МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”
Інститут прикладної математики та фундаментальних наук

Кафедра прикладної математики

ЗВІТ

про виконання лабораторних робіт
з дисципліни

«МАТЕМАТИЧНІ ОСНОВИ ЦИФРОВОЇ ОБРОБКИ СИГНАЛІВ»

Виконала:
студентка групи ПМ-33
Вітюк Аліна Сергіївна

Прийняв:
канд. фіз.-мат. наук, доц.
Пабирівський Віктор
Володимирович

Львів-2023

Лабораторна робота №3

Тема: швидке перетворення Фур'є

Мета: розробити комп'ютерну програму для виконання швидкого перетворення Фур'є (ШПФ).

Варіант: 2

Постановка задачі:

Реалізувати на мові програмування: C++, C#, Python, JavaScript (*за згодою керівника можна використати іншу мову програмування*):

1. Програмний код що реалізує ШПФ на основі програмного коду запропонованого у [1,2].
2. При складанні підпрограм оцінювати такі показники:
   a. Час обчислення .
   b. Кількість операцій (множення, додавання).
3. Порівняти швидкодію алгоритму ШПФ за оцінками часу та кількості операцій з результатами для ДПФ із попередньої ЛР.
4. Генератор послідовності значень вхідного сигналу.

Теоретичні відомості:

Швидке перетворення Фур'є (ШПФ)— це алгоритм обчислення, який успішно використовує властивості періодичності тригонометричних функцій, *щоб уникнути непотрібних обчислень* у дискретному перетворенні Фур'є. Це, безперечно, важливий інструмент при обробці сигналів, але алгоритм обчислення досить складний, і слід сказати, що суть ШПФ скоріше полягає не в обробці сигналів, а в методі обчислення числових значень.

І у разі ШПФ, і у разі ДПФ результат обчислень один й той самий.

(*Отже,у ШПФ число операцій менше, тому ступінь точності обчислень вищий*).

Швидке перетворення Фур'є (ШПФ) - *це алгоритм ефективного обчислення з використанням закономірностей, прихованих усередині матриці,* що виражає дискретне перетворення Фур'є (ДПФ) .

Таким чином, якщо правильно переставити елементи матриці, то кількість операцій множення зменшиться.

Однак порівняно з відніманням і додаванням на множення витрачається набагато більше часу, тому, якщо зменшити кількість операцій множення, які займають майже весь обсяг обчислень, то в результаті можна скоротити час, витрачений на весь процес обчислення.

Хід роботи:

Для реалізації поставлених завдань обрала мову програмування Matlab.

1. Програмний код що реалізує ШПФ на основі програмного коду запропонованого у першому джерелі:

```
% Швидке перетворення Фур'є за допомогою алгоритму Кулі-Тьюкі
function X = fft(x)
n = length(x);
if n == 1
X = x;
else
even = fft(x(1:2:end));
odd = fft(x(2:2:end));
factor = exp(-2i * pi / n * (0:n/2-1));
X = [even + factor .* odd, even - factor .* odd];
end
End
```

2. Оцінка показників:

   a. час обчислення.

```
elapsed_time = toc;
fprintf('Час обчислення: %.10fc\n', elapsed_time);
```

   b. кількість операцій (множення, додавання).

```
add_operations = N;
mult_operations = 4*N;
fprintf('Кількість операцій множення: %.0f\n', mult_operations);
fprintf('Кількість операцій додавання: %.0f\n\n', add_operations);
```

3. Порівняння швидкості алгоритму ШПФ за оцінками часу та кількості операцій з результатами для ДПФ із попередньої ЛР.

```
>> Lab2
Час обчислення: 0.018506
Кількість операцій множення: 104244
Кількість операцій додавання: 102
>> Lab3
Час обчислення: 0.0047660000c
Кількість операцій множення: 408
Кількість операцій додавання: 102
```

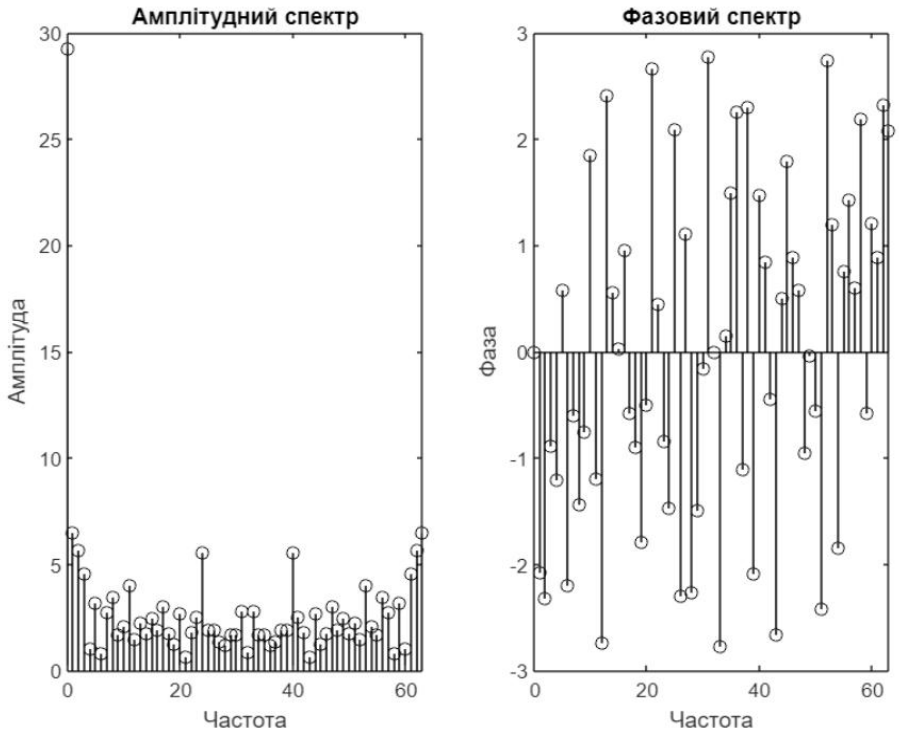
4. Генератор послідовності значень вхідного сигналу.

```
% Генерація випадкової послідовності
clear all
n = 2;
N = 50 + n;
x = rand(1, N);
```

Код програми:

```
% Генерація випадкової послідовності
clear all
n = 2;
N = 50 + n;
x = rand(1, N);

% Доповнюємо вхідний сигнал нулями до степеня 2
M = 2^nextpow2(N);
x = [x, zeros(1, M - N)];

% Обчислення ШПФ та оцінка часу обчислення
tic;
X = myfft(x);
elapsed_time = toc;

% Оцінка кількості операцій
add_operations = N;
mult_operations = 4*N;

fprintf('Час обчислення: %.10fc\n', elapsed_time);
fprintf('Кількість операцій множення: %.0f\n', mult_operations);
fprintf('Кількість операцій додавання: %.0f\n\n', add_operations);
for i = 1:length(X)
fprintf('C_%d = %.10f + %.10fi\n', i-1, real(X(i)), imag(X(i)));
end

% Обчислення спектра амплітуд та фаз та побудова графіків
amp_spectrum = abs(X);
phase_spectrum = angle(X);
freq_axis = 0:M-1;

figure;
subplot(1, 2, 1);
stem(freq_axis, amp_spectrum, 'k');
title('Амплітудний спектр');
xlabel('Частота');
ylabel('Амплітуда');

subplot(1, 2, 2);
stem(freq_axis, phase_spectrum, 'k');
title('Фазовий спектр');
xlabel('Частота');
ylabel('Фаза');

% Швидке перетворення Фур'є за допомогою алгоритму Кулі-Тьюкі
function X =myfft(x)
n = length(x);
if n == 1
X = x;
else
even = fft(x(1:2:end));
odd = fft(x(2:2:end));
factor = exp(-2i * pi / n * (0:n/2-1));
X = [even + factor .* odd, even - factor .* odd];
end
end
```

Результат виконання програмної реалізації:




```
Command Window
>> Lab3
Час обчислення: 0.0048740000c
Кількість операцій множення: 208
Кількість операцій додавання: 52

C_0 = 25.5867255273 + 0.0000000000i
C_1 = -4.8357757466 + -2.9016328656i
C_2 = -2.8417758628 + -5.0538639754i
C_3 = 0.8592609722 + -3.4923365728i
C_4 = 1.3383376624 + 1.7805791344i
C_5 = 0.1049611207 + 1.1057992942i
C_6 = 0.6315335939 + -1.9728116816i
C_7 = 1.1283176983 + -0.4361906224i
C_8 = 0.7172159748 + -4.4040086834i
C_9 = -0.8738415550 + 2.9418979624i
C_10 = 0.6968874637 + -3.8547277831i
C_11 = -1.8424210240 + -0.2404864019i
C_12 = 1.2762081405 + 0.5100309422i
C_13 = -1.7010809789 + 0.2785477708i
C_14 = 0.3171124900 + -1.9272023086i
C_15 = 0.2390649424 + 0.5347204802i
C_16 = 1.6765625141 + 1.2566904005i
C_17 = -1.0205233290 + 0.2069411167i
C_18 = 2.2129052153 + -2.8822930092i
C_19 = 0.7852438214 + -2.0073495483i
C_20 = -1.3527846914 + -1.1553261228i
C_21 = -0.2601375170 + 1.0743799545i
C_22 = -1.3555012432 + -3.6449232354i
C_23 = -0.7374784912 + -0.5639844249i
C_24 = -2.0769677189 + 1.1292188545i
C_25 = 0.4425104742 + -0.8322486985i
```

…

```
C_40 = -2.0769677189 + -1.1292188545i
C_41 = -0.7374784912 + 0.5639844249i
C_42 = -1.3555012432 + 3.6449232354i
C_43 = -0.2601375170 + -1.0743799545i
C_44 = -1.3527846914 + 1.1553261228i
C_45 = 0.7852438214 + 2.0073495483i
C_46 = 2.2129052153 + 2.8822930092i
C_47 = -1.0205233290 + -0.2069411167i
C_48 = 1.6765625141 + -1.2566904005i
C_49 = 0.2390649424 + -0.5347204802i
C_50 = 0.3171124900 + 1.9272023086i
C_51 = -1.7010809789 + -0.2785477708i
C_52 = 1.2762081405 + -0.5100309422i
C_53 = -1.8424210240 + 0.2404864019i
C_54 = 0.6968874637 + 3.8547277831i
C_55 = -0.8738415550 + -2.9418979624i
C_56 = 0.7172159748 + 4.4040086834i
C_57 = 1.1283176983 + 0.4361906224i
C_58 = 0.6315335939 + 1.9728116816i
C_59 = 0.1049611207 + -1.1057992942i
C_60 = 1.3383376624 + -1.7805791344i
C_61 = 0.8592609722 + 3.4923365728i
C_62 = -2.8417758628 + 5.0538639754i
C_63 = -4.8357757466 + 2.9016328656i
```

Висновок: виконання лабораторної роботи №3 допомогло пригадати теоретичні відомості про перетворення Фур'є. У ході виконання лабораторної роботи навчилася реалізовувати комп'ютерну програму мовою програмування Matlan для виконання швидкого перетворення Фур'є.

Посилання на GitHub: https://github.com/AlinaVitiuk/Mathematical-foundations-of-digital-signal-processing