МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«КРЫМСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ им. В. И. ВЕРНАДСКОГО»

ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ

Кафедра компьютерной инженерии и моделирования

**Сигналы с ограниченным спектром**

Отчет по лабораторной работе 5

по дисциплине «**Обработка сигналов**»

студента 3 курса группы ИВТ-б-о-222  
Гоголева Виктора Григорьевича

Направления подготовки 09.03.01«Информатика и вычислительная техника»

Симферополь, 2025

**Лабораторная работа №5**

**Тема:** Спектральный анализ периодических сигналов

Цель работы: Задан сигнал. Определить эффективную ширину спектра данного сигнала. Рассчитать отсчетные значения этого сигнала, необходимые для его однозначного восстановления. Восстановить сигнал по его отсчетным значениям. Построить следующие графики:

1. исходный сигнал;
2. отсчетные значения исходного сигнала по оси времени;
3. восстановленный сигнал.

Параметры сигнала приведены в таблице

|  |  |  |
| --- | --- | --- |
| **Сигнал** | **E, В** | **t, мкс** |
|  | -10 | 6 |

**Теоретические сведения**

Любой сигнал имеет бесконечный спектр. Однако, как правило, существует эффективная ширина спектра (ωв), в пределах которой передается основная мощность сигнала. Определяется эффективная ширина спектра с использованием теоремы Парсеваля (22). Эффективная ширина выбирается исходя из потери не более 10% энергии сигнала.

Сигнал с ограниченным спектром может быть представлен в виде набора дискретных отсчетных значений сигнала, взятых через равные промежутки времени .

Произвольный сигнал s(t) с ограниченным спектром может быть разложен в обобщенный ряд Фурье по базису Котельникова:



где ck – коэффициенты обобщенного ряда Фурье; Sck(t;ωв) – k-ая отсчетная функция, совокупность которых образует базис Котельникова. Sck(t;ωв) вычисляется по следующей формуле:



Коэффициенты обобщенного ряда Фурье вычисляются о следующей формуле:



Зная спектральную плотность S(ω) заданного сигнала s(t) и используя обобщенную формулу Рэлея, можно найти коэффициенты разложения через интеграл по частотному спектру:



Мгновенное значение сигнала s(t) в k-ой отсчетной точке tk=k⋅π/ωв=k/(2⋅fв):



Тогда ряд Котельникова имеет вид:

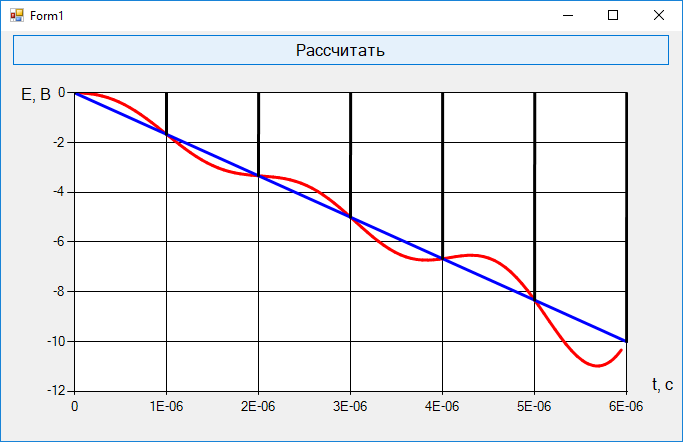
,



**Ход работы**

При запуске программы появляется окно, с кнопкой. При нажатии строятся исходный сигнал, отсчетные значения исходного сигнала по оси времени, восстановленный сигнал.

**Результаты**

****

**Анализ результатов**

Ширина спектра сигнала 3141592 Гц. При такой частоте будет 7 отсчетов. В результате выполнения программы можно сказать, что восстановленная функция верна, т.к. она проходит через все дискретные точки.

**Вывод:** во время выполнения данной работы был изучен принцип нахождения ширины спектра сигнала, расчета дискретных значений сигнала, восстановления по отсчетным значениям с помощью ряда Котельникова. Также была найдена ширина спектра, отсчетные значения сигнала и построен восстановленный сигнал.

Была создана программа, которая вычисляет необходимые величины и строит графики отсчетных значений, изначального и восстановленного сигнала.

**Приложение**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace \_5.Сигналы\_с\_ограниченным\_спектром

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private static double Ec()

{

return 2.79 \* 6 \* 6 \* 6/3;

}

private static double A(double w)

{

return -10 \* 0.000006 / (6 \* w) \* Math.Sin(0.000006 \* w) + 10 / (6 \*w \*w) \* Math.Cos(0.000006 \* w);

}

private static double B(double w)

{

return 10 \* 0.000006 / (6 \* w) \* Math.Cos(0.000006 \* w) - 10 / (6 \* w \* w) \* Math.Sin(0.000006 \* w);

}

private static double Polinom(double S, double t, int k, int wv)

{

return S\*Math.Sin(wv \* (t - k \* Math.PI / wv)) / (wv \* (t - k \* Math.PI / wv));

}

private void button1\_Click(object sender, EventArgs e)

{

chart3.ChartAreas[0].AxisX.Minimum = 0;

chart3.ChartAreas[0].AxisX.Maximum = 0.00000601;

chart3.ChartAreas[0].AxisX.Interval = 0.000001;

int wvn = 3141592;

double[] SV = new double [100];

double Ew = 0;

double wv = 0;

int line=2;

for (double x = 0; x <= 0.000006; x += 0.000001)

{

chart3.Series[line].Points.AddXY(x,0);

chart3.Series[line].Points.AddXY(x, x\*1000000\*-1.6667);

line++;

}

for (double w = 1; Ew / Ec() <= 0.95; w++)

{

Func<double, double> Si = x => A(w) + B(w);

Integral I = new Integral(0.000001, w, Si);

double Sum = Math.Abs(I.Simpson(0.000001, w, 1000));

Ew += 1 / Math.PI \* Sum;

wv = w;

}

int T = 7;

double[] S = new double[T];

double[] Sk = new double[T];

for (int i = 0; i < T; i++)

{

Func<double, double> Si = x => Math.Sqrt(Math.Pow(1.67 / x \* Math.Sin(x \* 0.000006), 2) + Math.Pow(1.67 / x \* Math.Cos(x \* 0.000006), 2)) \* Math.Cos(i \* Math.PI \* x / wvn);

Integral I = new Integral(-wv, wv, Si);

double Sum = Math.Abs(I.Simpson(-wv, wv, 10000));

Sk[i] = Sum / (2 \* Math.PI);

}

int count = 0;

for (int k = 0; k < 7; k++)

{

for (double t = 0; t <= 0.000006; t += 0.000006 / 100)

{

SV[count] += Polinom(S[k], t, k, wvn);

count++;

}

count = 0;

}

count = 0;

for (double i = 0; i <= 0.000006; i += 0.000006 / 100)

{

chart3.Series[0].Points.AddXY(i, SV[count]);

count++;

}

count = 0;

for (double i = 0; i <= 0.000006; i += 0.000001)

{

chart3.Series[1].Points.AddXY(i, S[count]);

count++;

}

}

}

}