

Лабораторная работа 1. Моделирование и исследование синусоидальных сигналов с помощью Scilab (ЦОС, DSP)

Краткие сведения о Scilab

Пакет прикладных программ Scilab предназначен для инженерных (технических) и научных расчётов является альтернативой MATLAB, которая свободно распространяется и бесплатна. Интерфейс Scilab похож на интерфейс MATLAB и он очень прост и удобен для пользователя.

Scilab содержит значительное число математических функций, и предусмотрена возможность добавления новых, написанных на различных языках (C, C++, Fortran и т. д.). В системе доступны разнообразные структуры данных (списки, полиномы, рациональные функции, линейные системы), интерпретатор и язык высокого уровня.

Scilab - открытая система, и пользователи могут добавлять в него свои типы данных и операции путём перегрузки.

В системе доступно множество инструментов:

- 2D- и 3D-графики, анимация
- Линейная алгебра, разреженные матрицы (sparse matrices)
- Полиномиальные и рациональные функции
- Интерполяция, аппроксимация
- Моделирование: решение ОДУ и ДУ
- Scicos: гибрид системы моделирования динамических систем и симуляции
- Дифференциальные и не дифференциальные оптимизации
- Обработка сигналов
- Параллельная работа
- Статистика
- Работа с компьютерной алгеброй
- Интерфейс к Fortran, Tcl/Tk, C, C++, Java, LabVIEW

Scilab имеет схожий с MATLAB язык программирования. В состав пакета входит утилита, позволяющая конвертировать документы Matlab в Scilab.

Scilab позволяет работать с элементарными и большим числом специальных функций (Бесселя, Неймана, интегральные функции), имеет мощные средства работы с матрицами, полиномами (в том числе и символично), производить численные вычисления (например, численное интегрирование) и решение задач линейной алгебры, оптимизации и симуляции, мощные статистические функции, а также средство для построения и работы с графиками.

Для численных расчётов используются библиотеки Lapack, LINPACK, ODEPACK, Atlas и другие.

В состав пакета также входит Scicos — инструмент для редактирования блочных диаграмм и моделирования (аналог simulink в пакете MATLAB). Имеется возможность совместной работы Scilab с программой LabVIEW.

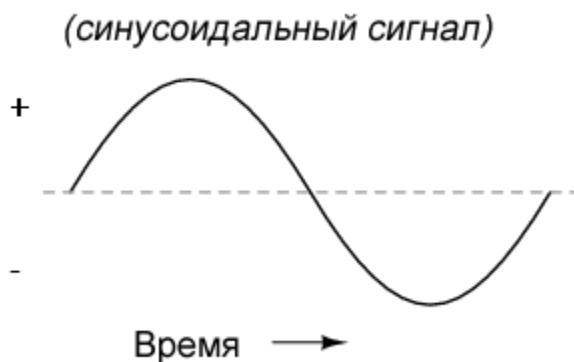
Цифровая обработка сигналов

Цифровая обработка сигналов (ЦОС, DSP, digital signal processing) – это обработка принятых цифровых сигналов, таких как звук, речь, видео, давление, температура, спектральная плотность и их математическая обработка. Цифровая обработка сигналов также, как и аналоговая, является составной частью теории обработки или исследования сигналов

Синусоидальный сигнал

Синусоидальный сигнал (синусоида), представляет собой математическую кривую, моделирующую непрерывное, плавное периодическое колебание. Волны гармонического сигнала - синусоиды имеют повторяющуюся картину. Длина повторения отождествляется с длиной волны. Синусоиды часто встречаются и исследуются в математике, физике, технике и радиоэлектронике при обработке сигналов.

Пример синусоидального сигнала



Анализ оцифрованных синусоидальных сигналов

Как аналоговые, так и цифровые синусоидальные сигналы имеют амплитуду, частоту и фазу. При обработке оцифрованных синусоидальных сигналов и при переходе от аналога к цифре фаза, и амплитуда сигнала обычно не меняются, а частота требует дополнительного анализа. В аналоговой области частота определяет количество периодов относительно времени. Единицы времени одинаковы всегда и везде – например, 1 Гц (это одно колебание или период в секунду) означает в каждом инженерном проекте одно и то же. В цифровой области частота теряет свое отношение к неизменной единице времени. В оцифрованных сигналах представляются отдельные значения амплитуд в соответствии с частотой дискретизации. Это может привести к путанице по двум причинам: во-первых, используется много разных частот дискретизации, и во-вторых, информация о частоте дискретизации не содержится в последовательности значений амплитуд.

Генерация синусоиды в Scilab

Рассмотрим эту задачу на примере. Пусть необходимо сгенерировать в Scilab один период синусоидального сигнала, который имеет 100 отсчетов за период. Для этого используем команду:

```
n = 0:99;
```

При этом будет создан массив, начинающийся с 0 и заканчивающийся на 99. В этом можно убедиться в “Variable Browser” (Обозревателе переменных), что что n действительно является одномерным массивом длиной 100.



	Name	Value	Type	Visibility
	n	1x100	Double	local

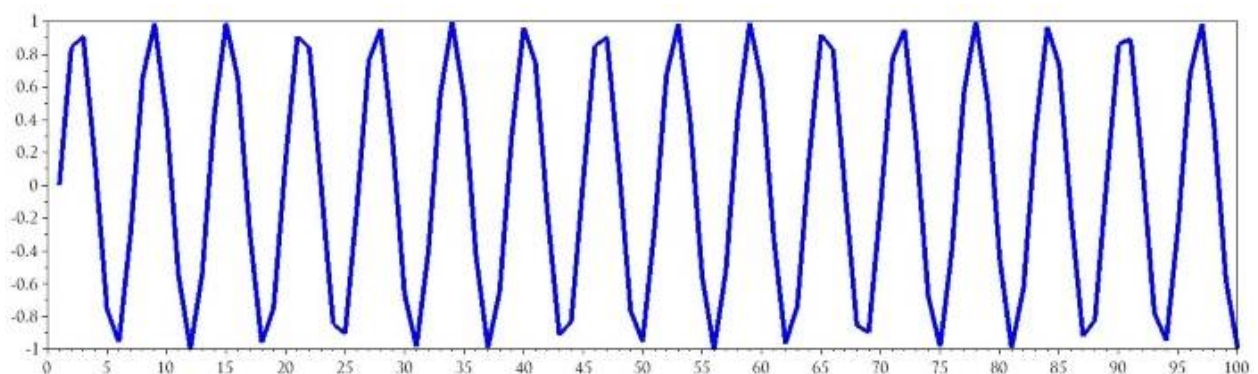
Созданный одномерный массив в обозревателе переменных

В данном случае массив n является цифровым эквивалентом t (т.е. времени) в типовом представлении синусоиды $\sin(\omega t)$, которое используется в аналоговой области. Затем вводим следующие две команды:

```
y = sin(n);
```

```
plot(y)
```

Получаем результат:



Созданная синусоида

Очевидно, предполагался другой результат моделирования, нужно чтобы один период состоял из 100 отсчетов. Необходимо разобраться - почему получился ошибочный результат. Для этого были следующие причины:

- синус – это просто функция, которая выдает значения в зависимости от аргумента;
- функция синуса не генерирует уникальные значения для каждого аргумента, то есть значения повторяются, когда аргумент изменяется на 2π : $\sin(0) = \sin(2\pi) = \sin(4\pi) = \sin(6\pi) \dots$
- таким образом, один период значений синуса соответствует диапазону аргумента значений в 2π ;
- это означает, что для генерации одного периода значений синусоиды необходимо изменить массив аргументов так, чтобы он изменялся от 0 до 2π .

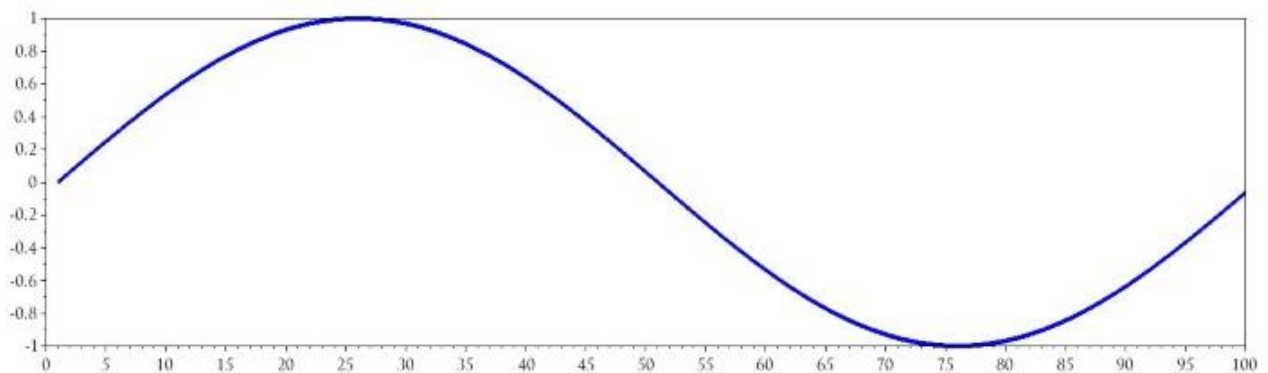
Команда $y = \sin(n)$ не создает сигнал нужной формы, потому что аргумент, переданный функции синуса, представляет собой массив, который изменяется от 0 до 99.

Решение состоит в том, чтобы разделить n на требуемое количество отсчетов за период, что в данном случае равно 100, а также умножить на 2π :

$$y = \sin(2\pi n/100);$$

Можете просто проверить, что это соответствует действительности: если n равно нулю, весь аргумент равен нулю; если n равно 100, аргумент будет равен $(2\pi \times 100/100) = 2\pi$; и все числа между ними масштабируются соответствующим образом. Таким образом, уменьшаем диапазон аргумента до 2π , и также создаем 100 выборок. При этом n не доходит до 100, и значение 100 должно быть первой выборкой во втором периоде. Другими словами, первый период покрывается 100 значениями от 0 до 99, второй период будет покрываться 100 значениями от 100 до 199, и так далее. Получаем следующий результат:

`plot(y)`



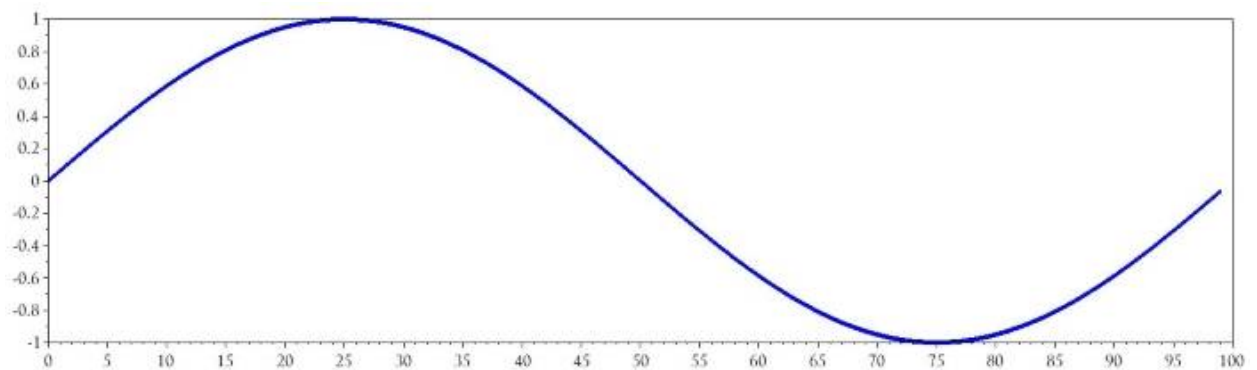
Созданная синусоида (исправленный график)

Коррекция графика

Однако, получившийся график следует скорректировать. Поскольку $\sin(0) = 0$, но на графике сигнал имеет значение 0 для значения 1 на горизонтальной оси. Другими словами, сигнал сдвигается на одну выборку вправо. Это происходит потому, что не указан список значений горизонтальной оси, которые соответствуют значениям вертикальной оси, содержащимся в массиве y . Если задать в Scilab `plot(y)`, то используются значения по умолчанию для горизонтальной оси, и, видимо, эти значения по умолчанию начинаются с 1.

Команда `y = sin(2*pi*n/100)` генерирует значения y , которые соответствуют номерам в массиве n . Если необходимо, чтобы график поддерживал эту связь между y и n , можно использовать следующую команду:

`plot(n, y)`



Созданная синусоида (правильное соответствие значений y и n)

Добавление в график частоты сигнала и частоты дискретизации

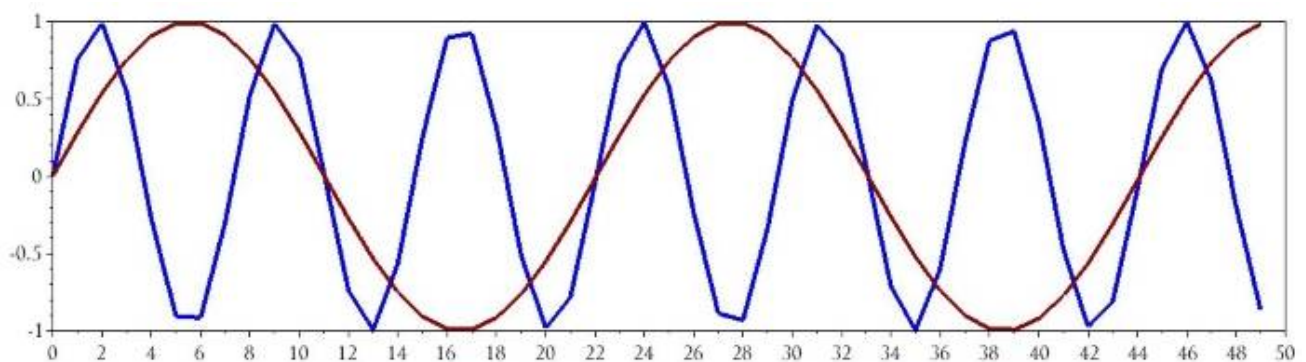
Когда обрабатываются реальные синусоидальные сигналы, используются частота сигнала (в аналоговой области) и частота дискретизации (в цифровой области). Следовательно, необходимо добавить эти физические единицы измерения в формальные математические отсчеты, представляемые ранее на графиках.

Для этого вспомним, что аргумент для функции синуса должен включать в себя коэффициент (2π / количество выборок за период). Можно рассчитать количество отсчетов за период для реальных физических сигналов следующим образом:

$$\text{количество выборок за период} = (\text{частота дискретизации})/(\text{частота сигнала})$$

Предположим, у нас есть система, которая оцифровывает аудиосигнал 6 кГц и отдельный аудиосигнал 2 кГц. Частота дискретизации составляет 44,1 кГц, а АЦП заполняет буфер на 50 выборок. Следующую последовательность команд Scilab можно использовать для генерации значений, соответствующих данным, созданным реальной системой.

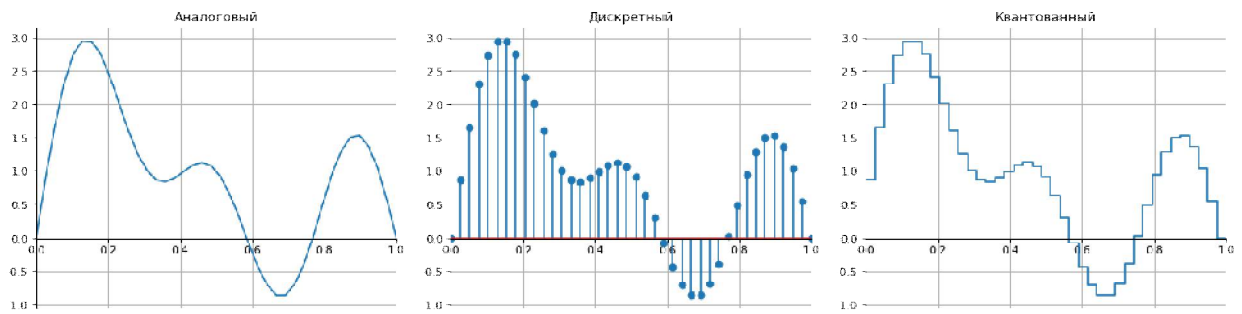
```
SignalFrequency_1 = 6e3;  
SignalFrequency_2 = 2e3;  
SamplingFrequency = 44.1e3;  
n = 0:49;  
Signal_1 = sin(2*%pi*n / (SamplingFrequency/SignalFrequency_1));  
Signal_2 = sin(2*%pi*n / (SamplingFrequency/SignalFrequency_2));  
plot(n, Signal_1)  
plot(n, Signal_2)
```



Синусоиды 6 кГц и 2 кГц, сгенерированные в Scilab

Таким образом, синусоидальный сигнал графически можно представить в различной форме: аналоговой, дискретной и квантованной.

Различные формы представления сигнала можно проиллюстрировать на следующем примере.



Следовательно, оцифрованный синусоидальный сигнал представляется в дискретной и квантованной форме, причем в дискретной форме сигнал часто представляется без вертикальных линий, отражающих значения амплитуд в точках дискретизации, как в примерах моделирования с помощью пакета Scilab в описании этой лабораторной работы.

Задание на лабораторную работу 1

1. Изучить интерфейс пакета Scilab, основные команды управления его работой и моделирования, обработки сигналов и построения графиков, сделать конспект
2. Изучить описание лабораторной работы Моделирование и исследование синусоидальных сигналов на Scilab (ЦОС, DSP).
3. Получить у преподавателя значение частоты для моделирование гармонического синусоидального сигнала, выполнить его генерацию с помощью пакета Scilab, дискретизацию по времени и квантование по уровню.
4. Построить графики аналогового, дискретного и квантованного сигнала на заданной частоте в формальных отсчетах и абсолютных значениях, выраженных в кГц в соответствии с рекомендациями описания лабораторной работы.
5. Определить максимальную ошибку при дискретизации сигнала на разных частотах, выразить её в процентах к амплитудному значению синусоидального сигнала.
6. Сделать выводы по результатам экспериментов о изменении ошибки с увеличением частоты дискретизации.

Лабораторная работа 2. Анализ сигналов в частотной области с использованием быстрого преобразования Фурье (БПФ, FFT) в Scilab

Быстрое преобразование Фурье

Преобразование Фурье позволяет получить частотного содержания сигнала по исходному временному представлению. Классическое преобразование Фурье представляет собой математическую процедуру, на вход которой поступает выражение, являющееся функцией времени, а на выходе она формирует выражение, представляющее функцию от частоты. При этом представление одночастотного синусоидального сигнала, как и любого гармонического сигнала, в частотной области не представляет особого интереса с точки зрения частотного анализа и может быть полезно только для ознакомления с самой процедурой преобразования. Кроме того, известно, что если необходимо получить частотное представление для дискретизированных сигналов, то следует использовать дискретное преобразование Фурье, сокращенно ДПФ или DFT. Следует отметить, что в зависимости от решаемых задач используют нужный вид классического преобразования Фурье дискретное или быстрое. Однако, следует помнить, что быстрое преобразование Фурье (БПФ) реализуется в виде алгоритмов, которые значительно сокращают время вычислений преобразования Фурье и могут эффективно выполнять вычисления дискретного преобразования - DFT.

При генерации данных сигнала для дискретного преобразования Фурье (DFT) в Scilab можно включить соответствующий массив в качестве аргумента команды **fft()**. Однако такая команда, как **fft(BasebandSignal)**, не будет выводить данные, которые могут отображаться как типовая диаграмма в частотной области.

Одна из причин этого в том, что результаты вычисления DFT являются комплексными числами, которые передают информацию об амплитуде и фазе. Если по условию задачи необходимо получить только амплитуду частотных составляющих сигнала, то можно извлечь данные об амплитуде с помощью команды **abs()**.

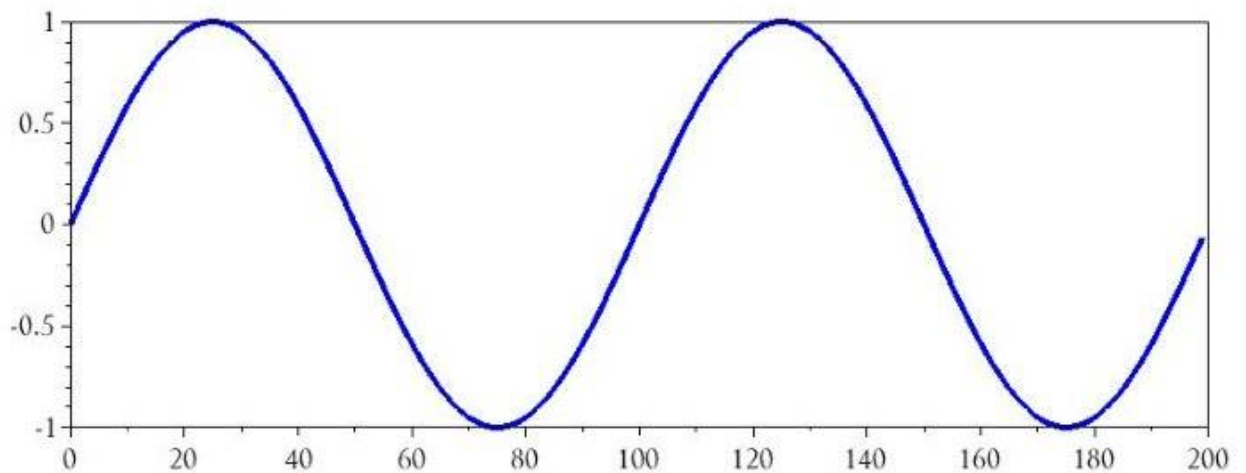
Другой причиной является отсутствие реальных частот физического сигнала при формальном представлении синусоидального сигнала, как в первой лабораторной работе при исследовании сигналов во временной области. При использовании команды **fft(BasebandSignal)** входной массив представляет собой последовательность формальных чисел. Эта последовательность чисел не связана с соответствующими частотами реального сигнала, и поэтому, чтобы создать частотный спектр реального сигнала, необходимо добавить в график информацию о значениях частот. Следует отметить, что команда **fft(BasebandSignal)** и команда **fft(BasebandSignal, -1)** дадут одинаковый результат Фурье преобразования, а команда **fft(BasebandSignal, +1)** позволяет получить результат обратного преобразования Фурье.

Дискретное преобразование Фурье синусоидального сигнала

В качестве немодулированного, низкочастотного сигнала выберем сигнал основной полосы 10 кГц, а несущую, для удобства работы, выберем в 100 кГц, хотя для реальных радиочастотных сигналов частота несущей должна быть значительно выше. Поэтому частоту дискретизации можно взять не очень высокой, например, равной 1 МГц.

Начнем с создания информационного сигнала основной полосы, а затем рассмотрим графики во временной и частотной областях.

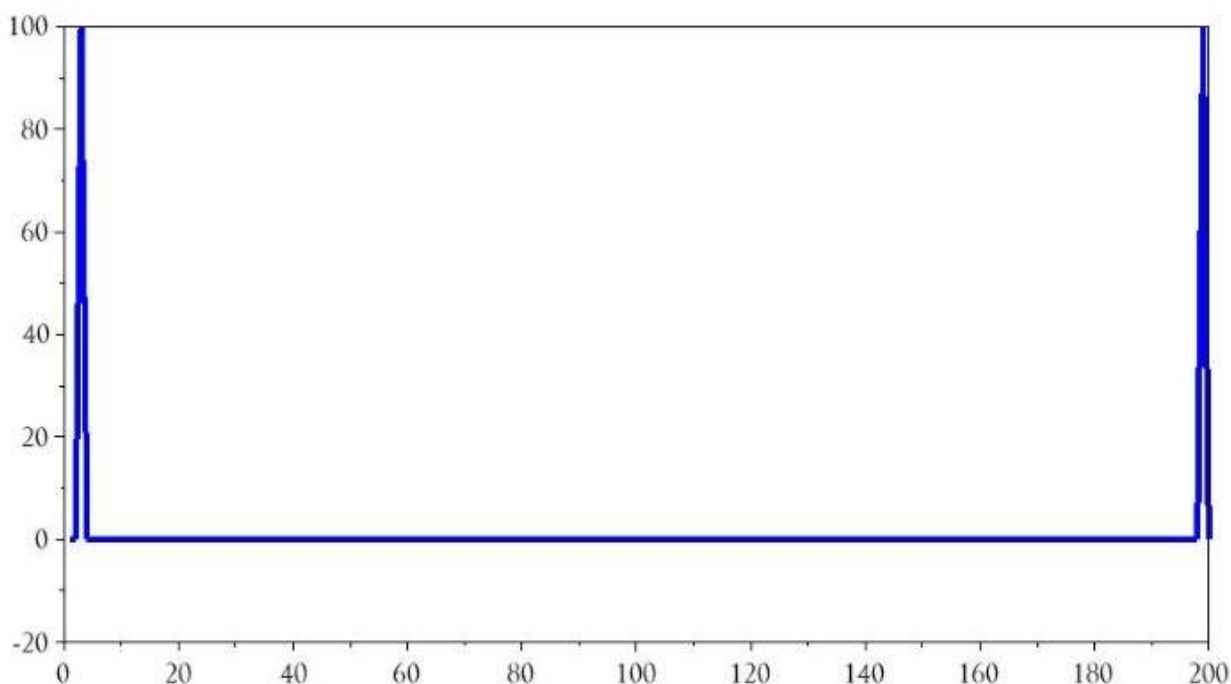
```
BasebandFrequency = 10e3;  
SamplingFrequency = 1e6;  
BufferLength = 200;  
n = 0:(BufferLength - 1);  
BasebandSignal = sin(2*%pi*n / (SamplingFrequency/BasebandFrequency));  
plot(n, BasebandSignal)
```



Представление низкочастотного сигнала во временной области

Следующие команды будут генерировать представление этого сигнала в частотной области:

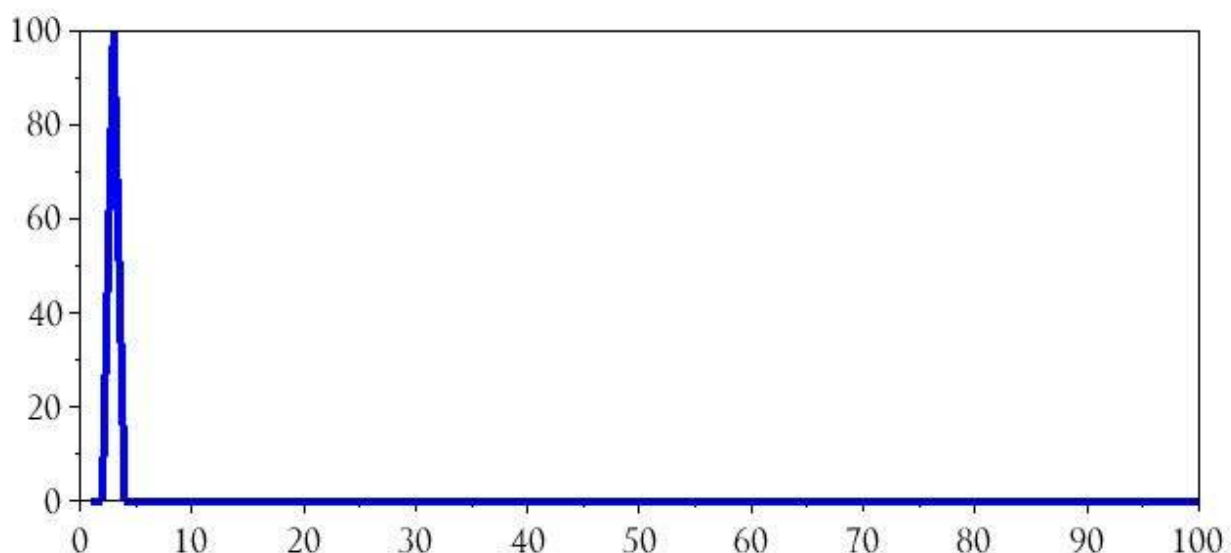
```
BasebandDFT = fft(BasebandSignal);  
BasebandDFT_magnitude = abs(BasebandDFT);  
plot(BasebandDFT_magnitude)
```

Представление низкочастотного сигнала в частотной области

Конечно, представленный график не является информативным, однако он представляет результат работы команд перехода в частотную область. На графике одночастотный синусоидальный сигнал представляется двумя частотными составляющими при обработке сигнала DFT. Появление на графике второй частотной составляющей связано с тем, что расчеты DFT генерируют симметричные результаты, т.е. правая половина графика является зеркальным отображением левой половины графика. Однако, устранить зеркальную составляющую можно следующим образом:

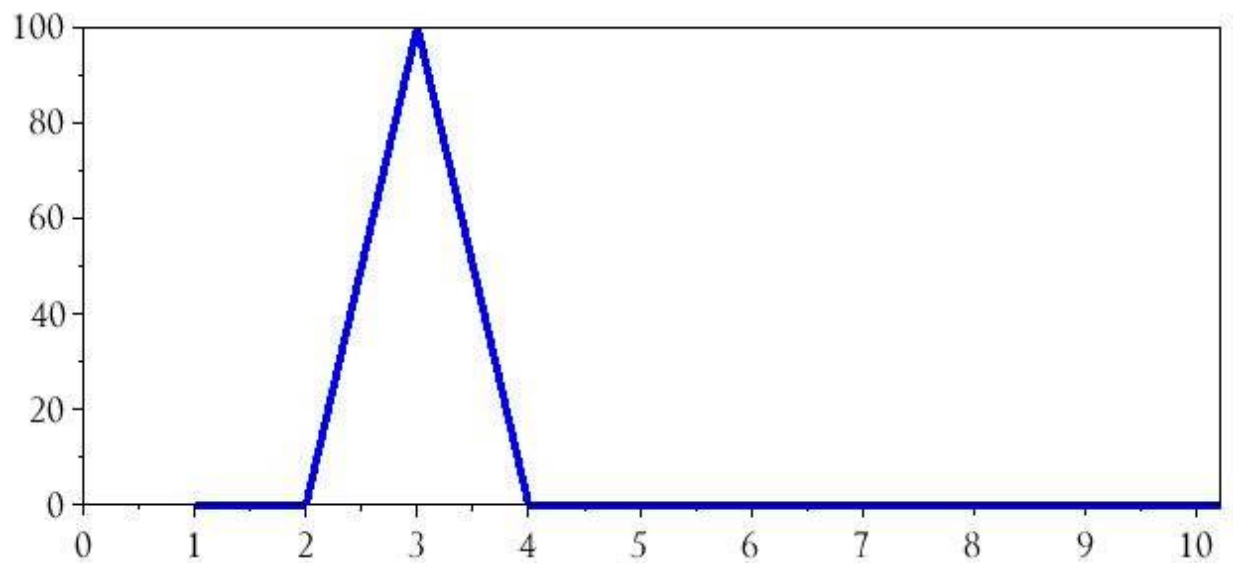
```
plot(BasebandDFT_magnitude(1:(BufferLength/2)))
```



Представление низкочастотного сигнала в частотной области при удалении зеркальной частотной составляющей.

Для устранения зеркальной составляющей была скорректирована команда `plot()` на отображение только первой половины данных. Полученный график сигнала в частотной

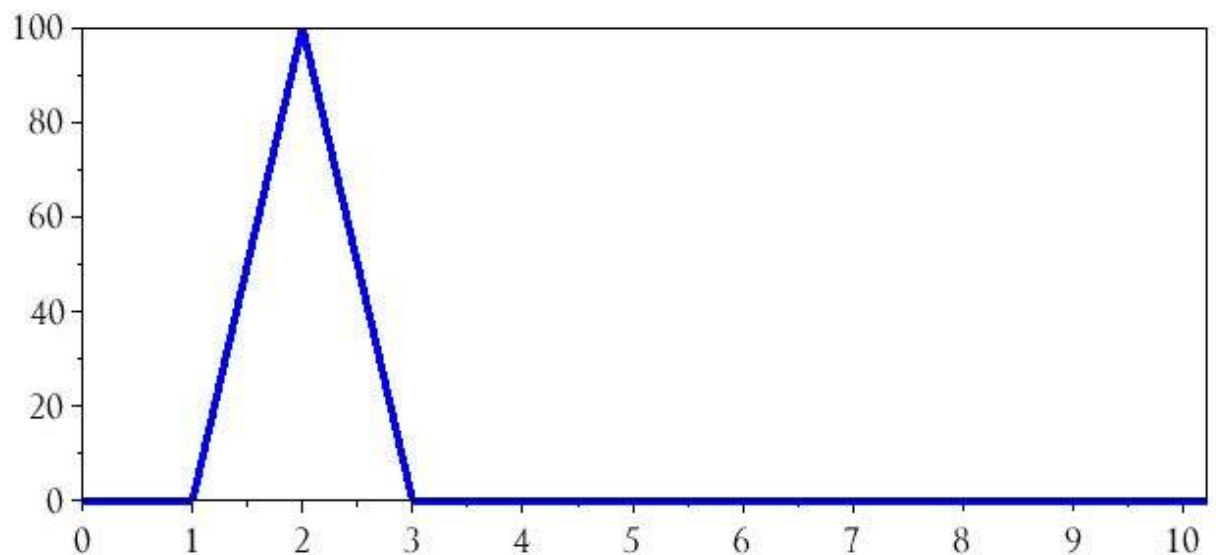
области можно представить более подробно, для этого масштаб графика необходимо увеличить.



Спектр низкочастотного сигнала в частотной области при увеличенном масштабе

Частотная составляющая спектра заданного низкочастотного сигнала отстоит от начала координат на 3 единицы. Это связано с тем, что во временной области при заданной длине массива в 200 единиц, который обрабатывался командой `fft()`, укладываются два периода низкочастотного синусоидального сигнала. Для дискретного преобразования Фурье DFT это соответствует формальной частоте 2, и поскольку график на горизонтальной оси начинается со значения 1, а не с 0, то частота 2 смещается по горизонтальной оси на отметку 3. Для сдвига частотной составляющей спектра на нужную отметку 2 необходимо привести в соответствие результаты преобразования FFT массиву `n`.

```
plot(n(1:(BufferLength/2)), BasebandDFT_magnitude(1:(BufferLength/2)))
```



Спектр низкочастотного сигнала в частотной области после перемещения частотной составляющей

Переход к значениям частот реальных сигналов

На спектре низкочастотного сигнала одна частотная составляющая на отметке 2, которая не связана с частотами реальных сигналов, выраженными, например, в кГц. Поэтому необходимо изменить размерность горизонтальной оси так, чтобы значения на ней соответствовали реальным частотам. Для этого следует учитывать следующие особенности.

1. На результат DFT влияет длина буфера. Если бы использовался буфер с 300 выборками, было бы 3 полных периода низкочастотного сигнала (сигнала основной полосы), и частотная составляющая DFT была бы не на отметке 2, а на отметке 3 горизонтальной оси. Следовательно, в корректирующее соотношение для единиц измерения частоты горизонтальной оси необходимо включить длину буфера.

2. Частота дискретизации является основным эталоном для всей системы дискретизации. Как только сигналы оцифрованы, их отношение к абсолютному времени заменяется отношением к частоте дискретизации.

3. Самая высокая частота, которую может обрабатывать система, равна частоте дискретизации (f_d), деленной на 2. В рамках реализуемой системы дискретизации частоты выше $f_d/2$ не существуют, потому что их нельзя отличить от соответствующих нижних частот.

Значения частотных составляющих на горизонтальной оси, формируемых как результат ДПФ, подсчитываются с равномерно разнесенными шагами от 0 до максимальной частоты системы, а размер шага определяется длиной буфера. Таким образом, для введения реальных частот в спектр ДПФ необходимо построить график амплитуд относительно последовательности значений, которые увеличиваются от 0 до $f_d/2$ (фактически, чтобы получить правильную длину массива, значения диапазона могут увеличиваться до значения непосредственно предшествующего значению $f_d/2$). Реализуется это следующим образом.

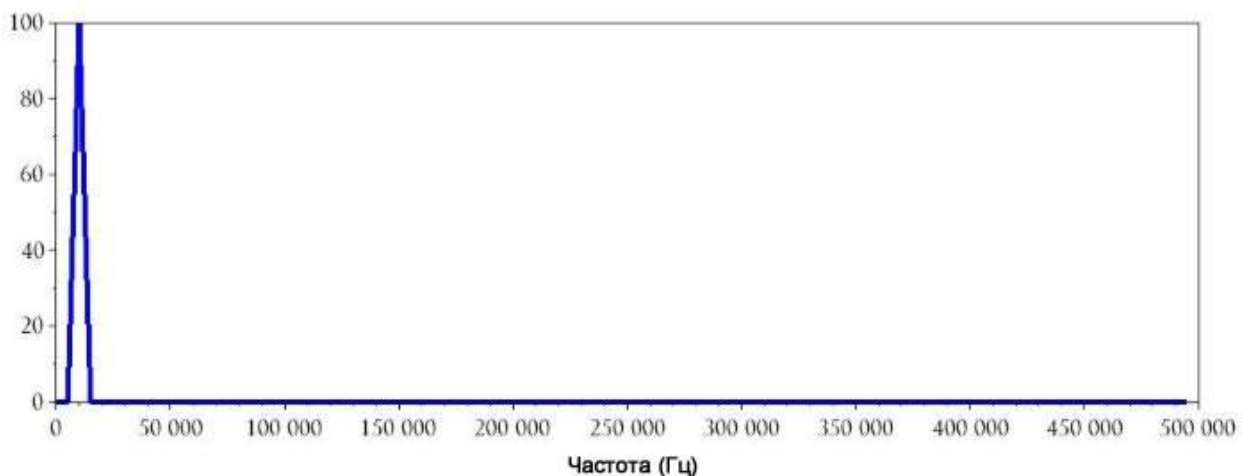
```
HalfBufferLength = BufferLength/2;
```

```
HorizAxisIncrement = (SamplingFrequency/2)/HalfBufferLength;
```

```
DFTHorizAxis = 0:HorizAxisIncrement:((SamplingFrequency/2)-HorizAxisIncrement);
```

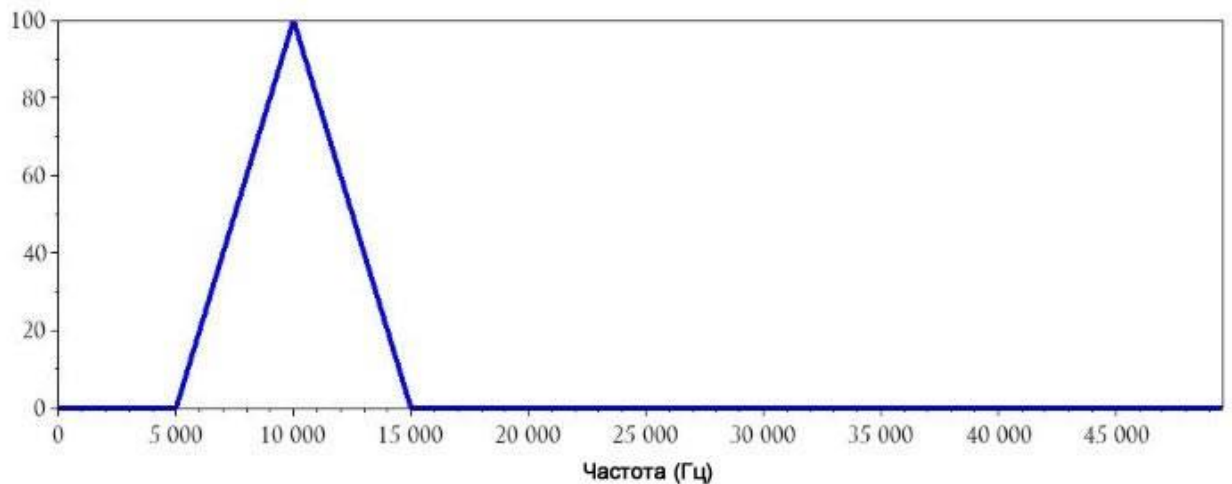
```
plot(DFTHorizAxis, BasebandDFT_magnitude(1:HalfBufferLength))
```

```
xlabel("Frequency (Hz)")
```



Спектр низкочастотного сигнала в частотной области с реальными частотами

Более детальный спектр с увеличенным масштабом по оси частот:



Спектр низкочастотного сигнала в частотной области с реальными частотами, в увеличенном масштабе

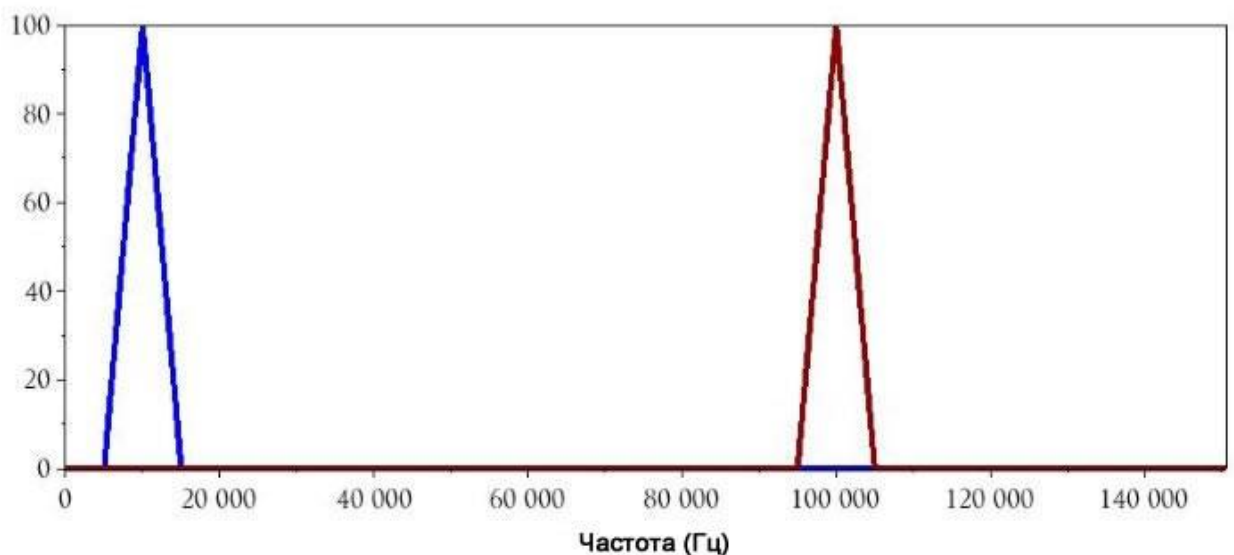
На графике частотная составляющая низкочастотного сигнала теперь представляется как 10 кГц. Для сигнала несущей частоты 100 кГц можно выполнить такие же процедуры.

```
CarrierSignal = sin(2*%pi*n / (SamplingFrequency/CarrierFrequency));
```

```
CarrierDFT = fft(CarrierSignal);
```

```
CarrierDFT_magnitude = abs(CarrierDFT);
```

```
plot(DFTHorizAxis, CarrierDFT_magnitude(1:HalfBufferLength))
```



Спектр сигналов низкочастотного (сигнала основной полосы) и несущей в частотной области

Задание на лабораторную работу 2

1. Изучить описание лабораторной работы Анализ сигналов в частотной области с использованием быстрого преобразования Фурье (БПФ, FFT) в Scilab.

2. Получить у преподавателя значение частот для моделирование низкочастотного сигнала основной полосы и несущей, выполнить их генерацию с помощью пакета Scilab во временной области, выполнить преобразование Фурье DFT этих сигналов, получить их представление в частотной области.

3. Построить графики заданных сигналов и уменьшенного по частоте на 0,5 кГц заданного низкочастотного сигнала во временной и частотных областях после преобразования Фурье DFT в реальных значениях частот, выраженных в кГц в соответствии с рекомендациями описания лабораторной работы.

4. Повторно выполнить преобразование Фурье DFT для всех сигналов, полученных на последнем графике в частотной области с реальными значениями частот, выраженных в кГц. Выполнить обратное преобразование Фурье. Сравнить полученные сигналы после повторного преобразования Фурье DFT и обратного преобразования Фурье с исходными сигналами.

5. Сделать выводы по результатам работы.

Лабораторная работа 3. Анализ амплитудно-модулированных радиочастотных сигналов с помощью Scilab

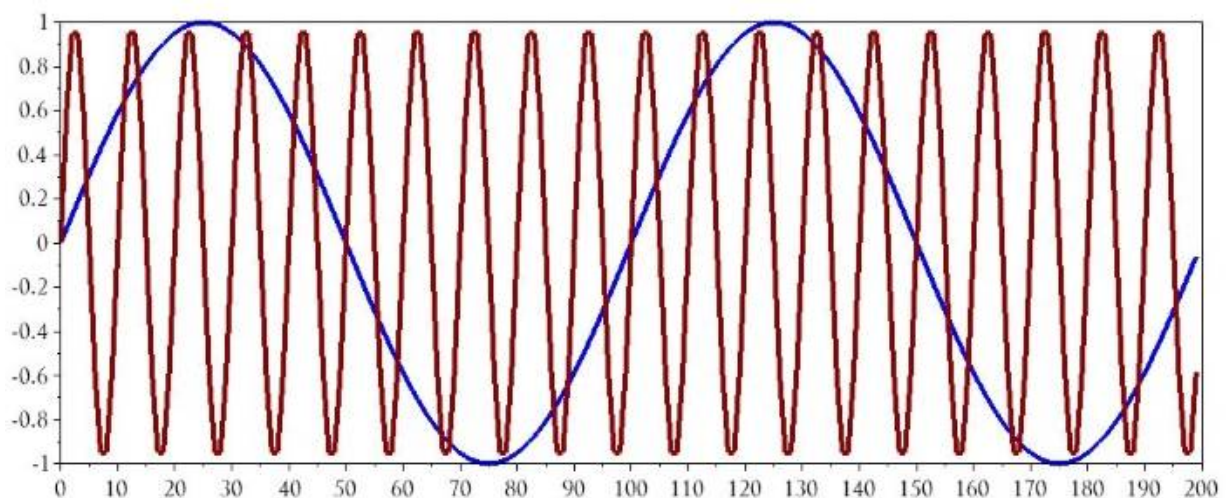
Известно, что амплитудная модуляция приводит к изменениям сигнала, которые обычно хорошо видны на диаграмме модулированного сигнала во временной области. Однако, спектр модулированного сигнала в частотной области представляет важную информацию, которую трудно или в некоторых случаях невозможно получить из диаграммы во временной области. Анализ частотного спектра позволяет оценить взаимодействие между частотами низкочастотного модулирующего сигнала и более высокой частотой сигнала несущей.

Формирование графиков в частотной области для модулированных сигналов

Формирование графиков в частотной области необходимо начать с описания основных переменных, которые будут использоваться в дальнейших построениях.

```
BasebandFrequency = 10e3;  
CarrierFrequency = 100e3;  
SamplingFrequency = 1e6;  
BufferLength = 200;  
n = 0:(BufferLength - 1);  
BasebandSignal = sin(2*%pi*n / (SamplingFrequency/BasebandFrequency));  
CarrierSignal = sin(2*%pi*n / (SamplingFrequency/CarrierFrequency));  
plot(n, BasebandSignal)  
plot(n, CarrierSignal)
```

Представленные в описании сигналы можно показать в графическом виде.



Графики во временной области низкочастотного сигнала или сигнала основной полосы и сигнала несущей

Реализация амплитудной модуляции одночастотным сигналом

Математическим соотношением, описывающим амплитудную модуляцию, можно представить в виде:

$$x_{AM}(t) = \sin(\omega_c t) (1 + x_{нч}(t))$$

В формуле сигнал несущей представлен функцией $\sin(\omega_c t)$. Для реализации амплитудной модуляции необходимо выполнить сдвиг низкочастотного информационного сигнала или сигнала основной полосы вверх так, чтобы его значения всегда были положительными. Далее необходимо умножить несущую на сдвинутый низкочастотный сигнал. Команда Scilab, соответствующая этим преобразованиям, может быть записана в виде:

```
ModulatedSignal_AM = CarrierSignal .* (1+BasebandSignal);
```

В представленной формуле следует обратить внимание на точку перед звездочкой. В системе Scilab моделируемые сигналы `CarrierSignal` и `BasebandSignal` представляются матрицами, и при использовании только звездочки между матрицами, система Scilab будет выполнять матричное умножение. При этом выполнить матричное умножение в этом случае невозможно, так как количество столбцов в первой матрице не равно количеству строк во второй матрице. Кроме этого, умножение матриц не даст нужного результата. Для получения необходимого результата следует выполнить типовое аналоговое умножение, при котором мгновенное значение одного сигнала многократно умножается на соответствующее мгновенное значение другого сигнала. Следовательно, необходимо использовать последовательное поэлементное умножение, при котором первый элемент `CarrierSignal` умножается на первый элемент `BasebandSignal`, второй элемент `CarrierSignal` умножается на второй элемент `BasebandSignal`, и так далее. Таким образом, нужная процедура умножения реализуется при использовании операции `.*` вместо операции `*`.

Для представления результатов во временной области необходимо использовать команды:

```
plot(n, BasebandSignal)
```

```
plot(n, ModulatedSignal_AM)
```

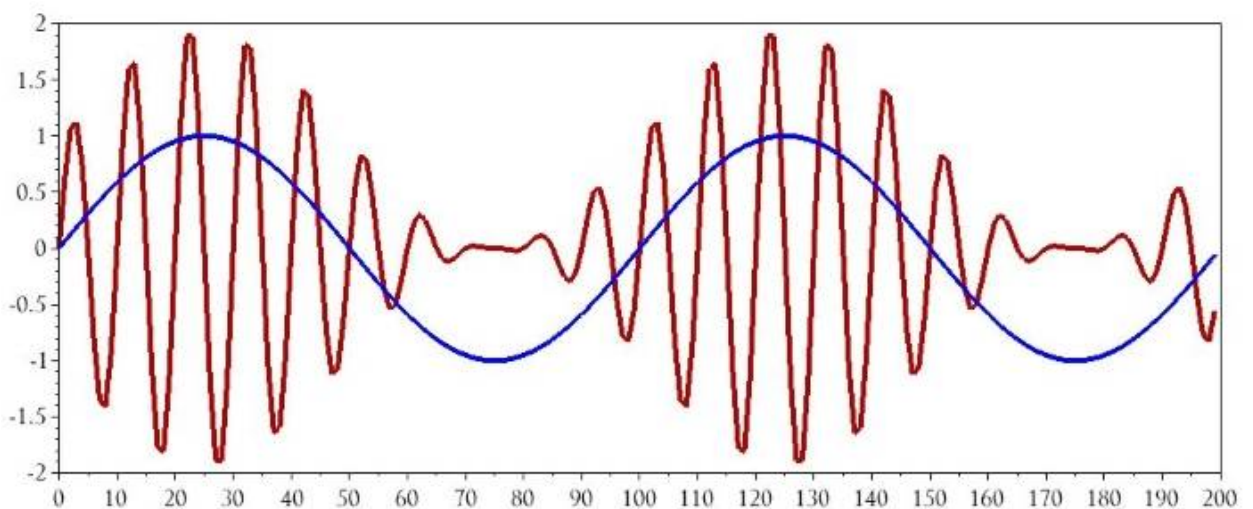


График полученного амплитудно-модулированного сигнала во временной области

Далее для перехода из временной в частотную область необходимо сначала сформировать горизонтальную ось для представления частотного спектра сигнала после преобразования Фурье (DFT) с помощью команд:

```
HalfBufferLength = BufferLength/2;
```

```
HorizAxisIncrement = (SamplingFrequency/2)/HalfBufferLength;
```

```
DFTHorizAxis = 0:HorizAxisIncrement:((SamplingFrequency/2)-HorizAxisIncrement);
```

С помощью следующих команд можно получить график амплитудно-модулированного сигнала в частотной области:

```
AM_DFT = fft(ModulatedSignal_AM);
```

```
AM_DFT_magnitude = abs(AM_DFT);
```

```
plot(DFTHorizAxis, AM_DFT_magnitude(1:HalfBufferLength))
```

```
xlabel("Frequency (Hz)")
```

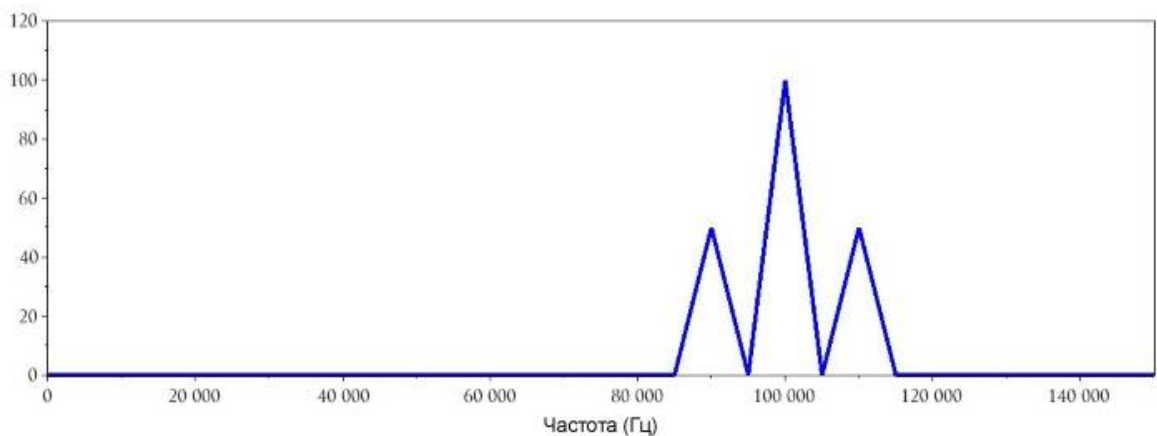


График полученного амплитудно-модулированного сигнала в частотной области

На полученном графике амплитудно-модулированного сигнала представлен спектр, состоящий из основной несущей частоты максимальной амплитуды и двух боковых составляющих, соответствующих частоте несущего колебания плюс и минус значение частоты низкочастотного сигнала. Умножение низкочастотного сигнала на сигнал несущей привело к сдвигу спектра низкочастотного сигнала к значению частоты несущего колебания и появлению двух спектральных составляющих, центрированных относительно несущей частоты, соответствующих положительному и отрицательному значениям 10кГц.

Реализация амплитудной модуляции модулирующим сигналом, содержащим несколько значений частот

Поскольку модуляция используется для передачи информации, модулирующий сигнал обычно не является одночастотным и представляется гораздо более сложным сигналом, состоящим из большого числа гармоник. Например, при передаче звукового или речевого сигнала или изменяющихся показаний температуры, давления и влажности частота информационного низкочастотного сигнала основной полосы будет изменяться в некотором диапазоне. Чтобы смоделировать более реальный спектр амплитудно-модулированного сигнала можно сформировать низкочастотный сигнал основной полосы,

содержащий несколько значений частот, а далее использовать рассмотренную ранее процедуру Фурье преобразования (DFT).

Для формирования более сложного низкочастотного модулирующего сигнала целесообразно увеличить объем буфера и создать, например, пять гармонических сигналов с различными частотами с помощью следующих команд.

```
BufferLength = 1000;  
n = 0:(BufferLength - 1);  
Baseband_1 = sin(2*%pi*n / (SamplingFrequency/BasebandFrequency));  
Baseband_2 = sin(2*%pi*n / (SamplingFrequency/(BasebandFrequency/2)));  
Baseband_3 = sin(2*%pi*n / (SamplingFrequency/(BasebandFrequency/4)));  
Baseband_4 = sin(2*%pi*n / (SamplingFrequency/(BasebandFrequency/8)));  
Baseband_5 = sin(2*%pi*n / (SamplingFrequency/(BasebandFrequency/10)));  
BasebandSignal = [Baseband_1 Baseband_2 Baseband_3 Baseband_4 Baseband_5];  
BufferLength = BufferLength * 5;  
n = 0:(BufferLength - 1);
```

Представленные команды позволяют сформировать пять синусоидальных гармонических сигналов с различными частотами в диапазоне от 1кГц до 10кГц. При этом, чтобы каждый из синусоидальных сигналов имел не менее одного периода в модулирующем сигнале, длина буфера увеличена до 1000. Далее выполнено объединение синусоидальных сигналов в общий низкочастотный модулирующий сигнал основной полосы при увеличенной длине буфера. Поэтому график модулирующего сигнала во временной области будет иметь следующий вид.

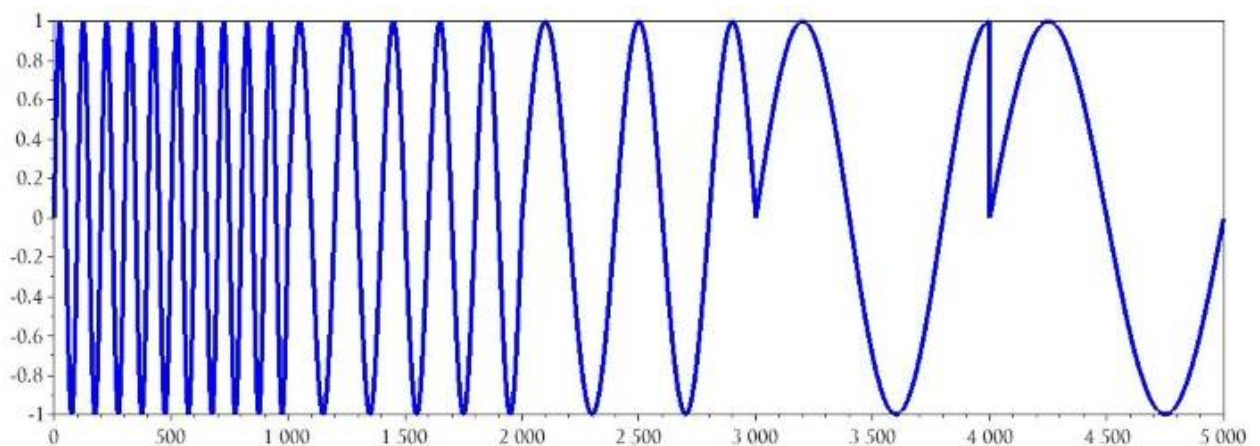


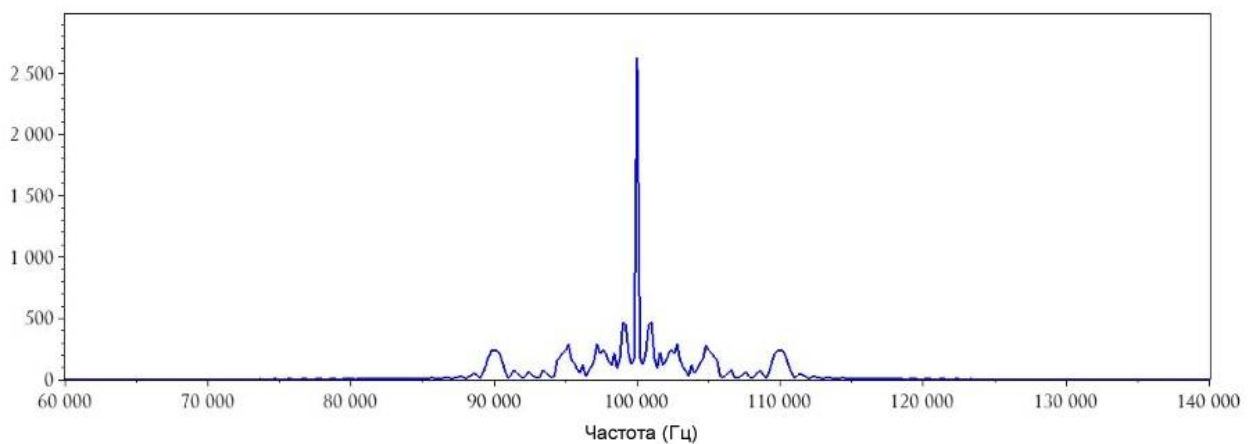
График модулирующего сигнала основной полосы, содержащего несколько значений частот, во временной области

Не смотря на то, что частота несущего сигнала оставлена без изменений, так как длины массивов CarrierSignal и BasebandSignal должны быть одинаковы, необходимо сформировать новый сигнал. Для этого можно использовать следующую команду:

```
CarrierSignal = sin(2*%pi*n / (SamplingFrequency/CarrierFrequency));
```

Далее можно реализовать новый амплитудно-модулированный сигнал с полученной низкочастотной огибающей, содержащей несколько значений частот. Для этого нужно сформировать новую последовательность значений для горизонтальной оси частотного спектра, выполнить преобразование Фурье (DFT) и вывести полученные данные на график в частотной области.

```
ModulatedSignal_AM = CarrierSignal .* (1+BasebandSignal);  
HalfBufferLength = BufferLength/2;  
HorizAxisIncrement = (SamplingFrequency/2)/HalfBufferLength;  
DFTHorizAxis = 0:HorizAxisIncrement:((SamplingFrequency/2)-HorizAxisIncrement);  
AM_DFT = fft(ModulatedSignal_AM);  
AM_DFT_magnitude = abs(AM_DFT);  
plot(DFTHorizAxis, AM_DFT_magnitude(1:HalfBufferLength))
```



Спектр АМ сигнала, модулированного сигналом, содержащим несколько значений частот.

Задание на лабораторную работу 3

1. Изучить описание лабораторной работы Анализ амплитудно-модулированных радиочастотных сигналов с помощью Scilab.
2. Получить у преподавателя значение частот для модулирующего низкочастотного сигнала основной полосы и несущей, выполнить их генерацию с помощью пакета Scilab во временной области, сформировать амплитудно-модулированный сигнал во временной области, получить спектр амплитудно-модулированного сигнала в частотной области.
3. Сформировать низкочастотный модулирующий сигнал основной полосы во временной области, содержащий последовательность из нескольких сигналов, имеющих различные значения частот, в соответствии с рекомендациями описания лабораторной работы.
4. Сформировать низкочастотный модулирующий сигнал основной полосы, содержащий одновременно сигналы с разными значениями частот, выполнить амплитудную модуляцию этим сигналом несущего сигнала во временной области.
5. Получить спектры АМ сигналов, модулированных сигналами, содержащими несколько значений частот, сформированными в п.3 и п.4, сравнить их между собой.
6. Сделать выводы по результатам работы.

Лабораторная работа 4. Анализ частотно-модулированных радиочастотных сигналов с помощью Scilab

Использование при амплитудной модуляции в качестве основной операции операцию умножения сигналов во временной области, смещает спектр амплитудно-модулированного сигнала, располагая его симметрично относительно несущей или модулируемой частоты. Поэтому обычно результаты амплитудной модуляции в частотной области понятны и легко воспринимаются. Однако, при частотной модуляции сигнал подвергается более сложным преобразованиям и может быть представлен математическим соотношением:

$$x_{\text{ЧМ}}(t) = \sin(\omega_{\text{нес}}t + \int_{-\infty}^t x_{\text{нч}}(t)dt)$$

Таким образом, частотно-модулированный сигнал реализуется при сложении аргумента тригонометрической функции синуса несущей частоты с интегралом от модулирующего сигнала. Следовательно, частотно-модулированный сигнал содержит $\sin(\omega_{\text{нес}}t)$, то есть синусоидальный несущий сигнал с угловой частотой $\omega_{\text{нес}}$ с добавлением изменяющейся во времени фазы, вычисляемой как интеграл от низкочастотного модулирующего сигнала.

Известно, что с частотной модуляцией связана фазовая модуляция, которую можно описать выражением:

$$x_{\text{ФМ}}(t) = \sin(\omega_{\text{нес}}t + x_{\text{нч}}(t))$$

Поэтому для анализа сигнала, модулированного по фазе, а не по частоте, в качестве аргумента фазы сигнала необходимо использовать не интеграл от модулирующего низкочастотного сигнала, а сам сигнал, изменяющийся во времени. Все остальные рекомендации описания этой лабораторной работы, относящиеся к частотно-модулированному сигналу, можно применить в равной степени к сигналу, модулированному по фазе.

Реализация интегрирования низкочастотного модулирующего сигнала

Для получения модулирующего низкочастотного сигнала необходимо задать массивы для низкочастотного модулирующего сигнала и несущего сигнала. При частотной модуляции, в отличие от амплитудной, для расширения диапазона изменения частоты полезно увеличить частоту дискретизации и длину буфера, например, в десять раз, то есть, чтобы на высокочастотных участках модулированного сигнала было больше выборок за период для более точного их представления. Таким образом, описание низкочастотного модулирующего сигнала и несущего сигнала можно представить в виде:

```
BasebandFrequency = 10e3;  
CarrierFrequency = 100e3;  
SamplingFrequency = 1e7;  
BufferLength = 2000;  
n = 0:(BufferLength - 1);  
BasebandSignal = sin(2*%pi*n / (SamplingFrequency/BasebandFrequency));  
CarrierSignal = sin(2*%pi*n / (SamplingFrequency/CarrierFrequency));
```

```
plot(n, BasebandSignal)
```

```
plot(n, CarrierSignal)
```

График сигналов несущей и низкочастотного модулирующего сигнала основной полосы во временной области в соответствии со сделанным выше описанием:

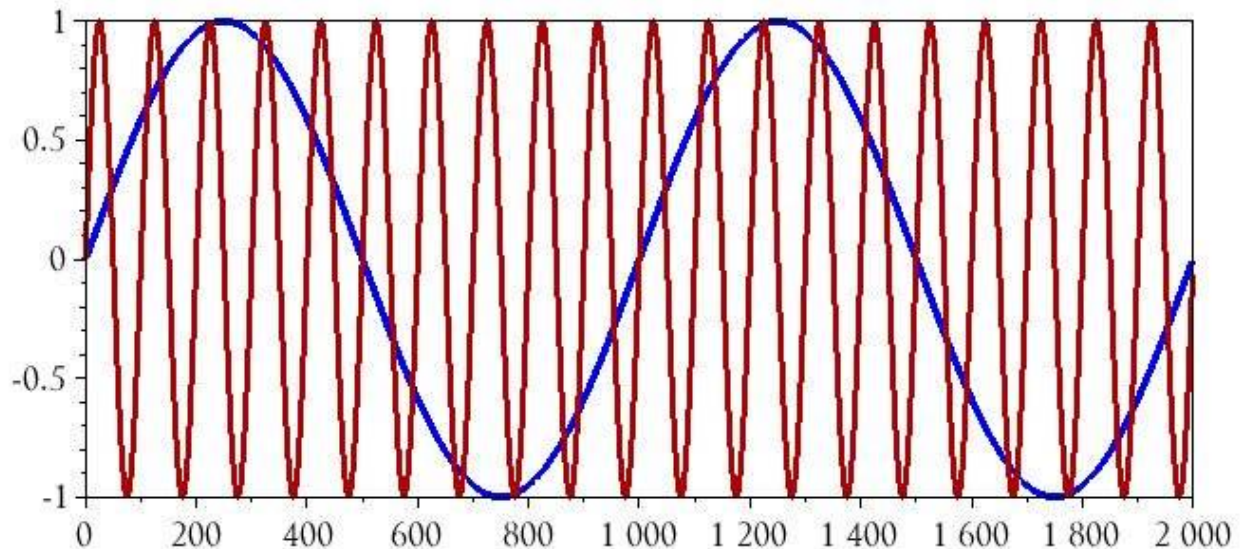


График во временной области несущего сигнала и низкочастотного модулирующего сигнала

Для вычисления неопределенного интеграла низкочастотного модулирующего сигнала основной полосы в Scilab можно использовать команду `integrate()`, однако, применение команды `integrate()` имеет определенные особенности и может вызывать дополнительные трудности. Для расчета интеграла более простым способом можно использовать следующую информацию:

- низкочастотный модулирующий сигнал основной полосы является одночастотным синусоидальным сигналом,

- результатом вычисления неопределенного интеграла синусоидального сигнала является отрицательный косинусоидальный сигнал, к которому добавляется константа, равная нулю в решаемой задаче.

Таким образом, для вычисления интеграла можно просто заменить команду: `BasebandSignal = sin(...)` на `BasebandSignal_integral = -cos(...)`.

Следовательно, с учетом отмеченной замены получаем:

```
BasebandSignal_integral = -cos(2*%pi*n / (SamplingFrequency/BasebandFrequency));
```

```
plot(n, BasebandSignal)
```

```
plot(n, BasebandSignal_integral)
```

При этом график исходного и проинтегрированного сигналов можно представить в виде:

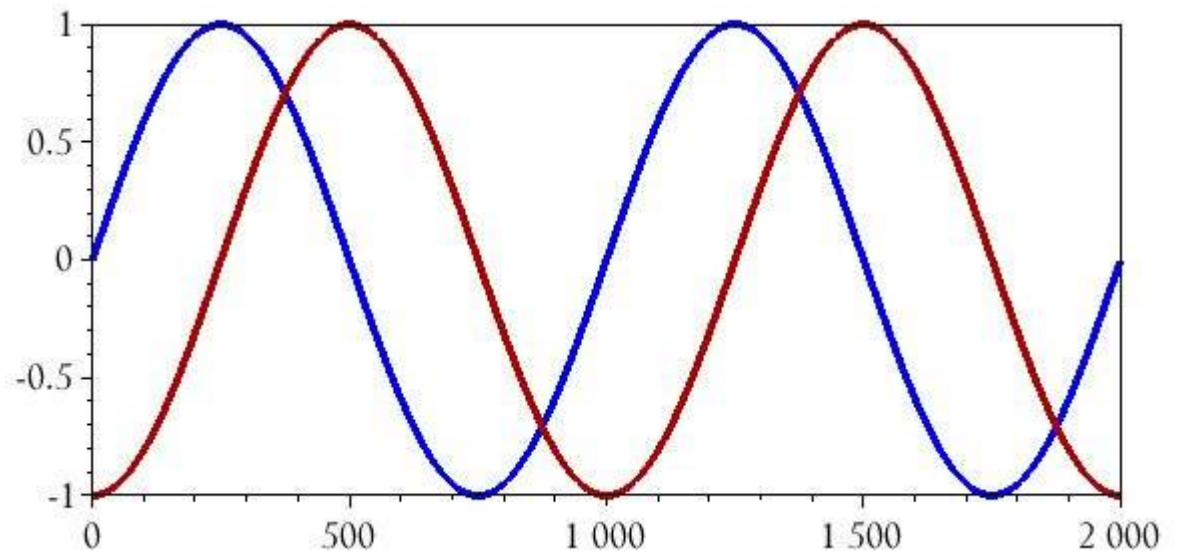


График низкочастотного модулирующего сигнала основной полосы до интегрирования изображен синим цветом и после интегрирования изображен красным цветом.

Формирование частотно-модулированного сигнала во временной области

Используя полученный проинтегрированный модулирующий сигнал, можно сформировать частотно-модулированный сигнал во временной области. Для этого необходимо в команде формирования сигнала несущей добавить рассчитанные значения интеграла в виде массива `BasebandSignal_integral` к аргументу функции `sin()` в соответствии с приведенном выше математическим соотношением. После этого вывести результат в виде графика. Таким образом, можно записать:

```
ModulatedSignal_FM = sin((2*%pi*n / (SamplingFrequency/CarrierFrequency)) +  
BasebandSignal_integral);
```

```
plot(n, ModulatedSignal_FM)
```

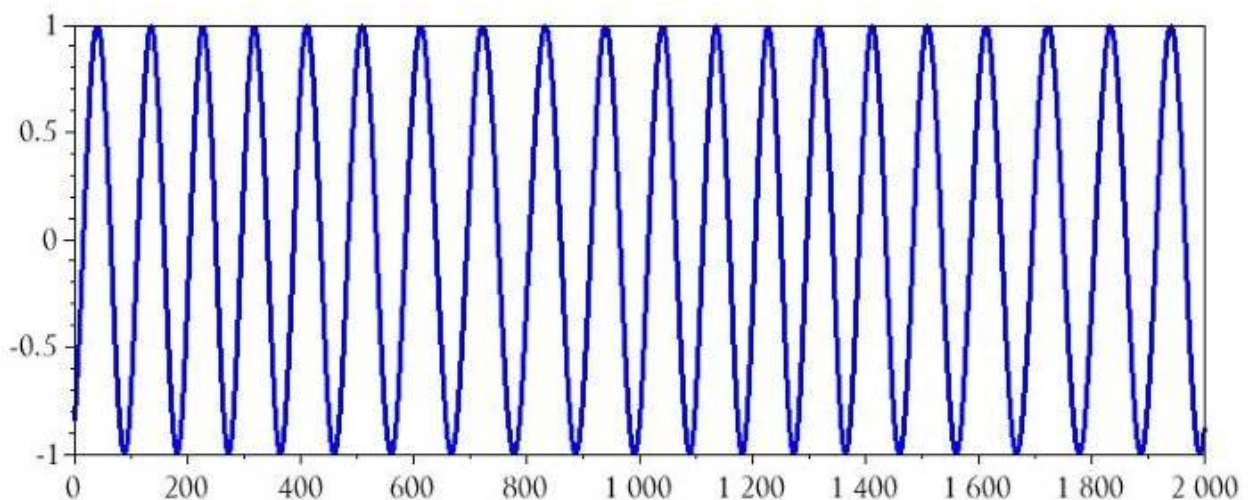


График полученного частотно-модулированного сигнала во временной области

Однако, в представленном графике частотная модуляция плохо видна, поскольку изменения частоты достаточно малы по сравнению с несущей частотой. Известно, что для задания величины изменения частоты при частотной модуляции вводится, так называемый, индекс модуляции, обозначенный символом m . Таким образом, индекс модуляции, используется как для увеличения, так и для уменьшения величины изменения значения частоты, вызванной заданным значением модулирующего сигнала:

$$x_{\text{ЧМ}}(t) = \sin(\omega_{\text{нес}}t + m \cdot \int_{-\infty}^t x_{\text{нч}}(t)dt)$$

Для получения на графике более очевидного и наглядного эффекта модуляции необходимо в команду генерации данных частотной модуляции включить увеличенный индекс модуляции, равный, например, четырем.

```
ModulatedSignal_FM = sin((2*%pi*n / (SamplingFrequency/CarrierFrequency)) +  
(4*BasebandSignal_integral));  
plot(n, ModulatedSignal_FM)
```

В результате получим график с более очевидным эффектом модуляции.

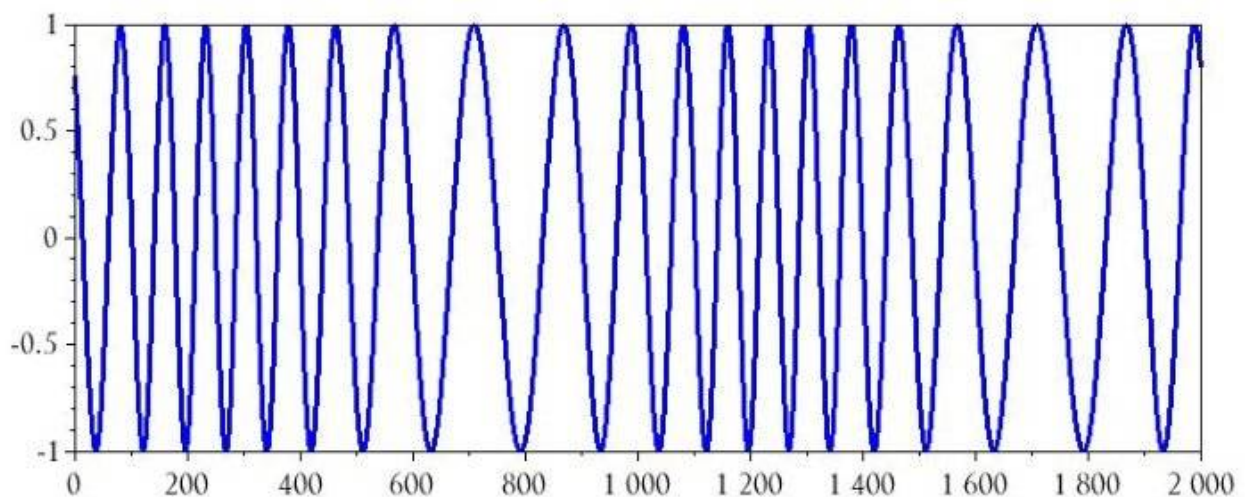


График частотно-модулированного сигнала во временной области со значением индекса модуляции, равным 4

Для иллюстрации соотношения между исходным модулирующим сигналом, модулирующим сигналом после интегрирования и результирующим частотно-модулированным сигналом можно построить их все три на одном графике, то есть добавить на этот график еще два сигнала с помощью команд:

```
plot(n, BasebandSignal)  
plot(n, BasebandSignal_integral)
```

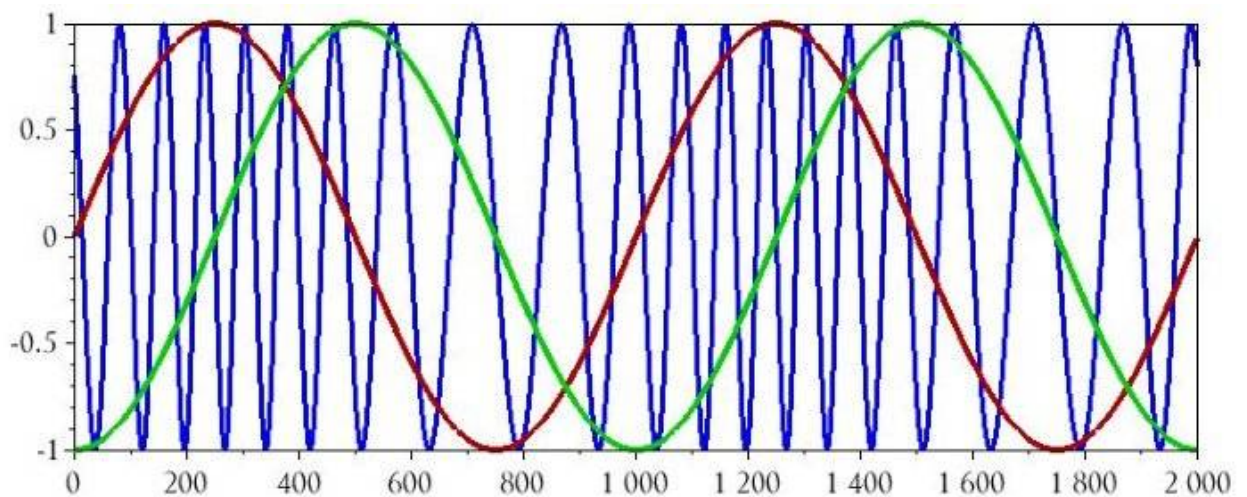


График исходного низкочастотного модулирующего сигнала, модулирующего сигнала после интегрирования и результирующего частотно-модулированного сигнала во временной области.

Формирование частотно-модулированного сигнала в частотной области

Для формирования и иллюстрации частотно-модулированного сигнала в частотной области необходимо использовать следующую последовательность команд:

```
HalfBufferLength = BufferLength/2;
HorizAxisIncrement = (SamplingFrequency/2)/HalfBufferLength;
DFTHorizAxis = 0:HorizAxisIncrement:((SamplingFrequency/2)-HorizAxisIncrement);
FM_DFT = fft(ModulatedSignal_FM);
FM_DFT_magnitude = abs(FM_DFT);
plot(DFTHorizAxis, FM_DFT_magnitude(1:HalfBufferLength))
xlabel("Frequency (Hz)")
```

При выполнении этих команд получим график частотно-модулированного сигнала в частотной области, выраженный в Гц:

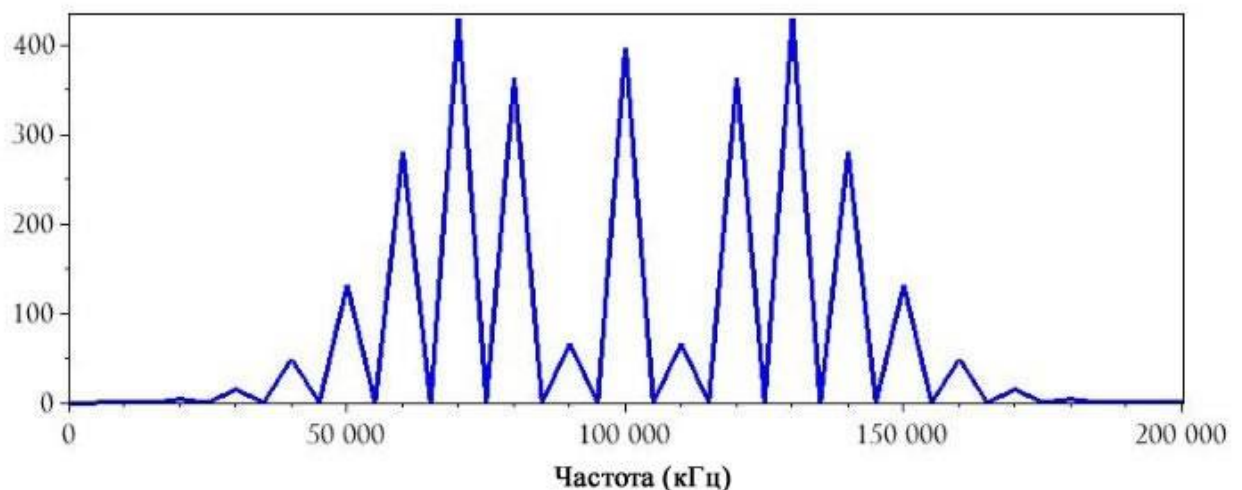


График частотно-модулированного сигнала в частотной области

Следует отметить, что амплитуда боковых частотных составляющих спектра может быть выше частотной составляющей на основной несущей частоте. Кроме того, для модулированного сигнала диапазон или полоса частот относительно несущей частоты составляет около ± 70 кГц, что значительно превышает диапазон или ширину полосы частот модулирующего сигнала, равную ± 10 кГц.

Указанные результаты и особенности не обязательно должны проявляться на всех случаях частотной модуляции, так как на характеристики спектра частотно-модулированного сигнала влияют различные факторы. Так при уменьшении индекса модуляции с четырех до двух, диапазон или полоса частот относительно несущей частоты уменьшится до приблизительно ± 40 кГц, что хорошо видно на графике с частотами, выраженными в Гц:

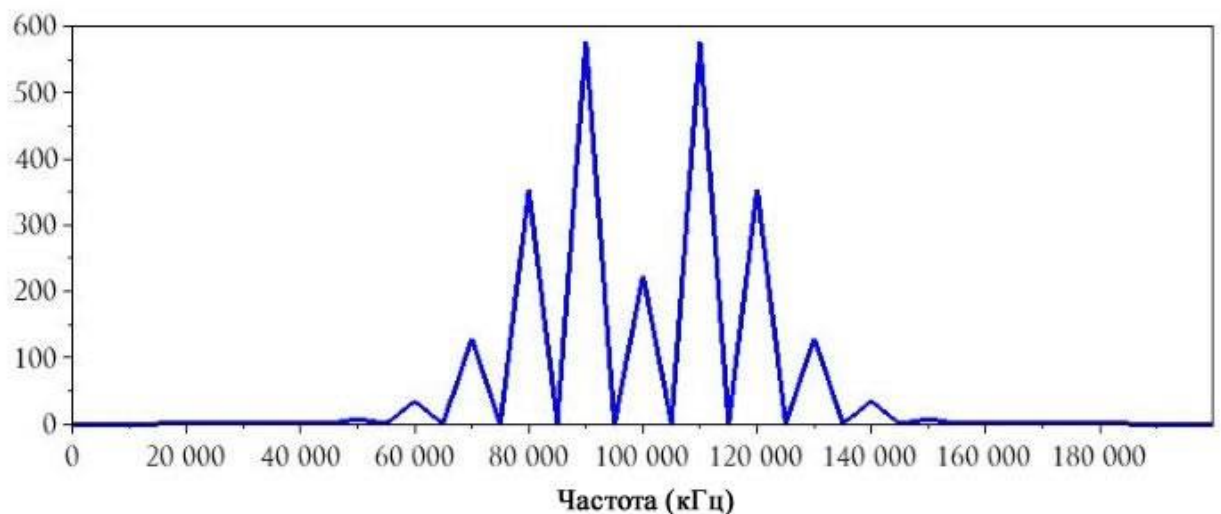


График частотно-модулированного сигнала в частотной области с индексом модуляции 2

Если уменьшить частоту модулирующего сигнала в два раза, то есть задать, равную 5 кГц и увеличить индекс модуляции до четырех, получим следующий график частотно-модулированного сигнала в частотной области.

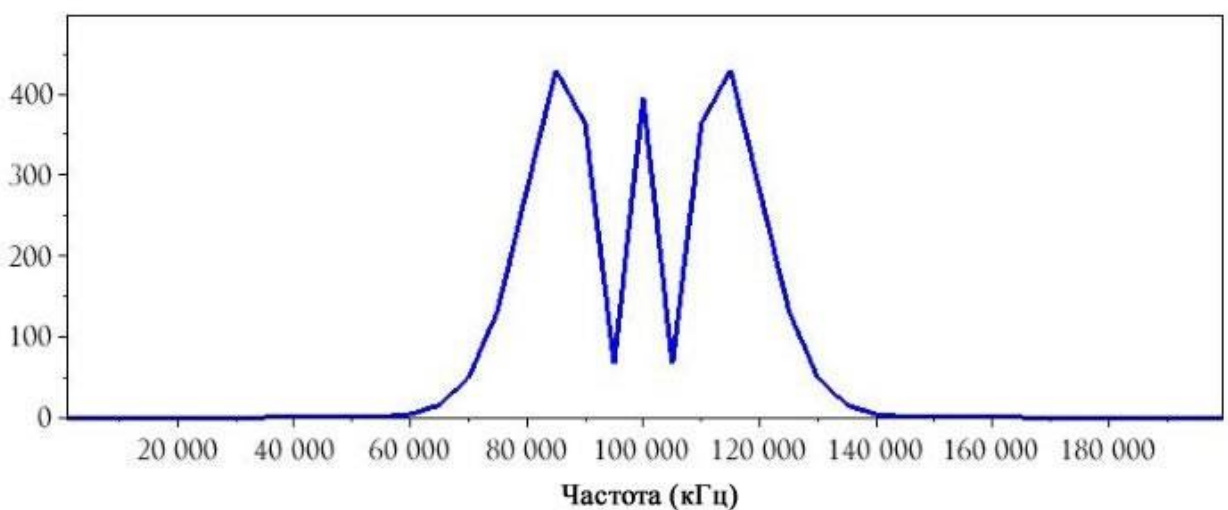


График частотно-модулированного сигнала в частотной области при частоте модулирующего сигнала 5 кГц и индексе модуляции 4.

Задание на лабораторную работу 4

1. Изучить описание лабораторной работы Анализ частотно-модулированных радиочастотных сигналов с помощью Scilab.
2. Получить у преподавателя значение частот для модулирующего низкочастотного сигнала основной полосы и несущей, выполнить их генерацию с помощью пакета Scilab во временной области, сформировать и построить график частотно-модулированного сигнала во временной области.
3. Сформировать и построить график частотно-модулированного сигнала во временной области со значением индекса модуляции, равным 4. Построить на одном графике исходный низкочастотный модулирующий сигнал, модулирующий сигнал после интегрирования и результирующий частотно-модулированный сигнал во временной области в соответствии с рекомендациями описания лабораторной работы
4. Получить спектр сформированного частотно-модулированного сигнала в частотной области. Сформировать спектр частотно-модулированного сигнала в частотной области с индексом модуляции 2 и представить его на графике.
5. Получить спектр частотно-модулированного сигнала в частотной области при снижении частоты модулирующего сигнала в два раза и индексе модуляции, равным 4, построить соответствующий график.
6. Сделать выводы по результатам работы.

Лабораторная работа 5. Применение фильтров для удаления шума в аудиозаписи с помощью Scilab

Как правило любая обработка звукового сигнала приводит к внесению в него изменений, которые часто приводят к снижению его качества. Поэтому при любых преобразованиях звукового сигнала стараются не допустить снижения качества обрабатываемого сигнала или максимально его улучшить, особенно когда исходный сигнал не имел высокого качества или содержал шумы. В настоящее время обработка звуковых сигналов, например, их запись и воспроизведение стали бытовыми, обыденными процедурами, поэтому редко кто задумывается о снижении качества звука при их выполнении. Однако, оптимизация качества звуковых сигналов должна выполняться на всех устройствах приема, обработки и передачи звуковых сигналов, то есть от микрофона, преобразователей сигнала до выходных акустических систем, воспроизводящих звук. В частности, интерес представляет обработка цифровых звуковых сигналов на блоках DSP (сигнальных процессорах), в которых цифровые звуковые сигналы обрабатываются с помощью специальных разработанных алгоритмов. Кроме того, подобная обработка позволяет снизить недостатки, вносимые другими устройствами в цепочке преобразования звуковых сигналов.

Снижение фоновых звуков шипения

Одной из известных и важных задач является удаление фоновых звуков при записи речи человека. К таким фоновым звукам, обычно сопровождающих любую запись речи человека, относят фоновое шипение, хорошо прослушиваемое в начале и конце записи и

иногда между словами в речевых сообщениях. В качестве примера возьмем аудио файл записи первых двух строк "Ворона" Эдгара Аллана По. Эффект шипения, конечно, присутствует на всем протяжении записи, но наиболее он заметен в начале и конце записи, когда, соответственно, речевое сообщение еще не началось или уже закончилось.



01 OnceUponaMidnightDreary.wav

Известно, что полное удаление шипения из записанного файла является достаточно сложной задачей, которая не всегда может быть решена. Для удаления фоновых звуков шипения обычно применяют фильтры, которые ослабляют шипение, но не удаляют его полностью, особенно, если используются простые фильтры.

Загрузка звукового файла и его анализ в Scilab

Для подключения, ввода звукового файла и последующего его анализа необходимо преобразовать входной WAV файл в переменные Scilab с помощью команды, в которой нужно прописать полный путь до этого файла с указанием его имени, то есть:

C:\Полный путь\Имя файла.wav. Например, команда может иметь вид:

```
[OriginalAudio, Fs] =  
wavread("C:\Users\Vladimir\Documents\OnceUponaMidnightDreary.wav");
```

Следует отметить, что команда `wavread()` может выполняться немного иначе в последней версии и отличаться от предыдущей. Так, можно включить в команду две возвращаемые переменные и с помощью второй из них задавать частоту дискретизации аудио файла.

Такая интерпретация удобнее поскольку, как было показано выше, дискретный по времени анализ в частотной области требует знания частоты дискретизации.

Для того, чтобы проверить правильность выполненной записи или правильность выбранного фрагмента записи можно использовать другую команду вида:

```
playsnd(OriginalAudio, Fs)
```

Для получения спектральных характеристик сигнала можно было, как и в предыдущих лабораторных работах, применить команду `fft()`, а затем по данным построить график в соответствии с методикой, описанной ранее в тех же работах. Однако, в Scilab существует и более простой способ при использовании команды `analyze()`. Если передать этой команде массив, который представляет оцифрованный звуковой сигнал, вместе с частотой дискретизации этого сигнала, то она создаст спектр, то есть график в частотной области с горизонтальной осью, отображающей частоты в герцах. При этом необходимо также указывать минимальную и максимальную частоты и количество выборок.

Рассмотрим пример использования команды `analyze()` для формирования и вывода спектра на экран при $F_{min} = 20\text{Гц}$ и $F_{max} = 20\text{кГц}$.

```
Fmin = 20;  
Fmax = 20e3;  
analyze(OriginalAudio, Fmin, Fmax, Fs, length(OriginalAudio))
```

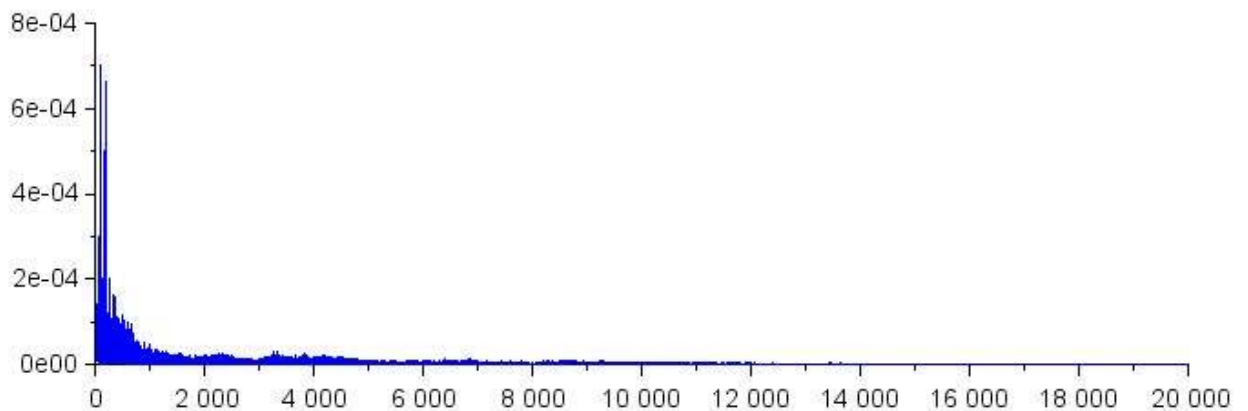


График в частотной области при использовании команды `analyze()` и $F_{\max} = 20\text{ кГц}$

Теперь снизим максимальную частоту до 2 кГц и построим спектр с помощью команд:

```
Fmax = 2e3;  
analyze(OriginalAudio, Fmin, Fmax, Fs, length(OriginalAudio))
```

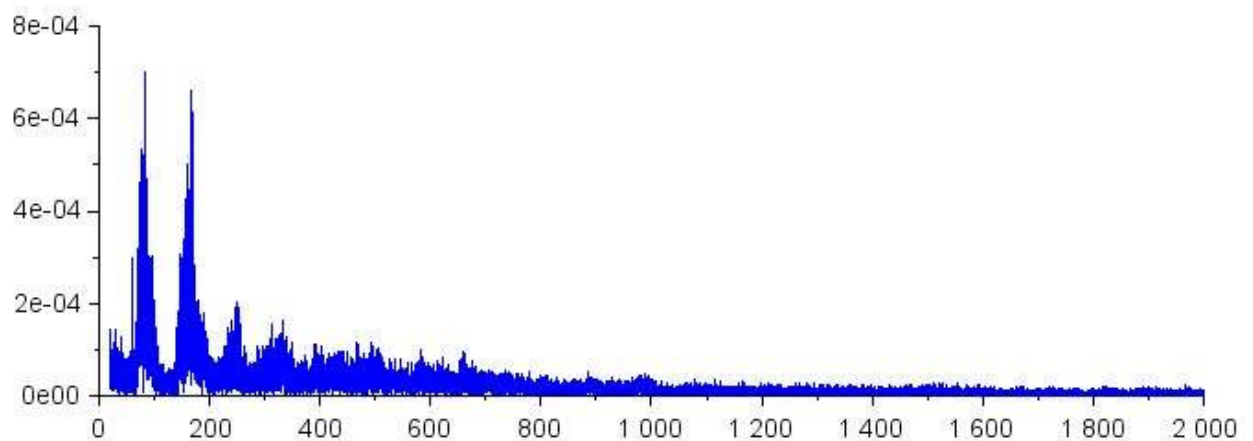


График в частотной области при использовании команды `analyze()` и $F_{\max} = 2\text{ кГц}$

Анализ фоновых звуков шипения

Для снижения фоновых звуков шипения можно использовать фильтр, который должен быть настроен на частотные составляющие спектра звуков шипения. Для определения частотного спектра звуков шипения необходимо проанализировать спектр аудио фрагмента, который содержит только фоновый шум. В выбранном файле речевое сообщение заканчивается приблизительно на девятой секунде, а запись всего файла заканчивается на двенадцатой секунде. Поэтому целесообразно выбрать данные файла с десятой по двенадцатую секунды и построить частотный спектр этого фрагмента, то есть спектр фонового шипения, а потом его проанализировать. Для построения графика в частотной области необходимо выполнить следующие команды:

```

Fmax = 20e3;
OriginalAudio_NoiseOnly = OriginalAudio(10*Fs : 12*Fs);
playsnd(OriginalAudio_NoiseOnly, Fs) // чтобы подтвердить, что фрагмент содержит
    только фоновый шум
analyze(OriginalAudio_NoiseOnly, Fmin, Fmax, Fs, length(OriginalAudio_NoiseOnly))

```

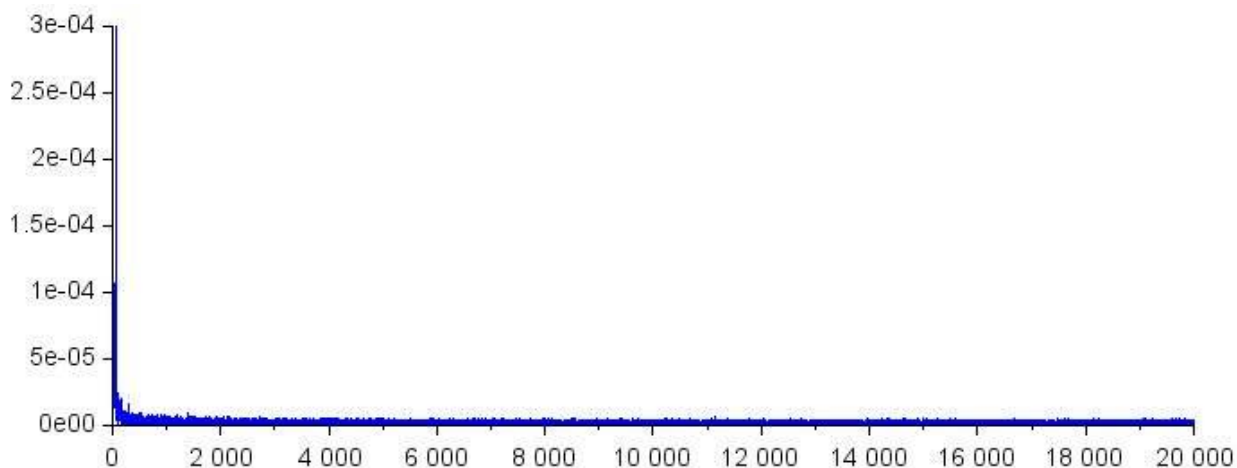


График в частотной области фрагмента аудио файла, содержащего только фоновое шипение при $F_{\max} = 20\text{кГц}$

Однако, при выбранной максимальной частоте $F_{\max} = 20\text{кГц}$ отображается весь звуковой диапазон, при этом низкочастотная область плохо различима, хотя спектры шумов часто содержат частотные составляющие в этой области. Для повышения разрешения низкочастотной части спектра следует ограничить максимальную частоту, например, значением 1кГц . Для этого можно использовать команды:

```

Fmax = 1e3;
analyze(OriginalAudio_NoiseOnly, Fmin, Fmax, Fs, length(OriginalAudio_NoiseOnly))

```

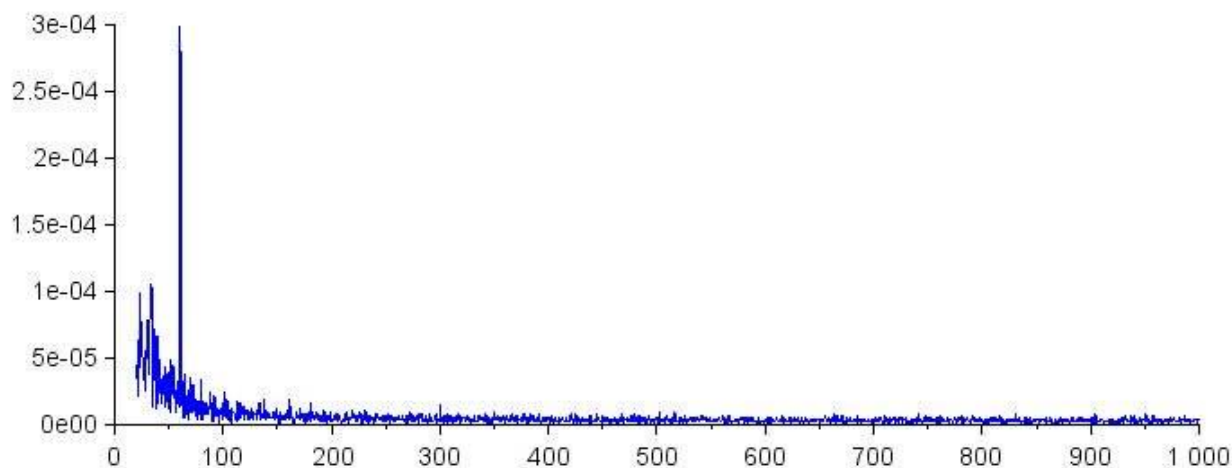


График в частотной области фрагмента аудио файла, содержащего только фоновое шипение, ограниченный $F_{\max} = 1\text{кГц}$

Графики в частотной области показывают, что фоновый шум обладает достаточно постоянной амплитудой в широкой полосе частот, при этом наибольшую амплитуду шумы имеют в пределах небольшой низкочастотной области ниже 100Гц. Кроме того, на частоте около 60 Гц имеется спектральная составляющая с максимальной амплитудой, что объясняется помехой питающего напряжения на частоте 60Гц. Поэтому можно утверждать, что команда `analyze()` формирует вполне достоверную информацию о частотах.

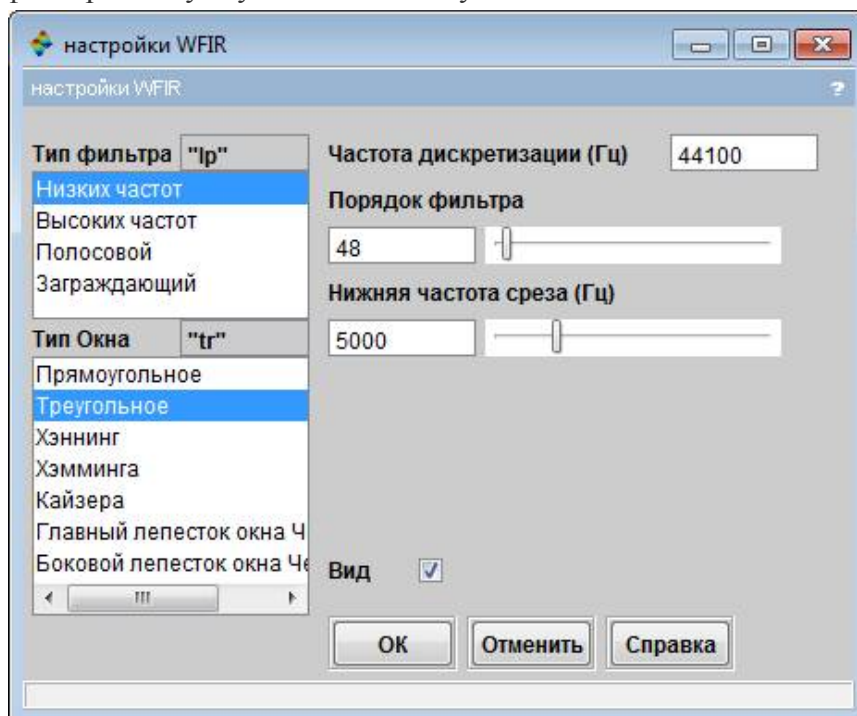
Формирование и реализация фильтра

Для решения задачи фильтрации шума выберем фильтр с конечной импульсной характеристикой, то есть КИХ фильтр, а для формирования функций фильтра и его реализации в Scilab будем использовать графический инструмент проектирования FIR фильтров в виде:

```
FIR_coefficients = wfir();
```

Следует отметить, что простые фильтры, к сожалению, не обеспечивают эффективной фильтрации. Поэтому простое удаление низкочастотной спектральной составляющей на 60Гц или низкочастотного шума с наибольшей амплитудой ниже этой составляющей по частоте вряд ли приведет к существенному очищению от шума звукового сигнала и повышению его качества, ощутимого на слух, так как исходный звуковой сигнал обладает большими по амплитуде спектральными составляющими с центром около 80 Гц.

Следовательно, для реализации фильтра и подавления низких частот фонового шума необходимо правильно выбрать частоту среза фильтра, а также тип, порядок фильтра и функцию окна. Однако, фоновое шипение формируется за счет наличия в звуке частотных составляющих в высокочастотной, а не в низкочастотной части спектра. Поэтому для удаления фоновых звуков шипения необходимо отфильтровывать высокочастотные составляющие спектра, начиная, например, с 5кГц. Поэтому в окне настройки КИХ-фильтра следует установить следующие значения:



Окно настройки КИХ-фильтра нижних частот с частотой среза 5кГц

Далее для сохранения в переменной `FIR_coefficients` всех выставленных коэффициентов, и необходимо нажать кнопку ОК. При заданных параметрах фильтра его амплитудно-частотная характеристика будет иметь следующий вид:

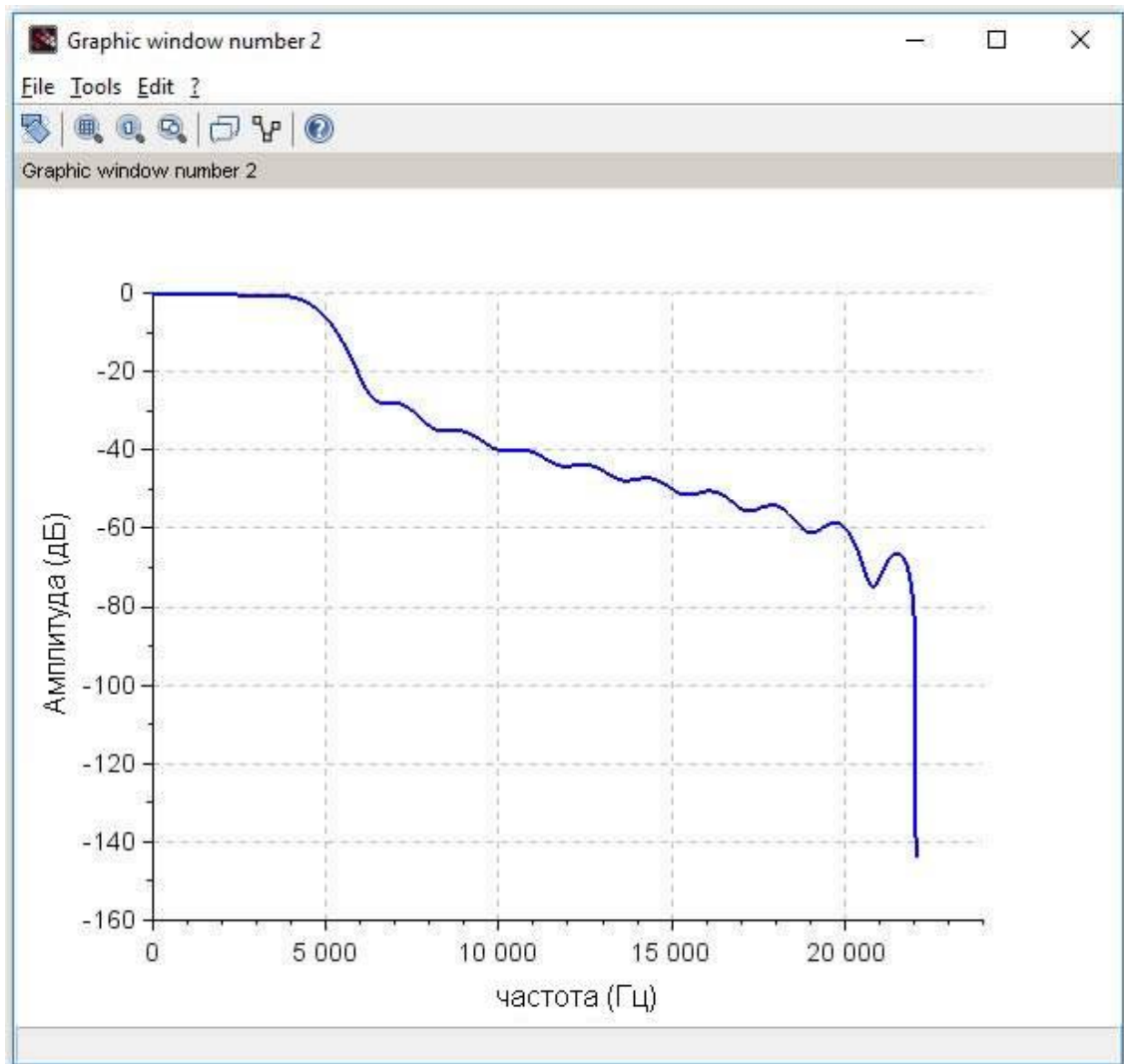


График амплитудно-частотной характеристики полученного низкочастотного КИХ-фильтра

После этого можно применить фильтр, используя свертку. Для выполнения свертки используется команда `convol()`.

```
FilteredAudio = convol(FIR_coefficients, OriginalAudio);  
analyze(FilteredAudio, Fmin, Fmax, Fs, length(FilteredAudio))
```

Спектр звукового сигнала после фильтрации высокочастотных составляющих фонового шипения при заданной частоте среза 5кГц будет выглядеть следующим образом:

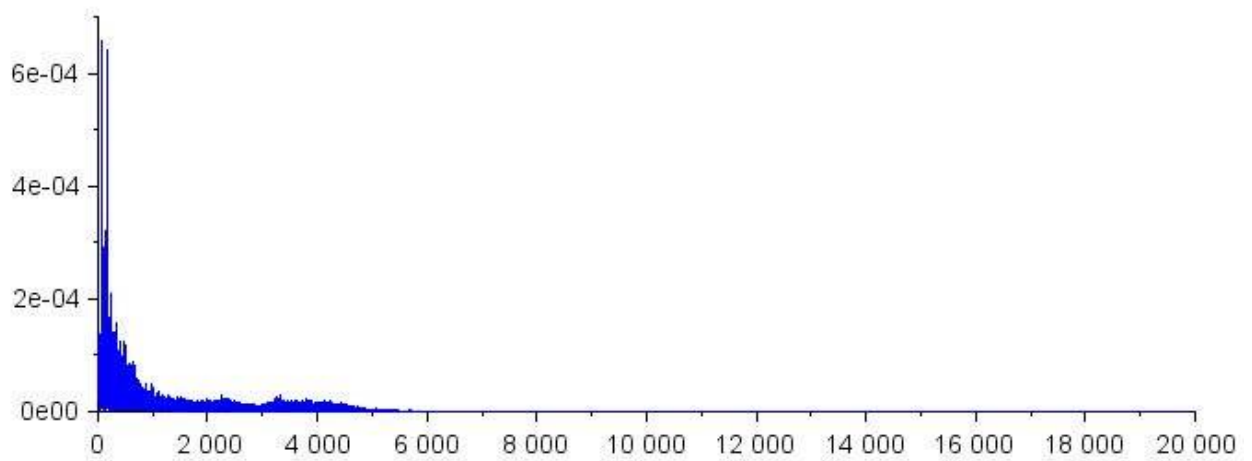


График в частотной области звукового сигнала после удаления фонового шипения

На графике высокочастотные составляющие фонового шипения отсутствуют, то есть были отфильтрованы и уровень высокочастотных шумов снижен. Поэтому звук после фильтрации изменился.



02 OnceUponaMidnightDreary__filtered_5kHz.wav

Следует отметить, что качество звука и голоса после фильтрации было изменено, голос звучит более приглушенно, но вполне разборчиво. Высокие частоты влияют на окраску звука, поэтому при их удалении голос стал менее естественным и выразительным. Для оценки и понимания степени влияния на качество звука и речи более высокой или низкой частоты среза (f_{cp}) полезно сравнить и проанализировать следующие записи с различными частотами среза фильтров, например, 7кГц и 3кГц.

$$f_{cp} = 7 \text{ кГц}$$



03 OnceUponaMidnightDreary__filtered_7kHz.wav

$$f_{cp} = 3 \text{ кГц}$$



04 OnceUponaMidnightDreary__filtered_3kHz.wav

Задание на лабораторную работу 5

1. Изучить описание лабораторной работы Применение фильтров для удаления шума в аудиозаписи с помощью Scilab
2. Получить у преподавателя значение частоты среза КИХ-фильтра для удаления высокочастотных шумов с помощью Scilab. В окне настройки КИХ-фильтра установить заданную нижнюю частоту среза для фильтра низких частот. Выполнить фильтрацию высокочастотных шумов звукового файла треугольным окном. Построить амплитудно-

частотную характеристику низкочастотного КИХ-фильтра и спектр звукового сигнала после фильтрации в соответствии с рекомендациями описания лабораторной работы. Прослушать звуковой файл до и после фильтрации, охарактеризовать и описать изменения звука.

3. Получить у преподавателя значение частоты среза КИХ-фильтра для удаления низкочастотных шумов с помощью Scilab. В окне настройки КИХ-фильтра установить заданную верхнюю частоту среза для фильтра высоких частот. Выполнить фильтрацию низкочастотных шумов звукового файла треугольным окном. Построить амплитудно-частотную характеристику высокочастотного КИХ-фильтра и спектр звукового сигнала после фильтрации. Прослушать звуковой файл до и после фильтрации, охарактеризовать и описать изменения звука.
4. Выполнить пункты 2 и 3 задания этой лабораторной работы, используя для фильтрации окно Хэмминга. Построить амплитудно-частотные характеристики КИХ-фильтров и спектры звуковых сигналов после фильтрации. Прослушать звуковые файлы до и после фильтрации, охарактеризовать и описать изменения звука.
5. Сравнить амплитудно-частотные характеристики и спектры сигналов, полученные для фильтров с треугольным окном и окном Хемминга, а также охарактеризовать и описать изменения звука по результатам прослушивания.
6. Сделать выводы по результатам работы.

Лабораторная работа 6. Реализация спектрального вычитания для подавления шума в аудиозаписи с помощью Scilab

Предыдущая лабораторная работа была посвящена изучению возможности удаления шумов с помощью фильтров. В процессе выполнения лабораторной работы было показано, что использование фильтров для подавления шума не всегда эффективно, поскольку фильтры не позволяют подавлять частоты шумов, близкие к частотам информационного сигнала, что проявляется в более значительной степени при отсутствии возможности в системе реализации фильтров высоких порядков. Другой недостаток применения фильтров при подавлении шумов состоит в отсутствии избирательности частотных составляющих шума в пределах фильтруемой полосы частот. Поэтому фильтрация часто приводит к искажениям информационного сигнала или голоса при фильтрации речевого сигнала.

Поэтому для удаления шумов часто применяют другие методы, основанные на спектральном вычитании, которое может удалять узкополосные частотные составляющие спектра шума, и может быть реализовано в следующей последовательности.

1. Создать или сформировать аудиозапись, которая содержит только фоновый шум, например, можно выделить из сделанной ранее записи отдельный фрагмент, который содержит только шум.
2. Выполнить преобразование сигнала из временной в частотную область, получив спектр шумового сигнала, вычисляя FFT и записывая данные об амплитудах частотных составляющих спектра шума.
3. При необходимости, для повышения эффективности спектра шума, можно выполнить дополнительную предварительную обработку сигнала шума, например, усреднение и предварительную его графическую интерпретацию, отрисовку.

4. Выполнить процедуру вычитания амплитуд спектральных составляющих шума из амплитуд результатов преобразования FFT исходного сигнала аудиозаписи, создав спектр аудиофайла сигнала с уменьшенным уровнем шума.
5. Добавить информацию о фазе в спектр сигнала с уменьшенным уровнем шума дублированием информации о фазе из результатов полученного преобразования FFT исходной записи.
6. С помощью обратного быстрого преобразования Фурье сформировать сигнал с уменьшенным шумом во временной области.

Все перечисленные операции целесообразно выполнять в выбранной системе Scilab, поскольку она, помимо реализации фильтров, позволяет удалять узкополосные частотные составляющие спектра шума более эффективно, чем рассмотренное в предыдущей лабораторной работе применение фильтров.

Формирование спектра шума

Для формирования спектра шума полезно использовать осреднение шумового сигнала, для этого необходимо записать фоновый сигнал в паузе достаточной для осреднения длительности, превосходящей длительность информационного речевого сигнала, составляющего 10 секунд, например, длительностью для осреднения около 60 секунд. Записанный фоновый сигнал в паузе 60 секунд можно разделить на 6 фрагментов, длительностью по 10 секунд каждый. Далее можно вычислить спектр для каждого фрагмента и усреднить их. Понятно, что фоновый сигнал шума имеет достаточно равномерный характер и при таких его свойствах осреднение может быть полезно. Команды программы формирования из записанного аудиофайла фонового сигнала шума и его осреднения можно представить в виде.

```
[NoiseAudio, Fs] = wavread("C:\Users\Vladimir\Documents\Audio\Noise.wav");
NoiseAudio_1 = NoiseAudio(1:Fs*10);
NoiseAudio_2 = NoiseAudio(1+Fs*10:Fs*20);
NoiseAudio_3 = NoiseAudio(1+Fs*20:Fs*30);
...
NoiseAudio_6 = NoiseAudio(1+Fs*50:Fs*60);

NoiseAudio_1_FFTMag = abs(fft(NoiseAudio_1));
NoiseAudio_2_FFTMag = abs(fft(NoiseAudio_2));
...
NoiseAudio_6_FFTMag = abs(fft(NoiseAudio_6));

for k = 1:length(NoiseAudio_1_FFTMag)
    > Noise_FFTMag(1,k) = (NoiseAudio_1_FFTMag(k) + NoiseAudio_2_FFTMag(k) +
NoiseAudio_3_FFTMag(k) + NoiseAudio_4_FFTMag(k) + NoiseAudio_5_FFTMag(k) +
NoiseAudio_6_FFTMag(k))/6;
```

> end

Усреднение шумового спектра позволяет повысить качество исходного сигнала аудиозаписи, однако это улучшение не является значительным и не сильно проявляется при прослушивании.

Для получения графика в частотной области общей формы спектра шума можно не преобразовывать горизонтальную ось в частоты, выраженные в Гц, то есть в начале для оценки общего вида спектра сигнала шума можно представить этот спектр как обычный массив точек данных. Поэтому график сигнала шума в частотной области в относительных единицах может содержать зеркальную составляющую, как показано на рисунке.

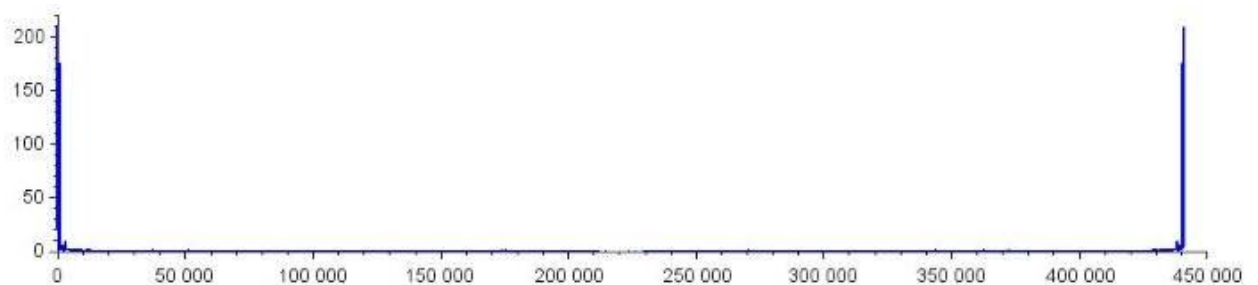


График сигнала шума в частотной области в относительных единицах после преобразования FFT

После удаления зеркальной составляющей при представлении первой половины спектра сигнала шума график будет иметь вид.

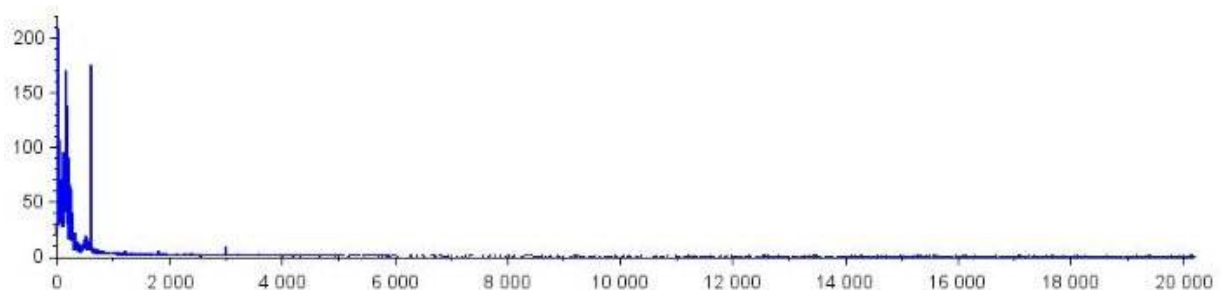


График сигнала шума в частотной области в относительных единицах после удаления зеркальной составляющей

Однако, полученный спектр из-за слишком широкого диапазона представления сигнала в частотной области не является информативным. Поэтому целесообразно сократить частотный диапазон представления сигнала шума, сохранив более информативную низкочастотную область, что позволит детализировать частотные составляющие спектра шумового сигнала, как показано на рисунке.

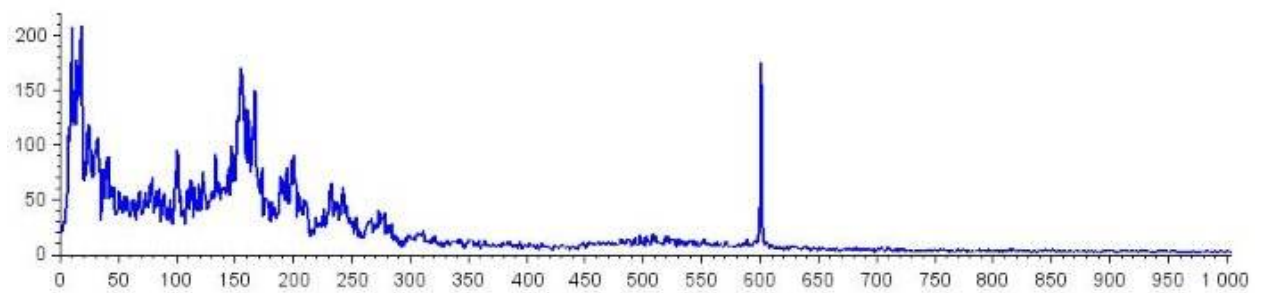


График сигнала шума в частотной области после сокращения диапазона частот.

Реализация спектрального вычитания

Для упрощения процедуры спектрального вычитания можно использовать два допущения. Так можно отказаться от обработки комплексных чисел, которые являются результатом преобразования FFT, и ограничиться только амплитудами, полученными после преобразования FFT. Кроме того, в спектрах сигнала шума и полученном после вычитания не должно быть отрицательных значений амплитуд, то есть после проверки результатов на отрицательность нужно заменить все отрицательные значения на ноль. Необходимо также выровнять массивы с результатами преобразования FFT исходной аудиозаписи и массив, содержащий спектр сигнала шума, то есть эти два массива должны быть одинаковой длины в 10 секунд. Команды соответствующей программы можно представить в следующем виде.

```
[OriginalAudio, Fs] =
wavread("C:\Users\Vladimir\Documents\Audio\OnceUponaMidnightDreary.wav");

OriginalAudio = OriginalAudio(1:Fs*10); // Укороченная запись точно до 10 секунд.

// Массив, содержащий результаты FFT исходной записи,
// и массив, содержащий шумовой спектр,
// должны быть одинаковой длины.

OriginalAudio_FFTMag = abs(fft(OriginalAudio));
for k = 1:length(OriginalAudio_FFTMag)
    > FFTMag_NoiseSubtracted(1,k) = OriginalAudio_FFTMag(k) - Noise_FFTMag(k);
    > if(FFTMag_NoiseSubtracted(1,k)) < 0
    > FFTMag_NoiseSubtracted(1,k) = 0;
    > end
    > end
```

Формирование фазы путем восстановления

Для преобразования полученного спектра амплитуд исходной записи звукового сигнала с уменьшенным уровнем шума после вычитания во временную область необходимо восстановить фазу. Понятно, что искусственно полученный спектр амплитуд при

выполнении вычитания в частотной области не содержит информации о фазе. Фаза содержится только в исходном звуковом сигнале с шумом. Таким образом, после изменения амплитуды спектра Фурье преобразования в результате спектрального вычитания необходимо сохранить фазу исходного сигнала у результирующего спектра Фурье. Следует отметить, что поскольку равномерный шум в большей степени связан с амплитудой, а не с фазой, то сохранение фазы исходного сигнала в результирующем спектре Фурье не должно вызвать проблемы. Известно, что результаты преобразования FFT исходного звукового сигнала состоят из комплексных чисел, также и результаты FFT с уменьшенным шумом после спектрального вычитания должны быть комплексными числами для того, чтобы они могли быть без затруднений преобразованы обратно во временную область с помощью команды `ifft()`. Следовательно, необходимо получить фазу из действительной и мнимой составляющих спектра исходного звукового сигнала, и далее сформировать новый массив комплексных чисел со значениями амплитуд полученного после вычитания нового спектра. Иллюстрировать связь между амплитудой и фазой действительной и мнимой частями комплексного числа можно с помощью рисунка.

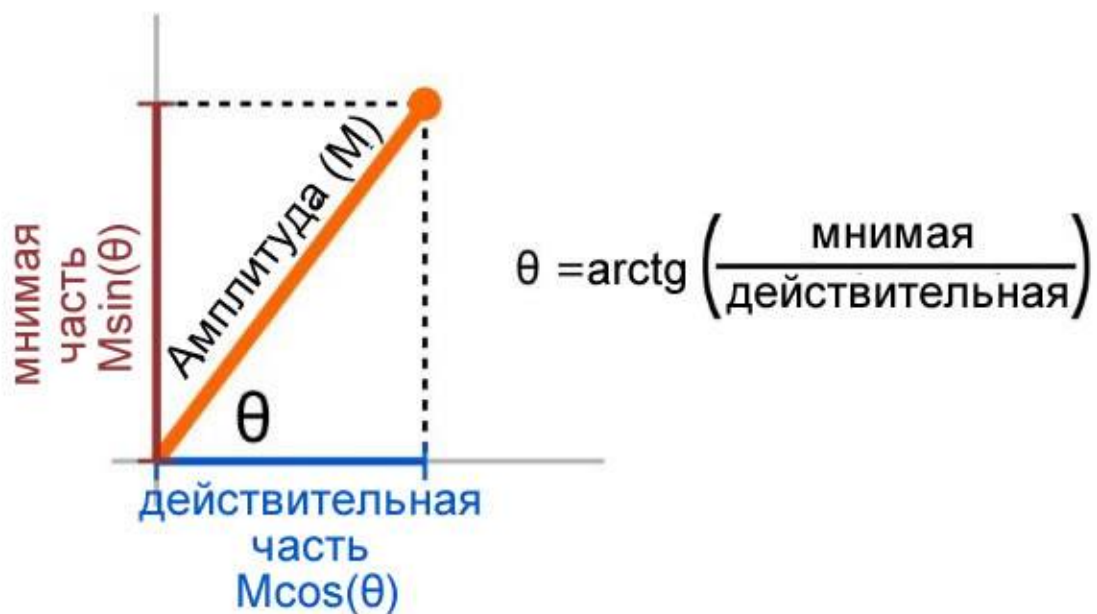


Рис. Иллюстрация связи амплитуды и фазы действительной и мнимой частей комплексного числа.

Команды программы формирования комплексных чисел преобразования FFT с уменьшенным шумом после спектрального вычитания можно представить в виде.

```
OriginalAudio_FFT = fft(OriginalAudio);
for k = 1:length(OriginalAudio_FFTMag)
    > OriginalPhase = atan(imag(OriginalAudio_FFT(k)), real(OriginalAudio_FFT(k)));
    > RealPart = FFTMag_NoiseSubtracted(k) * cos(OriginalPhase);
    > ImagPart = FFTMag_NoiseSubtracted(k) * sin(OriginalPhase);
    > FFT_NoiseSubtracted(1,k) = complex(RealPart, ImagPart);
```

> end

С помощью следующих графиков можно проиллюстрировать избирательные возможности удаления частотных составляющих шума при реализации метода спектрального вычитания. Из-за представления спектров в относительных единицах без перехода к реальным частотам, выраженным в Гц, на первом графике на оси частот при значении 600, соответствующем частоте 60Гц наблюдается всплеск, то есть имеется частотная составляющая шума на этой частоте. На втором графике после спектрального вычитания эта частотная составляющая отсутствует, то есть она избирательно удалена спектральным вычитанием без изменения и искажения соседних частотных составляющих.

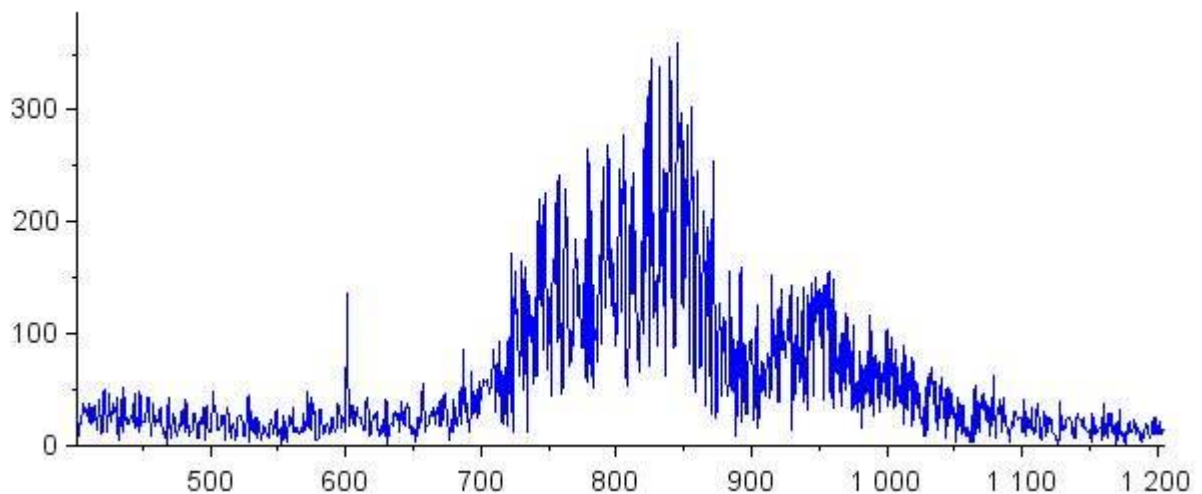


Рис. График в частотной области звукового сигнала с частотной составляющей шума на значении 600, соответствующей частоте 60Гц.

Результат спектрального вычитания имеется частотной составляющей шума на частоте 60Гц можно представить на следующем рисунке.

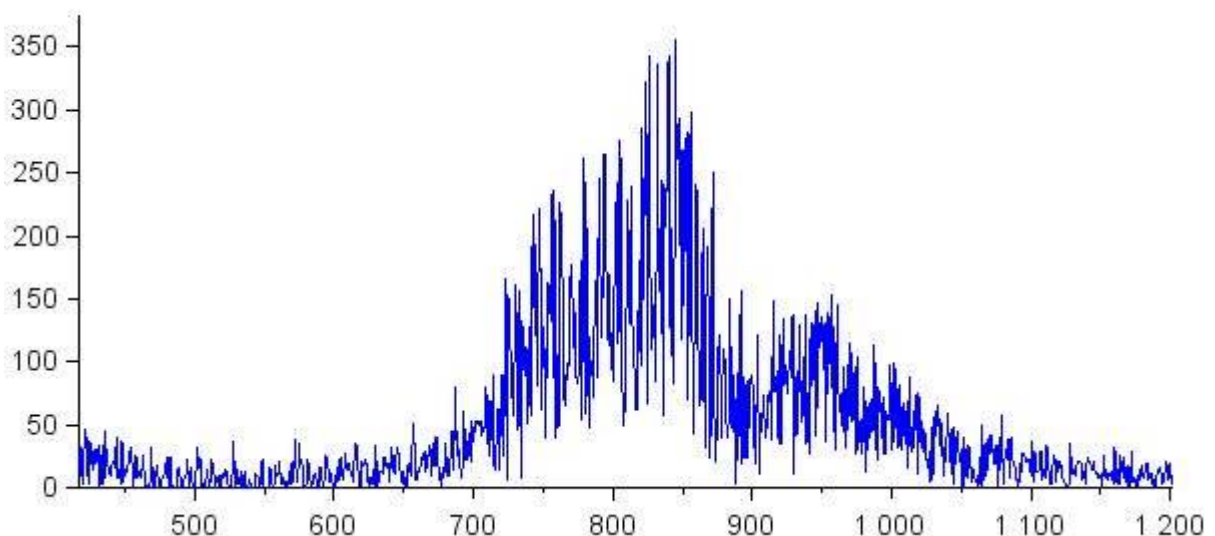


Рис. График в частотной области звукового сигнала после спектрального вычитания

Преобразование частотного спектра сигнала после спектрального вычитания во временную область

Полученный после спектрального вычитания очищенный частотный спектр звукового сигнала можно преобразовать во временную область с помощью обратного быстрого преобразования Фурье командой `ifft` в виде:

```
Audio_NoiseSubtracted = ifft(FFT_NoiseSubtracted);
```

Ниже представлены исходный звуковой сигнал и очищенный сигнал, полученный после спектрального вычитания, которые можно сравнить между собой.

Исходный звуковой сигнал



01 OnceUponaMidnightDreary.wav

Очищенный звуковой сигнал после спектрального вычитания



02 OnceUponaMidnightDreary__SpectrumSubtraction.wav

Следует отметить, что спектральное вычитание не привело к ощутимому на слух значительному удалению шума, хотя изменило его качество, немного изменив тембр шума. При этом голос диктора до и после спектрального вычитания звучит одинаково, что является значительным улучшением по сравнению с результатами применения фильтров.

Задание на лабораторную работу 6

1. Изучить описание лабораторной работы Реализация спектрального вычитания для подавления шума в аудиозаписи с помощью Scilab
 2. Использовать полученные в предыдущей лабораторной работе результаты фильтрации КИХ-фильтром с треугольным окном высокочастотных шумов для заданного преподавателем значения частоты среза в качестве спектра сигнала шума для спектрального вычитания с помощью Scilab или сформировать спектр сигнала шума из фрагментов сигналов в паузах исходного звукового сигнала.
 3. Выполнить спектральное вычитание из спектра исходного звукового сигнала спектра, выбранного в качестве сигнала шума, полученного после фильтрации треугольным окном высокочастотных шумов.
 4. При необходимости восстановить фазу спектра сигнала после спектрального вычитания и с помощью обратного преобразования Фурье командой `ifft` преобразовать частотный спектр сигнала во временную область в соответствии с рекомендациями описания лабораторной работы.
- Прослушать звуковой файл до и после фильтрации, охарактеризовать и описать изменения звука.

5. Использовать полученные в предыдущей лабораторной работе результаты фильтрации КИХ-фильтром с треугольным окном низкочастотных шумов для заданного преподавателем значения частоты среза в качестве спектра сигнала шума для спектрального вычитания с помощью Scilab. Выполнить пункты 3 и 4 задания этой лабораторной работы для полученного сигнала после фильтрации треугольным окном низкочастотных шумов.
6. Выполнить пункты 2 - 5 задания этой лабораторной работы, используя для фильтрации окно Хэмминга.
7. Сравнить спектры сигналов и сигналы во временной области, полученные после спектрального вычитания и обратного преобразования Фурье для фильтров с треугольным окном и окном Хемминга, а также охарактеризовать и описать изменения звука по результатам прослушивания.
8. Сделать выводы по результатам работы.