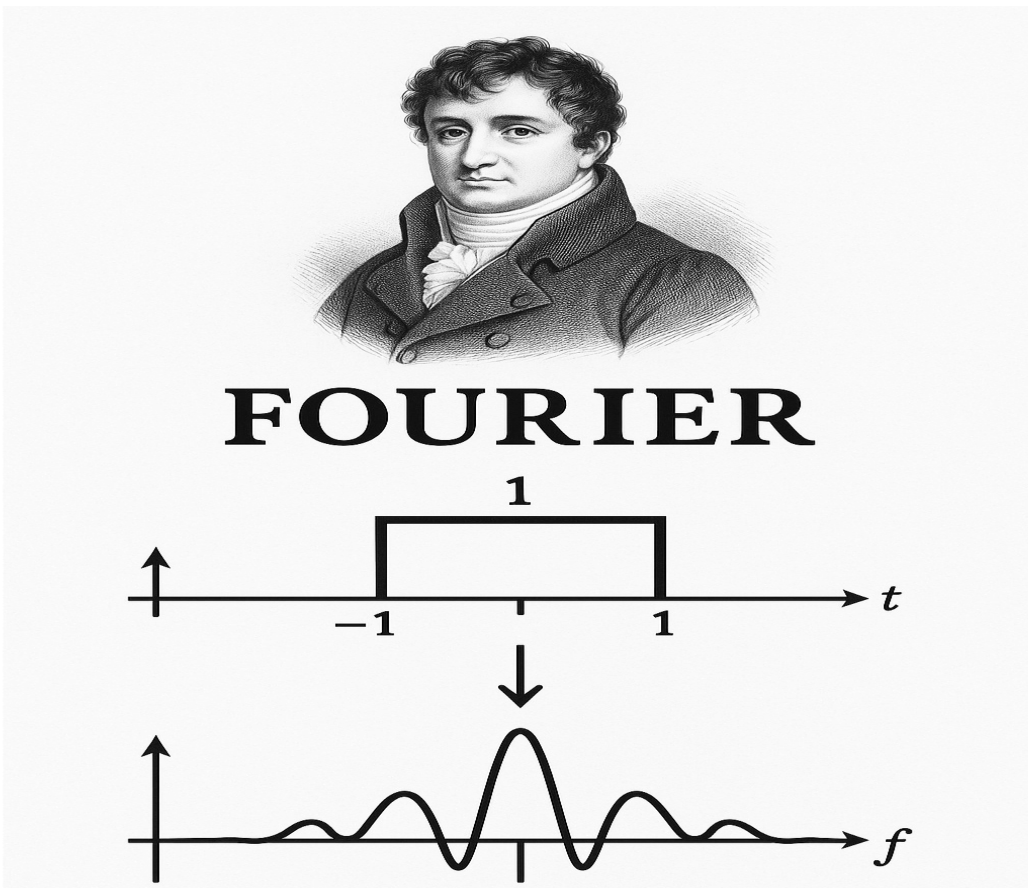
אותות ומערכות – תרגיל מחשב 1



מגישה: הדס רם
ת"ז: 214068843
תאריך הגשה: 03.05.2025
מרצה: ד"ר שוורץ עופר

בס"ד

סעיף א:

נתון אות חלון הבא :

$$a(n) = \begin{cases} 1 & |n| < 100 \\ 0 & \text{אחרת} \end{cases}$$

א. קבעו וקטור זמנים, $n = -1000:1:1000$, ציירו את האות.

קוד:

```python
import numpy as np
import cmath
import math
import matplotlib.pyplot as plt

plt.close('all')

#a

# vector times
n = np.arange(-1000, 1001)

# The signal a(n)
a_n = np.where(np.abs(n) < 100, 1, 0)

# drawing the graph
plt.figure(figsize=(10,4))
plt.plot( *args: n, a_n)
plt.title("The Window signal |n| < 100")
plt.xlabel("n")
plt.xticks(np.arange(min(n), max(n)+1, 100))
plt.ylabel("a(n)")
plt.grid(True)
plt.show()


##################################################
```

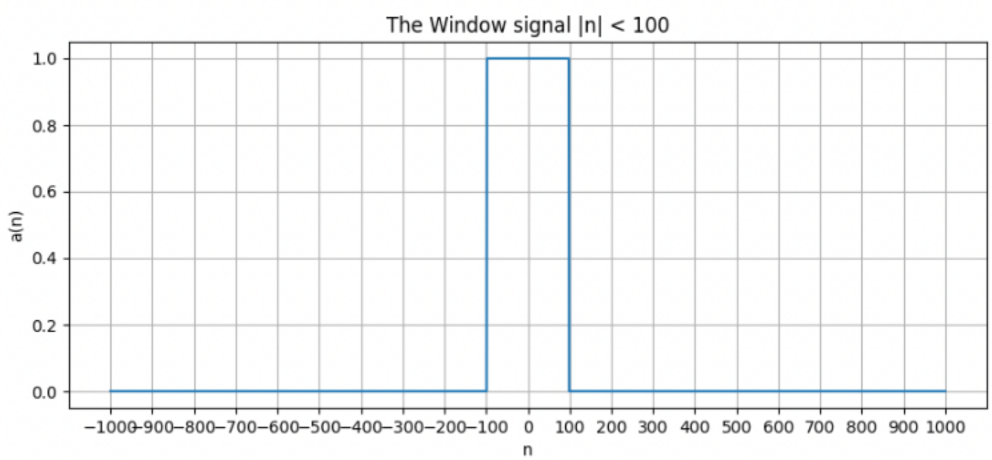
יצרתי וקטור של אינדקסים שלמים בטווח [-1000, 1000], סה"כ 2001 נקודות בסך הכל.

הגדרתי את האות a(n) כאות חלון:

$$a(n) = \begin{cases} 1 & |n| < 100 \\ 0 & \text{אחרת} \end{cases}$$

הדפסתי בגרף את האות a(n).

גרף:



סעיף ב:

ב. התייחסו אל האות בסעיף א' כאל מחזור יחיד של אות מחזורי במחזור: $N = 2001$.

עתה נרצה להציג את מקדמי טור פורייה של האות

$$a_k = \frac{1}{N}\sum_{n=<N>} x(n)e^{-jk2\pi n/N}$$

הציגו את מקדמי פורייה של האות . כיון ש $a(n)$ הוא ממשי וסימטרי בזמן הראו כי הוא a_k ממשי וסימטרי בתדר.

קוד:

```python
#b

# Parameters
N = 2001

# Compute Fourier coefficients a(k)
k_values = np.arange(-1000, 1001)
a_k = np.array([np.sum(a_n * np.exp(-1j * 2 * np.pi * k * n / N)) for k in k_values])

# Normalize Fourier coefficients by N
a_k = a_k / N

# Check if imaginary part is negligible
max_imag_part = np.max(np.abs(np.imag(a_k)))

# Remove negligible imaginary parts manually using a for loop
a_k_ = []

for val in a_k:
    if abs(val.imag) < 1e-10:
        a_k_.append(val.real)  # keep only the real part
    else:
        a_k_.append(val)  # keep complex value

# Convert list back to array
a_k = np.array(a_k_)
```

הגדתי את הערכים הבאים: N, k_values.
הגדרתי את a_k לפי מקדמי פוריה של האות a(n).

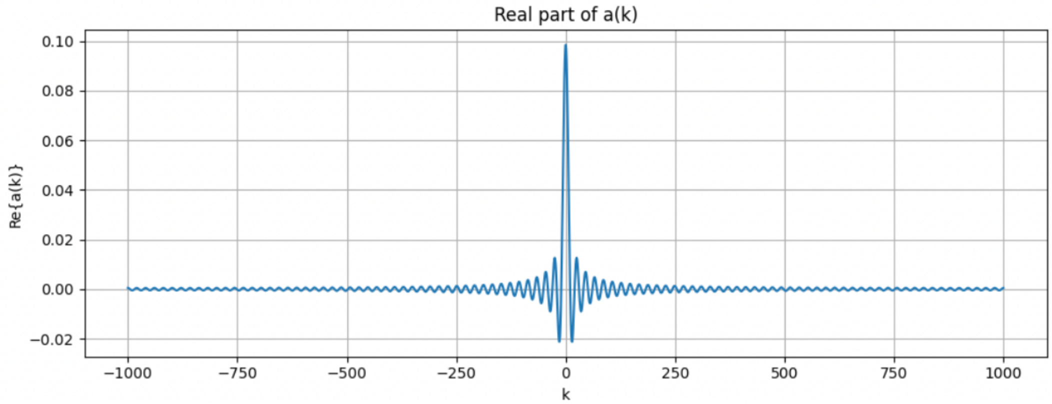
$$a_k = \frac{1}{N}\sum_{n=<N>} x(n)e^{-\frac{jk2\pi n}{N}}$$

בדקתי שהחלק המדומה זניח והשארתי רק את החלק הממשי במקדמים אם הוא אכן זניח. ושמרתי במערך a_k את החלק הממשי של מקדמי פוריה.

```python
# Plot real part
plt.figure(figsize=(10, 4))
plt.plot(k_values, np.real(a_k))
plt.title("Real part of a(k)")
plt.xlabel("k")
plt.ylabel("Re{a(k)}")
plt.grid(True)
plt.tight_layout()
plt.show()

# Plot imaginary part
plt.figure(figsize=(10, 4))
plt.plot(k_values, np.imag(a_k))
plt.title("Imaginary part of a(k)")
plt.xlabel("k")
plt.ylabel("Im{a(k)}")
plt.grid(True)
plt.tight_layout()
plt.show()

# Plot Phase part
plt.figure(figsize=(10, 4))
plt.stem(k_values, np.angle(a_k, deg=True))
plt.title("Phase part of a(k)")
plt.xlabel("k")
plt.ylabel("angle{a(k)}")
plt.yticks([-180, -135, -90, -45, 0, 45, 90, 135, 180])
plt.grid(True)
plt.tight_layout()
plt.show()
```

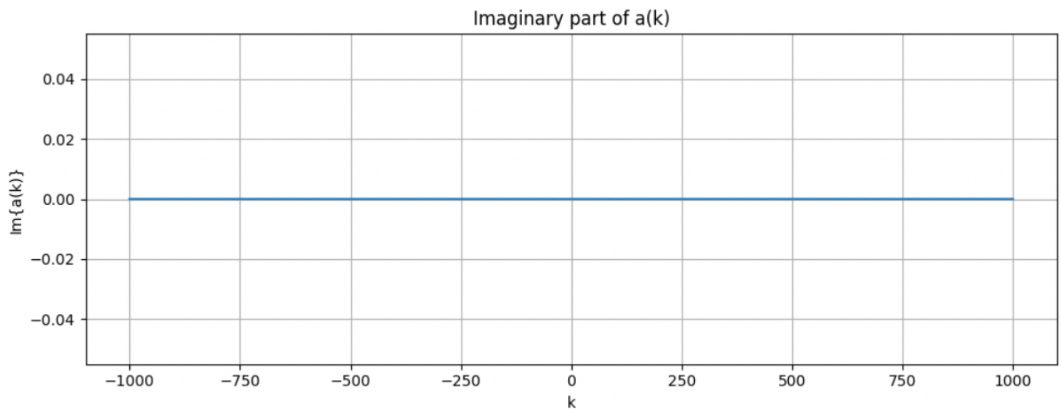
הדפסתי גרפים של החלק הממשי והמדומה של a(k).
וגרף של הפאזה של a(k).

הגרפים:
החלק הממשי:



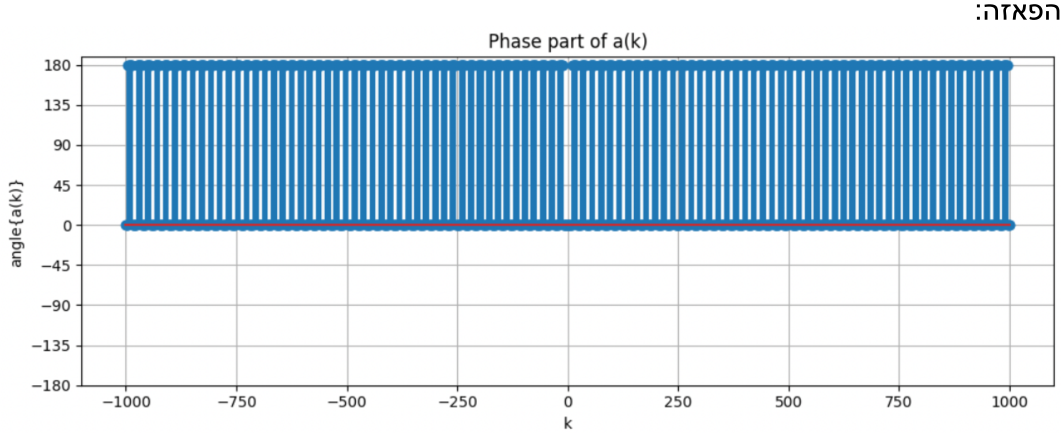
ניתן לראות כי החלק הממשי של a(k) קיים.

החלק המדומה:



ניתן לראות כי החלק המדומה של a(k) זניח ולכן יתאפס כולו.

הפאזה:



ניתן לראות שהפאזה היא 0 משמע מספר ממשי חיובי, או 180 מעלות משמע מספר ממשי שלילי.

המשך קוד:

```
# Check if a_k is real
if np.allclose(np.imag(a_k), b: 0, atol=1e-10):
    print("All Fourier coefficients a_k are real.")
else:
    print("Fourier coefficients a_k are not real.")

# Check if a_k is symmetric

# Find the center index
center_idx = len(a_k) // 2

# Assume symmetric unless found otherwise
is_symmetric = True
for i in range(len(a_k) // 2):
    left = a_k[i]
    right = a_k[-(i+1)]
    if not np.allclose(left, right, atol=1e-10):
        print(f"Mismatch at index {i}: {left} vs {right}")
        is_symmetric = False
        break

if is_symmetric:
    print("The Fourier coefficients a_k are symmetric.")
else:
    print("The Fourier coefficients a_k are NOT symmetric.")


##############################################################
```

בדיקה האם a(k) הוא סימטרי וממשי בתדר.
ביצעתי בדיקה האם החלק המדומה זניח והדפסת הפלט בהתאם.
ביצעתי בדיקה האם a(k) הוא סימטרי בעזרת לולאה שרצה מהאמצע של האות וכל פעם משווה בין צעד ימינה למעד שמאלה מאמצע האות. הדפסת הפלט המתאים מודפס.

הפלט:

```
All Fourier coefficients a_k are real.
The Fourier coefficients a_k are symmetric.
```

הראנו שכאשר a(n) ממשי וסימטרי בזמן אזי a(k) ממשי וסימטרי בתדר.

סעיף ג:

ג. נרצה עתה לבחון את הזהות של הזזה בזמן ⇔ הכפלה באקפוננט בתדר

חשב את : $b_k = a_k e^{-jk2\pi 100/N}$

חשבו והציגו את האות בזמן $b(n)$ ע"י ההתמרה ההופכית של b_k ואשרו את הזהות הנ"ל.

- **הסבר את ההבדל בין הגרפים a_k, b_k.**
- **שרטט את גרף הפאזה של a_k, b_k.**
- **צייר את המקדם b_{10} במרחב הזמן. מהו הביטוי המתמטי?**

קוד:

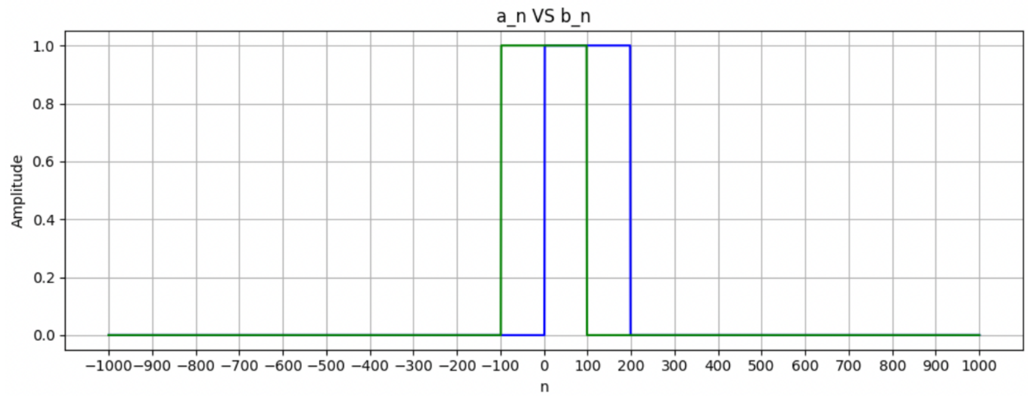
```python
#c

shift = 100

# Compute b_k
b_k = []  # Initialize list to hold b_k values

for i, k in enumerate(k_values):
    exponential_factor = np.exp(-1j * 2 * np.pi * k * shift / N)
    b_k.append(a_k[i] * exponential_factor)

# Convert to array
b_k = np.array(b_k)

# Reconstruct b(n) from b_k using the inverse Fourier series
b_n = []  # List to store reconstructed time-domain signal

for n_ in n:
    sum_val = 0
    for i, k in enumerate(k_values):
        sum_val += b_k[i] * np.exp(1j * 2 * np.pi * k * n_ / N)
    b_n.append(sum_val.real)  # Only real part (imaginary should be negligible)

# Convert to array
b_n = np.array(b_n)
```

חישבתי את b(k) והכנסתי את כל הערכים למערך.
עשיתי התמרת פורייה הפוכה בעזרת לולאה על b(k) וקיבלתי את b(n)ושמרתי את הערכים שלו במערך.

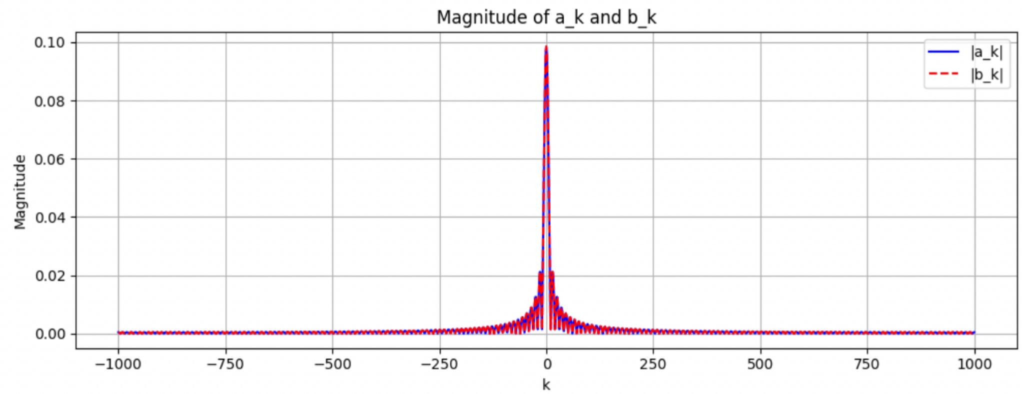
```python
# Plot the reconstructed signal
plt.figure(figsize=(10, 4))
plt.plot(n, b_n, color = 'blue', label='b(n)')
plt.plot(n, a_n, color = 'green', label='a(n)')
plt.title("a_n VS b_n")
plt.xlabel("n")
plt.xticks(np.arange(min(n), max(n)+1, 100))
plt.ylabel("Amplitude")
plt.grid(True)
plt.tight_layout()
plt.show()


# Plot Magnitude of a_k and b_k
plt.figure(figsize=(10, 4))
plt.plot(k_values, np.abs(a_k), label='|a_k|', color='blue')
plt.plot(k_values, np.abs(b_k), '--', label='|b_k|', color='red')
plt.title("Magnitude of a_k and b_k")
plt.xlabel("k")
plt.ylabel("Magnitude")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

הדפסת הגרף של האותות a(n) וb(n) (שיתקבל מהתמרת פורייה ההפוכה).
הדפסת גרף של המגניטודה של a(k) וb(k).

הגרפים:



ניתן לראות כי גרף b(n) הוא תזוזה ב100 יחידות של גרף a(n) בזמן.



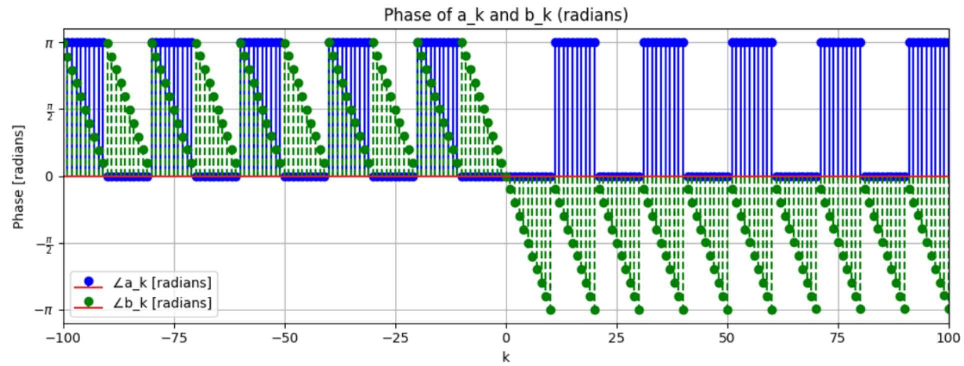
ניתן לראות כי המגניטודה בתדר של a(k) וb(k) היא אותה המגניטודה כי שציפיתי לקבל כיוון שתזוזה בזמן לא אמורה להשפיע על המגניטודה בתדר.

המשך הקוד:

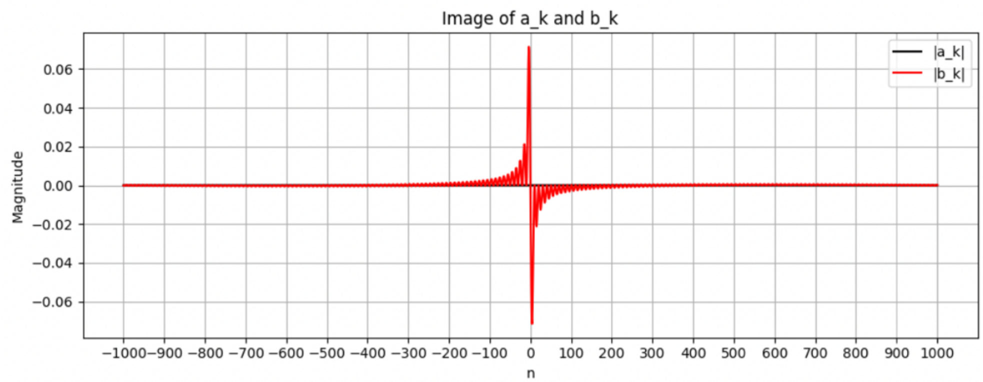
הכנסתי את הערכים של הפאזה של כל מקדם למערך.
והדפסתי את גרף הפאזה של a(k) וb(k).
בנוסף הדפסתי גרף של החלקים המדומים של a(k) וb(k).

```python
a_k_phase = [cmath.phase(x) for x in a_k]
b_k_phase = [cmath.phase(x) for x in b_k]

a_k_phase = np.array(a_k_phase)
b_k_phase = np.array(b_k_phase)

# Plot Phase of a_k and b_k
plt.figure(figsize=(10, 4))
plt.stem(k_values, a_k_phase, linefmt='b-', label='∠a_k [radians]')
plt.stem(k_values, b_k_phase, linefmt='g--', label='∠b_k [radians]')
plt.title("Phase of a_k and b_k (radians)")
plt.xlabel("k")
plt.ylabel("Phase [radians]")
plt.legend(loc='best')
plt.grid(True)
plt.tight_layout()
plt.xlim(-100, 100)
plt.yticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
           [r"$-\pi$", r"$-\frac{\pi}{2}$",r"$0$", r"$\frac{\pi}{2}$", r"$\pi$"])
plt.show()

#a_k and b_k image part
plt.figure(figsize=(10, 4))
plt.plot(n, a_k.imag, label='|a_k|', color='black')
plt.plot(n, b_k.imag, label='|b_k|', color='red')
plt.title("Image of a_k and b_k")
plt.xlabel("n")
plt.xticks(np.arange(min(k_values), max(k_values)+1, 100))
plt.ylabel("Magnitude")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

הגרפים:



ניתן לראות כי הפאזה של הגרף b(k) שונה מהפאזה של a(k).



ניתן לראות כי החלק המדומה של b(k) שונה מהחלק המדומה a(k) (שהוא 0 כיוון שהוא ממשי)

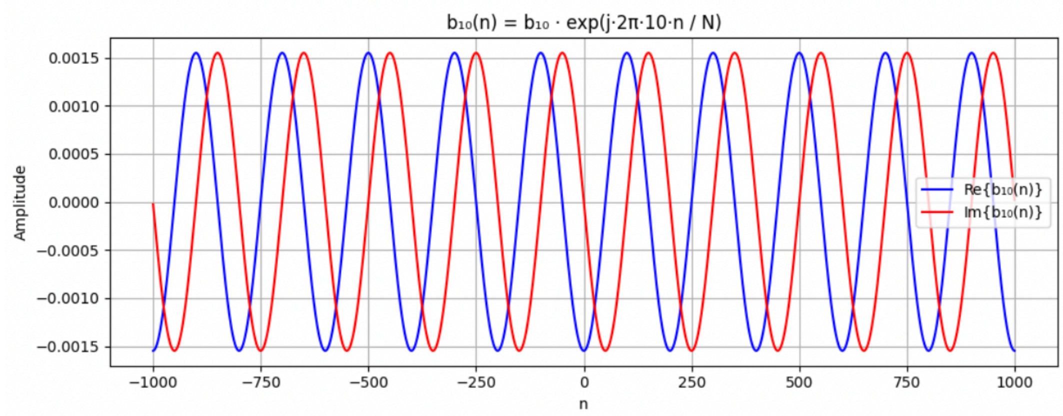
לכן ניתן לראות כי הזהות אכן מתקיימת - הזזה בזמן זה כפל באקספוננט בתדר כלומר פאזה שונה.

קוד עבור המקדם b_{10}:

```python

# Calculating b10
b10 = b_k[10 + N//2]

# Calculate b_10(n)
k_index = 10 + N // 2  # because k_values starts at -N//2
b10 = b_k[k_index]
b10_n = b10 * np.exp(1j * 2 * np.pi * 10 * n / N)

# Plot the real and imaginary parts of b_10(n)
plt.figure(figsize=(10, 4))
plt.plot(n, np.real(b10_n), label='Re{b₁₀(n)}', color='blue')
plt.plot(n, np.imag(b10_n), label='Im{b₁₀(n)}', color='red')
plt.title("b₁₀(n) = b₁₀ · exp(j·2π·10·n / N)")
plt.xlabel("n")
plt.ylabel("Amplitude")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()



#########################################################
```

חישבתי את המקדם b_{10} במרחב התדר, מכיוון שk_values מוגדר בין N/2- ל N/2, האידקס המתאים לk=10 הוא 10+N/2.
בעזרת התמרת פורייה חישבתי את b_{10} במרחב הזמן והדפסתי את הגרף שלו.

הגרף של b_{10}:



הביטוי המתמטי של b_{10}:

המקדמים בתחום התדר:

$$b_k = a_k e^{-j\frac{2\pi}{N}kn}$$

המרה לתחום הזמן:

$$b_n = b_k * e^{\frac{j2\pi kn}{N}}$$

המקדם b_{10} בזמן:

$$b_{10}(n) = b_{10} * e^{\frac{j2\pi 10n}{N}}$$

העברה לאמפליטודה ופאזה:

$$b_{10}(n) = |b_{10}| * e\text{^}(j\left(\frac{2\pi 10n}{N} + \angle b_{10}\right))$$

סעיף ד:

ד. נרצה עתה לבחון את הזהות של גזירה בזמן ⇔ "הכפלה ב k " בתדר.

חשבו את : $c_k = a_k\left(1 - e^{-jk2\pi/N}\right)$

חשבו והציגו את האות בזמן $c(n)$ ע"י ההתמרה ההופכית של c_k . האם קבלתם גזירה בזמן?

אשרו זאת אנליטית.

הקוד:

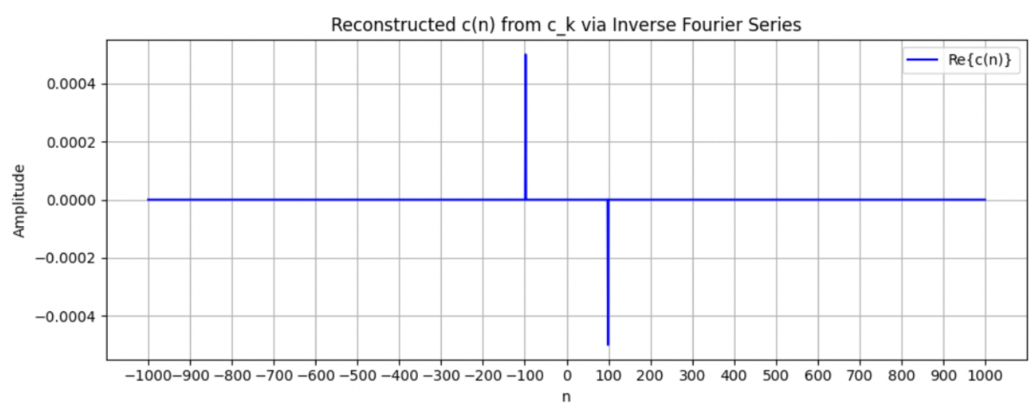
```
#d

# Calculate c_k according to the given formula
c_k = a_k * (1 - np.exp(-1j * 2 * np.pi * k_values / N))

# Reconstruct c(n) from c_k using the inverse Fourier series
c_n_reconstructed = np.zeros_like(n, dtype=complex)

for i, n_ in enumerate(n):
    c_n_reconstructed[i] = np.sum(c_k * np.exp(1j * 2 * np.pi * k_values * n_ / N)) / N


plt.figure(figsize=(10, 4))
plt.plot( *args: n, np.real(c_n_reconstructed), label="Re{c(n)}", color='blue')
plt.title("Reconstructed c(n) from c_k via Inverse Fourier Series")
plt.xlabel("n")
plt.xticks(np.arange(min(n), max(n)+1, 100))
plt.ylabel("Amplitude")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()


################################################
```

חישבתי את c(k) כך:

$$c_k = a_k\left(1 - e^{-jk2\pi/N}\right)$$

לאחר מכן עשיתי התמרת פורייה הפוכה בלולאה וקיבלתי את c(n).

הגרף:



ניתן לראות כי קיבלתי גזירה של האות a(n), כל הערכים הם אפסים חוץ מ:
n=-100 , n=100
כפי שציפינו לקבל מגזירה של אות חלון.

אנליטית:

a: בתדר - $a\_k = (1/N) * \sum a(n) * e^{-j\cdot 2\pi kn/N}$, התמרה הפוכה - $a(n) = \sum a\_k * e^{j\cdot 2\pi kn/N}$.

c: בתדר - $c\_k = a\_k * (1 - e^{-j\cdot 2\pi k/N})$, התמרה הפוכה - $c(n) = \sum c\_k * e^{j\cdot 2\pi kn/N}$.

$$c(n) = \sum a\_k * (1 - e^{-j\cdot\frac{2\pi kn}{N}}) * e^{j\cdot\frac{2\pi kn}{N}} = \Sigma a\_k * e^{j\cdot\frac{2\pi kn}{N}} - \Sigma a\_k * e^{(j\cdot 2\pi k(n-1)/N)} = a(n) - a(n-1)$$

קיבלנו שכאשר כופלים את מקדמי פורייה a(k) בפונקציה $(1 - e^{-j\cdot 2\pi k/N})$, יתקבל אות שהוא $a(n) - a(n-1)$, שזה בדיוק הנגזרת הבדידה בזמן.

<u>סעיף ה:</u>

ה. נרצה עתה לבחון את הזהות של קונבלוציה בזמן ⇔ הכפלה בתדר.

חשבו את : $d_k = Na_k^2$

חשבו והציגו את האות בזמן $d(n)$ שהינו ההתמרה ההופכית של d_k . האם קבלתם קונבלוציה בזמן? אשרו זאת אנליטית.

חשב את $\max\{d_n\}$ והסבר את התוצאה.

קוד:

```python
#e

# Compute d_k = N * (a_k)^2
d_k = N * a_k**2

# Reconstruct d(n) using inverse Fourier series
d_n = np.zeros_like(n, dtype=complex)
for i, n_ in enumerate(n):
    d_n[i] = np.sum(d_k * np.exp(1j * 2 * np.pi * k_values * n_ / N))

# Plot real part of d_k
plt.figure(figsize=(10, 4))
plt.plot( *args: k_values, np.real(d_k), label="Re{d_k}", color='blue')
plt.title("d_k = N * (a_k)^2 (Real Part)")
plt.xlabel("k")
plt.ylabel("Amplitude")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()

# Plot Re{d(n)}
plt.figure(figsize=(10, 4))
plt.plot( *args: n, np.real(d_n), label="Re{d(n)}", color='blue')
plt.title("Reconstructed d(n) from d_k")
plt.xlabel("n")
plt.ylabel("Amplitude")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
```

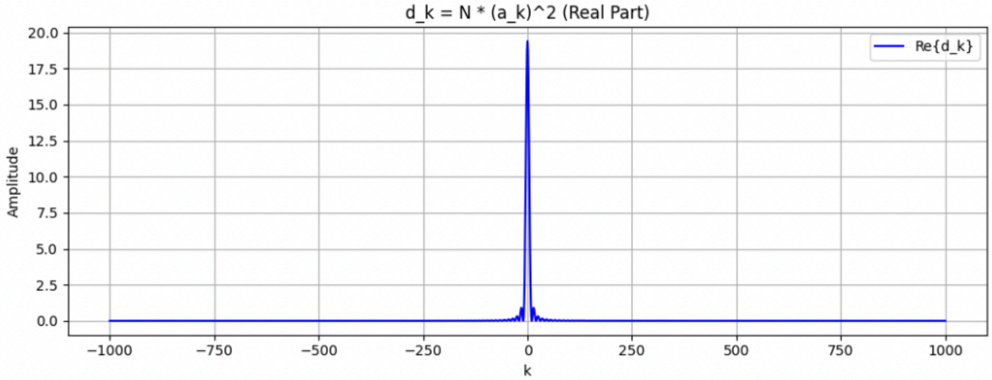
הגדרתי את d(k) כך:

$$d_k = Na_k^2$$

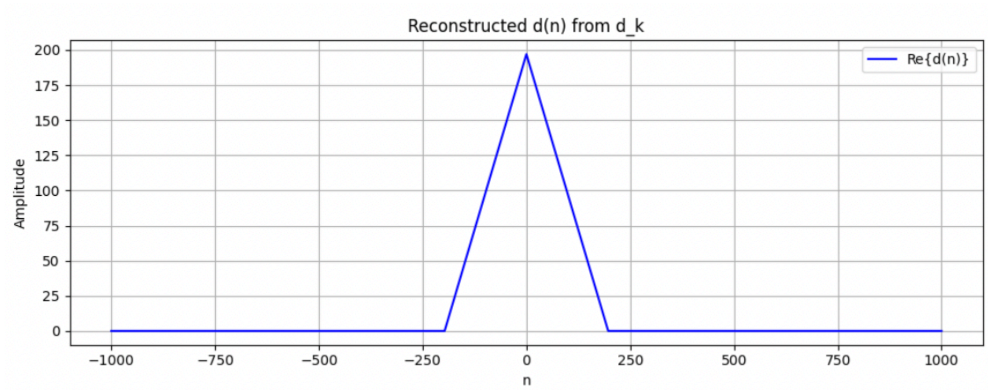
עשיתי התמרת פורייה הפוכה בלולאה ומצאתי את d(n).
לאחר מכן הדפסתי את הגרף של החלק הממשי של d(k) ואת הגרף של החלק הממשי של d(n).

הגרפים:

החלק הממשי של d(k):



החלק הממשי של d(n):



ניתן לראות כי קיבלנו משולש שזו תוצאה של קונבולציה של שני אותות חלון.

המשך קוד:

```python
# Compute direct linear convolution of a(n) with itself
conv_direct = np.convolve(a_n, a_n, mode='full')

# Create time vector for full convolution result
n_conv = np.arange(2 * n[0], 2 * n[-1] + 1)  # This will go from -2000 to +2000

# Center the result to match the original time vector n = [-1000, 1000]
start_idx = (len(conv_direct) - len(n)) // 2
conv_centered = conv_direct[start_idx : start_idx + len(n)]

# --- Loop to check if values are equal ---
tolerance = 1e-10
all_equal = True

for i in range(len(n)):
    diff = np.abs(np.real(d_n[i]) - conv_centered[i])
    if diff > tolerance:
        print(f"Mismatch at n = {n[i]}: d_freq = {np.real(d_n[i])}, d_time = {conv_centered[i]}")
        all_equal = False
        break

if all_equal:
    print("Values match ⇒ convolution in time domain.")

# Max d(n)
max_dn = np.max(np.real(d_n))
print(f"max{{d(n)}} = {max_dn}")


########################################
```

עשיתי קונבולוציה בזמן של האות a(n) עם עצמו ובעזרת לולאה השוותי את הערכים שהתקבלנו מהקונבולוציה בזמן לעומת הערכים שיש בd(n).
קיבלנו כי כל הערכים שווים ולכן אכן קיבלנו קונבולוציה בזמן.

בנוסף לכן חישבתי את הערך המקסימלי של d(n).

פלט:

```
Values match ⇒ convolution in time domain.
max{d(n)} = 199.0
```

בקונבולציה של אות מלבן עם עצמו יתקבל הערך המקסימלי כאשר תהיה חפיפה מלאה בין המלבנים.
המלבנים אצלנו הם מ-99- עד 99 (האמצע הוא באינדקס 0). לכן נקבל חפיפה מלאה כאשר:

$$d(0) = \sum_{-99}^{99} |a_n|^2$$

כאשר כל $a_n = 1$ ולכן נקבל 199.

נראה אנליטית כי מתקבלת קונבולוציה בזמן:

מהתמרת פוריה מתקבל: $a_n \Longrightarrow a_k$, $\overline{a}_n \Longrightarrow \overline{a}_k$.

בנוסף לכך ידוע כי: $a_k\overline{a_k} = |a_k|^2$, $\overline{a}_k \Longrightarrow \overline{a_{-n}}$, $|a_k|^2 \Longrightarrow a_n * \overline{a_{-n}}$.

כיוון ש a(n) אצלנו ממשי וסימטרי ולכן נקבל כי: $a_n = \overline{a_{-n}}$.

לכן נקבל מהתמרת פורייה הפוכה: $|a_k|^2 \Longrightarrow a_n * a_n$.

אצלנו נתון כי: $d_k = N|a_k|^2$ לכן בהתמרת פורייה הפוכה נקבל $d(n) = a_n * a_n$ (גורם הנירמול N נמצא בתוך המקדמים).

הראנו כי כפל בתדר שווה לקונבולוציה בזמן וכיוון ש(a(n ממשי וסימטרי קיבלנו קונבולוציה של האות עם עצמו.

בס"ד

סעיף ו:

ו. נרצה עתה לבחון את שוויון פרסיבל.

חשבו בנפרד את : $\frac{1}{N}\sum_{n=<N>}|d(n)|^2$

ואת : $\sum_{k=<N>}|d_k|^2$

והראו כי מתקיים שוויון.

קוד:

```python
#f

# parseval in time
parseval_time = np.sum(np.abs(d_n)**2) / N

# parseval in frequency
parseval_freq = np.sum(np.abs(d_k)**2)

# Print results
print(f"Parseval in time domain:      {parseval_time:.6f}")
print(f"Parseval in frequency domain: {parseval_freq:.6f}")

# Check if they match
if np.allclose(parseval_time, parseval_freq, atol=1e-10):
    print("Parseval's equality holds.")
else:
    print("Parseval's equality does NOT hold.")


####################################################
```

חישבתי את פרסבל בצד של הזמן, כך:

$\frac{1}{N}\sum_{n=<N>}|d(n)|^2$

ואת פרסבל בצד של התדר, כך:

$\sum_{k=<N>}|d_k|^2$

הדפסתי את הערכים ובעזרת לולאה בדקתי האם הם שווים.

פלט:

```
Parseval in time domain:      2625.586707
Parseval in frequency domain: 2625.586707
Parseval's equality holds.
```

ניתן לראות כי שיוון פרסבל מתקיים.

<u>סעיף ז:</u>

ז. נרצה עתה לבחון את הזהות הכפלה בזמן ⇔ קונבלוציה בחדר.

חשבו את : $e(n) = a(n)b(n)$

חשבו והציגו את האות בתדר e_k ע"י ההתמרה של $e(n)$. חשבו והציגו גם את e_k המתקבל ע"י קונבלוציה ציקלית בתדר דהיינו $\hat{e}_k = \sum_{l=<N>} a_l b_{k-l}$. אשרו כי האות e_k זהה ל $\hat{e}_k$.

קוד:

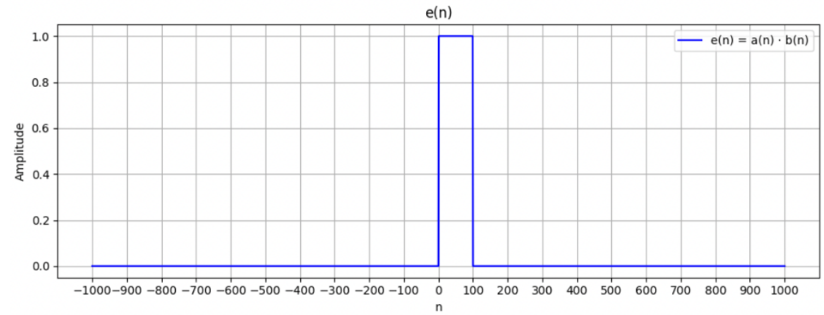
```python
#g

# --- e(n) = a(n) * b(n) ---
e_n = a_n * b_n

# Compute Fourier coefficients of e(n)
e_k = np.array([np.sum(e_n * np.exp(-1j * 2 * np.pi * k * n / N)) for k in k_values])
e_k /= N  # Normalize


# --- Plot e(n) ---
plt.figure(figsize=(10, 4))
plt.plot(n, e_n, label="e(n) = a(n) · b(n)", color='blue')
plt.title("e(n)")
plt.xlabel("n")
plt.xticks(np.arange(min(k_values), max(k_values)+1, 100))
plt.ylabel("Amplitude")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()


# --- Plot Magnitude of e_k ---
plt.figure(figsize=(10, 4))
plt.plot(k_values, np.abs(e_k), label="|e_k|", color='green')
plt.title("Magnitude of e_k")
plt.xlabel("k")
plt.ylabel("|e_k|")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()


# --- Plot Phase of e_k ---
plt.figure(figsize=(10, 4))
plt.plot(k_values, np.angle(e_k), label= "∠e_k", color='blue')
plt.title("Phase of e_k")
plt.xlabel("k")
plt.ylabel("Phase [radians]")
plt.yticks([-2 * np.pi, -np.pi, 0, np.pi, 2 * np.pi],
           [r"$-2\pi$", r"$-\pi$", r"$0$", r"$\pi$", r"$2\pi$"])
plt.grid(True)
plt.legend()
plt.xlim(-100, 100)
plt.tight_layout()
plt.show()
```

הגדרתי את e(n) כך:

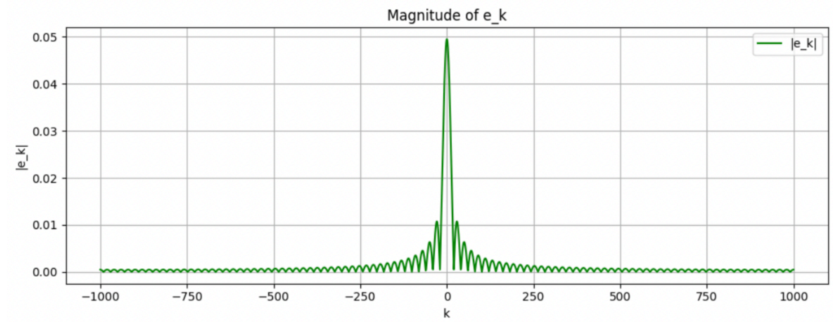
$$e(n) = a(n)b(n)$$

עשיתי התמרת פורייה וקיבלתי את e(k).
לאחר מכן הדפסתי את הגרף של e(n),
את גרף המגניטודה של e(k) ואת גרף
הפאזה של e(k).

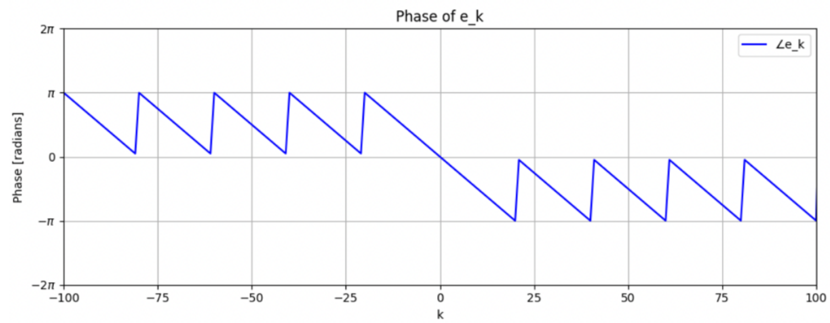
הגרפים:



האות e(n) מתקבל כתוצאה מהכפלה בין שני חלונות – האחד מקורי והשני מוזז. לכן, האות החדש הוא חלון מוזז, בהתאם למידת החפיפה בין a(n) ל- b(n).



כיוון ש b(k) שווה ל a(k) כפול פאזה לינארית (ואתה המגניטודה), מתקבלת קונבולציה של האות עם עצמו. לכן מתקבל צורת פעמון סביב k=0, דבר שמאפיין קונבולציה של אות עם עצמו.



הגרף מציג פאזה לינארית יורדת – תוצאה ישירה של הזזה בזמן. קונבולוציה של אות עם גרסה מוזזת שלו יוצרת פאזה כמו של מסור המשקפת את ההזזה.

המשך קוד:

```python
# Circular convolution of a_k and b_k

# Manual FFT shift and IFFT shift
def manual_fftshift(arr):
    N = len(arr)
    return np.concatenate((arr[N//2:], arr[:N//2]))

def manual_ifftshift(arr):
    N = len(arr)
    return np.concatenate((arr[- N//2:], arr[:- N//2]))

# Align a_k and b_k to regular index order (0 to N-1)
a_k_shifted = manual_ifftshift(a_k)
b_k_shifted = manual_ifftshift(b_k)

# Compute circular convolution ê_k
e_k_hat = np.zeros(N, dtype=complex)
for k in range(N):
    for l in range(N):
        e_k_hat[k] += a_k_shifted[l] * b_k_shifted[(k - l) % N]

# Shift ê_k back to centered indexing (to match k_values)
e_k_hat = manual_fftshift(e_k_hat)

# Compare ê_k to e_k
if np.allclose(e_k, e_k_hat, atol=1e-2):
    print("ê_k match e_k")
else:
    print("ê_k mismatch e_k")
```

הגדרתי 2 פונקציות:
manual_ifftshift(arr) – מעבירה את האיברים כך שה-0 יהיה בהתחלה (הכנה לפני הקונבולוציה).
manual_fftshift(arr) – מחזירה את התוצאה לסידור הסימטרי כאשר $k \in [-N/2, N/2]$, כדי שאוכל להשוות את e(k) מההתמרה של e(n) והקונבולוציה הציקלית.

הזזתי את a(k) וb(n) שיתחיל מה-0 ועשיתי את הקונבולוציה הציקלית כך:

$$\hat{e}_k = \sum_{l=<N>} a_l b_{k-l}$$

ולאחר מכן הזזתי בחזרה את e(k) שיתקבל חזרה למיקום האמיתי שלו.

עשיתי השוואה בין הערכים של $\hat{e}(k)$ לערכים של e(n).

הפלט:

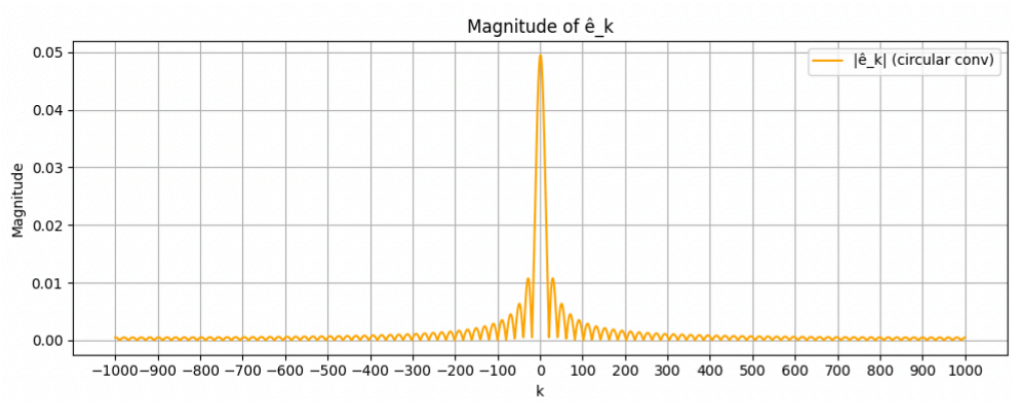
```
ê_k match e_k
```

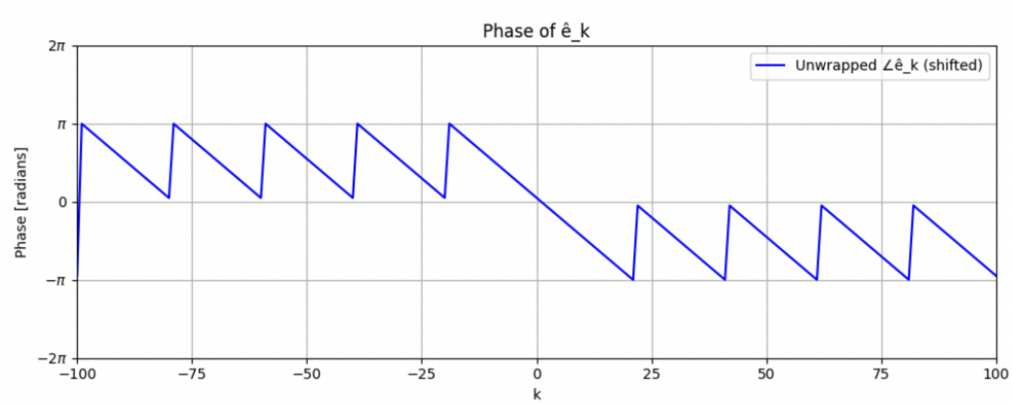
המשך הקוד:

```python

# Plot comparison of magnitudes
plt.figure(figsize=(10, 4))
plt.plot( *args: k_values, np.abs(e_k_hat), label="|ê_k| (circular conv)", color='orange')
plt.title("Magnitude of ê_k")
plt.xlabel("k")
plt.ylabel("Magnitude")
plt.xticks(np.arange(min(k_values), max(k_values)+1, 100))
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()


# Plot comparison
plt.figure(figsize=(10, 4))
plt.plot( *args: k_values, np.angle(e_k_hat), label="Unwrapped ∠ê_k (shifted)", color='blue')
plt.title("Phase of ê_k")
plt.xlabel("k")
plt.ylabel("Phase [radians]")
plt.yticks( ticks: [-2 * np.pi, -np.pi, 0, np.pi, 2 * np.pi],
            labels: [r"$-2\pi$", r"$-\pi$", r"$0$", r"$\pi$", r"$2\pi$"])
plt.grid(True)
plt.legend()
plt.xlim( *args: -100, 100)
plt.tight_layout()
plt.show()


####################################################
```

הדפסת הגרף של המגניטודה והפאזה של $\hat{e}(k)$.

הגרפים:





ניתן לראות כי התקבלו אותם גרפים כמו e(k) שחושב דרך התמרת פוריה.

האות e(k) זהה לאות $\hat{e}(k)$.

סעיף ח:

ח. נרצה עתה לבחון את ההכפלה בקוסינוס בזמן.

חשבו את : $g(n) = a(n)cos(2\pi \cdot 500 \cdot n/N)$

חשבו והציגו את האות בתדר g_k ע"י ההתמרה של $g(n)$. הצדיקו אנליטית את התוצאה.

קוד:

```python
#h

# Define g(n)
g_n = a_n * np.cos(2 * np.pi * 500 * n / N)

# Allocate output array for g_k
g_k = np.zeros(len(k_values), dtype=complex)

# Compute Fourier coefficients g_k
for i, k in enumerate(k_values):
    sum_val = 0
    for j, n_ in enumerate(n):
        sum_val += g_n[j] * np.exp(-1j * 2 * np.pi * k * n_ / N)
    g_k[i] = sum_val / N  # normalization

# Find all indices where |g_k| equals the maximum value (within tolerance)
max_val = np.max(np.abs(g_k))
indices = np.where(np.isclose(np.abs(g_k), max_val, atol=1e-10))[0]

# Print all k values with maximum magnitude
for idx in indices:
    print(f"Maximum |g_k| at k = {k_values[idx]}, magnitude = {np.abs(g_k[idx]):.4f}")
```

הגדרתי את g(n) כך:

$g(n) = a(n)cos(2\pi \cdot 500 \cdot n/N)$

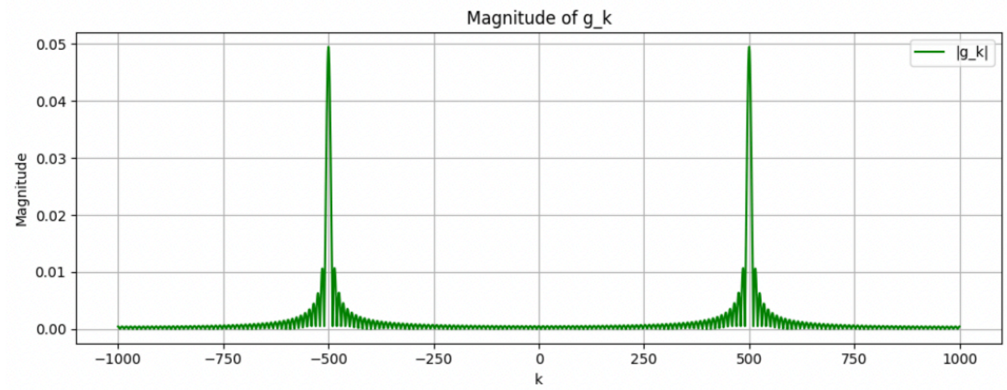
בעזרת לולאה חישבנו את התמרת פורייה וקיבלנו את g(k).
חישבנו את הערכים המקסימליים בg(n).

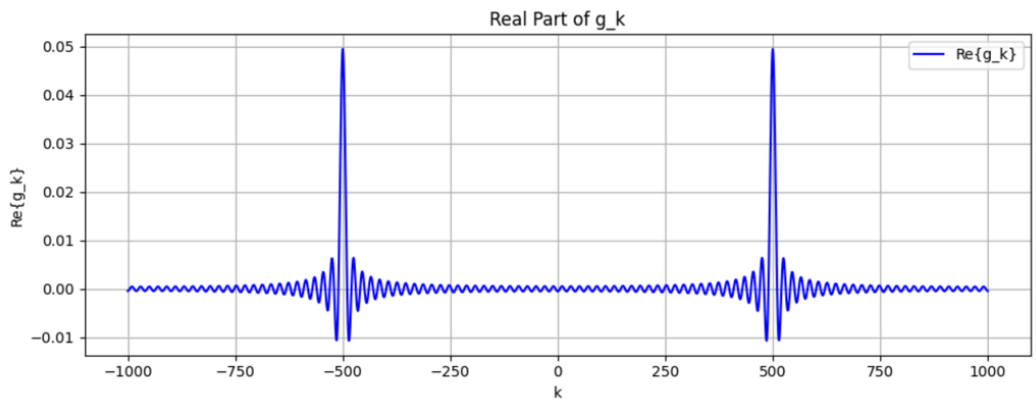
```python
# Plot magnitude of g_k
plt.figure(figsize=(10, 4))
plt.plot(k_values, np.abs(g_k), label="|g_k|", color='green')
plt.title("Magnitude of g_k")
plt.xlabel("k")
plt.ylabel("Magnitude")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()

# Plot Re{g_k}
plt.figure(figsize=(10, 4))
plt.plot(k_values, np.real(g_k), label="Re{g_k}", color='blue')
plt.title("Real Part of g_k")
plt.xlabel("k")
plt.ylabel("Re{g_k}")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()

# Plot Im{g_k}
plt.figure(figsize=(10, 4))
plt.plot(k_values, np.imag(g_k), label="Im{g_k}", color='red')
plt.title("Imaginary Part of g_k")
plt.xlabel("k")
plt.ylabel("Im{g_k}")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
```

הדפסת גרף המגניטודה של g(k),
והחלק הממשי והמדומה של g(k).

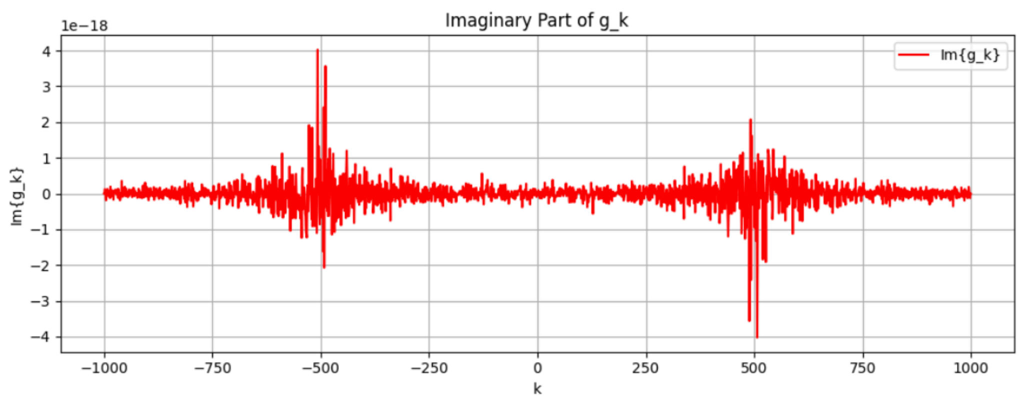
הגרפים:

הכפלה בקוסינוס בזמן זו מודולציה בתדר ולכן קיבלנו פיקים של אנרגיה בk=500 ובk=-500.





ניתן לראות כי החלק המדומה זניח וקטן מאוד.



פלט:

```
Maximum |g_k| at k = -500, magnitude = 0.0495
Maximum |g_k| at k = 500, magnitude = 0.0495
```

הצדקה אנליטית:

בעזרת זהות אוילר נקבל:

$$g(n)=a(n)\cdot cos\left(\frac{2\pi\cdot 500\cdot n}{N}\right)=a(n)\cdot\frac{1}{2}\left(e^{j\cdot\frac{2\pi\cdot 500\cdot n}{N}}+e^{-j\cdot\frac{2\pi\cdot 500\cdot n}{N}}\right)=\frac{1}{2}a(n)e^{j\cdot\frac{2\pi\cdot 500\cdot n}{N}}+\frac{1}{2}a(n)e^{-j\cdot\frac{2\pi\cdot 500\cdot n}{N}}$$

נעשה התמרת פורייה לתדר ונקבל: $a(n)\to a_k$, $e^{j\cdot\frac{2\pi\cdot 500\cdot n}{N}}\to\delta(k-500)$, $e^{-j\cdot\frac{2\pi\cdot 500\cdot n}{N}}\to\delta(k+500)$ (אקספוננט בזמן זה דלתא בתדר).

לכן נקבל: $g(n)\to g_k=\frac{1}{2}a_{k-500}+\frac{1}{2}a_{k+500}$

האות הוא מחזורי, ולכן נקבל ש $a_{k+500}=a_{k-(N-500)}$ ולכן קיבלנו פיקים ב500 וב500- ובעלי אותו ערך (המקסימלי).

☺☺☺☺☺