1. **א.** אם נדגום 100 פעמים את הסיגנל, נשפר את הSNR פי 10.

**ב.** נבצע למעשה החלקה - ניקח חלון זמן מלבני ונמצע את הנקודות בעזרתו.

נסכום את הנקודות הסמוכות ונחלק בגודל החלון. השיפור בSNR יהיה שווה ל-פי שורש גודל החלון, כך שאם נתון כי גודל החלון שלנו הוא 100 נקודות, השיפור יהיה פי 10.

**ג.** החלקה פעמיים עם מלבן בגודל 2 לא משפרת את הSNR מאשר שיפרה את ההחלקה הראשונה באותו הגודל, ושווה להחלקה עם משולש בגודל 3.  
לכן החלקה זו "משפרת" את הSNR בשורש 2. אך ההחלקה פוגעת גם באות ולא רק ברעש, לכן לא נכון לומר את זה.

**2)**

def cross\_correlation(first, second):  
 # zero padding, so both will be the same size  
 if len(first) > len(second):  
 while len(first) > len(second):  
 second.append(0)  
 elif len(first) < len(second):  
 while len(first) < len(second):  
 first.append(0)  
 num\_of\_vals = 2 \* len(second) - 1  
 corr = [0] \* num\_of\_vals  
 index = 0  
 for x in range(num\_of\_vals):  
 val = x - len(second) + 1  
 for i in range(len(first)):  
 if len(second) > (i + val) >= 0:  
 p = first[i] \* second[i + val]  
 corr[index] += p  
 index += 1  
 return corr  
  
vectora = np.array([3, 2, 1])  
vectorb = np.array([6, 5, 4])  
  
# testing that np.correlate and our code produces same result  
our\_xcorr\_arr = np.array(cross\_correlation(vectora, vectorb))  
builtin\_xcorr\_arr = np.correlate([6,5,4], [3,2,1], 'full')  
  
np.testing.assert\_equal(our\_xcorr\_arr, builtin\_xcorr\_arr)

1. חישוב ידני:

0 0 6 5 4 0 0

3 2 1 = 6

0 0 6 5 4 0 0

3 2 1 = 12+5=17

0 0 6 5 4 0 0

3 2 1 = 18+10+4=32

0 0 6 5 4 0 0

3 2 1 = 15+8=23

0 0 6 5 4 0 0

3 2 1 = 12

f\*g=[6, 17, 32, 23, 12]

**ב)**

0 0 3 2 1 0 0

6 5 4 = 12

0 0 3 2 1 0 0

6 5 4 = 15+8=23

0 0 3 2 1 0 0

6 5 4 = 18+10+4=32

0 0 3 2 1 0 0

6 5 4 = 12+5=17

0 0 3 2 1 0 0

6 5 4 = 6

[6, 17, 32, 23, 12] מכאן רואים כי f\*g =\=g\*f

על מנת לקבל וקטור זהה עלינו להפוך את כיוון הקריאה, כי מקבלים וקטור בעל מספרים זהים בכיוון ההופכי.

**ג)** מספר ההכפלות שיש לבצע הוא כפל בין אורך 2 הוקטורים-וקטור החלון ווקטור האות(במקרה שלנו 3\*3 = 9 הכפלות).

**3) א)**

![Chart

Description automatically generated]()

ניתן להבחין בצורה מאוד גסה שיש 20 פיקים ב2 שניות, כלומר התידרות היא 10 הרץ. קביעה כזאת כן דורשת הבנה מקדימה של מה אנו מצפים לראות, כך שללא כיהרות מקדימה עם צורת הסיגנל (sin(20\*pi…) לא היינו יכולים לקבוע את התדירות בצורה מדויקת

**ב)**

![Chart

Description automatically generated]()

# plotting w/noise  
t = np.arange(0, 2, 0.01)  
x = np.sin(t\*(20 \* np.pi)) + np.random.rand(len(t))  
  
fig = plt.figure()  
plt.xlabel('Time (Seconds)')  
plt.ylabel('Amplitude')  
plt.title('sin(20pi\*t + rand(1, length(t))')  
plt.plot(x[:len(t)])  
plt.show()  
  
# plotting with smoothing using rec. window  
q = 0.2 \* np.ones(5)  
z = cross\_correlation(x.tolist(), q.tolist())  
# print (len(z))  
# z = np.correlate(x, q)  
fig = plt.figure()  
plt.xlabel('Time (Seconds)')  
plt.ylabel('Amplitude')  
plt.title('sin(20pi\*t + rand(1, length(t)) smoothed by rectangular window (N=5)')  
plt.plot(z[:len(t)])  
plt.show()

**4)**

**א)** התשובה היא [1, -1] משום שקונבולוציה מוגדרת כך: Text

Description automatically generated with medium confidence

לעומת קרוס קורלציה שמוגדרת כך

![Text, letter

Description automatically generated]()

לכן נצטרך להפוך את הסימן של המינוס בשני הערכים בוקטור כדי לקבל את אותה התוצאה (נגזרת ראשונה).

**ב)** כדי לקבל נגזרת שניה נצטרך לבצע קונבולוציה על [1, -1] פעמיים, כלומר:

f’ = f\*[1, -1]

f’’ = ( f’)’ = f’ \* [-1,1] = (f\*[1,-1] ) \* [-1,1]

לפי תכונת אסוציאטיביות של קונבולוציה התשובה היא [1,-1]\*[1-1] = [1,-2 ,1]

**ג)** כדי להחליק את האות נבצע קרוס קורלציה:

CC(f, [0.5, 0.5])

כדי לגזור אותו נבצע לתוצאה קונבולוציה עם [1, -1]:

CC(f, [0.5, 0.5]) \* [1, -1] = f \* [-0.5, -0.5] \* [1, -1] = f \* [-0.5, 0, 0.5]

כלומר הוקטור שאיתו נעשה קונבולוציה פעם אחת כדי לקבל את מה שנדרש בסעיף ג׳ הוא [-0.5, 0, 0.5].

**ד)** נצטרך לעשות קונבולוציה עם הוקטור [1-1-,1,1] כדי שבמכפלה בקונבלוציה (שנעשית עם הוקטור במהופך) נקבל סכום של 2 איברים עוקבים ראשונים פחות סכום איברים עוקבים שבאים 2 אחריהם.

**5)**

כדי למצוא הפרש פאזה בין שני סינוסואידים נבצע CC (אוטו-קורלציה) בין שני האותות. הערך הגבוה המקסימלי שיתקבל בוקטור הוא הפרש הזמנים בין האותות. אם מצאנו את הפרש הזמנים ניתן למצוא את הפאזה ע"י שימוש בנוסחא wt =∅ כאשר נתון w = 1Hz וt הוא הפרש הזמנים שקיבלנו בCC ונקבל את הזווית ברדיאנים.

**6)**

אנו מנחשים כי החלק מהשיר (ySnip) שייך לקובץ שיר yHurt.

בהרצה של CC מתקבל הערך הגדול ביותר כאשר עושים CC עם השיר yHurt (y=1381), ערך קטן יותר עם yHeathens (y=150), והערך הקטן ביותר היה עם ySilence (y=136). ככה שצדקנו ואכן הסניפט נלקח מהשיר Hurt.

לגבי המיקום בשיר- נבדוק לפי הערך של הפיק ונחלק בfs -1103000 / 44100 ונקבל 25 כלומר אחרי כ-25 שניות מתחילת ניגון הקובץ של השיר.