

Лабораторные работы кафедры ВТ ИКТИБ ЮФУ для курсов «Проблемно-ориентированные методы и средства цифровой обработки сигналов», «Специализированные методы и средства цифровой фильтрации»

Лабораторная работа №2.

Процедурная реализация быстрого преобразования Фурье на Python

Цель работы.

Лабораторная работа №2 нацелена на изучение принципов процедурной реализации алгоритма быстрого дискретного преобразования Фурье.

Подготовка к работе.

Лабораторная работа предполагает реализацию программы выполнения быстрого дискретного преобразования Фурье на языке Python с использованием IDE PyCharm или Visual Studio Code (VSCode).

Ход работы:

Задание №1: написать программу на языке Python, реализующую заданный алгоритм БПФ и работающую с комплексными числами в формате floating point.

1. Выполнить построение дискретного сигнала с использованием языка Python, как это было сделано в задании №2 пп.1 лабораторной работы №1:

$$x_n = \sum_{k=1}^7 A_k \cdot \sin(2\pi \cdot f_k \cdot t_n)$$

Если $\max(f_n)$ из варианта лабораторного задания больше 30, определить частоту дискретизации $fd = 110$ КГц, иначе $fd = 75$ КГц. Количество семплов $N = 16$.

2. Выполнить прямое дискретное преобразование Фурье с помощью функции `fft` библиотеки `scipy` или `numpy`, как это было сделано в задании №2 пп.4 лабораторной работы №1.
3. Написать функцию, реализующую базовую операцию БПФ (бабочку) по формулам для вида прореживания, выбранного согласно варианту лабораторного задания:

С прореживанием по времени:

С прореживанием по частоте:

$$\begin{cases} \dot{A}_{i+1} = \dot{A}_i + \dot{B}_i \cdot \dot{W}_k; \\ \dot{B}_{i+1} = \dot{A}_i - \dot{B}_i \cdot \dot{W}_k; \\ i = 0, 1, \dots, (N-1); \quad k = 0, 1, \dots, (N/2-1), \end{cases} \quad \begin{cases} \dot{A}_{i+1} = \dot{A}_i + \dot{B}_i; \\ \dot{B}_{i+1} = (\dot{A}_i - \dot{B}_i) \cdot \dot{W}_k; \\ i = 0, 1, \dots, (N-1); \quad k = 0, 1, \dots, (N/2-1), \end{cases}$$

Функция должна принимать на вход три аргумента: `A`, `B`, `W` и возвращать два значения: `A_out`, `B_out`. При этом входные аргументы не должны являться массивами, а лишь их значениями.

Индексация входных массивов `X` и `W` будет выполняться вне тела функции.

4. Рассчитать коэффициенты БПФ. Помнить, по формуле:

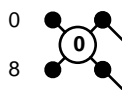
$$W_N = e^{-i2\frac{\pi}{N}}$$

```
import math
import cmath

W = []

# определим лямбда-функцию с входным аргументом x
W_func = lambda x: (math.e ** -(cmath.sqrt(-1) * 2 * (math.pi / N) * x))
for j in range(N//2):
    W.append(W_func(j))
print(W)
```

5. Выполнить обход графа БПФ, выбранного согласно варианту лабораторного задания.



Каждая вершина графа является одной бабочкой. Слева в вершину поступает два входных отсчета: верхний – отсчет `A`, нижний –

отсчет B . Числа перед входными портами обозначают индекс элемента в массиве исходного дискретного сигнала x_n . Цифра внутри кружка обозначает индекс коэффициента W . Таким образом, на бабочку, приведенную на рисунке в качестве аргумента A подается $x[0]$, а в качестве аргумента B – $x[8]$, а в качестве аргумента W – $W[0]$.

Обход графа представляет собой выполнение всех базовых операций БПФ в заданном графе. Порядок операций определяется готовностью данных промежуточных вычислений. Рекомендуется выполнять операции последовательно сверху вниз в рамках одного слоя графа, и только после выполнения всех 8 операций, переходить к следующему слою.

Отметим, что имеется возможность реализовать циклическое выполнение базовых операций (бабочек), для чего важно правильно формировать массивы входных и промежуточных данных. При реализации циклов стоит использовать бит-инверсный порядок адресов данных (см. пояснение в лекции или в Интернете).

6. Сравнить полученные в результате преобразования значения со значениями, полученными после применения библиотечной функции Python в пп.2 данной работы.

Задание №2: написать программу на языке Python, реализующую заданный алгоритм БПФ и работающую с действительными числами в формате `int`.

1. Исходный сигнал x_n , полученный в пп.1 задания №1 данной работы, конвертировать в целые числа, отбросив все знаки после запятой с помощью функции `int()`.
2. Масштабировать значения коэффициентов W , отобразив их на разрядную сетку, заданную вариантом лабораторного задания. Для этого рассчитать диапазон разрядной сетки по формуле: $D = 2^q$, где q – требуемая разрядность данных, а шаг разрядной сетки равен $1/D$. Так как работа ведется над числами со знаком (signed), старший бит от общей разрядности

отведен под знак. В связи с этим, масштабировать коэффициенты необходимо согласно формуле: $W_{scale} = W * D/2$.

Так как $W[0] = 1$, после масштабирования коэффициентов $W_{scale}[0]$ выйдет за рамки допустимого диапазона. В связи с этим, необходимо дополнительно сделать следующую операцию:

```
Wscale[0] = Wscale[0]-1
```

3. Переписать функцию, реализующую бабочку БПФ для работы с действительными числами. Для этого необходимо реализовать формулы операций комплексного умножения (1) и комплексного сложения (2):

$$C1 * C2 = (a + jb) * (c + jd) = (a * c - b * d) + j(a * d + b * c) \quad (1)$$

$$C1 + C2 = (a + jb) + (c + jd) = (a + c) + j(b + d) \quad (2)$$

В данных формулах два комплексных числа $C1$ и $C2$ представлены в виде: $C1 = (a + jb)$, $C2 = (c + jd)$. Для того, чтобы работать с действительной и мнимой частями комплексного числа можно использовать функции `real` и `imag`:

```
a = C1.real
b = C1.imag
c = C2.real
d = C2.imag
res_mul_real = a*c - b*d
...
```

Для объединения результата операции в комплексное число можно использовать функцию `res_mul = complex(res_mul_real, res_mul_imag)`.

4. Так как после каждой операции **целочисленного умножения** разрядность результата `res_mul` увеличивается вдвое, необходимо **обязательно** выполнять масштабирование путем округления их младшей части до середины разрядной сетки $D/2$:

```
res_mul_scale = res_mul / (D/2)
```

Масштабирование результата операции сложения выполнять не нужно.

5. Выполнить обход графа алгоритмом, разработанным в пп.5 задания №1 данной работы. Это можно реализовать лишь подменой имени функции,

разработанной в пп.3 задания №1 на имя функции, разработанной в пп.3,4 задания №2.

6. Сравнить полученные в результате преобразования значения со значениями, полученными в задании №1 данной работы.
7. Сделать вывод о точности целочисленной реализации алгоритма БПФ по сравнению с реализацией в формате floating point.

Варианты заданий к лабораторной работе №2

Варианты лабораторного задания выбираются согласно номеру студента в общем списке группы. Нечетный номер реализует БПФ с помощью библиотеки `scipy`, четный – с помощью `numpy`.

В таблице приведены значения частот и амплитуд для генерации модельного сигнала, схема бабочки алгоритма БПФ и разрядность выполняемых операций.

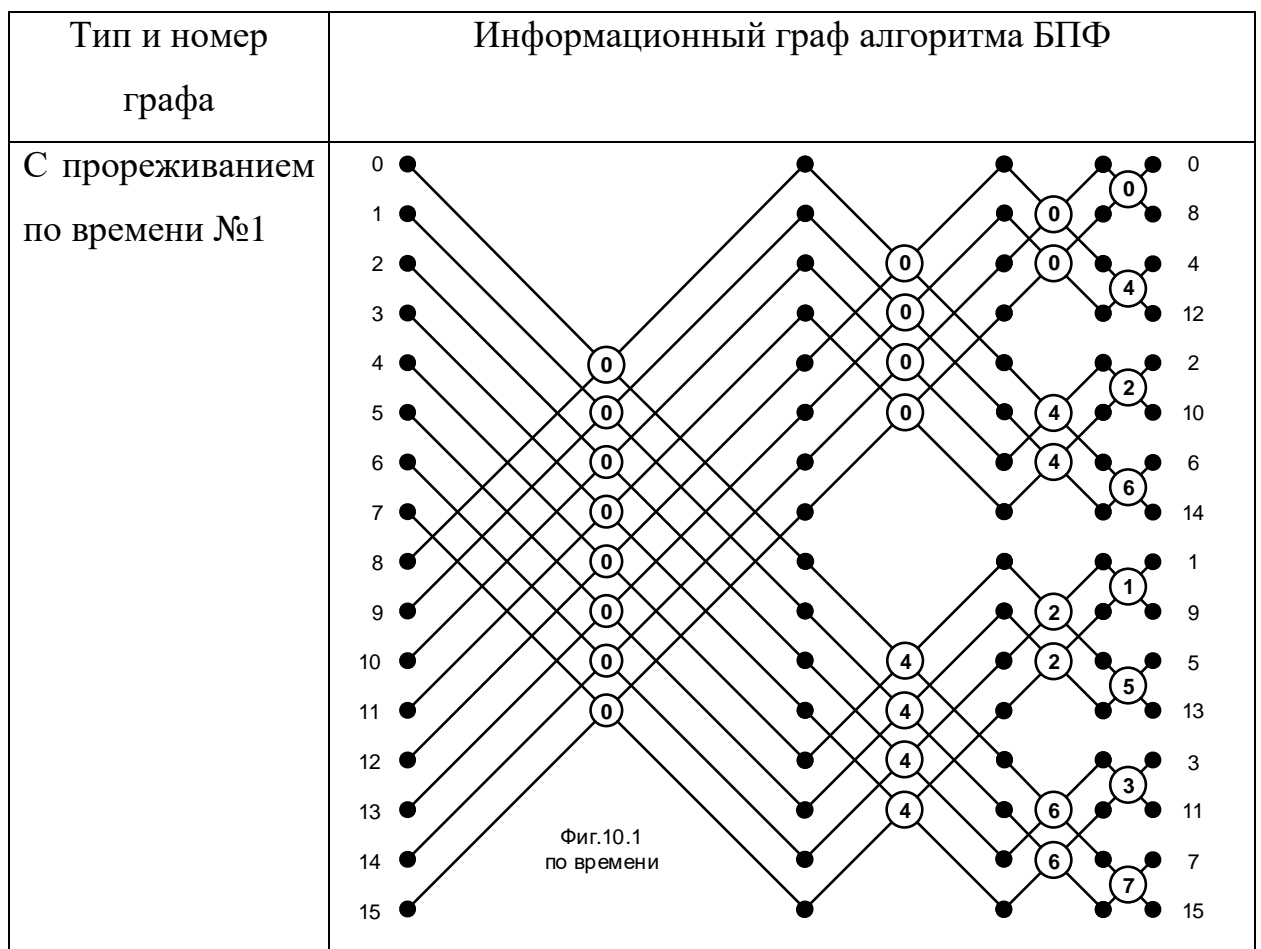
Таблица 1 – варианты лабораторных заданий

№		1	2	3	4	5	6	7	Схема бабочки:	Разрядность:
1	F, кГц	0,5	1	2	5	7	9	12	С прореживанием по времени 1	8 бит
	A	1	5	3	7	3	2	1		
2	F, кГц	3	5	6	9	12	15	18	С прореживанием по частоте 1	9 бит
	A	7	9	8	12	7	4	6		
3	F, кГц	0,9	1	2	9	12	15	18	С прореживанием по времени 2	10 бит
	A	2	5	3	12	7	4	6		
4	F, кГц	3	5	6	8	9	10	12	С прореживанием по частоте 2	11 бит
	A	7	9	8	7	3	2	5		
5	F, кГц	0,5	1	2	9	12	14	17	С прореживанием по времени 3	12 бит
	A	1	5	3	12	15	4	6		
6	F, кГц	2	4	5	7	9	12	15	С прореживанием по частоте 3	13 бит
	A	11	15	12	13	21	10	9		
7	F, кГц	9	11	15	17	21	29	35	С прореживанием по времени 1	14 бит
	A	17	19	36	34	28	25	40		
8	F, кГц	0,3	0,56	1,2	3,5	5	7	9	С прореживанием по частоте 1	15 бита
	A	21	18	23	19	13	20	11		
9	F, кГц	10	13	14	17	21	22	25	С прореживанием по времени 2	16 бита
	A	9	14	15	19	21	26	30		
10	F, кГц	1	5	7	13	18	19	29	С прореживанием по частоте 2	17 бита
	A	9	11	12	17	21	29	35		
11	F, кГц	10	11	12	13	15	20	25	С прореживанием по времени 3	18 бита
	A	9	11	15	17	21	29	35		
12	F, кГц	21	25	32	33	41	45	50	С прореживанием по частоте 3	17 бита
	A	9	11	15	17	21	29	35		

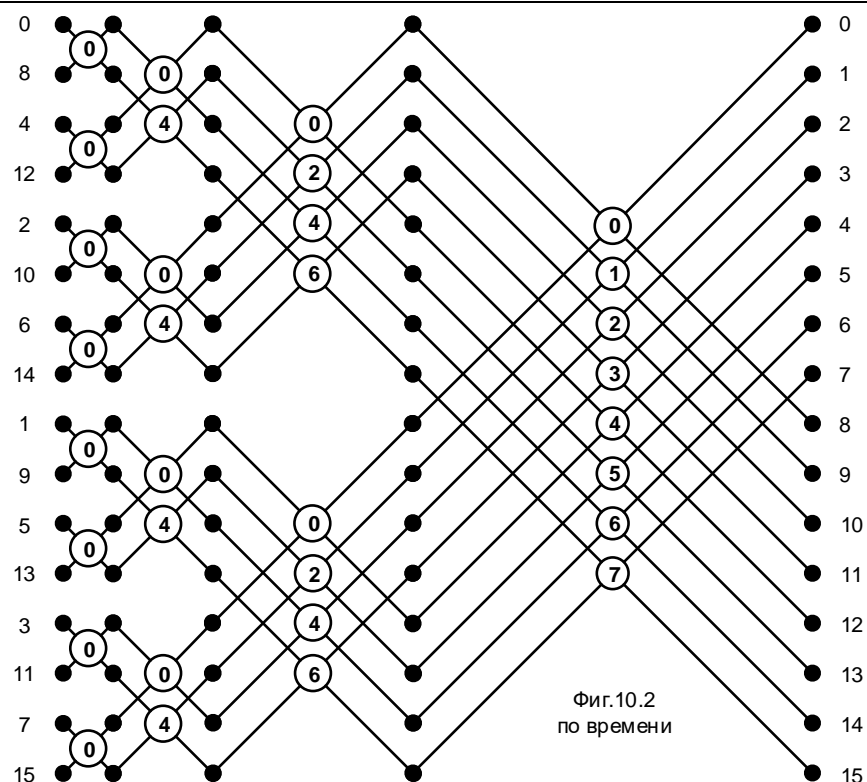
13	F, кГц	20	21	22	23	27	30	40	С прореживанием по времени 1	16 бит
	A	9	11	15	17	21	29	35		
14	F, кГц	11	15	12	13	21	23	25	С прореживанием по частоте 1	15 бит
	A	19	12	15	27	23	29	25		
15	F, кГц	5	11	12	13	21	23	30	С прореживанием по времени 2	14 бит
	A	9	11	15	17	21	29	35		
16	F, кГц	11	13	15	17	21	23	25	С прореживанием по частоте 2	13 бит
	A	22	23	25	17	21	29	35		
17	F, кГц	1	2	3	7	9	11	13	С прореживанием по времени 3	12 бит
	A	2	12	11	15	3	8	13		
18	F, кГц	3	9	14	16	21	26	32	С прореживанием по частоте 3	11 бит
	A	12	2	17	5	9	15	6		
19	F, кГц	5	7	12	17	24	26	43	С прореживанием по времени 1	10 бит
	A	5	14	3	23	6	20	7		
20	F, кГц	1	4	14	19	20	23	29	С прореживанием по частоте 1	8 бит
	A	8	5	12	7	3	15	8		

Различные виды информационных графов алгоритма БПФ приведены в таблице 2.

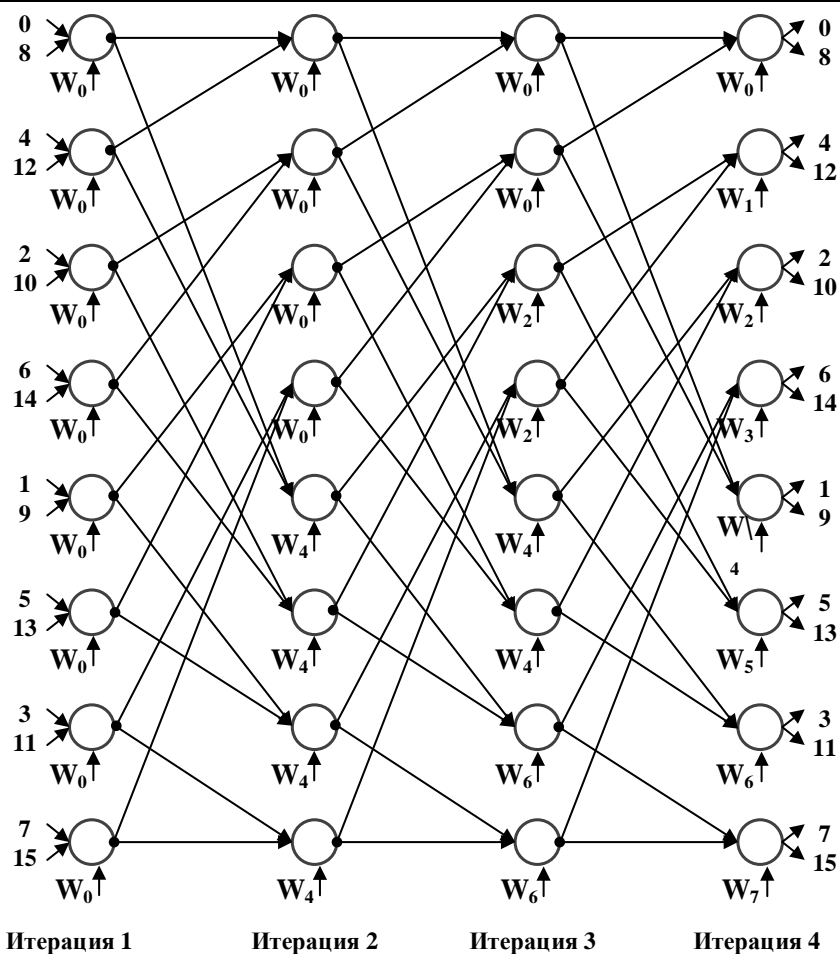
Таблица 2 – варианты информационных графов алгоритма БПФ



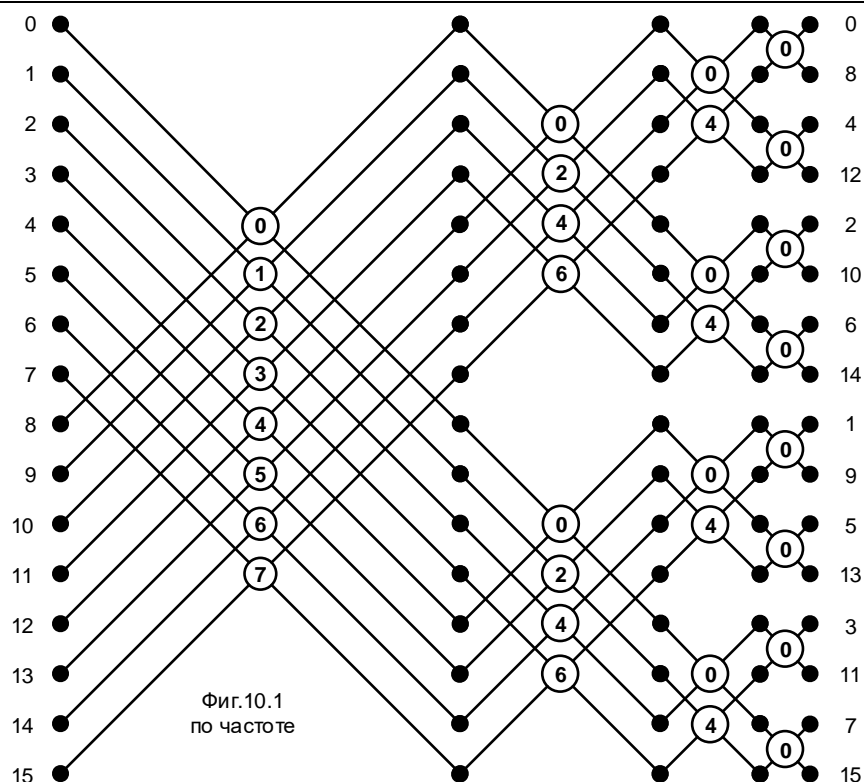
С прореживанием
по времени №2



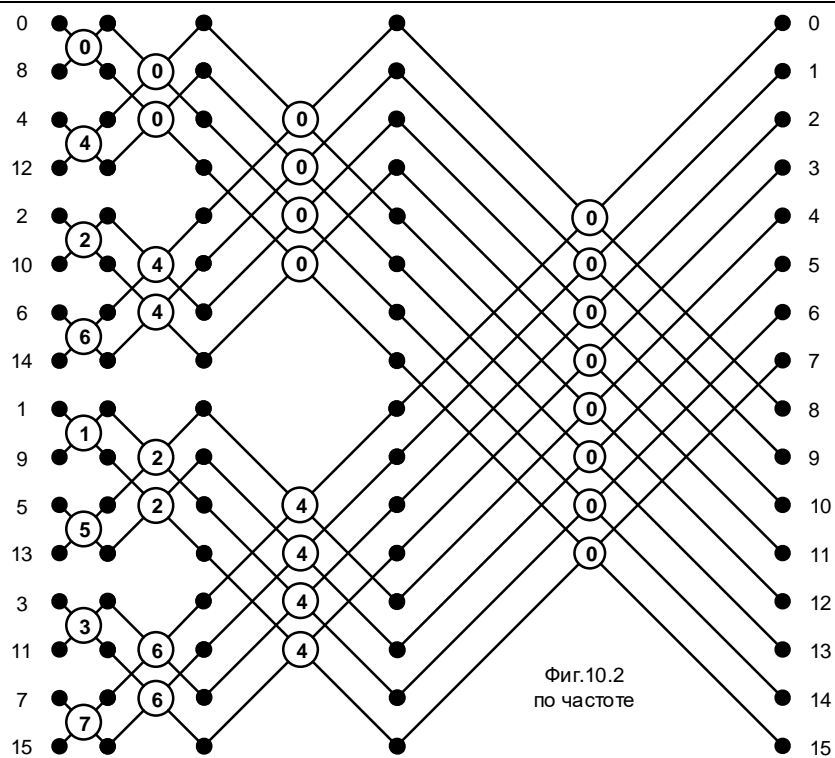
С прореживанием
по времени №3



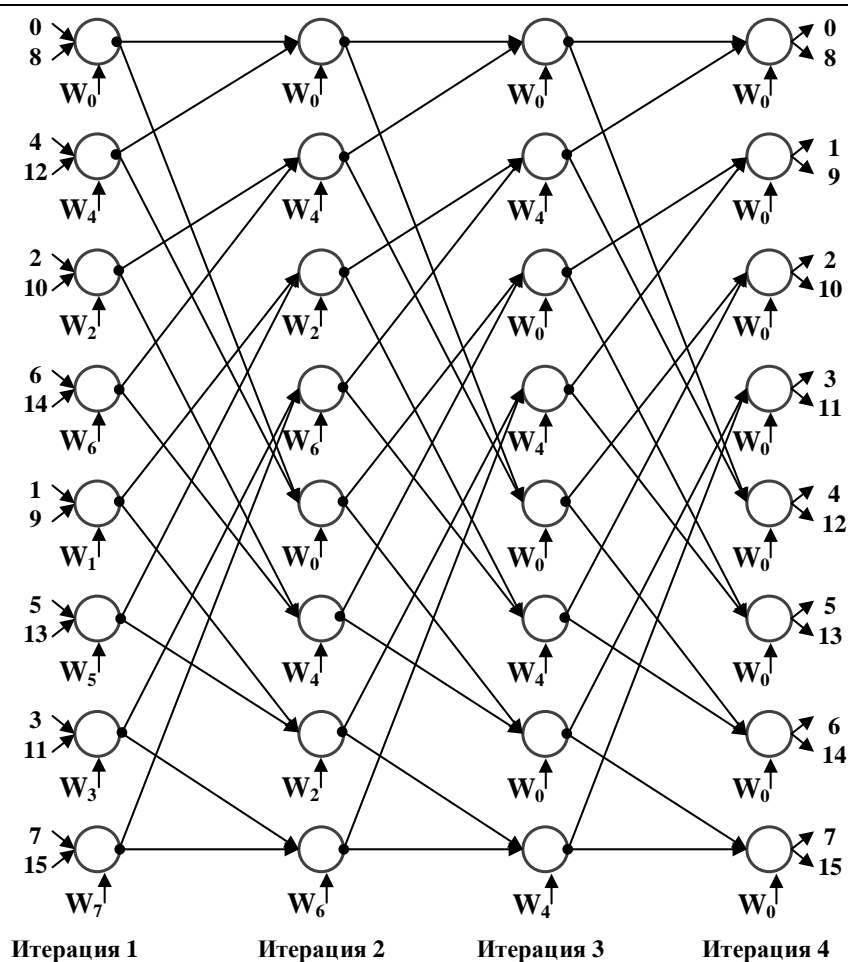
С прореживанием
по частоте №1



С прореживанием
по частоте №2



С прореживанием
по частоте №3



Требования к отчету и защита

Процесс выполнения лабораторной работы документируется с помощью текстового редактора MS Word, полученные сведения служат основой для формирования отчета о выполнении лабораторной работы. Отчет в общем случае должен включать:

- титульный лист;
- описание задач в выбранном варианте лабораторной работы;
- график исходного сигнала во временной области
- график исходного сигнала в частотной области
- график сигнала в частотной области после обхода графа
- выводы
- листинг программы

Защита отчета о выполнении лабораторной работы сопровождается демонстрацией работоспособности кода программ, теоретических знаний и ответов на дополнительные вопросы преподавателя по теме занятия.

ПРИЛОЖЕНИЕ 1.

1. Инструкция по установке IDE PyCharm.

- a. Зайти на страницу загрузки IDE PyCharm с официального сайта JetBrains, опуститься вниз страницы и нажать Download PyCharm Community Edition. Данная версия программного продукта не нуждается в лицензировании.

<https://www.jetbrains.com/pycharm/download/?section=windows>

- b. Установить IDE PyCharm. При установке выбрать все чекбоксы.
- c. С официального сайта Python скачать и установить последнюю версию интерпретатора языка. При установке отметить все чекбоксы (обязательно выбрать «Add python.exe to PATH»).

<https://www.python.org/downloads/>

2. Инструкция по установке VSCode и начальная настройка среды.

- a. Зайти на главную страницу официального сайта VSCode и нажать кнопку Download. VSCode не нуждается в лицензировании.
- b. Установить VSCode. При установке выбрать все чекбоксы.
- c. Открыть среду VSCode и зайти в раздел расширения (Extensions).



- d. Найти и установить расширение «Russian Language Pack for Visual Studio Code» (опционально).
- e. Найти и установить расширение «Python».
- f. С официального сайта Python скачать и установить последнюю версию интерпретатора языка. При установке отметить все чекбоксы (обязательно выбрать «Add python.exe to PATH»).


<https://www.python.org/downloads/>

3. Подготовка работы с IDE.

- a. Проверить, установлена ли на компьютере IDE PyCharm или VSCode. Если нет, выполнить установку согласно описанной выше инструкции.
- b. Создать рабочую директорию с вашей фамилией. Желательно, чтобы папка находилась не на рабочем столе Windows.
- c. Запустить IDE и выполнить команду File => Open Folder, где выбрать путь к созданной вами директории.
- d. Создать новый файл Python File: команда File => New => Python File и задать имя файла с расширением *.py (например, lab1.py).
- e. Файл открылся в правой области окна проекта. Здесь можно набирать код программы.

4. Пример написания и запуска программы в IDE.

Хорошей традицией при изучении первого языка программирования является написание программы, выводящей на экран компьютера приветствие «Hello World!». Для создания такой программы необходимо предварительно создать и открыть новый python файл, как это было описано выше.

- a. Наберите команду: `print('Hello Python!')`
- b. Выполните запуск программы нажав на клавишу RUN , расположенную в правом верхнем углу IDE. Результат работы программы появится мгновенно в окне вывода, расположенном в нижней части IDE.
- c. Для запуска программы в первый раз нужно щелкнуть правой кнопкой мыши в окне с текстом программы и выбрать пункт Run.
- d. Если запуск в IDE PyCharm произошел с ошибкой, проверьте, указан ли интерпретатор языка в окне Run => Edit Configuration.

5. Инструкция по установке пакетов matplotlib, scipy, numpy.

- a. В IDE VSCode зайти в раздел расширения (Extensions); в IDE PyCharm открыть вкладку Python Packages.
- b. Найти и установить библиотеки matplotlib, scipy, numpy.