**Гордійчук Денис ІПС-31**

**Мова програмування: Python**

**Використовуючи бібліотеку matplotlib**

Графік функції для проміжка

# Початкові дані

T = 5 #period

Y = np.fromfile('f4.txt', sep=' ')

X = np.linspace(0, T, len(Y))

N = len(Y)

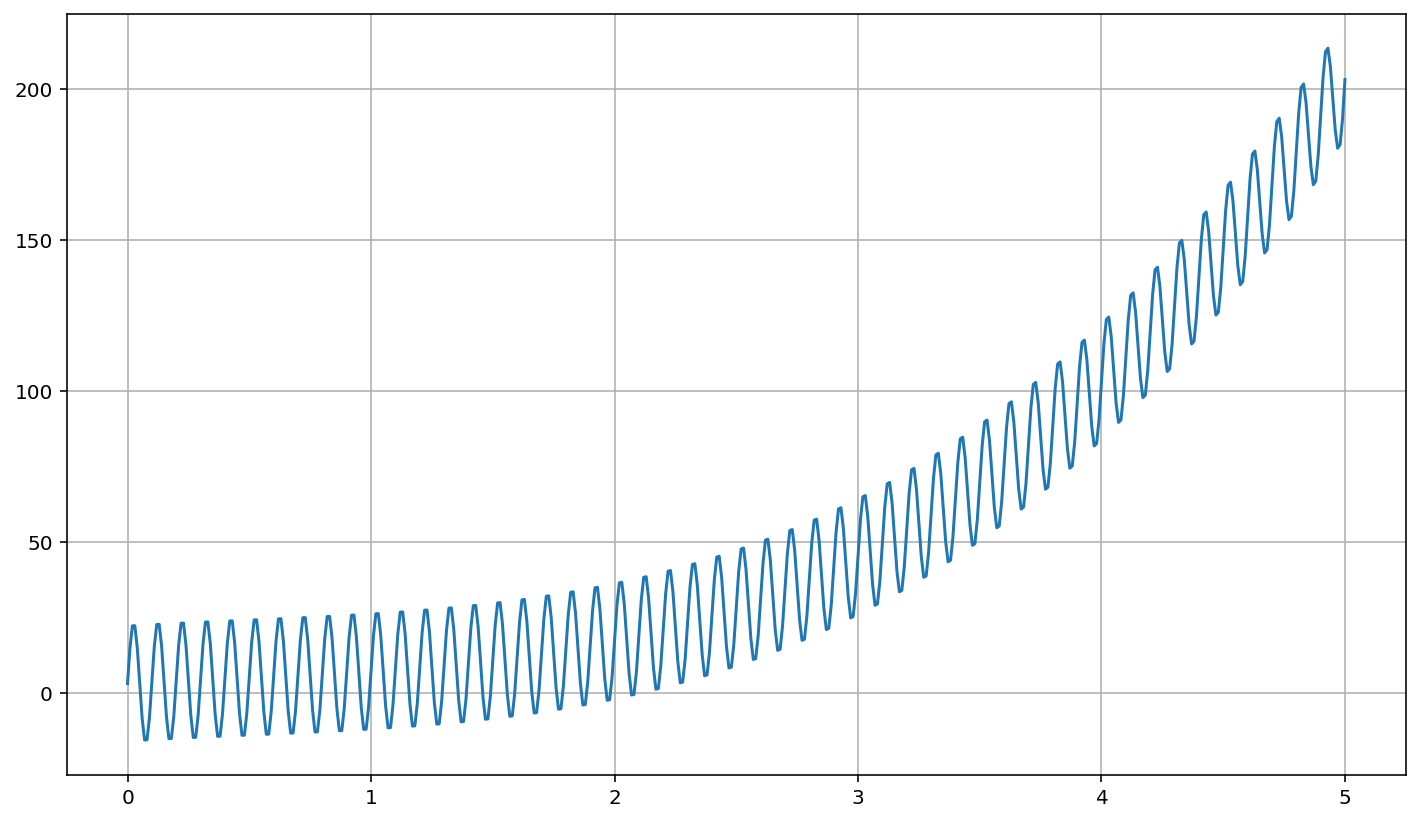
N1 = np.linspace(0, len(Y)//T, len(Y))

# Будуємо графік спостережень

plt.plot(X, Y)

plt.grid()

plt.show()



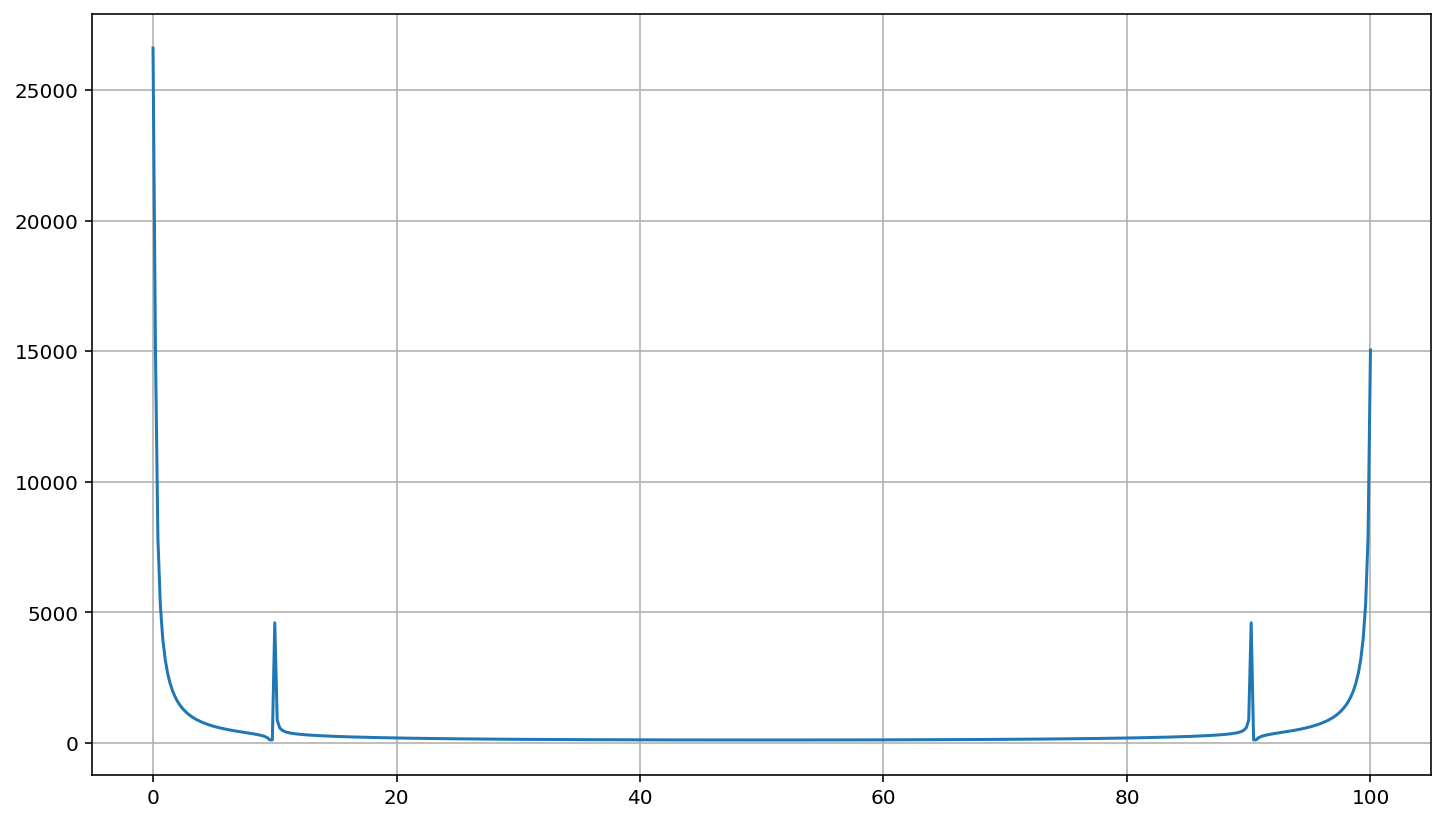
перетворення Фур’є для екстремумів.

Fur = np.abs(np.fft.fft(Y))

plt.plot(N1, Fur)

plt.grid()

plt.show()



Графік модуля перетворення Фур’є на першій половині вибірки

# Графік модуля перетворення Фур’є на першій половині вибірки

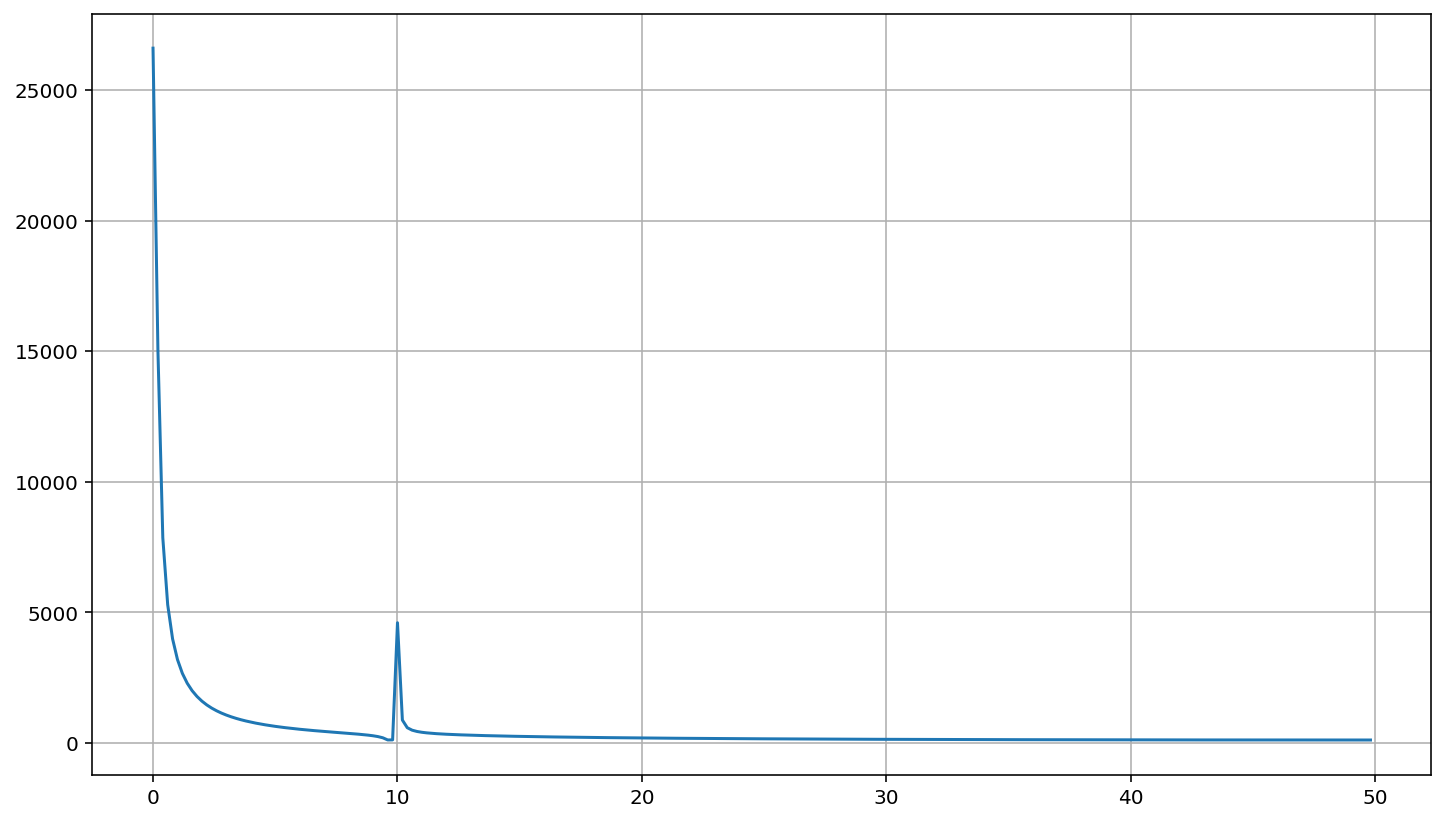
Furh = Fur[:N//2]

N2 = N1[:N//2]

plt.plot(N2, Furh)

plt.grid()

plt.show()



локальний максимум

extr, = np.where(np.logical\_and.reduce(np.array([Furh > np.roll(Furh, i) for i in range(1, 10)]), axis=0))

extr = extr[(extr >= T)]

max1 = round(extr[0] / T)

print(max1)

**Answer: 10.0**

Знаходимо за допомогою МНК.

def first(arr\_x, arr\_f):

a1 = 0;

a2 = 0;

a3 = 0;

a4 = 0;

a5 = 0;

minus = 0;

for x,f in zip(arr\_x, arr\_f):

a1 += pow(x, 6)

a2 += pow(x, 5)

a3 += pow(x, 4)

a4 += pow(x,3)\*np.sin(2 \* np.pi \* max1 \* x)

a5 += pow(x,3)

minus += pow(x, 3)\*f

return [[a1, a2, a3, a4, a5], minus]

def second(arr\_x, arr\_f):

a1 = 0;

a2 = 0;

a3 = 0;

a4 = 0;

a5 = 0;

minus = 0;

for x,f in zip(arr\_x, arr\_f):

a1 += pow(x, 5)

a2 += pow(x, 4)

a3 += pow(x, 3)

a4 += pow(x,2)\*np.sin(2 \* np.pi \* max1 \* x)

a5 += pow(x,2)

minus += pow(x,2)\*f

return [[a1, a2, a3, a4, a5], minus]

def third(arr\_x, arr\_f):

a1 = 0;

a2 = 0;

a3 = 0;

a4 = 0;

a5 = 0;

minus = 0;

for x,f in zip(arr\_x, arr\_f):

a1 += pow(x, 4)

a2 += pow(x, 3)

a3 += pow(x, 2)

a4 += pow(x,1)\*np.sin(2 \* np.pi \* max1 \* x)

a5 += pow(x,1)

minus += pow(x,1)\*f

return [[a1, a2, a3, a4, a5],minus]

def fourth(arr\_x, arr\_f):

a1 = 0;

a2 = 0;

a3 = 0;

a4 = 0;

a5 = 0;

minus = 0;

for x,f in zip(arr\_x, arr\_f):

a1 += pow(x, 3) \* np.sin(2 \* np.pi \* max1 \* x)

a2 += pow(x, 2) \* np.sin(2 \* np.pi \* max1 \* x)

a3 += pow(x, 1) \* np.sin(2 \* np.pi \* max1 \* x)

a4 += np.sin(2 \* np.pi \* max1 \* x) \* np.sin(2 \* np.pi \* max1 \* x)

a5 += np.sin(2 \* np.pi \* max1 \* x)

minus += np.sin(2 \* np.pi \* max1 \* x)\*f

return [[a1, a2, a3, a4, a5], minus]

def fifth(arr\_x, arr\_f):

a1 = 0;

a2 = 0;

a3 = 0;

a4 = 0;

a5 = 0;

minus = 0;

for x,f in zip(arr\_x, arr\_f):

a1 += pow(x, 3)

a2 += pow(x, 2)

a3 += pow(x, 1)

a4 += np.sin(2 \* np.pi \* max1 \* x)

a5 += 1

minus += f

return [[a1, a2, a3, a4, a5], minus]

A = np.stack((first(X,Y)[0], second(X,Y)[0], third(X,Y)[0], fourth(X,Y)[0], fifth(X,Y)[0]), axis=1)

c = [first(X,Y)[1], second(X,Y)[1], third(X,Y)[1], fourth(X,Y)[1], fifth(X,Y)[1]]

print(A)

print(c)

Розв’язуємо систему рівнянь і отримуємо коефіцієнти

Result = np.linalg.solve(A, c)

tuple(Result.round(4))

**Answer: (2.0, -3.0, 5.0, 20.0, 3.0)**

Підставляємо отримані коефіцінти у функцію, обчислюємо її значення:

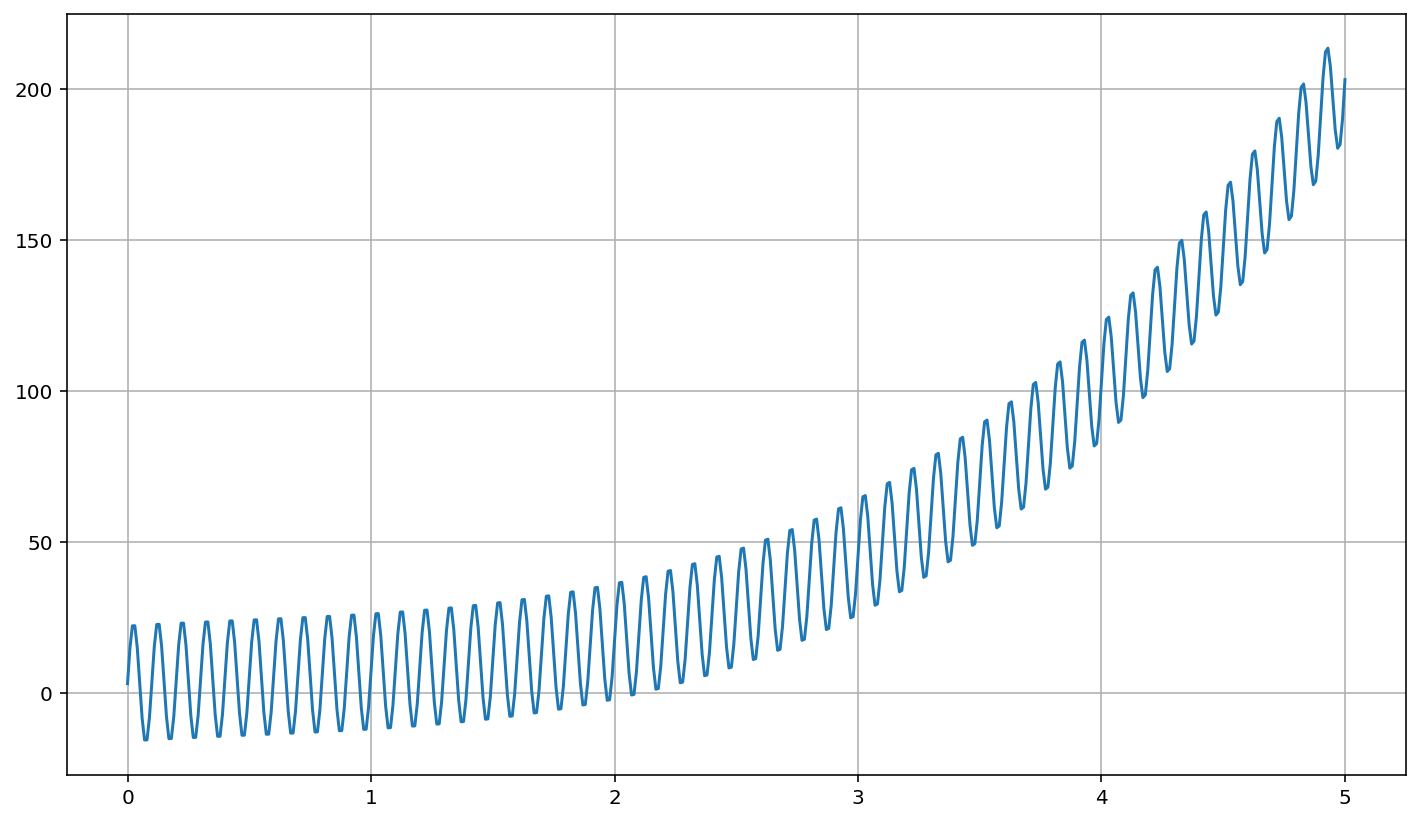
approximated = np.vectorize(lambda t: Result[0] \* t \*\* 3 + Result[1] \* t \*\* 2 + Result[2] \* t + Result[3] \* np.sin(2 \* np.pi \* max1 \* t) + Result[4])

y\_approximated = approximated(X)

plt.plot(X, y\_approximated)

plt.grid()

plt.show()



Обчислюємо квадратичну похибку функції

np.sum((y\_approximated - Y) \*\* 2)

**Answer: 4.498561577585358e-07**