

Санкт-Петербургский Политехнический Университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Телекоммуникационные технологии

Отчет по лабораторной работе №2
Ряд Фурье. Преобразование Фурье. Корреляция

Работу
выполнил:
Балсутьев В.А.
Группа: 33501/4
Преподаватель:
Богач Н.В.

Санкт-Петербург
2017

Содержание

1. Цель работы	2
2. Постановка задачи	2
3. Теоретическая информация	2
3.1. Ряд Фурье	2
3.2. Преобразование Фурье	3
4. Ход работы	4
4.1. Гармонический сигнал	4
4.2. Прямоугольный импульс	5
4.3. Периодический прямоугольный сигнал	7
4.4. Пилообразный сигнал	9
4.5. Функция Дирихле	10
4.6. Корреляция	11
5. Вывод	12

1. Цель работы

Получить представление о спектрах телекоммуникационных сигналов.

2. Постановка задачи

- Для сигналов, построенных в лабораторной работе No1, выполните расчет преобразования Фурье, получите спектры.
- С помощью функции корреляции найдите позицию синхросылки [101] в сигнале [0001010111000010]. Получите пакет данных, если известно, что его длина составляет 8 бит без учета синхросылки. Вычислите корреляцию прямым методом, воспользуйтесь алгоритмом быстрой корреляции, сравните время работы обоих алгоритмов.
- Теоретические положения к лабораторной работе: Справочные материалы: А.Б.Сергиенко Цифровая обработка сигналов. Глава 1, сс.25–55, Глава 5, сс. 284–285.

3. Теоретическая информация

3.1. Ряд Фурье

Разложению в ряд Фурье могут подвергаться периодические сигналы. При этом они представляются в виде суммы гармонических функций либо комплексных экспонент с частотами, образующими арифметическую прогрессию. Для того чтобы такое разложение существовало, фрагмент сигнала длительностью в один период должен удовлетворять условиям Дирихле:

- не должно быть разрывов второго рода (с уходящими в бесконечность ветвями функции)
- число разрывов первого рода (скачков) должно быть конечным
- число экстремумов должно быть конечным (в качестве примера функции, которая на конечном интервале имеет бесконечное число экстремумов, можно привести $\sin(1/x)$ в окрестности нуля)

В зависимости от конкретной формы базисных функций различают несколько форм записи ряда Фурье.

Следует отметить, что ряд Фурье может быть применен для разложения и не периодические сигналы. При этом оговаривается временной интервал, для которого строится ряд Фурье, а в остальные моменты времени сигнал считается равным нулю.

Как уже было сказано выше существуют несколько форм представления ряда Фурье:

- Синусно-косинусная форма

$$s(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} (a_k \cos(k\omega_1 t) + b_k \sin(k\omega_1 t))$$

В данной формуле $\omega_1 = 2\pi/T$ - круговая частота, соответствующая периоду повторения сигнала, равному T . Входящие в формулу кратные ей частоты $k\omega_1$ называются гармониками (k - ая гармоника $k\omega_1$).

- Вещественная форма

Некоторое неудобство предыдущей формы заключается в том, что для каждого значения индекса суммирования k в формуле фигурируют два слагаемых - синус и косинус. С помощью тригонометрических формул и не хитрых преобразований можно синусно-косинусную форму привести к более короткой записи:

$$s(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} (A_k \cos(k\omega_1 t + \phi_k))$$

- Комплексная форма

Данная форма представления ряда Фурье является самой общей, поскольку охватывает надмножество вещественных чисел - комплексные. Данная форма получается подставлением косинуса в виде полусуммы комплексных экспонент (следствие из формулы Эйлера):

$$\cos x = \frac{1}{2}(e^{jx} + e^{-jx})$$

Применив данное преобразование к вещественной форме ряда Фурье, получим суммы комплексных экспонент с положительными и отрицательными показателями:

$$s(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \frac{A_k}{2} (\exp(jk\omega_1 t + j\phi_k) + \exp(-jk\omega_1 t - j\phi_k))$$

Теперь будем трактовать экспоненты со знаком минус в показателе как члены ряда с отрицательными номерами. В рамках этого же общего подхода постоянное слагаемое $a_0/2$ станет членом ряда с нулевым номером. В результате получится комплексная форма записи ряда Фурье:

$$\sum_{k=1}^{\infty} C_k e^{-jk\omega_1 t}$$

Из чего следует, что коэффициенты C_k ряда Фурье в комплексной форме можно рассчитывать по формуле:

$$C_k = \frac{1}{T} \int_{-T/2}^{T/2} s(t) \exp(-jk\omega_1 t) dt$$

Совокупность амплитуд гармоник ряда Фурье часто называют амплитудным спектром, а совокупность их фаз - фазовым спектром.

3.2. Преобразование Фурье

Как уже было сказано выше, основной формой представления ряда Фурье является комплексная и при помощи достаточно сложных математических выводов, нетривиальных переходов и рассуждений получаем из формулы ряда Фурье формулу прямого преобразования Фурье, с помощью которой будем находить спектры, а значит и осуществлять спектральный анализ:

$$S(\omega) = \int_{-\infty}^{\infty} s(t) e^{-jk\omega t} dt$$

В данном случае $S(\omega)$ не что иное, как комплексный коэффициент ряда Фурье, именно после вычисления данных коэффициентов можно строить спектры и именно данные коэффициенты возвращает функция `fft` как MATLAB, так и в python. Также существует формула обратного преобразования Фурье, которая нам пригодится для вычисления быстрой корреляции:

$$s(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S(\omega) e^{-j\omega t} d\omega$$

4. Ход работы

4.1. Гармонический сигнал

Для начала построим гармонический сигнал $s_1(t) = \cos(2\pi f_0 t + \phi)$ и с помощью пэкиджа `fftpack` из `scipy` построим спектр:

Листинг 1: `source01.py`

```
1 import numpy as np
2 import math
3 import matplotlib.pyplot as plt
4 from scipy import signal
5 from scipy.fftpack import fft, fftfreq
6
7 %matplotlib inline
8
9 Fs = 8000 # discrete frequency
10 t = np.matrix(np.arange(0.0, 1.0, (1 / Fs))).transpose()
11 A = 2 # amplitude
12 f0 = 1000
13 phi = np.pi / 4
14 s1 = A * np.cos(2 * f0 * t + phi) # harmonic signal
15 fig = plt.figure(figsize=(6,6))
16 plt.title('Гармонический_сигнал')
17 plt.plot(t[1:100], s1[1:100])
18 fig.savefig('pictures/001_1harmonic.png', dpi=200)
19 plt.show()
20 N = len(t)
21 T = 1.0 / Fs
22 yf = fft(s1)
23 xf = fftfreq(N, 1.0 / Fs)
24 fig = plt.figure()
25 plt.title('Амплитудный_спектр')
26 plt.ylim(0, 50)
27 plt.xlim(-500, 500)
28 plt.plot(xf, np.abs(yf))
29 fig.savefig('pictures/001_2harmsspectr.png', dpi=70)
30 plt.show()
```

Получаем в результате интерпретации нашей программы графики исходного сигнала и его спектр:

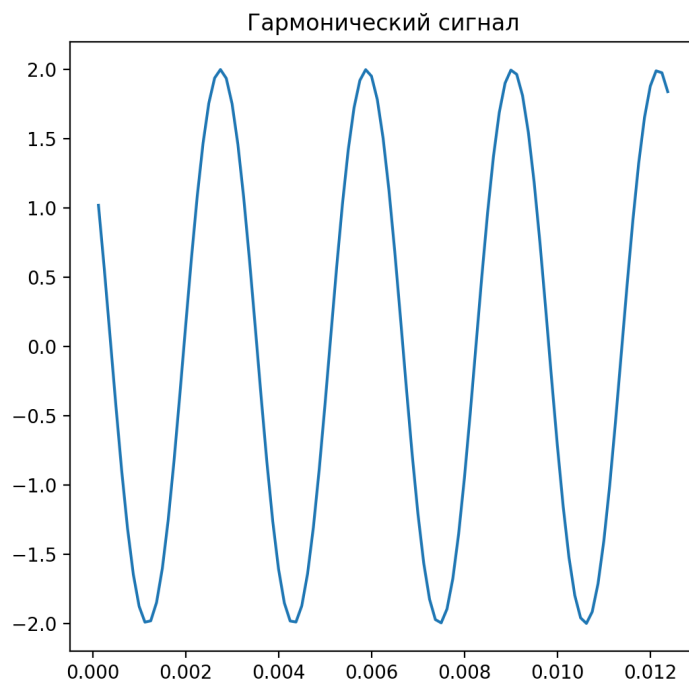


Рисунок 4.1. Гармонический сигнал

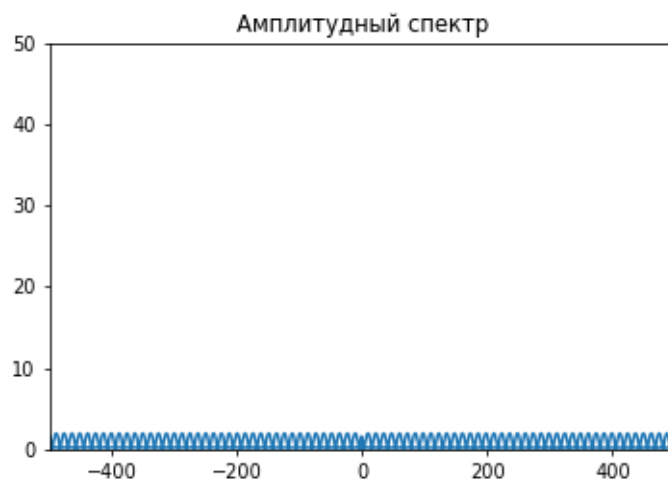


Рисунок 4.2. Амплитудный спектр

4.2. Прямоугольный импульс

Из-за отсутствия функции одиночного импульса в `scipy.signal` реализуем собственную `rectimpls(t, width)`, где `t` - время в секундах, `width` - ширина импульса также в секундах. И повторим процедуру построения спектра с помощью `fftpack`:

Листинг 2: `source02.py`

```

1
2 import matplotlib.pyplot as plt
3 import numpy as np
4 from scipy import signal
5 from scipy.fftpack import fft, fftfreq
6
7
8 # rectangular impulse
9 def rect_impls_1(t, width):
10     sig = np.zeros(len(t))
11     for i in sig:
12         if -width/2 <= i < width / 2:
13             i = 1
14     for i in range(len(sig)):
15         if -width/2 <= t[i] < width / 2:
16             sig[i] = 1
17     return sig
18
19 Fs = 800
20 t = np.linspace(-1, 1, 2 * Fs)
21 width = 0.5
22 fig = plt.figure()
23 y = rect_impls_1(t, width)
24 plt.plot(t, y)
25 plt.title('Прямоугольный импульс')
26 fig.savefig('pictures/002_1rectImpl.png', dpi=100)
27 plt.show()
28 N = len(t)
29 T = 1.0 / Fs
30 yf = fft(y)
31 xf = fftfreq(N, 1.0 / Fs)
32 fig = plt.figure()
33 plt.title('Амплитудный спектр')
34 plt.ylim(0, 200)
35 plt.xlim(-70, 70)
36 plt.plot(xf, np.abs(yf))
37 fig.savefig('pictures/002_2rectImplSpectr.png', dpi=100)
38 plt.show()

```

Получаем в качестве спектра ожидаемый $s(f) = \text{sinc}(f)$

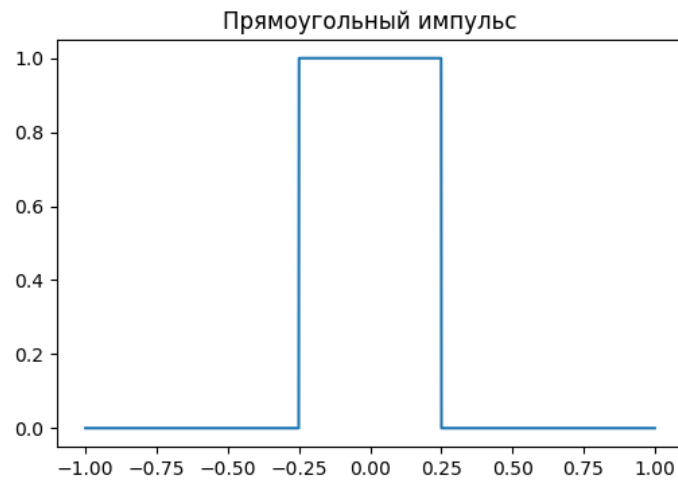


Рисунок 4.3. Прямоугольный импульс

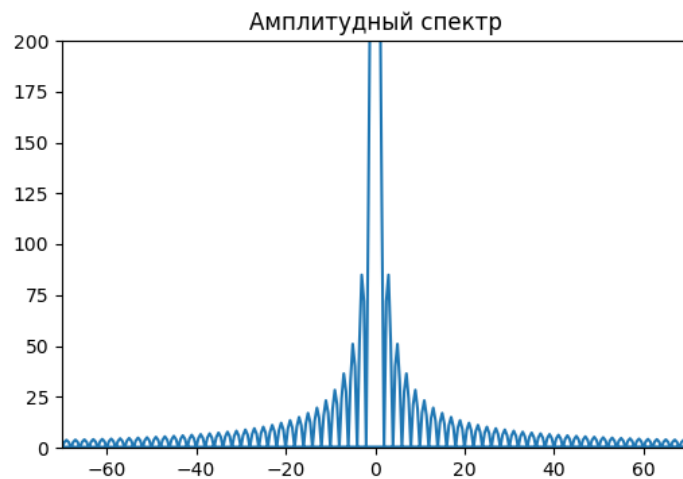


Рисунок 4.4. Амплитудный спектр

4.3. Периодический прямоугольный сигнал

Далее проведем тот же эксперимент для периодического прямоугольного сигнала:

Листинг 3: source03.py

```

1
2
3 import numpy as np
4 import math
5 import matplotlib.pyplot as plt
6 from scipy import signal
7
8 Fs = 1000 # discrete frequency
9 t = np.linspace(-1, 1, 2 * Fs)
10 A = 5
11 fig = plt.figure()

```



```

12 y = A * signal.square( 2 * np.pi * 5 * t, 0.2)
13 plt.plot(t, y)
14 # fig.savefig('pictures/003_1rectImplses.png', dpi=100)
15 plt.show()
16 N = len(t)
17 T = 1.0 / Fs
18 yf = fft(y)
19 xf = fftfreq(N, 1.0 / Fs)
20 fig = plt.figure()
21 plt.title('Амплитудный_спектр')
22 # plt.ylim(0, 200)
23 plt.xlim(-70, 70)
24 plt.plot(xf, np.abs(yf))
25 # fig.savefig('pictures/003_2rectImplSpectr.png', dpi=100)
26 plt.show()

```

Получаем в соответствии с теорией:

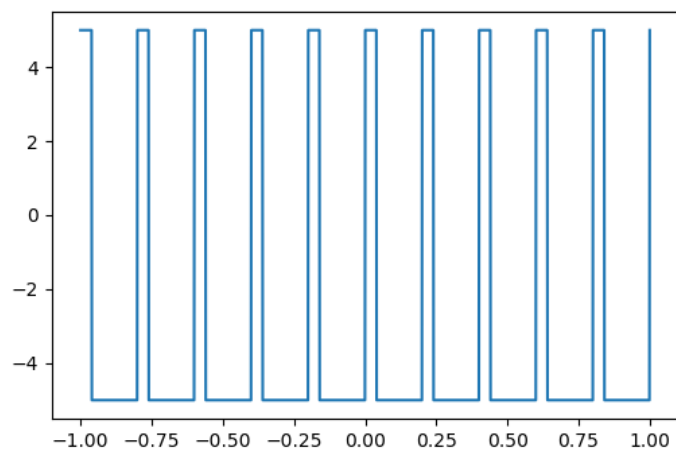


Рисунок 4.5. Прямоугольный сигнал

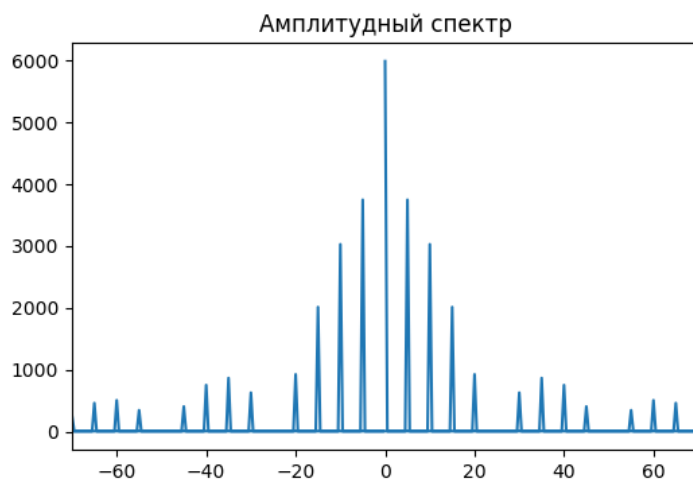


Рисунок 4.6. Амплитудный спектр

4.4. Пилообразный сигнал

Воспользовавшись стандартной функцией `sawtooth` из `scipy.signal` строим пилообразный сигнал и далее его спектр:

Листинг 4: `source04.py`

```
1
2
3 import numpy as np
4 import matplotlib.pyplot as plt
5 from scipy import signal
6
7 Fs = 8000 # discrete frequency
8 t = np.linspace(0, 0.5, Fs)
9 fig = plt.figure()
10 plt.title('Последовательность_треугольных_импульсов')
11 y = signal.sawtooth(2 * np.pi * 8 * t)
12 plt.plot(t, y)
13 fig.savefig('pictures/004_1sawtooth.png', dpi=100)
14 plt.show()
15 N = len(t)
16 T = 1.0 / Fs
17 yf = fft(y)
18 xf = fftfreq(N, 1.0 / Fs)
19 fig = plt.figure()
20 plt.title('Амплитудный_спектр')
21 # plt.ylim(0, 200)
22 # plt.xlim(-70, 70)
23 plt.xlim(-10, 10)
24 plt.plot(xf, np.abs(yf))
25 fig.savefig('pictures/004_2sawToothSpectr.png', dpi=100)
26 plt.show()
```

Получаем в соответствии с теорией:

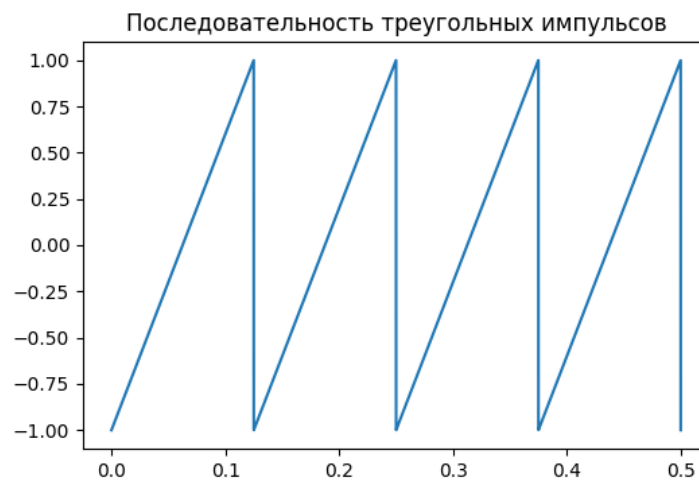


Рисунок 4.7. Пилообразный сигнал



Рисунок 4.8. Амплитудный спектр

4.5. Функция Дирихле

С помощью `diric` из `scipy.special` сигнал соответствии с формулой Дирихле:

$$\frac{\sin(nx/2)}{n\sin(x/2)}, n \in \mathbb{Z}$$

и далее его спектр для степеней $n = 7$ и 8 соответственно:

Листинг 5: `source05.py`

```

1
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from scipy import signal
5 from scipy import special
6
7 Fs = 8000
8 t = np.linspace(-8*np.pi, 8*np.pi, Fs)
9 # fig = plt.figure(figsize=(8, 8))
10 plt.subplot(2, 1, 1)
11 y1 = special.diric(t, 7)
12 plt.plot(t, y1)
13 plt.title('Функции Дирихле, n={}, n={}'.format(7, 8))
14 plt.subplot(2, 1, 2)
15 y2 = special.diric(t, 8)
16 plt.plot(t, y2)
17 # plt.title('Функция Дирихле, n={}'.format(8))
18 fig.savefig('pictures/005_1dirichle.png', dpi = 200)
19 plt.show()
20
21 N = len(t)
22 T = 1.0 / Fs
23 yf1 = fft(y1)
24 yf2 = fft(y2)
25 xf = fftfreq(N, 1.0 / Fs)
26
27
28 fig = plt.figure()
29 plt.subplot(2, 1, 1)

```

```

30 plt.xlim(-100,100)
31 plt.plot(xf, np.abs(yf1))
32 plt.title('Спектры_Функции_Дирихле, n={}, n={}'.format(7,8))
33 plt.subplot(2, 1, 2)
34 plt.plot(xf, np.abs(yf2))
35 plt.xlim(-100,100)
36 # plt.title('Функция_Дирихле, n={}'.format(8))
37 fig.savefig('pictures/005_2dirichleSpectr.png', dpi = 200)
38 plt.show()

```

Получаем в соответствии с теорией:

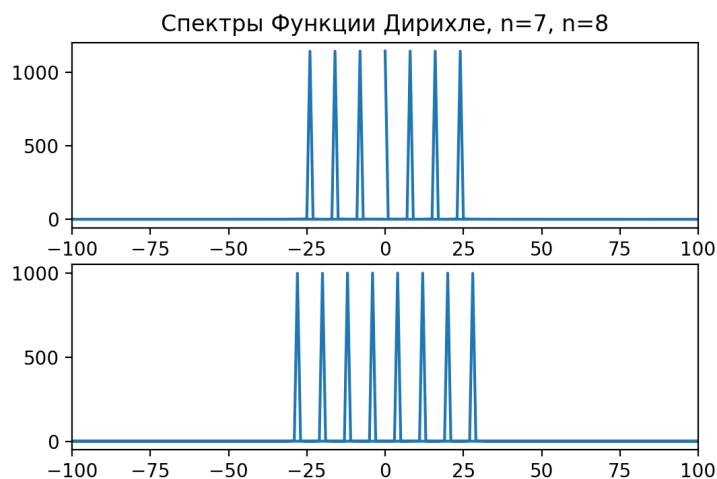


Рисунок 4.9. Функции Дирихле

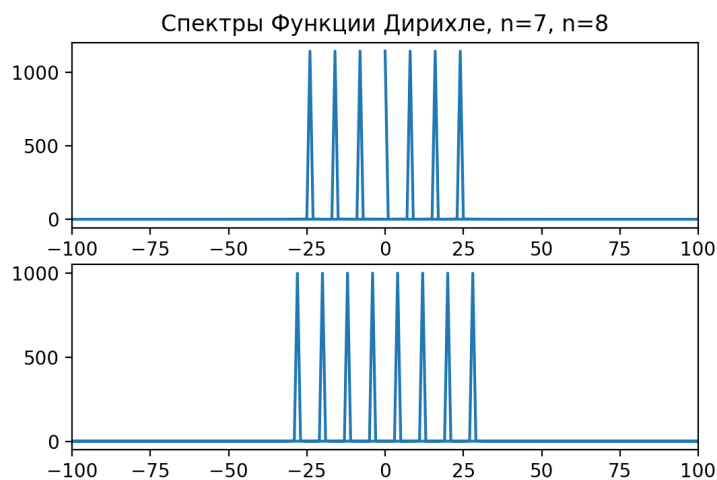


Рисунок 4.10. Амплитудный спектр

4.6. Корреляция

Найдя функцию для вычисления корреляции в `scipy.signal`, найдем корреляцию. Далее в соответствии с пособием из которого мы взяли формулу вычисления быстрой корреляции(стр 303 параграф 5.2.3):

$$r(j) = \frac{F_D^{-1}[X_1^*(k)X_2(k)]}{N},$$

где r - корреляция, F_D^{-1} - обратное дискретное преобразования Фурье, N - степень дискретизации, $X_1(k), X_2(k)$ - ДПФ - образы для периодических последовательностей $x_1(l), x_2(r)$, корреляцию которых считаем.

Листинг 6: source06.py

```

1
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import math
5 from scipy import signal
6 from scipy.fftpack import fft, fftfreq, ifft
7
8
9 sig = np.array([0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0])
10 parcel = np.array([1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
11 sigF = fft(sig)
12 parcelF = fft(parcel)
13 corr_fast = ifft(sigF * parcelF.conjugate()) / len(sig)
14 corr_fast = corr_fast.real
15
16 print('Fast_correlation')
17 print(len(corr_fast))
18 for i in corr_fast:
19     print('{0:2.4f}'.format(i), end=' ')
20
21 corr = signal.correlate(sig, parcel, mode='same') / len(sig)
22 print('\nUsual_correlation')
23 print(len(corr))
24 for i in corr: print('{0:2.4f}'.format(i), end=' ')

```

Выполняем сценарий и получаем:

```

Fast correlation
16
0.0000 0.0625 -0.0000 0.1250 0.0000 0.1250 0.0625 0.1250 0.0625 0.0625 0.0000 0.0000 0.0625 0.0000 0.0625 0.000
Usual correlation
16
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0625 0.0000 0.1250 0.0000 0.1250 0.0625 0.1250

```

Рисунок 4.11. Корреляция

5. Вывод

Таким образом мы рассмотрели преобразование Фурье и научились находить его для различных сигналов с помощью библиотеки `python-scipy.signal`. Следует отметить, что преобразование Фурье в обработке сигналов обычно рассматривается как декомпозиция сигнала на частоты и амплитуды, то есть обратимый переход от временного пространства в частотное пространство.