МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Курсовая работа
по курсу «Численные методы»

Аппроксимация функций с использованием вейвлет-анализа.

Выполнил: *К. А. Полонский*
Группа: *М8О-408Б-20*
Преподаватель: Д. Е. Пивоваров
Дата:
Оценка:
Подпись:

Москва, 2023

Постановка задачи

Аппроксимация некоторой периодической функции с помощью дискретного вейвлет-преобразования (ДВП) и получение исходной функции с помощью обратного ДВП.

Теоретические сведения

Дискретное вейвлет-преобразование

Подобно разложению в ряд Фурье разложение в вейвлет-ряд ставит в соответствие функции непрерывного аргумента некоторую последовательность коэффициентов. В том случае, когда подлежащая разложению функция является дискретной (т. е. последовательностью чисел), получаемая последовательность коэффициентов называется дискретным вейвлет-преобразованием функции f(x). Например, если f(n) = $f(x_0 + n\Delta x)$ для некоторых значений x_0, Δx и n = 0, 1, 2, ..., M−1 коэффициенты разложения f(x) в вейвлет-ряд становятся коэффициентами прямого ДВП последовательности f(n):

$$W_\phi(j_0, k) = \frac{1}{\sqrt{M}} \sum_n f(n)\phi_{j_0,k}(n),$$

$$W_\psi(j, k) = \frac{1}{\sqrt{M}} \sum_n f(n)\psi_{j,k}(n) \quad \text{для} \quad j \geq j_0.$$

В формулах выше

$\phi_{j0,k}(n)$ и $\psi_{j,k}(n)$ — дискретные версии базисных функций $\phi_{j0,k}(x)$ и $\psi_{j,k}(x)$

$W_\phi(j_0, k)$ — коэффициенты приближения

$W_\psi(j\ , k)$ — коэффициенты деталей

Дополнением к прямому будет обратное ДВП вида

$$f(n) = \frac{1}{\sqrt{M}} \sum_n W_\phi(j_0, k)\phi_{j_0,k}(n) + \frac{1}{\sqrt{M}} \sum_{j=j_0}^{+\infty} \sum_k W_\psi(j, k)\psi_{j,k}(n).$$

Обычно полагают $j_0 = 0$ и выбирают число M так, чтобы оно было степенью двойки (т.е. $M = 2^J$). Для системы Хаара дискретные аналоги масштабирующих функций и вейвлет-функций (т.е. базисных функций), участвующих в преобразовании, соответствуют строкам M×M матрицы преобразования Хаара H.

Матрица H состоит из из базисных функций Хаара $h_k(z)$. Эти функции определены на непрерывном замкнутом интервале z ∈ [0, 1] при k = 0, 1, 2, ..., N−1, где $N = 2^n$.

Чтобы получить H, зададим целое k такое, что $k = 2^p + q - 1$,

где $0 \leq p \leq n - 1$,

$$q = \begin{cases} 0, 1 & \text{при } l = 0 \\ 1 \le q \le 2^p & \text{при } l \ne 0 \end{cases}$$

Тогда базисные функции Хаара будут

$$h_0(z) = h_{00}(z) = \frac{1}{\sqrt{N}}, \quad z \in [0, 1]$$

и

$$h_k(z) = h_{pq}(z) = \frac{2^{p/2}}{\sqrt{N}} \begin{cases} 1 & \text{при } (q-1)/2^p \le z \le (q-0.5)/2^p \\ -1 & \text{при } (q-0.5)/2^p \le z \le q/2^p \\ 0 & \text{в остальных случаях, } z \in [0, 1] \end{cases}$$

Само преобразование состоит из M коэффициентов, минимальный масштаб равен нулю, а максимальный равен J − 1.

Быстрое вейвлет-преобразование

Быстрое вейвлет-преобразование (БВП) представляет собой эффективный метод реализации вычислений дискретного вейвлет-преобразования (ДВП), который использует взаимосвязь между коэффициентами ДВП соседних масштабов. Метод БВП называют также иерархическим алгоритмом Малла.

ДВП сигнала x получают применением набора фильтров. Сначала сигнал пропускается через низкочастотный (low-pass) фильтр. Одновременно сигнал раскладывается с помощью высокочастотного (high-pass) фильтра. В результате получаются детализирующие коэффициенты (после ВЧ-фильтра) и коэффициенты аппроксимации (после НЧ-фильтра). Это разложение можно повторить несколько раз для дальнейшего увеличения частотного разрешения с дальнейшим прореживанием коэффициентов после НЧ и ВЧ-фильтрации.

Исходный код

```
import matplotlib.pyplot as plt
import numpy as np


def f1(x):
    return np.cos(2 * x)


def f2(x):
    if x <= 10:
        return np.sin(x) + np.cos(2 * x)
    else:
        return np.sin(x * 0.3) + np.cos(6 * x)
```

```python
def f3(x):
    return np.cos(x ** 2)


def f4():
    return np.cos(2 * np.pi * (2 ** np.linspace(2, 10, N)) *
np.arange(N) / 48000) + np.random.normal(0, 1, N) * 0.15


def discreteWaveletTransform(tabData):
    size = len(tabData) // 2
    cA = np.zeros(size)
    cD = np.zeros(size)

    for i, j in zip(range(0, len(tabData), 2), range(size)):
        c = 2 * (tabData[i] + tabData[i + 1]) / np.sqrt(N)
        cA[j] = c

    for i, j in zip(range(0, len(tabData), 2), range(size)):
        c = 2 * (tabData[i] - tabData[i + 1]) / np.sqrt(N)
        cD[j] = c

    return cA, cD


def waveletDeconstr(tabData, level=1):  # Returns [cA_n, cD_n,
cD_n-1, ..., cD2, cD1]
    coeffs = []
    a = tabData
    for i in range(level):
        a, d = discreteWaveletTransform(a)
        coeffs.append(d)
    coeffs.append(a)
    coeffs.reverse()

    return coeffs


def inverseDWT(a, d):
    res = []
    for i in range(len(a)):
        x = (a[i] + d[i]) * np.sqrt(N) / 4
        y = (a[i] - d[i]) * np.sqrt(N) / 4
```

```
        res.extend([x, y])
    return np.array(res)


def waveletReconstr(coeffs):
    a, ds = coeffs[0], coeffs[1:]
    for d in ds:
        a = inverseDWT(a, d)

    return a


scale = int(input())
N = int(2**scale)
level = 1
start = 0
stop = N
X = np.linspace(0, stop, N)
Y = [f1(x) for x in X]
# Y = f4()
c = waveletDeconstr(Y, level)
X1 = np.linspace(0, int(stop / (2**level)), int(N /
(2**level)))

figure, axis = plt.subplots(2, 2)
axis[0, 0].plot(X, Y)
axis[0, 0].set_title('Original signal')
axis[1, 0].plot(X1, c[0])
axis[1, 0].set_title(f'Approximation Coefficients,
level={level}')
axis[1, 1].plot(X1, c[1], c='orange')
axis[1, 1].set_title('Detail Coefficients')

inv = waveletReconstr(c)
axis[0, 1].plot(X, inv)
axis[0, 1].set_title('Reconstructed signal')
plt.show()
```

Результат

Масштаб scale = 5.

Аппроксимируемая функция: $f(x) = \cos(2x)$.

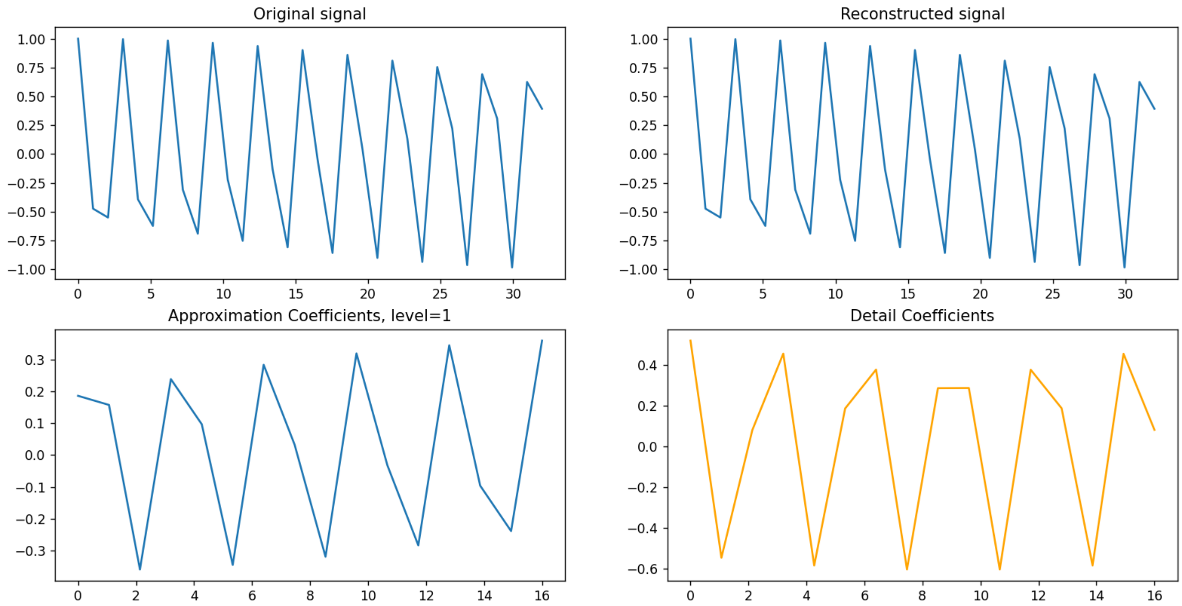

Рисунок 1. Графики 1-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 1.

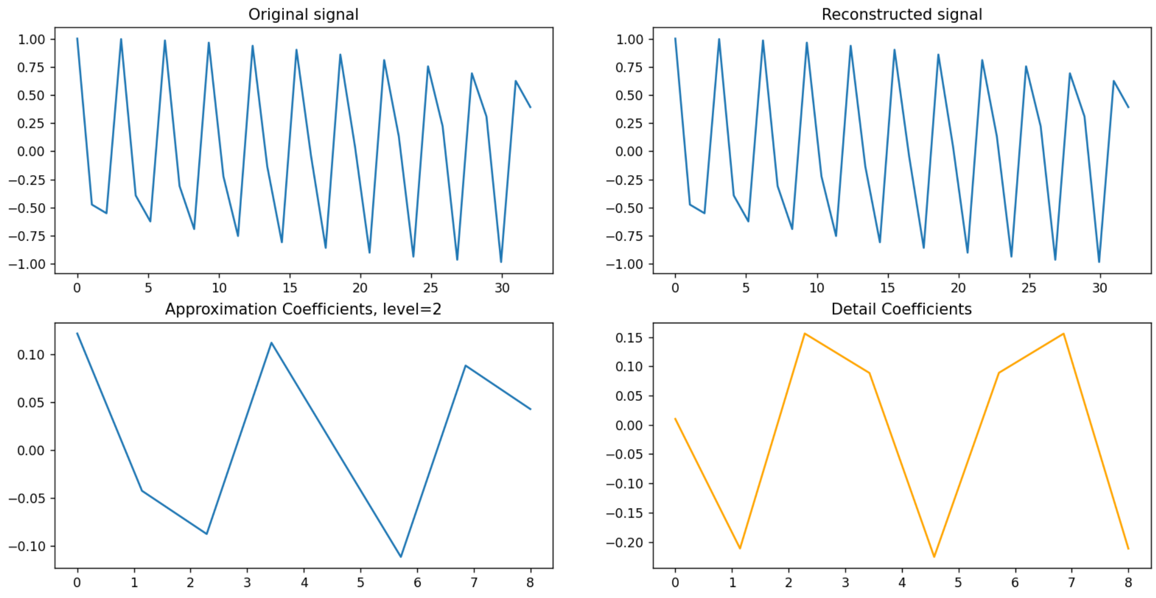

Рисунок 2. Графики 1-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 2.

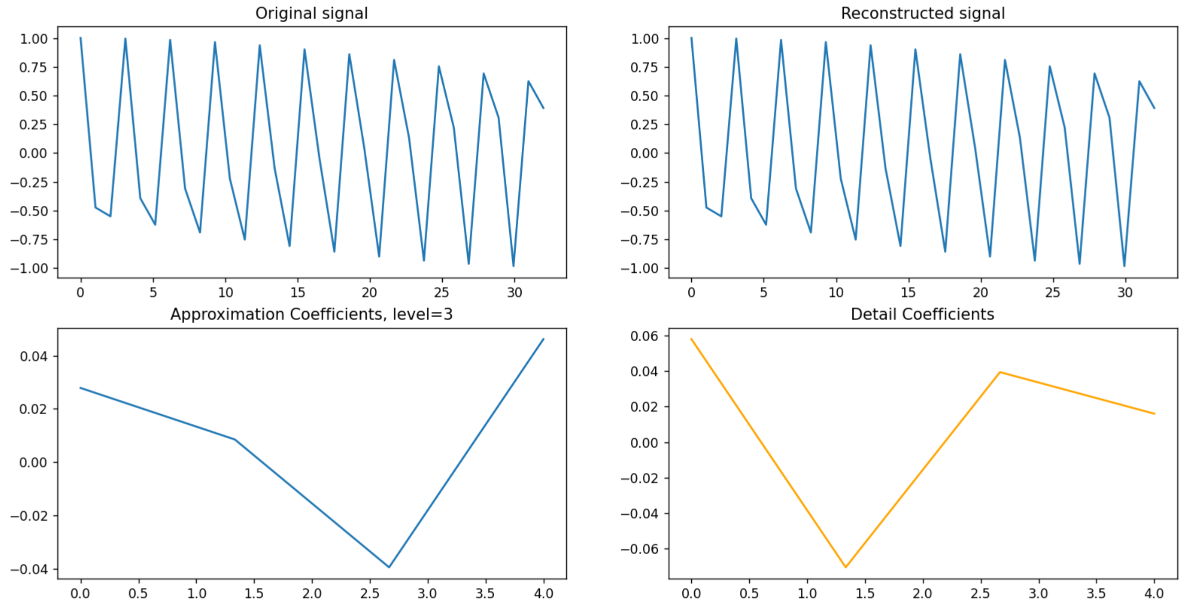

Рисунок 3. Графики 1-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 3.

Аппроксимируемая функция:
f(x) = sin(x) + cos(2x), x≤10; sin(0.3x) + cos(6x), иначе.

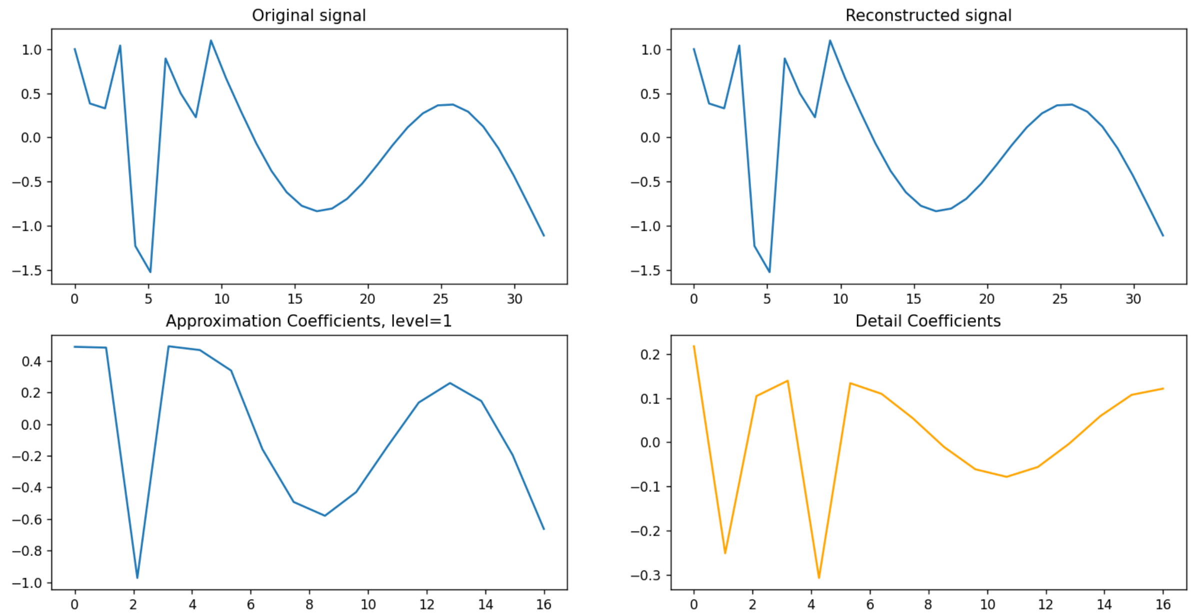

Рисунок 4. Графики 2-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 1.

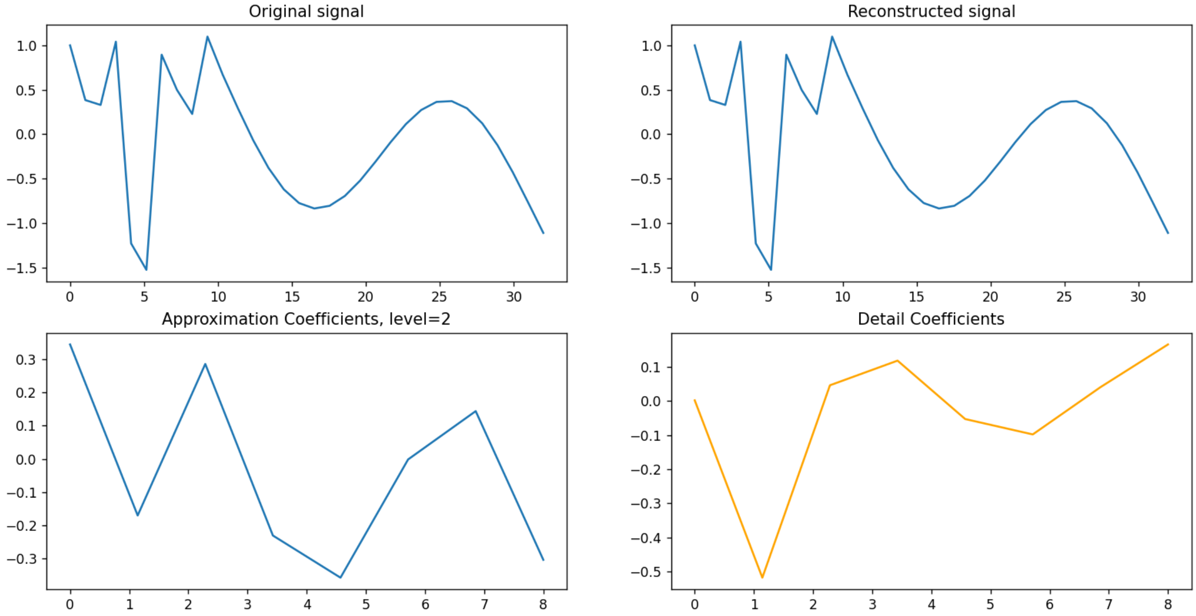

Рисунок 5. Графики 2-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 2.

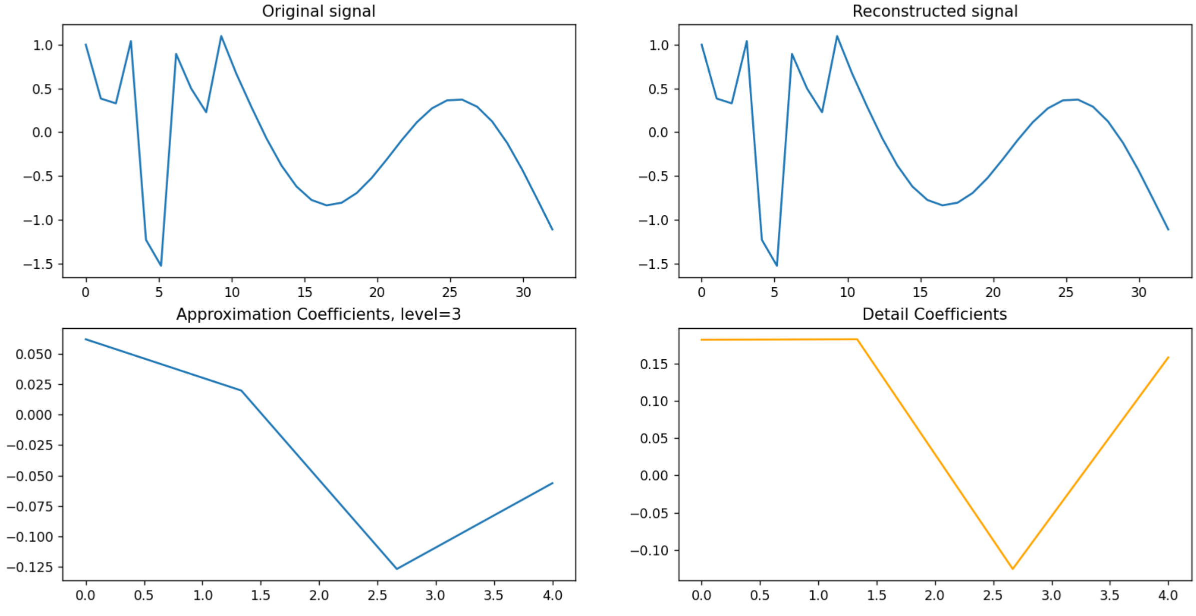

Рисунок 6. Графики 2-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 3.

Аппроксимируемая функция: $f(x) = \cos(x^2)$

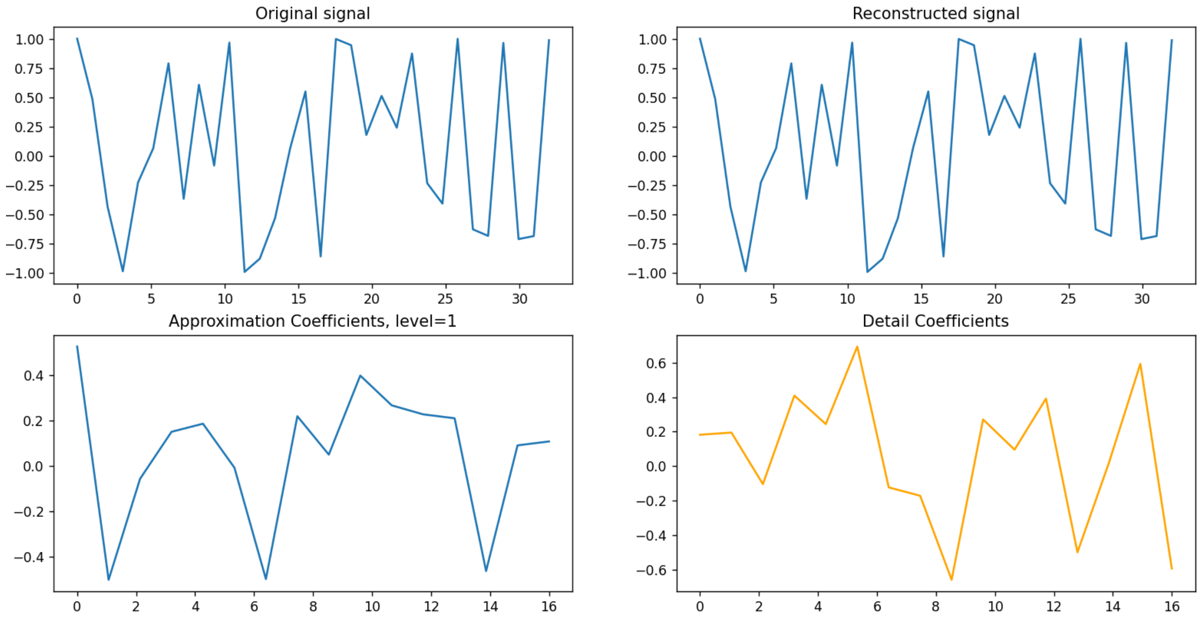

Рисунок 7. Графики 3-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 1.

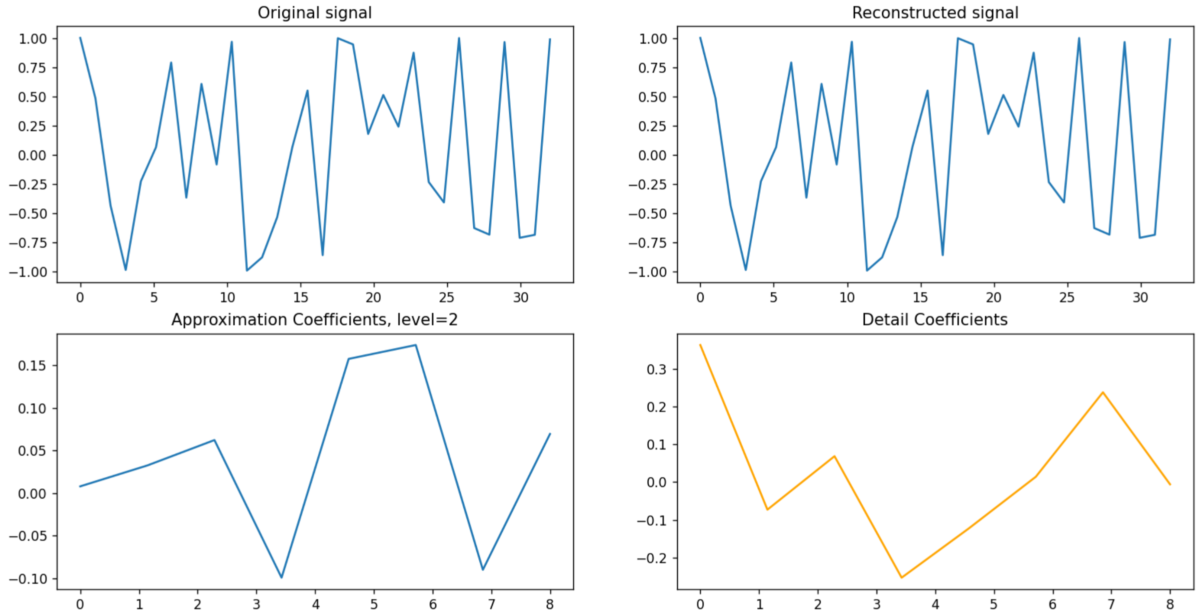

Рисунок 8. Графики 3-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 2.

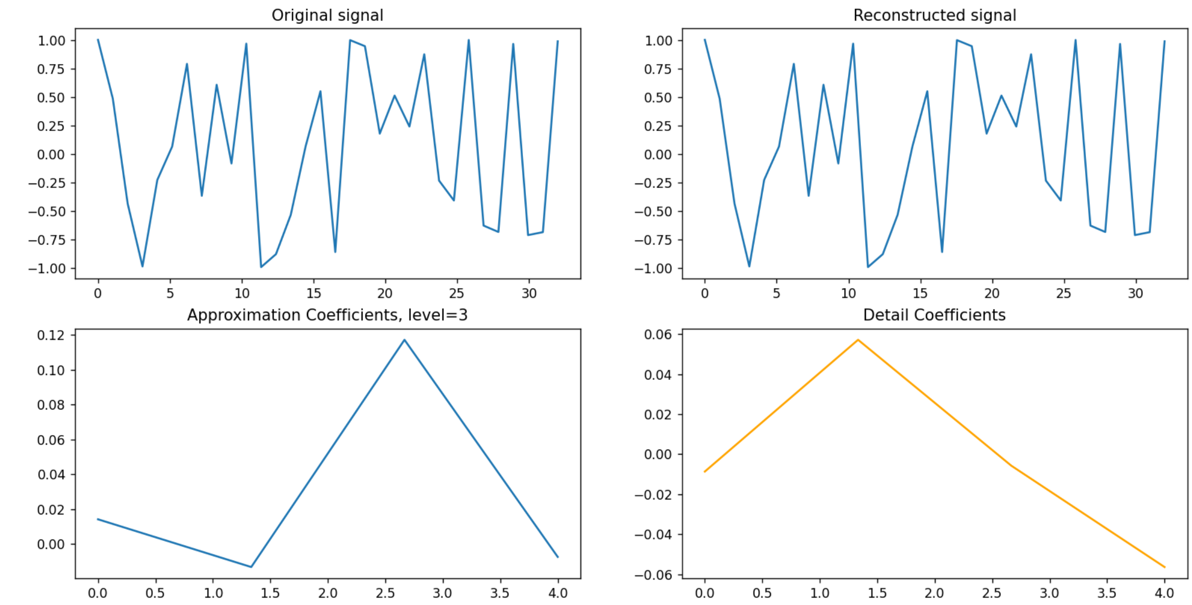

Рисунок 9. Графики 3-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 3.

Рассмотрим одно из применений вейвлет-анализа: очищение зашумлённого сигнала. Для наглядности поднимем scale до 7.

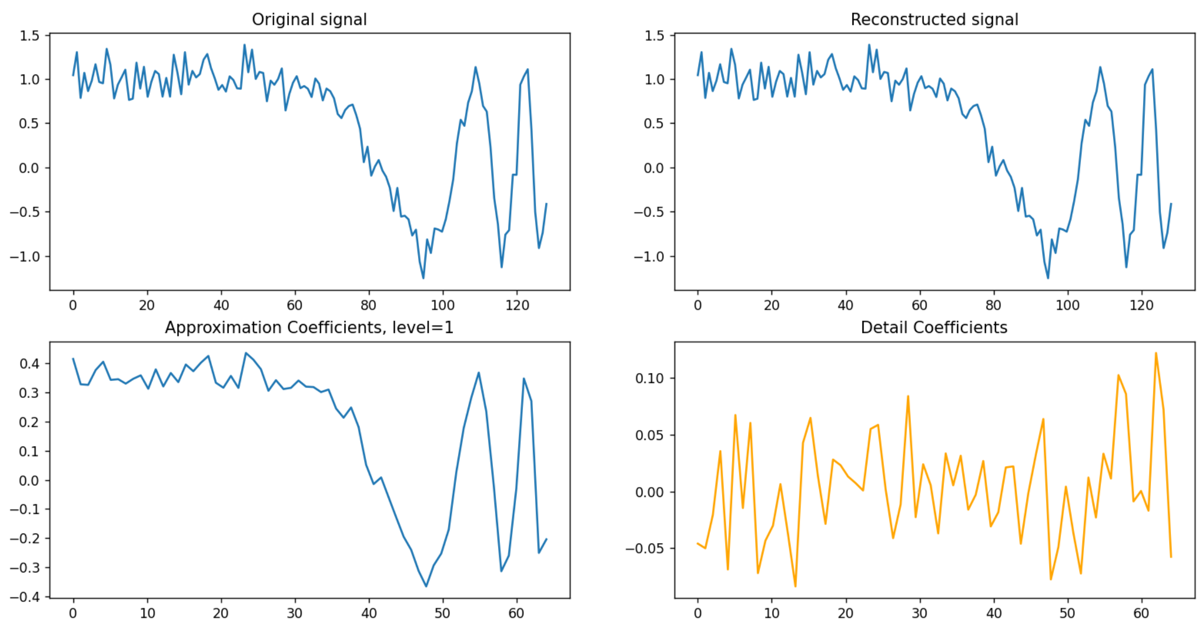

Рисунок 10. Графики 4-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 1.

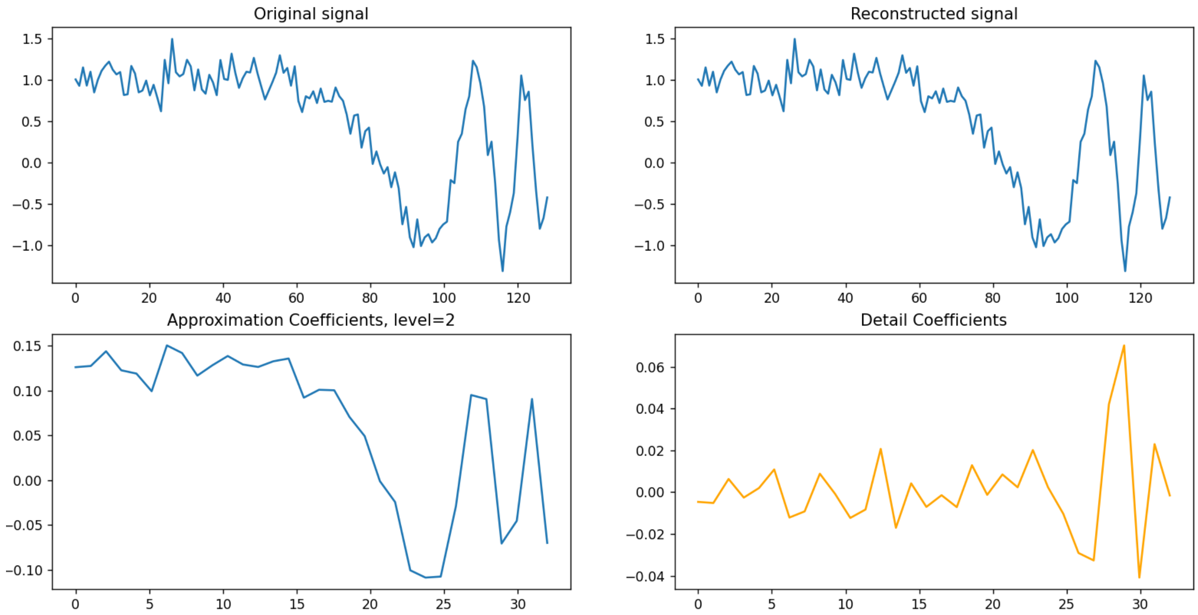

Рисунок 11. Графики 4-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 2.

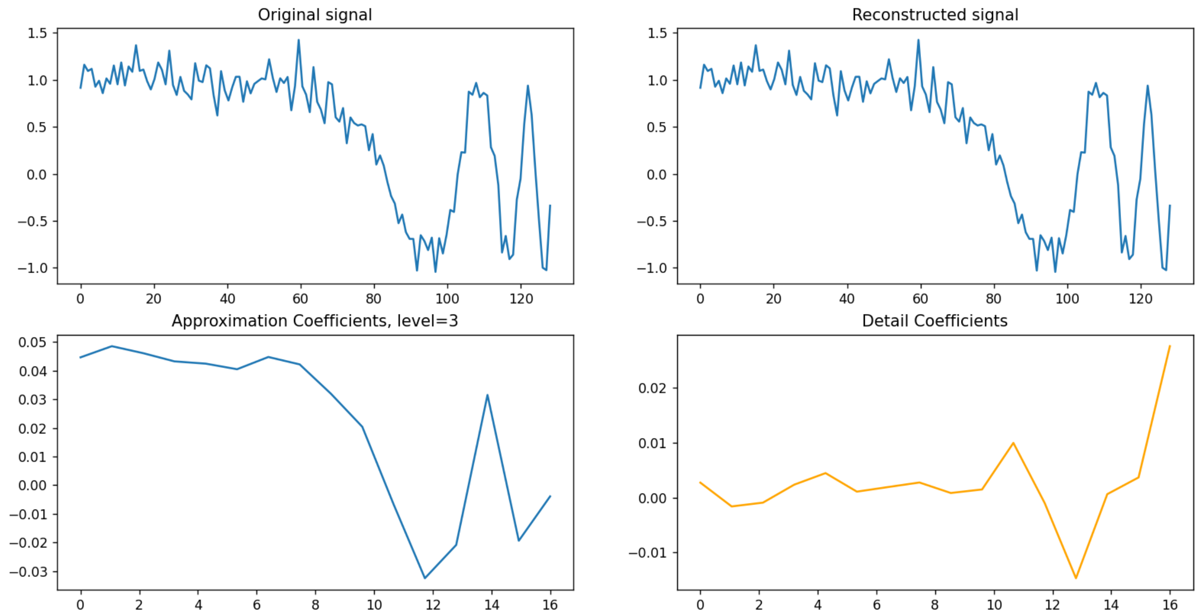

Рисунок 12. Графики 4-го сигнала, коэффициентов аппроксимации и детализации и реконструированного сигнала при уровне аппроксимации равном 3.

Выводы

В процессе выполнения курсовой работы я познакомился с понятием вейвлет-анализа и реализовал быстрое преобразование Хаара. Задача вейвлет-анализа функций имеет широкое применение не только в обработке сигналов, но и в анализе изображений и сжатии данных, что показывает её универсальность и важность.

Список литературы

1. PyWavelets [Электронный ресурс] URL: https://pywavelets.readthedocs.io/en/latest/index.html (дата обращения: 08.01.2023).
2. Вейвлет — анализ. Основы. [Электронный ресурс] URL: https://habr.com/ru/articles/449646/ (дата обращения: 07.01.2023).
3. Вейвлет — анализ. Часть 1. [Электронный ресурс] URL: https://habr.com/ru/articles/451278/ (дата обращения: 07.01.2023).
4. Википедия [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B9%D0%B2%D0%BB%D0%B5%D1%82_%D0%A5%D0%B0%D0%B0%D1%80%D0%B0 (дата обращения: 08.01.2023).
5. Introduction to Wavelet Transform using Python [Электронный ресурс] URL: https://scicoding.com/introduction-to-wavelet-transform-using-python/ (дата обращения: 08.01.2023).
6. Youtube-канал selfedu [Электронный ресурс] URL: https://www.youtube.com/@selfedu_rus (дата обращения: 07.01.2023).