**Министерство образования и науки Российской Федерации**

**ФГАОУ ВО «УрФУ имени первого Президента России Б.Н. Ельцина»**

**Институт радиоэлектроники и информационных технологий - РТФ**

**Школа «информационных технологий и автоматики»**

**Виды сортировок и их эффективность**

**Отчет по лабораторной работе №2 по дисциплине**

**«Цифровая обработка сигналов»**

Преподаватель                                                                              Тимошенкова Ю. С.

Студенты гр. РИ-360005                                                                Куприянов М. А.

Екатеринбург

2018

# Оглавление

[Цель работы 2](#_Toc526582327)

[Подготовка 2](#_Toc526582328)

[Листинг 1. Подготовительная часть. 6](#_Toc526582329)

[Функция linearInterpolation 7](#_Toc526582330)

[Листинг 2. Функция linearInterpolation 8](#_Toc526582331)

[Функция pol 8](#_Toc526582332)

[Листинг 3. Функция pol. 12](#_Toc526582333)

[Функция newton 13](#_Toc526582334)

[Листинг 4. Функция newton 16](#_Toc526582335)

[Функция lagrange 17](#_Toc526582336)

[Листинг 5. Функция lagrange 20](#_Toc526582337)

[Функция spline 21](#_Toc526582338)

[Интерполирование 24](#_Toc526582339)

[Листинг 6. Использование интерполяций 25](#_Toc526582340)

[Оцифровка сигнала, функция adc 27](#_Toc526582341)

[Листинг 7. Функция adc 29](#_Toc526582342)

[Шум 29](#_Toc526582343)

[Вывод 31](#_Toc526582344)

# Цель работы

Целью данной работы является практическое знакомство с интерполяцией и аналогово-цифровым преобразованием.

# Подготовка

Перед работой с интерполированием, необходимо создать векторы отсчетов и векторы значений, которые мы получим в результате использования функции, предложенной в задании и варианта задания, предоставленным преподавателем.

Векторы отсчетов создаются по следующим условиям: “Выберите длительность временного интервала *Ts* такой, чтобы на нем укладывалось 2 периода анализируемого сигнала. Выберите частоты дискретизации так, чтобы на одном периоде сигнала укладывалось 2, 4, 8, 16 отсчетов дискретного сигнала.”

Далее создаем 4 графика для каждого варианта отсчетов, в которых сравниваем дискретные сигналы и исходный непрерывный (график 1, график 2, график 3, график 4)

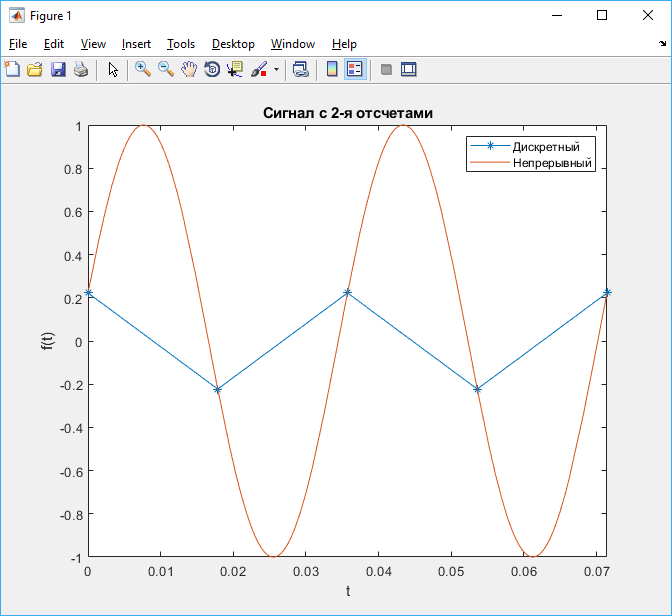


График 1. 2 отсчета на 1 период сигнала.

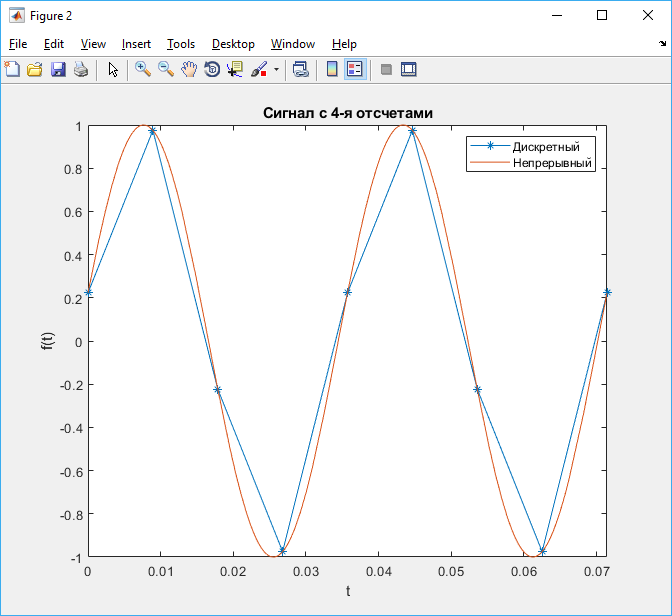


График 2. 4 отсчета на 1 период сигнала.

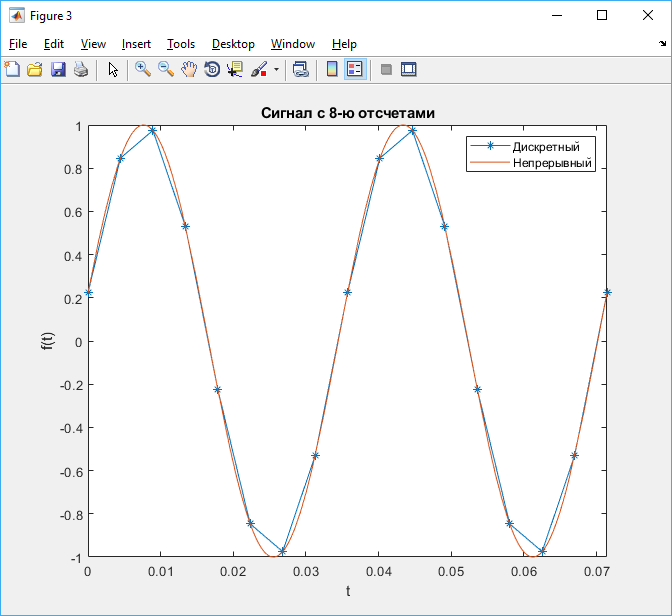


График 3. 8 отсчетов на 1 период сигнала.

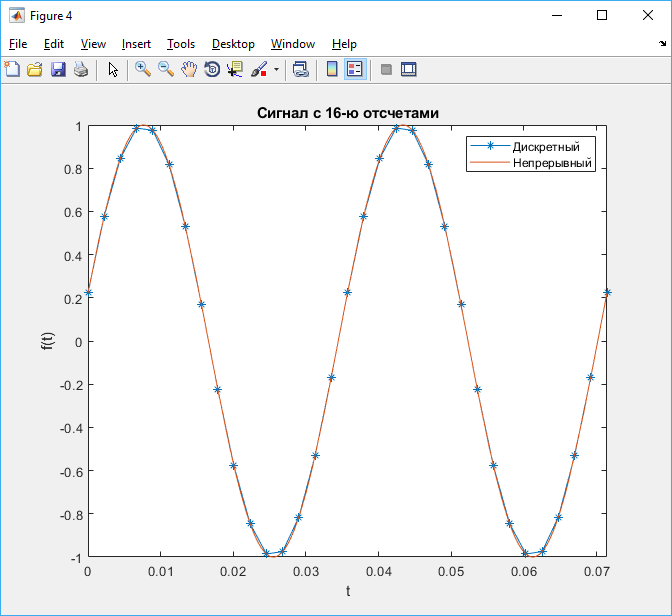


График 4. 16 отсчетов на 1 период сигнала.

## Листинг 1. Подготовительная часть.

N = 28; %сюда вставить свой вариант

%отсчеты

t2 = 0: 1/56: 1/14;%2 отчета на период сигнала

t4 = 0: 1/112: 1/14;%4

t8 = 0: 1/224: 1/14;%8

t16 = 0: 1/448: 1/14;%16

%значения

f2 = sin((2\*pi\*N\*t2)+((2\*pi)./N));

f4 = sin((2\*pi\*N\*t4)+((2\*pi)./N));

f8 = sin((2\*pi\*N\*t8)+((2\*pi)./N));

f16 = sin((2\*pi\*N\*t16)+((2\*pi)./N));

%%

figure;%2 отсчета

plot(t2, f2, '-\*');

axis ([0 1/14 -1 1]);

hold on

fplot(@(t)sin((2\*pi\*N\*t)+((2\*pi)./N)));

legend('Дискретный','Непрерывный');

title('Сигнал с 2-я отсчетами');

xlabel('t');

ylabel('f(t)');

figure;%4

plot(t4, f4, '-\*');

axis ([0 1/14 -1 1]);

hold on

fplot(@(t)sin((2\*pi\*N\*t)+((2\*pi)./N)));

legend('Дискретный','Непрерывный');

title('Сигнал с 4-я отсчетами');

xlabel('t');

ylabel('f(t)');

figure;%8

plot(t8, f8, '-\*');

axis ([0 1/14 -1 1]);

hold on

fplot(@(t)sin((2\*pi\*N\*t)+((2\*pi)./N)));

legend('Дискретный','Непрерывный');

title('Сигнал с 8-ю отсчетами');

xlabel('t');

ylabel('f(t)');

figure;%16

plot(t16, f16, '-\*');

axis ([0 1/14 -1 1]);

hold on

fplot(@(t)sin((2\*pi\*N\*t)+((2\*pi)./N)));

legend('Дискретный','Непрерывный');

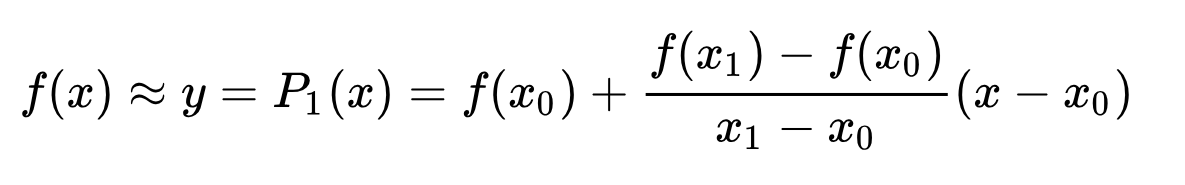
title('Сигнал с 16-ю отсчетами');

xlabel('t');

ylabel('f(t)');

# Функция linearInterpolation

Чтобы работать с сигналом после получения его дискретных значений, нужно использовать интерполяцию, которая сделает из дискретного сигнала сигнал более близкий к изначальному аналоговому сигналу. Первым видом интерполяции будет линейная интерполяция, которая была реализована в функции linearInterpolation. Она принимает точки данных и узел интерполяции, где нужно получить значение. Далее по формуле возвращает значение сигнала. Формула линейной интерполяции:



Графики этой интерполяции можно увидеть на графиках 1, 2, 3, 4.

## Листинг 2. Функция linearInterpolation

function y = linearInterpolation(x, xS, yS)

first = 1;

second = 2;

i = 1;

while i <= length(xS)

if x < xS(i)

second = i;

break;

end

i = i + 1;

end

if second == 1

second = 2;

end

if second == 2 && i == length(xS)+1

second = length(xS);

end

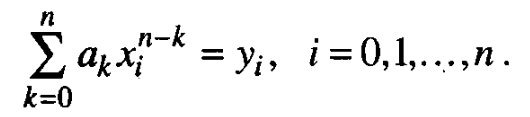
first = second - 1;

y = yS(first) + ((yS(second)-yS(first))\*(x - xS(first)))/(xS(second)-xS(first));

end

# Функция pol

Следующим способом интерполяции нужно реализовать интерполяцию многочлена *N*-ой степени. Она была реализована в функции pol. Функция принимает точки данных и вектор отсчетов, где нужно найти интерполированные значения. Данный способ реализуется через решение следующего уравнения:

,

Далее графики, которые показывают результат реализации этой функции.

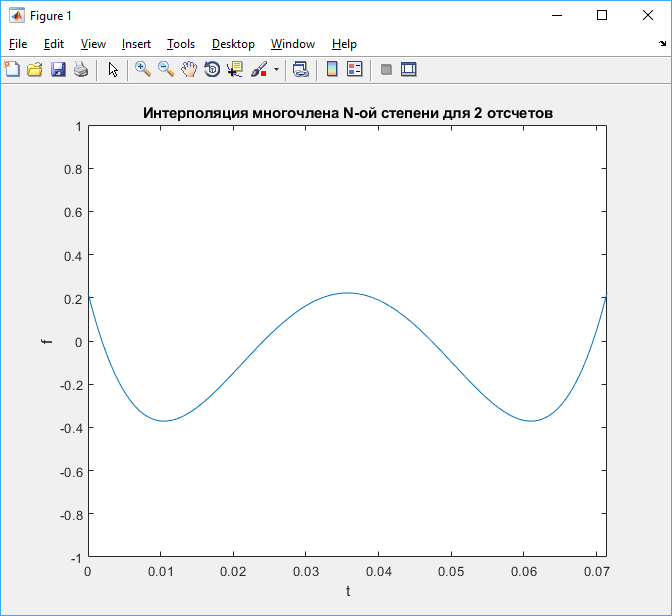


График 5. Интерполяция многочлена N-ой степени для 2 отсчетов

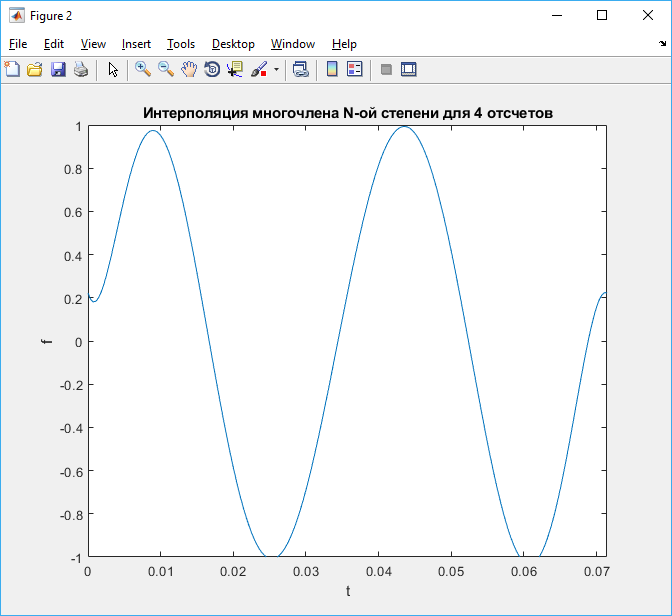


График 6. Интерполяция многочлена N-ой степени для 4 отсчетов

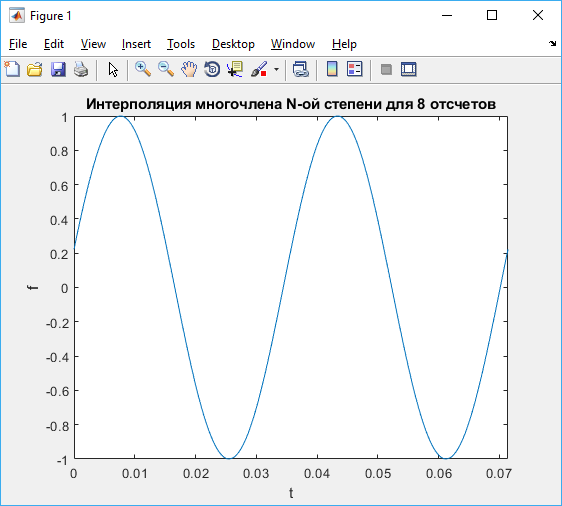


График 7. Интерполяция многочлена N-ой степени для 8 отсчетов

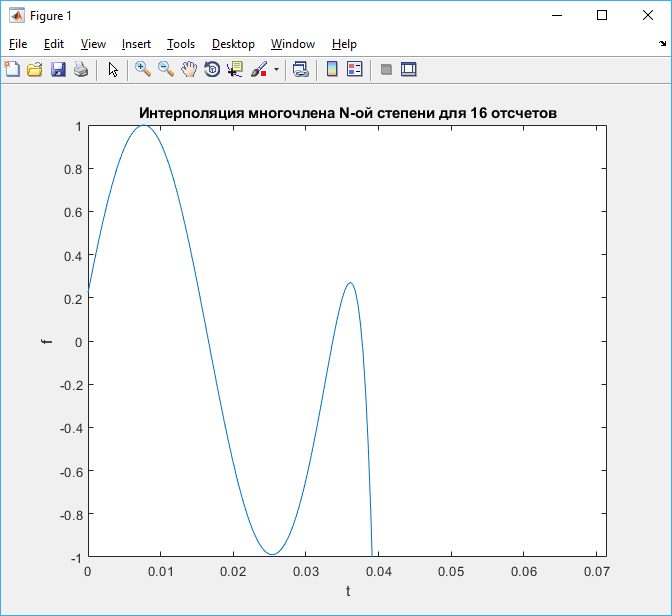


График 8. Интерполяция многочлена N-ой степени для 16 отсчетов

## Листинг 3. Функция pol.

function y = pol(xS, yS, T)

N = length(xS);

z = ones(N, N);

for i = 1:N

for j = 1:N

z(i,j) = xS(i).^(N-j);

end

end

a = z^-1\*yS';

Nn = length(a);

M = length(T);

y = zeros(1, M);

for j= 1:M

s = 0;

for i = 1:Nn

s = s+a(i)\*T(j).^(Nn-i);

end

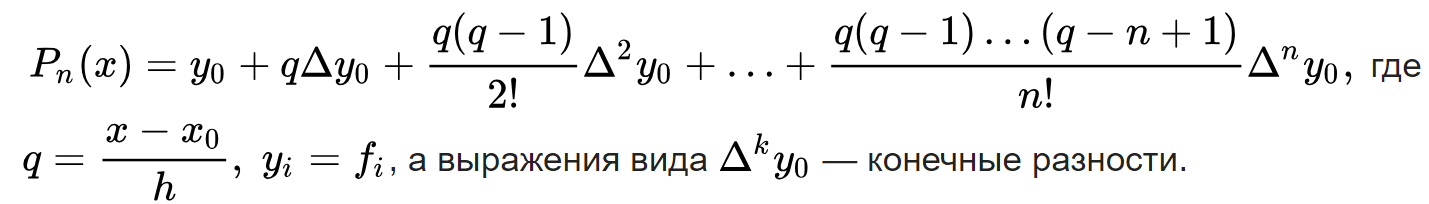
y(j)=s;

end

end

# Функция newton

Следующим способом интерполяции нужно реализовать интерполяцию многочленов Ньютона. Она была реализована в функции newton. Функция принимает точки данных и узел интерполяции, где нужно получить значение. Данный способ реализуется с помощью следующей формулы:



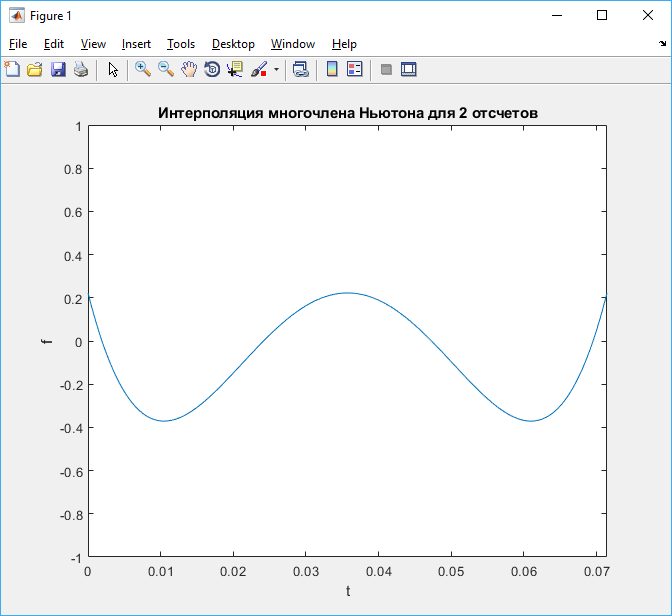


График 9. Интерполяция формулой Ньютона для 2 отсчетов

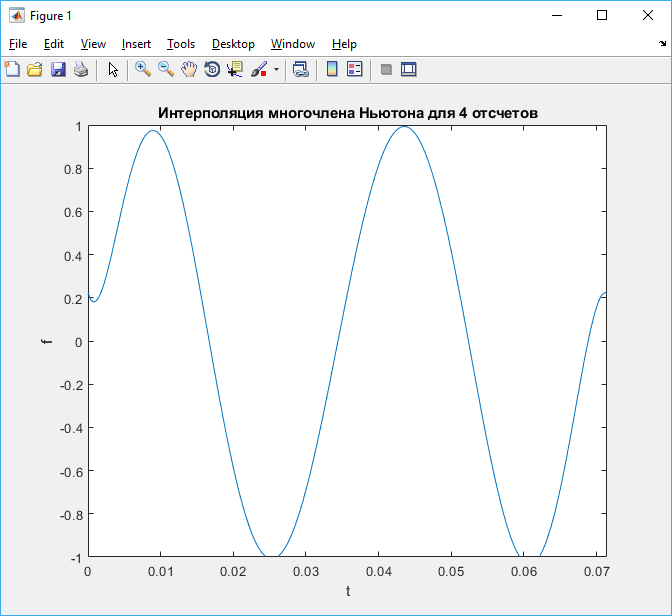


График 10. Интерполяция формулой Ньютона для 4 отсчетов

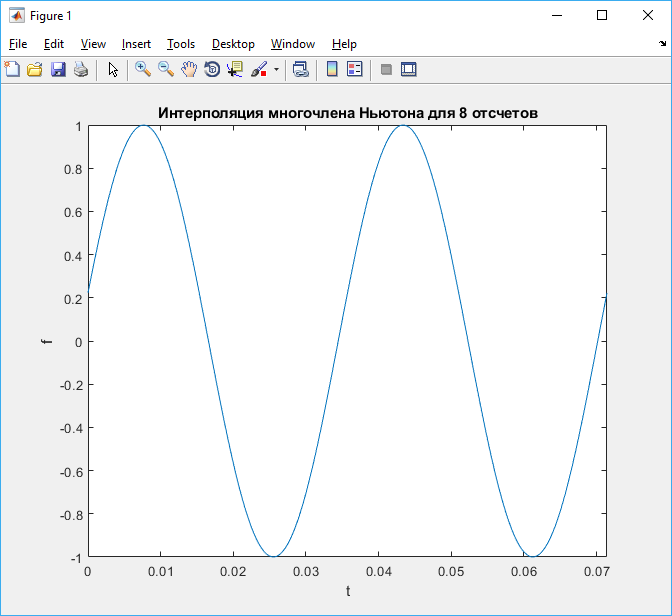


График 11. Интерполяция формулой Ньютона для 8 отсчетов

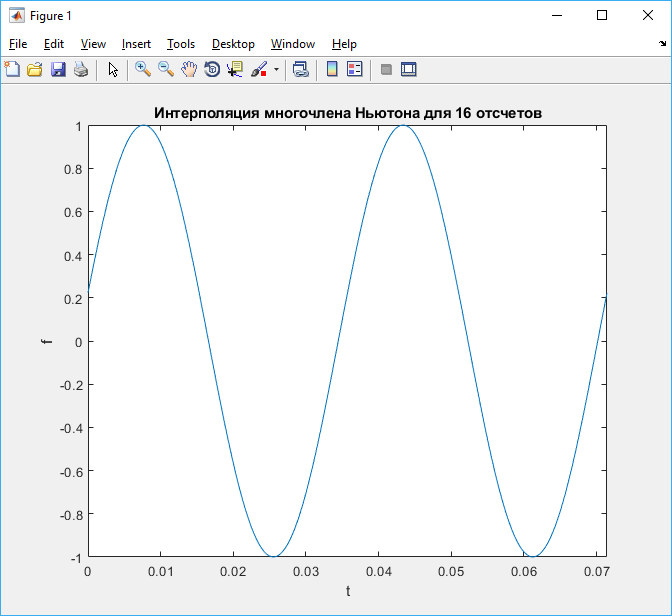


График 12. Интерполяция формулой Ньютона для 16 отсчетов

## Листинг 4. Функция newton

function y = newton(x, xS, yS)

N = length(xS);

% вычисляем разделенные разности

diff = yS;

for k = 1 : N-1

for i = 1: N - k

diff(i) = (diff(i+1) - diff(i)) / (xS(i+k) - xS(i));

end

end

% вычисляем значения интерполяционного полинома в точках xS

% с использованием операции поэлементного умножения .\*

% для получения сразу всех значений полинома yS

y = diff(1) \* ones(size(x));

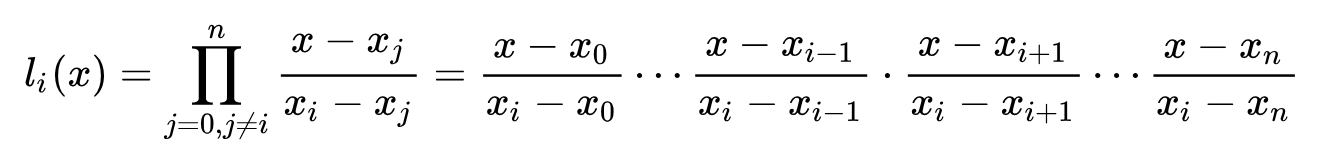
for k = 2 : N

y = diff(k) + (x - xS(k)) .\* y;

end

# Функция lagrange

Следующим способом интерполяции нужно реализовать интерполяцию многочленов Лагранжа. Она была реализована в функции lagrange. Функция принимает точки данных и узел интерполяции, где нужно получить значение. Данный способ реализуется с помощью следующей формулы:



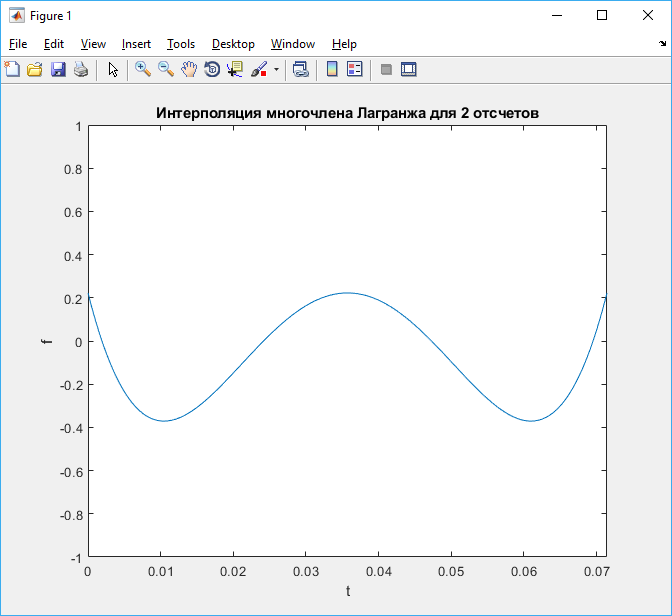


График 13. Интерполяция многочленом Лагранжа для 2 отсчетов

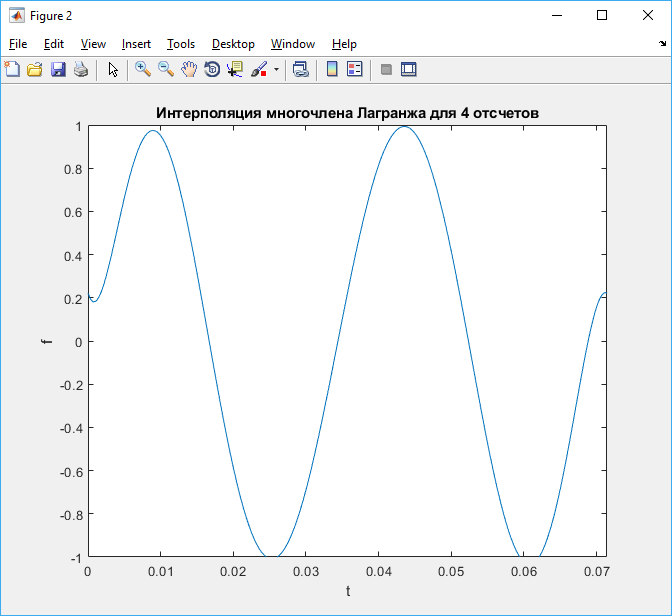


График 14. Интерполяция многочленом Лагранжа для 4 отсчетов

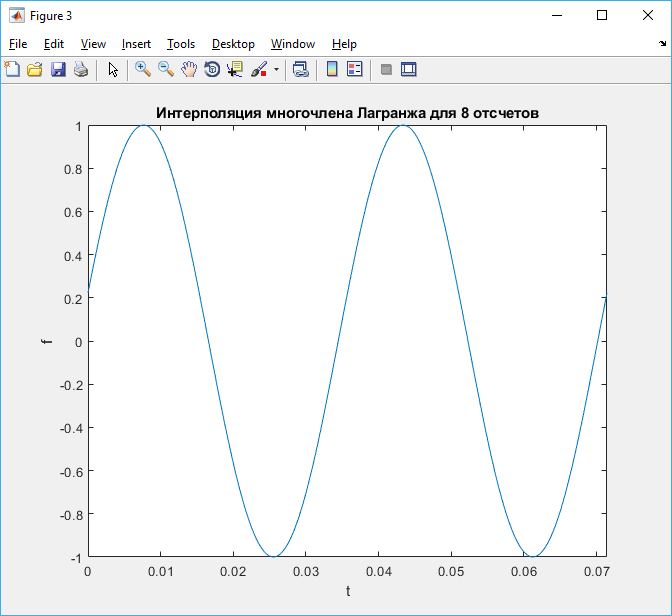


График 15. Интерполяция многочленом Лагранжа для 8 отсчетов

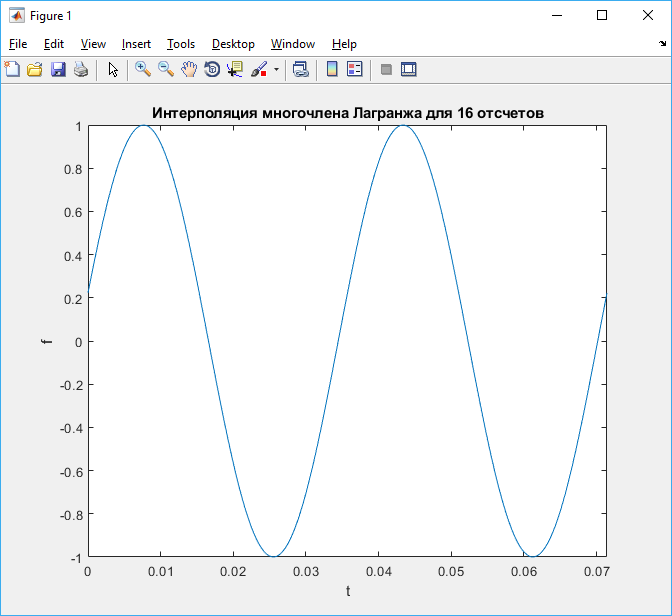


График 16. Интерполяция многочленом Лагранжа для 16 отсчетов

## Листинг 5. Функция lagrange

function y = lagrange(x, xS, yS)

i = 1;

j = 1;

y = 0;

bazis = 1;

while i <= length(xS)

while j <= length(xS)

if i~=j

bazis = bazis \* (x - xS(j))/(xS(i)-xS(j));

end

j = j+1;

end

y = y + yS(i) \* bazis;

bazis = 1;

j = 1;

i = i+1;

end

# Функция spline

Последний способ интерполяции – это кубический сплайн, он уже реализован как стандартная функция matlab под названием spline. Функция принимает точки данных и вектор отсчетов, где нужно найти интерполированные значения.

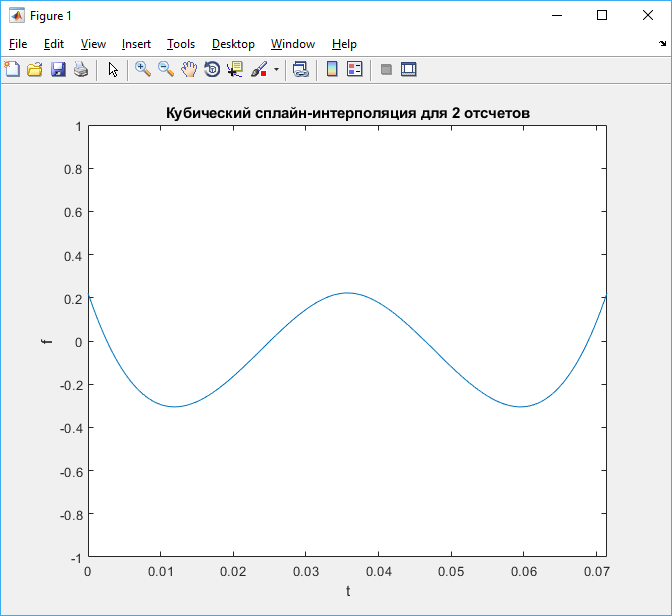


График 17. Интерполяция кубическим сплайном для 2 отсчетов

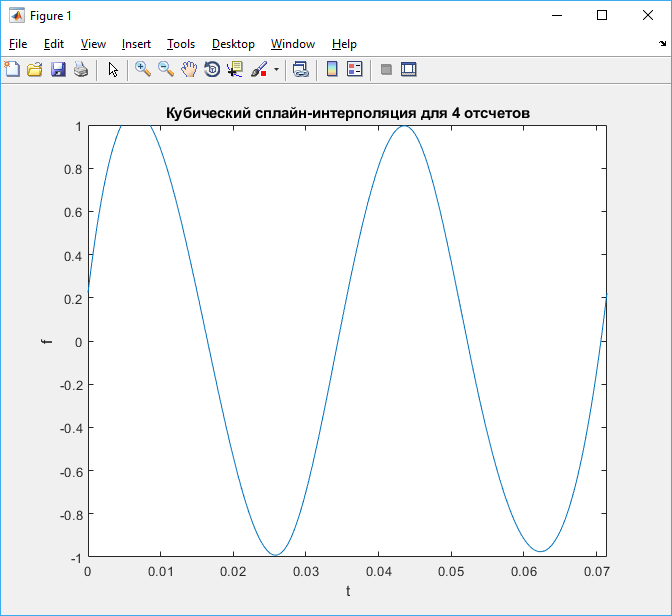


График 18. Интерполяция кубическим сплайном для 4 отсчетов

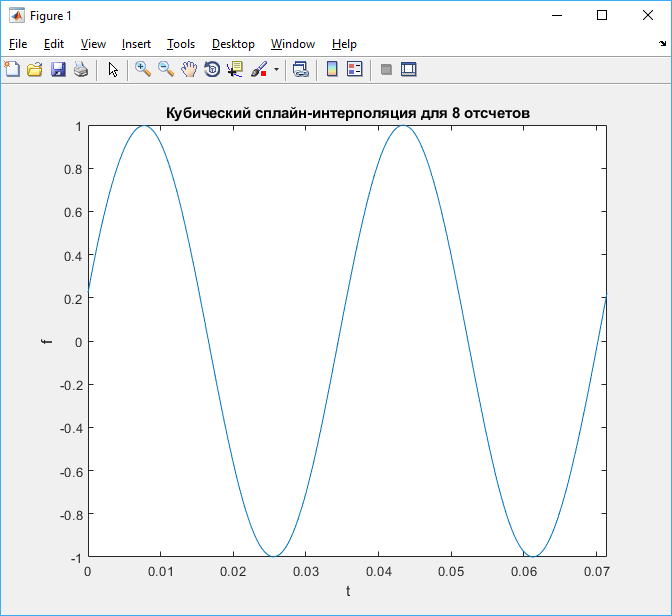


График 19. Интерполяция кубическим сплайном для 8 отсчетов

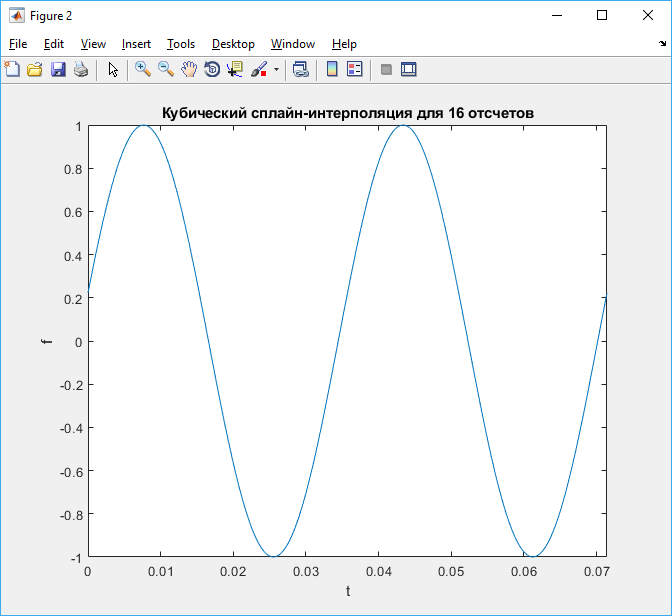


График 20. Интерполяция кубическим сплайном для 16 отсчетов

# Интерполирование

После интерполяции несколькими способами, можно сравнить точность решения задачи у каждого метода и узнать, как зависит точность решения задачи интерполяции от числа табличных отсчетов. В данном случае было решено взять модуль разностей между значениями оригинального сигнала и значениями сигналов после интерполирования. Для наглядности в таблицу 1 были вынесены средние значения для всех модулей разности на протяжении всей выбранной длительности сигнала. Интересный факт, что для 2-ух точек отсчета при выборе любого метода интерполяции в результате получается одно и то же значение.

Таблица 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 2 отчета | 4 отсчета | 8 отсчетов | 16 отсчетов |
| Линейная | 0,6173673469 | 0.1323285846 | 0,0320344955 | 0,0080474541 |
| N-ой степени | 0,6173673469 | 0,0552653920 | 0,0000352562 | 66267,484671 |
| Ньютон | 0,6173673469 | 0,0552653920 | 0,0000043909 | 5e-11 |
| Лагранж | 0,6173673469 | 0,0552653920 | 0,0000043909 | 6e-11 |
| Сплайн | 0,6173673469 | 0,0419880287 | 0,0011076053 | 0,0000336685 |

При 2 отсчетах все виды показали себя одинаково, при 4 – сплайн, при 8 – Ньютон и Лагранж, при 16 – Ньютон незначительно лучше Лагранжа

## Листинг 6. Использование интерполяций

%отсчеты для интеполирования

T = 0 : 1/2800 :1/14;

line2 = zeros(1, length(T));

new2 = line2;

lagr2 = line2;

line4 = line2;

new4 = line2;

lagr4 = line2;

line8 = line2;

new8 = line2;

lagr8 = line2;

line16 = line2;

new16 = line2;

lagr16 = line2;

%значения после интерполяции

for i = 1 : length(T)

line2(i) = linearInterpolation(T(i), t2, f2);

line4(i) = linearInterpolation(T(i), t4, f4);

line8(i) = linearInterpolation(T(i), t8, f8);

line16(i) = linearInterpolation(T(i), t16, f16);

new2(i) = newton(T(i), t2, f2);

new4(i) = newton(T(i), t4, f4);

new8(i) = newton(T(i), t8, f8);

new16(i) = newton(T(i), t16, f16);

lagr2(i) = lagrange(T(i), t2, f2);

lagr4(i) = lagrange(T(i), t4, f4);

lagr8(i) = lagrange(T(i), t8, f8);

lagr16(i) = lagrange(T(i), t16, f16);

end

spl2 = spline(t2, f2, T);

spl4 = spline(t4, f4, T);

spl8 = spline(t8, f8, T);

spl16 = spline(t16, f16, T);

pol2 = pol(t2, f2, T);

pol4 = pol(t4, f4, T);

pol8 = pol(t8, f8, T);

pol16 = pol(t16, f16, T);

figure;

plot(T, spl16);

axis ([0 1/14 -1 1]);

title ('Кубический сплайн-интерполяция для 16 отсчетов');

xlabel('t');

ylabel('f');

f = sin((2\*pi\*N\*T)+((2\*pi)./N));

%%

%проверка точности

accLine = zeros(1,4);

accNew = accLine;

accLagr = accLine;

accSpl = accLine;

accPol = accLine;

accLine(1) = mean(abs(f-line2));

accLine(2) = mean(abs(f-line4));

accLine(3) = mean(abs(f-line8));

accLine(4) = mean(abs(f-line16));

accNew(1) = mean(abs(f-new2));

accNew(2) = mean(abs(f-new4));

accNew(3) = mean(abs(f-new8));

accNew(4) = mean(abs(f-new16));

accLagr(1) = mean(abs(f-lagr2));

accLagr(2) = mean(abs(f-lagr4));

accLagr(3) = mean(abs(f-lagr8));

accLagr(4) = mean(abs(f-lagr16));

accSpl(1) = mean(abs(f-spl2));

accSpl(2) = mean(abs(f-spl4));

accSpl(3) = mean(abs(f-spl8));

accSpl(4) = mean(abs(f-spl16));

accPol(1) = mean(abs(f-pol2));

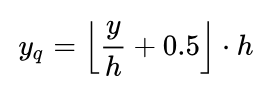
accPol(2) = mean(abs(f-pol4));

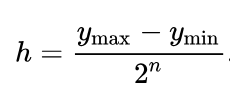
accPol(3) = mean(abs(f-pol8));

accPol(4) = mean(abs(f-pol16));

# Оцифровка сигнала, функция adc

Функция adc (АЦП) является математической моделью аналого-цифрового преобразователя, она принимает значения дискретного сигнала и число разрядов и возвращает значения цифрового сигнала. Преобразование было реализовано с помощью следующих формул:





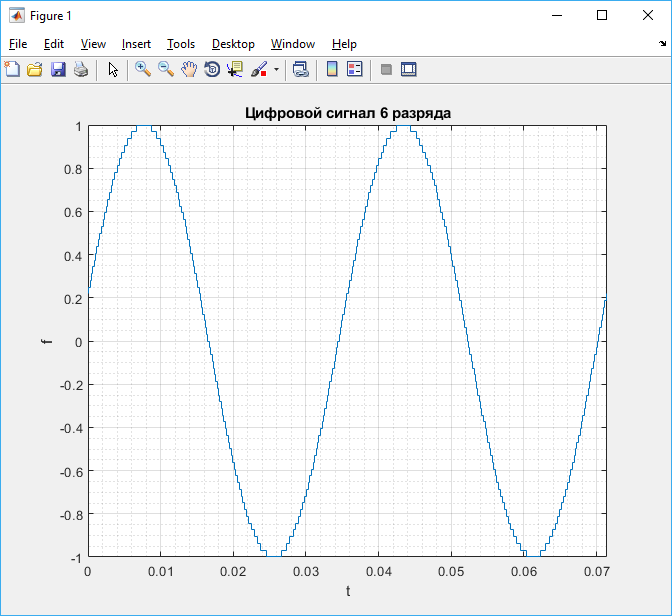


График 21. Оцифровка сигнала для 6 разрядного кода

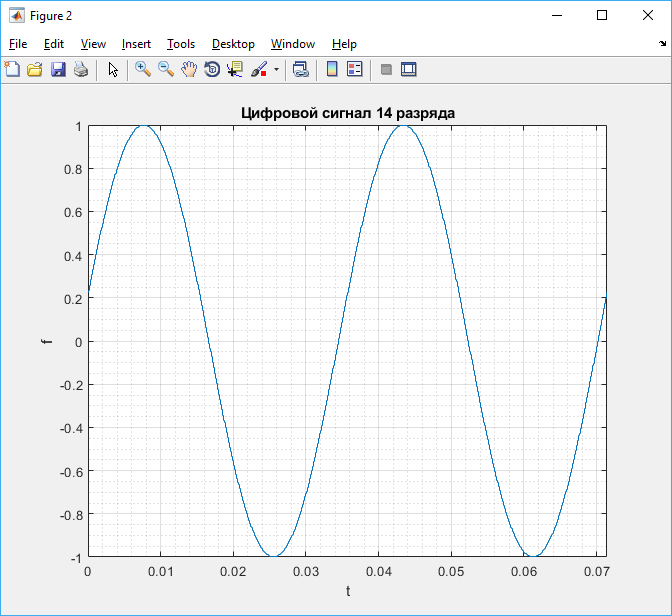


График 22. Оцифровка сигнала для 14 разрядного кода

## Листинг 7. Функция adc

function res = adc(xS, yS, N)

res = zeros(1, length(xS));

h = (max(yS) - min(yS)) / 2.^N;

for i = 1 : length(xS)

res(i) = floor(yS(i)/h + 0.5) \* h;

end

# Шум

Шум находится как разность оригинального сигнала и сигнала после обработки.

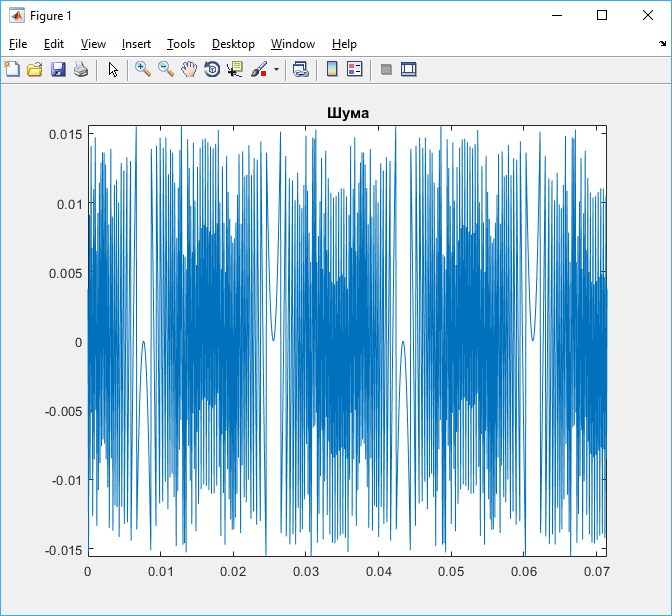


График 23. Визуализация шума для 6 разрядного кода

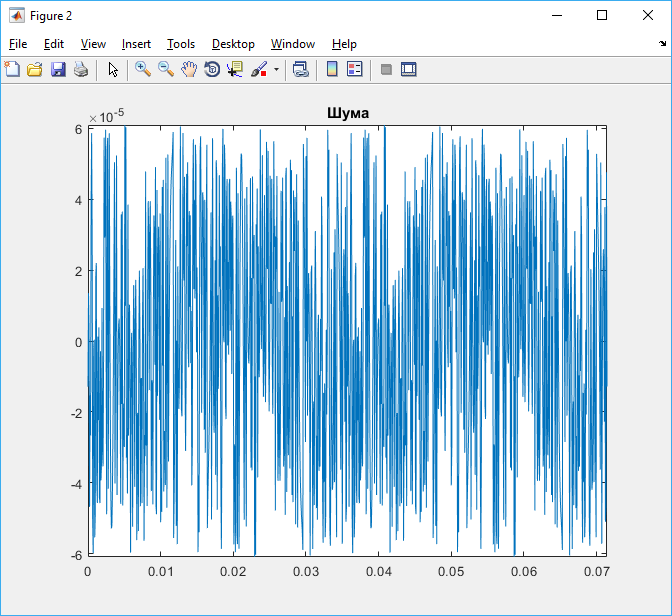


График 24. Визуализация шума для 14 разрядного кода

6 разрядов

Дисперсия = 7,76985355583373e-05

СКО = 0,00881467728044182

14 разрядов

Дисперсия= 1,20011994900572e-09

СКО= 3,46427474228838e-05

# Вывод

Были получены практические знания о интерполяции, дискретизации и квантовании.