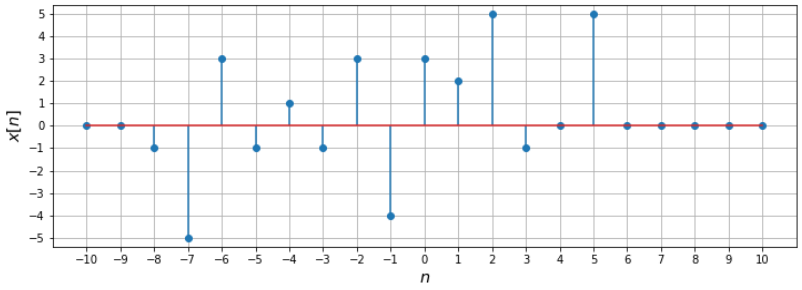
Signals & Systems
Homework #2 Solutions

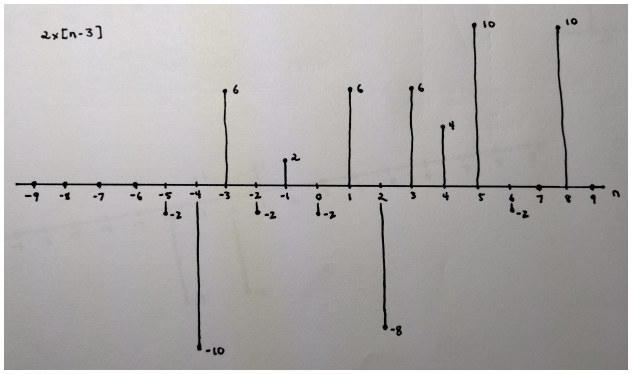
ECE 315 – Fall 2022
190 points total

Due Monday, October 24, 2022

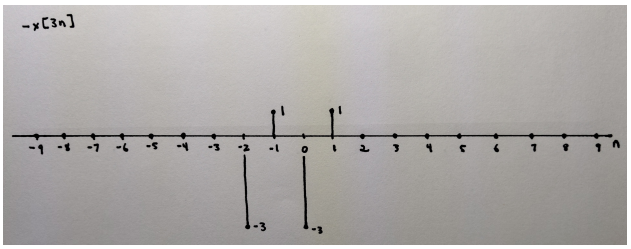
1. For the signal $x[n]$, plot the following signals over a large enough time interval to show all nonzero samples by hand. (Note that $x[n] = 0$ for $n \leq -9$ and for $n \geq 6$.) (34 points total)



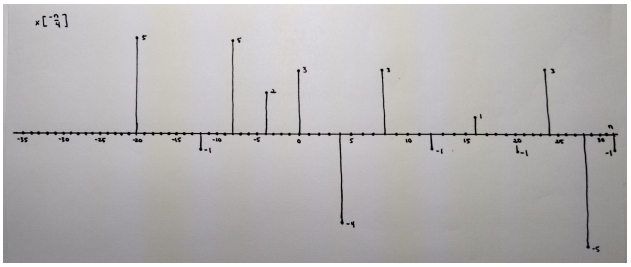
(a) $2x[n-3]$ (6 points)



(b) $-x[3n]$ (6 points)



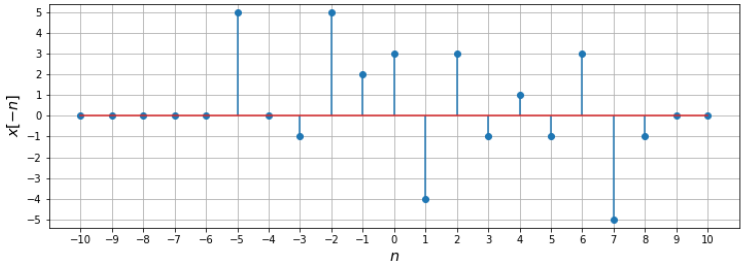
(c) $x[-\frac{n}{4}]$ (6 points)



(d) The even and odd parts of $x[n]$ (12 points)

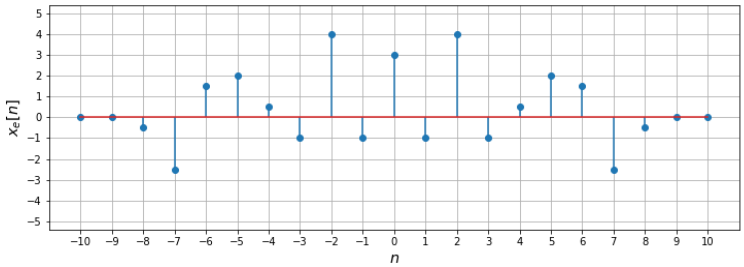
Here

$$x[n] = \begin{cases} -1, & n = -8 \\ -5, & n = -7 \\ 3, & n = -6 \\ -1, & n = -5 \\ 1, & n = -4 \\ -1, & n = -3 \\ 3, & n = -2 \\ -4, & n = -1 \\ 3, & n = 0 \\ 2, & n = 1 \\ 5, & n = 2 \\ -1, & n = 3 \\ 0, & n = 4 \\ 5, & n = 5 \\ 0, & \text{otherwise} \end{cases} \quad \text{and} \quad x[-n] = \begin{cases} 5, & n = -5 \\ 0, & n = -4 \\ -1, & n = -3 \\ 5, & n = -2 \\ 2, & n = -1 \\ 3, & n = 0 \\ -4, & n = 1 \\ 3, & n = 2 \\ -1, & n = 3 \\ 1, & n = 4 \\ -1, & n = 5 \\ 3, & n = 6 \\ -5, & n = 7 \\ -1, & n = 8 \\ 0, & \text{otherwise} \end{cases}$$



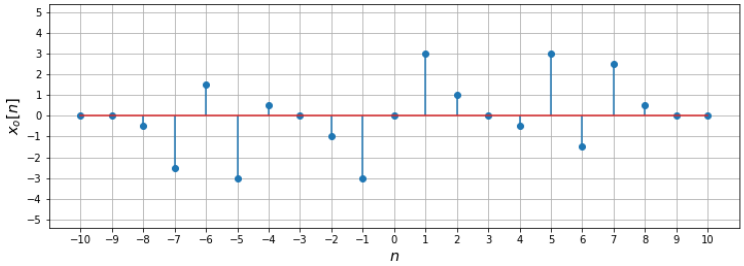
So, the even part of $x[n]$ is

$$x_e[n] = \frac{1}{2}(x[n]+x[-n]) = \frac{1}{2}\begin{cases} -1, & n=-8 \\ -5, & n=-7 \\ 3, & n=-6 \\ 4, & n=-5 \\ 1, & n=-4 \\ -2, & n=-3 \\ 8, & n=-2 \\ -2, & n=-1 \\ 6, & n=0 \\ -2, & n=1 \\ 8, & n=2 \\ -2, & n=3 \\ 1, & n=4 \\ 4, & n=5 \\ 3, & n=6 \\ -5, & n=7 \\ -1, & n=8 \\ 0, & \text{otherwise} \end{cases} = \begin{cases} -\frac{1}{2}, & n=-8 \\ -\frac{5}{2}, & n=-7 \\ \frac{3}{2}, & n=-6 \\ 2, & n=-5 \\ \frac{1}{2}, & n=-4 \\ -1, & n=-3 \\ 4, & n=-2 \\ -1, & n=-1 \\ 3, & n=0 \\ -1, & n=1 \\ 4, & n=2 \\ -1, & n=3 \\ \frac{1}{2}, & n=4 \\ 2, & n=5 \\ \frac{3}{2}, & n=6 \\ -\frac{5}{2}, & n=7 \\ -\frac{1}{2}, & n=8 \\ 0, & \text{otherwise} \end{cases}$$



and the odd part of $x[n]$ is

$$x_o[n] = \frac{1}{2}(x[n]-x[-n]) = \frac{1}{2}\begin{cases} -1, & n=-8 \\ -5, & n=-7 \\ 3, & n=-6 \\ -6, & n=-5 \\ 1, & n=-4 \\ 0, & n=-3 \\ -2, & n=-2 \\ -6, & n=-1 \\ 0, & n=0 \\ 6, & n=1 \\ 2, & n=2 \\ 0, & n=3 \\ -1, & n=4 \\ 6, & n=5 \\ -3, & n=6 \\ 5, & n=7 \\ 1, & n=8 \\ 0, & \text{otherwise} \end{cases} = \begin{cases} -\frac{1}{2}, & n=-8 \\ -\frac{5}{2}, & n=-7 \\ \frac{3}{2}, & n=-6 \\ -3, & n=-5 \\ \frac{1}{2}, & n=-4 \\ 0, & n=-3 \\ -1, & n=-2 \\ -3, & n=-1 \\ 0, & n=0 \\ 3, & n=1 \\ 1, & n=2 \\ 0, & n=3 \\ -\frac{1}{2}, & n=4 \\ 3, & n=5 \\ -\frac{3}{2}, & n=6 \\ \frac{5}{2}, & n=7 \\ \frac{1}{2}, & n=8 \\ 0, & \text{otherwise} \end{cases}$$



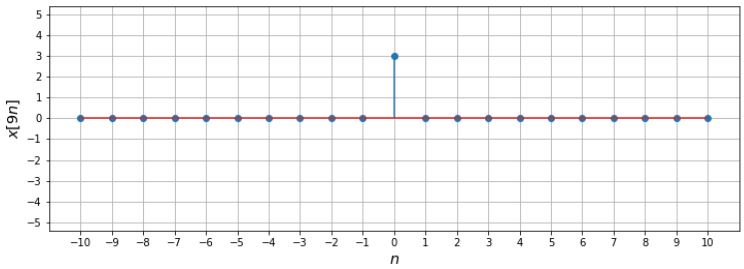
Note: I did not ask you to plot $x[-n]$ and only included the graph for your information. Also, it is completely fine if you only expressed $x_e[n]$ and $x_o[n]$ graphically and didn't write them piecewise.

(e) What is the smallest integer $k > 0$ for which $x[kn]$ contains only one nonzero sample from the original signal $x[n]$? (4 points)

Since $x[-8]$ is the nonzero sample of $x[n]$ that is furthest from the origin, $x[kn]$ contains only one nonzero sample from the original

signal, specifically $x[0]$, for $k \geq 9$. Therefore, the smallest value of k for which $x[kn]$ contains only one nonzero sample is $k = 9$.

More specifically, $x[kn] = x[0]\delta[n] = 3\delta[n]$ for $k \geq 9$, as shown below, where $x[9n] = 0$ for all values of n not shown.



As before, I included the graph for your information. It is not required in your solutions.

2. (32 points total)

(a) Determine the forward differences $y_1[n]$ and $y_2[n]$, respectively, of the signals

$$x_1[n] = \begin{cases} 5 - \frac{n}{2}, & 0 \leq n \leq 20 \\ 0, & \text{otherwise} \end{cases}$$

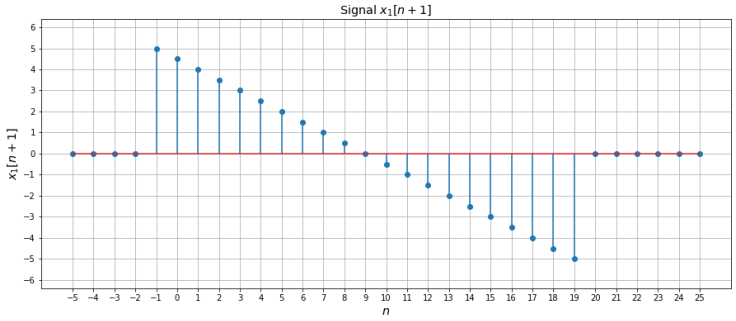
and

$$x_2[n] = \begin{cases} 5 - \frac{n}{2} - 1, & 0 \leq n \leq 20 \\ -1, & \text{otherwise} \end{cases}$$

by hand. How do $y_1[n]$ and $y_2[n]$ compare to each other? Use MATLAB or python to plot the signals and their forward differences. (Note: MATLAB and matplotlib have functions called "stem" for making stem plots.) (16 points)

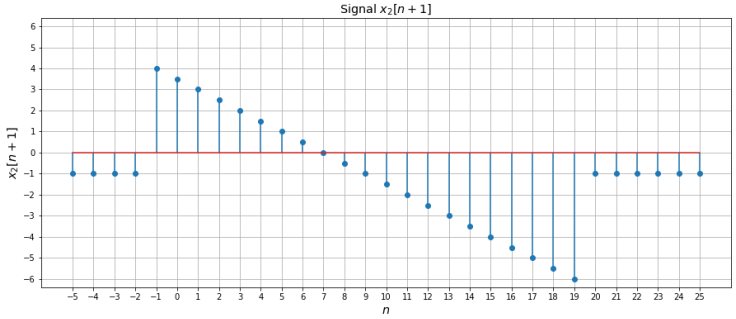
The signal $x_1[n]$ shifted one time step to the left is

$$x_1[n+1] = \begin{cases} 5 - \frac{n+1}{2}, & 0 \leq n+1 \leq 20 \\ 0, & \text{otherwise} \end{cases} = \begin{cases} 5 - \frac{n+1}{2}, & -1 \leq n \leq 19 \\ 0, & \text{otherwise} \end{cases}$$



and the signal $x_2[n]$ shifted one time step to the left is

$$x_2[n+1] = \begin{cases} 5 - \frac{n+1}{2} - 1, & 0 \leq n+1 \leq 20 \\ -1, & \text{otherwise} \end{cases} = \begin{cases} 5 - \frac{n+1}{2} - 1, & -1 \leq n \leq 19 \\ -1, & \text{otherwise} \end{cases}$$



Therefore, the forward difference of $x_1[n]$ is

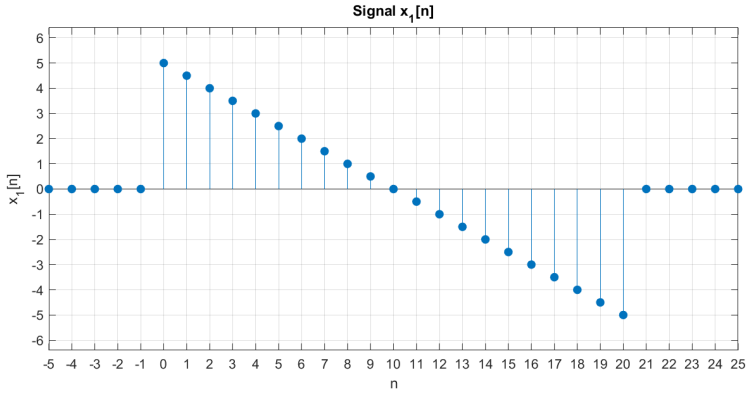
$$\begin{aligned} y_1[n] &= x_1[n+1] - x_1[n] \\ &= \begin{cases} 5 - \frac{n+1}{2}, & -1 \leq n \leq 19 \\ 0, & \text{otherwise} \end{cases} - \begin{cases} 5 - \frac{n}{2}, & 0 \leq n \leq 20 \\ 0, & \text{otherwise} \end{cases} \\ &= \begin{cases} 5, & n = -1 \\ 5 - \frac{n+1}{2} - \left(5 - \frac{n}{2}\right), & 0 \leq n \leq 19 \\ 5, & n = 20 \\ 0, & \text{otherwise} \end{cases} \\ &= \begin{cases} 5, & n = -1 \\ -\frac{1}{2}, & 0 \leq n \leq 19 \\ 5, & n = 20 \\ 0, & \text{otherwise} \end{cases} \end{aligned}$$

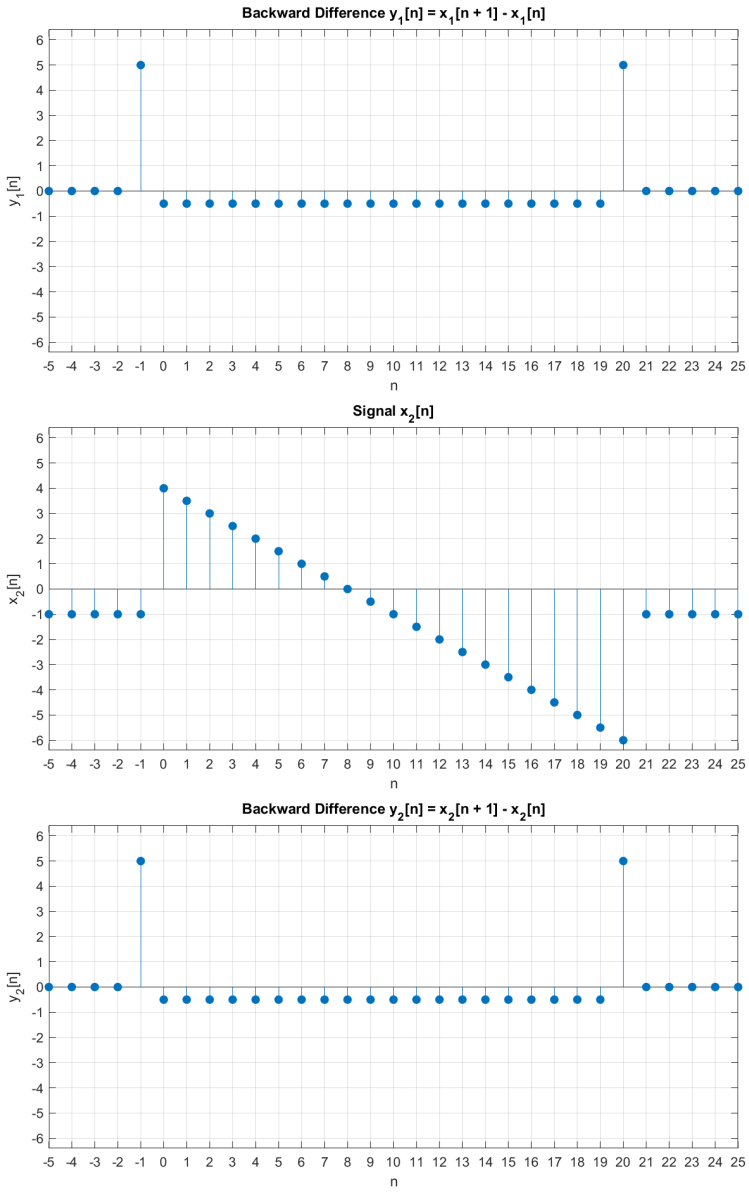
and the forward difference of $x_2[n]$ is

$$
\begin{aligned}
y_2[n] &= x_2[n+1] - x_2[n] \\
&= \begin{cases} 5 - \frac{n+1}{2} - 1, & -1 \le n \le 19 \\ -1, & \text{otherwise} \end{cases} - \begin{cases} 5 - \frac{n}{2} - 1, & 0 \le n \le 20 \\ -1, & \text{otherwise} \end{cases} \\
&= \begin{cases} 4 - (-1), & n = -1 \\ 5 - \frac{n+1}{2} - 1 - \left(5 - \frac{n}{2} - 1\right), & 0 \le n \le 19 \\ -1 - (5 - 10 - 1), & n = 20 \\ -1 - (-1), & \text{otherwise} \end{cases} \\
&= \begin{cases} 5, & n = -1 \\ -\frac{1}{2}, & 0 \le n \le 19 \\ 5, & n = 20 \\ 0, & \text{otherwise} \end{cases}
\end{aligned}
$$

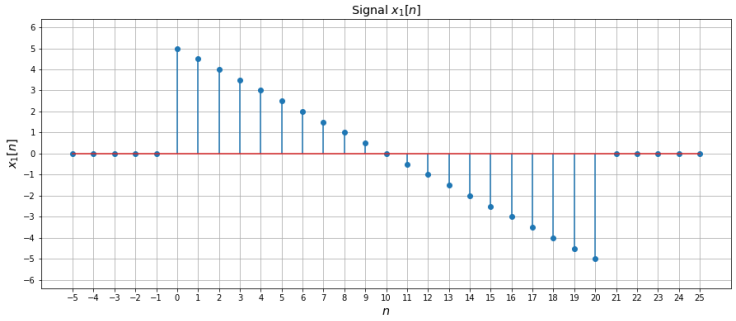
Notice that $y_2[n] = y_1[n]$, i.e. the forward differences of $x_1[n]$ and $x_2[n]$ are the same.

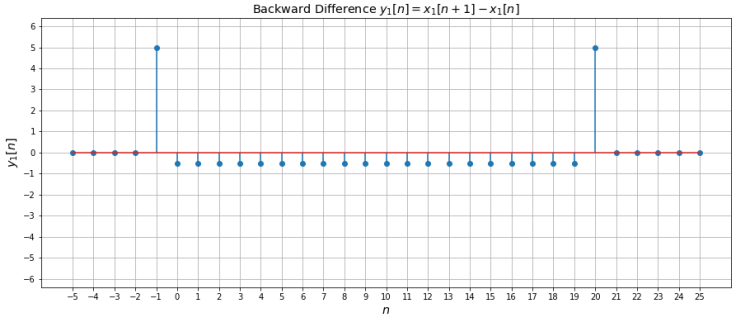
MATLAB code for plotting the signals $x_1[n]$ and $x_2[n]$ and their backward differences $y_1[n]$ and $y_2[n]$, respectively, is in Appendix A. This code produces the plots

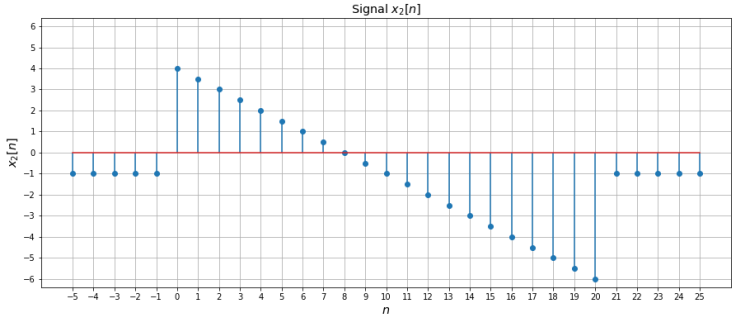


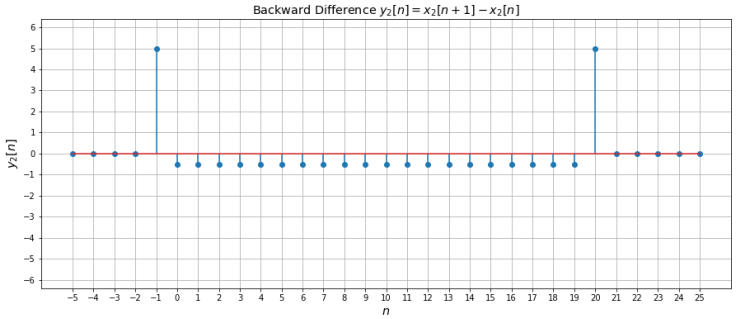


Python code for plotting the signals $x_1[n]$ and $x_2[n]$ and their backward differences $y_1[n]$ and $y_2[n]$, respectively, is in Appendix B. This code produces the plots


Signal $x_1[n]$
$x_1[n]$
n


Backward Difference $y_1[n] = x_1[n+1] - x_1[n]$
$y_1[n]$
n


Signal $x_2[n]$
$x_2[n]$
n


Backward Difference $y_2[n] = x_2[n+1] - x_2[n]$
$y_2[n]$
n

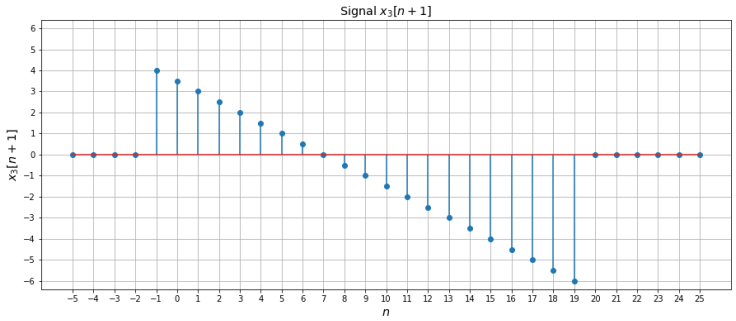
(b) Determine the forward difference $y_3[n]$ of the signal

$$x_3[n] = \begin{cases} 5 - \frac{n}{2} - 1, & 0 \le n \le 20 \\ 0, & \text{otherwise.} \end{cases}$$

by hand. Use MATLAB or python to plot $x_3[n]$ and $y_3[n]$. How does $y_3[n]$ differ from $y_1[n]$? (8 points)

The signal $x_3[n]$ shifted one time step to the left is

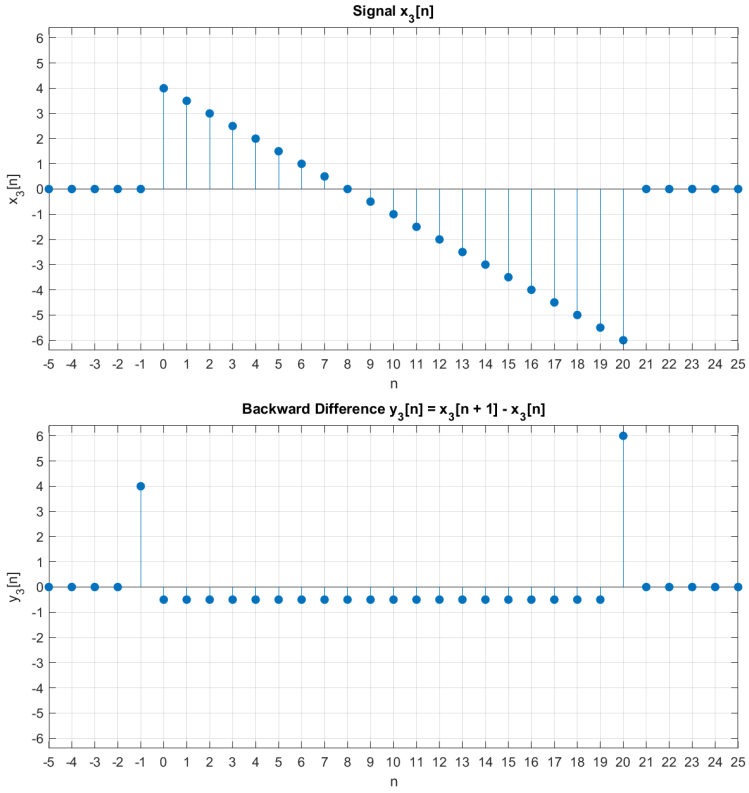
$$x_3[n+1] = \begin{cases} 5 - \frac{n+1}{2} - 1, & 0 \le n+1 \le 20 \\ 0, & \text{otherwise.} \end{cases} = \begin{cases} 5 - \frac{n+1}{2} - 1, & -1 \le n \le 19 \\ 0, & \text{otherwise.} \end{cases}$$



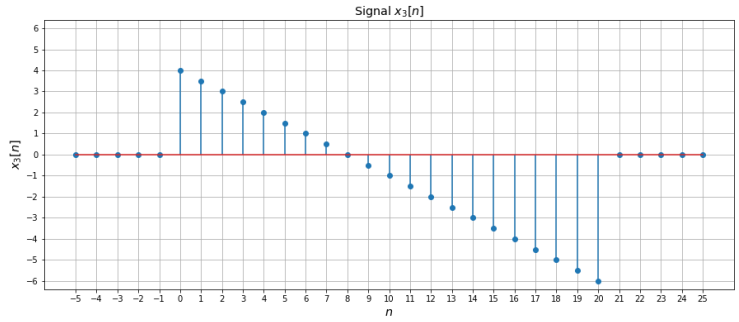
Therefore, the forward difference of $x_3[n]$ is

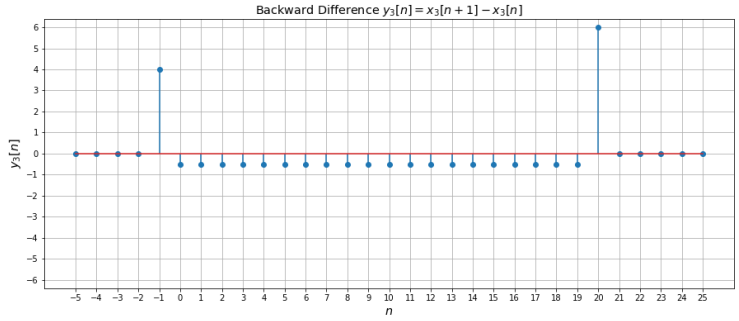
$$\begin{aligned} y_3[n] &= x_3[n+1] - x_3[n] \\ &= \begin{cases} 5 - \frac{n+1}{2} - 1, & -1 \le n \le 19 \\ 0, & \text{otherwise.} \end{cases} - \begin{cases} 5 - \frac{n}{2} - 1, & 0 \le n \le 20 \\ 0, & \text{otherwise.} \end{cases} \\ &= \begin{cases} 4, & n = -1 \\ 5 - \frac{n+1}{2} - 1 - \left(5 - \frac{n}{2} - 1\right), & 0 \le n \le 19 \\ -(5 - 10 - 1), & n = 20 \\ 0, & \text{otherwise} \end{cases} \\ &= \begin{cases} 4, & n = -1 \\ -\frac{1}{2}, & 0 \le n \le 19 \\ 6, & n = 20 \\ 0, & \text{otherwise} \end{cases} \end{aligned}$$

MATLAB code for plotting $x_3[n]$ and its backward difference $y_3[n]$ is in Appendix C. This code produces the plots



Python code for plotting $x_3[n]$ and its backward difference $y_3[n]$ is in Appendix D. This code produces the plots





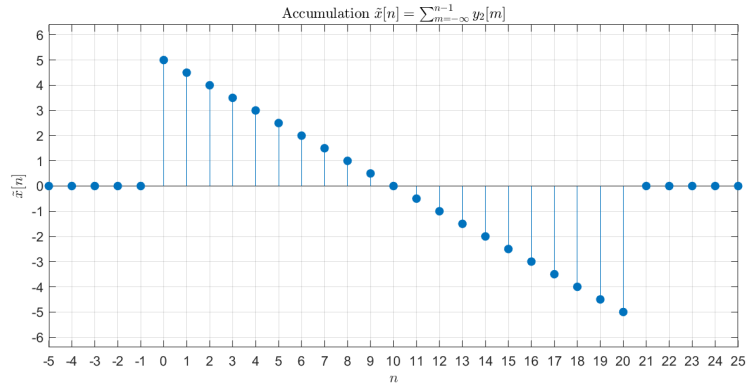
Notice that $y_3[-1] \neq y_1[-1]$ and $y_3[20] \neq y_1[20]$. Otherwise, the backward differences are the same.

(c) Calculate the accumulation

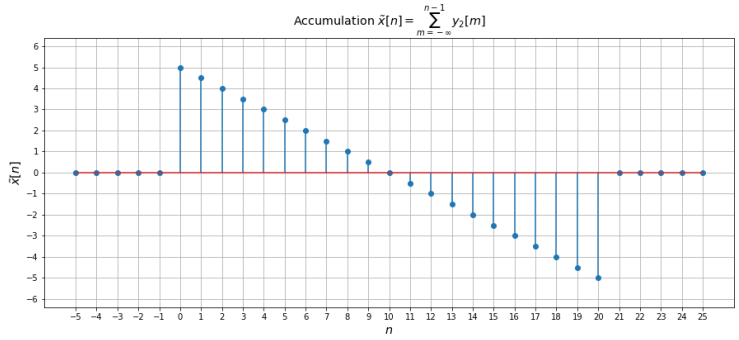
$$\tilde{x}[n] = \sum_{m=-\infty}^{n-1} y_2[m]$$

and plot it using MATLAB or python. Which of the three signals $x_1[n]$, $x_2[n]$, or $x_3[n]$ does it match? (8 points)

MATLAB code for calculating and plotting the accumulation $\tilde{x}[n]$ of $y_2[n]$, as given by the above formula is in Appendix E. This code generates the following plot.



Python code for calculating and plotting the accumulation $\tilde{x}[n]$ of $y_2[n]$, as given by the above formula is in Appendix F. This code generates the following plot.



Notice that $\tilde{x}[n]$ is identical to $x_1[n]$, even though it came from the forward difference of $x_2[n]$.

3. Determine whether the following signals are periodic and plot them. Create all plots in this problem using MATLAB or python. If a signal is complex, determine its magnitude and phase and find and plot its real and imaginary parts. For each periodic signal, determine its

   - fundamental period,
   - fundamental frequency,
   - fundamental angular frequency,
   - and average power.

   Use MATLAB or python to calculate the average power. Include the units for these quantities. Assume that the unit of $x[n]$ is furlongs per fortnight. (56 points total)

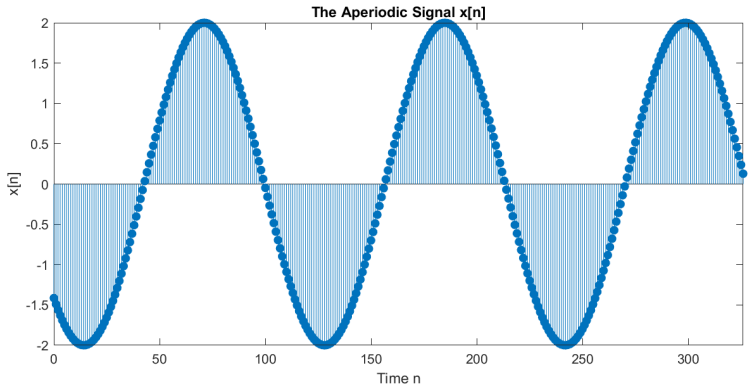
   (a) $x[n] = 2\sin\left(\frac{9n}{163} - \frac{3\pi}{4}\right)$ (10 points)

   Here

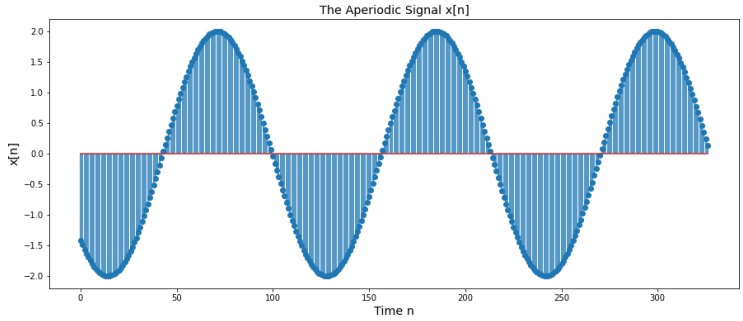
   $$x[n] = 2\sin\left(2\pi\left(\frac{9}{326\pi}\right)n - \frac{3\pi}{4}\right).$$

   So, the fundamental frequency is $F_0 = \frac{9}{326\pi}$, which is irrational. Therefore, $x[n]$ is aperiodic.

   MATLAB code for plotting this signal is in Appendix G. The output from this code is below.



Python code for plotting this signal is in Appendix H. The output from this code is below.



Notice that it is not apparent $x[n]$ is aperiodic from its graph.

(b) $x[n] = 2\cos\left(\frac{3\pi n}{163} - \frac{\pi}{4}\right)$ (10 points)

In this case,

$$x[n] = 2\cos\left(2\pi\left(\frac{3}{326}\right)n - \frac{\pi}{4}\right) = 2\cos\left(2\pi F_0 n - \frac{\pi}{4}\right).$$

So, $F_0 = \frac{3}{326}$, which is rational. Therefore, $x[n]$ is periodic. The fundamental period N_0 is the smallest integer that is a multiple of $\frac{1}{F_0} = \frac{326}{3}$, i.e.

$$N_0 = 326.$$

Technically, the fundamental frequency is

$$\hat{F}_0 = \frac{1}{N_0} = \frac{1}{326}$$

and the fundamental angular frequency is

$$\hat{\Omega}_0 = 2\pi\hat{F}_0 = \frac{\pi}{163}.$$

However, you will receive full credit if you said that the fundamental frequency is F_0 and the fundamental angular frequency is

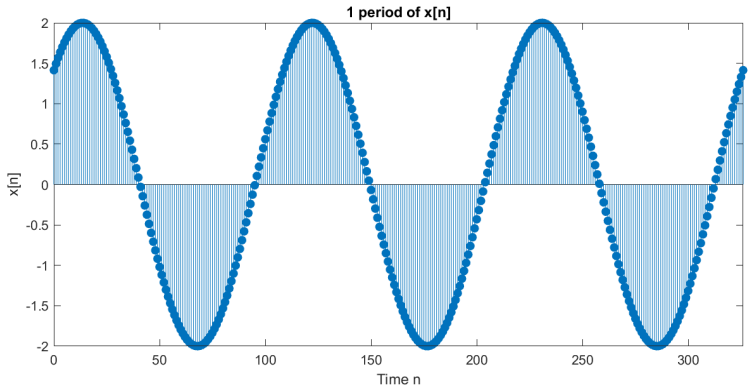
$$\Omega_0 = 2\pi F_0 = \frac{3\pi}{163}.$$

These are the values that a frequency detector would report.

The average power is

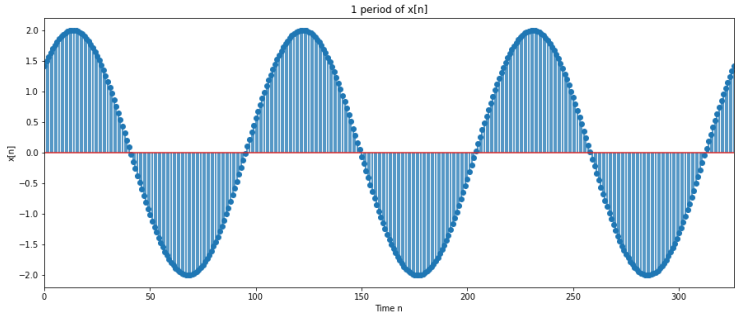
$$P_x = \frac{1}{N_0}\sum_{n=0}^{N_0-1}|x[n]|^2 = \frac{1}{326}\sum_{n=0}^{325}|x[n]|^2.$$

MATLAB code for plotting the signal and determining its average power is in Appendix I. Output from this code is below. Strictly speaking, I only plotted one period.



`The average power of x[n] is 2 (furlongs per fortnight)`^{2}.

Python code for plotting the signal and determining its average power is in Appendix J. Output from this code is below. As above, I only plotted one period.



`The average power of x[n] is 2.0 (furlongs per fortnight)`^{2}.

(c) $x[n] = 3e^{(-\pi/48+j2\pi/253)n}$ (12 points)

Using the laws of exponents and Euler's formula, this signal can be written as

$$x[n] = 3e^{-\pi n/48}e^{j2\pi n/253} = 3e^{-\pi n/48}\left(\cos\left(\frac{2\pi n}{253}\right) + j\sin\left(\frac{2\pi n}{253}\right)\right).$$

Since the first exponential factor decreases exponentially as n increases, this is an aperiodic signal. Another way to see this is to determine the magnitude of the signal. In this case,

$$|x[n]| = 3\left|e^{-\pi n/48}\right|\left|e^{j2\pi n/253}\right| = 3e^{-\pi n/48},$$

which exponentially decreases as n increases. The phase can be read from the cosine and sine terms (or you can use the formula) to find

$$\angle x[n] = \frac{2\pi n}{253}.$$

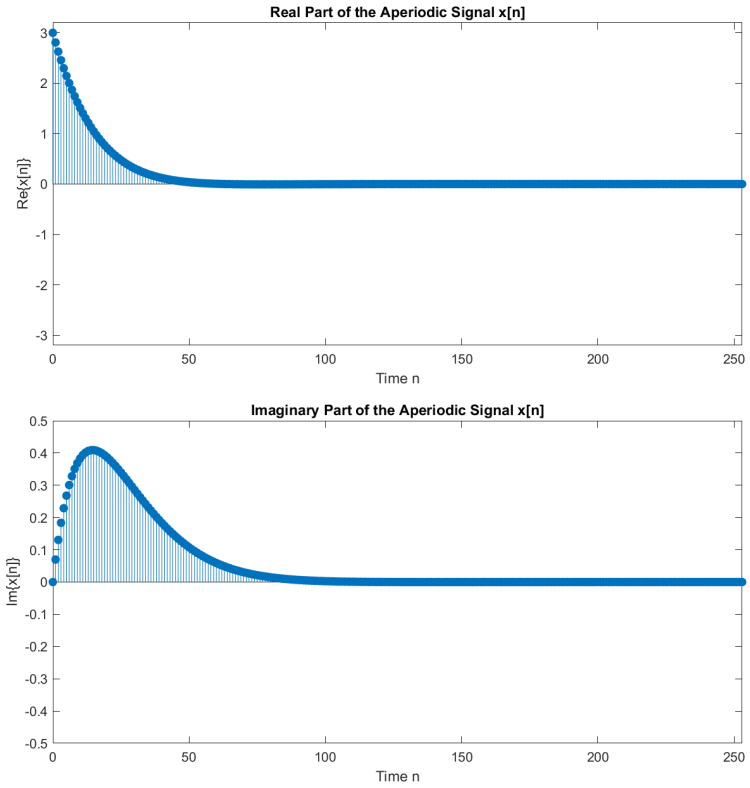
The real part of $x[n]$ is

$$\mathrm{Re}\{x[n]\} = 3e^{-\pi n/48}\cos\left(\frac{2\pi n}{253}\right)$$

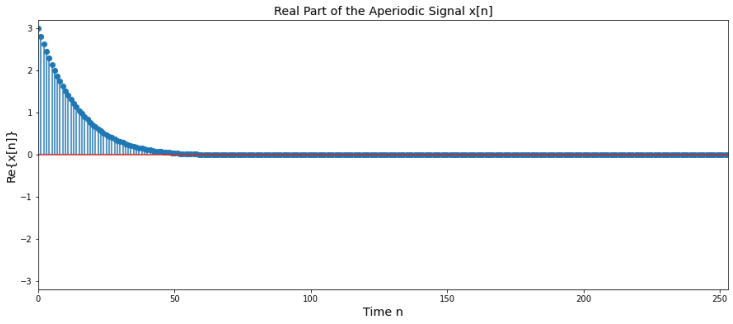
and the imaginary part of $x[n]$ is

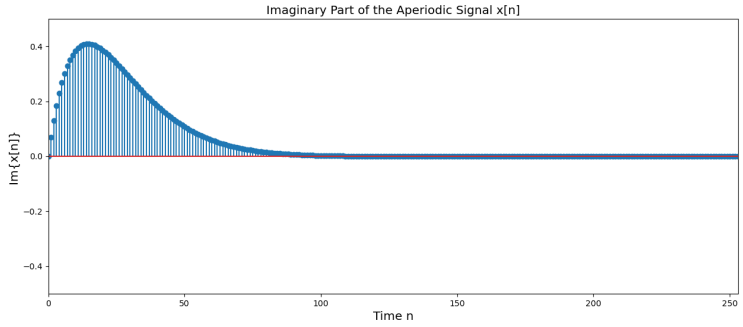
$$\mathrm{Im}\{x[n]\} = 3e^{-\pi n/48}\sin\left(\frac{2\pi n}{253}\right).$$

MATLAB code for plotting the real and imaginary parts of this signal is in Appendix K. The output from this code is below.



Python code for plotting the real and imaginary parts of this signal is in Appendix L. The output from this code is below.





(d) $x[n] = 3e^{j(8\pi n/287 - 3\pi/2)}$ (12 points)

In this case,

$$x[n] = 3e^{j(2\pi(4/287)n - 3\pi/2)} = 3e^{j(2\pi F_0 n - 3\pi/2)},$$

where $F_0 = \frac{4}{287}$. Since F_0 is rational, the signal is periodic. The fundamental period N_0 is the smallest integer that is a multiple of $1/F_0 = 287/4$, i.e.

$$N_0 = 287.$$

Technically, the fundamental frequency is

$$\hat{F}_0 = \frac{1}{N_0} = \frac{1}{287}$$

and the fundamental angular frequency is

$$\hat{\Omega}_0 = 2\pi\hat{F}_0 = \frac{2\pi}{287}.$$

However, you will receive full credit if you said that the fundamental frequency is F_0 and the fundamental angular frequency is

$$\Omega_0 = 2\pi F_0 = \frac{8\pi}{287}.$$

These are the values that a frequency detector would report.

The signal $x[n]$ is complex and has magnitude

$$|x[n]| = \left|3e^{j(8\pi n/287 - 3\pi/2)}\right| = 3\left|e^{j(8\pi n/287 - 3\pi/2)}\right| = 3$$

and phase

$$\angle x[n] = \frac{8\pi}{287}n - \frac{3\pi}{2}.$$

Using Euler's formula, it's possible to see that the real part of $x[n]$ is

$$\mathrm{Re}\{x[n]\} = 3\cos\left(\frac{8\pi}{287}n - \frac{3\pi}{2}\right)$$

and the imaginary part of $x[n]$ is

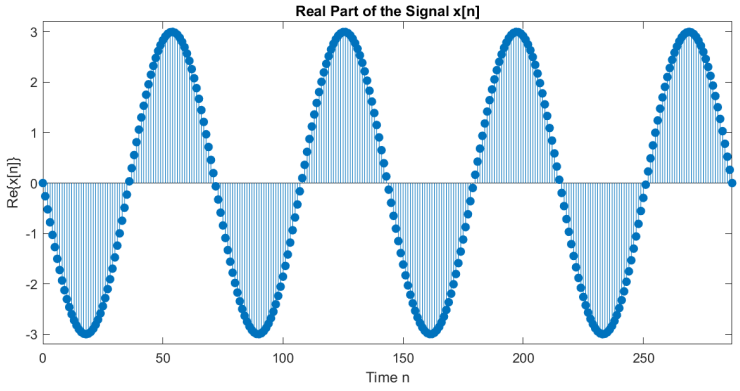
$$\mathrm{Im}\{x[n]\} = 3\sin\left(\frac{8\pi}{287}n - \frac{3\pi}{2}\right).$$

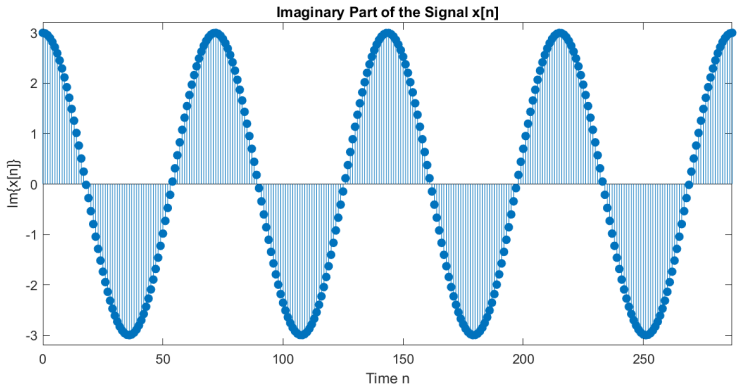
The average power of $x[n]$ is

$$\begin{aligned} P_x &= \frac{1}{N_0}\sum_{n=0}^{N_0-1}|x[n]|^2 = \frac{1}{287}\sum_{n=0}^{286}3^2 = \frac{1}{287}\sum_{n=0}^{286}9 \\ &= \frac{1}{287}(287 * 9) \\ &= 9 \text{ furlongs per fortnight}^2 \end{aligned}$$

If you used MATLAB or python to determine the average power, you should have obtained the same value.

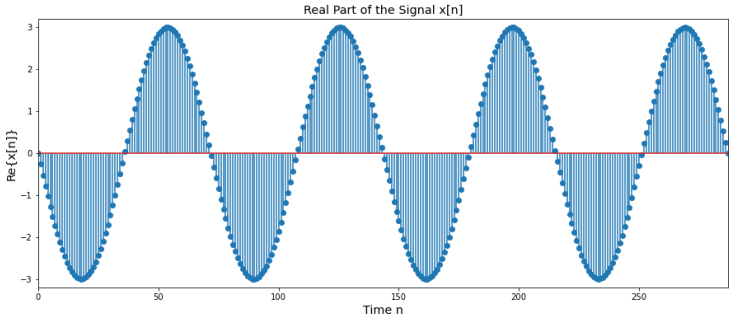
MATLAB code for plotting the real and imaginary parts and calculating the average power of $x[n]$ is in Appendix M. The output of this code is below. Notice that, technically, I plotted the real and imaginary parts for a single period.

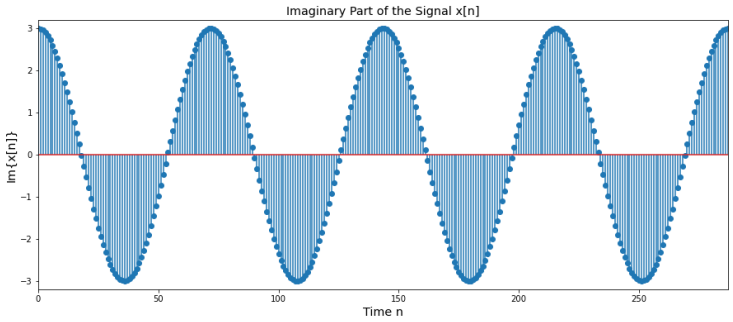




`The average power of x[n] is 9 (furlongs per fortnight)`^{2}.

Python code for plotting the real and imaginary parts and calculating the average power of $x[n]$ is in Appendix N. The output of this code is below. As above, I plotted the real and imaginary parts for a single period.





`The average power of x[n] is 9.0 (furlongs per fortnight)`^{2}.

(e) $x[n] = 3e^{j(-5n/96-\pi/3)}$ (12 points)

In this case,

$$x[n] = 3e^{j\left(2\pi\left(-\frac{5}{192\pi}\right)n-\frac{\pi}{3}\right)} = 3e^{j\left(2\pi F_0 n-\frac{\pi}{3}\right)}.$$

So, $F_0 = -\frac{5}{192\pi}$, which is irrational. Therefore, the signal is aperiodic.

This signal is complex and has magnitude

$$|x[n]| = \left|3e^{j\left(-\frac{5n}{96}-\frac{\pi}{3}\right)}\right| = |3|\left|e^{j\left(-\frac{5n}{96}-\frac{\pi}{3}\right)}\right| = 3$$

and phase

$$\angle x[n] = -\frac{5n}{96} - \frac{\pi}{3}.$$

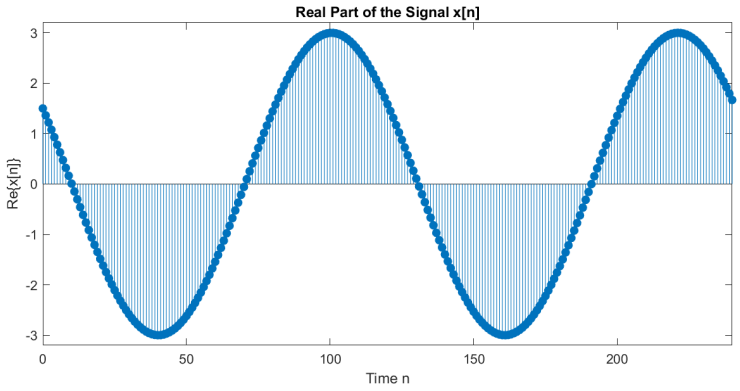
Euler's formula implies that the real part of $x[n]$ is

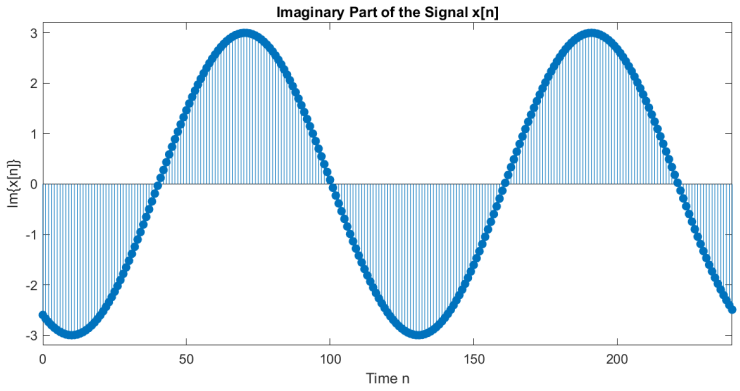
$$\mathrm{Re}\{x[n]\} = 3\cos\left(-\frac{5n}{96} - \frac{\pi}{3}\right)$$

and the imaginary part of $x[n]$ is

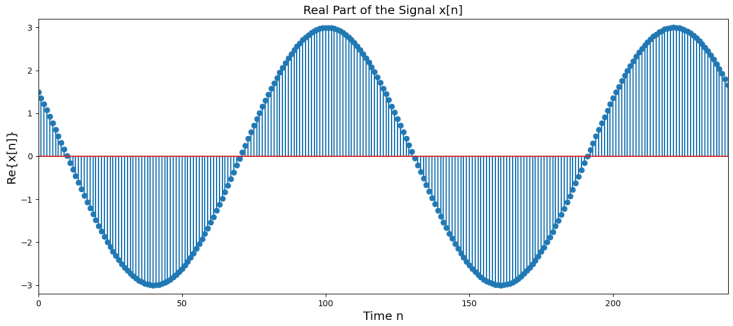
$$\mathrm{Im}\{x[n]\} = 3\sin\left(-\frac{5n}{96} - \frac{\pi}{3}\right).$$

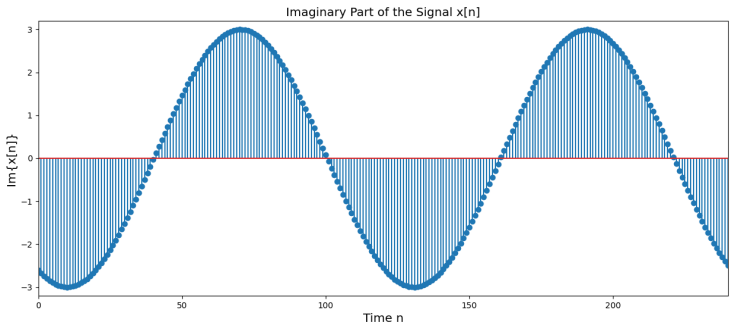
MATLAB code for plotting the real and imaginary parts of $x[n]$ is in Appendix O. The output from this code is below. Notice that it isn't apparent that $x[n]$ is aperiodic.





Python code for plotting the real and imaginary parts of $x[n]$ is in Appendix P. The output from this code is below. As above, notice that it isn't apparent that $x[n]$ is aperiodic.





4. Plot the real and imaginary parts of the two signals

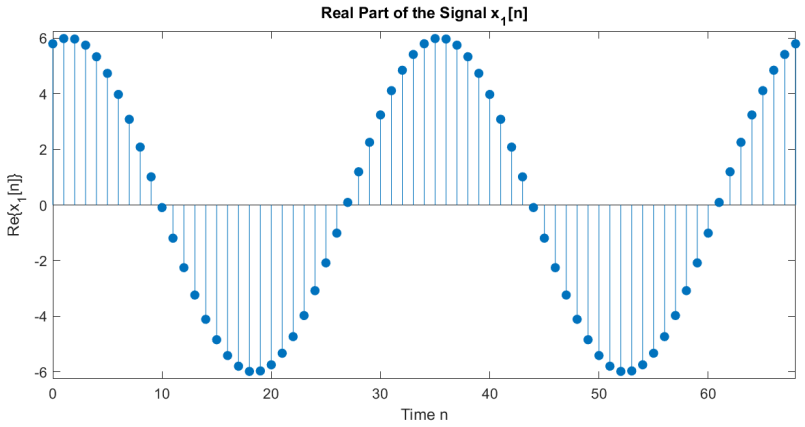
$$x_1[n] = 6\exp\left(j\left(\frac{3\pi n}{51} - \frac{\pi}{12}\right)\right)$$
$$x_2[n] = 6\exp\left(j\left(\frac{207\pi n}{51} - \frac{\pi}{12}\right)\right)$$

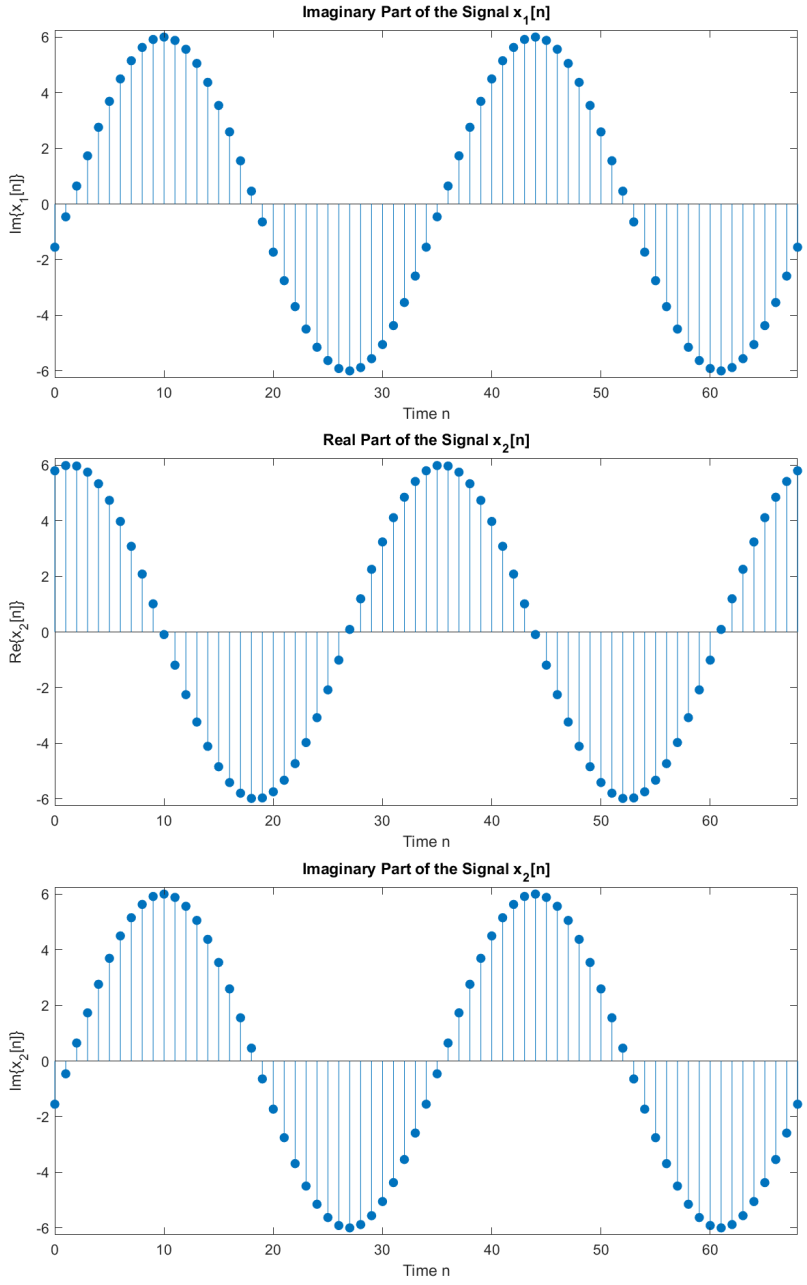
using MATLAB or python. Describe how the two signals are related to each other. (12 points)

The real and imaginary parts of $x_1[n]$ and $x_2[n]$ are

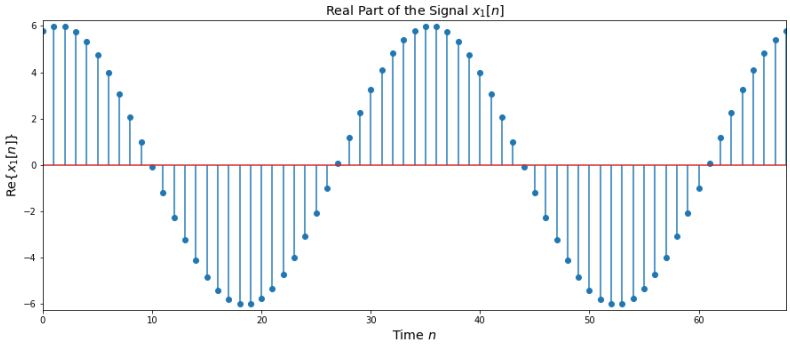
$$\mathrm{Re}\{x_1[n]\} = 6\cos\left(\frac{3\pi n}{51} - \frac{\pi}{12}\right)$$
$$\mathrm{Im}\{x_1[n]\} = 6\sin\left(\frac{3\pi n}{51} - \frac{\pi}{12}\right)$$
$$\mathrm{Re}\{x_2[n]\} = 6\cos\left(\frac{207\pi n}{51} - \frac{\pi}{12}\right)$$
$$\mathrm{Im}\{x_2[n]\} = 6\sin\left(\frac{207\pi n}{51} - \frac{\pi}{12}\right).$$

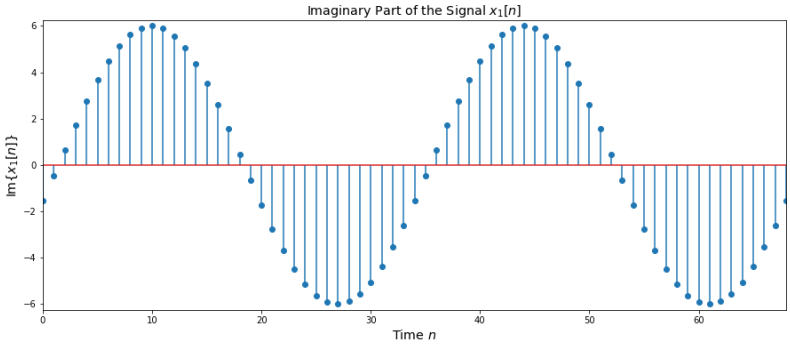
MATLAB code for plotting the real and imaginary parts of $x_1[n]$ and $x_2[n]$ is in Appendix Q. The output of this code is below.

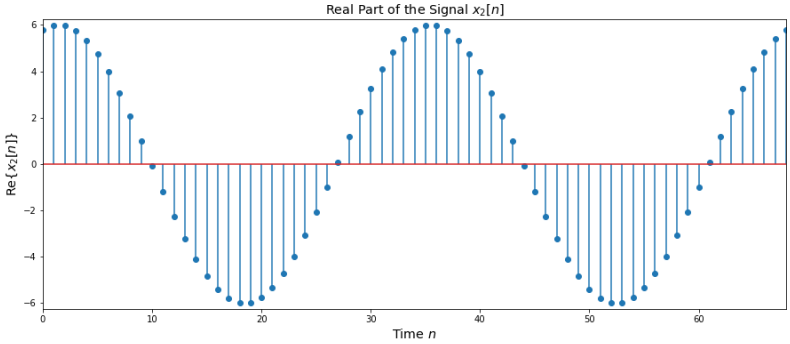


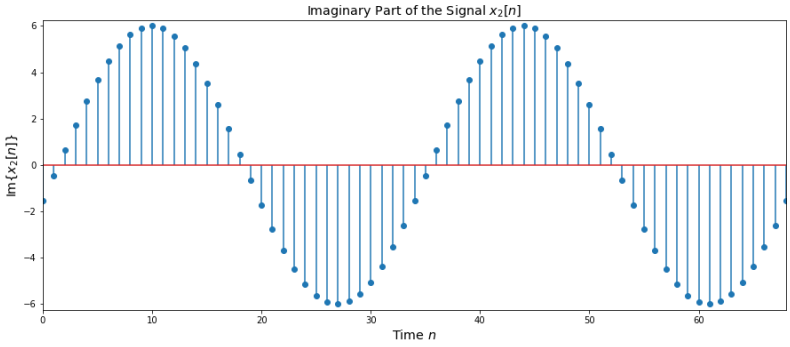


Python code for plotting the real and imaginary parts of $x_1[n]$ and $x_2[n]$ is in Appendix R. The output of this code is below.


Real Part of the Signal $x_1[n]$
$\mathrm{Re}\{x_1[n]\}$
Time n


Imaginary Part of the Signal $x_1[n]$
$\mathrm{Im}\{x_1[n]\}$
Time n


Real Part of the Signal $x_2[n]$
$\mathrm{Re}\{x_2[n]\}$
Time n



In this case,

$$x_1[n] = 6\exp\left(j\left(2\pi\left(\frac{3}{102}\right)n - \frac{\pi}{12}\right)\right) = 6\exp\left(j\left(2\pi F_{01}n - \frac{\pi}{12}\right)\right)$$

and

$$x_2[n] = 6\exp\left(j\left(2\pi\left(\frac{207}{102}\right)n - \frac{\pi}{12}\right)\right) = 6\exp\left(j\left(2\pi F_{02}n - \frac{\pi}{12}\right)\right),$$

where $F_{01} = 3/102$ and $F_{02} = 207/102$. Since

$$F_{02} - F_{01} = \frac{204}{102} = 2,$$

the signals $x_1[n]$ and $x_2[n]$ are identical.

5. Determine whether the following systems have the properties linearity, time invariance, BIBO stability, memory, causality, or invertibility. If the system is invertible, find its inverse. (56 points total)

   (a) $y(t) = (t^2 - 1)x(t - 1)$ (14 points)

   - Linearity

     Suppose that $y_1(t) = (t^2-1)x_1(t-1)$ and $y_2(t) = (t^2-1)x_2(t-1)$ for arbitrary signals $x_1(t)$ and $x_2(t)$ and that a and b are constants. If the input for the system is $\tilde{x}(t) = ax_1(t)+bx_2(t)$, then the corresponding output is

     $$\begin{aligned}\tilde{y}(t) &= (t^2-1)\tilde{x}(t-1) = (t^2-1)(ax_1(t-1) + bx_2(t-1)) \\ &= a(t^2-1)x_1(t-1) + b(t^2-1)x_2(t-1) \\ &= ay_1(t) + by_2(t).\end{aligned}$$

Therefore, the system, is linear.

- Time Invariance

  Suppose that $y(t) = (t^2 - 1)x(t - 1)$ for an arbitrary signal $x(t)$ and that $t_0 \neq 0$ is an arbitrary time delay. Then the output delayed by t_0 is

  $$y(t - t_0) = ((t - t_0)^2 - 1)x(t - t_0 - 1).$$

  If the input is $x_D(t) = x(t - t_0)$, then the output is

  $$\begin{aligned} y_D(t) &= (t^2 - 1)x_D(t - 1) = (t^2 - 1)x(t - t_0 - 1) \\ &\neq ((t - t_0)^2 - 1)x(t - t_0 - 1) = y(t - t_0). \end{aligned}$$

  Therefore, the system is time varying.

- BIBO Stability

  Suppose that $x(t) = 1$ for all t. Then $|x(t)| \leq 1$ for all t, which means that $x(t)$ is bounded. However, the corresponding output is

  $$y(t) = (t^2 - 1)x(t - 1) = t^2 - 1$$

  so that

  $$\lim_{t \to \infty} y(t) = \infty,$$

  which means that the output $y(t)$ is unbounded. Since a bounded input produces an unbounded output, the system is BIBO unstable.

- Memory

  Since the output at time t depends on the input at a past time $x(t - 1)$, the system has memory.

- Causality

  Since the output at time t depends on the input at a past time $x(t - 1)$ but no future times, the system is causal.

- Invertibility

Suppose that $x_1(t)$ is an arbitrary signal, $y_1(t) = (t^2-1)x_1(t-1)$ and define the signal

$$x_2(t) = \left\{ \begin{array}{ll} x_1(t), & t \neq 0 \\ x_1(0)+1, & t = 0 \end{array} \right\}$$

Then $x_2(t) \neq x_1(t)$ since $x_2(0) = x_1(0) + 1$. But the output of the system when $x_2(t)$ is the input is

$$\begin{aligned} y_2(t) = (t^2-1)x_2(t-1) &= \left\{ \begin{array}{ll} (t^2-1)x_1(t-1), & t \neq 1 \\ 0 \cdot x_2(0), & t = 1 \end{array} \right. \\ &= \left\{ \begin{array}{ll} (t^2-1)x_1(t-1), & t \neq 1 \\ 0, & t = 1 \end{array} \right. = \left\{ \begin{array}{ll} (t^2-1)x_1(t-1), & t \neq 1 \\ 0 \cdot x_1(0), & t = 1 \end{array} \right. \\ &= (t^2-1)x_1(t-1) = y_1(t) \text{ for all } t. \end{aligned}$$

Therefore, two distinct input signals produce the same output signal, which means that the system is not invertible.

(b) $y[n] = e^{-x[n]}$ (14 points)

- Linearity

  Suppose that $y_1[n] = e^{-x_1[n]}$ and $y_2[n] = e^{-x_2[n]}$ for arbitrary signals $x_1[n]$ and $x_2[n]$ and that a and b are constants. If the input for the system is $\tilde{x}[n] = ax_1[n] + bx_2[n]$, then the corresponding output is

  $$\begin{aligned} \tilde{y}[n] = e^{-\tilde{x}[n]} &= e^{-(ax_1[n]+bx_2[n])} = e^{-ax_1[n]}e^{-bx_2[n]} \\ &\neq ae^{x_1[n]} + be^{x_2[n]} = ay_1[n] + by_2[n]. \end{aligned}$$

  Therefore, the system, is nonlinear.

- Time Invariance

  Suppose that $y[n] = e^{-x[n]}$ for an arbitrary signal $x[n]$ and that $n_0 \neq 0$ is an arbitrary integer time delay. Then the output delayed by n_0 is

  $$y[n-n_0] = e^{-x[n-n_0]}.$$

  If the input is $x_D[n] = x[n-n_0]$, then the output is

  $$y_D[n] = e^{-x_D[n]} = e^{-x[n-n_0]} = y[n-n_0].$$

Therefore, the system is time invariant.

- BIBO Stability

  Suppose that $x[n]$ is a signal with $|x[n]| \leq B$ for some $0 \leq B < \infty$ and $y[n] = e^{-x[n]}$. Then,

  $$|y[n]| = \left|e^{-x[n]}\right|.$$

  If $x[n]$ is real-valued, then

  $$|y[n]| = e^{-x[n]} \leq e^{|-x[n]|} \leq e^{B}.$$

  If $x[n]$ is complex-valued, then $x[n] = x_r[n] + jx_i[n]$ and

  $$B^2 \geq |x[n]|^2 = x_r[n]^2 + x_i[n]^2 \geq x_r[n]^2 = |x_r[n]|^2.$$

  So,

  $$|x_r[n]| \leq B$$

  and

  $$\begin{aligned} |y[n]| &= \left|e^{-x[n]}\right| = \left|e^{-(x_r[n]+jx_i[n])}\right| = \left|e^{-x_r[n]}e^{-jx_i[n]}\right| \\ &= \left|e^{-x_r[n]}\right|\left|e^{-jx_i[n]}\right| = \left|e^{-x_r[n]}\right| = e^{-x_r[n]} \\ &\leq e^{|-x_r[n]|} \leq e^{B}. \end{aligned}$$

  Therefore, the system is BIBO stable.

- Memory

  Since the output at time n only depends on the input at time n, the system is memoryless.

- Causality

  Since the system is memoryless, it is causal.

- Invertibility

  Suppose that $x[n]$ is an arbitrary signal and $y[n] = e^{-x[n]}$ and define the system $\tilde{y}[n] = -\ln \tilde{x}[n]$. If $x[n]$ is real-valued and $\tilde{x}[n] = y[n]$, then

  $$\tilde{y}[n] = -\ln \tilde{x}[n] = -\ln y[n] = -\ln\left(e^{-x[n]}\right) = -(-x[n]) = x[n].$$

If $x[n]$ is complex-valued, taking the principal branch of the logarithm in the definition of the tilde system still allows the natural logarithm to be the inverse for this system. (Full credit if you assumed that $x[n]$ is real-valued and defined the same inverse system.) Therefore, the system is invertible.

(c) $y(t) = 1 - x(t)$ (14 points)

- Linearity

  Suppose that $y_1(t) = 1 - x_1(t)$ and $y_2(t) = 1 - x_2(t)$ for arbitrary signals $x_1(t)$ and $x_2(t)$ and that a and b are constants. If the input for the system is $\tilde{x}(t) = ax_1(t) + bx_2(t)$, then the corresponding output is

  $$\begin{aligned}\tilde{y}(t) &= 1 - \tilde{x}(t) = 1 - (ax_1(t) + bx_2(t)) = 1 - ax_1(t) - bx_2(t) \\ &\neq a - ax_1(t) + b - bx_2(t) = a(1 - x_1(t)) + b(1 - x_2(t)) \\ &= ay_1(t) + by_2(t)\end{aligned}$$

  since $a + b \neq 1$ for arbitrary a and b. Therefore, the system, is nonlinear.

- Time Invariance

  Suppose that $y(t) = 1 - x(t)$ for an arbitrary signal $x(t)$ and that $t_0 \neq 0$ is an arbitrary time delay. Then the output delayed by t_0 is

  $$y(t - t_0) = 1 - x(t - t_0).$$

  If the input is $x_D(t) = x(t - t_0)$, then the output is

  $$y_D(t) = 1 - x_D(t) = 1 - x(t - t_0) = y(t - t_0).$$

  Therefore, the system is time invariant.

- BIBO Stability

  Suppose that $x(t)$ is a signal with $|x(t)| \leq B$ for some $0 \leq B < \infty$ and $y(t) = 1 - x(t)$. Then, using the triangle inequality,

  $$|y(t)| = |1 - x(t)| \leq |1| + |x(t)| \leq 1 + B < \infty \text{ for all} t.$$

  Therefore, the system is BIBO stable.

- Memory

  Since the output at time t only depends on the value of the input at time t, the system is memoryless.

- Causality

  Since the system is memoryless, it is also causal.

- Invertibility

  Let $y(t) = 1 - x(t)$ for an arbitrary signal $x(t)$. If $y(t)$ is the input to the system, then the output is

  $$\tilde{y}(t) = 1 - y(t) = 1 - (1 - x(t)) = x(t).$$

  Therefore, the system is invertible and is its own inverse.

(d) $y[n] = \sum_{k=n_0}^{n} x[n-k]$ where n_0 is a fixed, finite time (14 points)

I didn't define this part properly. So, you'll receive full credit, regardless of your answer. I rewrote the problem above in the way I originally intended.

- Linearity

  Suppose that $y_1[n] = \sum_{k=n_0}^{n} x_1[n-k]$ and $y_2[n] = \sum_{k=n_0}^{n} x_1[n-k]$ for arbitrary signals $x_1[n]$ and $x_2[n]$ and that a and b are constants. If the input for the system is $\tilde{x}[n] = ax_1[n]+bx_2[n]$, then the corresponding output is

  $$\begin{aligned}\tilde{y}[n] &= \sum_{k=n_0}^{n} \tilde{x}[n-k] = \sum_{k=n_0}^{n} (ax_1[n-k] + bx_2[n-k]) \\ &= a\sum_{k=n_0}^{n} x_1[n-k] + b\sum_{k=n_0}^{n} x_2[n-k] \\ &= ay_1[n] + by_2[n].\end{aligned}$$

  Therefore, the system, is linear.

- Time Invariance

  Suppose that $y[n] = \sum_{k=n_0}^{n} x[n-k]$ for an arbitrary signal $x[n]$ and that $\hat{n} \neq 0$ is an arbitrary integer time delay. Then

the output delayed by $\hat{n}$ is

$$y[n-\hat{n}] = \sum_{k=n_0}^{n-\hat{n}} x[n-\hat{n}-k].$$

If the input is $x_D[n] = x[n-\hat{n}]$, then the output is

$$\begin{aligned} y_D[n] &= \sum_{k=n_0}^{n} x_D[n-k] = \sum_{k=n_0}^{n} x[n-\hat{n}-k] \\ &= \begin{cases} \sum_{k=n_0}^{n-\hat{n}} x[n-\hat{n}-k] + \sum_{k=n-\hat{n}+1}^{n} x[n-\hat{n}-k] & \hat{n} > 0 \\ \sum_{k=n_0}^{n-\hat{n}} x[n-\hat{n}-k] & \hat{n} = 0 \\ \sum_{k=n_0}^{n-\hat{n}} x[n-\hat{n}-k] - \sum_{k=n+1}^{n-\hat{n}} x[n-\hat{n}-k] & \hat{n} < 0 \end{cases} \\ &\neq y[n-\hat{n}] \end{aligned}$$

Therefore, the system is time-varying.

- BIBO Stability

  If $x[n] = 1$ for all n, then $|x[n]| \leq 1$, which means that $x[n]$ is bounded. Let $y[n] = \sum_{k=n_0}^{n} x[n-k]$. Then the sum contains $|n-n_0|+1$ terms and

  $$y[n] = \sum_{k=n_0}^{n} 1 = |n-n_0| + 1.$$

  Therefore,

  $$\lim_{n\to\infty} y[n] = \lim_{n\to\infty} |n-n_0| + 1 = \infty.$$

  Since a bounded input produces an unbounded output, the system is unstable.

- Memory

  Since the output at time n depends on past or future values of the input, the system has memory.

- Causality

  If $n < 0$, then the output at time n depends on future values of the input, which means that the system is acausal.

- Invertibility

  This system is invertible, which can be shown by induction. But this is too complicated and I am giving you full credit, regardless of your answer.

A MATLAB Code for Problem 2, Part (a)

```
%---------------------------------%
% Signals for Problem 2, Part (a) %
%---------------------------------%

nmin = -5;
nmax = 25;
xmin = -6.0;
xmax =  6.0;
dx   =  0.4;

n = nmin:nmax;

x1 = x1prob2(n);

figure('Position', [0, 0, 900, 400])
stem(n, x1, 'filled')
ylim([xmin - dx, xmax + dx])
xticks(n)
yticks(xmin:xmax)
xlabel("n")
ylabel("x_{1}[n]")
title("Signal x_{1}[n]")
grid on
saveas(gcf, 'prob2a_x1_matlab.png')

x1shift = x1prob2(n + 1);
y1 = x1shift - x1;

figure('Position', [40, 40, 900, 400])
stem(n, y1, 'filled')
ylim([xmin - dx, xmax + dx])
xticks(n)
yticks(xmin:xmax)
xlabel("n")
ylabel("y_{1}[n]")
title("Backward Difference y_{1}[n] = x_{1}[n + 1] - x_{1}[n]")
```

```
grid on
saveas(gcf, 'prob2a_y1_matlab.png')

x2 = x2prob2(n);

figure('Position', [80, 80, 900, 400])
stem(n, x2, 'filled')
ylim([xmin - dx, xmax + dx])
xticks(n)
yticks(xmin:xmax)
xlabel("n")
ylabel("x_{2}[n]")
title("Signal x_{2}[n]")
grid on
saveas(gcf, 'prob2a_x2_matlab.png')

x2shift = x2prob2(n + 1);
y2 = x2shift - x2;

figure('Position', [120, 120, 900, 400])
stem(n, y2, 'filled')
ylim([xmin - dx, xmax + dx])
xticks(n)
yticks(xmin:xmax)
xlabel("n")
ylabel("y_{2}[n]")
title("Backward Difference y_{2}[n] = x_{2}[n + 1] - x_{2}[n]")
grid on
saveas(gcf, 'prob2a_y2_matlab.png')

function x = x1prob2(n)

    x = zeros(size(n));

    index = 1;
    for m = n
        if m >= 0 && m <= 20
            x(index) = 5.0 - 0.5*m;
```

```
        end
        index = index + 1;
    end

end

function x = x2prob2(n)

    x = zeros(size(n));

    index = 1;
    for m = n
        if m >= 0 && m <= 20
            x(index) = 5.0 - 0.5*m - 1;
        else
            x(index) = -1;
        end
        index = index + 1;
    end

end
```

B Python Code for Problem 2, Part (a)

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

#---------------------------------#
# Signals for Problem 2, Part (a) #
#---------------------------------#

def x1prob2(n):

    x = np.zeros(np.size(n))

    index = 0
    for m in n:
```

```
        if m >= 0 and m <= 20:
            x[index] = 5.0 - 0.5*m
        index += 1

    return x

nmin = -5
nmax = 25
xmin = -6
xmax =  6
dx   =  0.4

n = np.arange(nmin, nmax + 1)

x1 = x1prob2(n)

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, x1)
ax.set_ylim([xmin - dx, xmax + dx])
ax.set_xticks(n)
ax.set_yticks(np.arange(xmin, xmax + 1, 1))
ax.set_xlabel("$n$", fontsize="x-large")
ax.set_ylabel("$x_{1}[n]$", fontsize="x-large")
plt.title("Signal $x_{1}[n]$", fontsize="x-large")
ax.grid(True)
plt.savefig('prob2a_x1_python.png')
plt.show()

x1shift = x1prob2(n + 1)
y1 = x1shift - x1

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, y1)
ax.set_ylim([xmin - dx, xmax + dx])
ax.set_xticks(n)
ax.set_yticks(np.arange(xmin, xmax + 1, 1))
ax.set_xlabel("$n$", fontsize="x-large")
ax.set_ylabel("$y_{1}[n]$", fontsize="x-large")
```

```
plt.title("Backward Difference $y_{1}[n] = x_{1}[n + 1] - x_{1}[n]$",\
          fontsize="x-large")
ax.grid(True)
plt.savefig('prob2a_y1_python.png')
plt.show()

def x2prob2(n):

    x = np.zeros(np.size(n))

    index = 0
    for m in n:
        if m >= 0 and m <= 20:
            x[index] = 5.0 - 0.5*m - 1
        else:
            x[index] = -1
        index += 1

    return x

x2 = x2prob2(n)

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, x2)
ax.set_ylim([xmin - dx, xmax + dx])
ax.set_xticks(n)
ax.set_yticks(np.arange(xmin, xmax + 1, 1))
ax.set_xlabel("$n$", fontsize="x-large")
ax.set_ylabel("$x_{2}[n]$", fontsize="x-large")
plt.title("Signal $x_{2}[n]$", fontsize="x-large")
ax.grid(True)
plt.savefig('prob2a_x2_python.png')
plt.show()

x2shift = x2prob2(n + 1)
y2 = x2shift - x2

fig, ax = plt.subplots(1, figsize=(15,6))
```

```
ax.stem(n, y2)
ax.set_ylim([xmin - dx, xmax + dx])
ax.set_xticks(n)
ax.set_yticks(np.arange(xmin, xmax + 1, 1))
ax.set_xlabel("$n$", fontsize="x-large")
ax.set_ylabel("$y_{2}[n]$", fontsize="x-large")
plt.title("Backward Difference $y_{2}[n] = x_{2}[n + 1] - x_{2}[n]$",\
          fontsize="x-large")
ax.grid(True)
plt.savefig('prob2a_y2_python.png')
plt.show()
```

C MATLAB Code for Problem 2, Part (b)

```
%---------------------------------%
% Signals for Problem 2, Part (b) %
%---------------------------------%

nmin = -5;
nmax = 25;
xmin = -6.0;
xmax =  6.0;
dx   =  0.4;

n = nmin:nmax;

x3 = x3prob2(n);

figure('Position', [0, 0, 900, 400])
stem(n, x3, 'filled')
ylim([xmin - dx, xmax + dx])
xticks(n)
yticks(xmin:xmax)
xlabel("n")
ylabel("x_{3}[n]")
title("Signal x_{3}[n]")
grid on
```

```
saveas(gcf, 'prob2b_x3_matlab.png')

x3shift = x3prob2(n + 1);
y3 = x3shift - x3;

figure('Position', [50, 50, 900, 400])
stem(n, y3, 'filled')
ylim([xmin - dx, xmax + dx])
xticks(n)
yticks(xmin:xmax)
xlabel("n")
ylabel("y_{3}[n]")
title("Backward Difference y_{3}[n] = x_{3}[n + 1] - x_{3}[n]")
grid on
saveas(gcf, 'prob2b_y3_matlab.png')

function x = x3prob2(n)

    x = zeros(size(n));

    index = 1;
    for m = n
        if m >= 0 && m <= 20
            x(index) = 5.0 - 0.5*m - 1.0;
        end
        index = index + 1;
    end

end
```

D Python Code for Problem 2, Part (b)

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

#---------------------------------#
# Signals for Problem 2, Part (b) #
```

```
#--------------------------------#

def x3prob2(n):

    x = np.zeros(np.size(n))

    index = 0
    for m in n:
        if m >= 0 and m <= 20:
            x[index] = 5.0 - 0.5*m - 1.0
        index += 1

    return x

nmin = -5
nmax = 25
xmin = -6
xmax =  6
dx   =  0.4

n = np.arange(nmin, nmax + 1)

x3 = x3prob2(n)

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, x3)
ax.set_ylim([xmin - dx, xmax + dx])
ax.set_xticks(n)
ax.set_yticks(np.arange(xmin, xmax + 1, 1))
ax.set_xlabel("$n$", fontsize="x-large")
ax.set_ylabel("$x_{3}[n]$", fontsize="x-large")
plt.title("Signal $x_{3}[n]$", fontsize="x-large")
ax.grid(True)
plt.savefig('prob2b_x3_python.png')
plt.show()

x3shift = x3prob2(n + 1)
y3 = x3shift - x3
```

```
fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, y3)
ax.set_ylim([xmin - dx, xmax + dx])
ax.set_xticks(n)
ax.set_yticks(np.arange(xmin, xmax + 1, 1))
ax.set_xlabel("$n$", fontsize="x-large")
ax.set_ylabel("$y_{3}[n]$", fontsize="x-large")
plt.title("Backward Difference $y_{3}[n] = x_{3}[n + 1] - x_{3}[n]$",\
          fontsize="x-large")
ax.grid(True)
plt.savefig('prob2b_y3_python.png')
plt.show()
```

E MATLAB Code for Problem 2, Part (c)

```
%---------------------------------%
% Signals for Problem 2, Part (c) %
%---------------------------------%

x2 = x2prob2(n);
x2shift = x2prob2(n + 1);
y2 = x2shift - x2;

maxIndex = length(n);
xtilde   = zeros(size(n));

xtilde(1) = 0.0;
for index = 2:maxIndex
    xtilde(index) = xtilde(index - 1) + y2(index - 1);
end

figure('Position', [0, 0, 900, 400])
stem(n, xtilde, 'filled')
ylim([xmin - dx, xmax + dx])
xticks(n)
yticks(xmin:xmax)
```

```
xlabel("$n$", 'Interpreter', 'latex')
ylabel("$\tilde{x}[n]$", 'Interpreter', 'latex')
title("Accumulation $\tilde{x}[n] = \sum_{m = -\infty}^{n - 1}y_{2}[m]$", ...
      'Interpreter', 'latex')
grid on
saveas(gcf, 'prob2c_matlab.png')

function x = x2prob2(n)

    x = zeros(size(n));

    index = 1;
    for m = n
        if m >= 0 && m <= 20
            x(index) = 5.0 - 0.5*m - 1;
        else
            x(index) = -1;
        end
        index = index + 1;
    end

end
```

F Python Code for Problem 2, Part (c)

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

#---------------------------------#
# Signals for Problem 2, Part (c) #
#---------------------------------#

def x2prob2(n):

    x = np.zeros(np.size(n))

    index = 0
```

```
    for m in n:
        if m >= 0 and m <= 20:
            x[index] = 5.0 - 0.5*m - 1
        else:
            x[index] = -1
        index += 1

    return x

x2 = x2prob2(n)
x2shift = x2prob2(n + 1)
y2 = x2shift - x2

# Calculate and plot the accumulation of y2.

maxIndex = np.size(n)
xtilde   = np.zeros(maxIndex)

xtilde[0] = 0.0
for index in np.arange(1, maxIndex):
    xtilde[index] = xtilde[index - 1] + y2[index - 1]

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, xtilde)
ax.set_ylim([xmin - dx, xmax + dx])
ax.set_xticks(n)
ax.set_yticks(np.arange(xmin, xmax + 1, 1))
ax.set_xlabel("$n$", fontsize="x-large")
ax.set_ylabel("$\\tilde{x}[n]$", fontsize="x-large")
plt.title("Accumulation $\\tilde{x}[n] = \\sum_{m = -\\infty}^{n - 1}y_{2}[m]$",\
          fontsize="x-large")
ax.grid(True)
plt.savefig('prob2c_python.png')
plt.show()
```

G MATLAB Code for Problem 3, Part (a)

```
%-----------------------------------------%
% Plot the signal from Problem 3, Part (a) %
%-----------------------------------------%

N = 326;

n = 0:N;
x = 2.0*sin((9.0/163.0)*n - 0.75*pi);

figure('Position', [0, 0, 900, 400])
stem(n, x, 'filled')
xlim([0, N])
xlabel('Time n')
ylabel('x[n]')
title('The Aperiodic Signal x[n]')
saveas(gcf, 'prob3a_matlab.png')
```

H Python Code for Problem 3, Part (a)

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

#-------------------------------#
# Signal for Problem 3, Part (a) #
#-------------------------------#

N = 326

n = np.arange(0, N+1)
x = 2.0*np.sin((9.0/163.0)*n - 0.75*np.pi)

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, x)
ax.set_xlabel('Time n', fontsize="x-large")
ax.set_ylabel('x[n]', fontsize="x-large")
```

```
plt.savefig('prob3a_python.png')
plt.show()
```

I MATLAB Code for Problem 3, Part (b)

```
%----------------------------------%
% Plot the signal from Problem 3b. %
%----------------------------------%

N0 = 326;       % period
numPeriods = 1; % number of periods to plot, starting from n = 0
lastn = numPeriods*N0;

n = 0:lastn; % time vector
x = 2.0*cos((3.0*pi/163.0)*n - 0.25*pi);

figure('Position', [0, 0, 900, 400])
stem(n, x, 'filled')
xlim([0, lastn])
if numPeriods == 1
    ending = '';
else
    ending = 's';
end
titlestr = sprintf('%d period%s of x[n]', numPeriods, ending);
title(titlestr)
xlabel('Time n')
ylabel('x[n]')
saveas(gcf, 'prob3b_matlab.png')

avgPower = dot(x(1:N0),x(1:N0))/N0;
powerstr = sprintf('The average power of x[n] is %g (furlongs per fortnight)^2.',
disp(powerstr)
```

J Python Code for Problem 3, Part (b)

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

#-------------------------------#
# Signal for Problem 3, Part (b) #
#-------------------------------#

N0 = 326       # period
numPeriods = 1 # number of periods to plot, starting from n = 0
lastn = numPeriods*N0

n = np.arange(0, lastn + 1)  # time vector
x = 2.0*np.cos((3.0*np.pi/163.0)*n - 0.25*np.pi)

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, x)
ax.set_xlim([0, lastn])
if numPeriods == 1:
    ending = ''
else:
    ending = 's'
ax.set_title('{0} period{1} of x[n]'.format(numPeriods, ending))
ax.set_xlabel('Time n')
ax.set_ylabel('x[n]')
plt.savefig('prob3b_python.png')
plt.show()

avgPower = np.dot(x[0:N0],x[0:N0])/N0;
print('The average power of x[n] is {0} (units of x)^2.'.format(avgPower))
```

K MATLAB Code for Problem 3, Part (c)

```
%-------------------------------------------------------------------------%
% Plot the real and imaginary parts of the signal from Problem 3, Part (c) %
%-------------------------------------------------------------------------%
```

```
lastn = 253;
n = 0:lastn;  % time vector
mag = 3.0*exp(-(pi/48.0)*n);
phase = (2.0*pi/253.0)*n;

xre = mag.*cos(phase); % real part of x[n]
xim = mag.*sin(phase); % imag part of x[n]

figure('Position', [0, 0, 900, 400])
stem(n, xre, 'filled')
xlim([0, lastn])
ylim([-3.2, 3.2])
title('Real Part of the Aperiodic Signal x[n]')
xlabel('Time n')
ylabel('Re\{x[n]\}')
saveas(gcf, 'prob3cre_matlab.png')

figure('Position', [50, 50, 900, 400])
stem(n, xim, 'filled')
xlim([0, lastn])
ylim([-0.5, 0.5])
title('Imaginary Part of the Aperiodic Signal x[n]')
xlabel('Time n')
ylabel('Im\{x[n]\}')
saveas(gcf, 'prob3cim_matlab.png')
```

L Python Code for Problem 3, Part (c)

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

#-------------------------------------------------------------#
# Real and Imaginary Parts of Signal from Problem 3, Part (c) #
#-------------------------------------------------------------#

lastn = 253
```

```
n = np.arange(0, lastn + 1)  # time vector
mag = 3.0*np.exp(-(np.pi/48.0)*n)
phase = (2.0*np.pi/253.0)*n

xre = mag*np.cos(phase) # real part of x[n]
xim = mag*np.sin(phase) # imag part of x[n]

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, xre)
ax.set_xlim([0, lastn])
ax.set_ylim([-3.2, 3.2])
ax.set_title('Real Part of the Aperiodic Signal x[n]')
ax.set_xlabel('Time n')
ax.set_ylabel('Re{x[n]}')
plt.savefig('prob3cre_python.png')
plt.show()

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, xim)
ax.set_xlim([0, lastn])
ax.set_ylim([-1.2, 1.2])
ax.set_title('Imaginary Part of the Aperiodic Signal x[n]')
ax.set_xlabel('Time n')
ax.set_ylabel('Im{x[n]}')
plt.savefig('prob3cim_python.png')
plt.show()
```

M MATLAB Code for Problem 3, Part (d)

```
%-------------------------------------------------------------------------%
% Plot the real and imaginary parts of the signal from Problem 3, Part (d) %
%-------------------------------------------------------------------------%

N0 = 287;       % period
numPeriods = 1;
numTimes = N0*numPeriods;
n = 0:numTimes; % time vector
```

```
phase = (8.0*pi/287.0)*n - 1.5*pi;

xre = 3.0*cos(phase); % real part of x[n]
xim = 3.0*sin(phase); % imag part of x[n]

figure('Position', [0, 0, 900, 400])
stem(n, xre, 'filled')
xlim([0, numTimes])
ylim([-3.2, 3.2])
title('Real Part of the Signal x[n]')
xlabel('Time n')
ylabel('Re\{x[n]\}')
saveas(gcf, 'prob3dre_matlab.png')

figure('Position', [50, 50, 900, 400])
stem(n, xim, 'filled')
xlim([0, numTimes])
ylim([-3.2, 3.2])
title('Imaginary Part of the Signal x[n]')
xlabel('Time n')
ylabel('Im\{x[n]\}')
saveas(gcf, 'prob3dim_matlab.png')

avgPower = sum(xre(1:N0).*xre(1:N0) + xim(1:N0).*xim(1:N0))/N0;
powerstr = sprintf('The average power of x[n] is %g (furlongs per fortnight)^2.',
disp(powerstr)
```

N Python Code for Problem 3, Part (d)

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

#------------------------------------------------------------------------#
# Real and Imaginary Parts and Power of Signal from Problem 3, Part (d) #
#------------------------------------------------------------------------#

N0 = 287 # period
```

```
numPeriods = 1
numTimes = N0*numPeriods
n = np.arange(0, numTimes + 1) # time vector
phase = (8.0*np.pi/287.0)*n - 1.5*np.pi

xre = 3.0*np.cos(phase) # real part of x[n]
xim = 3.0*np.sin(phase) # imag part of x[n]

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, xre)
ax.set_xlim([0, numTimes])
ax.set_ylim([-3.2, 3.2])
ax.set_title('Real Part of the Signal x[n]', fontsize="x-large")
ax.set_xlabel('Time n', fontsize="x-large")
ax.set_ylabel('Re{x[n]}', fontsize="x-large")
plt.savefig('prob3dre_python.png')
plt.show()

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, xim)
ax.set_xlim([0, numTimes])
ax.set_ylim([-3.2, 3.2])
ax.set_title('Imaginary Part of the Signal x[n]', fontsize="x-large")
ax.set_xlabel('Time n', fontsize="x-large")
ax.set_ylabel('Im{x[n]}', fontsize="x-large")
plt.savefig('prob3dim_python.png')
plt.show()

avgPower = np.sum(xre[0:N0]*xre[0:N0] + xim[0:N0]*xim[0:N0])/N0;
print('The average power of x[n] is {0} (furlongs per fortnight)^2.'\
      .format(avgPower))
```

O MATLAB Code for Problem 3, Part (e)

```
%-------------------------------------------------------------------------%
% Plot the real and imaginary parts of the signal from Problem 3, Part (e) %
%-------------------------------------------------------------------------%
```

```
lastn = 240;
n = 0:lastn; % time vector
phase = -(5.0/96.0)*n - pi/3.0;

xre = 3.0*cos(phase); % real part of x[n]
xim = 3.0*sin(phase); % imag part of x[n]

figure('Position', [0, 0, 900, 400])
stem(n, xre, 'filled')
xlim([0, lastn])
ylim([-3.2, 3.2])
title('Real Part of the Signal x[n]')
xlabel('Time n')
ylabel('Re\{x[n]\}')
saveas(gcf, 'prob3ere_matlab.png')

figure('Position', [50, 50, 900, 400])
stem(n, xim, 'filled')
xlim([0, lastn])
ylim([-3.2, 3.2])
title('Imaginary Part of the Signal x[n]')
xlabel('Time n')
ylabel('Im\{x[n]\}')
saveas(gcf, 'prob3eim_matlab.png')
```

P Python Code for Problem 3, Part (e)

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

#-------------------------------------------------------------#
# Real and Imaginary Parts of Signal from Problem 3, Part (e) #
#-------------------------------------------------------------#

lastn = 240
n = np.arange(0, lastn + 1)  # time vector
```

```
phase = -(5.0/96.0)*n - np.pi/3.0

xre = 3.0*np.cos(phase) # real part of x[n]
xim = 3.0*np.sin(phase) # imag part of x[n]

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, xre)
ax.set_xlim([0, lastn])
ax.set_ylim([-3.2, 3.2])
ax.set_title('Real Part of the Signal x[n]', fontsize="x-large")
ax.set_xlabel('Time n', fontsize="x-large")
ax.set_ylabel('Re{x[n]}', fontsize="x-large")
plt.savefig('prob3ere_python.png')
plt.show()

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, xim)
ax.set_xlim([0, lastn])
ax.set_ylim([-3.2, 3.2])
ax.set_title('Imaginary Part of the Signal x[n]', fontsize="x-large")
ax.set_xlabel('Time n', fontsize="x-large")
ax.set_ylabel('Im{x[n]}', fontsize="x-large")
plt.savefig('prob3eim_python.png')
plt.show()
```

Q MATLAB Code for Problem 4

```
%----------------------------------%
% Plot the signals from Problem 4. %
%----------------------------------%

N = 34; % period
numPeriods = 2; % number of periods to plot, starting from n = 0
numTimes = N*numPeriods;

n = 0:numTimes; % time vector
phase1 = (pi/17.0)*n - pi/12.0;       %   3/51 =  1/17
```

```
phase2 = (69.0*pi/17.0)*n - pi/12.0;  % 207/51 = 69/17

x1re = 6.0*cos(phase1); % real part of x1
x1im = 6.0*sin(phase1); % imag part of x1
x2re = 6.0*cos(phase2); % real part of x2
x2im = 6.0*sin(phase2); % imag part of x2

figure('Position', [0, 0, 900, 400])
stem(n, x1re, 'filled')
xlim([0, numTimes])
ylim([-6.25, 6.25])
title('Real Part of the Signal x_{1}[n]')
xlabel('Time n')
ylabel('Re\{x_{1}[n]\}')
saveas(gcf, 'prob4x1re_matlab.png')

figure('Position', [50, 50, 900, 400])
stem(n, x1im, 'filled')
xlim([0, numTimes])
ylim([-6.25, 6.25])
title('Imaginary Part of the Signal x_{1}[n]')
xlabel('Time n')
ylabel('Im\{x_{1}[n]\}')
saveas(gcf, 'prob4x1im_matlab.png')

figure('Position', [100, 100, 900, 400])
stem(n, x2re, 'filled')
xlim([0, numTimes])
ylim([-6.25, 6.25])
title('Real Part of the Signal x_{2}[n]')
xlabel('Time n')
ylabel('Re\{x_{2}[n]\}')
saveas(gcf, 'prob4x2re_matlab.png')

figure('Position', [150, 150, 900, 400])
stem(n, x2im, 'filled')
xlim([0, numTimes])
ylim([-6.25, 6.25])
```

```
title('Imaginary Part of the Signal x_{2}[n]')
xlabel('Time n')
ylabel('Im\{x_{2}[n]\}')
saveas(gcf, 'prob4x2im_matlab.png')
```

R Python Code for Problem 4

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

#----------------------#
# Signals for Problem 4 #
#----------------------#

N = 34 # period
numPeriods = 2 # number of periods to plot, starting from n = 0
numTimes = N*numPeriods

n = np.arange(0, numPeriods*N + 1) # time vector
phase1 = (np.pi/17.0)*n - np.pi/12.0       #   3/51 =  1/17
phase2 = (69.0*np.pi/17.0)*n - np.pi/12.0  # 207/51 = 69/17

x1re = 6.0*np.cos(phase1) # real part of x1
x1im = 6.0*np.sin(phase1) # imag part of x1
x2re = 6.0*np.cos(phase2) # real part of x2
x2im = 6.0*np.sin(phase2) # imag part of x2

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, x1re)
ax.set_xlim([0, numTimes])
ax.set_ylim([-6.25, 6.25])
ax.set_title('Real Part of the Signal $x_{1}[n]$', fontsize="x-large")
ax.set_xlabel('Time $n$', fontsize="x-large")
ax.set_ylabel('Re{$x_{1}[n]$}', fontsize="x-large")
plt.savefig('prob4x1re_python.png')
plt.show()
```

```
fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, x1im)
ax.set_xlim([0, numTimes])
ax.set_ylim([-6.25, 6.25])
ax.set_title('Imaginary Part of the Signal $x_{1}[n]$', fontsize="x-large")
ax.set_xlabel('Time $n$', fontsize="x-large")
ax.set_ylabel('Im{$x_{1}[n]$}', fontsize="x-large")
plt.savefig('prob4x1im_python.png')
plt.show()

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, x2re)
ax.set_xlim([0, numTimes])
ax.set_ylim([-6.25, 6.25])
ax.set_title('Real Part of the Signal $x_{2}[n]$', fontsize="x-large")
ax.set_xlabel('Time $n$', fontsize="x-large")
ax.set_ylabel('Re{$x_{2}[n]$}', fontsize="x-large")
plt.savefig('prob4x2re_python.png')
plt.show()

fig, ax = plt.subplots(1, figsize=(15,6))
ax.stem(n, x2im)
ax.set_xlim([0, numTimes])
ax.set_ylim([-6.25, 6.25])
ax.set_title('Imaginary Part of the Signal $x_{2}[n]$', fontsize="x-large")
ax.set_xlabel('Time $n$', fontsize="x-large")
ax.set_ylabel('Im{$x_{2}[n]$}', fontsize="x-large")
plt.savefig('prob4x2im_python.png')
plt.show()
```