Pratool Gadtaula

Problem Set 6

1. The gunshot is an impulse because it produces a signal which is the sum of all frequencies at a maximum amplitude. The impulse response is the sound (series of air pressure changes) that the microphone actually picks up which is affected by the shape of the room. The filter that corresponds to the wave that corresponds to the sound picked up by the microphone is the impulse response. Multiplying the impulse response element-wise with the frequency-domain of another sound that one wants to put in that same “room” or more accurately stated as having the same impulse response.

2. $y(t) = \frac{1}{2}x(t-1) + \frac{1}{4}x(t-10)$

It is reasonable to call this an echo channel because in the time domain one would get softer, time-displaced signals. That is exactly what one hears when something echoes, a series of quieter versions of the some sound after certain time delays.

impulse response: $h(t) = \frac{1}{2}\delta(t-1) + \frac{1}{4}\delta(t-10)$, where

$$\delta(t) = \lim_{\Delta \to 0} b(t) \text{ with Figure 1.}$$

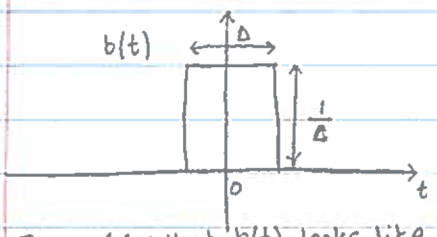


Figure 1: What b(t) looks like

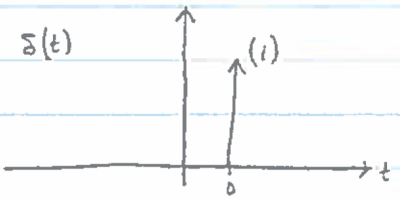


Figure 2: How $\delta(t)$ is represented

The impulse response of this echo channel looks like Figure 3 graphically.

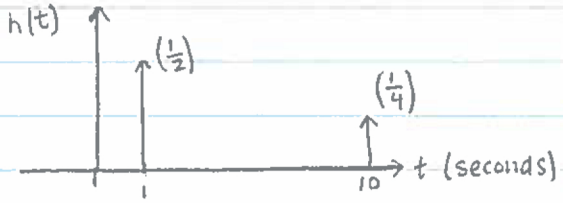
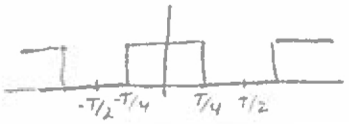


Figure 3: Impulse response graph



3.(a) $C_k = \frac{1}{T}\int_{-T/2}^{T/2} x(t)\, e^{-j\frac{2\pi}{T}kt}\, dt$ $\qquad x(t) = \begin{cases} 0, & \text{when } -T/2 \le t < -T/4,\ T/4 < t \le T/2 \\ 1, & \text{when } -T/4 \le t \le T/4 \end{cases}$

$$C_k = \frac{1}{T}\left[\int_{-T/2}^{-T/4}(0)\, e^{-j\frac{2\pi}{T}kt}\, dt + \int_{-T/4}^{T/4}(1)\, e^{-j\frac{2\pi}{T}kt}\, dt + \int_{T/4}^{T/2}(0)\, e^{-j\frac{2\pi}{T}kt}\, dt\right]$$

$$C_k = \frac{1}{T}\left[\int_{-T/4}^{T/4} e^{-j\frac{2\pi}{T}kt}\, dt\right]$$

$$C_k = \frac{1}{T}\cdot\frac{1}{-j\frac{2\pi}{T}k}\cdot\left[e^{-j\frac{2\pi}{T}kt}\right]_{-T/4}^{T/4}$$

$$C_k = \frac{1}{T}\cdot\frac{1}{j\frac{2\pi k}{T}}\cdot\left[e^{-j\frac{2\pi}{T}k\left(\frac{T}{4}\right)} - e^{-j\frac{2\pi}{T}k\left(-\frac{T}{4}\right)}\right]$$

$$C_k = \frac{1}{j2\pi k}\cdot\left[e^{-j\frac{\pi}{2}k} - e^{j\frac{\pi}{2}k}\right]$$

$$C_k = -\frac{1}{\pi k}\cdot\frac{1}{2j}\left(-e^{j\frac{\pi}{2}k} + e^{-j\frac{\pi}{2}k}\right)$$

$$C_k = \frac{1}{\pi k}\cdot -\frac{1}{2j}\left(e^{j\frac{\pi}{2}k} - e^{-j\frac{\pi}{2}k}\right)$$

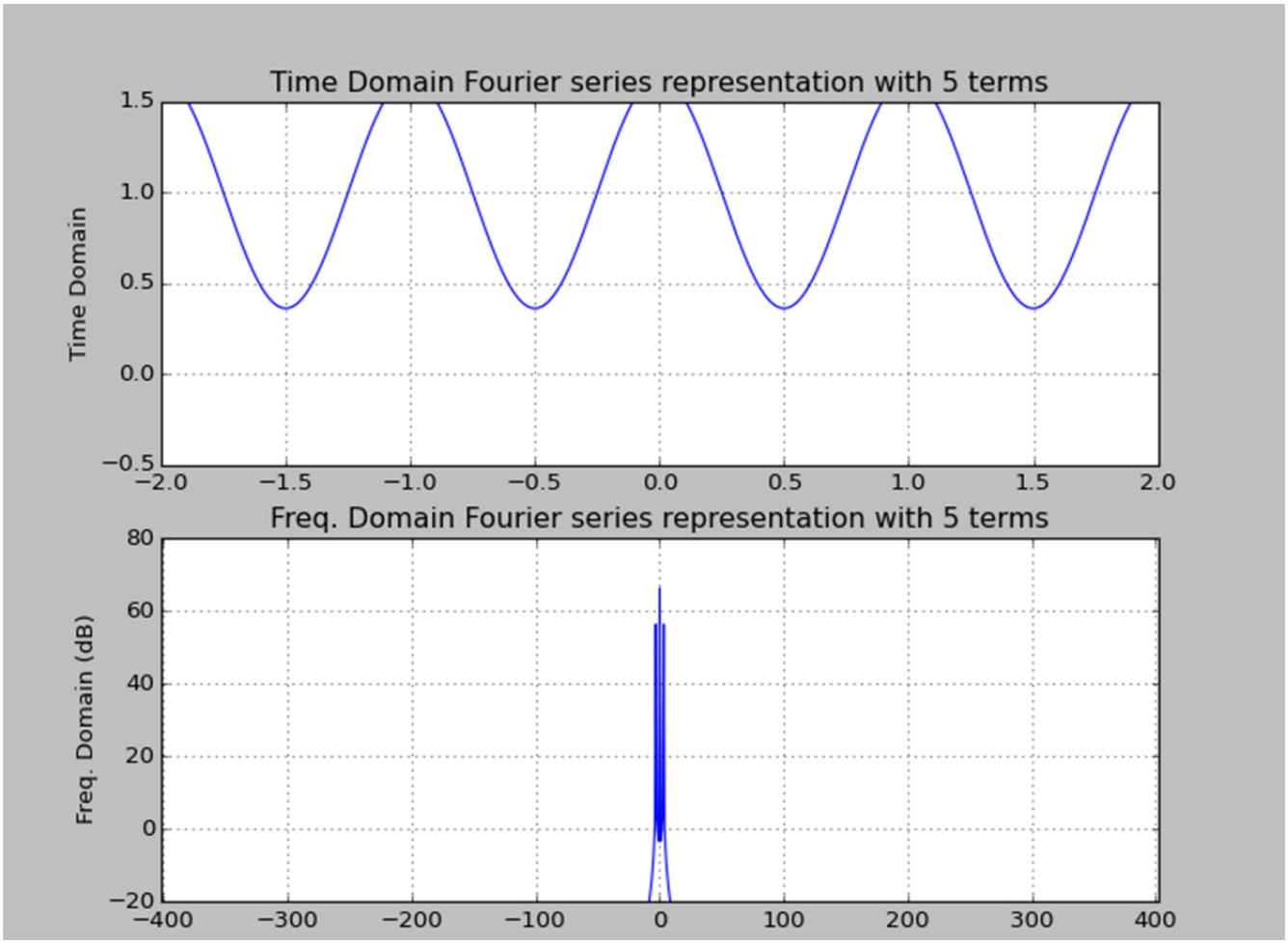
$$C_k = \frac{1}{\pi k}\cdot\sin\left(\frac{\pi}{2}k\right)$$

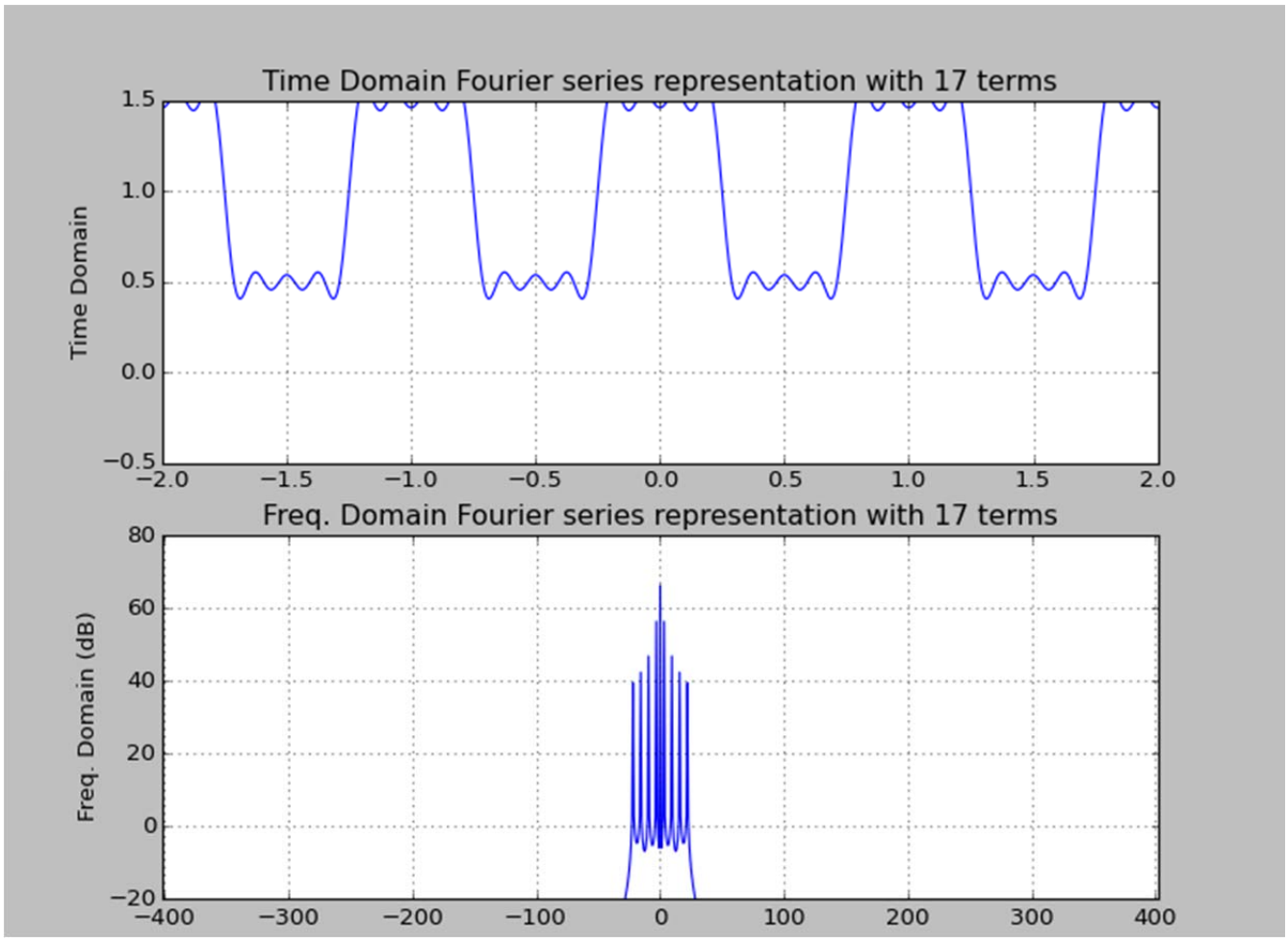
$$C_k = \frac{1}{\pi k}\sin\left(\frac{\pi}{2}k\right) \Rightarrow \quad \tilde{x}_k(t) = \sum_{k=-K}^{K} C_k e^{j\frac{2\pi}{T}kt}$$

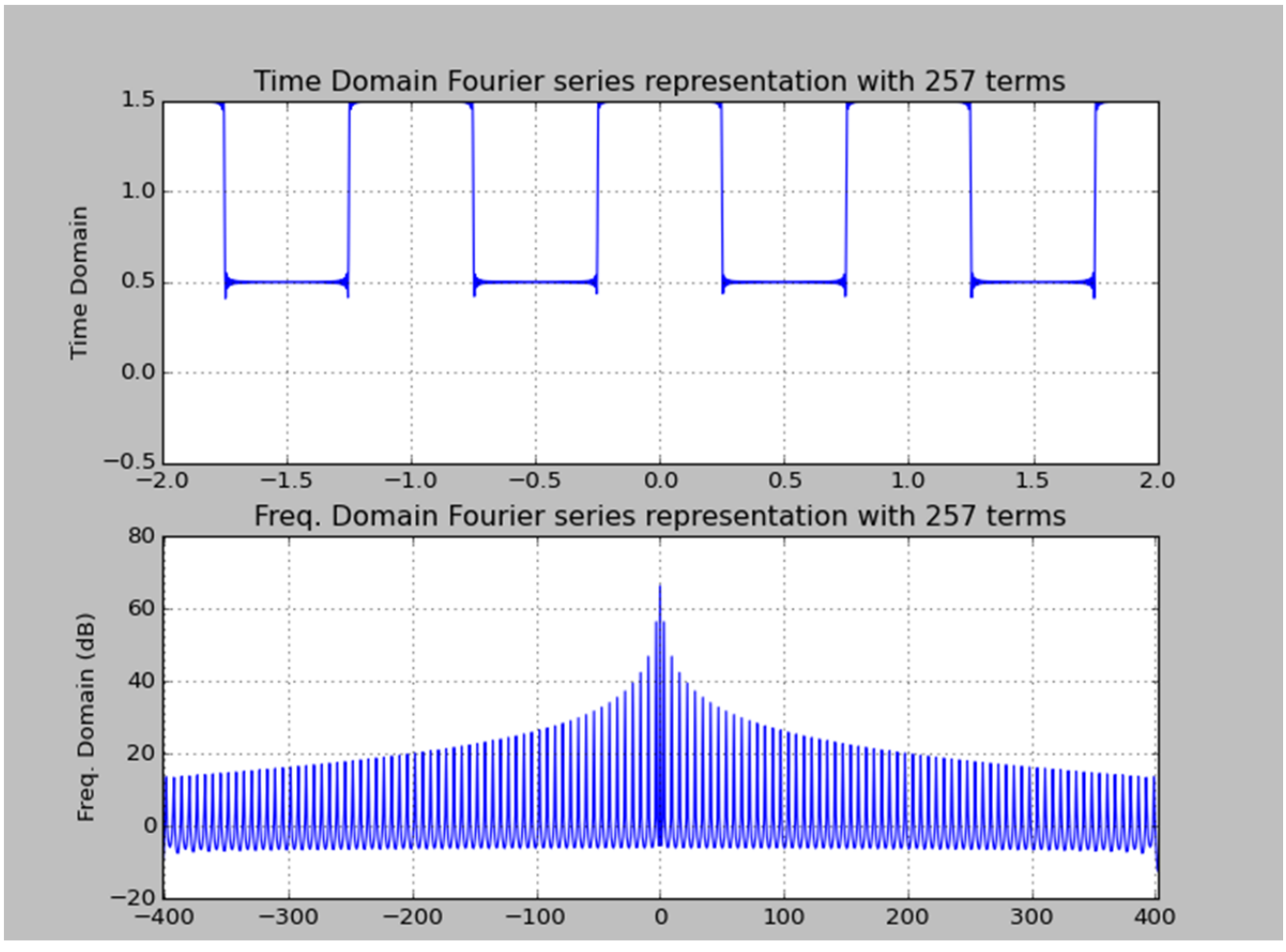
$$\tilde{x}_k(t) = \sum_{k=-K}^{K}\frac{1}{\pi k}\sin\left(\frac{\pi}{2}k\right)e^{j\frac{2\pi}{T}kt}$$

$$\tilde{x}_k(t) = \sum_{k=-K}^{K}\frac{1}{2}\cdot\frac{1}{\frac{\pi}{2}k}\sin\left(\frac{\pi}{2}k\right)e^{j\frac{2\pi}{T}kt}$$

3. (b)


Time Domain Fourier series representation with 5 terms
Time Domain
1.5
1.0
0.5
0.0
−0.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Freq. Domain Fourier series representation with 5 terms
Freq. Domain (dB)
80
60
40
20
0
−20
−400
−300
−200
−100
0
100
200
300
400


Time Domain Fourier series representation with 17 terms
Time Domain
1.5
1.0
0.5
0.0
−0.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Freq. Domain Fourier series representation with 17 terms
Freq. Domain (dB)
80
60
40
20
0
−20
−400
−300
−200
−100
0
100
200
300
400


Time Domain Fourier series representation with 257 terms
Time Domain
1.5
1.0
0.5
0.0
−0.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Freq. Domain Fourier series representation with 257 terms
Freq. Domain (dB)
80
60
40
20
0
−20
−400
−300
−200
−100
0
100
200
300
400

```
def fs_square(ts, M=3, T=4):
    # computes a fourier series representation of a square wave
    # with M terms in the Fourier series approximation
    # if M is odd, terms -(M-1)/2 -> (M-1)/2 are used
    # if M is even terms -M/2 -> M/2-1 are used

    # create an array to store the signal
    x = np.zeros(len(ts))

    # if M is even
    if np.mod(M,2) ==0:
        for k in range(-int(M/2), int(M/2)):

  ###############################################################################
            ##  change the following line to provide the Fourier series coefficients for
  the square wave
            ##  Coeff = ??
            if k == 0:
                Coeff = 1.0
            else:
                Coeff = (1.0 / (math.pi * k)) * math.sin(math.pi/2.0 * k)

            x = x + Coeff*np.exp(1j*2*np.pi/T*k*ts)

    # if M is odd
    if np.mod(M,2) == 1:
        for k in range(-int((M-1)/2), int((M-1)/2)+1):

  ###############################################################################
            ##  change the following line to provide the Fourier series coefficients for
  the square wave
            ##  Coeff = ??
            if k == 0:
                Coeff = 1.0
            else:
                Coeff = (1.0 / (math.pi * k)) * math.sin(math.pi/2.0 * k)
            x = x + Coeff*np.exp(1j*2*np.pi/T*k*ts)

    return x
```

3.(c) At the discontinuous points of the square wave, the Fourier series representation has a sharp spike which then smooths into a flat line and then increases into another spike at the next discontinuous point. To reconcile this difference the smoothing into a flat line

$$\int_{-T/2}^{T/2} |x(t) - \tilde{x}_k(t)|^2 dt \to 0 \quad \text{when } K \text{ is large} \qquad (10)$$

transitions into a sinusoidal with first decaying amplitude and then increasing amplitude. This allows for the integral between the the square wave and its Fourier series representation to approach 0.

4. (a) $C_k = \frac{1}{T}\int_{-T/2}^{T/2} x(t) e^{-j\frac{2\pi}{T}kt} dt$

$C_{k_y} = \frac{1}{T}\int_{-T/2}^{T/2} y(t) e^{-j\frac{2\pi}{T}kt} dt$

$= \frac{1}{T}\int_{-T/2}^{T/2} x(t-T_1) e^{-j\frac{2\pi}{T}kt} dt \qquad u = t - T_1$

$du = dt$

$= \frac{1}{T}\int_{-T/2-T_1}^{T/2-T_1} x(u) e^{-j\frac{2\pi}{T}k(u+T_1)} du$

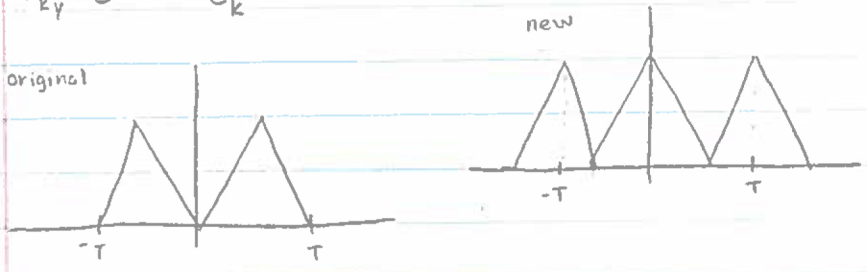
$= \frac{1}{T} \cdot e^{-j\frac{2\pi}{T}kT_1} \int_{-T/2-T_1}^{T/2-T_1} x(u) e^{-j\frac{2\pi}{T}ku} du$

$C_{k_y} = e^{-j\frac{2\pi}{T}kT_1} \cdot C_k$

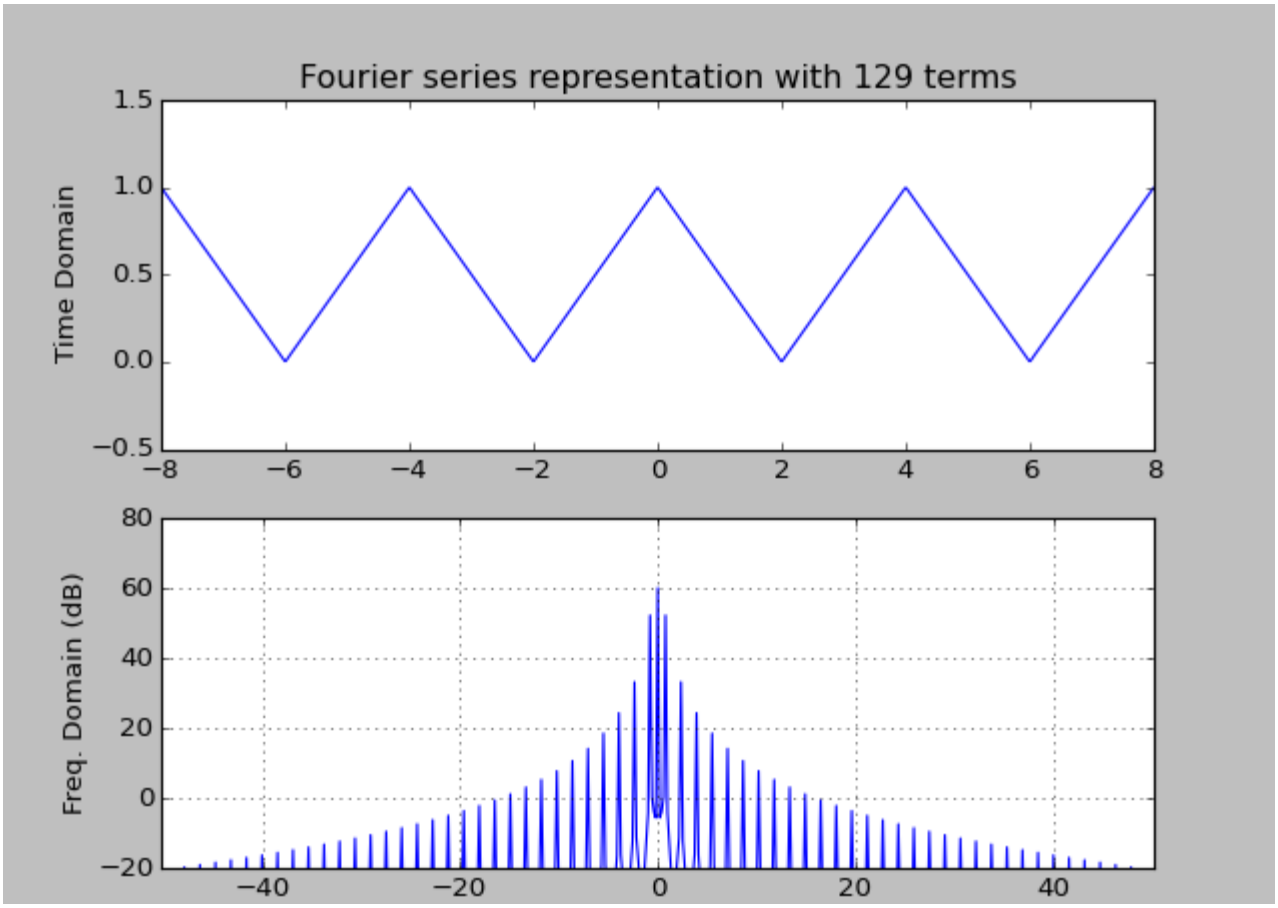
(b) $y(t) = x\left(t - \frac{T}{2}\right) = \frac{2}{T}\left|t - \frac{T}{2}\right|$

$C_{k_y} = e^{-j\frac{2\pi}{T}k\left(\frac{T}{2}\right)} \cdot C_k$ where $C_k = \begin{cases} -\frac{2}{\pi^2 k^2} & \text{if } k \text{ is odd} \\ \frac{1}{2} & \text{if } k = 0 \\ 0 & \text{otherwise} \end{cases}$

$C_{k_y} = e^{-j\pi k} \cdot C_k$



4. (b)


Fourier series representation with 129 terms
Time Domain
1.5
1.0
0.5
0.0
−0.5
−8
−6
−4
−2
0
2
4
6
8
Freq. Domain (dB)
80
60
40
20
0
−20
−40
−20
0
20
40

```
def fs_triangle(ts, M=3, T=4):
    # computes a fourier series representation of a triangle wave
    # with M terms in the Fourier series approximation
    # if M is odd, terms -(M-1)/2 -> (M-1)/2 are used
    # if M is even terms -M/2 -> M/2-1 are used

    # create an array to store the signal
    x = np.zeros(len(ts))

    # if M is even
    if np.mod(M,2) ==0:
        for k in range(-int(M/2), int(M/2)):
            # if n is odd compute the coefficients
            if np.mod(k, 2)==1:
                Coeff = -2/((np.pi)**2*(k**2))
            if np.mod(k,2)==0:
                Coeff = 0
            if n == 0:
                Coeff = 0.5
            Coeff = Coeff * np.exp(-1j*math.pi*k) # Phase shift
            x = x + Coeff*np.exp(1j*2*np.pi/T*k*ts)

    # if M is odd
    if np.mod(M,2) == 1:
        for k in range(-int((M-1)/2), int((M-1)/2)+1):
           # if n is odd compute the coefficients
            if np.mod(k, 2)==1:
                Coeff = -2/((np.pi)**2*(k**2))
            if np.mod(k,2)==0:
                Coeff = 0
            if k == 0:
                Coeff = 0.5
            Coeff = Coeff * np.exp(-1j*math.pi*k) # Phase shift
            x = x + Coeff*np.exp(1j*2*np.pi/T*k*ts)

    return x
```