РАБОТА №10. ФИЛЬТРАЦИЯ СИГНАЛОВ С ПРИМЕНЕНИЕМ НЕПРЕРЫВНОГО ВЕЙВЛЕТ-ПРЕОБРАЗОВАНИЯ

Цель работы: изучение возможностей применения непрерывного вейвлет-преобразования для фильтрации сигналов.

Планируемая продолжительность: от 2 до 4 академических часов.

Тип работы: с использованием компьютерных средств.

Фильтрация сигнала с применением непрерывного вейвлет-преобразования

В данном практическом занятии, как и на предыдущих, рассматривается НВП дискретных сигналов. Результатом такого НВП является массив коэффициентов (двумерный массив), в котором от столбца к столбцу меняется параметр b, связанный с временем, а от строки к строке – параметр a, связанный с частотой. Таким образом, массив вейвлет-коэффициентов можно охарактеризовать как поверхность, содержащая сведения о изменении сигнала во времени (по оси x) и по частоте (по оси y).

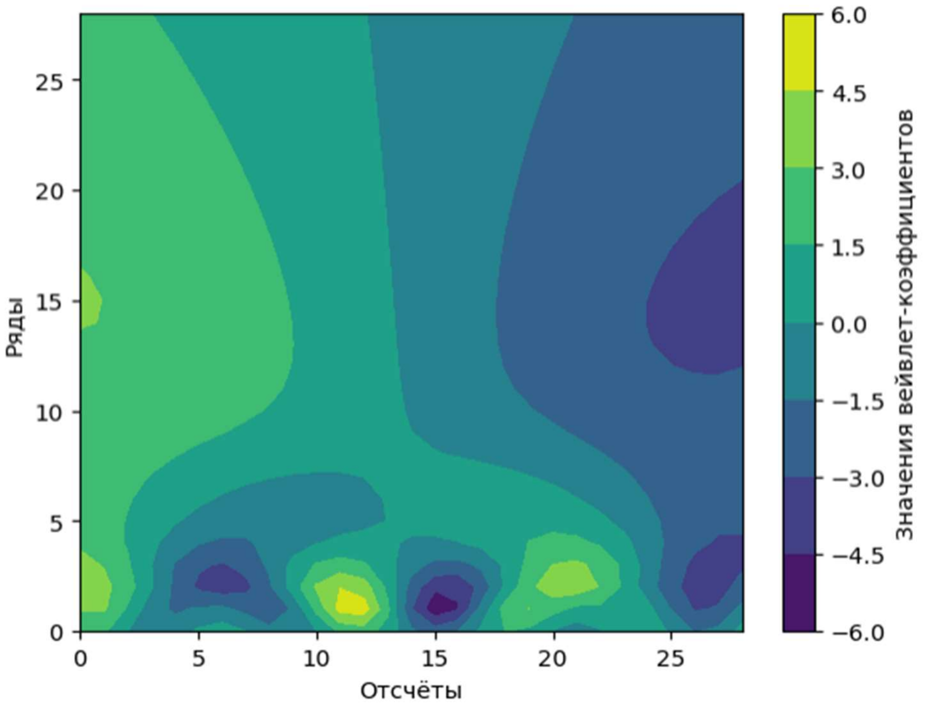

Рис. 10.1. Вейвлет-коэффициенты

Помимо этого известно, что имеется обратное НВП (ОНВП), позволяющее из массива вейвлет-коэффициентов восстановить исходный сигнал. Следовательно, **если обнулить часть вейвлет-коэффициентов в массиве и затем восстановить из него сигнал с помощью ОНВП, можно очистить сигнал от соответствующих частотных компонентов в соответствующем временном участке сигнала.** Именно таким образом и реализуется фильтрация на базе НВП.

Обратное непрерывное вейвлет преобразование

Известно выражение для прямого НВП:

$$W(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} f(t) \cdot \psi\left(\frac{t-b}{a}\right) dt,$$

где $\psi\left(\frac{t-b}{a}\right)$ - вейвлет, используемый для анализа;

a – параметр масштаба вейвлета;

b – параметр сдвига вейвлета.

Для дискретных сигналов данное выражение принимает вид:

$$W(a,b) = \frac{\Delta t}{\sqrt{a}} \sum_{j=1}^{N} S_j \cdot \psi\left(\frac{j \cdot \Delta t - b}{a}\right), \tag{14.1}$$

где Δt - интервал дискретизации;

S - дискретный сигнал.

При этом *a, b* также изменяют дискретно с некоторым шагом. В результате и формируется массив вейвлет-коэффициентов.

Для восстановления из массива вейвлет-коэффициентов исходного сигнала используют выражение:

$$f(t) = \frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{1}{\sqrt{a} \cdot a^2} W(a,b) \cdot \psi\left(\frac{t-b}{a}\right) da\ db,$$

где C_ψ - постоянная допустимости используемого вейвлета, **для используемого в работе вейвлета МНАТ $C_\psi = \pi$.**

Для дискретных сигналов с учетом $C_\psi = \pi$ данное выражение принимает вид:

$$S_j = \frac{1}{\pi} \sum_{k=1}^{L} \sum_{i=1}^{M} \frac{1}{\sqrt{i \cdot \Delta a} \cdot (i \cdot \Delta a)^2} W_{i,k} \cdot \psi\left(\frac{j \cdot \Delta t - k \cdot \Delta b}{i \cdot \Delta a}\right), \tag{14.2}$$

где $\varDelta t$ - интервал дискретизации;

$\varDelta a$ - шаг изменения масштаба (выбирается при прямом НВП);

$\varDelta b$ - шаг изменения сдвига (выбирается при прямом НВП);

i – текущая строка;

k – текущий столбец;

$W_{i,k}$ - текущий элемент в массиве вейвлет-коэффициентов;

j – текущий номер элемента в дискретном сигнале S.

Реализация алгоритма фильтрации с помощью НВП Python

Для начала импортируем следующие зависимости.

```
import numpy as np
import matplotlib.pyplot as plt
from typing import Callable
from scipy.fft import fft, fftfreq
%matplotlib widget
```

Рис. 10.2. Импорт зависимостей

А также из предыдущей работы нужно взять функции plot, plot_scalogram, mexican_hat, wavelet_transform

После чего – сам фильтруемый сигнал.

```
def y(x: np.ndarray) -> np.ndarray:
    result = np.zeros(x.size)
    for idx in range(result.size):
        result[idx] = np.sin(0.5 * 2 * np.pi * x[idx]) + \
            np.sin(0.8 * 2 * np.pi * x[idx] + (np.pi / 6))
    return result
```

Рис. 10.3. Тестовый сигнал

```python
N = 30
x = np.arange(start=1, stop=30)
signal = y(x * (2 * np.pi / N))
plot(signal)
```

Рис. 10.4. Создание тестового сигнала

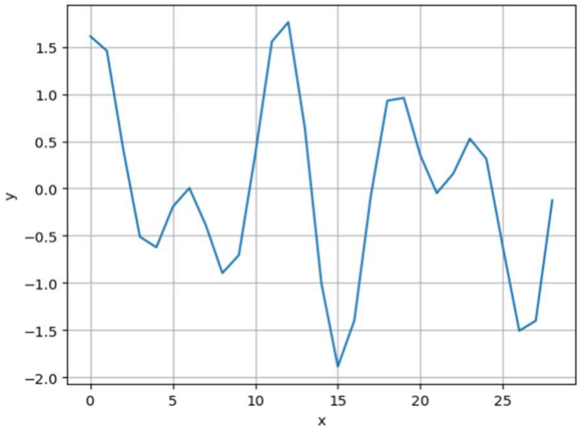

Рис. 10.5. График тестового сигнала

И обратное НВП (в нашем случае $\Delta t = \Delta a = \Delta b$).

```python
def inverse_wavelet_transform(
    array: np.ndarray,
    psi_func: np.ndarray
) -> np.ndarray:
    dt = 2 * np.pi / signal.size
    """
    Так как на вход передается массив, где
    элементы массива являются срезам по рядям, то
    нужно транспонировать матрицу, чтобы получить
    срезы по отсчётам
    """
    array = array.T

    def inverse_wavelet_element_transform(t: float) -> np.ndarray:
        result_element = np.zeros(shape=(array.shape[0], array.shape[0]))
        for y in range(1, result_element.shape[0] + 1):
            for x in range(1, result_element.shape[1] + 1):
                result_element[y - 1, x - 1] = array[y - 1, x - 1] * \
                    psi_func((t - y * dt) / (x * dt)) * \
                    (1 / (((x * dt) ** 2) * np.sqrt(x * dt)))
        return ((dt ** 2) / np.pi) * np.sum(result_element)

    result = []
    for t in range(array.shape[0]):
        result.append(inverse_wavelet_element_transform((t + 1) * dt))
    return np.array(result)
```

Рис. 10.6. Обратное НВП

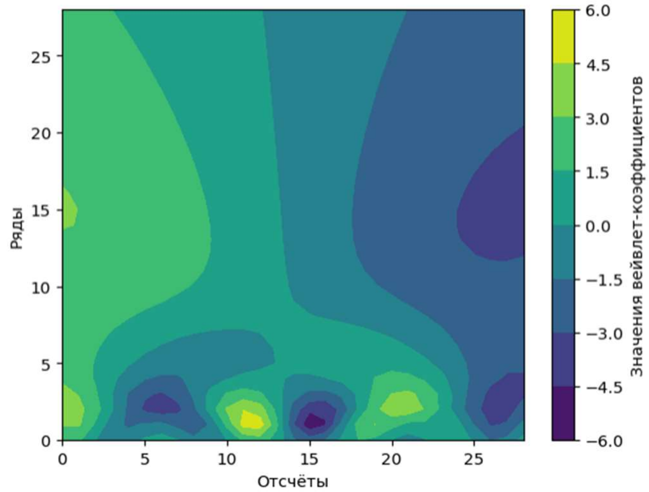

Рис. 10.7. График вейвлет-коэффициентов

Попробуем отфильтровать в первых 10 отсчётах высокие частоты, которые, согласно графику коэффициентов, сосредоточены в рядах 0-7 (вторая граница берётся не включительно поэтому индексы на единицу больше).

```
wt = wavelet_transform(signal, mexican_hat)
wt[0:8, 0:11] = 0
plot_scalogram(wt)
```

Рис. 10.8. Фильтрация коэффициентов

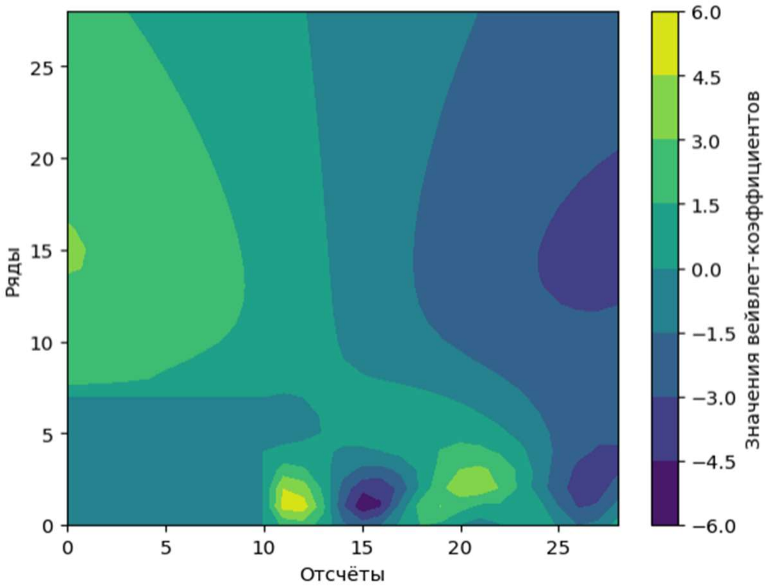

Рис. 10.9. Отфильтрованные коэффициенты

Строим график сигнала, полученного из отфильтрованного массива.

```
plot(inverse_wavelet_transform(wt, mexican_hat))
```

Рис. 10.10. Построение графика сигнала

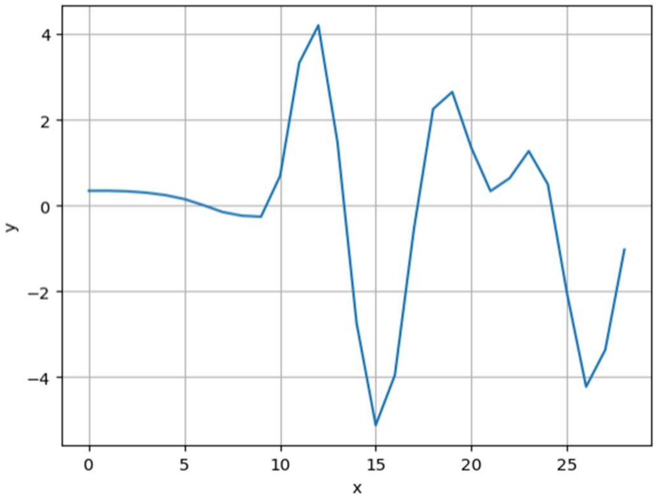

Рис. 10.11. Отфильтрованный сигнал

Как видим, в сигнале отсутствуют выбросы на границе фильтрации, при том, что коэффициенты были обнулены без какого-либо окна.

Попробуем устранить только высокочастотную составляющую в более широкой временной области сигнала, не подавив низкочастотную.

```
wt = wavelet_transform(signal, mexican_hat)
wt[0:3, 0:21] = 0
plot_scalogram(wt)
```

Рис. 10.12. Фильтрация коэффициентов

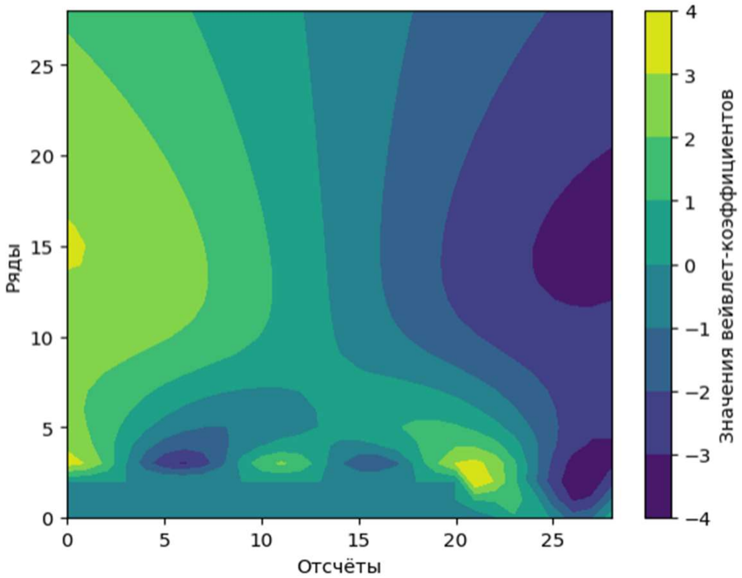

Рис. 10.13. Отфильтрованные коэффициенты

```
1  plot(inverse_wavelet_transform(wt, mexican_hat))
```

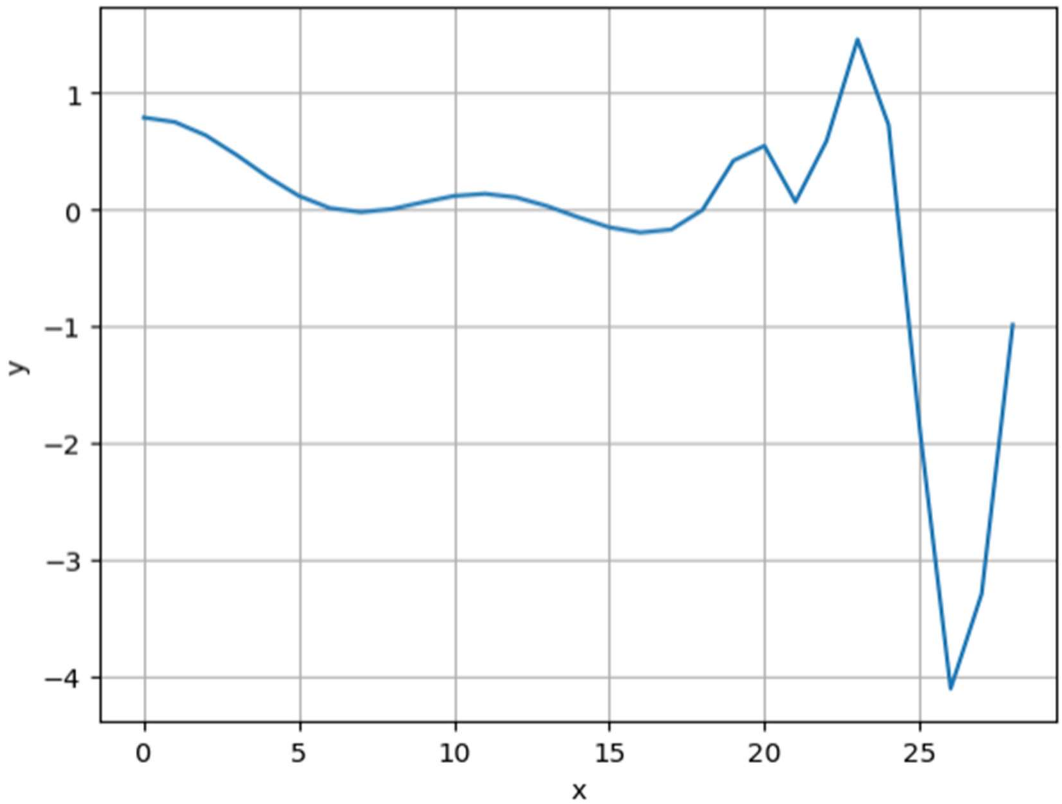

Рис. 10.14. Отфильтрованный сигнал

Задание

Реализовать алгоритм фильтрации из последнего раздела для заданного вариантом сигнала.

Таблица 14.1

Варианты заданий

Вариант	Сигнал
1,6,11, 16,21,26	$y(x) := \sin(0{,}4 \cdot 2 \cdot \pi \cdot x) + \sin\left(0{,}9 \cdot 2 \cdot \pi \cdot x + \frac{\pi}{6}\right)$
2,7,12, 17,22,27	$y(x) := \cos\left(0{,}4 \cdot 2 \cdot \pi \cdot x + \frac{\pi}{12}\right) + \sin\left(0{,}9 \cdot 2 \cdot \pi \cdot x + \frac{\pi}{9}\right)$
3,8,13, 18,23,28	$y(x) := \sin\left(0{,}4 \cdot 2 \cdot \pi \cdot x + \frac{\pi}{10}\right) + \cos\left(0{,}7 \cdot 2 \cdot \pi \cdot x + \frac{\pi}{4}\right)$
4,9,14, 19,24,29	$y(x) := \sin(0{,}4 \cdot 2 \cdot \pi \cdot x) + \cos\left(0{,}7 \cdot 2 \cdot \pi \cdot x + \frac{\pi}{7}\right)$
5,10,15, 20,25,30	$y(x) := \cos\left(0{,}6 \cdot 2 \cdot \pi \cdot x + \frac{\pi}{7}\right) + \cos\left(0{,}9 \cdot 2 \cdot \pi \cdot x + \frac{\pi}{11}\right)$