МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Институт кибернетики

Направление 09.03.01 «Информатика и вычислительная техника»

Кафедра Информационных систем и технологий

Отчет по лабораторным работам

По дисциплине: «Цифровая обработка сигналов»

Выполнил:

студент гр. 8В4Б \_\_\_\_\_\_\_\_\_\_\_\_\_ XXXXXXX.

\_\_\_.\_\_\_.\_\_\_\_\_\_

Проверил:

доцент кафедры ИСТ \_\_\_\_\_\_\_\_\_\_\_\_\_ Хамухин А.А.

\_\_\_.\_\_\_.\_\_\_\_\_\_

[Цель работы 2](#_Toc501584526)

[1. Работа со звуком 2](#_Toc501584527)

[1.1. Анализ посредством ПО Matlab 18](#_Toc501584528)

[2. Вейвлет-преобразования 20](#_Toc501584529)

[2.1. Главные признаки вейвлета 20](#_Toc501584530)

[2.2. Виды поддерживаемых вейвлетов в Matlab 22](#_Toc501584531)

[2.3. Выполнение sym3- и bior2.8-вейвлет преобразования (по варианту). 25](#_Toc501584532)

[3. Зашумление сигнала 27](#_Toc501584533)

[3.1. Удаление шума 29](#_Toc501584534)

[4. Фильтрация изображений 31](#_Toc501584535)

[5. Список самостоятельных работ 40](#_Toc501584536)

[6. Вывод 41](#_Toc501584537)

[7. Список литературы 42](#_Toc501584538)

# Цель работы

Ознакомление с теорией и практическими навыками цифровой обработки сигналов.

# Работа со звуком

Перед тем, как анализировать запись голоса, необходимо ознакомиться с основами анализа звука. Лучше всего это сделать при помощи работы с «эталонным» звуком. В данном случае используется запись звучания ноты «Ля» первой октавы на фортепиано.

Откроем аудиофайл в программе Audacity.



Рисунок 1. Открытие файла

Для того, чтобы увидеть частотный спектр аудиозаписи, выполним следующие действия: выделим всю аудиодорожку (нажатием Ctrl+a), затем выберем пункт «Построить график спектра» в меню «Анализ». После выполнения этой команды появится окно «Частотный анализ», отображающее частотный спектр аудиозаписи.

Для анализа частотных характеристик имеет смысл произвести некоторую предварительную настройку. Так, при использовании логарифмического масштаба оси графическое отображение спектра становится более наглядным. Для этого в окне «Частотный анализ» напротив пункта «Ось» установим логарифмический масштаб. Продемонстрируем графики получившегося спектра с помощью различных оконных преобразований Фурье, меняя параметры «Алгоритм» и «Размер окна».

В основе цифрового спектрального анализа лежит аппарат дискретного преобразования Фурье (ДПФ). При этом ДПФ имеет высокоэффективные быстрые алгоритмы (FFT, БПФ). Однако при использовании ДПФ часто возникают трудности обусловленные конечностью интервала обработки – алгоритмы разрабатывались для бесконечного сигнала, реальные же сигналы конечны. Оконные функции помогают «бороться» с конечностью сигнала – вместо спектра исходного сигнала после оконного преобразования мы получаем, по сути, спектр произведения исходного сигнала на оконную функцию, а в тех моментах времени, где исходный сигнал не существует, в простейшем случае будут дописаны «нули».

Однако у оконных функций есть серьезный недостаток – они позволяют лишь оценить спектр исходного сигнала, т.к. само по себе использование этих функций ведет к искажению сигнала и, как следствие, спектра сигнала.

Далее приведена таблица с параметрами некоторых оконных функций, а также их графики и спектры. Использовались данные и изображения из [2].

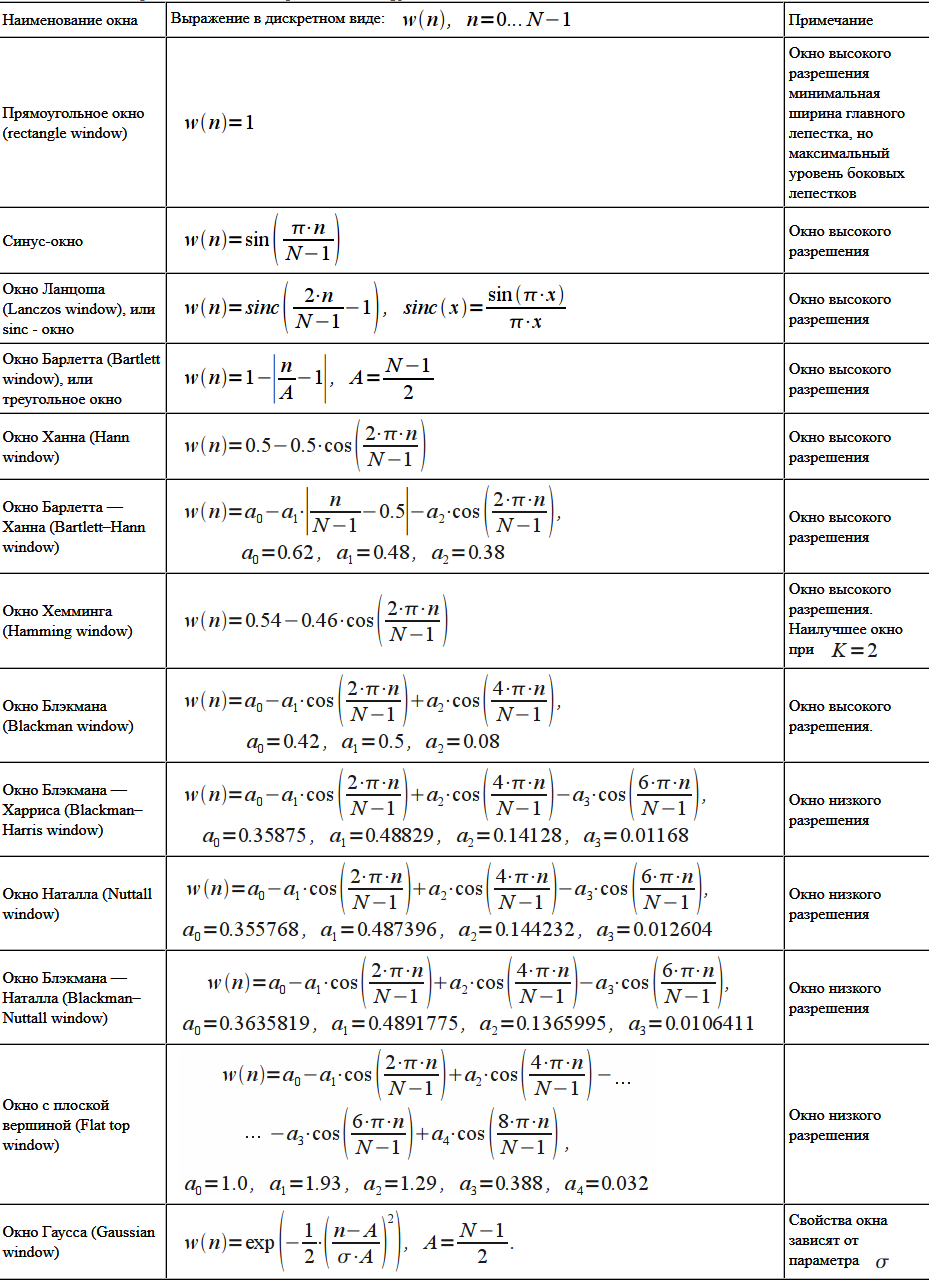


Рисунок 2. Свойства некоторых оконных функций

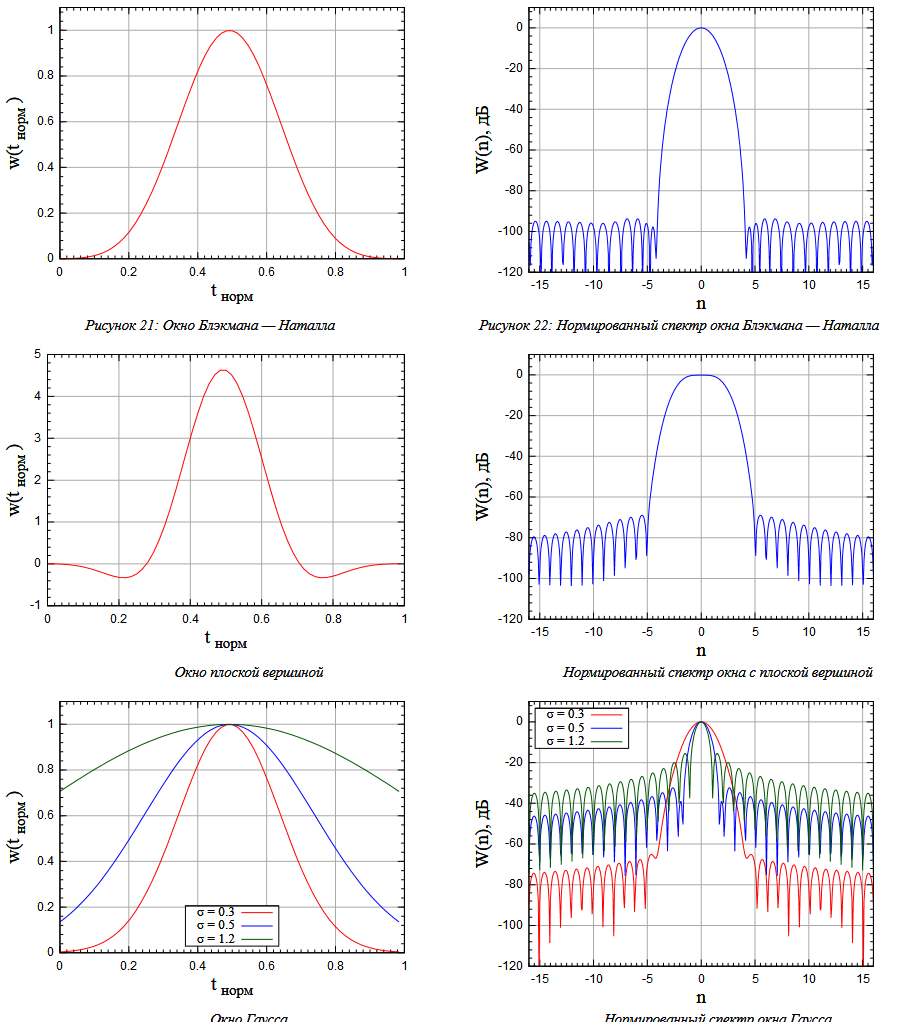


Рисунок 3. Графики и спектры оконных функций



Рисунок 4. Графики и спектры оконных функций

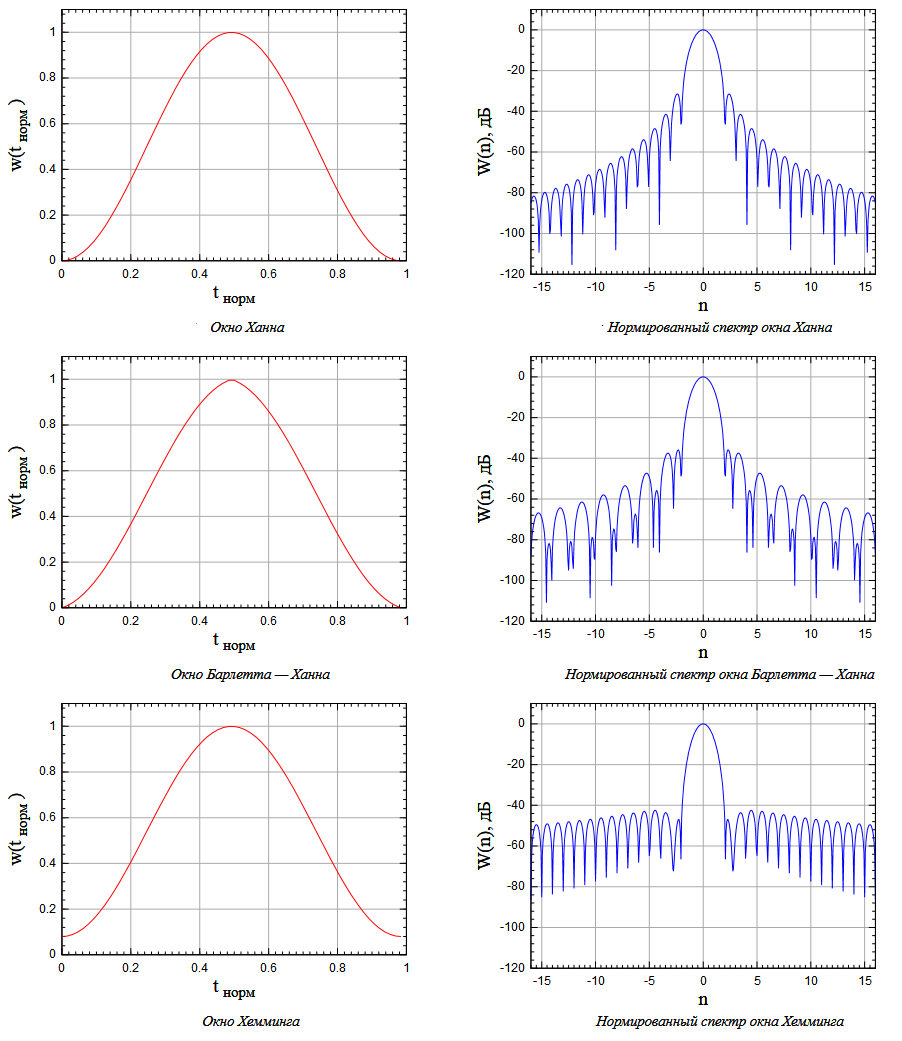


Рисунок 5. Графики и спектры оконных функций

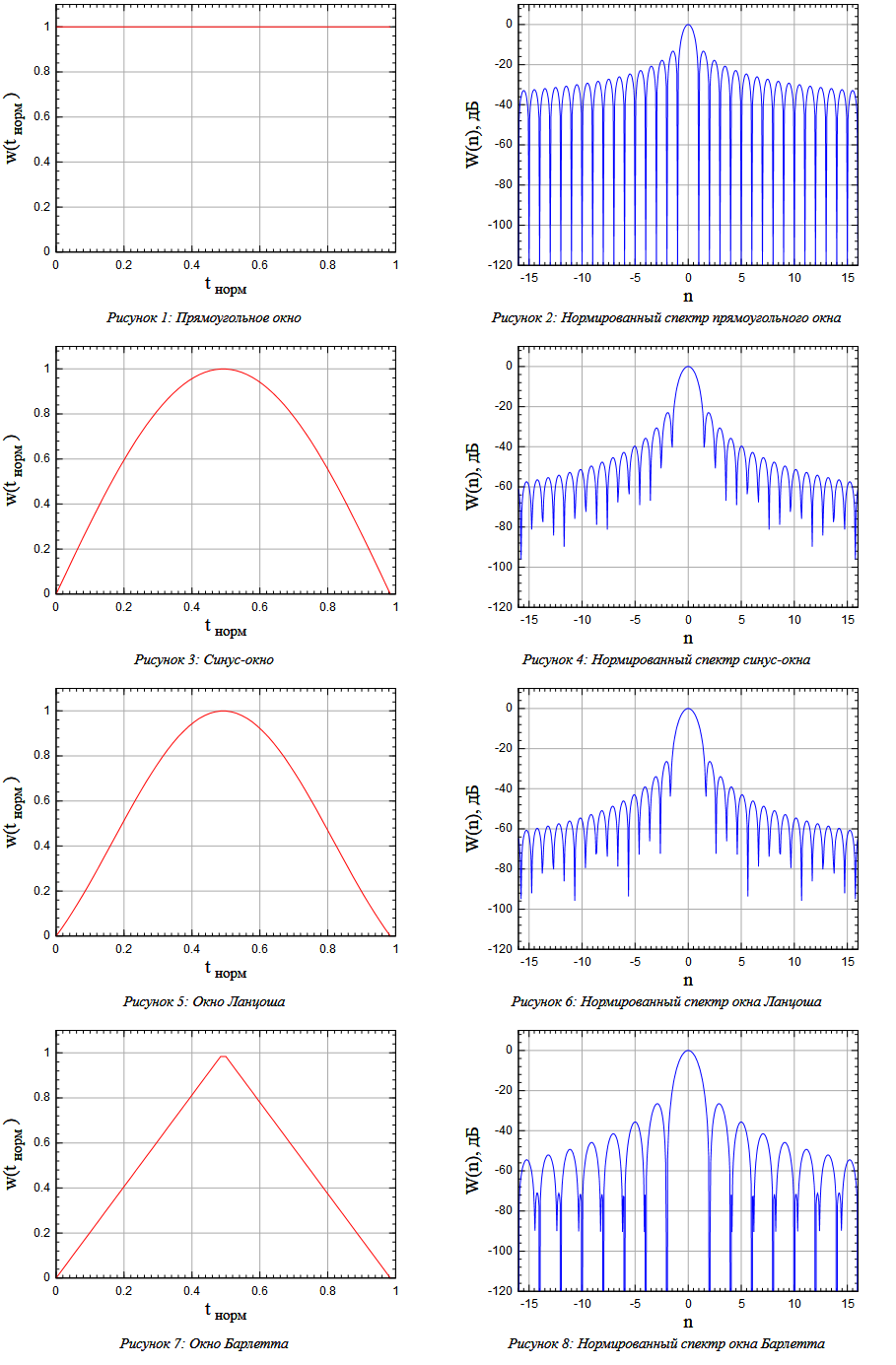


Рисунок 6. Графики и спектры оконных функций

Далее приведены результаты исследования частотного спектра аудиосигнала в программе Audacity с использованием различных оконных функций.

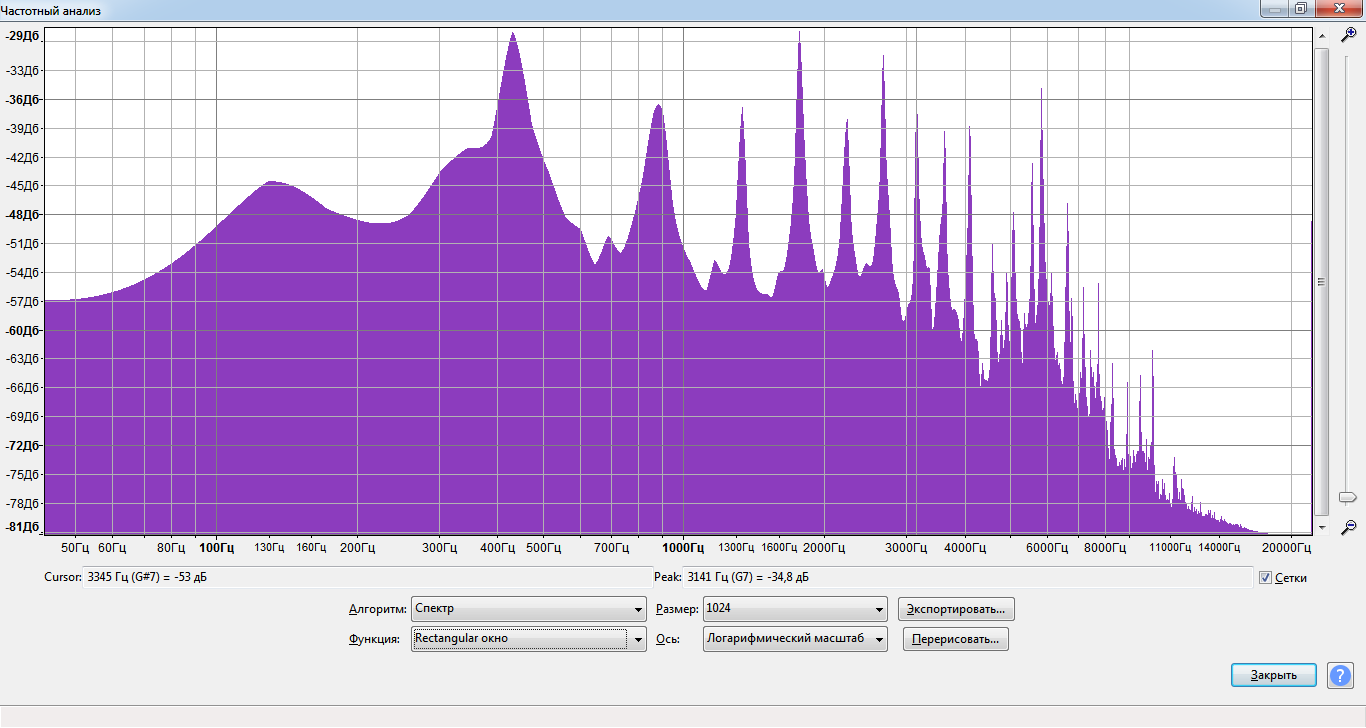


Рисунок 7. Спектр сигнала, полученный при использовании прямоугольного окна

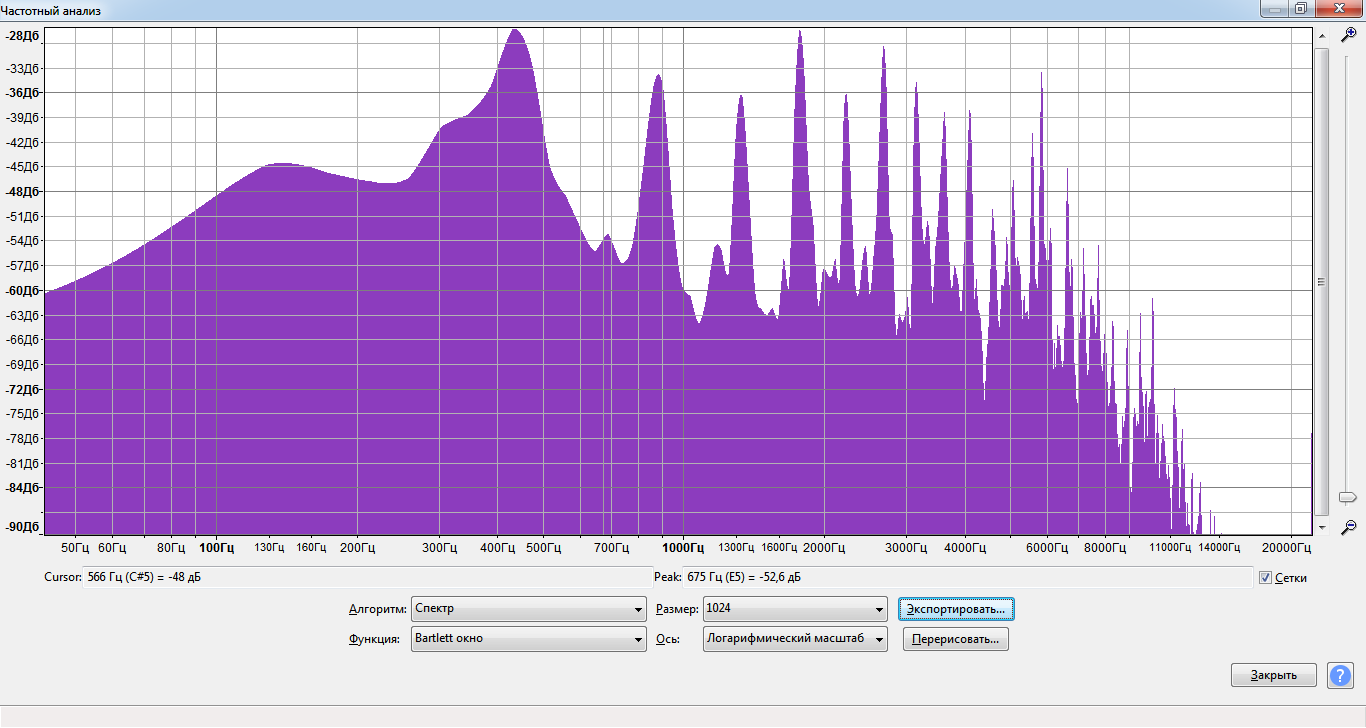


Рисунок 8. Спектр сигнала, полученный при использовании окна Барлетта

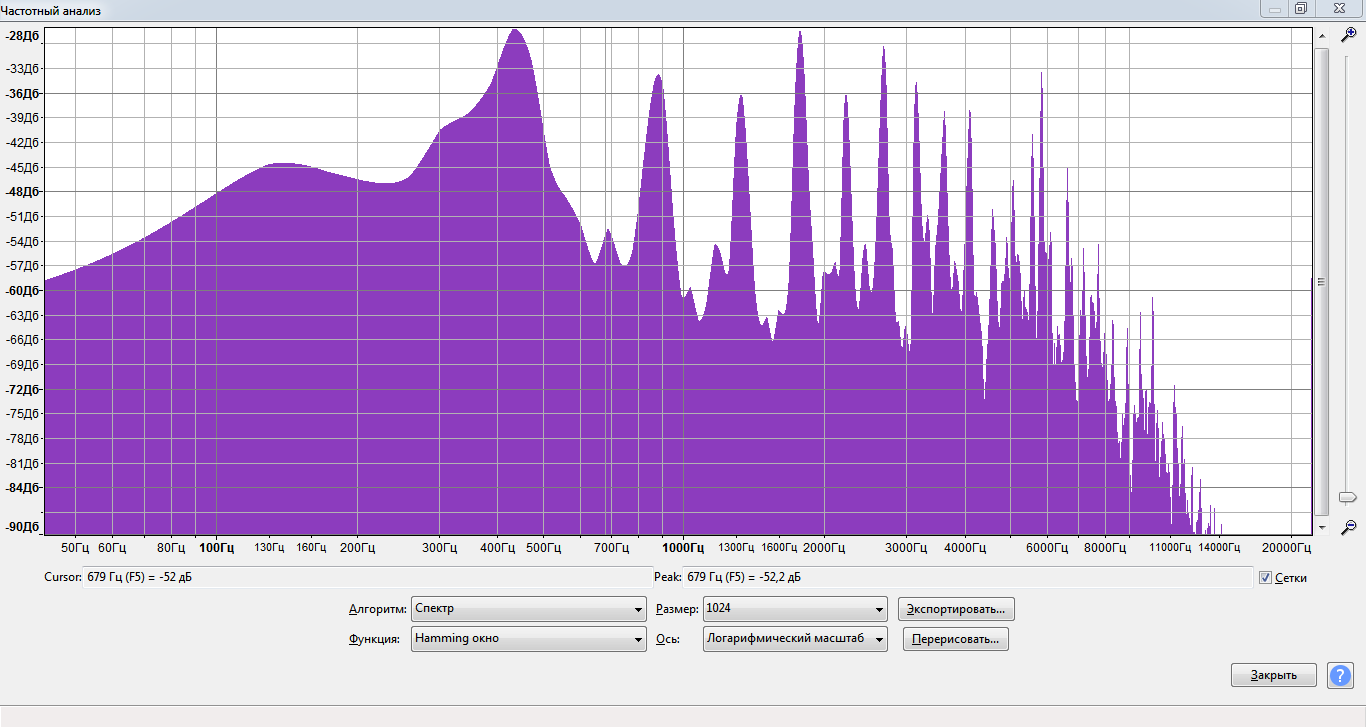


Рисунок 9. Спектр сигнала, полученный при использовании окна Хэмминга

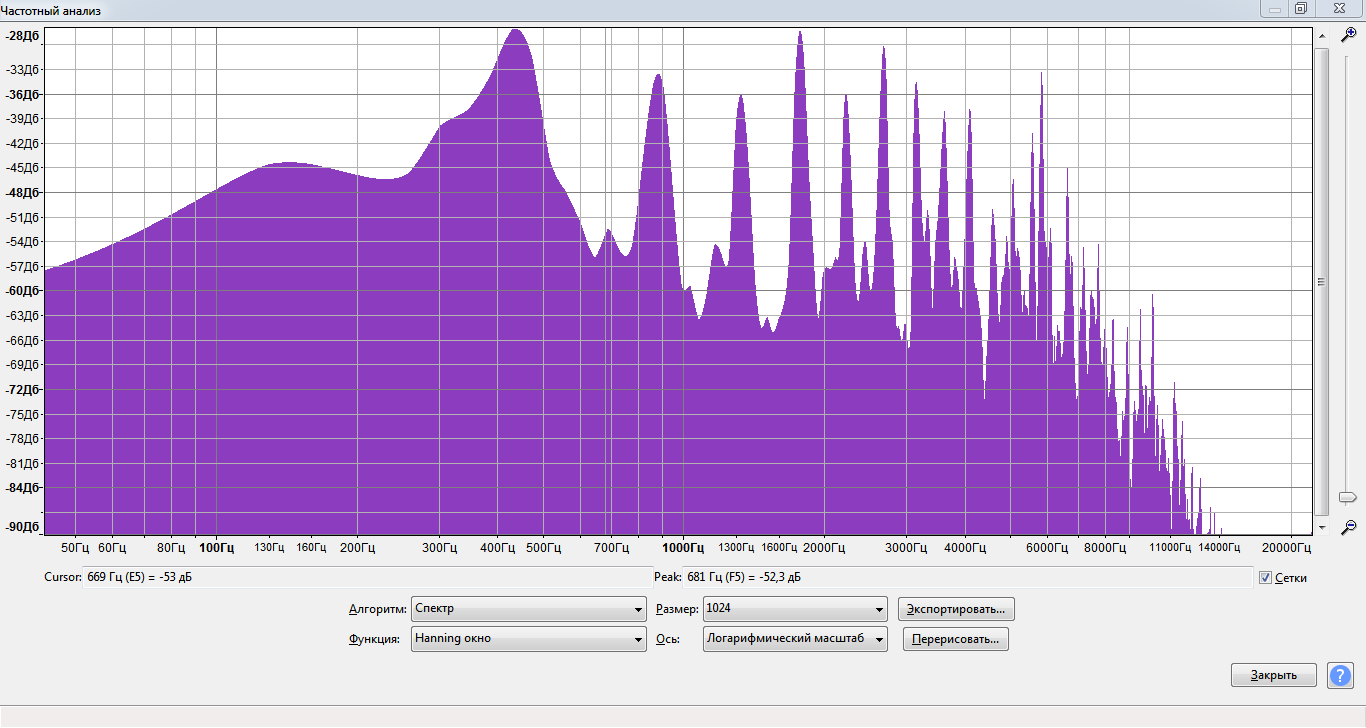


Рисунок 10. Спектр сигнала, полученный при использовании окна Хэннинга

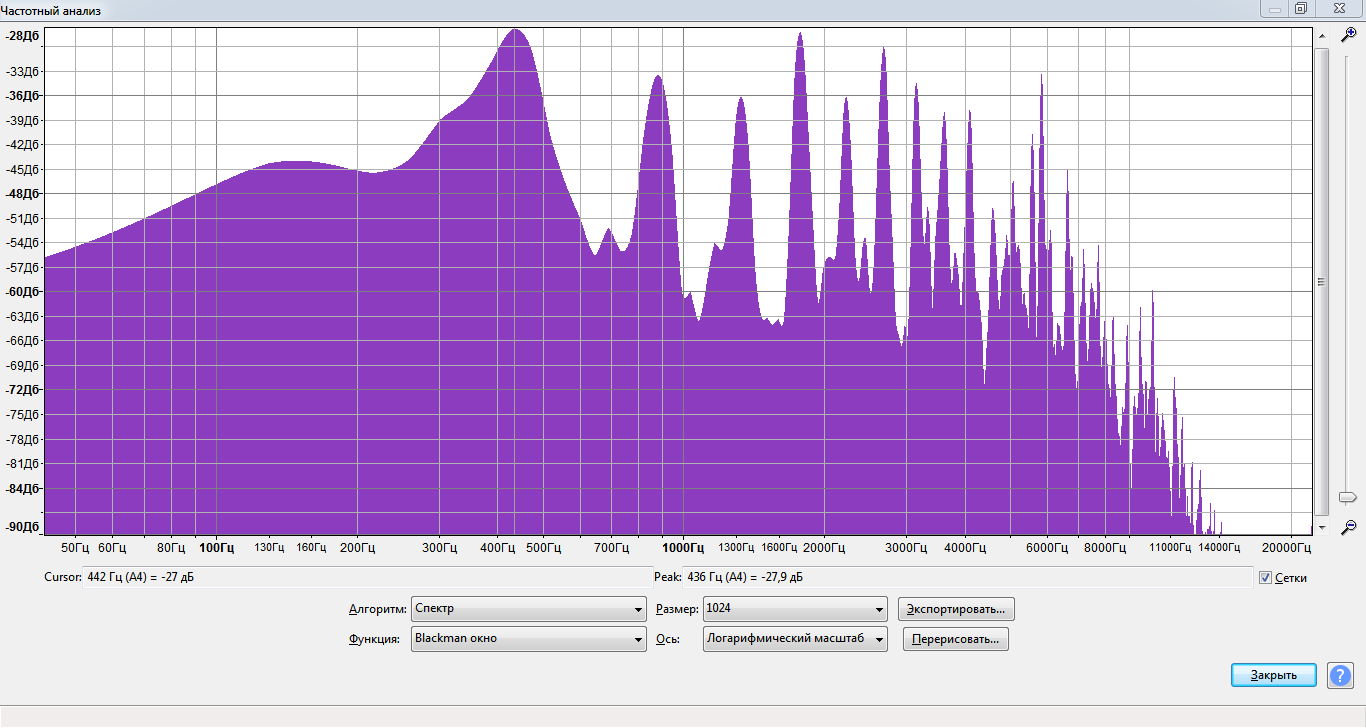


Рисунок 11. Спектр сигнала, полученный при использовании окна Блэкмэна

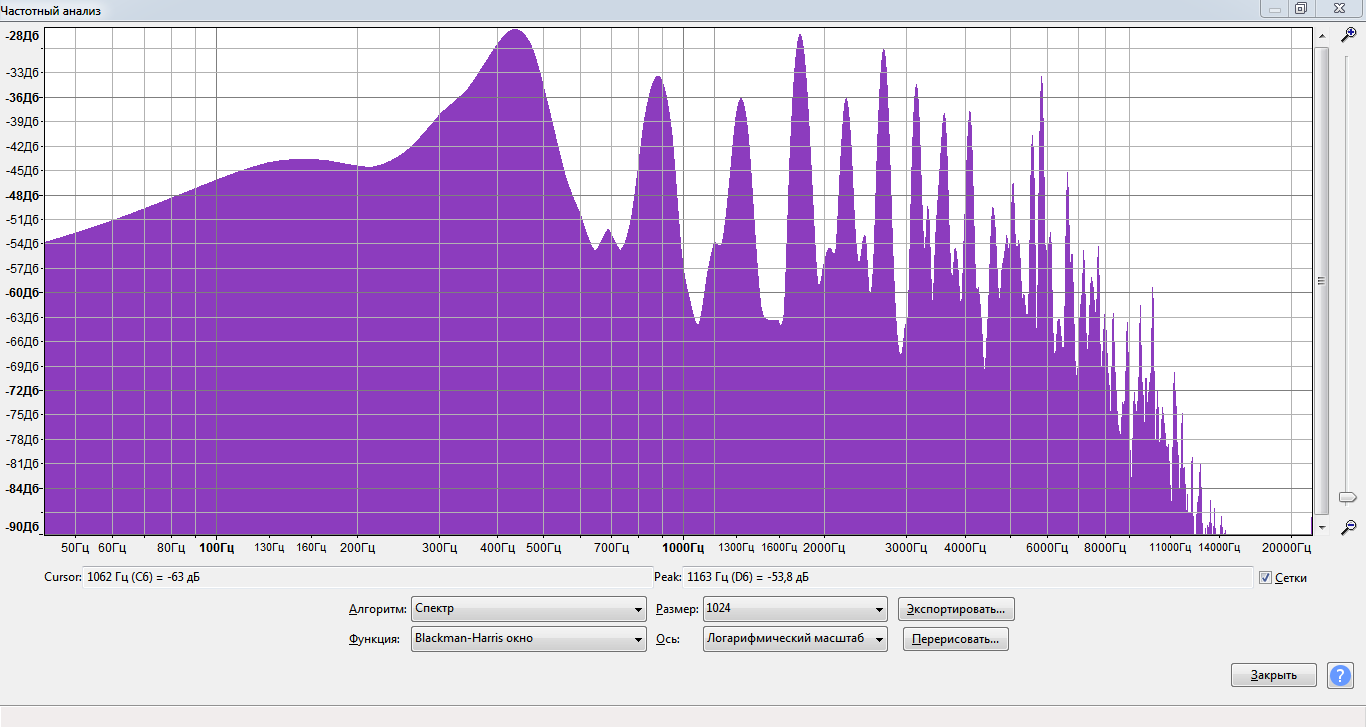


Рисунок 12. Спектр сигнала, полученный при использовании окна Блэкмэна-Харриса

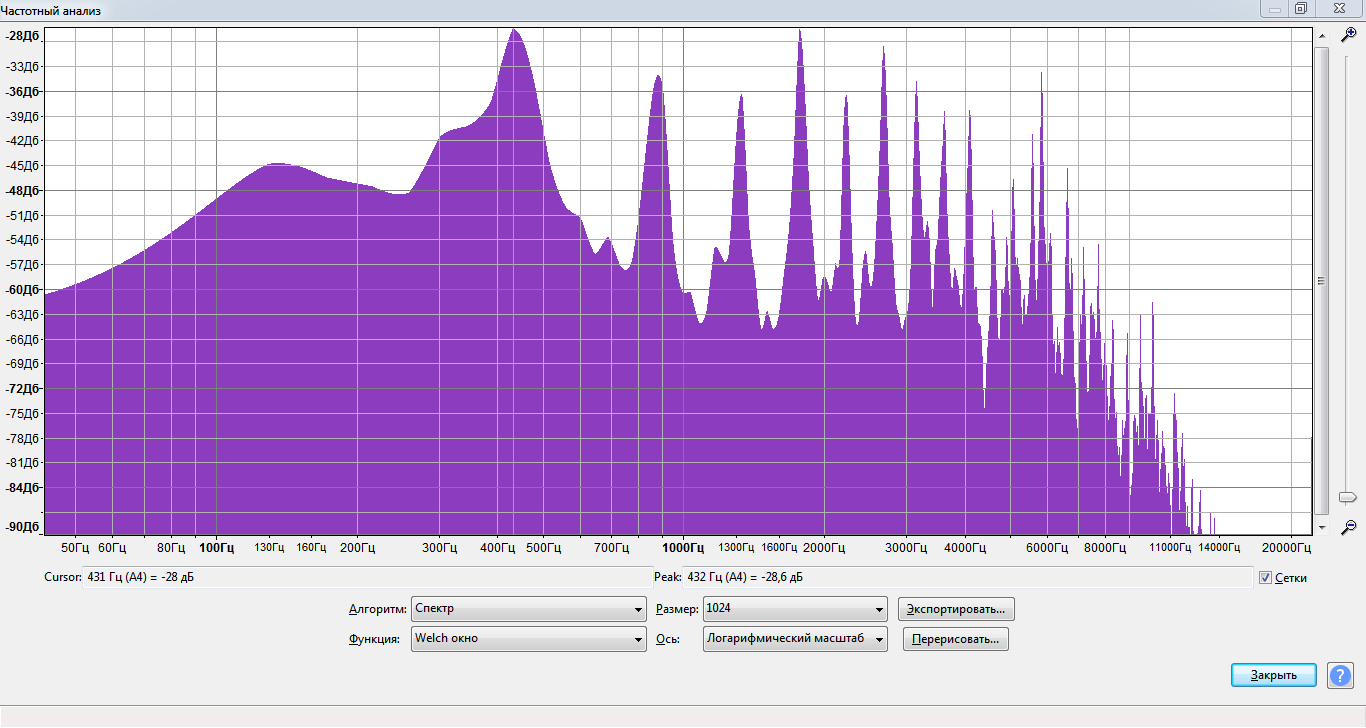


Рисунок 13. Спектр сигнала, полученный при использовании окна Уэлча

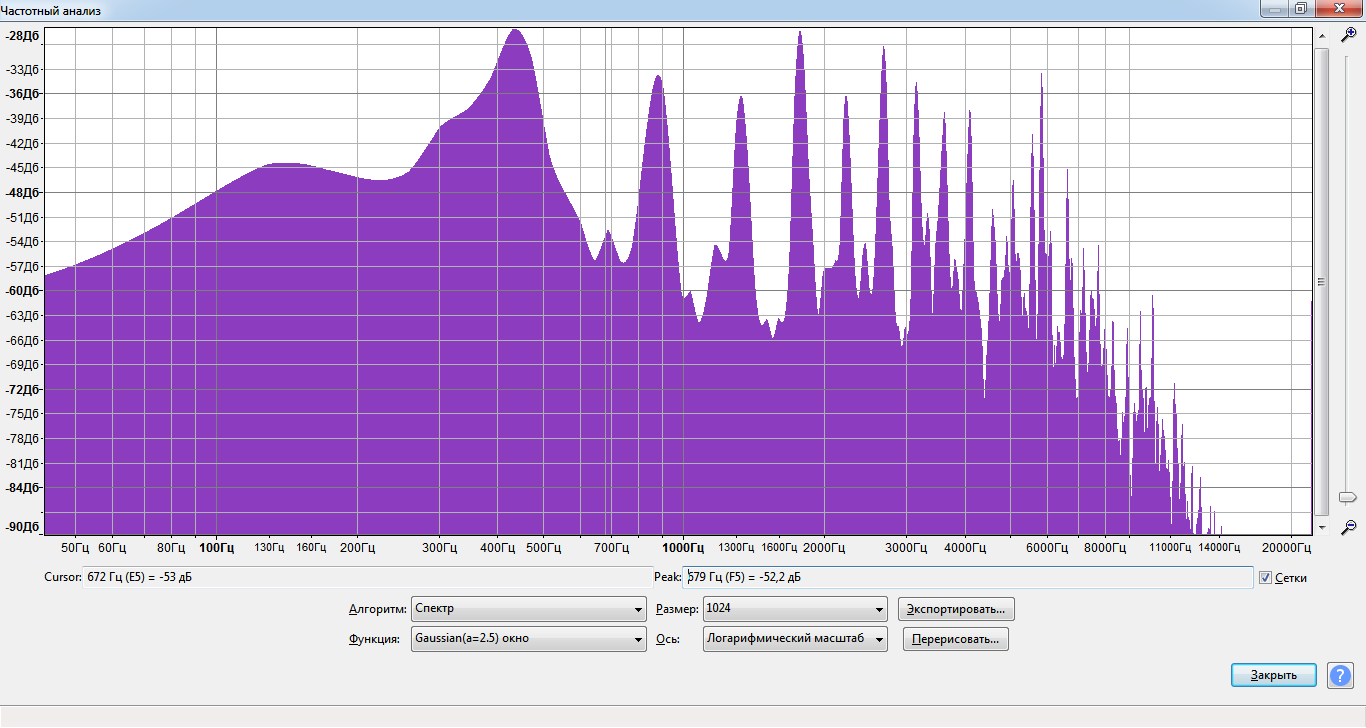


Рисунок 14. Спектр сигнала, полученный при использовании окна Гаусса (а = 2.5)

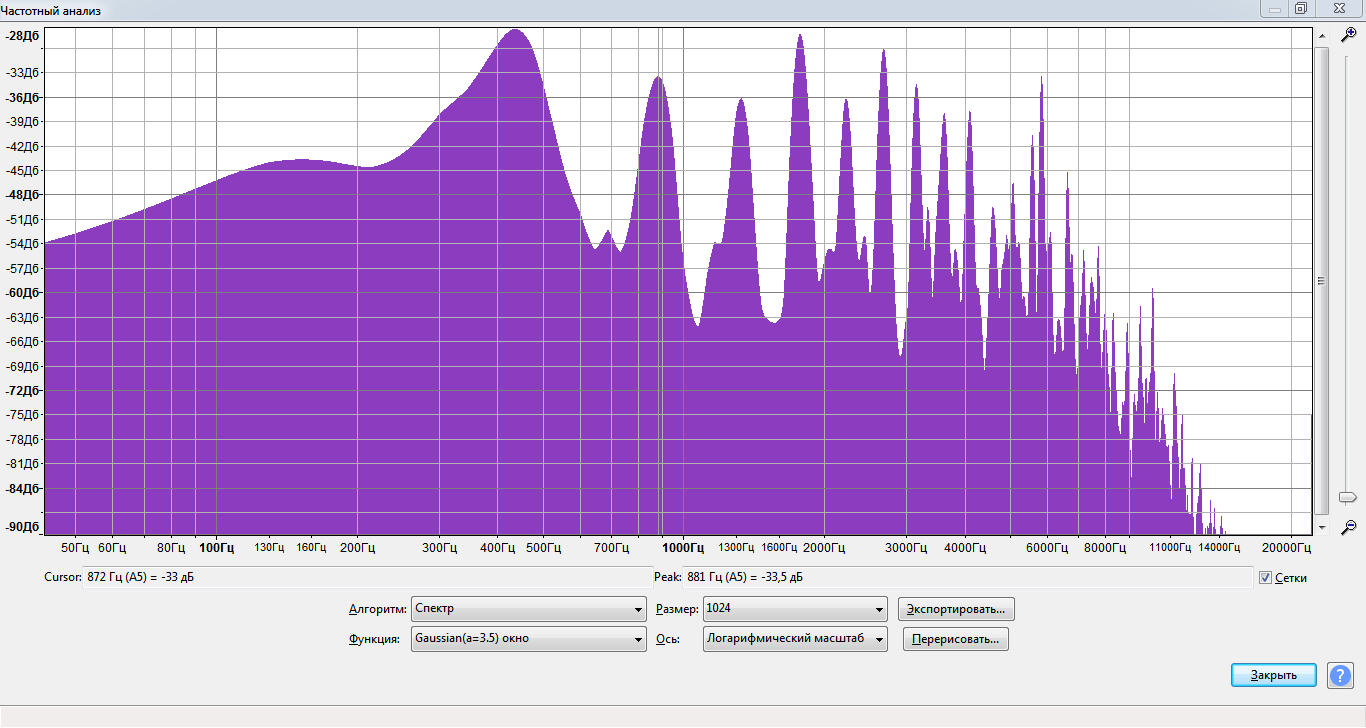


Рисунок 15. Спектр сигнала, полученный при использовании окна Гаусса (а = 3.5)

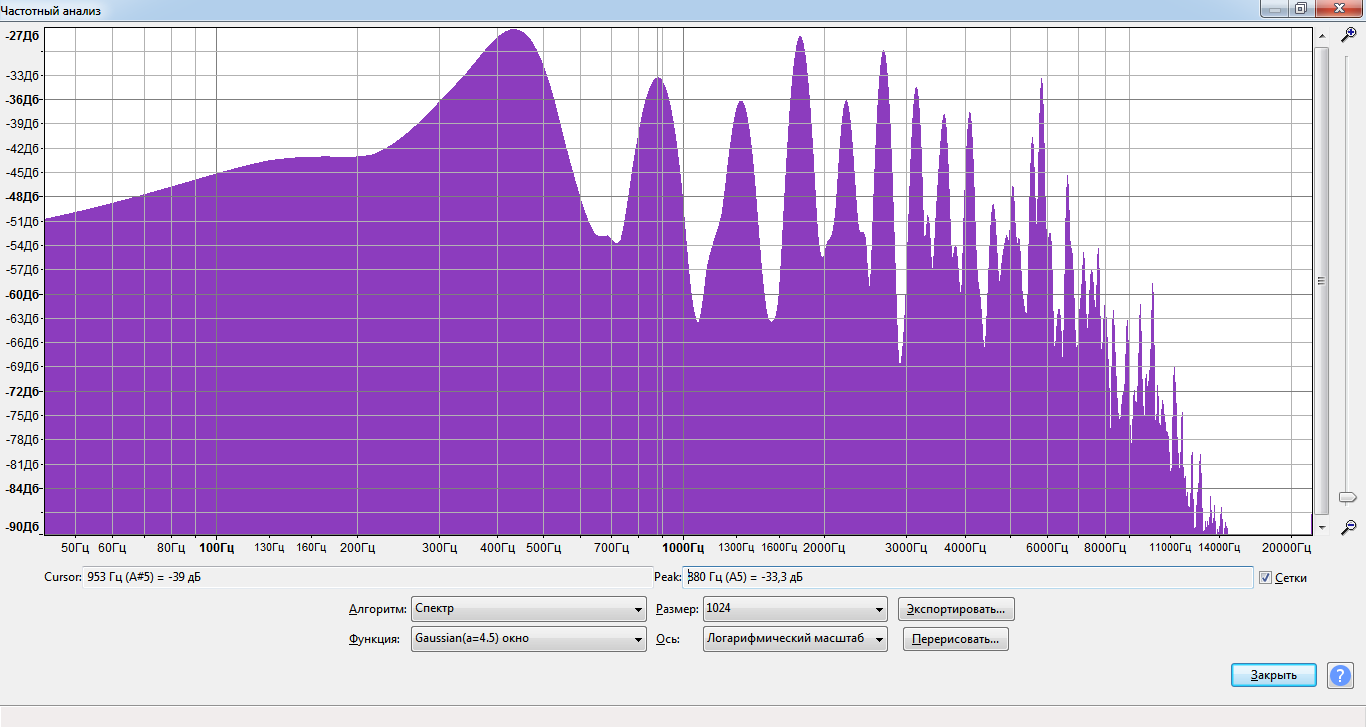


Рисунок 16. Спектр сигнала, полученный при использовании окна Гаусса (а = 4.5)

Как видно по полученным графикам спектров сигнала, в отличие от чистой ноты «Ля» первой октавы (например, при записи звучания камертона), данная аудиозапись содержит не только основной тон (440 Гц для ноты «Ля» первой октавы), но и обертоны – призвуки, частоты которых кратны частоте основного тона. Это хорошо заметно по регулярным максимумам на частотной диаграмме. В таблице 1 приведены значения основного тона, снятые при применении каждой оконной функции.

Таблица 1. Значения основного тона аудиосигнала при применении разных оконных функций

|  |  |  |
| --- | --- | --- |
| Тип окна | Частота, Гц | Амплитуда |
| Прямоугольное окно | 424 | -28.8 |
| Окно Барлетта | 434 | -28.3 |
| Окно Хэмминга | 434 | -28.3 |
| Окно Хэннинга | 436 | -28.1 |
| Окно Блэкмэна | 436 | -27.9 |
| Окно Блэкмэна-Харриса | 435 | -27.7 |
| Окно Уэлча | 432 | -28.6 |
| Окно Гаусса (а = 2.5) | 434 | -28.2 |
| Окно Гаусса (а = 3.5) | 435 | -27.7 |
| Окно Гаусса (а = 4.5) | 433 | -27.4 |

По данным из таблицы видно, что «чистых» 440 Гц не было получено ни при одном из оконных фильтров. Это можно объяснить как искажениями самих фильтров, так и несовершенством инструмента (фортепиано) и звукозаписывающего оборудования.

При помощи ПО Audacity был записал звуковой файл и экспортирован из проекта в двух форматах – mp3, wav. Для получения файла в формате amr была использована бесплатная программа Format Factory. Проведем визуальный анализ сигналов, содержащихся в данных файлах.

Mp3-файл был экспортирован со средним битрейтом (192 Кбит/с), и «на слух» разницы между ним и WAV-файлом практически нет, хоть wav-файл и занимает на диске почти в 7.5 раз больше – 187 КБ, а mp3-файл «весит» 26 КБ при их разнице в объеме в 10 раз (~50кбайт против ~500кбайт). При конвертации файла в формат amr появились искажения, но он и занимает лишь 1.6 КБ в памяти – более, чем в 100 раз меньше wav-файла. Сравним форму сигнала (амплитуду во времени) для трех получившихся файлов с помощью Audacity. На рисунке 17 отображены формы сигналов (сверху вниз – mp3, wav, amr).



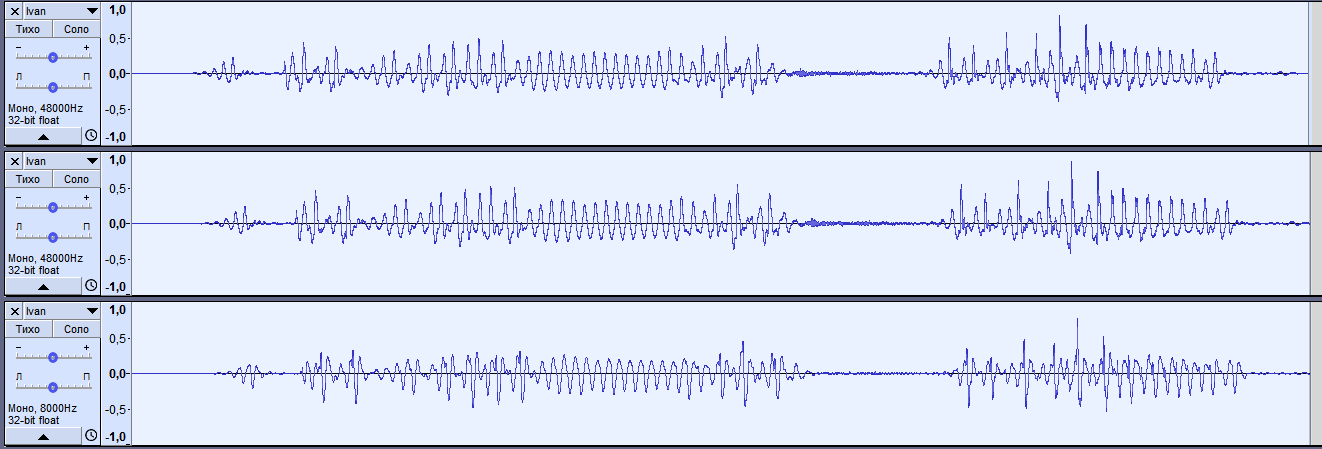


Рисунок 17. Открытие записанного голоса в Audacity

Выделим аудиофайлы и построим графики их спектра:

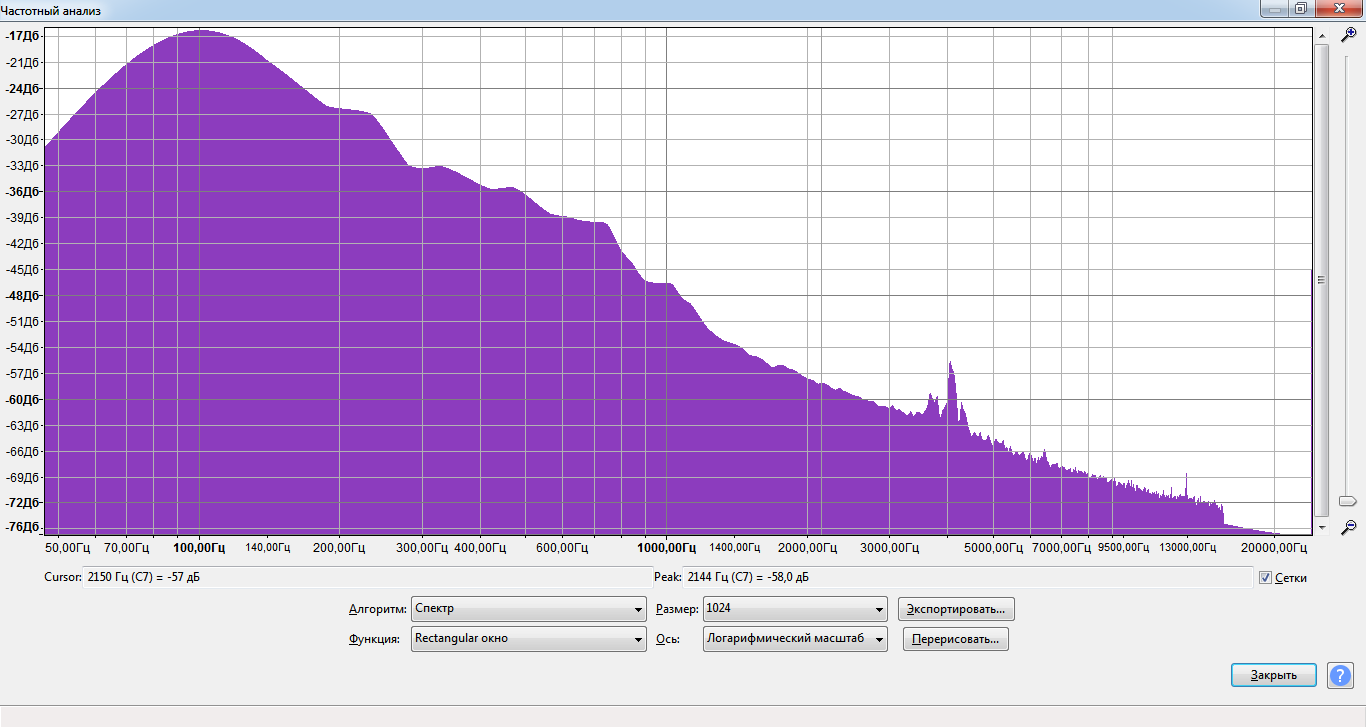


Рисунок 18. График спектра mp3-файла

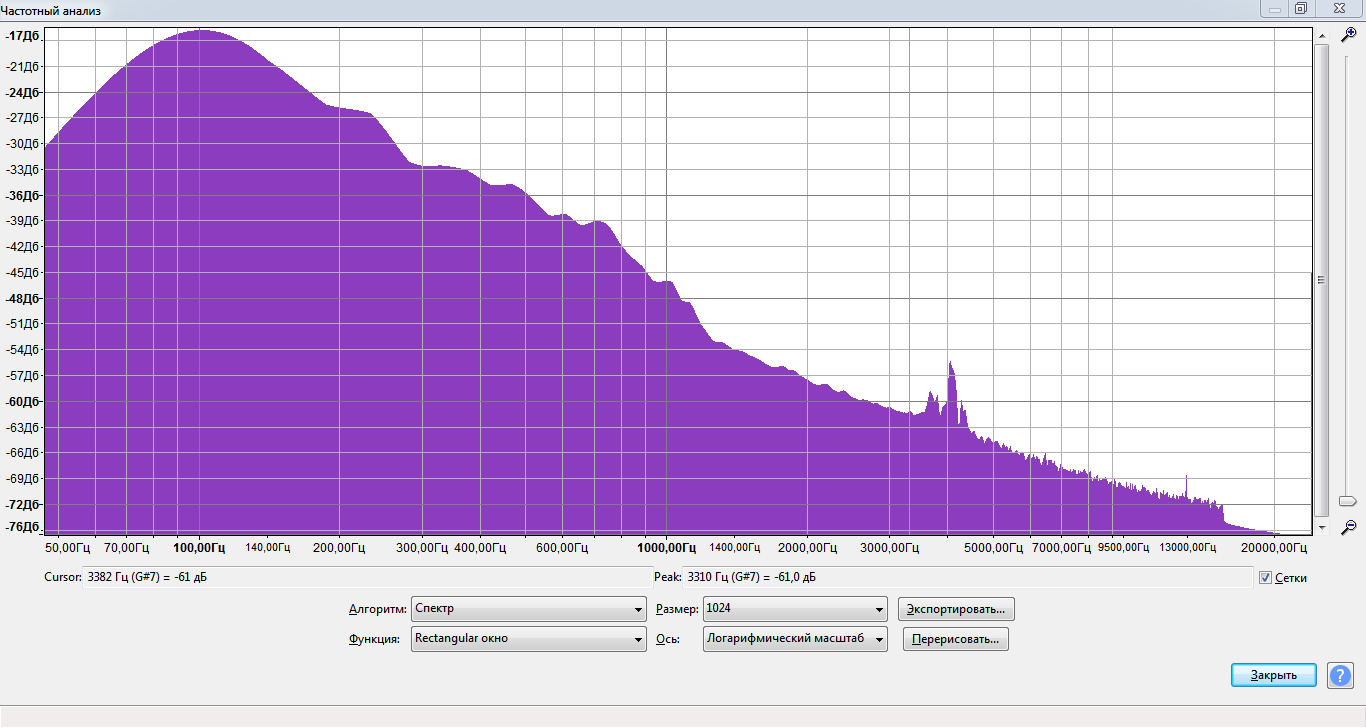


Рисунок 19. График спектра wav-файла

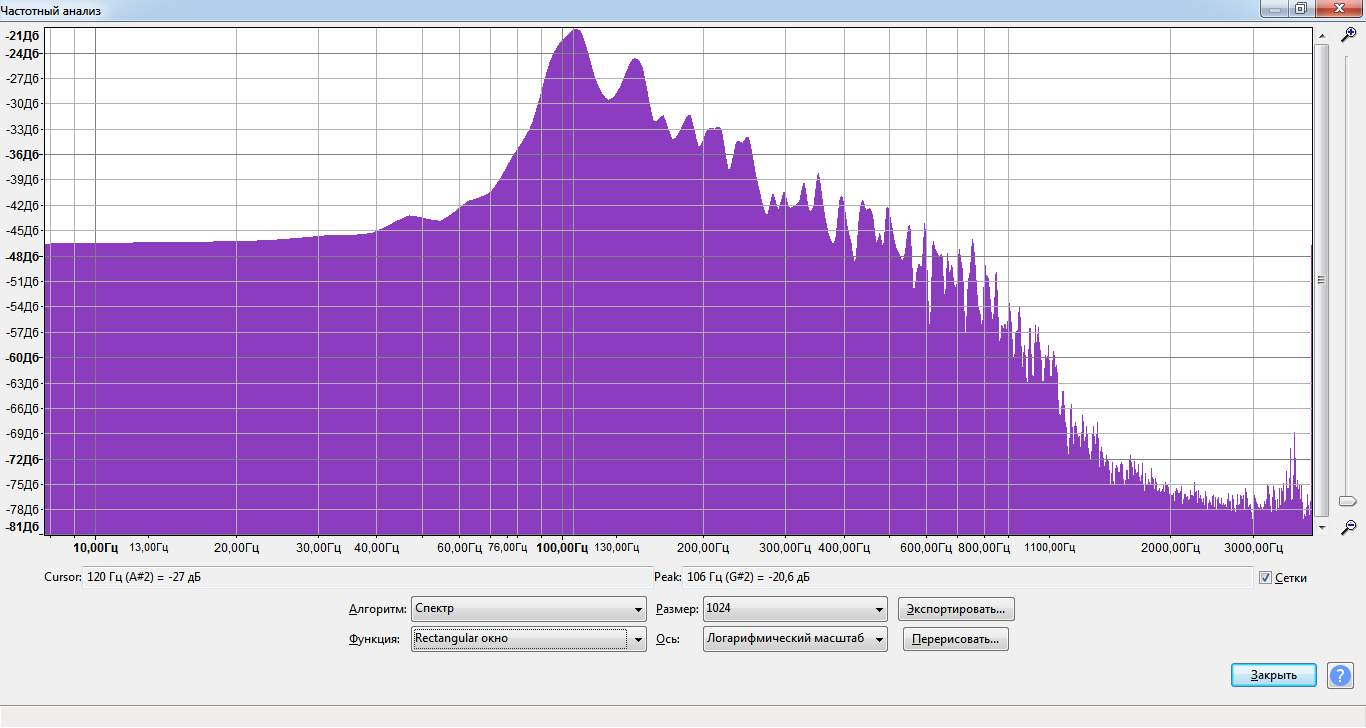


Рисунок 20. График спектра amr-файла

По графикам видно, что наиболее «заполненные» частоты находятся в диапазоне 100 – 200 Гц, что соответствует частотам мужского голоса. Также заметно, что при кодировании в формате amr частоты, лежащие за пределами типичных для звука человеческой речи, «отброшены». Собственно, это и является одной из главных особенностей этого формата, позволяющей экономить память для хранения файлов.

# Анализ посредством ПО Matlab

Для анализа аудиофайла воспользуемся пакетом MATLAB, применив следующий код:

Fs=8160;

samples=[1,Fs];

name = input('Введите имя wav-файла из папки wav на диске : ','s');

name1 = ['wav\' name '.wav'];

[myRecording,Fs] = audioread(name1,samples);

sound(myRecording,Fs);

Y = fft(myRecording,Fs);

YY = Y.\*conj(Y)/256;

f = (0:1000);

figure(1), plot(f, YY(1:1001)), grid;

title('Power Spectrum');

xlabel('Frequency(Hz)');

ylabel('Power(Watt)');

pause;

plot(myRecording(1000:4000)),grid;

title('Waveform');

xlabel('Time(s)');

ylabel('Signal(V)');

pause

Результатом выполнения такого кода будет являться построение следующих графиков:

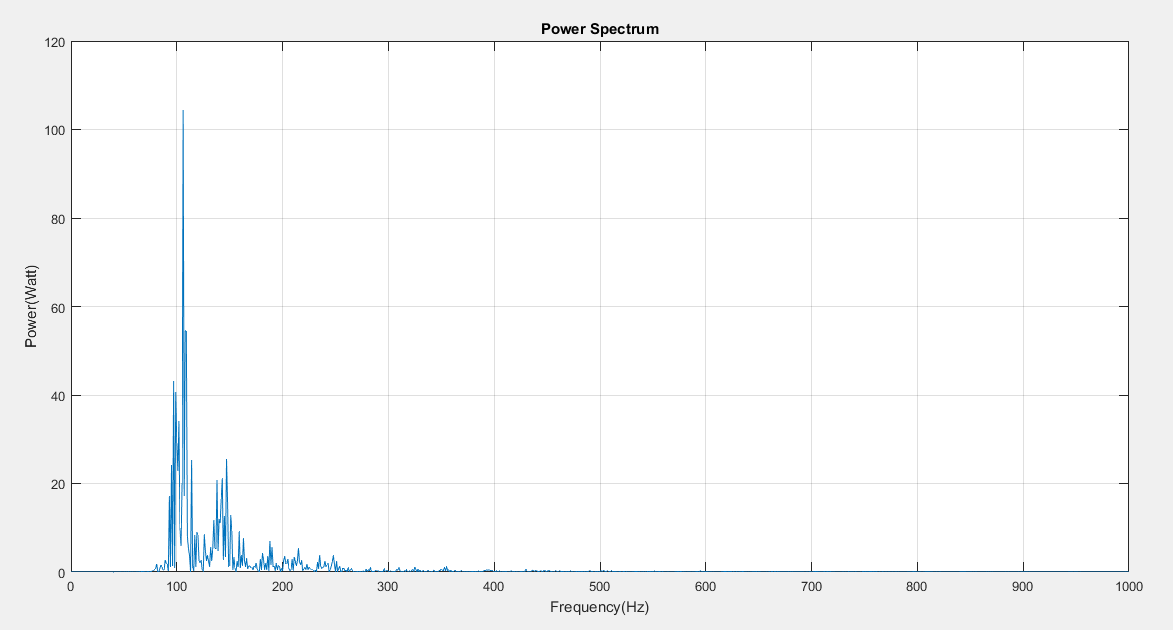


Рисунок 21. Частотный спектр сигнала

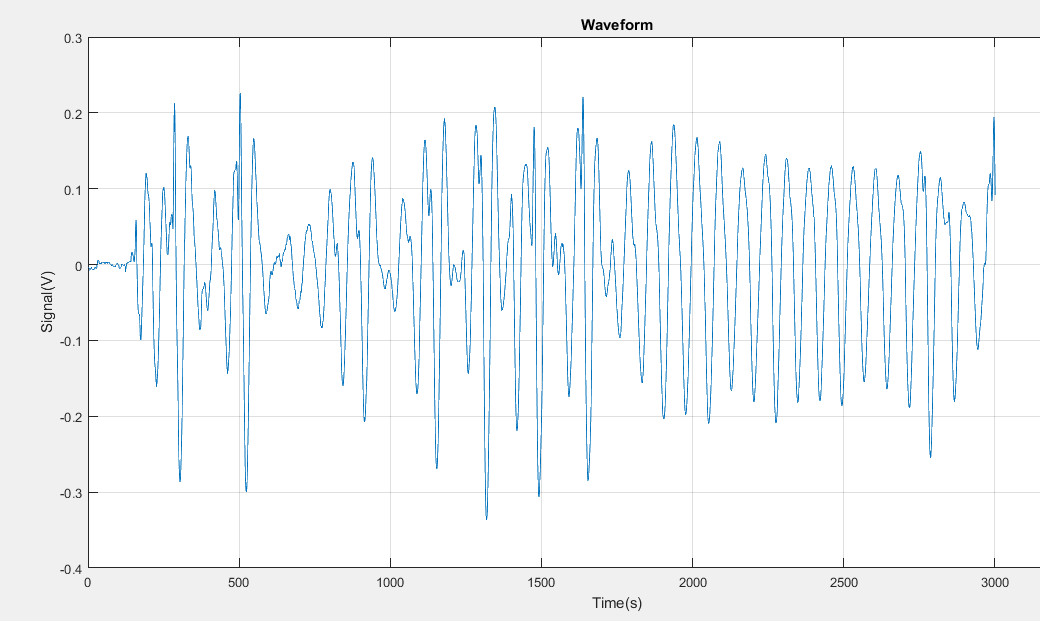


Рисунок 22. Вид рассматриваемого сигнала

# Вейвлет-преобразования

Вейвлет-преобразование (англ. Wavelet transform) — интегральное преобразование, которое представляет собой свертку вейвлет-функции с сигналом. Вейвлет-преобразование переводит сигнал из временного представления в частотно-временное.

Способ преобразования функции (или сигнала) в форму, которая или делает некоторые величины исходного сигнала более поддающимися изучению, или позволяет сжать исходный набор данных. Вейвлетное преобразование сигналов является обобщением спектрального анализа. Термин (англ. wavelet) в переводе с английского означает «маленькая волна». Вейвлеты — это обобщённое название математических функций определенной формы, которые локальны во времени и по частоте и в которых все функции получаются из одной базовой, изменяя её (сдвигая, растягивая).

# Главные признаки вейвлета

В качестве базисных функций, образующих ортогональный базис, можно использовать широкий набор вейвлетов. Для практического применения важно знать признаки, которыми непременно должна обладать исходная функция, чтобы стать вейвлетом. Приведем здесь основные из них.

*Ограниченность*. Квадрат нормы функции должен быть конечным:



*Локализация*. ВП в отличие от преобразования Фурье, вейвлет-преобразование использует локализованную исходную функцию и во времени, и по частоте. Для этого достаточно, чтобы выполнялись условия:



*Нулевое среднее*. График исходной функции должен осциллировать (быть знакопеременным) вокруг нуля на оси времени и иметь нулевую площадь:



Равенство нулю площади функции ψ (t), т.е. нулевого момента, приводит к тому, что фурье-преобразование Sψ (ω) этой функции имеет вид полосового фильтра.



Вейвлеты n-го порядка позволяют анализировать более тонкую (высокочастотную) структуру сигнала, подавляя медленно изменяющиеся его составляющие.

Характерным признаком ВП является его *самоподобие*. Все вейвлеты конкретного семейства имеют то же число осцилляций, что и материнский вейвлет, поскольку получены из него посредством масштабных преобразований и сдвига.

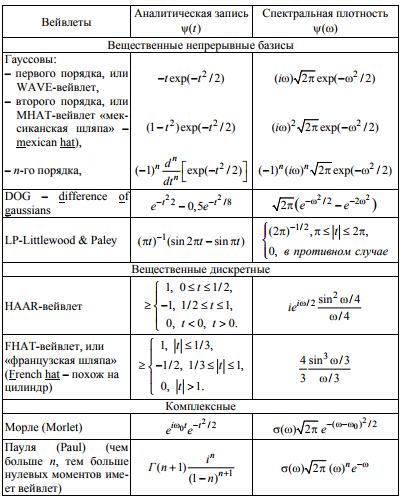


Рисунок 23. Некоторые виды вейвлетов

# ****Виды поддерживаемых вейвлетов в Matlab****

**Далее приведены описания и графики некоторых типов вейвлет-функций, доступных для применения с помощью Matlab Wavelet Toolbox (описания и графики из [5]).**

* **Haar – вейвлет Хаара,** образует полный ортонормированный базис. В пакете Matlab имеет название db1.

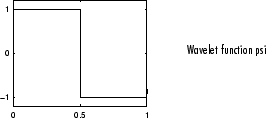


Рисунок 24. Вейвлет Хаара

* **Db – вейвлет Добеши, семейство ортогональных вейвлетов с компактным носителем, вычисляемым итерационным путём.**

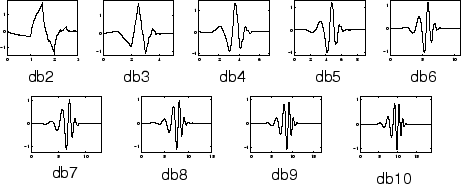


Рисунок 25. Вейвлет Добеши

* **Sym –** симмлеты, практически **симметричны вейвлетам Добеши, близки по свойствам. Хорошо подходят для восстановления сигналов;**

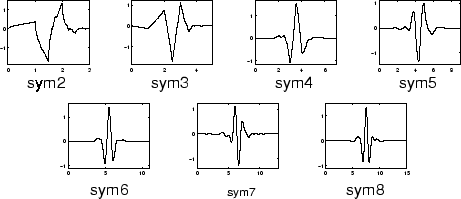


Рисунок 26. Симмлеты

* **Coif - дискретные вейвлеты, разработанные Ингрид Добеши и имеющие функции масштабирования с исчезающими моментами. Близки к симметричным.**

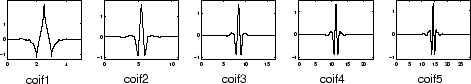


Рисунок 27. Вейвлеты Койфмана

* **Bior – вейвлет, в котором связанное вейвлет-преобразование обратимо, но не обязательно ортогонально. Использование биортогональных вейвлетов предлагает больше степеней свободы, чем ортогональные вейвлеты.**

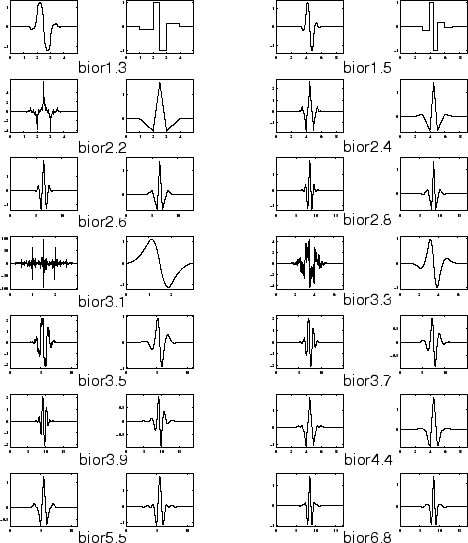


Рисунок 28. Биортоганальные вейвлеты

# Выполнение sym3- и bior2.8-вейвлет преобразования (по варианту).

Для выполнения вейвлет-преобразований записанного сигнала используем следующий код Matlab:

Fs=8160;

samples=[1,Fs];

name = input('Введите имя wav-файла из папки wav на диске : ','s');

name1 = ['wav\' name '.wav'];

%reads data from the file named filename, and returns sampled data, y, and a sample rate for that data, Fs.

[myRec,Fs] = audioread(name1,samples);

myRec=myRec(1:8160);

len = length(myRec);

cw1 = cwt(myRec,1:100,'sym3','plot');

title('Continuous Transform, absolute coefficients.')

ylabel('Scale')

[cw1,sc] = cwt(myRec,1:100,'sym3','scal');

title('Scalogram, sym3')

ylabel('Scale')

pause;

cw1 = cwt(myRec,1:100,'bior2.8','plot');

[cw1,sc] = cwt(myRec,1:100,'bior2.8','scal');

title('Scalogram, bior2.8')

ylabel('Scale')

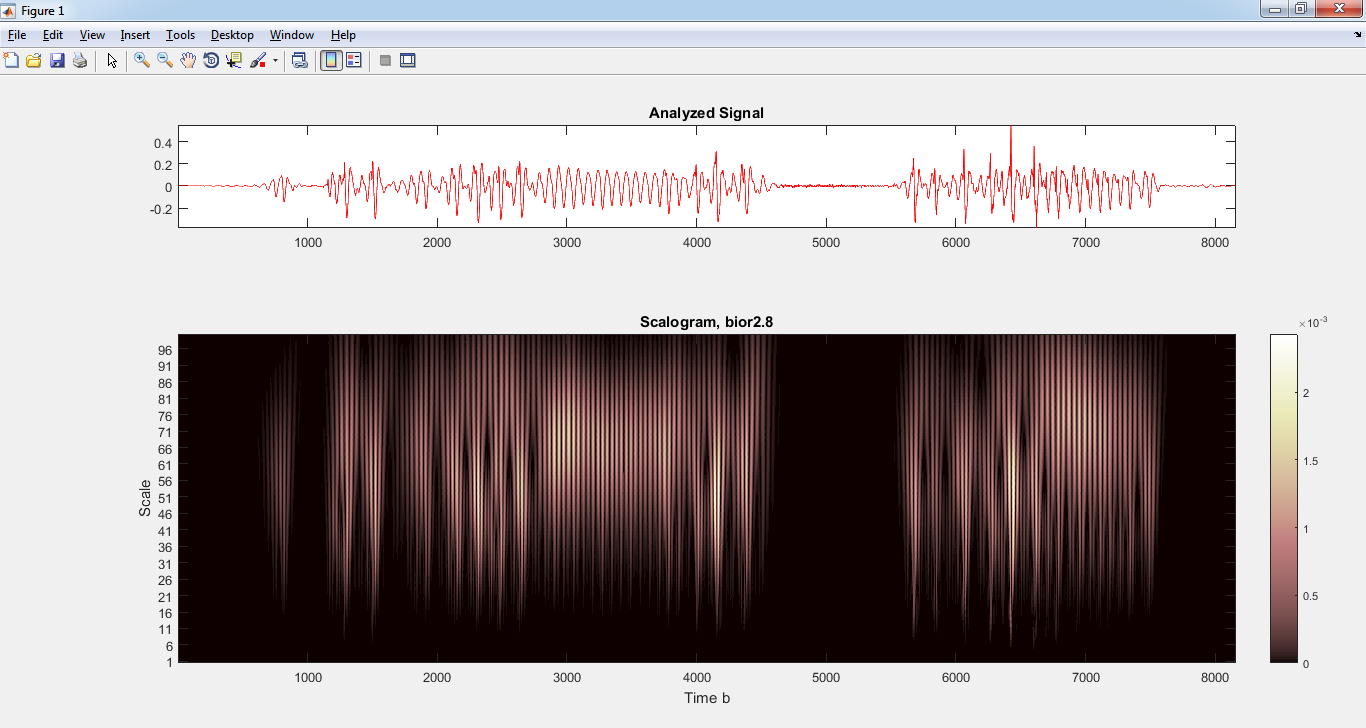


Рисунок 29. Bior2.8-преобразование сигнала

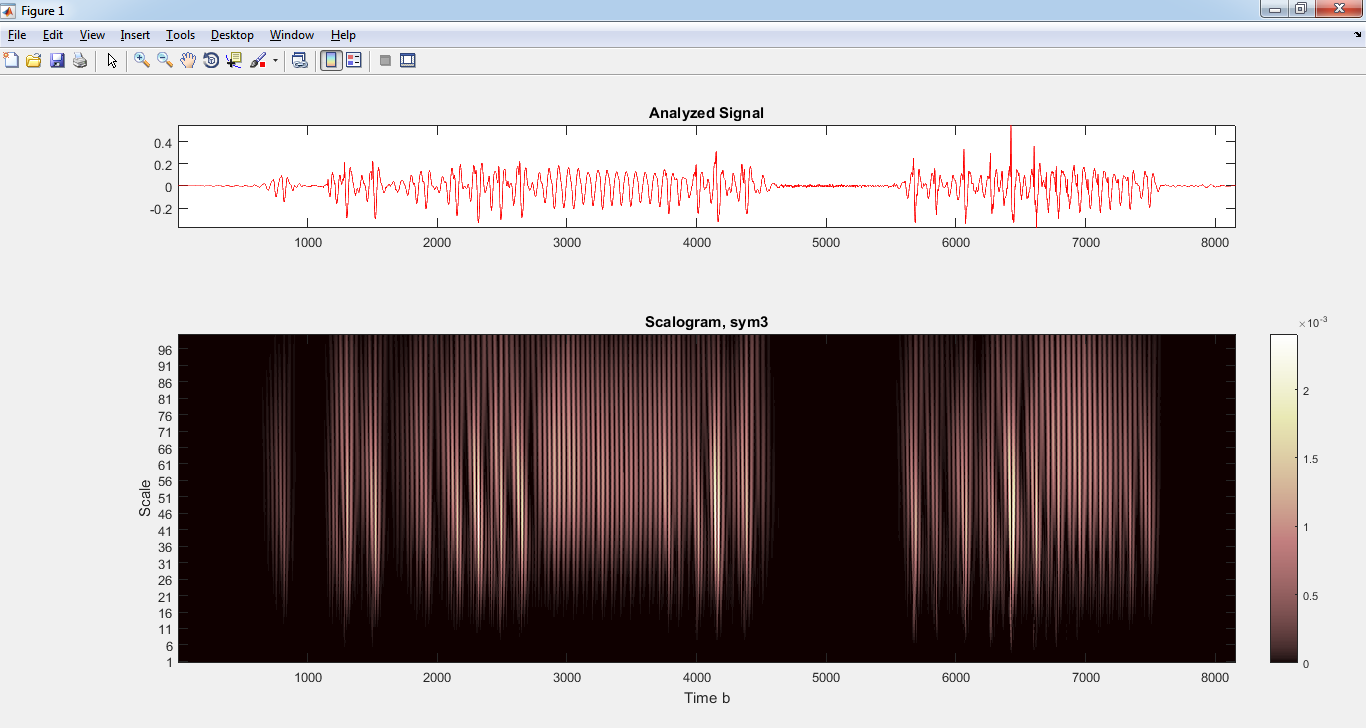


Рисунок 30. SYM3-преобразование сигнала

Как видно на рисунках, результат преобразования, несмотря на различные вейвлеты, практически идентичен. Это можно объяснить как особенностями анализируемого сигнала (его простотой), так и мощностью вейвлет-преобразований.

# Зашумление сигнала

Увеличение зашумленности достигается увеличением коэффициентом в переменной An. An выбрано в соответствии с вариантом по списку в журнале – 8.

Код Matlab:

%% Параметры

Tm=5;% Длина сигнала (с)

Fd=512;% Частота дискретизации (Гц)

Ak=0.5;% Постоянная составляющая (Попугаев)

A1=1;% Амплитуда первой синусоиды (Попугаев)

A2=0.7;% Амплитуда второй синусоиды (Попугаев)

F1=13;% Частота первой синусоиды (Гц)

F2=42;% Частота второй синусоиды (Гц)

Phi1=0;% Начальная фаза первой синусоиды (Градусов)

Phi2=37;% Начальная фаза второй синусоиды (Градусов)

An=8\*A1;% Дисперсия шума (Попугаев)

FftL=1024;% Количество линий Фурье спектра

%% Генерация рабочих массивов

T=0:1/Fd:Tm;% Массив отсчетов времени

Noise=An\*randn(1,length(T));% Массив случайного шума длиной равной массиву времени

Signal=Ak+A1\*sind((F1\*360).\*T+Phi1)+A2\*sind((F2\*360).\*T+Phi2);% Массив сигнала (смесь 2х синусоид и постоянной составляющей)

%% Спектральное представление сигнала

FftS=abs(fft(Signal,FftL));% Амплитуды преобразования Фурье сигнала

FftS=2\*FftS./FftL;% Нормировка спектра по амплитуде

FftS(1)=FftS(1)/2;% Нормировка постоянной составляющей в спектре

FftSh=abs(fft(Signal+Noise,FftL));% Амплитуды преобразования Фурье смеси сигнал+шум

FftSh=2\*FftSh./FftL;% Нормировка спектра по амплитуде

FftSh(1)=FftSh(1)/2;% Нормировка постоянной составляющей в спектре

%% Построение графиков

subplot(2,1,1);% Выбор области окна для построения

plot(T,Signal);% Построение сигнала

title('Сигнал');% Подпись графика

xlabel('Время (с)');% Подпись оси х графика

ylabel('Амплитуда (Попугаи)');% Подпись оси у графика

subplot(2,1,2);% Выбор области окна для построения

plot(T,Signal+Noise);% Построение смеси сигнал+шум

title('Сигнал+шум');% Подпись графика

xlabel('Время (с)');% Подпись оси х графика

ylabel('Амплитуда (Попугаи)');% Подпись оси у графика

F=0:Fd/FftL:Fd/2-1/FftL;% Массив частот вычисляемого спектра Фурье

figure% Создаем новое окно

subplot(2,1,1);% Выбор области окна для построения

plot(F,FftS(1:length(F)));% Построение спектра Фурье сигнала

title('Спектр сигнала');% Подпись графика

xlabel('Частота (Гц)');% Подпись оси х графика

ylabel('Амплитуда (Попугаи)');% Подпись оси у графика

subplot(2,1,2);% Выбор области окна для построения

plot(F,FftSh(1:length(F)));% Построение спектра Фурье сигнала

title('Спектр сигнала с шумом');% Подпись графика

xlabel('Частота (Гц)');% Подпись оси х графика

ylabel('Амплитуда (Попугаи)');% Подпись оси у графика

Выполняемым кодом создается сигнал следующего вида:

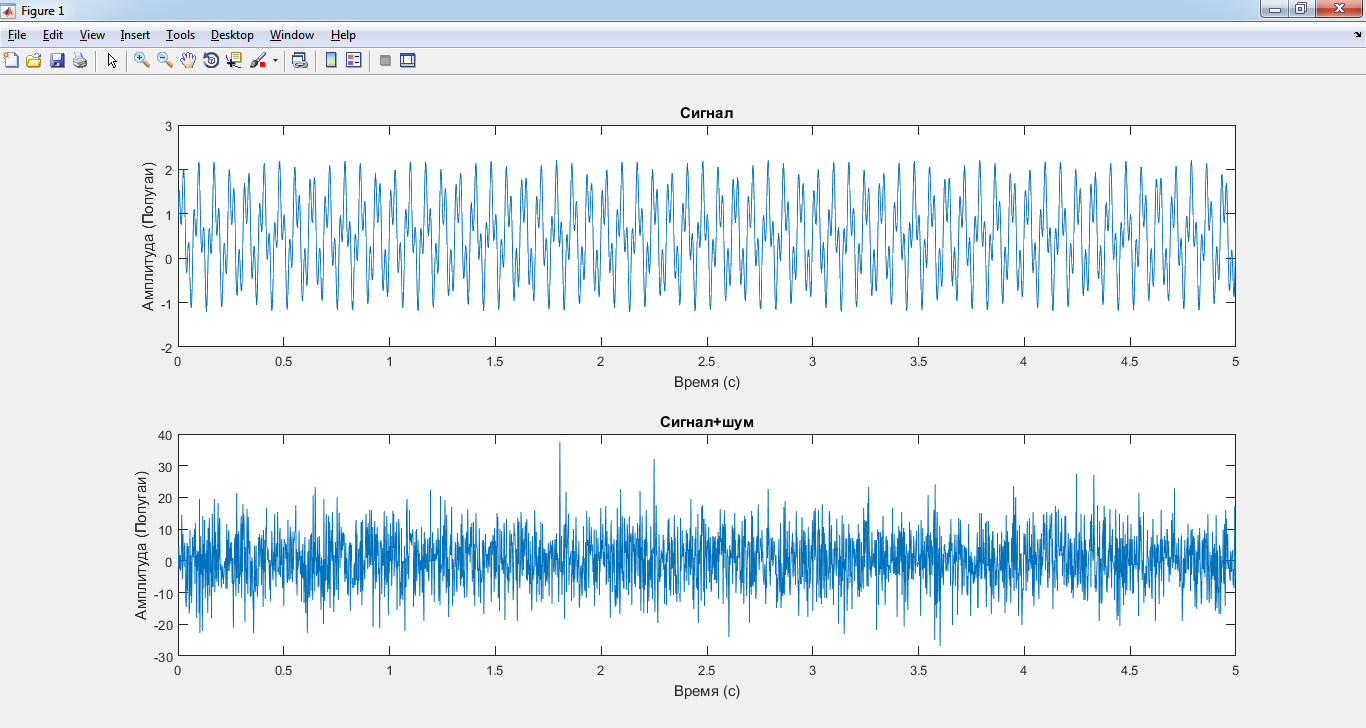


Рисунок 31. Сигнал и зашумленный сигнал

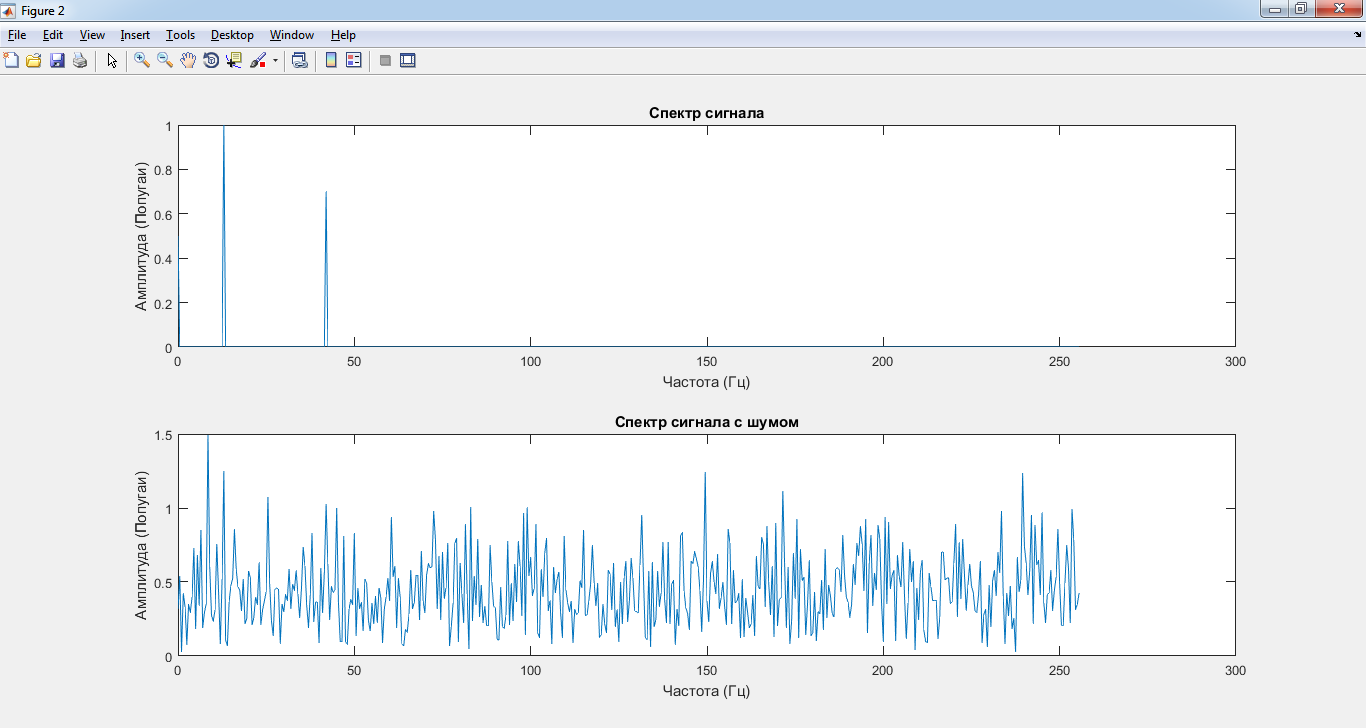


Рисунок 32. Спектры чистого и зашумленного сигналов

Как видно на рисунке 32, при добавлении шума спектр становится непрерывным, хотя спектр чистого сигнала представляет собой два треугольника, «пики» которых являются частотами синусоид.

# Удаление шума

Удаление шума будет производиться при помощи функции *wden*. Данная функция выполняет одномерное шумоподавление для данных, содержащихся в векторе, используя такие параметры:

* критерий, позволяющий выбрать правило расчета значения порога;
* принцип пороговой обработки, задающий мягкий или жесткий вид;
* глубина разложения данных;
* вейвлет-функция.

*XD = wden (X, TPTR, SORH, SCAL, N, 'wname')* возвращает версию сигнала X с удаленным шумом после применения вейвлет-коэффициентов.

Идентификатор символа TPTR содержит правило выбора порога:

* «rigrsure» использует принцип случайного риска Штейна;
* «sqtwolog» для универсального порога ;
* «minimaxi» для минимаксного порога.

*SORH* задает тип «порога» - мягкий или жесткий.

*SCAL* определяет мультипликативное пороговое масштабирование:

* *mln* для перемасштабирования, выполненного с использованием зависимой от уровня оценки уровня шума.

Разложение вейвлетов выполняется на уровне *lev*; «wname» - вектор символов, содержащий имя нужного ортогонального вейвлета.

Для текущего сигнала применялся следующий код:

lev = 7;

wname = 'sym8';

dnsig1 = wden(Signal+Noise,'rigrsure','h','mln',lev,wname);%адаптивный пороговый выбор с использованием принципа несмещенной оценки риска Штеина

dnsig2 = wden(Signal+Noise,'minimaxi','h','mln',lev,wname);%определение порога minimax

dnsig3 = wden(Signal+Noise,'sqtwolog','h','mln',lev,wname);%порог по sqrt(2\*log(length(X)))

subplot(3,1,1)

plot(T,dnsig1, 'r-')%восстановленный сигнал

hold on

plot(T,Signal, 'b')%исходный сигнал

title('Denoised Signal - SURE')

legend('восстановленный сигнал','исходный сигнал')

subplot(3,1,2)

plot(T,dnsig2, 'r-')

hold on

plot(T,Signal, 'b')%исходный сигнал

title('Denoised Signal - Minimax')

subplot(3,1,3)

plot(T,dnsig3, 'r-')

hold on

plot(T,Signal, 'b')%исходный сигнал

title('Denoised Signal - Donoho-Johnstone')

В результате получено три графика сигналов (для наглядности графики с зашумленным сигналом удалены), синяя линия – чистый сигнал, красная – восстановленный:

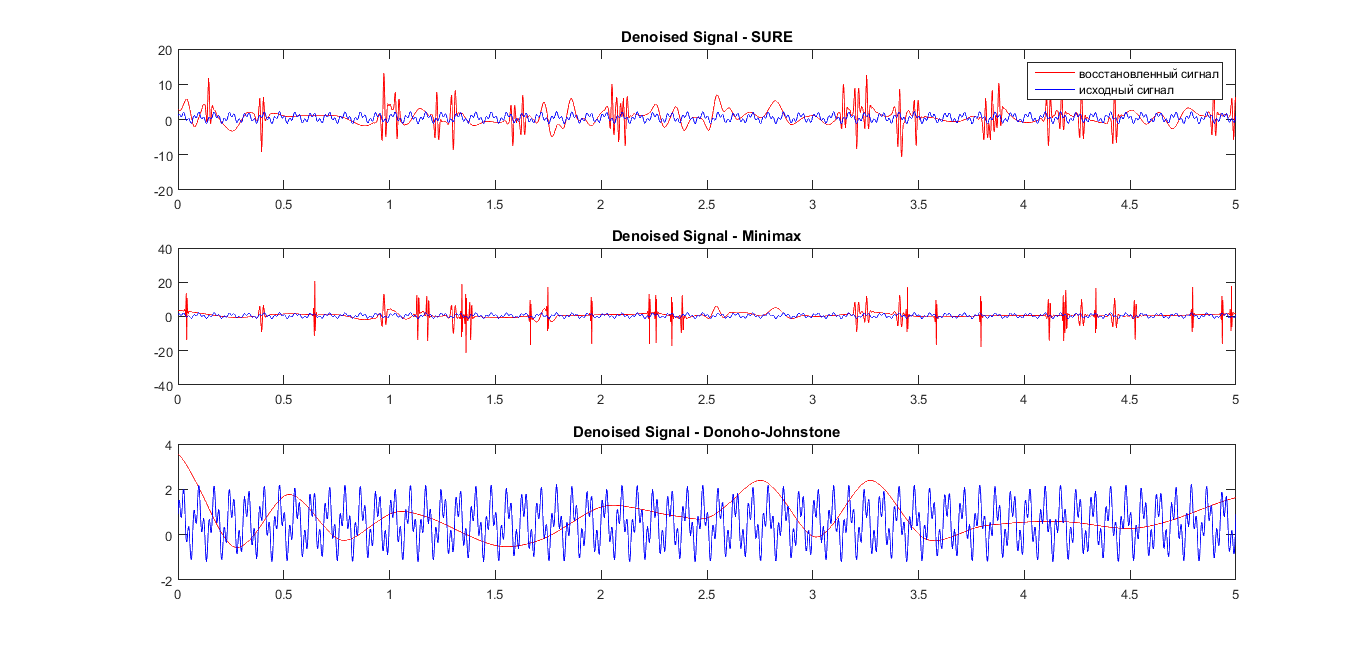


Рисунок 33. Исходные и восстановленные разными методами сигналы

# Фильтрация изображений

Matlab является мощным инструментом по обработке изображений. Пакет включает в себя большое количество встроенных функций, реализующих наиболее распространенные методы обработки изображений. Рассмотрим основные возможности пакета Image Processing Toolbox, входящего в Matlab.

Основные функции, предоставляемые пакетом:

* геометрическое преобразование изображений – кадрирование, изменения размеров и поворот;
* анализ изображений – построение гистограмм цветов и интенсивностей пикселей;
* улучшение изображений – перераспределение гистограммы яркостей, коррекция динамического диапазона изображений;
* фильтрация изображений – фильтры повышения резкости, усредняющий низкочастотный и пр.;
* сегментация изображений.

Откроем изображение в Matlab, переведем его в градации серого для упрощения машинной обработки и добавим шум типа salt&pepper (рисунки 34-36).

Код Matlab:

PIC = imread('test.jpg'); //считывание начального изображения

imshow(PIC); //отображение

G\_PIC = rgb2gray(PIC); //перевод изображения в градации серого

imshow(G\_PIC); //отображение серого изображения

NOISED\_PIC = imnoise(G\_PIC, 'salt & pepper', 0.2);//добавление шума на серое изображение

imshow(NOISED\_PIC);//отображение изображения с шумом

Можно было использовать любой из шумов функции imnoise:

|  |  |
| --- | --- |
| Значение | Пояснение |
| 'gaussian' | Гауссовский белый шум с постоянным средним значением и дисперсией |
| 'localvar' | Нулевой средний гауссовский белый шум с зависящей от интенсивности дисперсией |
| 'poisson' | Пуассоновский шум |
| 'salt & pepper' | Включает и исключает пиксели |
| 'speckle' | Мультипликативный шум |

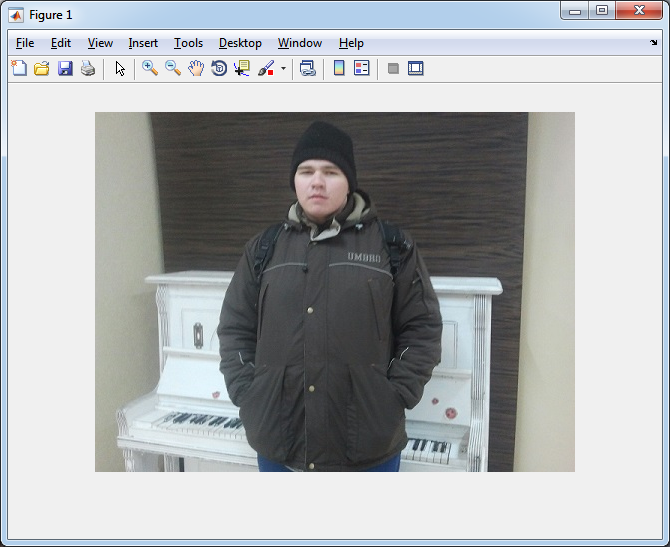


Рисунок 34. Изображение в Matlab

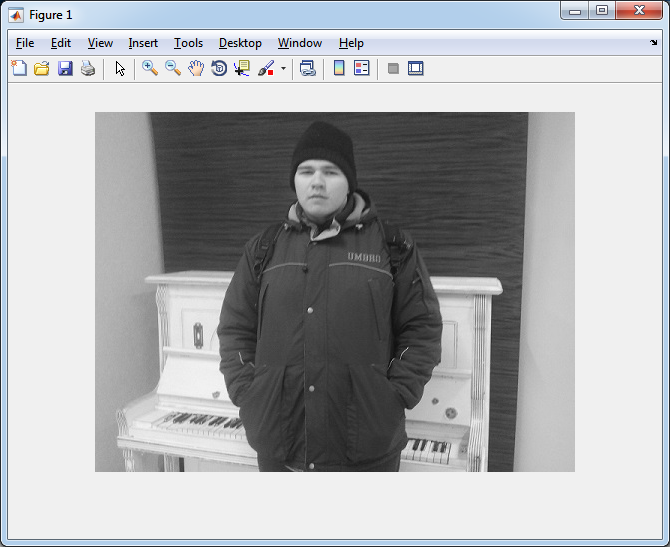


Рисунок 35. Изображение в градациях серого

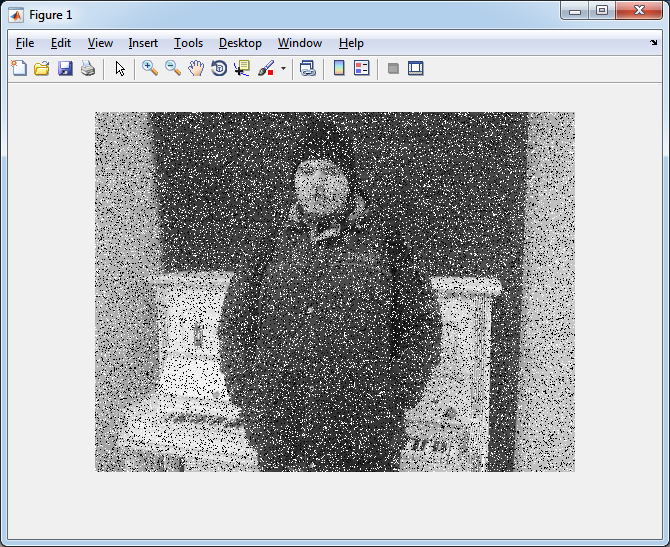


Рисунок 36. Наложение шума на изображение

Для удаления шумов в среде Matlab предусмотрено несколько фильтров – для начала воспользуемся медианным фильтром с размером окна 5х5 (рисунок 37).

Код Matlab:

MEDF\_PIC = medfilt2(NOISED\_PIC,[5 5]);

imshow(MEDF\_PIC);

Как видно на рисунке 37, шум был удален, однако изображение стало очень сильно размытым. На рисунке 38 изображен результат двух последовательных применений медианного фильтра с окном 3х3. Это изображение также очищено от шумов, но «размытость» изображения меньше, чем в первом случае.

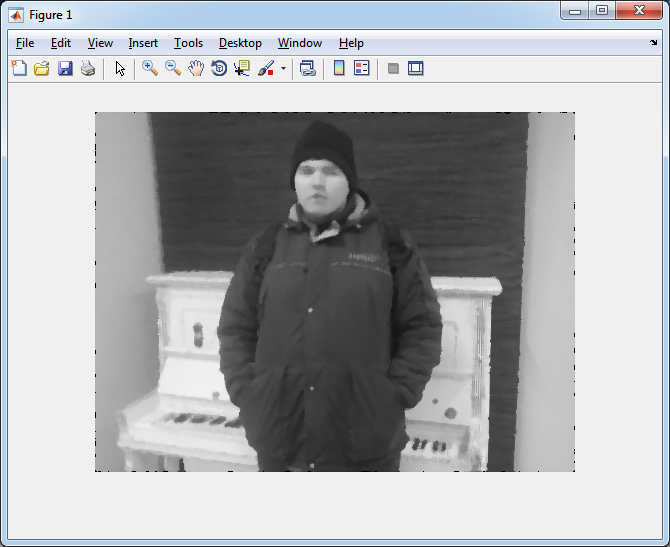


Рисунок 37. Результат применения медианного фильтра на сильно зашумленном изображении (размер окна – 5х5)

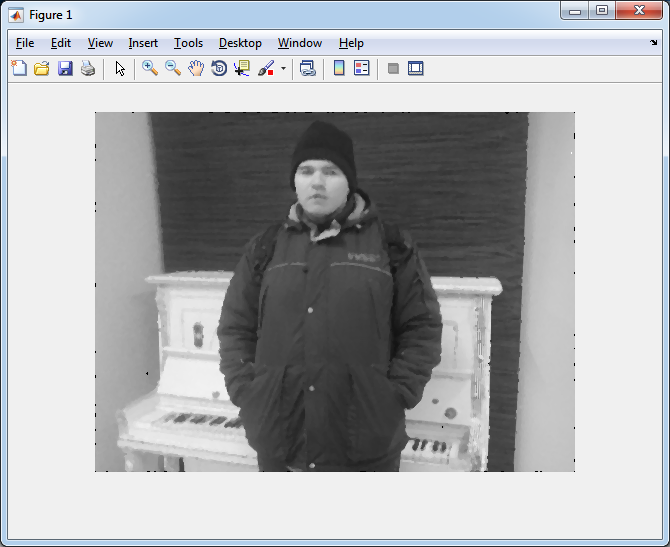


Рисунок 38. Результат двукратного последовательного применения медианного фильтра на сильно зашумленном изображении (размер окна – 3х3)

Поскольку фильтр сильно сгладил изображение, имеет смысл увеличить резкость (рисунок 39).

На рисунке 39 видно, что изображение стало более четким, но это же коснулось и оставшихся элементах шума – они стали более заметны.

Код Matlab:

MEDF\_PIC = medfilt2(NOISED\_PIC,[3 3]);

MEDF\_PIC = medfilt2(MEDF\_PIC,[3 3]);

sharp = fspecial('unsharp', 0.4);

SHARP\_PIC = imfilter(MEDF\_PIC, sharp);

imshow(SHARP\_PIC);

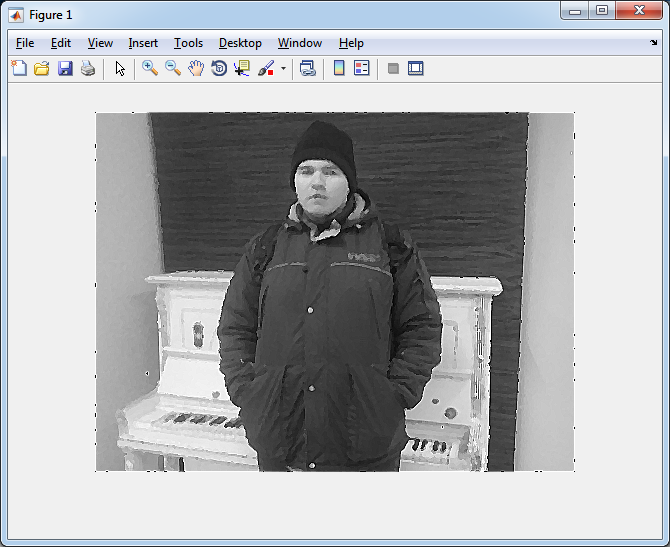


Рисунок 39. Результат увеличения резкости

Попробуем применить фильтр Гаусса для очистки изображения.

Код Matlab:

GAUSS\_PIC = imgaussfilt(NOISED\_PIC, 5);

imshow(GAUSS\_PIC);

Данная функция применяет к изображению двухмерное гауссово сглаживающее ядро со стандартным отклонением, равным 5 (рисунок 40). По рисунку 40 видно, что фильтр не справляется с удалением шума – результатом стало сильно размытое изображение, на котором видно шум.

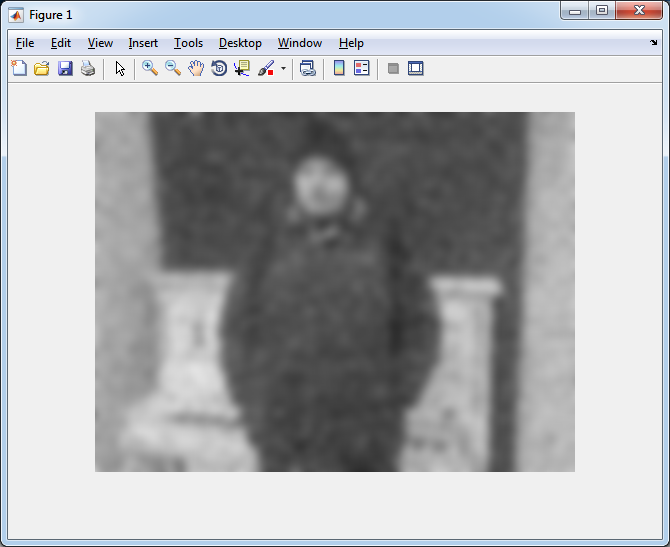


Рисунок 40. Результат использования фильтра Гаусса при отклонении, равном 5

Можно опробовать адаптивную фильтрацию – использовать фильтр Винера. Фильтр сильнее применяет сглаживание в зашумленных областях, как следствие, на сильно зашумленных изображениях непригоден к использованию. Функция wiener2 использует пиксельный адаптивный метод Винера, основанный на статистических данных, оцененных из локальной окрестности каждого пикселя. Результат, представленный на рисунке 41, не оправдал ожиданий – изображение сильно размыто, но шум не удален.

Код Matlab:

WIENER\_PIC = wiener2(NOISED\_PIC,[7 7]);

imshow(WIENER\_PIC);

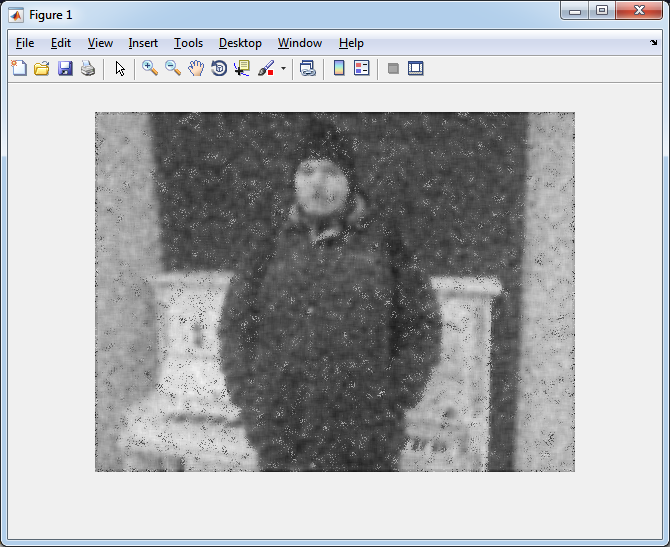


Рисунок 41. Фильтр Винера с окном 7х7

По результатам применения различных фильтров можно сделать вывод, что с сильно зашумленными изображениями они справиться в «автоматическом режиме» не могут – нужно долго и тщательно корректировать изображение. Однако несколько последовательных применений медианного фильтра дали заметно лучший результат, чем фильтры Винера и Гаусса, а после увеличения резкости получившегося изображения результат стал еще лучше, хоть и очень далек от первоначального изображения.

# Список самостоятельных работ

|  |  |
| --- | --- |
| **Часов** | **Деятельность** |
| 10 | Установка, настройка, освоение интерфейса MATLAB |
| 2 | Освоение работы с диаграммами в MATLAB |
| 5 | Установка, разбор функций Audacity |
| 2 | Разбор функций частотного анализа в Audacity |
| 2 | Изложение функций частотного анализа |
| 2 | Работа над построением графика спектра в MATLAB |
| 5 | Запись аудиодорожки, ее конвертация и рассмотрение волны |
| 5 | Ознакомление с теоретическим материалом по вейвлет преобразованиям |
| 2 | Изложение материала по вейвлет-преобразованиям |
| 5 | Поиск и изложение вейвлет-преобразований в MATLAB |
| 2 | Разбор работы с аудио в MATLAB |
| 3 | Рассмотрение методов добавления шума в аудиозапись |
| 1 | Добавление шума в аудиозапись |
| 5 | Рассмотрение методов удаления шума вейвлет преобразованием в пакете MATLAB |
| 5 | Практическое сравнение влияния выбранного вейвлета на вид фильтруемой волны |
| 10 | Ознакомление с теорией добавления и удаления шумов из сигналов |
| 2 | Изложение материала по добавлению шумов на изображение |
| 1 | Добавление шумов |
| 1 | Изложение материала по удалению шумов изображения |
| 2 | Удаление шумов |
| 10 | Обобщение результатов, оформление отчета |

# Вывод

В процессе выполнения лабораторных работ были освоены навыки работы с программами MATLAB и Audacity, рассмотрены возможности частотного анализа сигналов, алгоритмы получения частотных спектров. Также было произведено ознакомление с вейвлет-функциями, вейвлет-преобразованиями, целями и способами их применения. Были освоены принципы добавления шумов в цифровой сигнал и методы их удаления либо уменьшения зашумленности сигналов.

# Список литературы

1. Оконное преобразование Фурье [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/wiki/Оконное\_преобразование\_Фурье, дата обращения – 18.12.2017
2. Некоторые оконные функции и их параметры [Электронный ресурс] – режим доступа: http://www.dsplib.ru/content/winadd/win.html, дата обращения: 18.12.2017.
3. Цифровая обработка сигналов. Версия 1.0 [Электронный ресурс]: курс лекций / А. С. Глинченко. – Электрон. дан. (3 Мб). – Красноярск: ИПК СФУ, 2008.
4. Яковлев А.Н. Введение в вейвлет-преобразования: Учеб. пособие. – Новосибирск: Изд-во НГТУ, 2003. – 104 с.
5. Wavelet Toolbox [Электронный ресурс] – режим доступа: http://matlab.izmiran.ru/help/toolbox/wavelet/, дата обращения: 20.12.2017.
6. Вейвлет-преобразование [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Вейвлет-преобразование, (дата обращения 20.12.17).
7. Гонсалес Р., Вудс.Р., Эддинс С. Цифровая обработка изображений в среде MATLAB. Москва: Техносфера, 2006. – 616 с.
8. Imnoise // Mathworks URL: https://www.mathworks.com/help/images/ref/imnoise.html (дата обращения: 15.12.17).
9. Noise Removal // Mathworks URL: https://www.mathworks.com/help/images/noise-removal.html (дата обращения: 15.12.17).
10. Вейвлет-анализ // кафедра математики физический факультет МГУ URL: http://matematika.phys.msu.ru/files/stud\_spec/270/MM\_lec8.pdf (дата обращения: 10.12.17).
11. Window function // Wikipedia URL: https://en.wikipedia.org/wiki/Window\_function (дата обращения: 10.12.17).