Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №2.1

з дисципліни
«Інтелектуальні вбудовані системи»

на тему
«Дослідження параметрів алгоритму дискретного перетворення Фур'є»

Виконав:

студент групи ІП-84
Голубов Іван
номер залікової книжки: 8404

Перевірив:

викладач
Регіда Павло Геннадійович

Київ 2021

Основні теоретичні відомості

В основі спектрального аналізу використовується реалізація так званого дискретного перетворювача Фур'є (ДПФ) з неформальним (не формульним) поданням сигналів, тобто досліджувані сигнали представляються послідовністю відліків $x(k)$

$$F_x(p) = \sum_{k=0}^{N-1} x(k) \cdot e^{-jk\Delta tp\Delta\omega}$$

$$\omega \to \omega_p \to p\Delta\omega \to p \qquad \Delta\omega = \frac{2\pi}{T}$$

На всьому інтервалі подання сигналів **T**, **2π** - один період низьких частот. Щоб підвищити точність треба збільшити інтервал **T.**

$$t \to t_k \to k\Delta t \to k; \quad \Delta t = \frac{T}{N} = \frac{1}{k_{зап.}} \cdot f'гр.$$

ДПФ - проста обчислювальна процедура типу звірки (тобто Σ-е парних множень), яка за складністю також має оцінку $\mathbf{N^2 + N}$. Для реалізації ДПФ необхідно реалізувати поворотні коефіцієнти ДПФ:

$$W_N^{pk} = e^{-jk\Delta t\Delta\omega p}$$

Ці поворотні коефіцієнти записуються в ПЗУ, тобто є константами.

$$W_N^{pk} = e^{-jk\frac{T}{N}p\frac{2\pi}{T}} = e^{-j\frac{2\pi}{N}pk}$$

W_N^{pk} не залежать від **T**, а лише від розмірності перетворення **N.** Ці коефіцієнти подаються не в експоненційній формі, а в тригонометричній.

$$W_N^{pk} = \cos\left(\frac{2\pi}{N}pk\right) - j\sin\left(\frac{2\pi}{N}pk\right)$$

Ці коефіцієнти повторюються (тому і **p** до **N-1**, і **k** до **N-1**, а **(N-1) • (N-1)**) з періодом **N(2π)**.. Т.ч. в ПЗУ треба зберігати N коефіцієнтів дійсних і уявних частин. Якщо винести знак коефіцієнта можна зберігати **N/2** коефіцієнтів.

2π/N- деякий мінімальний кут, на який повертаються ці коефіцієнти. У ПЗУ окремо зберігаються дійсні та уявні частини компілюють коефіцієнтів. Більш загальна форма ДПФ представляється як:

$$F_x(p) = \sum_{k=0}^{N-1} x(k) \cdot W_N^{pk}$$

Коефіцієнти зручно представити у вигляді таблиці:

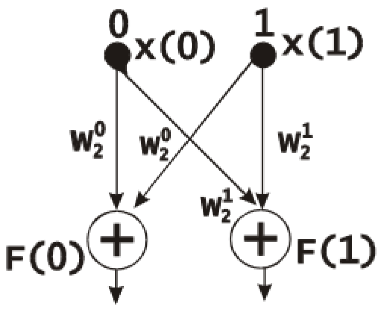
k \ p	0	1	2	3
0	W_4^0	W_4^0	W_4^0	W_4^0
1	W_4^0	W_4^1	W_4^2	W_4^3
2	W_4^0	W_4^2	W_4^0	W_4^2
3	W_4^0	W_4^3	W_4^2	W_4^1

Різних тут всього 4 коефіцієнта:

$$W_4^0 = \cos\left(\frac{2\pi}{4}\cdot 0\right) - j\sin\left(\frac{2\pi}{4}\cdot 0\right) = 1 \qquad (W_4^1 = -j;\ W_4^2 = -1;\ W_4^3 = +j)$$

Можна в пам'яті зберігати тільки 2, а решта брати з "-", якщо $\frac{N}{2} - 1 < pk$. 4 ДПФ це вироджені перетворення, по модулю ці коефіцієнти = 1 і всі 4 ДПФ можуть реалізуватися на 24-х суматора. Це буде далі використовуватися в реалізації ШПФ з основою 4.

2ДПФ реалізується ще простіше:

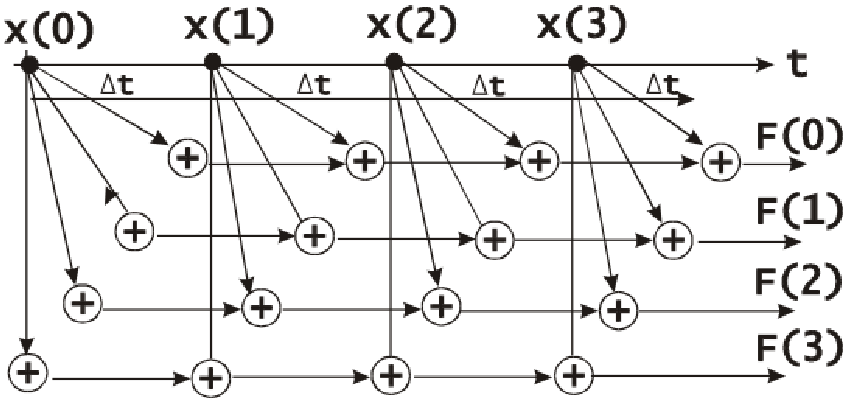


$(W_2^0 = +1;\ W_2^1 = -1)$

Спеціальна схема реалізації ДПФ з активним використанням пауз між відліками

При реалізації ДПФ можна організувати обробку в темпі надходження даних.

Реалізація схеми в БПФ з активним використанням пауз на 4-х точках виглядає так:



Ця схема сильно залежить от **Δt** и **N**.

Завдання на лабораторну роботу

Для згенерованого випадкового сигналу з Лабораторної роботи N 1
відповідно до
заданого варіантом (Додаток 1) побудувати його спектр, використовуючи
процедуру
дискретного перетворення Фур'є. Розробити відповідну програму і вивести
отримані
значення і графіки відповідних параметрів.

Варіант-04

Число гармонік в сигналі: 12.

Гранична частота: 2400.

Кількість дискретних відліків: 1024

Лістинг програми

lab2.m

```
X=zeros(3,1024);
x(1,:)=signal_generator(12,1024);
x(2,:)=signal_generator(12,1024);
x(3,:)=signal_generator(12,1024);
N = 1024;
Fd=1024;% Частота дискретизации
[F_DFT, title_DFT] = FT.DFT(x(1,:),N);
[F_FFT_h_W, title_FFT_h_W]= FT.FFT_handmade_without_W  (x(1,:),N);
[F_FFT_h, title_FFT_h]= FT.FFT_handmade (x(1,:),N);
```

```
[F_FFT_m, title_FFT_m]=  FT.FFT_matlab(x(1,:));
[A_lenH_DFT, H1]= FT.plot_FT (F_DFT,title_DFT,Fd,N);
[A_lenH_FFT, H2]= FT.plot_FT (F_FFT_h,title_FFT_h,Fd,N);
FT.plot_FT (F_FFT_h_W,title_FFT_h_W,Fd,N);
FT.plot_FT (F_FFT_m,title_FFT_m,Fd,N);
FT.compare_deltaA_FFT_DFT (A_lenH_DFT, A_lenH_FFT, H1);
f_DFT=@() FT.DFT(x(1,:),N);
f_FFT=@() FT.FFT_handmade(x(1,:),N);
t_DFT=MD.time_f(f_DFT,'ДПФ');
t_FFT=MD.time_f(f_FFT,'БПФ');
MD.difference_time (t_DFT,t_FFT, 'ДПФ', 'БПФ');
[numRows,numCols ] = size(X);
for i=1:numRows
   FT.compare_time_FFT(X(i,:), N);
end
```

FT.m

```
classdef FT
   methods (Static)
      function [F, title_FT] = DFT(x,N)
         F = complex(zeros(1,N));
         title_FT = 'АЧХ (ДПФ)';
         for p=0:N-1
            for k = 0:N-1
               F(1,p+1) = F(1,p+1) + x(k+1)*(cos(2*pi*p*k/N)-sin(2*pi*p*k/N)*1i);
            end
         end
      end
      function [F, title_FT]=FFT_handmade_without_W (x,N)
         F = complex(zeros(1,N));
         title_FT= 'АЧХ (БПФ без W)';
         w=zeros(2,N);
         for i = 0:N-1
            if i<=N/8
               w(1,i+1)=cos(2*pi*i/N);
               w(2,i+1)=sin(2*pi*i/N);
            elseif i <=N/4
               w(1,i+1)=w(2,N/4-i+1);
               w(2,i+1)=w(1,N/4-i+1);
            elseif i<=N/2
```

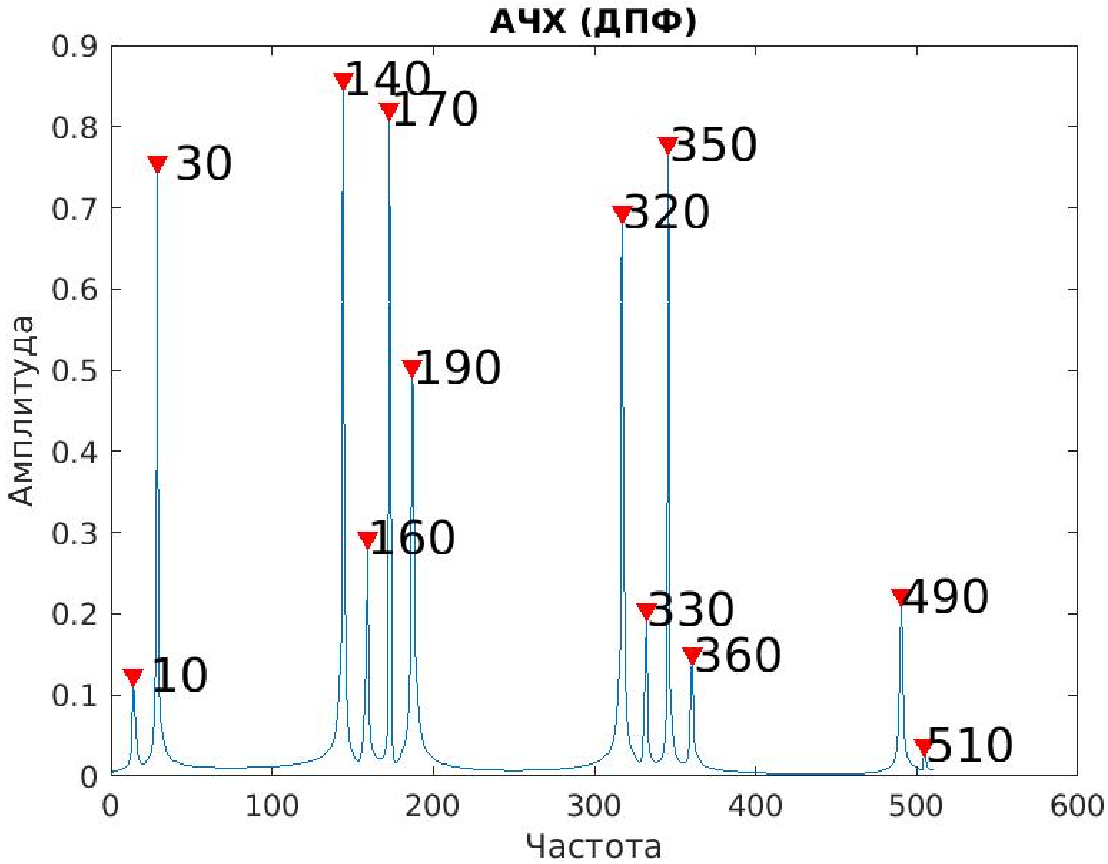
```
            w(1,i+1)=-w(2,i-N/4+1);
            w(2,i+1)=+w(1,i-N/4+1);
        else
            w(1,i+1)=-w(1,i-N/2+1);
            w(2,i+1)=-w(2,i-N/2+1);
        end
    end
    for p=0:N-1
        for k = 0:N-1
            a=mod(p*k,N);
            F(1,p+1) = F(1,p+1) + x(k+1)*(w(1,a+1)-w(2,a+1)*1i);
        end
    end
end
function [F, title_FT]=FFT_handmade (x,N)
    F = complex(zeros(1,N));
    title_FT= 'АЧХ (БПФ)';
    w=zeros(2,N/4+1);
    W=zeros(1,N);
    for i = 0:N/4
        if i<=N/8
            w(1,i+1)=cos(2*pi*i/N);
            w(2,i+1)=sin(2*pi*i/N);
        else
            w(1,i+1)=w(2,N/4-i+1);
            w(2,i+1)=w(1,N/4-i+1);
        end
    end
    for b= 0:N-1
        if b <=N/4
            W(1,b+1)= w(1,b+1)-w(2,b+1)*1i;
        elseif b<=N/2
            W(1,b+1)=W(1,b-N/4+1)*-1*1i;
        else
            W(1,b+1)=-W(1,b-N/2+1);
        end
    end
    for p=0:N-1
        for k = 0:N-1
```

```
            a=mod(p*k,N);
            F(1,p+1) = F(1,p+1) + x(k+1)*W(1,a+1);
        end
    end
end
function [F, title_FT]=FFT_matlab(x)
    F=fft(x);
    title_FT = 'АЧХ (БПФ, встроенная функция)';
end
function [A_lenH, H]=plot_FT (F,title_FT,Fd,N)
    A=abs(F);% Амплитуды преобразования Фурье сигнала
    A=2*A./N;% Нормировка спектра по амплитуде
    A(1)=A(1)/2;% Нормировка постоянной составляющей в спектре
    H=0:Fd/N:Fd/2-1/N;% Массив частот вычисляемого спектра Фурье
    A_lenH = A(1:length(H));

    figure
    plot(H,A_lenH);% Построение спектра Фурье сигнала
    hold on;
    title(title_FT);
    xlabel('Частота');
    ylabel('Амплитуда');
    [~,locs1] = findpeaks(A_lenH,'MinPeakHeight',0.02,...
        'MinPeakDistance',2);
    plot(H(locs1),A_lenH(locs1),'rv','MarkerFaceColor','r');
    cellpeaks = cellstr(num2str(round(H(locs1)',-1)));
    text(H(locs1),A_lenH(locs1),cellpeaks,'FontSize',16);
    hold off;
    saveas(gcf, sprintf('./res/%s.jpg',title_FT))
end
function compare_deltaA_FFT_DFT (A_lenH_DFT, A_lenH_FFT, H)
    figure
    plot(H,A_lenH_FFT-A_lenH_DFT);
    hold on;
    title('Отклонение БПФ от ДПФ');
    xlabel('Частота');
    ylabel('Отклонение');
    saveas(gcf,'./res/compare_deltaA_FFT_DFT.jpg');
end
```

```
        function compare_time_FFT(y,N)
            f1=@() FT.FFT_handmade_without_W(y,N);
            f2=@() FT.FFT_handmade(y,N);
            t1=MD.time_f(f1,'БПФ без W');
            t2=MD.time_f(f2,'БПФ');
            MD.difference_time (t1,t2, 'БПФ без W', 'БПФ');
        end
    end
end
```

Результати роботи програми



Висновки

Під час даної лабораторної роботи я ознайомився з принципами реалізації спектрального аналізу випадкових сигналів на основі алгоритму перетворення Фур'є, вивчив та дослідив особливості даного алгоритму з використанням засобів моделювання і сучасних програмних оболонок.