Modulation index : 31.4
   b. Bandwidth of u(t) : 45

c. Range of instantaneous frequency : 1000 to 1157.1 Hz

5. Demodulated the signal u(t) and plot the result.

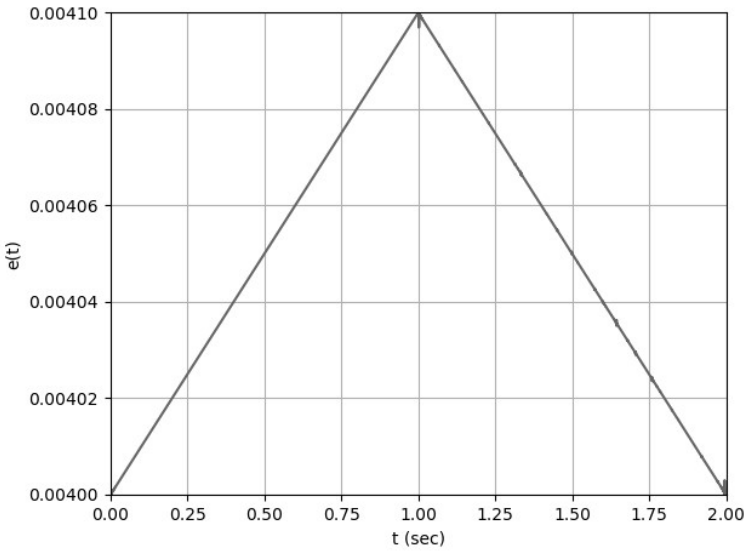

Fig 6: e(t) CODE

snippet:

```
# Demodulation
phase = np.unwrap(np.angle(sp.hilbert(u)))
e = np.diff(phase) / (2 * pi * kf)
```

The demodulated signal e(t) closely resembles message signal m(t) albeit greatly attenuated.

CODE:

```
# -*- coding: utf-8 -*- from
numpy import pi, cos import
numpy as np import
scipy.integrate as ing import
scipy.signal as sp import
matplotlib.pyplot as plt


# mathematical function of message signal def
message(t):
	if 0 <= t < 1:
		return t
	elif 1 <= t < 2:
		return -t + 2
	else:
		return 0


fc = 1000 		# carrier frequency
ts = 0.0001 		# sampling time
kf = 25 		# freq. deviation const.
```

```
 t = np.arange(0, 2,
ts) N = int(len(t) / 2)
f = [i / (ts * len(t)) for i in np.arange(N)]
m = [message(T) for T in t]                          # message signal
m_inti = [ing.quad(message, 0, T)[0] for T in t]     # integration of message
                                                     signal


# FM modulated signal{u(t)} and its FFT {U[k]}
u = [cos(2 * pi * fc * t[i] + 2 * pi * kf * m_inti[i]) for i in range(len(t))]
U = np.abs(np.fft.fft(u)[0:N]) / N
M = np.abs(np.fft.fft(m)[0:N]) / N


# Demodulation
phase = np.unwrap(np.angle(sp.hilbert(u)))
e = np.diff(phase) / (2 * pi * kf)
```