Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рыбинский государственный авиационный технический университет

имени П.А. Соловьева»

Институт «Информационные технологии и системы управления»

Кафедра математического и программного обеспечения электронных вычислительных средств

**ОТЧЕТ**

по дисциплине:

«Системы цифровой обработки сигналов»

на тему:

«Wavelet–преобразование (Wavelet–представление)»

Студенты группы ПИМ-24 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ананьев Г.Е.,

*(Код) (Подпись, дата) (Фамилия И. О.)*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Власенков А.Д.

*(Подпись, дата) (Фамилия И. О.)*

Руководитель к.т.н., доц. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Грызлова Т.П.

*(Уч. степень, звание) (Подпись, дата) (Фамилия И. О.)*

Рыбинск 2025

Содержание

[Задание на лабораторную работу 3](#_Toc192094248)

[1. Результаты тестирования программы на последовательности 4](#_Toc192094249)

[2. Результаты выполнения лабораторной работы 5](#_Toc192094250)

[2.1 Сигнал из файла N1.txt 7](#_Toc192094251)

[2.2 Сигнал из файла N31.txt 9](#_Toc192094252)

[2.3 Сигнал из файла rl.txt 11](#_Toc192094253)

[Заключение 12](#_Toc192094254)

[Приложение А. Исходный код программного модуля main.py по вычислению Wavelet-коэффициентов и их визуализации по уровням 13](#_Toc192094255)

Цель работы: реализация алгоритма Wavelet-разложения заданных сигналов в базисе Хаара и визуализация Wavelet-представления сигналов в пространстве уровень–время.

# Задание на лабораторную работу

1. Написать программу, принимающую последовательность цифровых отсчетов

или последовательность произвольной длины , ограничивая обработку до отсчетов.

Программа должна выполнять Wavelet-преобразование в базисе Хаара. Число уровней разложения при представлении результатов:

Wavelet-коэффициенты должны быть привязаны ко времени и сохранены в структурах или файлах для последующего анализа.

1. Тестировать программу на последовательности , приведенной в примере (рис. 1). Результат показать преподавателю.
2. Выполнить Wavelet-разложение на заданных сигналах.
3. Моделировать сигнал, заданный формулой, сохранить в файле.
4. В отчете представить графические представления Wavelet-коэффициентов для модельного сигнала и сигнала из файла, выданного преподавателем.

Входные данные: текстовые файлы реализаций Lab2Data.txt, N1.txt, N31.txt и rl.txt.

Выходные данные:

1. Текстовые файлы Wavelet-коэффициентов;
2. Графические представления Wavelet-коэффициентов для модельного сигнала Lab2Data.txt и сигналов из файлов N1.txt, N31.txt и rl.txt.

# 1. Результаты тестирования программы на последовательности , приведенной в примере (рис. 1).

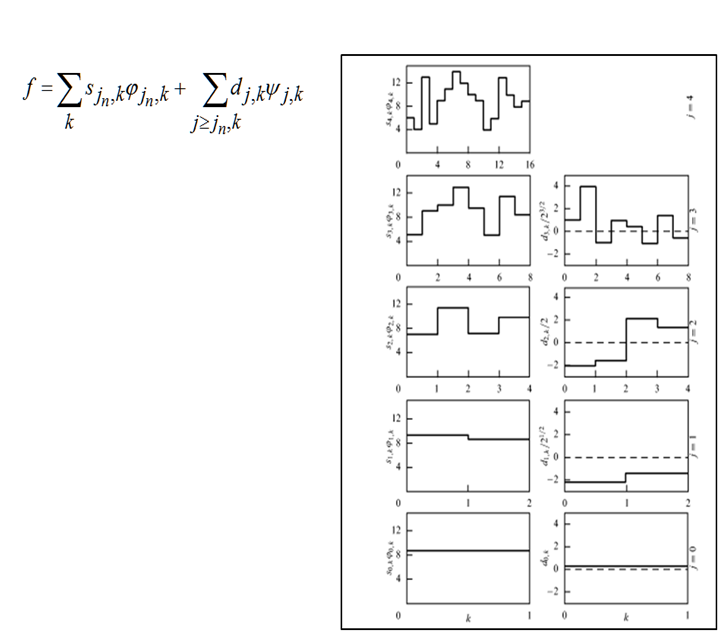


Рисунок 1 – Последовательность отсчетов, ее аппроксимации и детали в базисе Хаара

Результат тестирования работы программы на последовательности

представлен на рисунке 2.

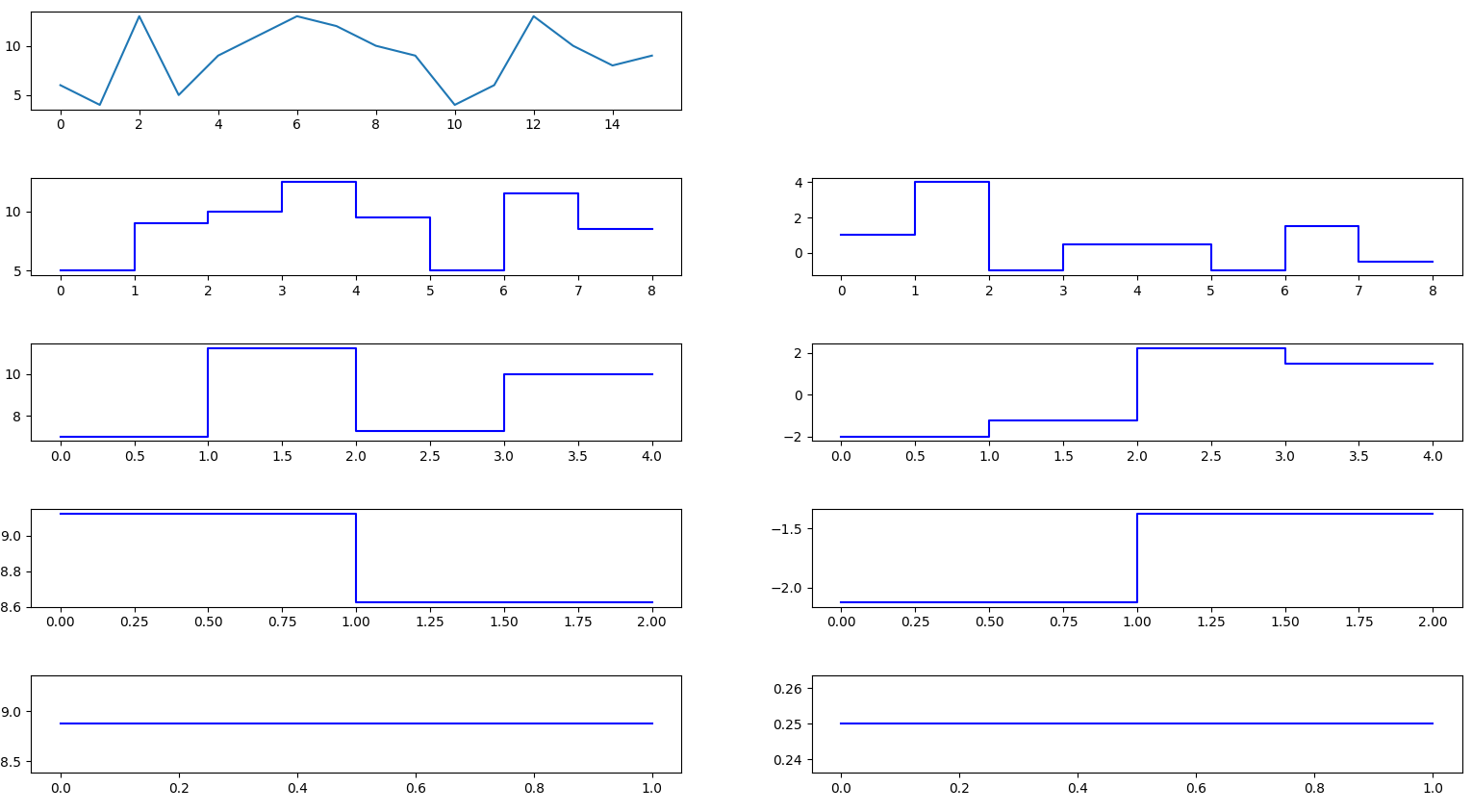


Рисунок 2 – Тестирование программы на последовательности

# 2. Результаты выполнения лабораторной работы

В ходе выполнения лабораторной работы были взяты сигналы N1.txt, N31.txt и rl.txt. Отобразим сигналы из файлов на графике ДСК.

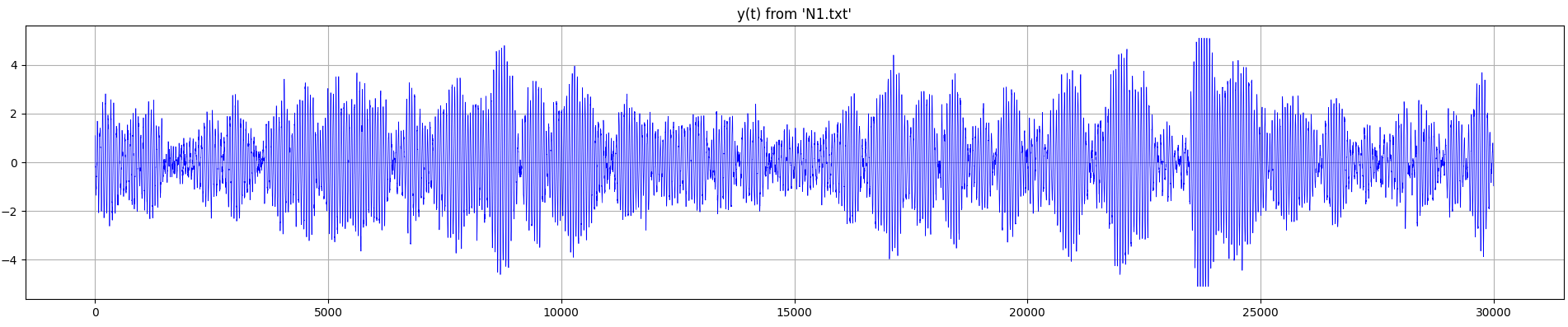


Рисунок 4 – Сигнал из файла «N1.txt»

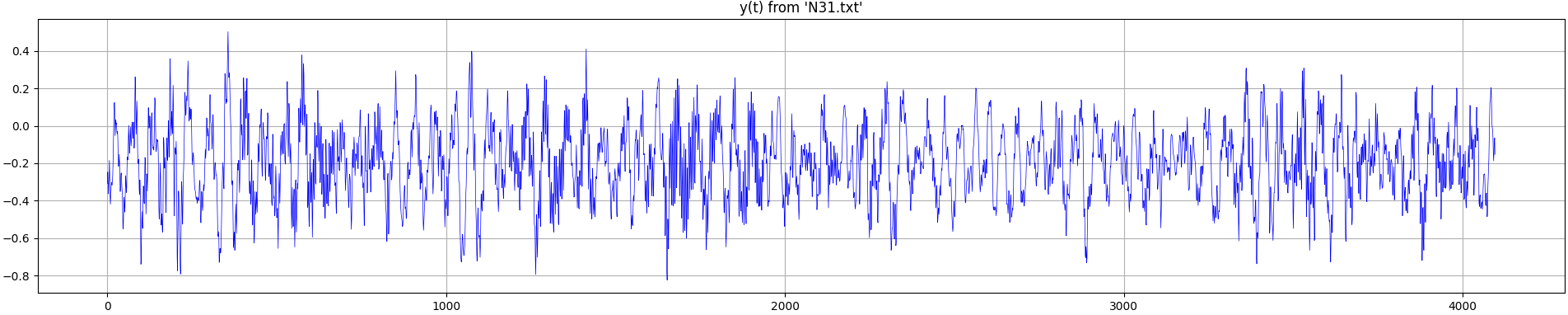


Рисунок 5 – Сигнал из файла «N31.txt»

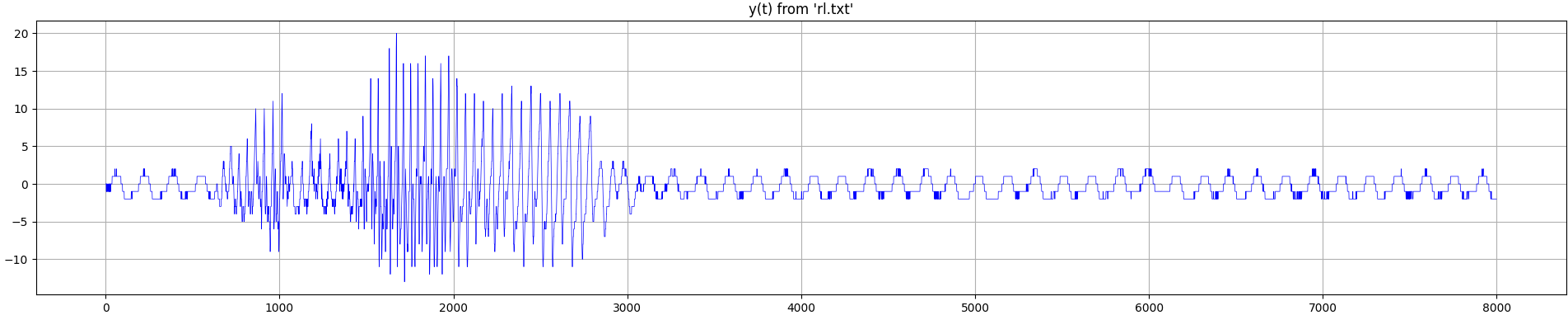


Рисунок 6 – Сигнал из файла «rl.txt»

По заданию необходимо реализовать алгоритм Wavelet-разложения заданных сигналов в базисе Хаара и визуализировать Wavelet-представления сигналов в пространстве уровень – время. В таблице 1 представлены исходные данные для визуализации сигналов, заданных в файлах.

Таблица 1 – Исходные данные для визуализации сигналов

|  |  |  |
| --- | --- | --- |
| Файл сигнала | Количество отсчетов N | Частота дискретизации fd (кГц) |
| N1.txt | 30000 | 16,384 |
| N31.txt | 4096 | 4,096 |
| rl.txt | 8000 | 4,096 |

При расчете wavelet-коэффициентов использовались следующие формулы:

Формула коэффициентов средних значений:

Формула коэффициентов разности:

, .

На ЯП Python 3.10 была написана программа, которая выполняет Wavelet-преобразование в базисе Хаара для уровней. Для расчетов использовалась библиотека NumPy, а для визуализации – библиотека matplotlib. Исходный код программного модуля приведен в Приложении А.

# 2.1 Сигнал из файла N1.txt

Графические представления Wavelet-коэффициентов средних значений для сигнала из файла N1.txt представлены на рисунке 7.

Графические представления Wavelet-коэффициентов разностей для сигнала из файла N1.txt представлены на рисунке 8.

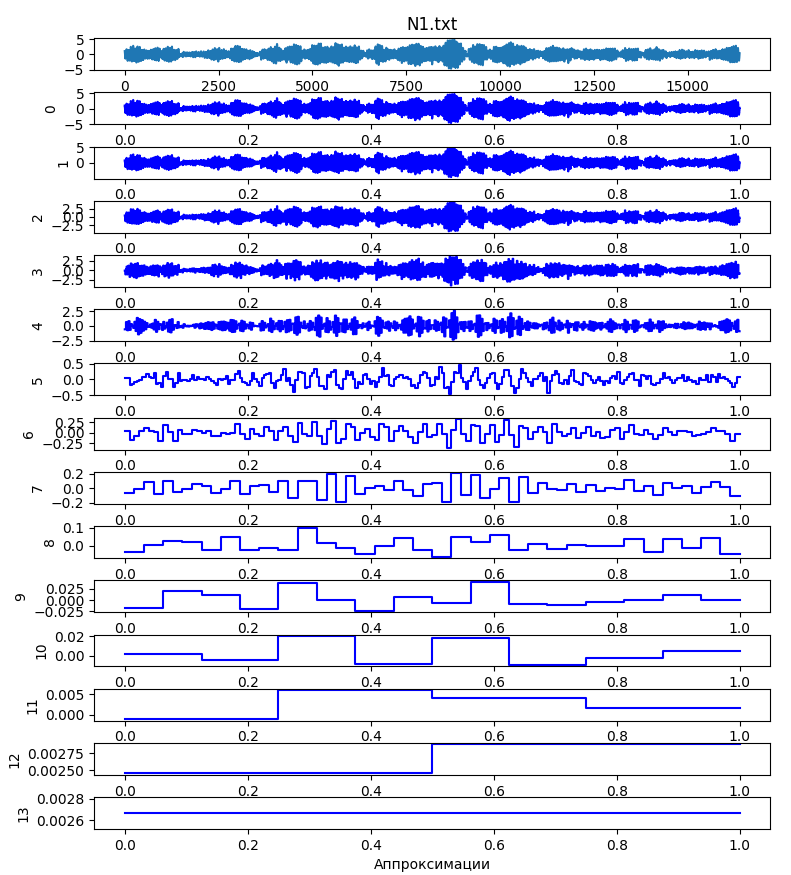


Рисунок 7 – Аппроксимации сигнала из файла N1.txt

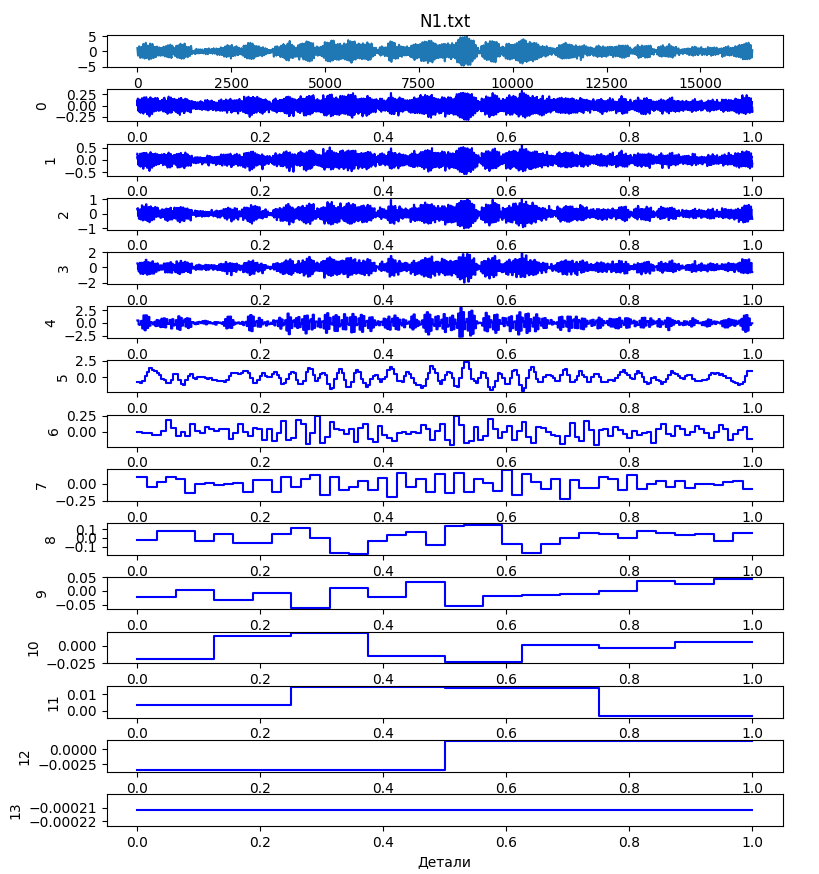


Рисунок 8 – Детали сигнала из файла N1.txt

# 2.2 Сигнал из файла N31.txt

Графические представления Wavelet-коэффициентов средних значений для сигнала из файла N31.txt представлены на рисунке 9.

Графические представления Wavelet-коэффициентов разностей для сигнала из файла N31.txt представлены на рисунке 10.

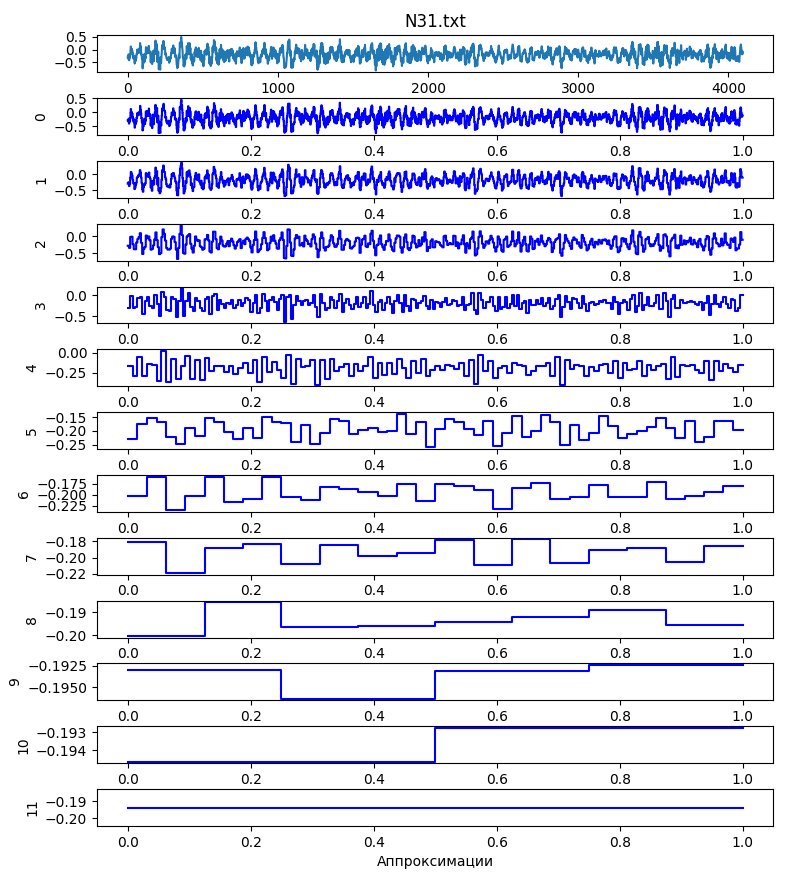


Рисунок 9 – Аппроксимации сигнала из файла N31.txt

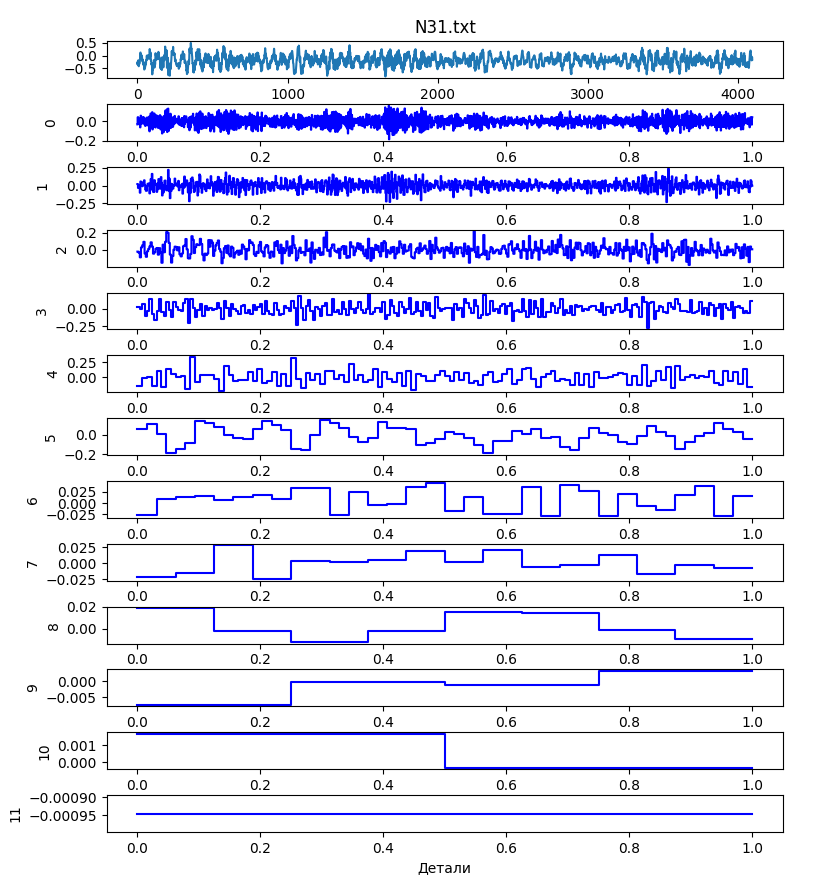


Рисунок 10 – Детали сигнала из файла N31.txt

# 2.3 Сигнал из файла rl.txt

Графические представления Wavelet-коэффициентов средних значений для сигнала из файла rl.txt представлены на рисунке 11.

Графические представления Wavelet-коэффициентов разностей для сигнала из файла rl.txt представлены на рисунке 12.

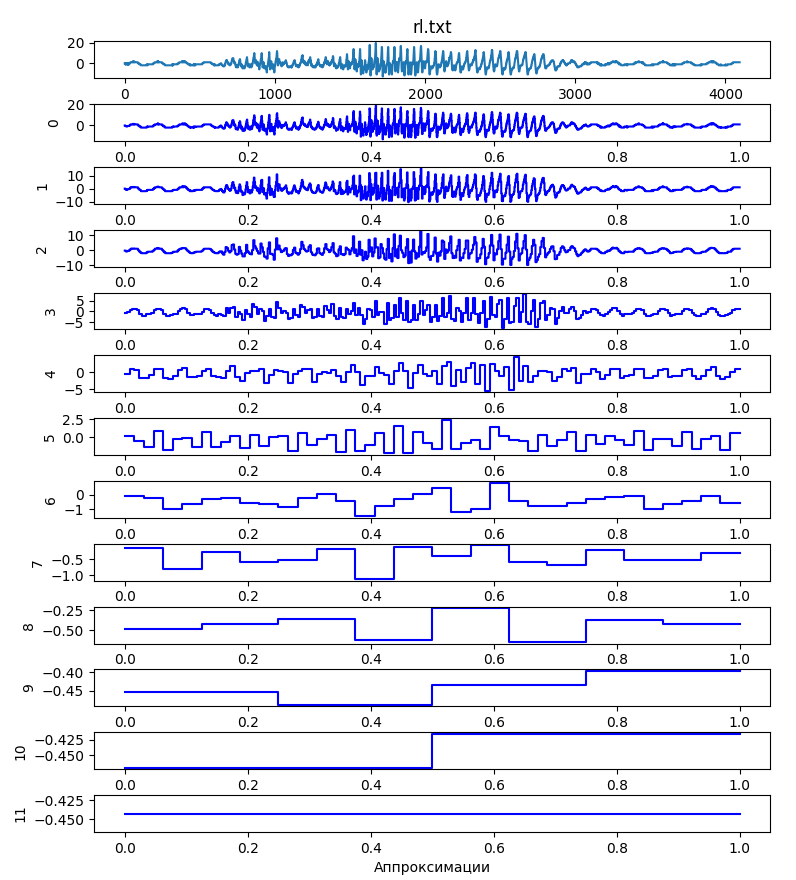


Рисунок 11 – Аппроксимации сигнала из файла rl.txt

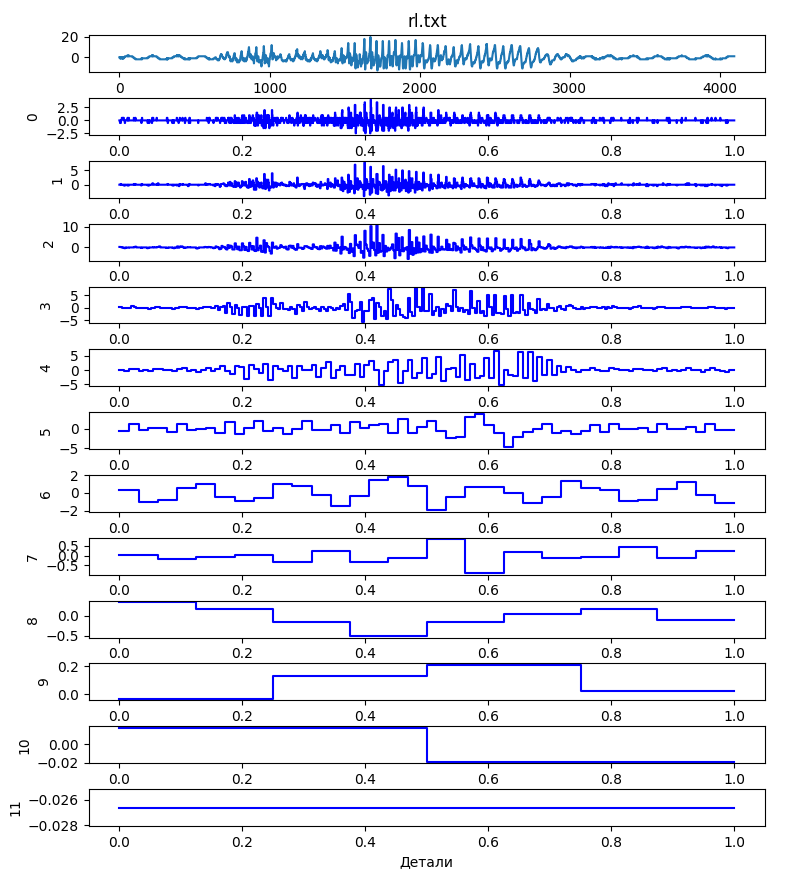


Рисунок 12 – Аппроксимации сигнала из файла rl.txt

# Заключение

В ходе выполнения лабораторной работы была составлена программа, которая выполняет Wavelet-преобразование. Были получены значения Wavelet-коэффициентов для разных уровней разложения сигналов, а также построены графики для уровней, где N – длина сигнала.

# Приложение А. Исходный код программного модуля main.py по вычислению Wavelet-коэффициентов и их визуализации по уровням

import matplotlib.pyplot as plt

import numpy as np

import math

import os

print('full name of file:')

path = input()

filename = os.path.splitext(os.path.basename(path))[0]

approx\_file = f"{filename}\_wavelet\_approx.txt"

detail\_file = f"{filename}\_wavelet\_details.txt"

with open(path) as file:

data = [float(line.strip().replace(',', '.')) for line in file]

print(f'length of signal: {len(data)}')

data = data[: 2 \*\* math.floor(math.log2(len(data)))]

print(f'length of Ns sequence: {len(data)}')

print('fd in Hz:')

fd = input()

N = len(data) # длина реализации

dt = 1 / int(fd) # шаг дискретизации

def wavelet():

tempWaveletAprxLevels = [data]

tempWaveletDetailLevel = [data]

curLevelInd = 0

while len(tempWaveletAprxLevels[curLevelInd]) >= 2:

tempWaveletAprxLevels.append([])

tempWaveletDetailLevel.append([])

for j in range(math.floor(len(tempWaveletAprxLevels[curLevelInd]) / 2)):

tempWaveletAprxLevels[curLevelInd + 1].append(

(tempWaveletAprxLevels[curLevelInd][2 \* j] + tempWaveletAprxLevels[curLevelInd][2 \* j + 1]) / 2)

tempWaveletDetailLevel[curLevelInd + 1].append(

(tempWaveletAprxLevels[curLevelInd][2 \* j] - tempWaveletAprxLevels[curLevelInd][2 \* j + 1]) / 2)

curLevelInd += 1

return tempWaveletAprxLevels[1:], tempWaveletDetailLevel[1:]

def save\_wavelet(waveletApproximations, waveletDetails):

with open(approx\_file, 'w') as f:

for level in waveletApproximations:

f.write("\n".join(map(str, level)) + "\n\n")

with open(detail\_file, 'w') as f:

for level in waveletDetails:

f.write("\n".join(map(str, level)) + "\n\n")

return approx\_file, detail\_file

def load\_wavelet():

def load\_file(filename):

with open(filename, 'r') as f:

content = f.read().strip().split("\n\n")

return [list(map(float, level.split("\n"))) for level in content]

return load\_file(approx\_file), load\_file(detail\_file)

def showWavelet(waveletApproximations, waveletDetails):

plt.figure(figsize=(19.2, 10.8))

plt.subplots\_adjust(hspace=0.7)

plt.subplot(len(waveletApproximations) + 1, 2, 1)

plt.plot(np.arange(len(data)), data)

plt.title(path)

plt.subplot(len(waveletApproximations) + 1, 2, 2)

plt.plot(np.arange(len(data)), data)

plt.title(path)

for i in range(len(waveletApproximations)):

plt.subplot(len(waveletApproximations) + 1, 2, 2 \* (i + 1) + 1)

level\_time = np.linspace(0, N \* dt, len(waveletApproximations[i]) + 1)

plt.step(level\_time, [waveletApproximations[i][0]] + waveletApproximations[i], color='b',

label='appr L' + str(i))

plt.subplot(len(waveletApproximations) + 1, 2, 2 \* (i + 1) + 2)

plt.step(level\_time, [waveletDetails[i][0]] + waveletDetails[i], color='b', label='detail L' + str(i))

for i in range(len(waveletApproximations)):

plt.subplot(len(waveletApproximations) + 1, 2, 2 \* (i + 1) + 1)

plt.ylabel(f"{i}")

plt.subplot(len(waveletApproximations) + 1, 2, 2 \* (i + 1) + 2)

plt.ylabel(f"{i}")

plt.show()

waveletApproximations, waveletDetails = wavelet()

save\_wavelet(waveletApproximations, waveletDetails)

loadedApproximations, loadedDetails = load\_wavelet()

showWavelet(loadedApproximations, loadedDetails)