

Сетевые технологии

Лабораторная работа №1

Тойчубекова Асель Нурлановна

2025-09-13

Содержание I

1. Информация

2. Выполнение лабораторной работы

Раздел 1

1. Информация

1.1 Докладчик

► Тойчубекова Асель Нурлановна

1.1 Докладчик

- ▶ Тойчубекова Асель Нурлановна
- ▶ Студент 3 курса

1.1 Докладчик

- ▶ Тойчубекова Асель Нурлановна
- ▶ Студент 3 курса
- ▶ факультет физико-математических и естественных наук

1.1 Докладчик

- ▶ Тойчубекова Асель Нурлановна
- ▶ Студент 3 курса
- ▶ факультет физико-математических и естественных наук
- ▶ Российский университет дружбы народов им. П. Лумумбы

1.1 Докладчик

- ▶ Тойчубекова Асель Нурлановна
- ▶ Студент 3 курса
- ▶ факультет физико-математических и естественных наук
- ▶ Российский университет дружбы народов им. П. Лумумбы
- ▶ 1032235033@rudn.ru

1.2 Цель работы

Целью данной лабораторной работы является изучение методов кодирования и модуляции сигналов с помощью высокоуровневого языка программирования Octave. Определение спектра и параметров сигнала. Демонстрация принципов модуляции сигнала на примере аналоговой амплитудной модуляции. Исследование свойства самосинхронизации сигнала.

1.3 Задание

1. Построить график функции $f = \sin x + (1/3)\sin 3x + (1/5)\sin 5x$ на интервале $[-10; 10]$, используя Octave и функцию `plot`. График экспортировать в файлы формата `.eps`, `.png`.

1.3 Задание

1. Построить график функции $\varphi = \sin x + (1/3)\sin 3x + (1/5)\sin 5x$ на интервале $[-10; 10]$, используя Octave и функцию `plot`. График экспортировать в файлы формата `.eps`, `.png`.
2. Добавить график функции $\varphi = \cos x + (1/3)\cos 3x + (1/5)\cos 5x$ на интервале $[-10; 10]$. График экспортировать в файлы формата `.eps`, `.png`

1.3 Задание

1. Построить график функции $\varphi = \sin x + (1/3)\sin 3x + (1/5)\sin 5x$ на интервале $[-10; 10]$, используя Octave и функцию `plot`. График экспортировать в файлы формата `.eps`, `.png`.
2. Добавить график функции $\varphi = \cos x + (1/3)\cos 3x + (1/5)\cos 5x$ на интервале $[-10; 10]$. График экспортировать в файлы формата `.eps`, `.png`.
3. Разработать код `m`-файла, результатом выполнения которого являются графики меандра, реализованные с различным количеством гармоник.

1.3 Задание

1. Построить график функции $\varphi = \sin x + (1/3)\sin 3x + (1/5)\sin 5x$ на интервале $[-10; 10]$, используя Octave и функцию `plot`. График экспортировать в файлы формата `.eps`, `.png`.
2. Добавить график функции $\varphi = \cos x + (1/3)\cos 3x + (1/5)\cos 5x$ на интервале $[-10; 10]$. График экспортировать в файлы формата `.eps`, `.png`.
3. Разработать код `m`-файла, результатом выполнения которого являются графики меандра, реализованные с различным количеством гармоник.
4. Определить спектр двух отдельных сигналов и их суммы.

1.4 Задание

5. Выполнить задание с другой частотой дискретизации. Пояснить, что будет, если взять частоту дискретизации меньше 80 Гц?

1.4 Задание

5. Выполнить задание с другой частотой дискретизации. Пояснить, что будет, если взять частоту дискретизации меньше 80 Гц?
6. Продемонстрировать принципы модуляции сигнала на примере аналоговой амплитудной модуляции.

1.4 Задание

5. Выполнить задание с другой частотой дискретизации. Пояснить, что будет, если взять частоту дискретизации меньше 80 Гц?
6. Продемонстрировать принципы модуляции сигнала на примере аналоговой амплитудной модуляции.
7. По заданных битовых последовательностей требуется получить кодированные сигналы для нескольких кодов, проверить свойства самосинхронизуемости кодов, получить спектры.

1.5 Теоретическое введение

Сигнал — это физическая величина, изменяющаяся во времени и несущая информацию. Он может быть аналоговым или дискретным. Для перевода в цифровую форму используется дискретизация и квантование, выполняемые аналого-цифровым преобразователем. Согласно теореме Котельникова, частота дискретизации должна быть более чем в два раза выше максимальной частоты сигнала.

1.6 Теоретическое введение

Для анализа применяют спектральное разложение, основанное на ряде и преобразовании Фурье, что позволяет выделить амплитуды и фазы гармонических составляющих. В цифровой обработке используются дискретное и быстрое преобразования Фурье.

1.7 Теоретическое введение

При передаче данных применяется модуляция — изменение амплитуды, частоты или фазы несущего колебания. Для представления двоичной информации используются различные способы кодирования сигналов: NRZ, AMI, RZ, а также коды Манчестер и дифференциальный Манчестер.

1.8 Теоретическое введение

Для моделирования и обработки сигналов в учебных задачах широко используется язык Octave. Он поддерживает работу с матрицами, содержит набор встроенных математических функций и средства визуализации, что делает его удобным инструментом для анализа сигналов.

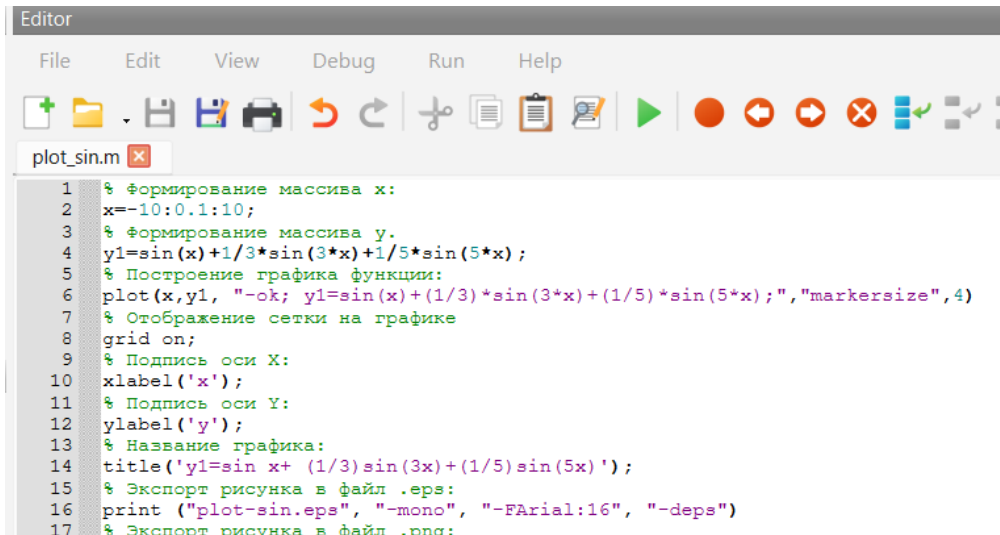
Раздел 2

2. Выполнение лабораторной работы

2.1 Выполнение лабораторной работы

Запускаю в вашей ОС Octave с оконным интерфейсом. Перехожу в окно редактора. Воспользовавшись меню или комбинацией клавиш `ctrl + n` создаю новый сценарий. Сохраняю его в рабочий каталог с именем, `plot_sin.m`. В окне редактора повторяю листинг по построению графика функции $y = \sin x + (1/3)\sin 3x + (1/5)\sin 5x$ на интервале $[-10; 10]$

2.2 Выполнение лабораторной работы



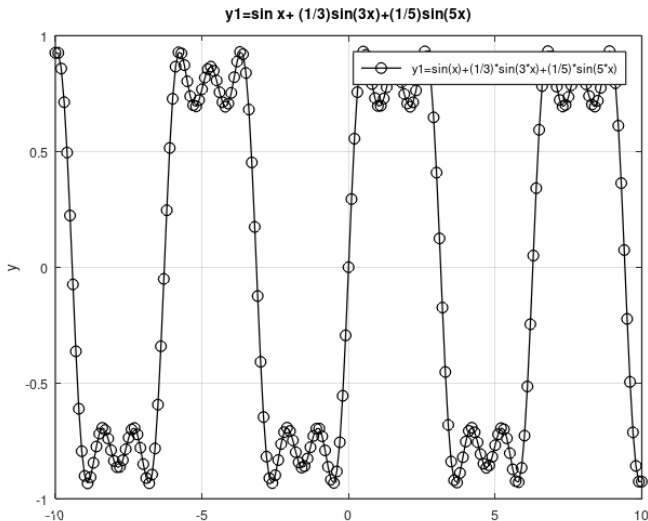
The screenshot shows the MATLAB Editor interface. The menu bar includes File, Edit, View, Debug, Run, and Help. The toolbar contains icons for file operations (new, open, save, print), editing (undo, redo, cut, copy, paste), and execution (run, step back, step forward, stop). The active file is plot_sin.m. The code in the editor is as follows:

```
1 % формирование массива x:
2 x=-10:0.1:10;
3 % формирование массива y.
4 y1=sin(x)+1/3*sin(3*x)+1/5*sin(5*x);
5 % Построение графика функции:
6 plot(x,y1, "-ok; y1=sin(x)+(1/3)*sin(3*x)+(1/5)*sin(5*x);", "markersize",4)
7 % Отображение сетки на графике
8 grid on;
9 % Подпись оси X:
10 xlabel('x');
11 % Подпись оси Y:
12 ylabel('y');
13 % Название графика:
14 title('y1=sin x+ (1/3) sin(3x)+(1/5) sin(5x) ');
15 % Экспорт рисунка в файл .eps:
16 print ("plot-sin.eps", "-mono", "-FArial:16", "-deps")
17 % Экспорт рисунка в файл .png:
```

2.3 Выполнение лабораторной работы

Запускаю сценарий на выполнение. В качестве результата выполнения кода открылось окно с построенным графиком и в рабочем каталоге появились файлы с графиками в форматах .eps, .png.

2.4 Выполнение лабораторной работы



laba2.exe

NTUSER.DAT

plot_sin.m

plot-sin.eps

plot-sin.png

Value.h

2.5 Выполнение лабораторной работы

Сохраняю сценарий под другим названием и изменяю его так, чтобы на одном графике располагались отличающиеся по типу линий графики функций $y = \cos x + (1/3)\cos 3x + (1/5)\cos 5x$ на интервале $[-10; 10]$.

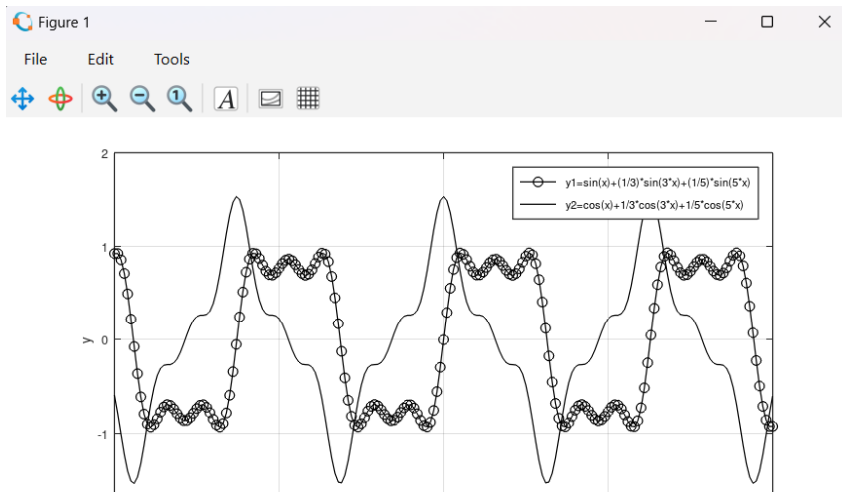
2.6 Выполнение лабораторной работы

plot_sin.m

```
1 % Формирование массива x:
2 x=-10:0.1:10;
3 % Формирование массива y.
4 y1=sin(x)+1/3*sin(3*x)+1/5*sin(5*x);
5 y2=cos(x)+1/3*cos(3*x)+1/5*cos(5*x);
6 % Построение графика функции:
7 plot(x,y1, "-ok; y1=sin(x)+(1/3)*sin(3*x)+(1/5)*sin(5*x);", "markersize", 4);
8 hold on
9 plot(x,y2, "-k; y2=cos(x)+1/3*cos(3*x)+1/5*cos(5*x);", "markersize", 4);
10 % Отображение сетки на графике
11 grid on;
12 % Подпись оси X:
13 xlabel('x');
14 % Подпись оси Y:
15 ylabel('y');
16 % Название графика:
17 title('y1=sin x+ (1/3)sin(3x)+(1/5)sin(5x) ', 'y2=cos(x)+1/3*cos(3*x)+1/5*cos(5*x)');
18 % Экспорт рисунка в файл .eps:
19 print ("plot-sin.eps", "-mono", "-FArial:16", "-deps")
20 % Экспорт рисунка в файл .png:
21 print("plot-sin.png");
```

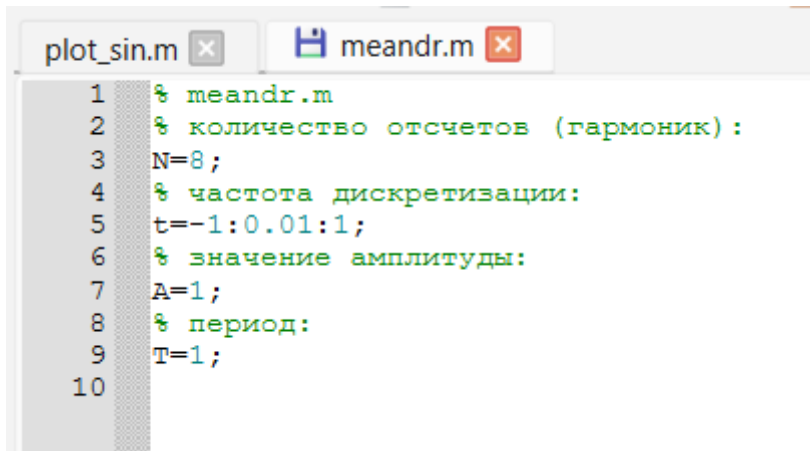
2.7 Выполнение лабораторной работы

Итоговое изображение



2.8 Выполнение лабораторной работы

Создаю новый сценарий и сохраните его в рабочий каталог с именем, meandr.m. В коде созданного сценария задаю начальные значения

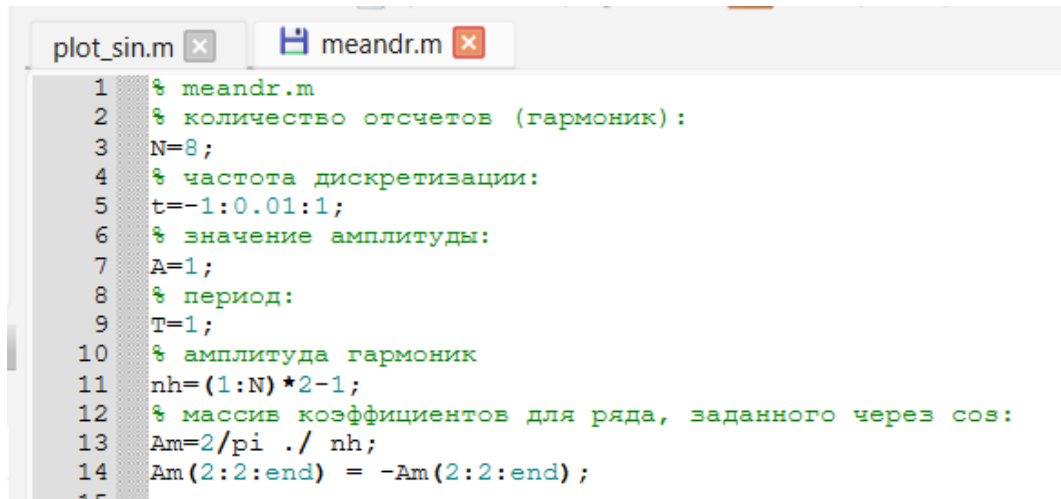


```
plot_sin.m x meandr.m x
1 % meandr.m
2 % количество отсчетов (гармоник) :
3 N=8;
4 % частота дискретизации:
5 t=-1:0.01:1;
6 % значение амплитуды:
7 A=1;
8 % период:
9 T=1;
10
```

2.9 Выполнение лабораторной работы

Разложение импульсного сигнала в форме меандра в частичный ряд Фурье можно задать формулой. Гармоники, образующие меандр, имеют амплитуду, обратно пропорциональную номеру соответствующей гармоники в спектре.

2.10 Выполнение лабораторной работы

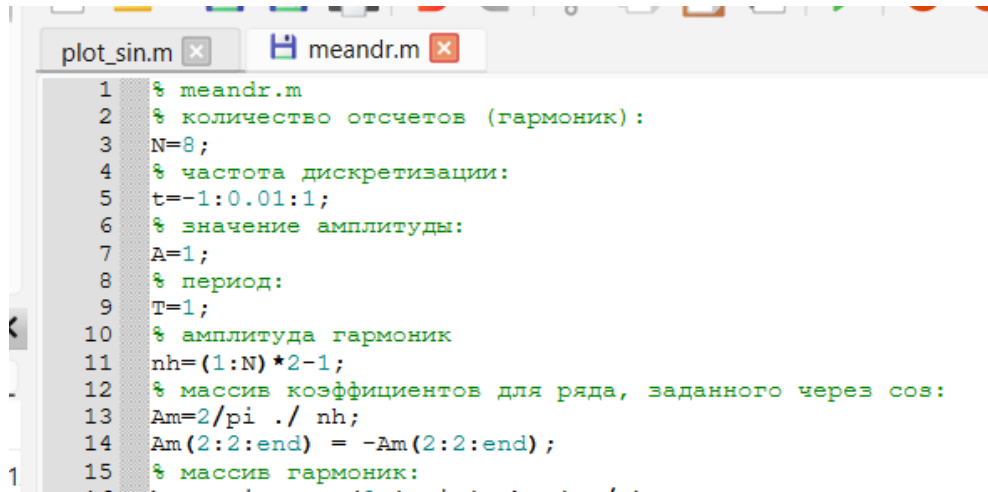


The screenshot shows a MATLAB script editor with two tabs: 'plot_sin.m' and 'meandr.m'. The 'meandr.m' tab is active, displaying the following code:

```
1 % meandr.m
2 % количество отсчетов (гармоник) :
3 N=8;
4 % частота дискретизации:
5 t=-1:0.01:1;
6 % значение амплитуды:
7 A=1;
8 % период:
9 T=1;
10 % амплитуда гармоник
11 nh=(1:N)*2-1;
12 % массив коэффициентов для ряда, заданного через cos:
13 Am=2/pi ./ nh;
14 Am(2:2:end) = -Am(2:2:end);
15
```

2.11 Выполнение лабораторной работы

Далее задаю массив значений гармоник массив элементов ряда.

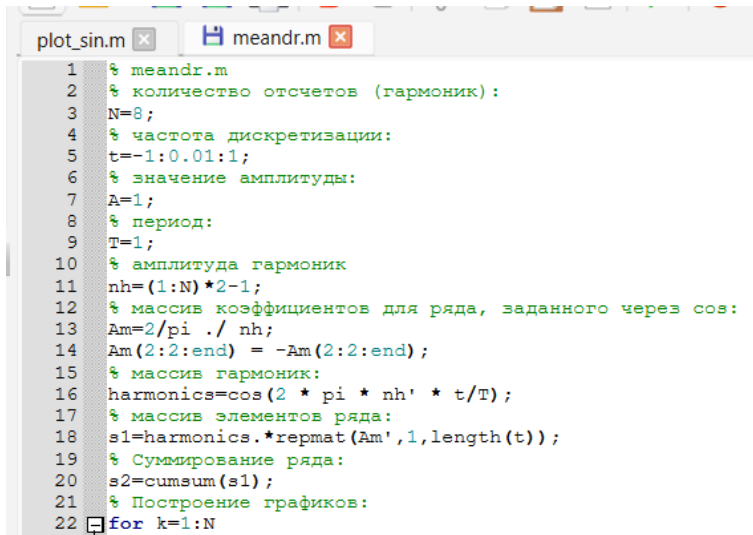


```
plot_sin.m x meandr.m x
1 % meandr.m
2 % количество отсчетов (гармоник):
3 N=8;
4 % частота дискретизации:
5 t=-1:0.01:1;
6 % значение амплитуды:
7 A=1;
8 % период:
9 T=1;
10 % амплитуда гармоник
11 nh=(1:N)*2-1;
12 % массив коэффициентов для ряда, заданного через соз:
13 Am=2/pi ./ nh;
14 Am(2:2:end) = -Am(2:2:end);
15 % массив гармоник:
```


2.12 Выполнение лабораторной работы

Далее для построения в одном окне отдельных графиков меандра с различным количеством гармоник реализую суммирование ряда с накоплением и воспользуюсь функциями `subplot` и `plot` для построения графиков.

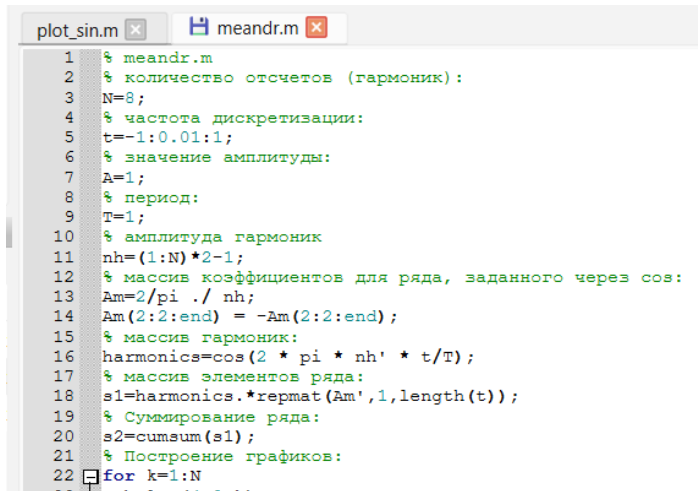
2.13 Выполнение лабораторной работы



```
plot_sin.m x meandr.m x
1 % meandr.m
2 % количество отсчетов (гармоник):
3 N=8;
4 % частота дискретизации:
5 t=-1:0.01:1;
6 % значение амплитуды:
7 A=1;
8 % период:
9 T=1;
10 % амплитуда гармоник
11 nh=(1:N)*2-1;
12 % массив коэффициентов для ряда, заданного через cos:
13 Am=2/pi ./ nh;
14 Am(2:2:end) = -Am(2:2:end);
15 % массив гармоник:
16 harmonics=cos(2 * pi * nh' * t/T);
17 % массив элементов ряда:
18 s1=harmonics.*repmat(Am',1,length(t));
19 % Суммирование ряда:
20 s2=cumsum(s1);
21 % Построение графиков:
22 for k=1:N
```

2.14 Выполнение лабораторной работы

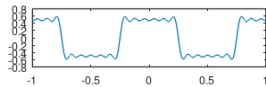
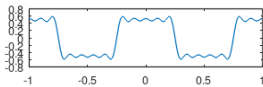
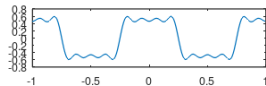
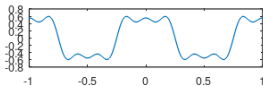
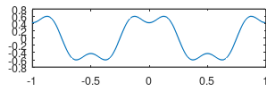
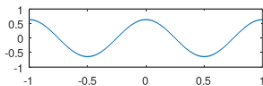
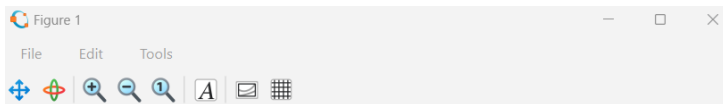
Экспортирую полученный график в файл в формате .png.



```
plot_sin.m x meandr.m x
1 % meandr.m
2 % количество отсчетов (гармоник):
3 N=8;
4 % частота дискретизации:
5 t=-1:0.01:1;
6 % значение амплитуды:
7 A=1;
8 % период:
9 T=1;
10 % амплитуда гармоник
11 nh=(1:N)*2-1;
12 % массив коэффициентов для ряда, заданного через cos:
13 Am=2/pi ./ nh;
14 Am(2:2:end) = -Am(2:2:end);
15 % массив гармоник:
16 harmonics=cos(2 * pi * nh' * t/T);
17 % массив элементов ряда:
18 s1=harmonics.*repmat(Am',1,length(t));
19 % Суммирование ряда:
20 s2=cumsum(s1);
21 % Построение графиков:
22 for k=1:N
```

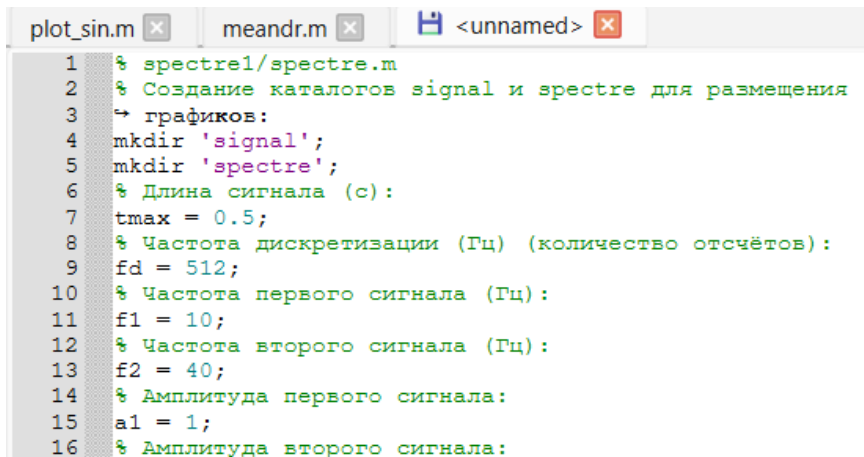
2.15 Выполнение лабораторной работы

Итоговый график.



2.16 Выполнение лабораторной работы

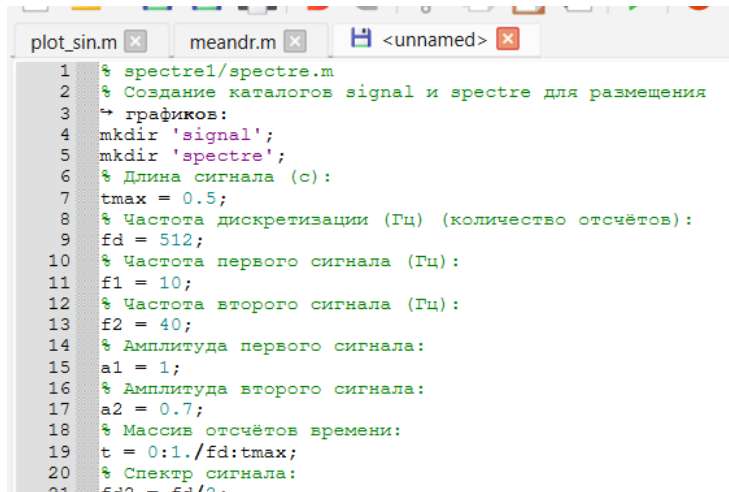
В рабочем каталоге создаю каталог `spectre1` и в нём новый сценарий с именем, `spectre.m`. В коде созданного сценария задаю начальные значения.



```
plot_sin.m x meandr.m x <unnamed> x
1 % spectre1/spectre.m
2 % Создание каталогов signal и spectre для размещения
3 % графиков:
4 mkdir 'signal';
5 mkdir 'spectre';
6 % Длина сигнала (с):
7 tmax = 0.5;
8 % Частота дискретизации (Гц) (количество отсчётов):
9 fd = 512;
10 % Частота первого сигнала (Гц):
11 f1 = 10;
12 % Частота второго сигнала (Гц):
13 f2 = 40;
14 % Амплитуда первого сигнала:
15 a1 = 1;
16 % Амплитуда второго сигнала:
```

2.17 Выполнение лабораторной работы

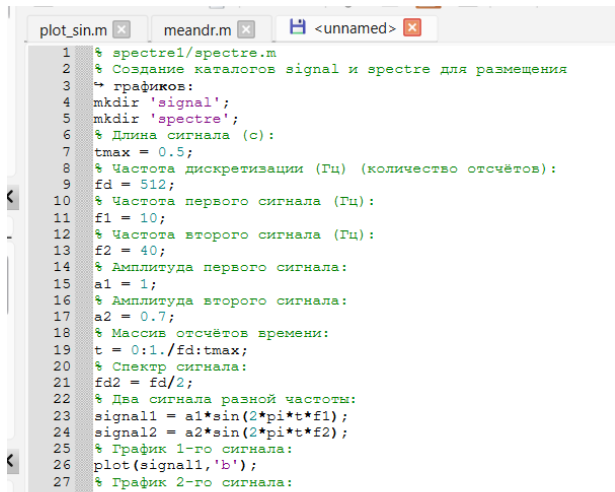
Далее в коде задаю два синусоидальных сигнала разной частоты.



```
plot_sin.m x meandr.m x <unnamed> x
1 % spectre1/spectre.m
2 % Создание каталогов signal и spectre для размещения
3 % графиков:
4 mkdir 'signal';
5 mkdir 'spectre';
6 % Длина сигнала (с):
7 tmax = 0.5;
8 % Частота дискретизации (Гц) (количество отсчётов):
9 fd = 512;
10 % Частота первого сигнала (Гц):
11 f1 = 10;
12 % Частота второго сигнала (Гц):
13 f2 = 40;
14 % Амплитуда первого сигнала:
15 a1 = 1;
16 % Амплитуда второго сигнала:
17 a2 = 0.7;
18 % Массив отсчётов времени:
19 t = 0:1./fd:tmax;
20 % Спектр сигнала:
21 f10 = f1/2;
```

2.18 Выполнение лабораторной работы

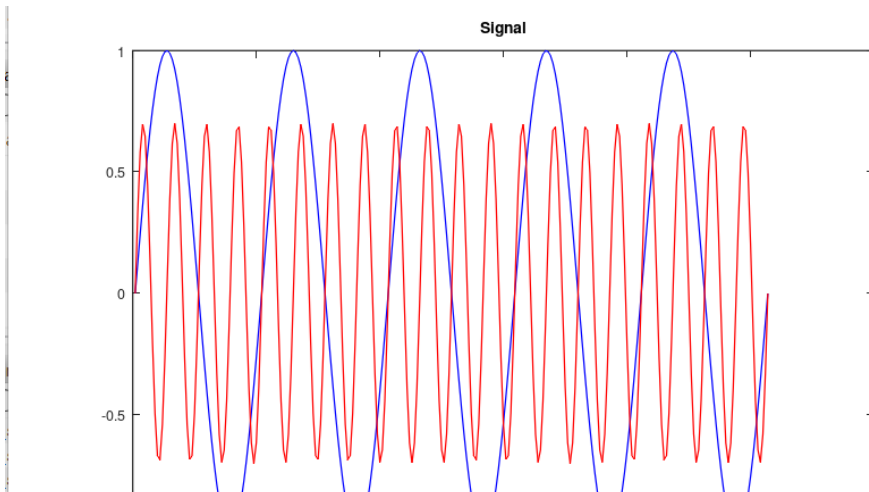
Строю графики сигналов.



```
plot_sin.m x meandr.m x <unnamed> x
1 % spectre1/spectre.m
2 % Создание каталогов signal и spectre для размещения
3 % графиков:
4 mkdir 'signal';
5 mkdir 'spectre';
6 % Длина сигнала (с):
7 tmax = 0.5;
8 % Частота дискретизации (Гц) (количество отсчётов):
9 fd = 512;
10 % Частота первого сигнала (Гц):
11 f1 = 10;
12 % Частота второго сигнала (Гц):
13 f2 = 40;
14 % Амплитуда первого сигнала:
15 a1 = 1;
16 % Амплитуда второго сигнала:
17 a2 = 0.7;
18 % Массив отсчётов времени:
19 t = 0:1./fd:tmax;
20 % Спектр сигнала:
21 fd2 = fd/2;
22 % Два сигнала разной частоты:
23 signal1 = a1*sin(2*pi*t*f1);
24 signal2 = a2*sin(2*pi*t*f2);
25 % График 1-го сигнала:
26 plot(signal1,'b');
27 % График 2-го сигнала:
```

2.19 Выполнение лабораторной работы

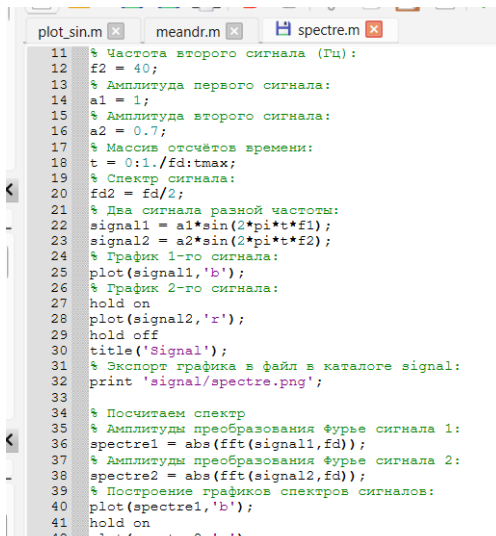
Итоговый график.



2.20 Выполнение лабораторной работы

С помощью быстрого преобразования Фурье нахожу спектры сигналов, добавив в файл `spectre.m` следующий код.

2.21 Выполнение лабораторной работы

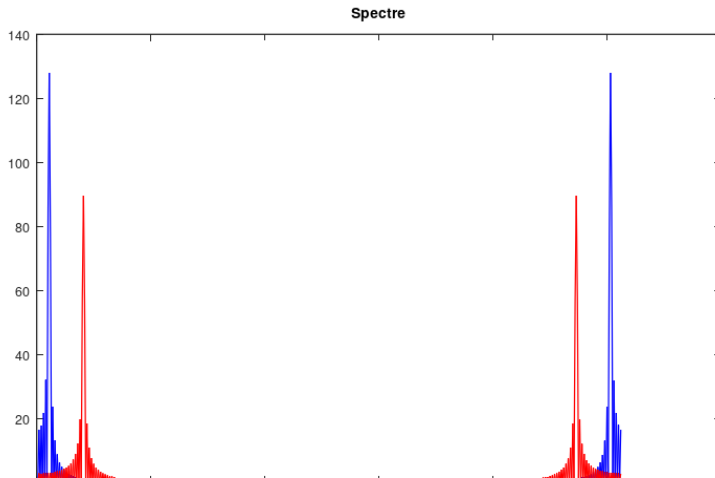
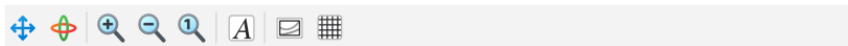


```
plot_sin.m x meandr.m x spectre.m x
11 % Частота второго сигнала (Гц):
12 f2 = 40;
13 % Амплитуда первого сигнала:
14 a1 = 1;
15 % Амплитуда второго сигнала:
16 a2 = 0.7;
17 % Массив отсчётов времени:
18 t = 0:1./fd:tmax;
19 % Спектр сигнала:
20 fd2 = fd/2;
21 % Два сигнала разной частоты:
22 signal1 = a1*sin(2*pi*t*f1);
23 signal2 = a2*sin(2*pi*t*f2);
24 % График 1-го сигнала:
25 plot(signal1,'b');
26 % График 2-го сигнала:
27 hold on
28 plot(signal2,'r');
29 hold off
30 title('Signal');
31 % Экспорт графика в файл в каталоге signal:
32 print 'signal/spectre.png';
33
34 % Посчитаем спектр
35 % Амплитуды преобразования Фурье сигнала 1:
36 spectre1 = abs(fft(signal1,fd));
37 % Амплитуды преобразования Фурье сигнала 2:
38 spectre2 = abs(fft(signal2,fd));
39 % Построение графиков спектров сигналов:
40 plot(spectre1,'b');
41 hold on
42 plot(spectre2,'r');
```

2.22 Выполнение лабораторной работы

Учитывая реализацию преобразования Фурье, корректирую график спектра: отбрасываю дублирующие отрицательные частоты, а также принимаю в расчёт то, что на каждом шаге вычисления быстрого преобразования Фурье происходит суммирование амплитуд сигналов. Итоговый график.

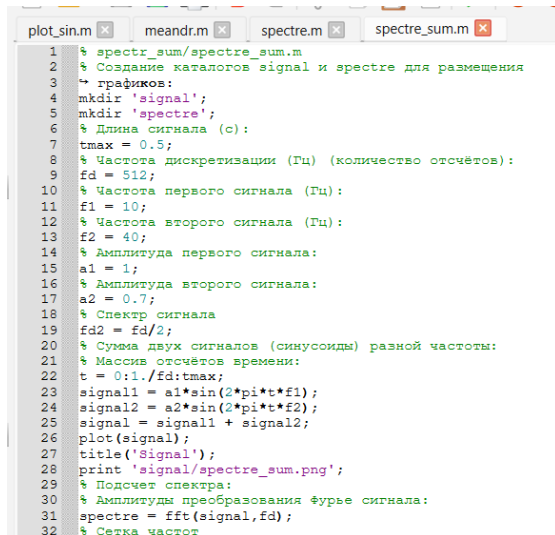
2.23 Выполнение лабораторной работы



2.24 Выполнение лабораторной работы

Нахожу спектр суммы рассмотренных сигналов, создав каталог `spectr_sum` и файл в нём `spectre_sum.m` с кодом.

2.25 Выполнение лабораторной работы

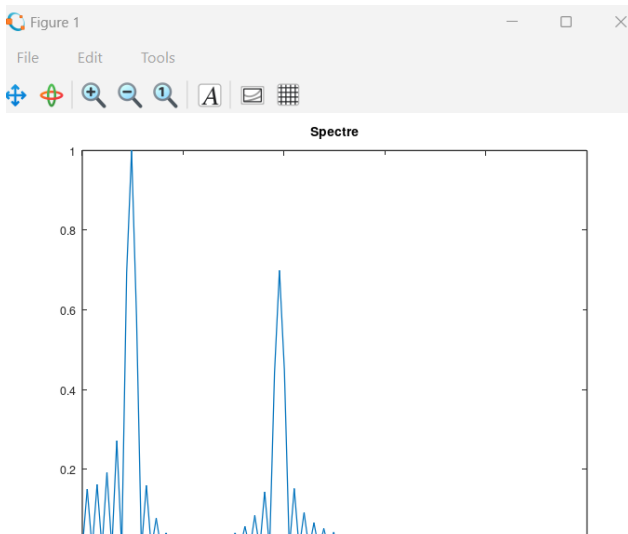


```
1 % spectr_sum/spectre_sum.m
2 % Создание каталогов signal и spectre для размещения
3 % графиков:
4 mkdir 'signal';
5 mkdir 'spectre';
6 % Длина сигнала (с):
7 tmax = 0.5;
8 % Частота дискретизации (Гц) (количество отсчётов):
9 fd = 512;
10 % Частота первого сигнала (Гц):
11 f1 = 10;
12 % Частота второго сигнала (Гц):
13 f2 = 40;
14 % Амплитуда первого сигнала:
15 a1 = 1;
16 % Амплитуда второго сигнала:
17 a2 = 0.7;
18 % Спектр сигнала
19 fd2 = fd/2;
20 % Сумма двух сигналов (синусоиды) разной частоты:
21 % Массив отсчётов времени:
22 t = 0:1./fd:tmax;
23 signal1 = a1*sin(2*pi*t*f1);
24 signal2 = a2*sin(2*pi*t*f2);
25 signal = signal1 + signal2;
26 plot(signal);
27 title('Signal');
28 print 'signal/spectre_sum.png';
29 % Подсчет спектра:
30 % Амплитуды преобразования Фурье сигнала:
31 spectre = fft(signal,fd);
32 % Сетка частот
```

2.26 Выполнение лабораторной работы

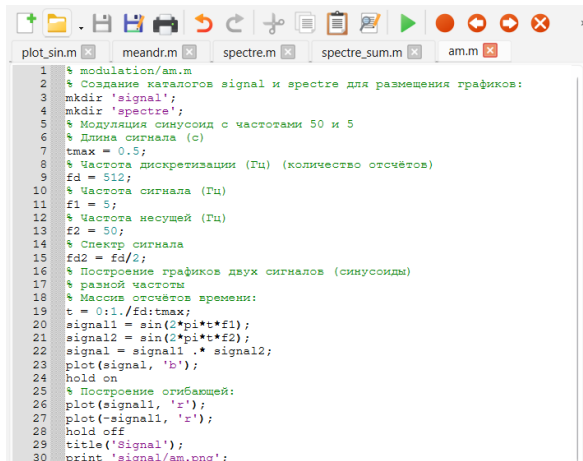
В результате получается аналогичный предыдущему результат, т.е. спектр суммы сигналов должен быть равен сумме спектров сигналов, что вытекает из свойств преобразования Фурье.

2.27 Выполнение лабораторной работы



2.28 Выполнение лабораторной работы

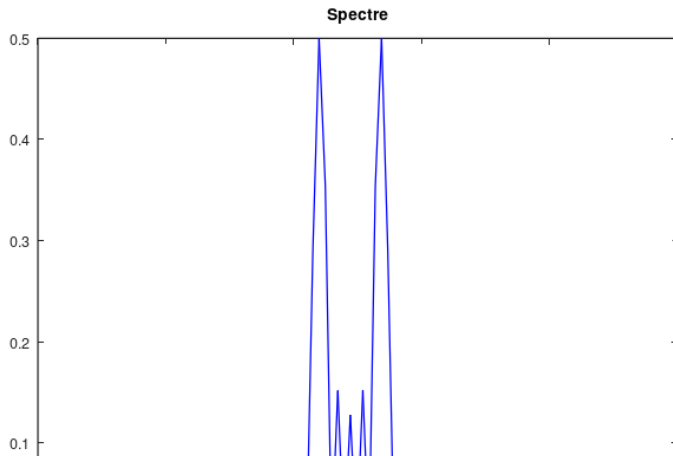
В рабочем каталоге создаю каталог modulation и в нём новый сценарий с именем am.m.



```
1 % modulation/am.m
2 % Создание каталогов signal и spectre для размещения графиков:
3 mkdir 'signal';
4 mkdir 'spectre';
5 % Модуляция синусоид с частотами 50 и 5
6 % Длина сигнала (с)
7 tmax = 0.5;
8 % Частота дискретизации (Гц) (количество отсчётов)
9 fd = 512;
10 % Частота сигнала (Гц)
11 f1 = 5;
12 % Частота несущей (Гц)
13 f2 = 50;
14 % Спектр сигнала
15 fd2 = fd/2;
16 % Построение графиков двух сигналов (синусоиды)
17 % разной частоты
18 % Массив отсчётов времени:
19 t = 0:1./fd:tmax;
20 signal1 = sin(2*pi*t*f1);
21 signal2 = sin(2*pi*t*f2);
22 signal = signal1 .* signal2;
23 plot(signal, 'b');
24 hold on
25 % Построение огибающей:
26 plot(signal1, 'r');
27 plot(-signal1, 'r');
28 hold off
29 title('Signal');
30 print 'signal/am.png';
```

2.29 Выполнение лабораторной работы

В результате получаю, что спектр произведения представляет собой свёртку спектров.



2.30 Выполнение лабораторной работы

В рабочем каталоге создаю каталог `coding` и в нём файлы `main.m`, `maptowave.m`, `unipolar.m`, `ami.m`, `bipolarnrz.m`, `bipolarrrz.m`, `manchester.m`, `diffmanc.m`, `calcspectre.m`.

2.31 Выполнение лабораторной работы









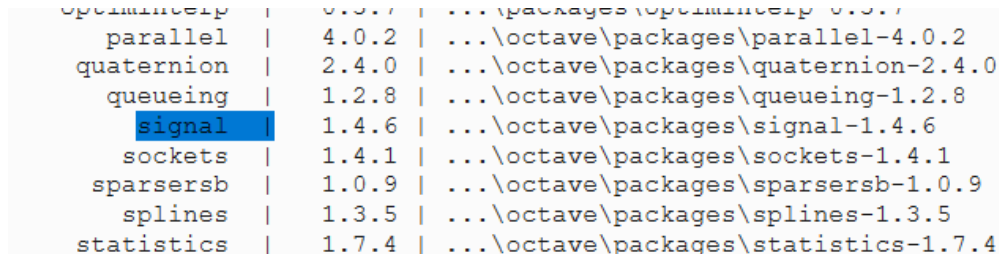
Имя	Дата изменения	Тип	Размер
 ami.m	13.09.2025 16:10	Octave.Document....	0 КБ
 bipolarnrz.m	13.09.2025 16:10	Octave.Document....	0 КБ
 bipolarrrz.m	13.09.2025 16:10	Octave.Document....	0 КБ
 diffmanc.m	13.09.2025 16:11	Octave.Document....	0 КБ
 main.m	13.09.2025 16:09	Octave.Document....	0 КБ
 manchester.m	13.09.2025 16:11	Octave.Document....	0 КБ
 maptowave.m	13.09.2025 16:09	Octave.Document....	0 КБ
 unipolar.m	13.09.2025 16:10	Octave.Document....	0 КБ

Рисунок 21: Создание каталога и файлов

2.32 Выполнение лабораторной работы

В окне интерпретатора команд проверяю, установлен ли у вас пакет расширений `signal`. Она у меня установлена.

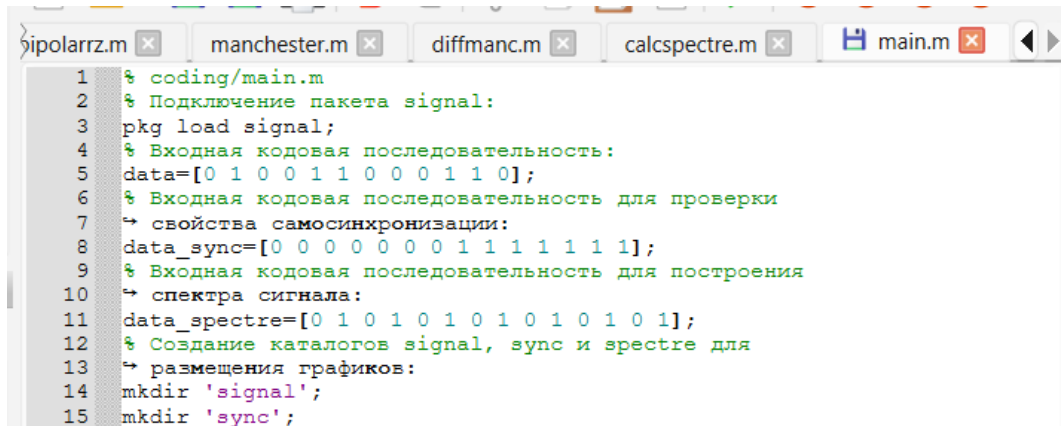


optiminterp		0.3.7		... \packages \optiminterp 0.3.7
parallel		4.0.2		... \octave \packages \parallel-4.0.2
quaternion		2.4.0		... \octave \packages \quaternion-2.4.0
queueing		1.2.8		... \octave \packages \queueing-1.2.8
signal		1.4.6		... \octave \packages \signal-1.4.6
sockets		1.4.1		... \octave \packages \sockets-1.4.1
sparsersb		1.0.9		... \octave \packages \sparsersb-1.0.9
splines		1.3.5		... \octave \packages \splines-1.3.5
statistics		1.7.4		... \octave \packages \statistics-1.7.4

Рисунок 22: Пакет расширений `signal`

2.33 Выполнение лабораторной работы

В файле `main.m` подключаю пакет `signal` и задаю входные кодовые последовательности.



```
1 % coding/main.m
2 % Подключение пакета signal:
3 pkg load signal;
4 % Входная кодовая последовательность:
5 data=[0 1 0 0 1 1 0 0 0 1 1 0];
6 % Входная кодовая последовательность для проверки
7 % свойства самосинхронизации:
8 data_sync=[0 0 0 0 0 0 0 1 1 1 1 1 1];
9 % Входная кодовая последовательность для построения
10 % спектра сигнала:
11 data_spectre=[0 1 0 1 0 1 0 1 0 1 0 1 0 1];
12 % Создание каталогов signal, sync и spectre для
13 % размещения графиков:
14 mkdir 'signal';
15 mkdir 'sync';
```

2.34 Выполнение лабораторной работы

Затем в этом же файле прописваю вызовы функций для построения графиков модуляций кодированных сигналов для кодовой последовательности `data`.

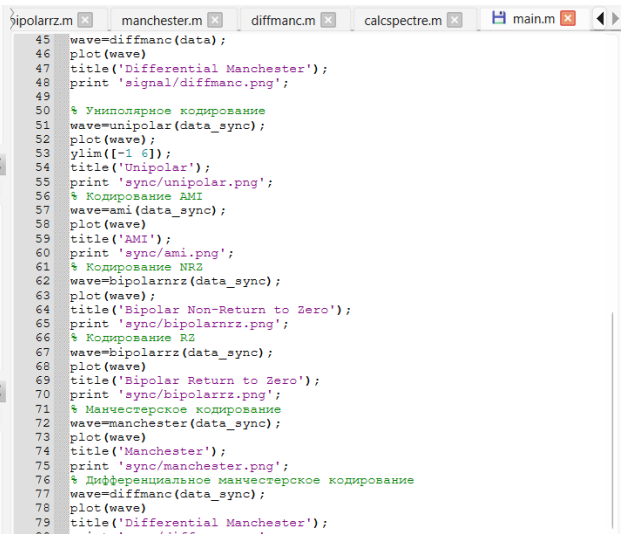
2.35 Выполнение лабораторной работы

```
bipolarrrz.m x manchester.m x diffmanc.m x calcspectre.m x main.m x
14 mkdir 'signal';
15 mkdir 'sync';
16 mkdir 'spectre';
17 axis("auto");
18 % Униполярное кодирование
19 wave=unipolar(data);
20 plot(wave);
21 ylim([-1 6]);
22 title('Unipolar');
23 print 'signal/unipolar.png';
24 % Кодирование ami
25 wave=ami(data);
26 plot(wave)
27 title('AMI');
28 print 'signal/ami.png';
29 % Кодирование NRZ
30 wave=bipolarnrz(data);
31 plot(wave);
32 title('Bipolar Non-Return to Zero');
33 print 'signal/bipolarnrz.png';
34 % Кодирование RZ
35 wave=bipolarrrz(data);
36 plot(wave)
37 title('Bipolar Return to Zero');
38 print 'signal/bipolarrrz.png';
39 % Манчестерское кодирование
40 wave=manchester(data);
41 plot(wave)
42 title('Manchester');
43 print 'signal/manchester.png';
44 % Дифференциальное манчестерское кодирование
```


2.36 Выполнение лабораторной работы

Затем в этом же файле прописываю вызовы функций для построения графиков модуляций кодированных сигналов для кодовой последовательности `data_sync`.

2.37 Выполнение лабораторной работы



```
bipolarrz.m x manchester.m x diffmanc.m x calcspectre.m x main.m x
45 wave=diffmanc(data);
46 plot(wave)
47 title('Differential Manchester');
48 print 'signal/diffmanc.png';
49
50 % Униполярное кодирование
51 wave=unipolar(data_sync);
52 plot(wave);
53 ylim([-1 6]);
54 title('Unipolar');
55 print 'sync/unipolar.png';
56 % Кодирование AMI
57 wave=ami(data_sync);
58 plot(wave)
59 title('AMI');
60 print 'sync/ami.png';
61 % Кодирование NRZ
62 wave=bipolarnrz(data_sync);
63 plot(wave);
64 title('Bipolar Non-Return to Zero');
65 print 'sync/bipolarnrz.png';
66 % Кодирование RZ
67 wave=bipolarrz(data_sync);
68 plot(wave)
69 title('Bipolar Return to Zero');
70 print 'sync/bipolarrz.png';
71 % Манчестерское кодирование
72 wave=manchester(data_sync);
73 plot(wave)
74 title('Manchester');
75 print 'sync/manchester.png';
76 % Дифференциальное манчестерское кодирование
77 wave=diffmanc(data_sync);
78 plot(wave)
79 title('Differential Manchester');
```

2.38 Выполнение лабораторной работы

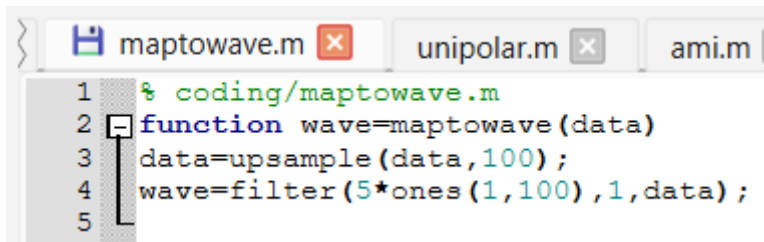
Далее в этом же файле прописываю вызовы функций для построения графиков спектров.

2.39 Выполнение лабораторной работы

```
bipolarrz.m x manchester.m x diffmanc.m x calcspectre.m x main.m x
76 % Дифференциальное манчестерское кодирование
77 wave=diffmanc(data_sync);
78 plot(wave)
79 title('Differential Manchester');
80 print 'sync/diffmanc.png';
81
82 % Униполярное кодирование:
83 wave=unipolar(data_spectre);
84 spectre=calcspectre(wave);
85 title('Unipolar');
86 print 'spectre/unipolar.png';
87 % Кодирование AMI:
88 wave=ami(data_spectre);
89 spectre=calcspectre(wave);
90 title('AMI');
91 print 'spectre/ami.png';
92 % Кодирование NRZ:
93 wave=bipolarnrz(data_spectre);
94 spectre=calcspectre(wave);
95 title('Bipolar Non-Return to Zero');
96 print 'spectre/bipolarnrz.png';
97 % Кодирование RZ:
98 wave=bipolarrz(data_spectre);
99 spectre=calcspectre(wave);
100 title('Bipolar Return to Zero');
101 print 'spectre/bipolarrz.png';
102 % Манчестерское кодирование:
103 wave=manchester(data_spectre);
104 spectre=calcspectre(wave);
105 title('Manchester');
106 print 'spectre/manchester.png';
107 % Дифференциальное манчестерское кодирование:
108 wave=diffmanc(data_spectre);
```

2.40 Выполнение лабораторной работы

В файле `maptowave.m` пропишите функцию, которая по входному битовому потоку строит график сигнала.



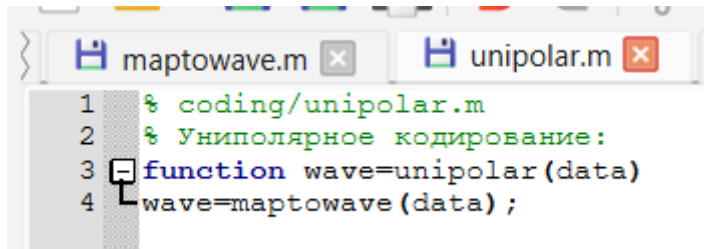
```
1 % coding/maptowave.m
2 function wave=maptowave(data)
3     data=upsample(data,100);
4     wave=filter(5*ones(1,100),1,data);
5
```

Рисунок 27: Редактирование `maptowave.m`

2.41 Выполнение лабораторной работы

В файлах `unipolar.m`, `ami.m`, `bipolarnrz.m`, `bipolarrz.m`, `manchester.m`, `diffmanc.m` прописываю соответствующие функции преобразования кодовой последовательности `data` с вызовом функции `marptowave` для построения соответствующего графика. Униполярное кодирование:

2.42 Выполнение лабораторной работы

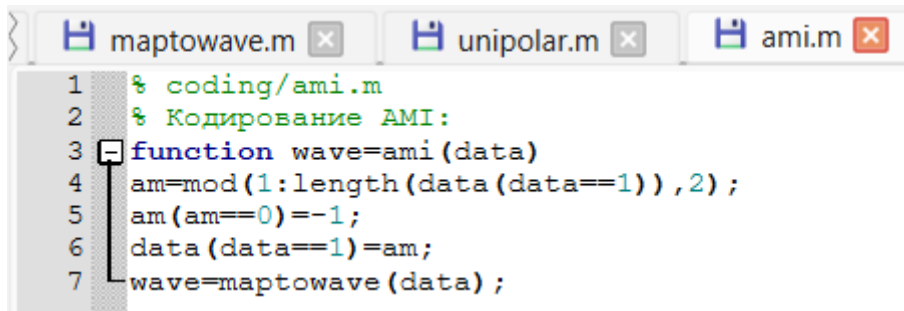


```
1 % coding/unipolar.m
2 % Униполярное кодирование:
3 function wave=unipolar(data)
4 wave=maptowave(data);
```

Рисунок 28: Редактирование unipolar.m

2.43 Выполнение лабораторной работы

Кодирование AMI:

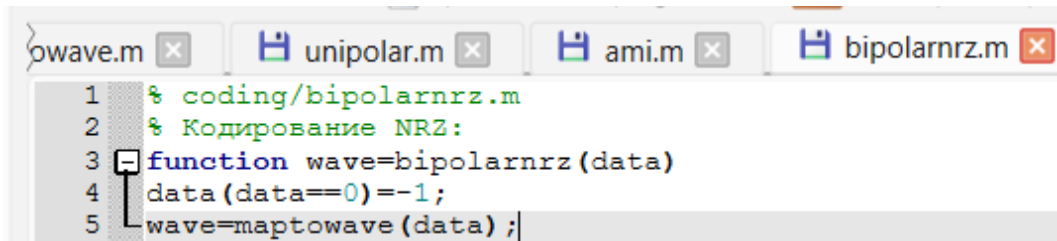


```
1 % coding/ami.m
2 % Кодирование AMI:
3 function wave=ami(data)
4     am=mod(1:length(data(data==1)),2);
5     am(am==0)=-1;
6     data(data==1)=am;
7     wave=maptowave(data);
```

Рисунок 29: Редактирование ami.m

2.44 Выполнение лабораторной работы

Кодирование NRZ:

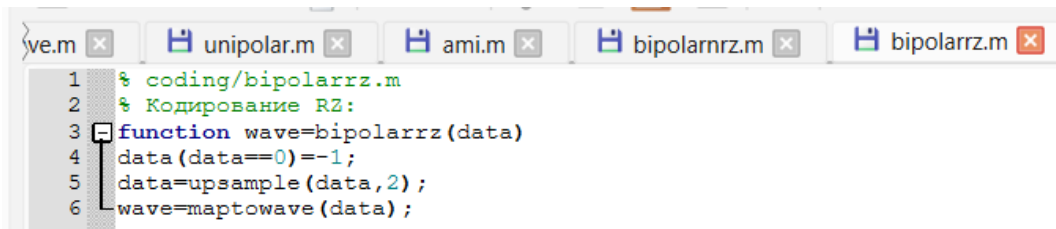


```
1 % coding/bipolarnrz.m
2 % Кодирование NRZ:
3 function wave=bipolarnrz(data)
4     data(data==0)=-1;
5     wave=maptowave(data);
```

Рисунок 30: Редактирование bipolarnrz.m

2.45 Выполнение лабораторной работы

Кодирование RZ:



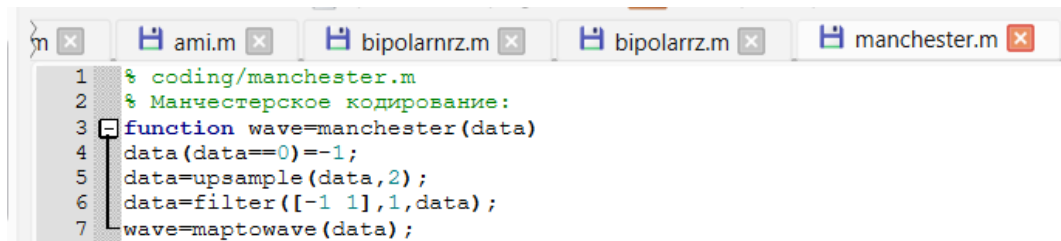
The screenshot shows a MATLAB editor window with several tabs: 've.m', 'unipolar.m', 'ami.m', 'bipolarnrz.m', and 'bipolarrz.m'. The 'bipolarrz.m' tab is active, displaying the following code:

```
1 % coding/bipolarrz.m
2 % Кодирование RZ:
3 function wave=bipolarrz(data)
4     data(data==0)=-1;
5     data=upsample(data,2);
6     wave=maptowave(data);
```

Рисунок 31: Редактирование bipolarrrz.m

2.46 Выполнение лабораторной работы

Манчестерское кодирование:



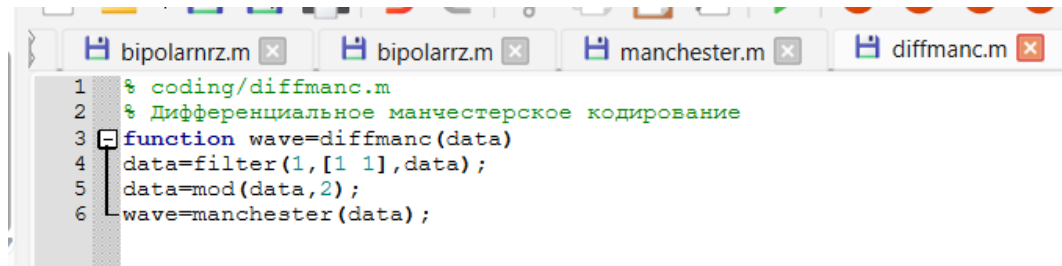
The screenshot shows a MATLAB editor window with five tabs: 'ami.m', 'bipolarnrz.m', 'bipolarrz.m', and 'manchester.m'. The 'manchester.m' tab is active, displaying the following code:

```
1 % coding/manchester.m
2 % Манчестерское кодирование:
3 function wave=manchester(data)
4 data(data==0)=-1;
5 data=upsample(data,2);
6 data=filter([-1 1],1,data);
7 wave=maptowave(data);
```

Рисунок 32: Редактирование manchester.m

2.47 Выполнение лабораторной работы

Дифференциальное манчестерское кодирование:

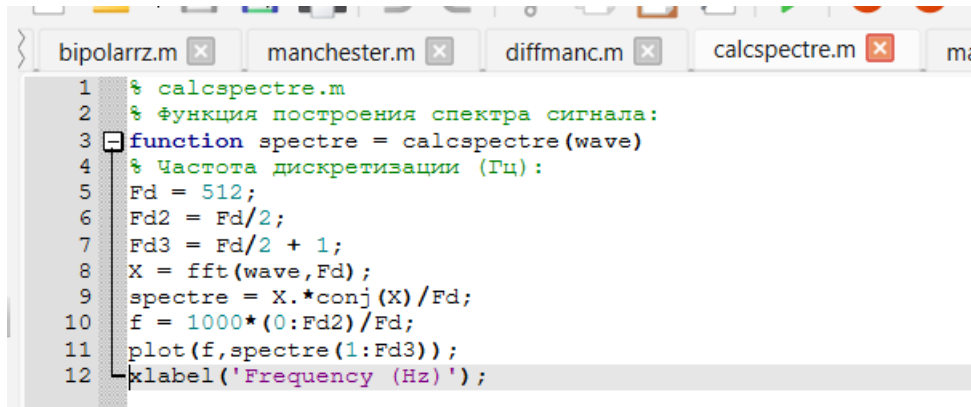


```
1 % coding/diffmanc.m
2 % Дифференциальное манчестерское кодирование
3 function wave=diffmanc(data)
4 data=filter(1,[1 1],data);
5 data=mod(data,2);
6 wave=manchester(data);
```

Рисунок 33: Редактирование diffmanc.m

2.48 Выполнение лабораторной работы

В файле calcspectre.m прописываю функцию построения спектра сигнала.



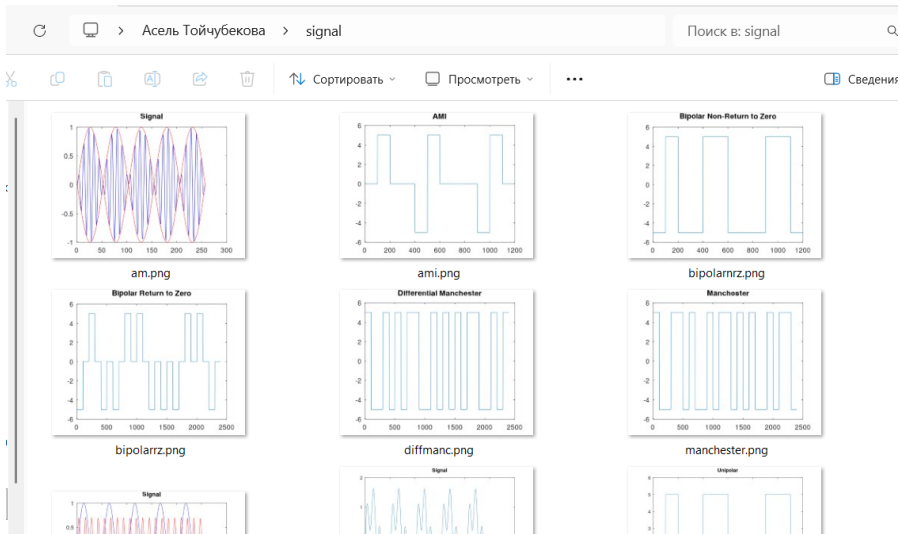
```
1 % calcspectre.m
2 % функция построения спектра сигнала:
3 function spectre = calcspectre(wave)
4 % Частота дискретизации (Гц):
5 Fd = 512;
6 Fd2 = Fd/2;
7 Fd3 = Fd/2 + 1;
8 X = fft(wave,Fd);
9 spectre = X.*conj(X)/Fd;
10 f = 1000*(0:Fd2)/Fd;
11 plot(f,spectre(1:Fd3));
12 xlabel('Frequency (Hz)');
```

Рисунок 34: Редактирование calcspectre.m

2.49 Выполнение лабораторной работы

Запускаю главный скрипт `main.m`. В каталоге `signal` получены файлы с графиками кодированного сигнала.

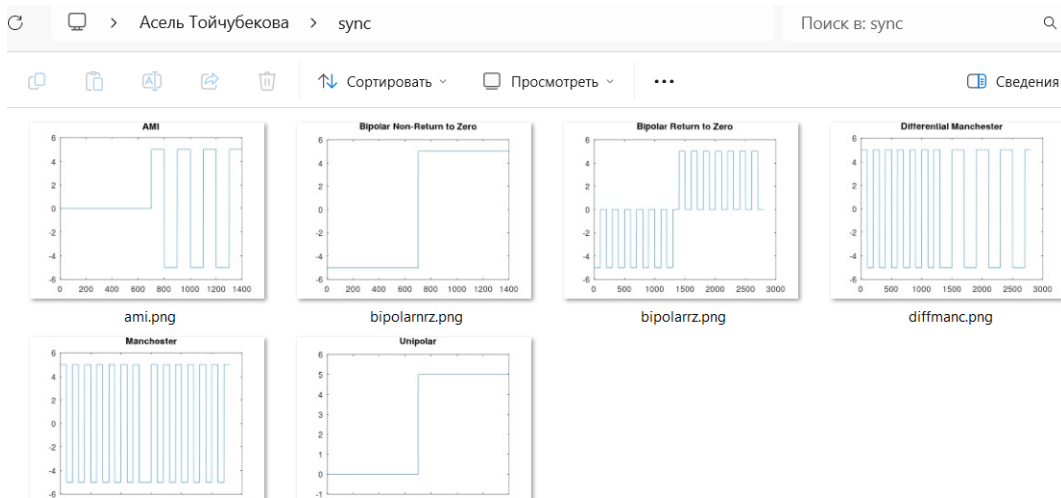
2.50 Выполнение лабораторной работы



2.51 Выполнение лабораторной работы

В каталоге `sync`— файлы с графиками, иллюстрирующими свойства самосинхронизации.

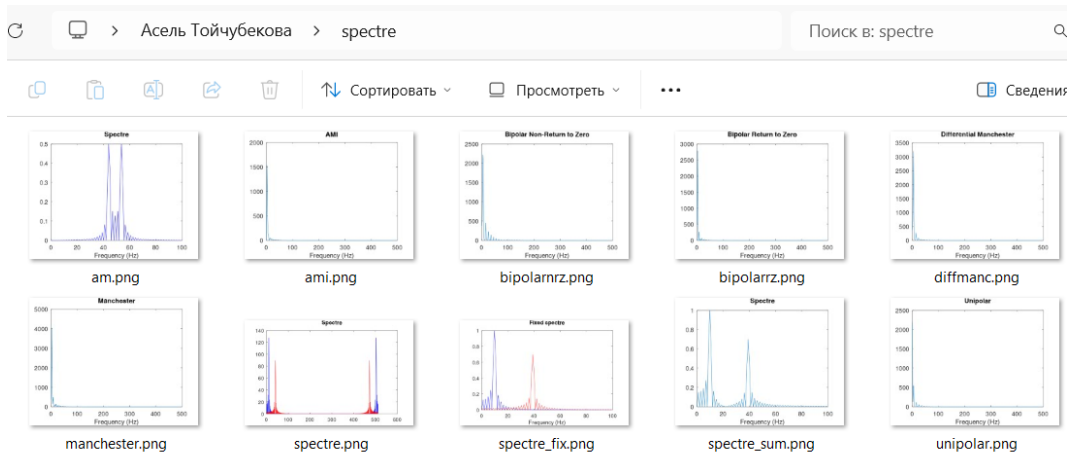
2.52 Выполнение лабораторной работы



2.53 Выполнение лабораторной работы

В каталоге spectre — файлы с графиками спектров сигналов.

2.54 Выполнение лабораторной работы



2.55 Выводы

В ходе выполнения лабораторной работы № 1 я изучила методы кодирования и модуляции сигналов с помощью высокоуровневого языка программирования Octave. Определила спектр и параметр сигнала. Демонстрировала принципы модуляции сигнала на примере аналоговой амплитудной модуляции. Исследовала свойства самосинхронизации сигнала.