

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ
Факультет физико-математических и естественных наук
Кафедра прикладной информатики и теории
вероятностей

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

дисциплина: Сетевые Технологии

Студент: Алхатиб Осама

Группа: НПИбд-02-20

МОСКВА

2022 г

Цели работы

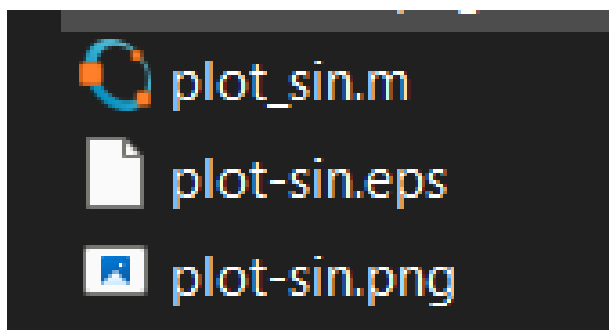
Изучение методов кодирования и модуляции сигналов с помощью высокоуровневого языка программирования Octave. Определение спектра и параметров сигнала. Демонстрация принципов модуляции сигнала на примере аналоговой амплитудной модуляции. Исследование свойства самосинхронизации сигнала.

1.3.1.1. Постановка задачи

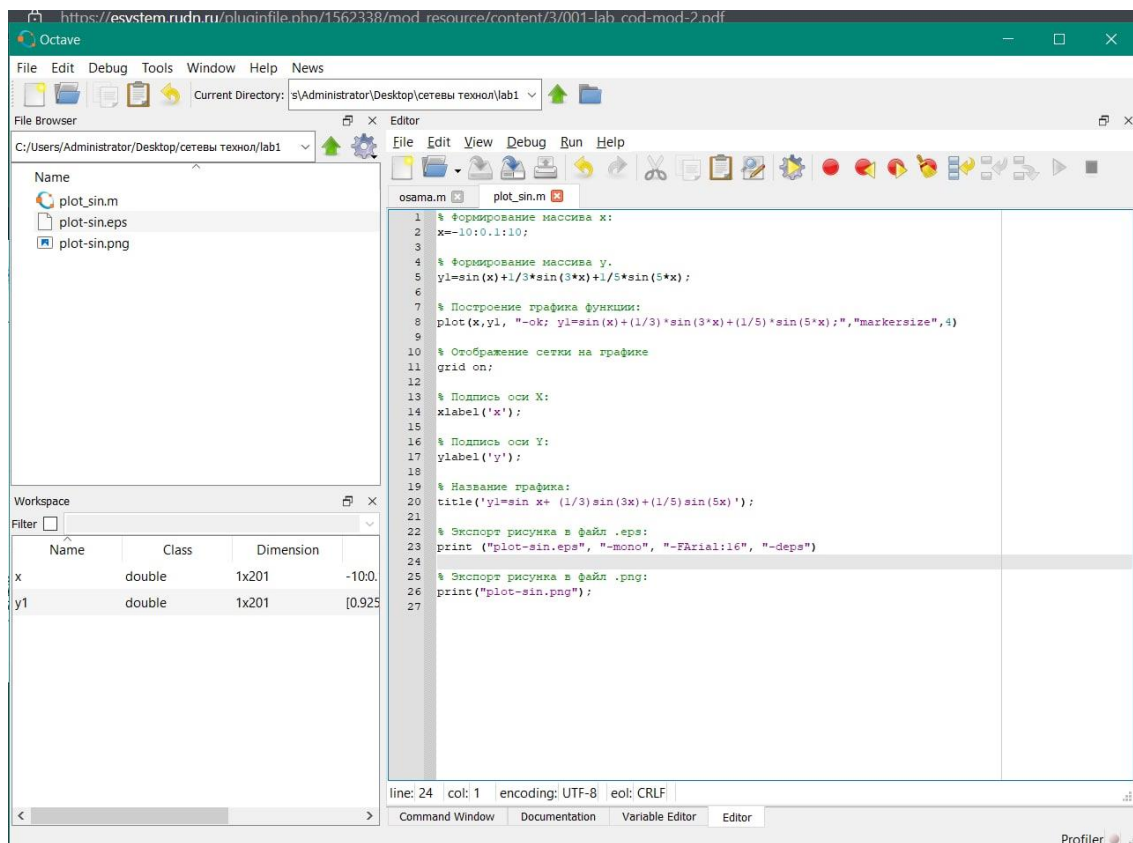
1. Построить график функции $y = \sin x + 1/3 \sin 3x + 1/5 \sin 5x$ на интервале $[-10; 10]$, используя Octave и функцию plot. График экспортировать в файлы формата .eps, .png.
2. Добавить график функции $y = \cos x + 1/3 \cos 3x + 1/5 \cos 5x$ на интервале $[-10; 10]$. График экспортировать в файлы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

1.3.1.2. Порядок выполнения работы

1. Запустите в вашей ОС Octave с оконным интерфейсом.
2. Перейдите в окно редактора. Воспользовавшись меню или комбинацией клавиш ctrl + n создайте новый сценарий. Сохраните его в ваш рабочий каталог с именем, например, plot_sin.m.



3. В окне редактора повторите следующий листинг по построению графика функции $y = \sin x + 1/3 \sin 3x + 1/5 \sin 5x$ на интервале $[-10; 10]$:



4. Запустите сценарий на выполнение (воспользуйтесь соответствующим меню окна редактора или клавишей F5). В качестве результата выполнения кода должно открыться окно с построенным графиком (рис. 1.1) и в вашем рабочем каталоге должны появиться файлы с графиками в форматах .eps, .png.

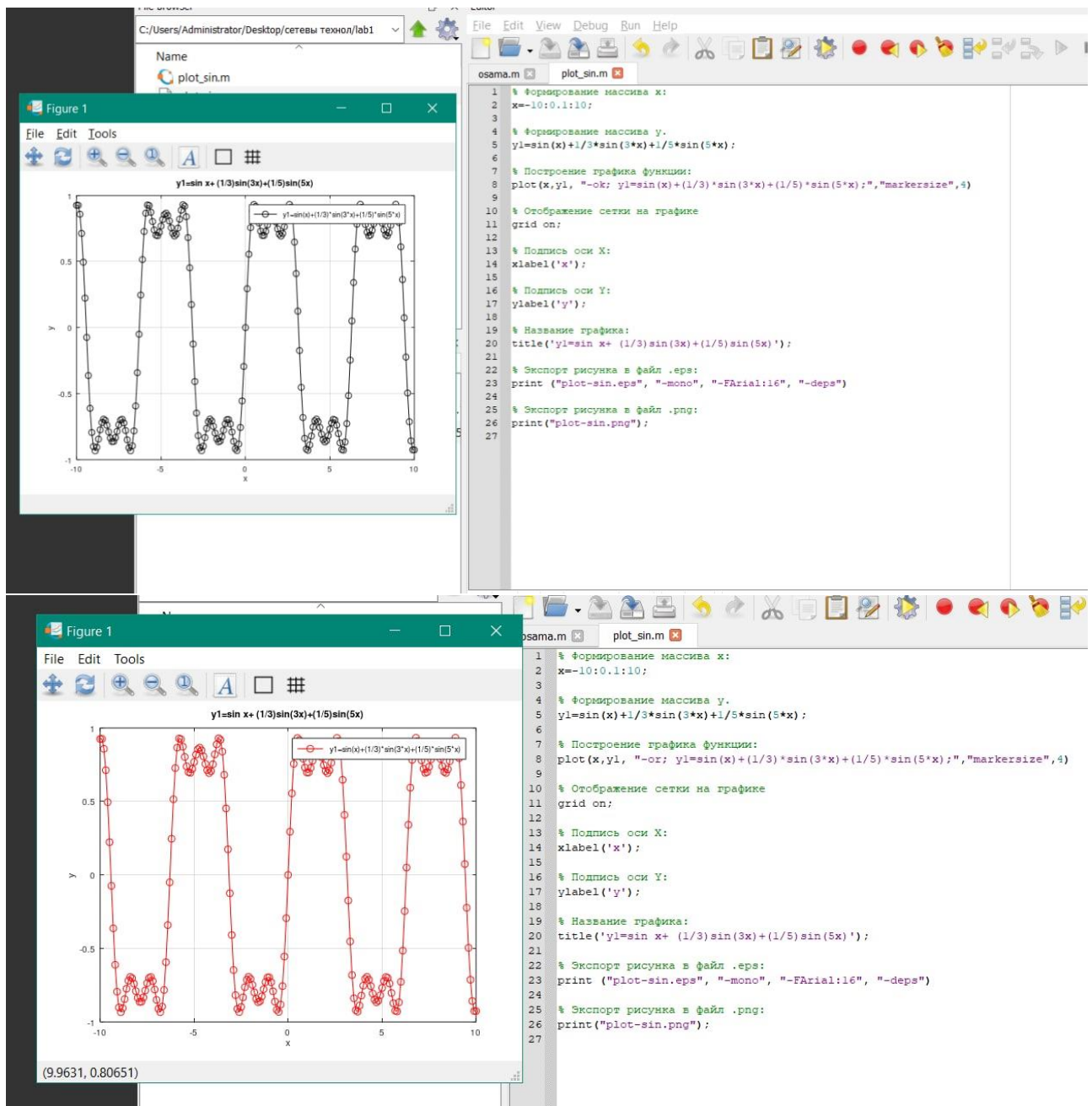


Рис. 1.1. График функции $y = \sin x + \frac{1}{3} \sin 3x + \frac{1}{5} \sin 5x$ на интервале $[-10; 10]$

5. Сохраните сценарий под другим названием и измените его так, чтобы на одном графике располагались отличающиеся по типу линий графики функций

$y_1 = \sin x + \frac{1}{3} \sin 3x + \frac{1}{5} \sin 5x$, $y_2 = \cos x + \frac{1}{3} \cos 3x + \frac{1}{5} \cos 5x$, например как изображено на рис. 1.2.

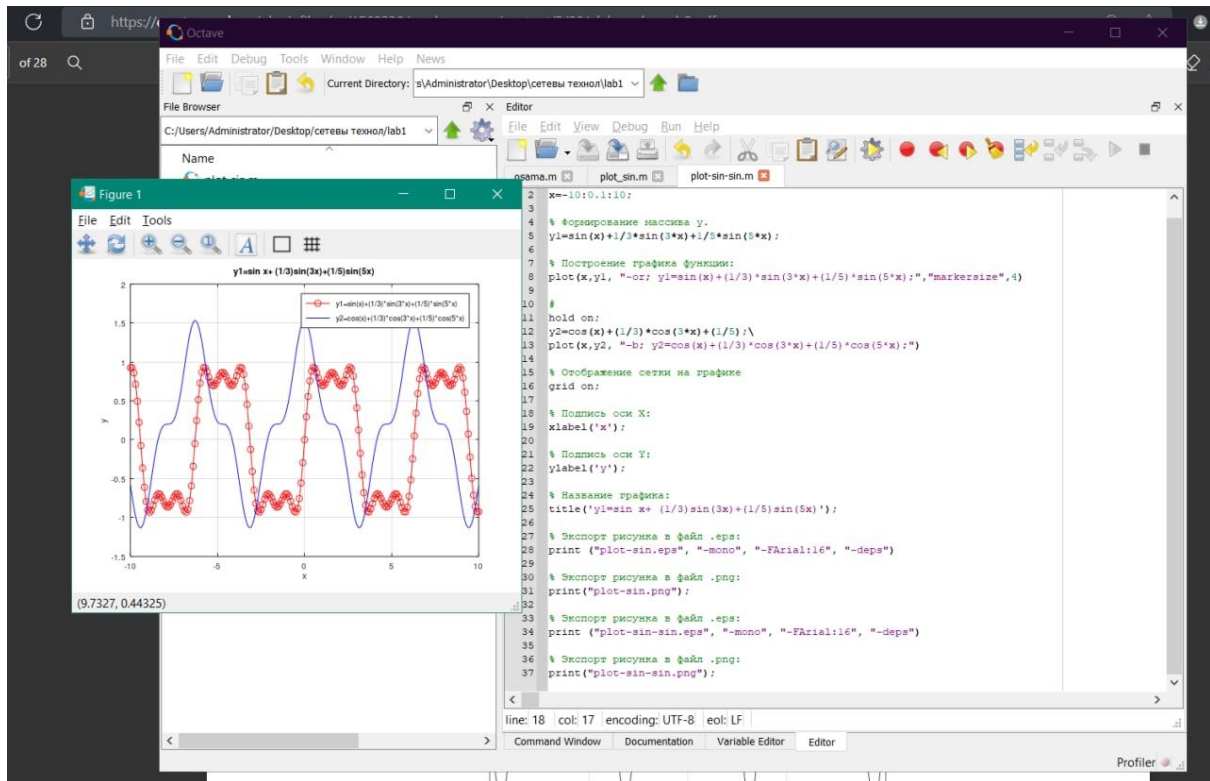


Рис. 1.2. График функций y_1 и y_2 на интервале $[-10; 10]$

1.3.2. Разложение импульсного сигнала в частичный ряд Фурь

1.3.2.1. Постановка задачи

1. Разработать код m-файла, результатом выполнения которого являются графики меандра (рис. 1.3), реализованные с различным количе

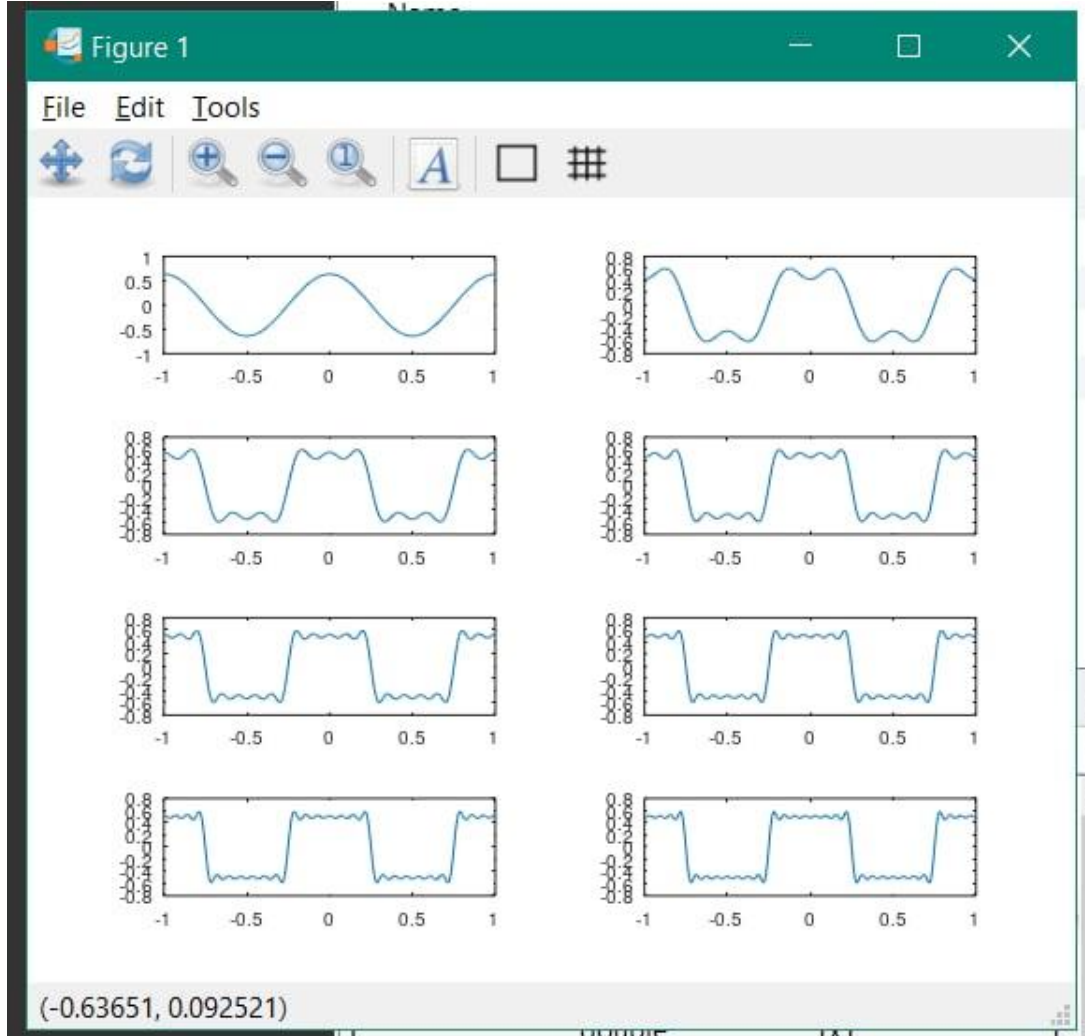
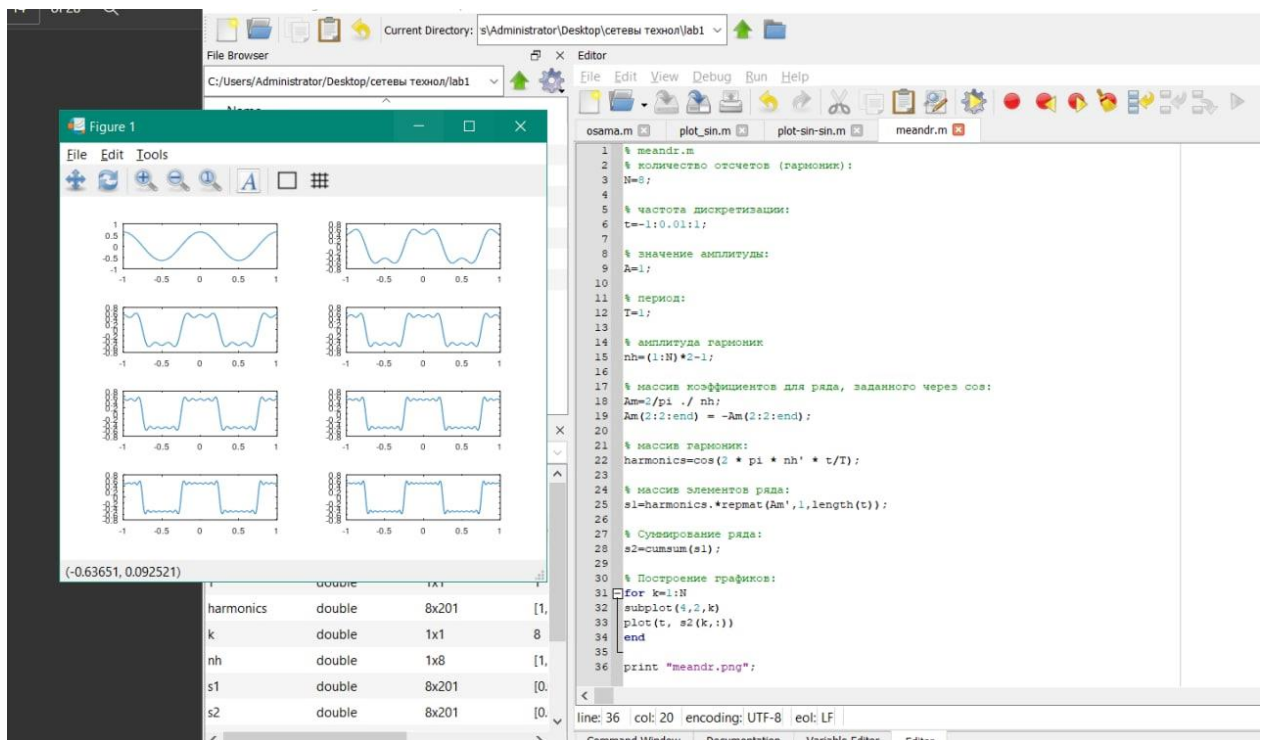


Рис. 1.3. Графики меандра, содержащего различное число гармоник

1.3.2.2. Порядок выполнения работы

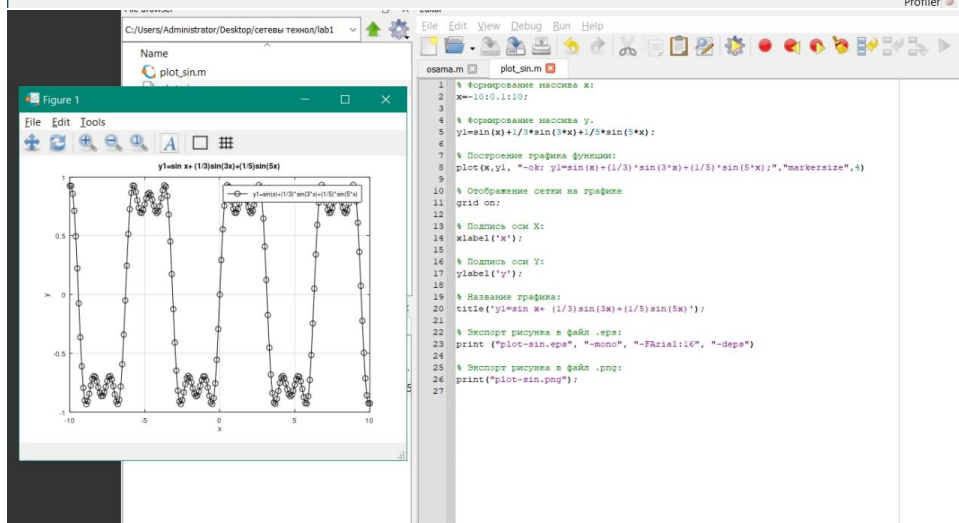
1. Создайте новый сценарий и сохраните его в ваш рабочий каталог с именем, например, meandr.m.

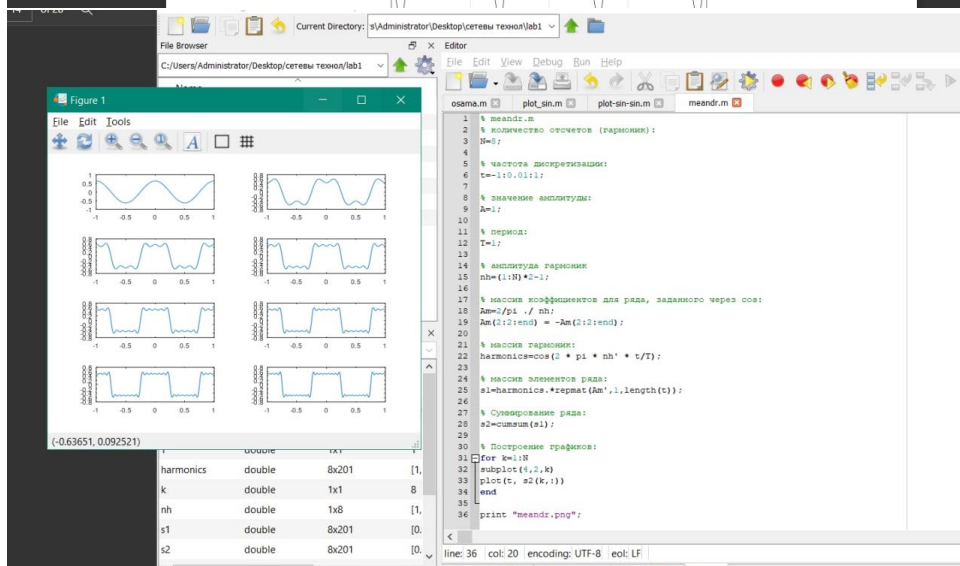
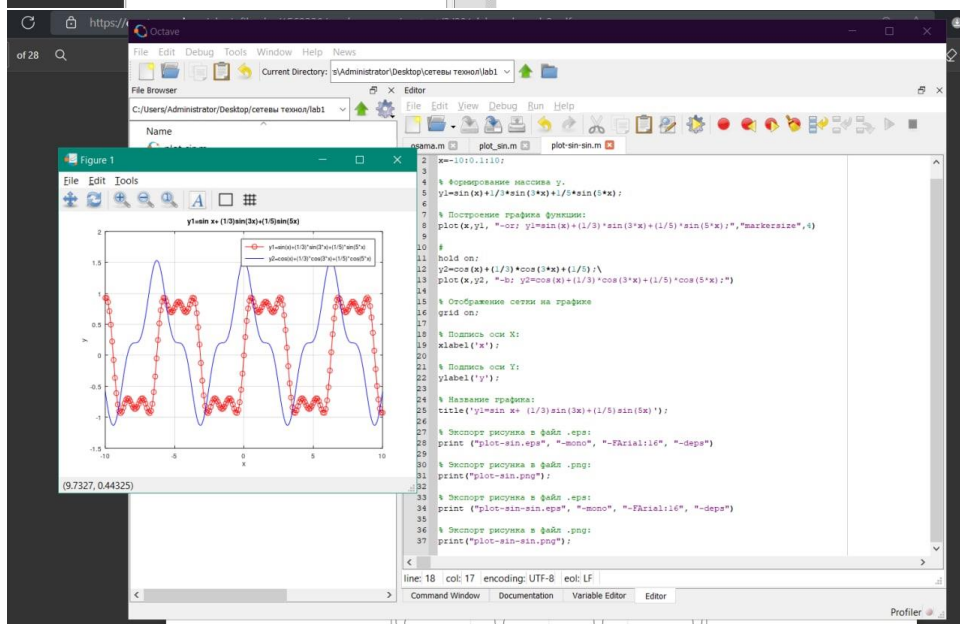
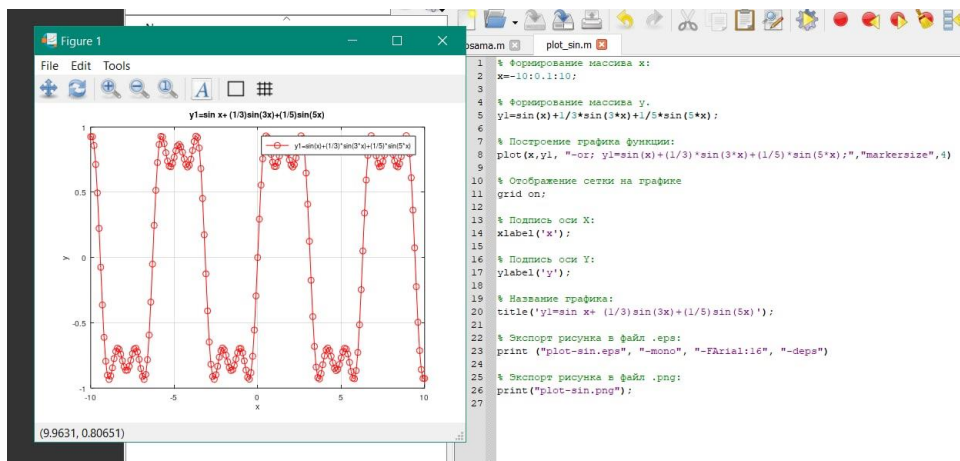
2. В коде созданного сценария задайте начальные значения:

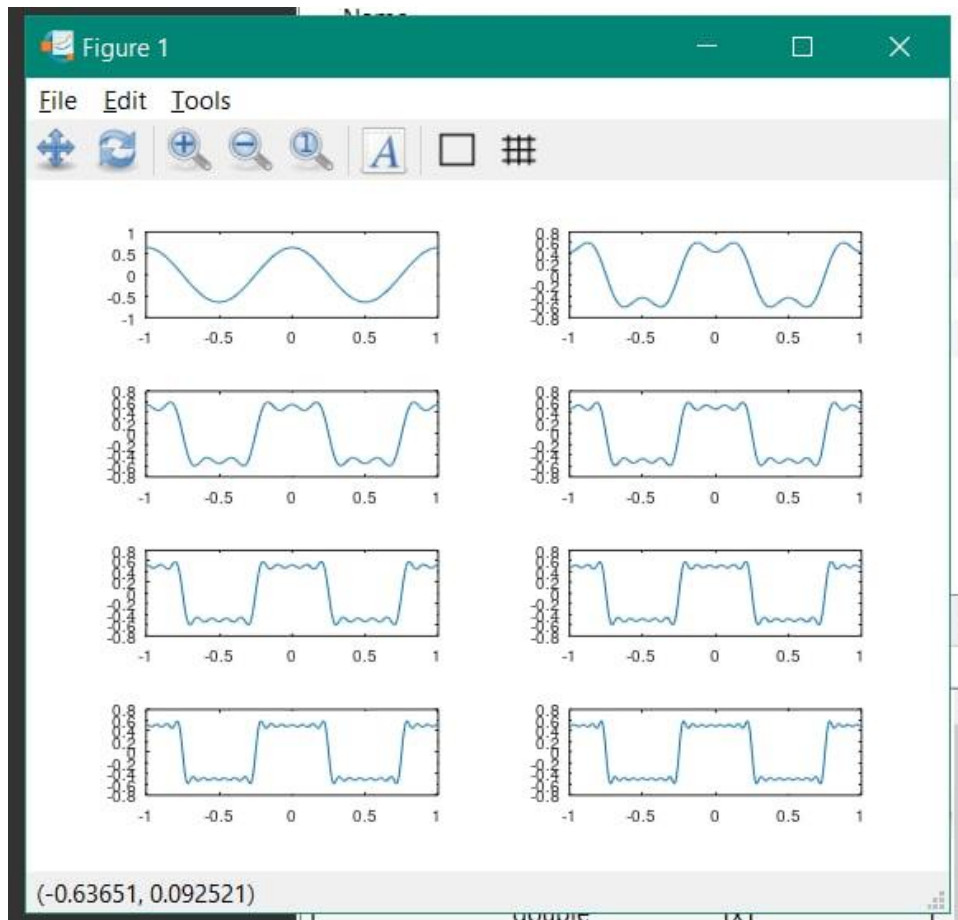


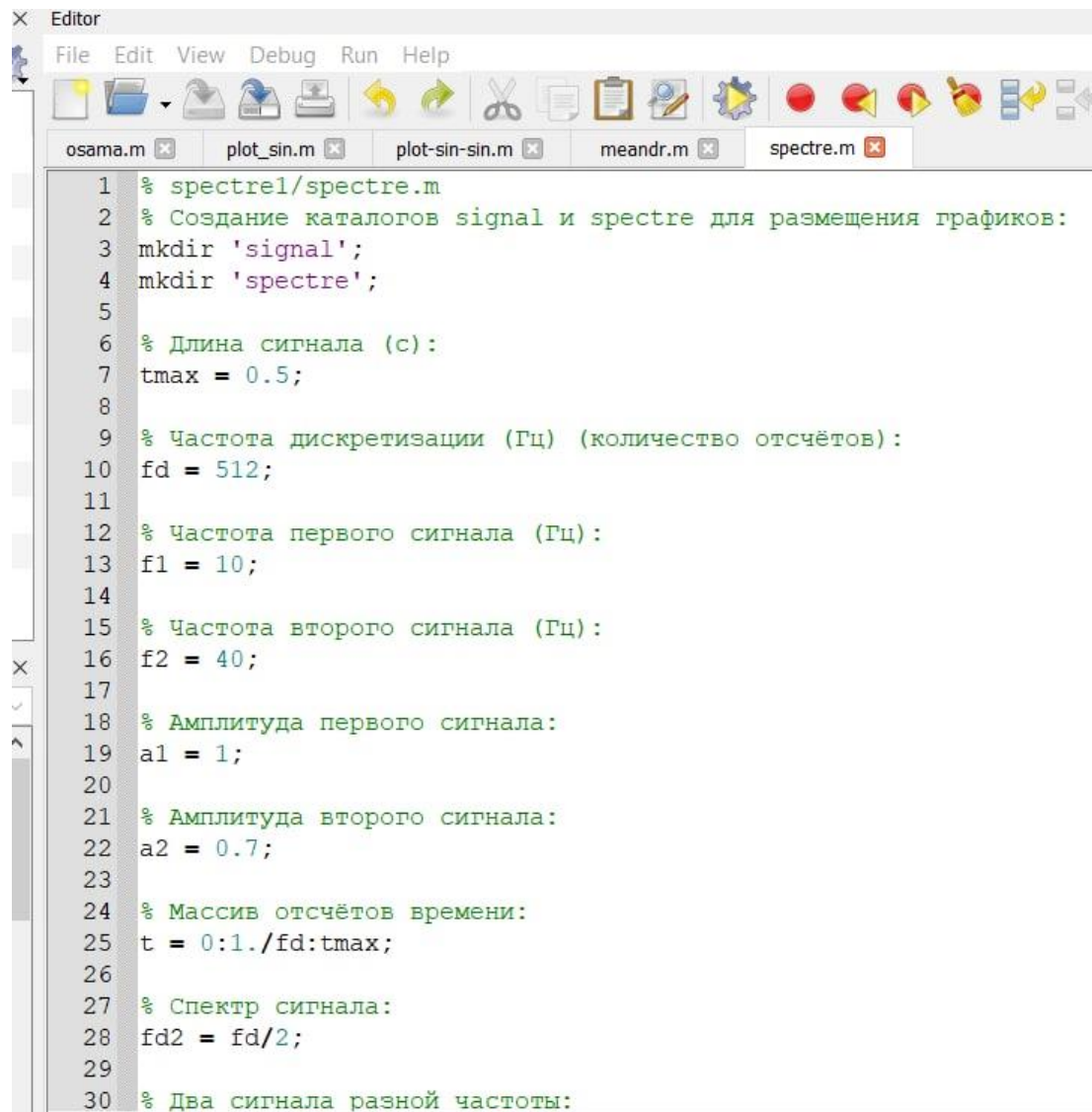
The screenshot shows the Octave IDE interface. The top menu bar includes File, Edit, Debug, Tools, Window, Help, and News. The current directory is set to 'C:\Users\Administrator\Desktop\сервисы\техно\lab1'. The file browser on the left shows files: plot_sin.m, plot_sin.eps, and plot_sin.png. The workspace table lists variables x and y1, both of type double, with dimensions 1x201 and 1x201 respectively. The editor window shows the following code:

```
1 % формирование массива x:
2 x=-10:0.1:10;
3
4 % формирование массива y.
5 y1=sin(x)+1/3*sin(3*x)+1/5*sin(5*x);
6
7 % Построение графика функции:
8 plot(x,y1, "o-k", 'y1=sin(x)+(1/3)*sin(3*x)+(1/5)*sin(5*x)', "markersize", 4)
9
10 % Отображение сетки на графике
11 grid on;
12
13 % Подпись оси X:
14 xlabel('x');
15
16 % Подпись оси Y:
17 ylabel('y');
18
19 % Название графика:
20 title('y1=sin x + (1/3)sin(3x)+(1/5)sin(5x)');
21
22 % Экспорт рисунка в файл .eps:
23 print ("plot-sin.eps", "-mono", "-FArial:16", "-deps")
24
25 % Экспорт рисунка в файл .png:
26 print("plot-sin.png");
27
```









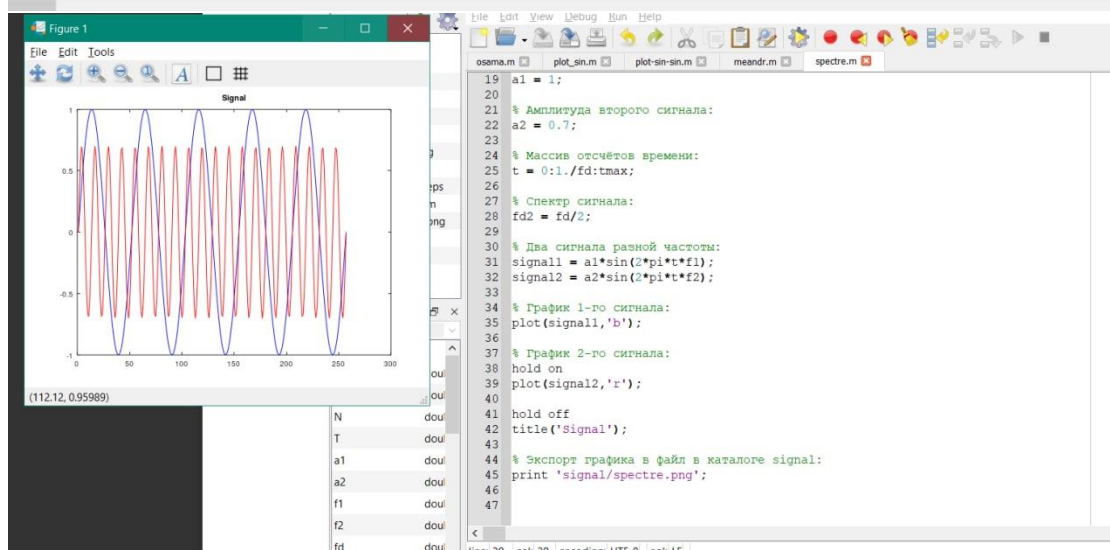
The image shows a MATLAB Editor window with a menu bar (File, Edit, View, Debug, Run, Help) and a toolbar. The toolbar includes icons for file operations (New, Open, Save, Print), editing (Undo, Redo, Cut, Copy, Paste), and execution (Run, Stop, Step Through). The workspace pane on the left shows the current script 'spectre.m' and other open files: 'osama.m', 'plot_sin.m', 'plot-sin-sin.m', 'meandr.m', and 'spectre.m'. The script content is as follows:

```
1 % spectre1/spectre.m
2 % Создание каталогов signal и spectre для размещения графиков:
3 mkdir 'signal';
4 mkdir 'spectre';
5
6 % Длина сигнала (с):
7 tmax = 0.5;
8
9 % Частота дискретизации (Гц) (количество отсчётов):
10 fd = 512;
11
12 % Частота первого сигнала (Гц):
13 f1 = 10;
14
15 % Частота второго сигнала (Гц):
16 f2 = 40;
17
18 % Амплитуда первого сигнала:
19 a1 = 1;
20
21 % Амплитуда второго сигнала:
22 a2 = 0.7;
23
24 % Массив отсчётов времени:
25 t = 0:1./fd:tmax;
26
27 % Спектр сигнала:
28 fd2 = fd/2;
29
30 % Два сигнала разной частоты:
```

```

29
30 % Два сигнала разной частоты:
31 signal1 = a1*sin(2*pi*t*f1);
32 signal2 = a2*sin(2*pi*t*f2);
33
34 % График 1-го сигнала:
35 plot(signal1,'b');
36
37 % График 2-го сигнала:
38 hold on
39 plot(signal2,'r');
40
41 hold off
42 title('Signal');
43
44 % Экспорт графика в файл в каталоге signal:
45 print 'signal/spectre.png';
46
47

```



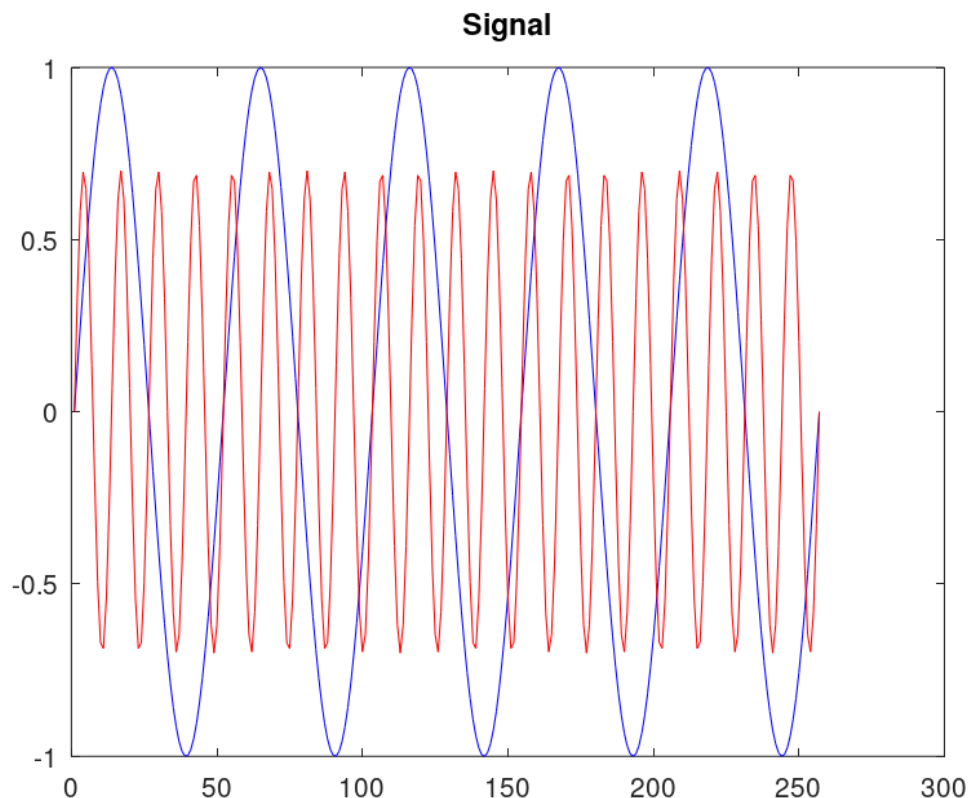
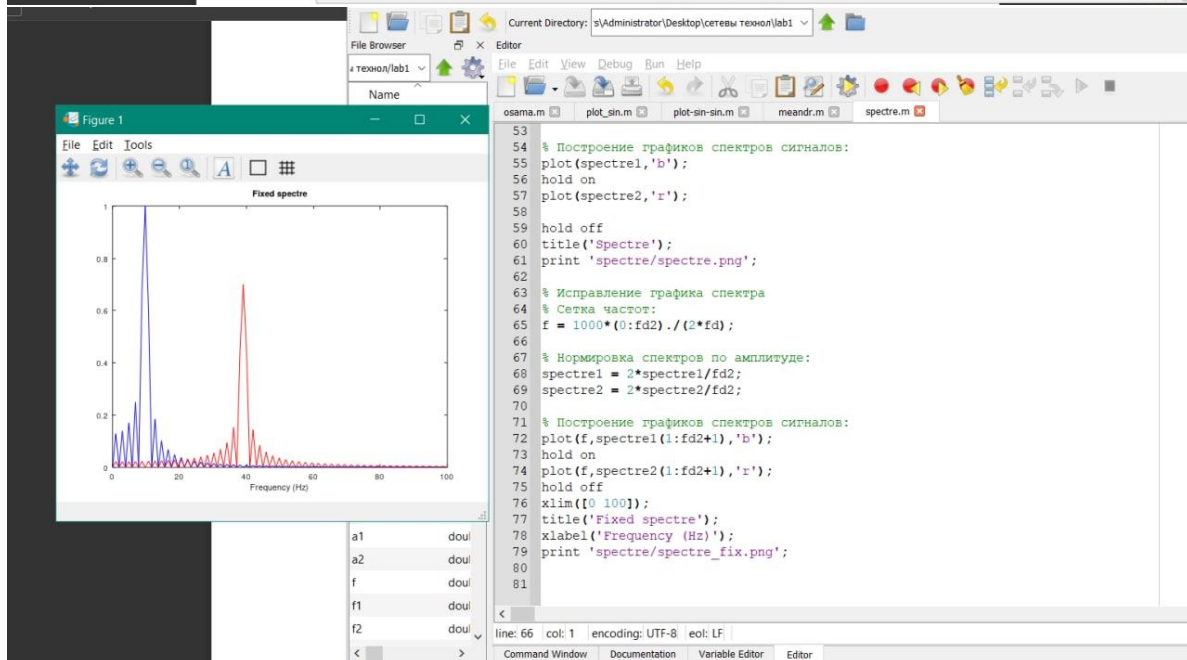
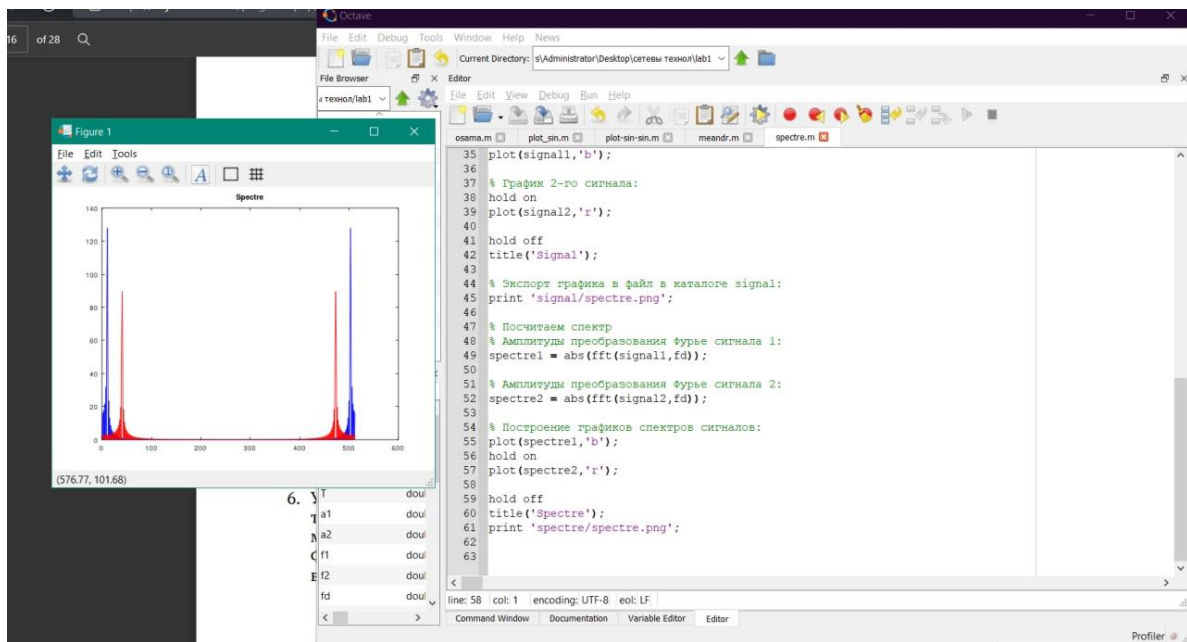


Рис. 1.4. Два синусоидальных сигнала разной частоты

5. С помощью быстрого преобразования Фурье найдите спектры сигналов (рис. 1.5), добавив в файл `spectre.m` следующий код:

6. Учитывая реализацию преобразования Фурье, скорректируйте график спектра (рис. 1.6): отбросьте дублирующие отрицательные частоты, а также примите в расчёт то, что на каждом шаге вычисления быстрого преобразования Фурье происходит суммирование амплитуд сигналов. Для этого добавьте в файл `spectre.m` следующий код:



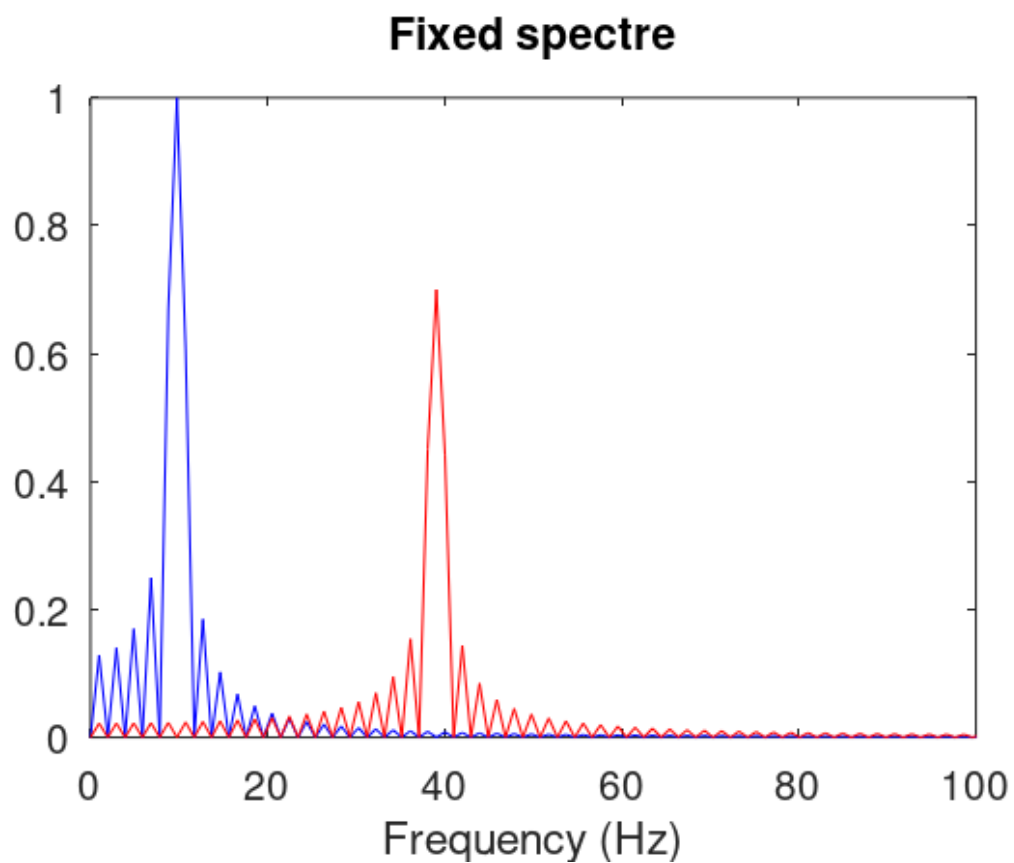


Рис. 1.6. Исправленный график спектров синусоидальных сигналов

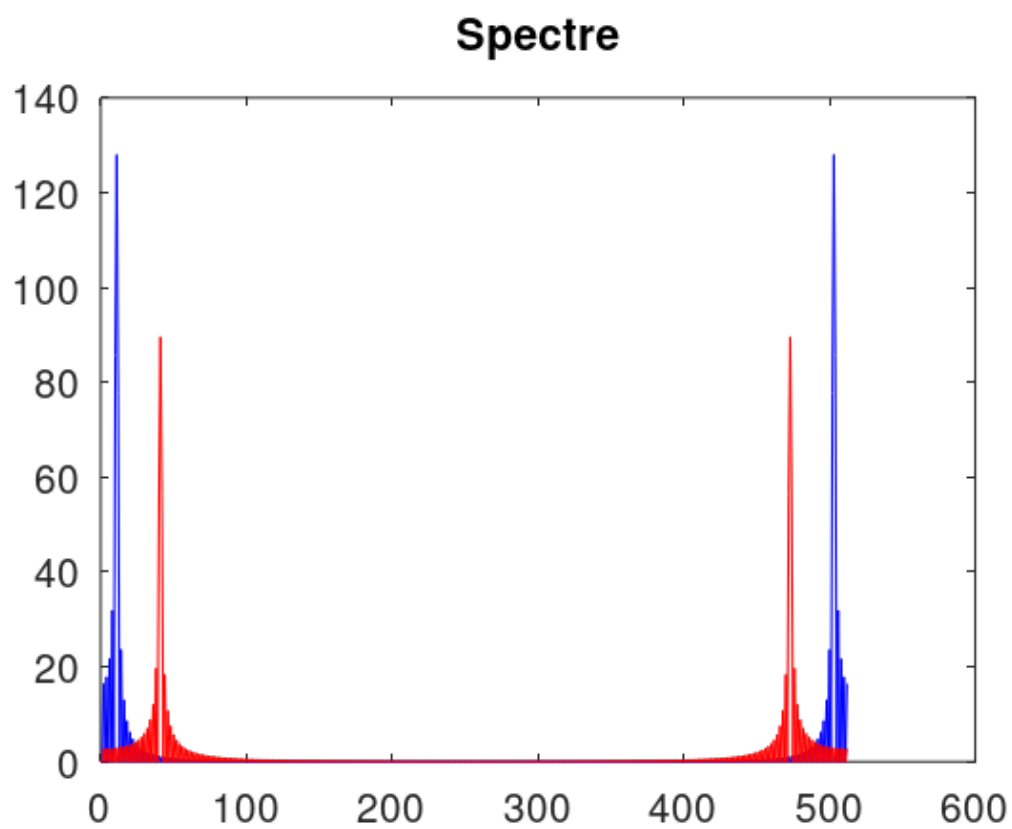
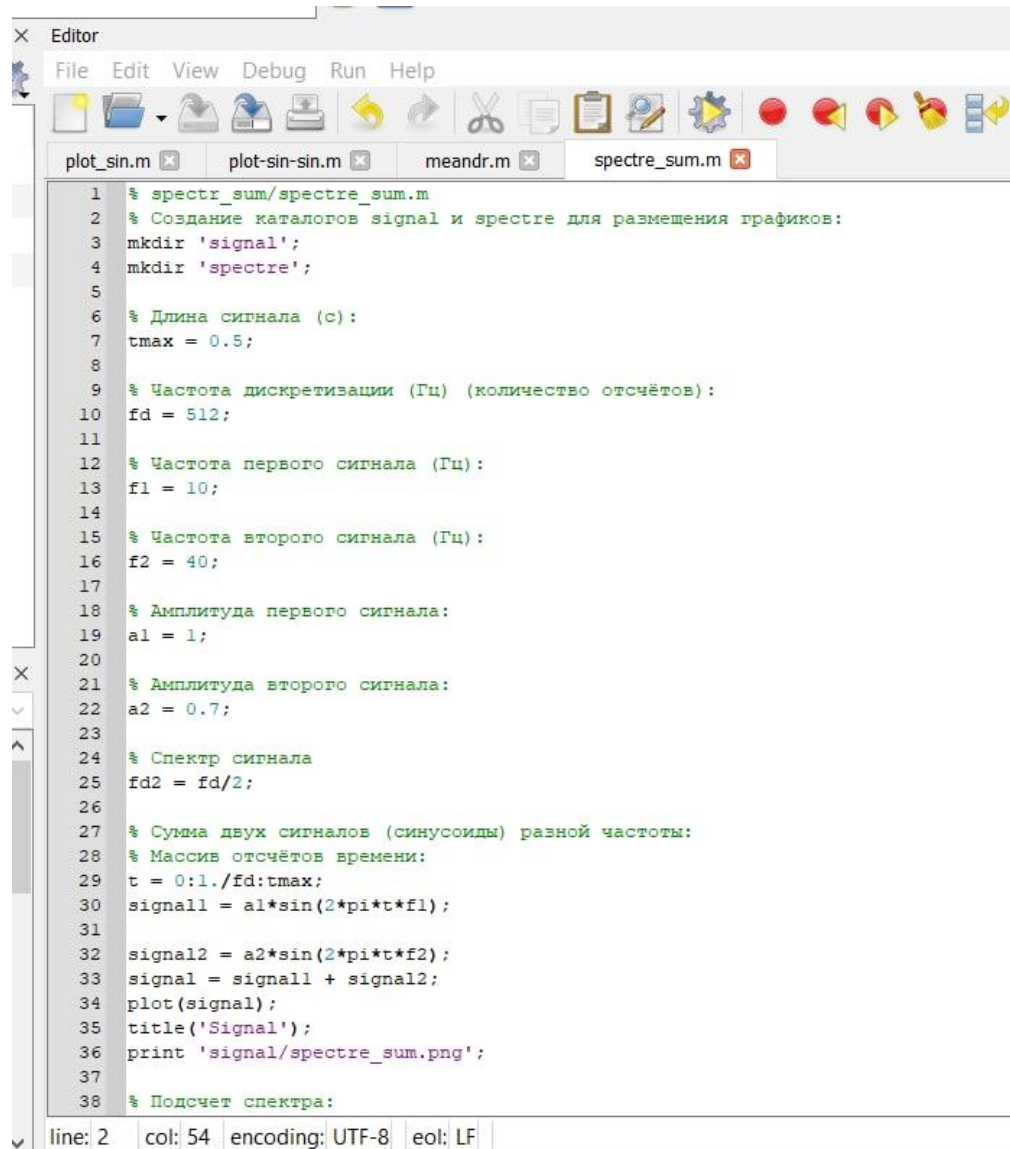


Рис. 1.5. График спектров синусоидальных сигналов

7. Найдите спектр суммы рассмотренных сигналов (рис. 1.7), создав каталог spectr_sum и файл в нём spectre_sum.m со следующим кодом:



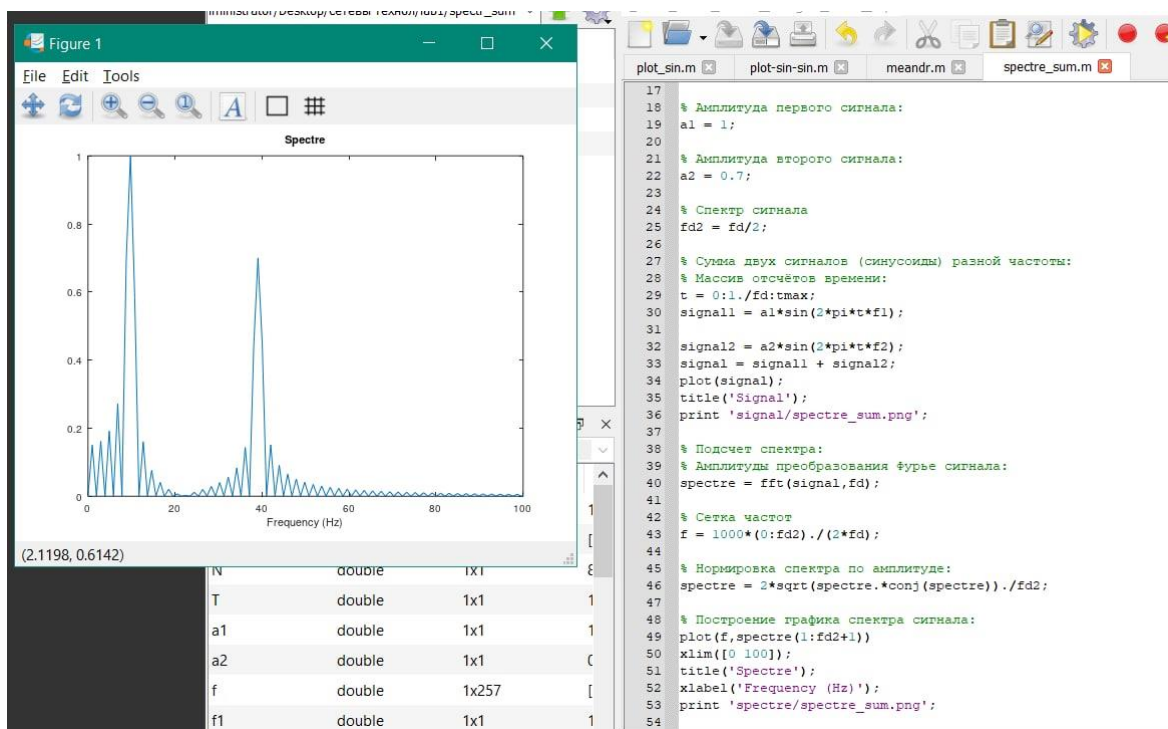
```
1 % spectr_sum/spectre_sum.m
2 % Создание каталогов signal и spectre для размещения графиков:
3 mkdir 'signal';
4 mkdir 'spectre';
5
6 % Длина сигнала (с):
7 tmax = 0.5;
8
9 % Частота дискретизации (Гц) (количество отсчётов):
10 fd = 512;
11
12 % Частота первого сигнала (Гц):
13 f1 = 10;
14
15 % Частота второго сигнала (Гц):
16 f2 = 40;
17
18 % Амплитуда первого сигнала:
19 a1 = 1;
20
21 % Амплитуда второго сигнала:
22 a2 = 0.7;
23
24 % Спектр сигнала
25 fd2 = fd/2;
26
27 % Сумма двух сигналов (синусоиды) разной частоты:
28 % Массив отсчётов времени:
29 t = 0:1./fd:tmax;
30 signall = a1*sin(2*pi*t*f1);
31
32 signal2 = a2*sin(2*pi*t*f2);
33 signal = signall + signal2;
34 plot(signal);
35 title('Signal');
36 print 'signal/spectre_sum.png';
37
38 % Подсчет спектра:
```

line: 2 col: 54 encoding: UTF-8 eol: LF


```

35 title('Signal');
36 print 'signal/spectre_sum.png';
37
38 % Подсчет спектра:
39 % Амплитуды преобразования Фурье сигнала:
40 spectre = fft(signal,fd);
41
42 % Сетка частот
43 f = 1000*(0:fd2)./(2*fd);
44
45 % Нормировка спектра по амплитуде:
46 spectre = 2*sqrt(spectre.*conj(spectre))./fd2;
47
48 % Построение графика спектра сигнала:
49 plot(f,spectre(1:fd2+1))
50 xlim([0 100]);
51 title('Spectre');
52 xlabel('Frequency (Hz)');
53 print 'spectre/spectre_sum.png';
54

```



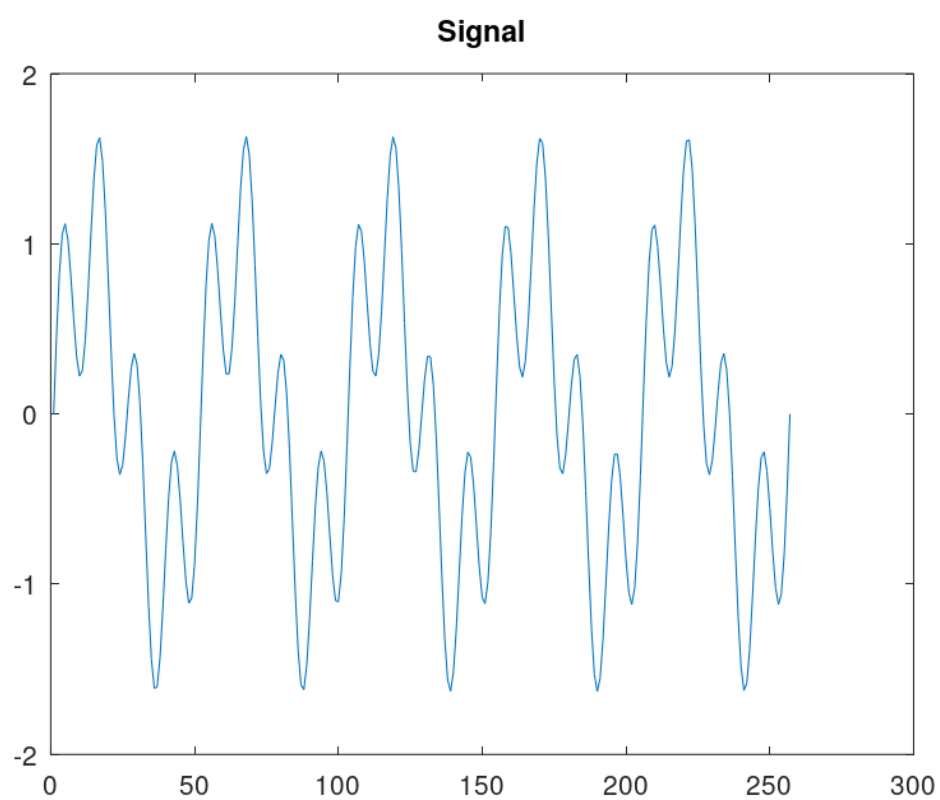


Рис. 1.7. Суммарный сигнал

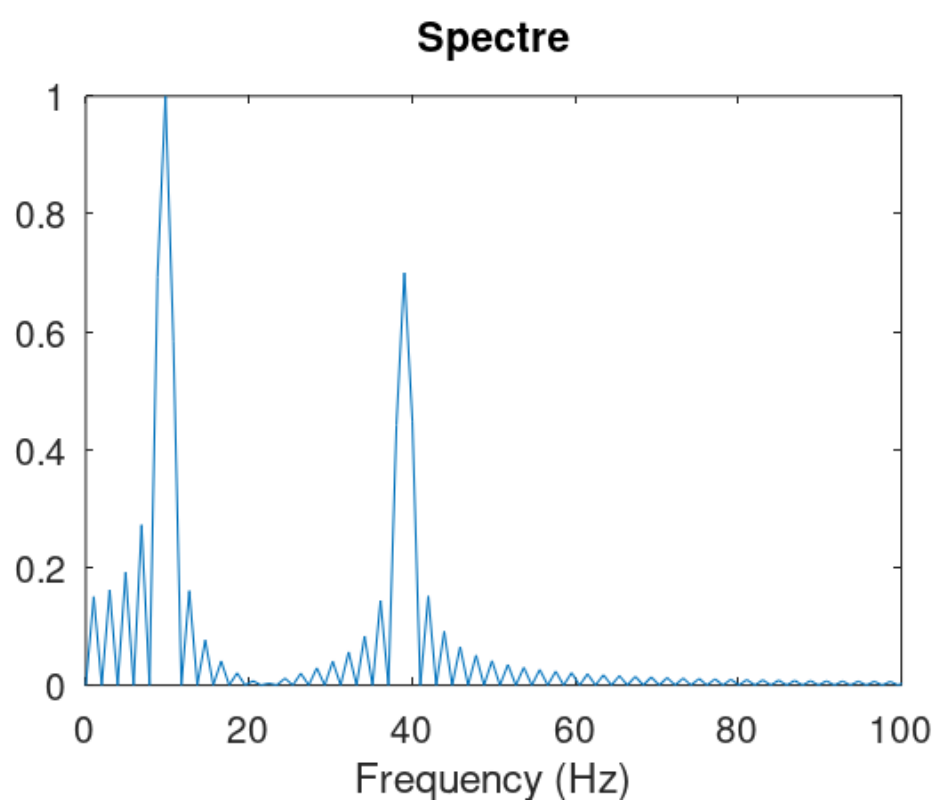


Рис. 1.8. Спектр суммарного сигнала

1.3.4. Амплитудная модуляция

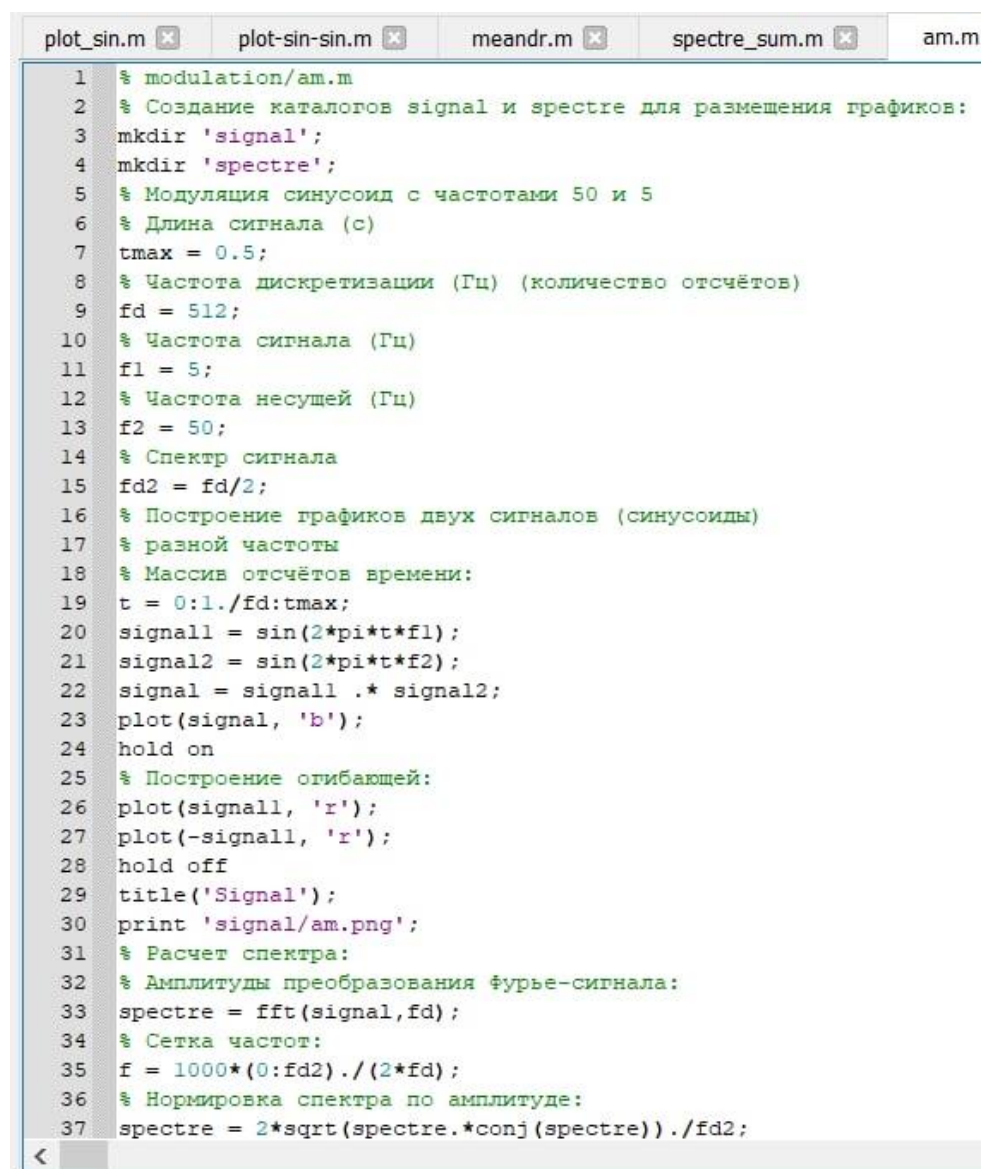
1.3.4.1. Постановка задачи

Продемонстрировать принципы модуляции сигнала на примере аналоговой амплитудной модуляции (рис.1.9).

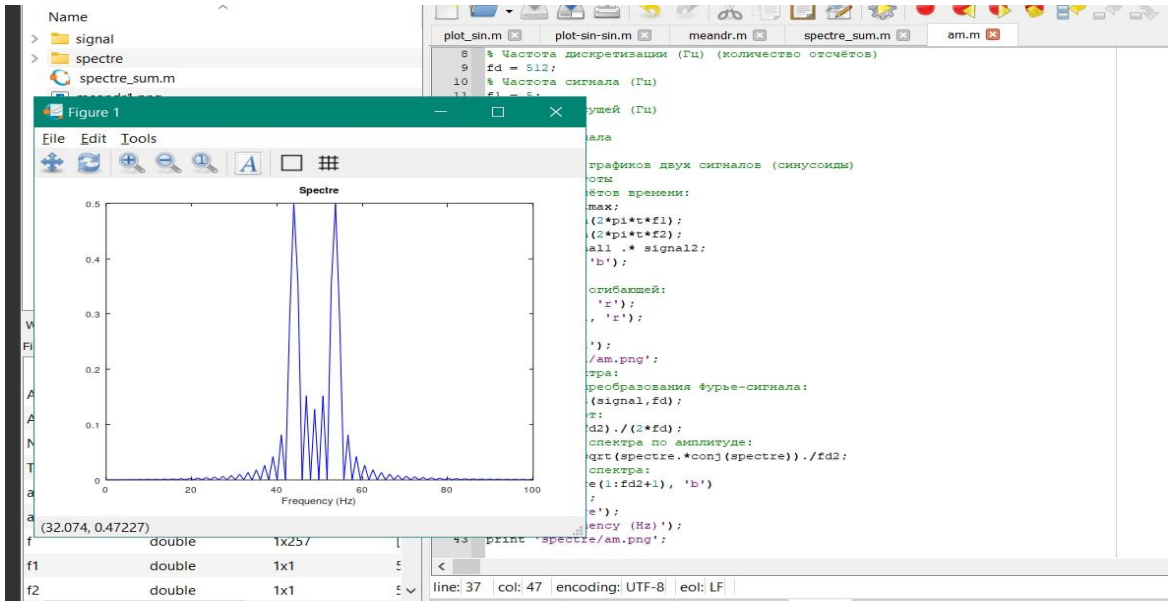
1.3.4.2. Порядок выполнения работы

1. В вашем рабочем каталоге создайте каталог modulation и в нём новый сценарий с именем am.m.

2. Добавьте в файле am.m следующий код:



```
plot_sin.m x plot-sin-sin.m x meandr.m x spectre_sum.m x 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';
31 % Расчет спектра:
32 % Амплитуды преобразования Фурье-сигнала:
33 spectre = fft(signal,fd);
34 % Сетка частот:
35 f = 1000*(0:fd2)/(2*fd);
36 % Нормировка спектра по амплитуде:
37 spectre = 2*sqrt(spectre.*conj(spectre))./fd2;
```



В результате получаем, что спектр произведения представляет собой свёртку спектров (рис. 1.10).

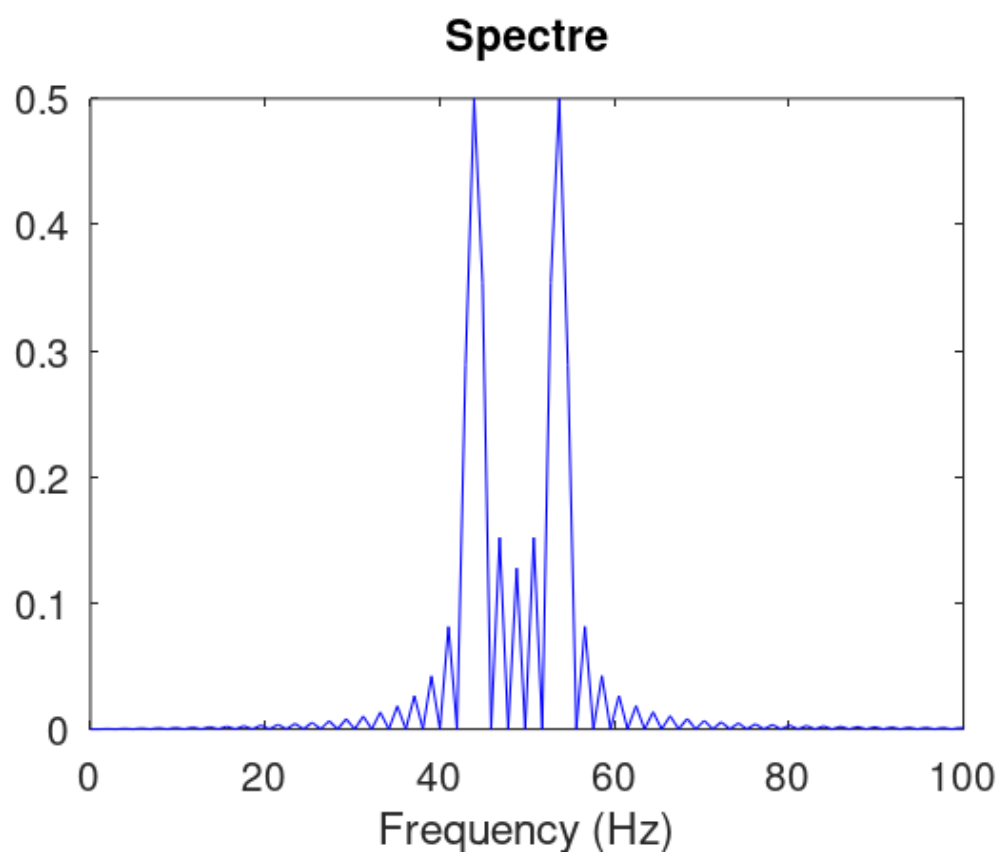


Рис. 1.10. Спектр сигнала при амплитудной модуляции

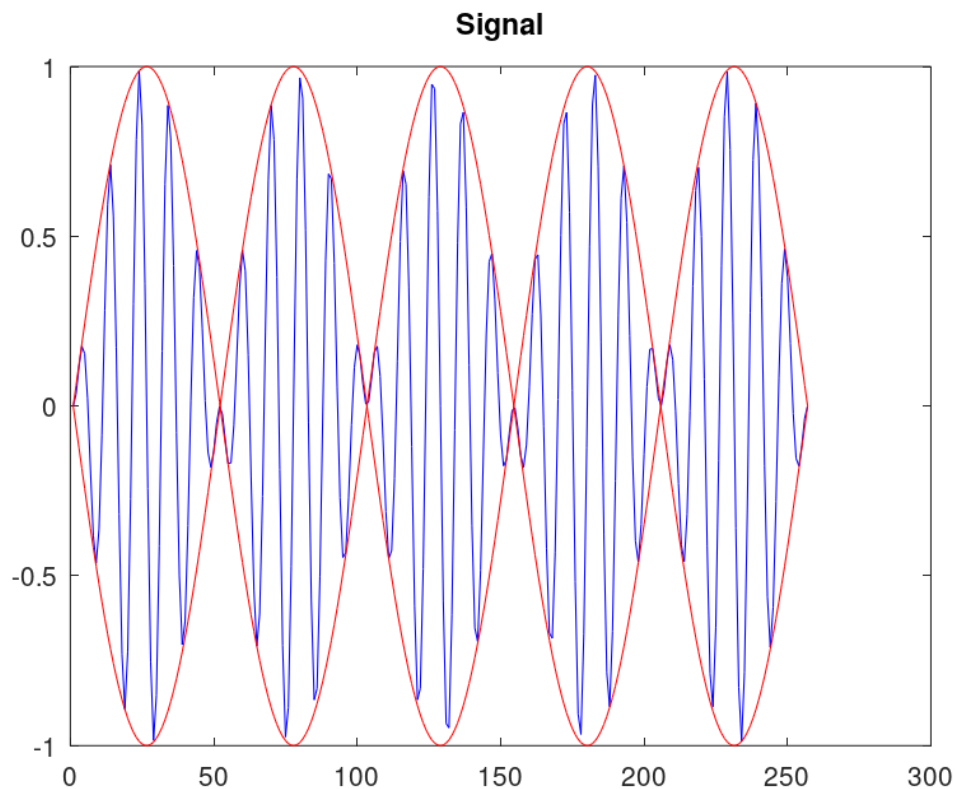


Рис. 1.9. Сигнал и огибающая при амплитудной модуляции

1.3.5. Кодирование сигнала. Исследование свойства самосинхронизации сигнала

1.3.5.1. Постановка задачи










По заданных битовых последовательностей требуется получить кодированные сигналы для нескольких кодов, проверить свойства самосинхронизуемости кодов, получить спектры.

1.3.5.2. Порядок выполнения работы

1. В вашем рабочем каталоге создайте 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. В окне интерпретатора команд проверьте, установлен ли у вас пакет расширений signal:

3. В файле main.m подключите пакет signal и задайте входные кодовые последовательности:



 calcspectre.m	9/7/2022 11:47 PM	M File	1 KB
 ami.m	9/7/2022 11:47 PM	M File	1 KB
 bipolarmrz.m	9/7/2022 11:47 PM	M File	1 KB
 bipolarrrz.m	9/7/2022 11:47 PM	M File	1 KB
 diffmanc.m	9/7/2022 11:47 PM	M File	1 KB
 main.m	9/7/2022 11:34 PM	M File	3 KB
 manchester.m	9/7/2022 11:47 PM	M File	1 KB
 maptowave.m	9/7/2022 11:47 PM	M File	1 KB
 unipolar.m	9/7/2022 11:47 PM	M File	1 KB


```
1 % coding/main.m
2 % Подключение пакета signal:
3 pkg load signal;
4
5 % Входная кодовая последовательность:
6 data=[0 1 0 0 1 1 0 0 0 1 1 0];
7
8 % Входная кодовая последовательность для проверки свойства самосинхронизации:
9 data_sync=[0 0 0 0 0 0 0 0 1 1 1 1 1 1];
10
11 % Входная кодовая последовательность для построения спектра сигнала:
12 data_spectre=[0 1 0 1 0 1 0 1 0 1 0 1 0 1];
13
14 % Создание каталогов signal, sync и spectre для размещения графиков:
15 mkdir 'signal';
16 mkdir 'sync';
17 mkdir 'spectre';
18 axis('auto');
19
20 % Униполярное кодирование
21 wave=unipolar(data);
22 plot(wave);
23 ylim([-1 6]);
24 title('Unipolar');
25 print 'signal/unipolar.png';
26
27 % Кодирование AMI
28 wave=ami(data);
29 plot(wave);
30 title('AMI');
31 print 'signal/ami.png';
32
33 % Кодирование NRZ
34 wave=bipolarnrz(data);
35 plot(wave);
36 title('Bipolar Non-Return to Zero');
37 print 'signal/bipolarnrz.png';
38
39 % Кодирование RZ
40 wave=bipolarrrz(data);
41 plot(wave);
42 title('Bipolar Return to Zero');
43 print 'signal/bipolarrrz.png';
44
45 % Манчестерское кодирование
46 wave=manchester(data);
47 plot(wave);
48 title('Manchester');
49 print 'signal/manchester.png';
50
51 % Дифференциальное манчестерское кодирование
52 wave=diffmanc(data);
53 plot(wave);
54 title('Differential Manchester');
55 print 'signal/diffmanc.png';
56
57 % Униполярное кодирование:
58 wave=unipolar(data_spectre);
59 spectre=calcspectre(wave);
60 title('Unipolar');
61 print 'spectre/unipolar.png';
62
63 % Кодирование AMI:
64 wave=ami(data_spectre);
65 spectre=calcspectre(wave);
66 title('AMI');
67 print 'spectre/ami.png';
68
69 % Кодирование NRZ:
70 wave=bipolarnrz(data_spectre);
71 spectre=calcspectre(wave);
72 title('Bipolar Non-Return to Zero');
73 print 'spectre/bipolarnrz.png';
74
75 % Кодирование RZ:
76 wave=bipolarrrz(data_spectre);
77 spectre=calcspectre(wave);
78 title('Bipolar Return to Zero');
79 print 'spectre/bipolarrrz.png';
80
81 % Манчестерское кодирование:
82 wave=manchester(data_spectre);
83 spectre=calcspectre(wave);
```

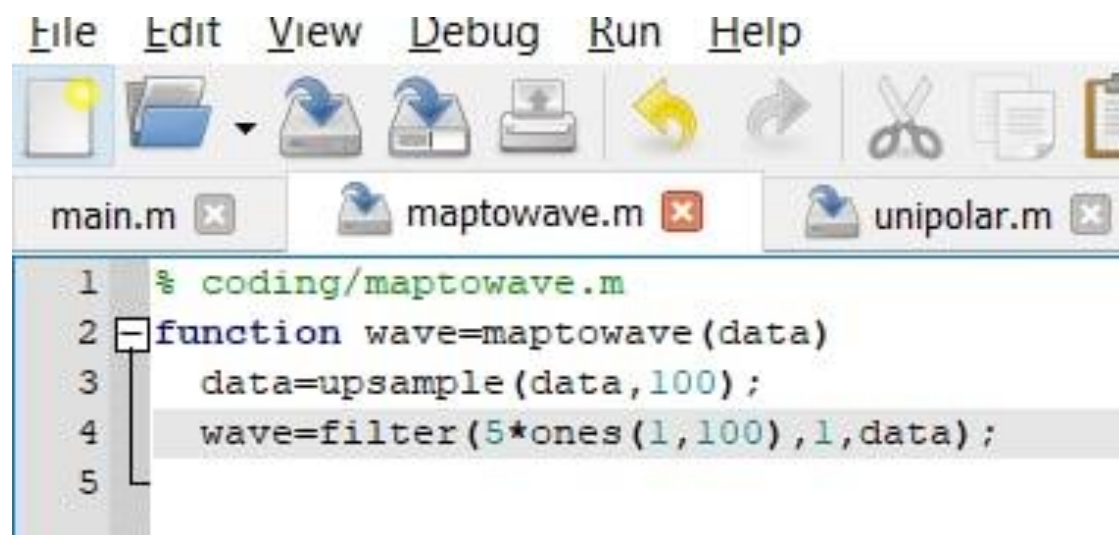
```

69 % Кодирование NRZ:
70 wave=bipolarnrz(data_spectre);
71 spectre=calcspectre(wave);
72 title('Bipolar Non-Return to Zero');
73 print 'spectre/bipolarnrz.png';
74
75 % Кодирование RZ:
76 wave=bipolarrz(data_spectre);
77 spectre=calcspectre(wave);
78 title('Bipolar Return to Zero');
79 print 'spectre/bipolarrz.png';
80
81 % Манчестерское кодирование:
82 wave=manchester(data_spectre);
83 spectre=calcspectre(wave);
84 title('Manchester');
85 print 'spectre/manchester.png';
86
87 % Дифференциальное манчестерское кодирование:
88 wave=diffmanc(data_spectre);
89 spectre=calcspectre(wave);
90 title('Differential Manchester');
91 print 'spectre/diffmanc.png';
92

```

line: 43 col: 30 encoding: UTF-8 eol: LF

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



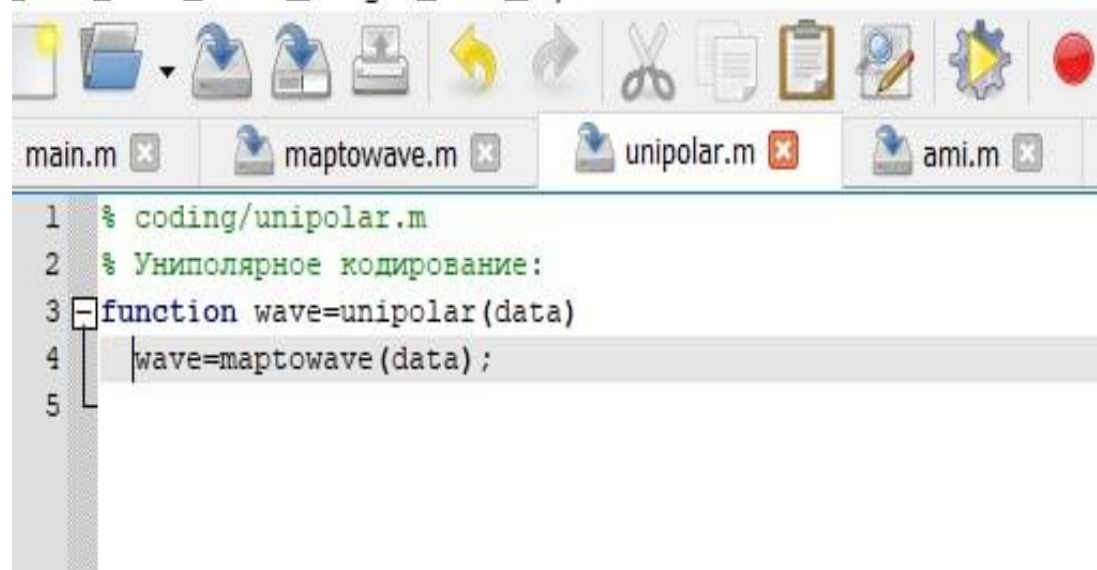
```

File Edit View Debug Run Help
[Icons]
main.m x maptowave.m x unipolar.m x
1 % coding/maptowave.m
2 function wave=maptowave(data)
3     data=upsample(data,100);
4     wave=filter(5*ones(1,100),1,data);
5

```

5. В файлах unipolar.m, ami.m, bipolarnrz.m, bipolarrrz.m, manchester.m, diffmanc.m пропишите соответствующие функции преобразования кодовой последовательности data с вызовом функции maptowave для построения соответствующего графика. Униполярное кодирование:

File Edit View Debug Run Help



The image shows a MATLAB editor window with the file 'unipolar.m' selected. The menu bar includes File, Edit, View, Debug, Run, and Help. The toolbar contains icons for file operations, editing, and execution. The file explorer at the top shows four files: main.m, maptowave.m, unipolar.m (active), and ami.m. The code in unipolar.m is as follows:

```
1 % coding/unipolar.m
2 % Униполярное кодирование:
3 function wave=unipolar(data)
4     wave=maptowave(data);
5
```

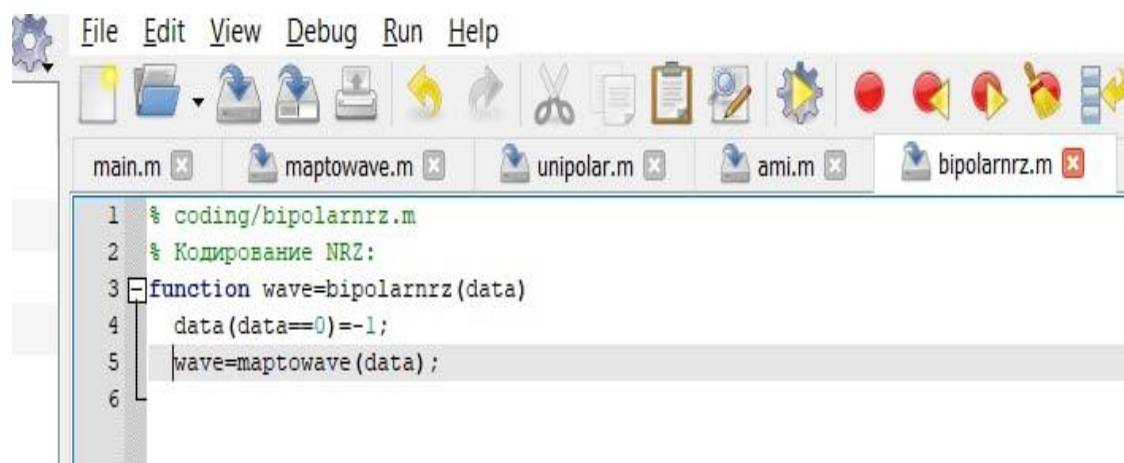
Кодирование AMI:



The image shows a MATLAB editor window with the file 'ami.m' selected. The menu bar includes File, Edit, View, Debug, Run, and Help. The toolbar contains icons for file operations, editing, and execution. The file explorer at the top shows five files: main.m, maptowave.m, unipolar.m, ami.m (active), and bi. The code in ami.m is as follows:

```
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);
8
```

Кодирование NRZ:



The image shows a MATLAB editor window with the file 'bipolarnrz.m' selected. The menu bar includes File, Edit, View, Debug, Run, and Help. The toolbar contains icons for file operations, editing, and execution. The file explorer at the top shows five files: main.m, maptowave.m, unipolar.m, ami.m, and bipolarnrz.m (active). The code in bipolarnrz.m is as follows:

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

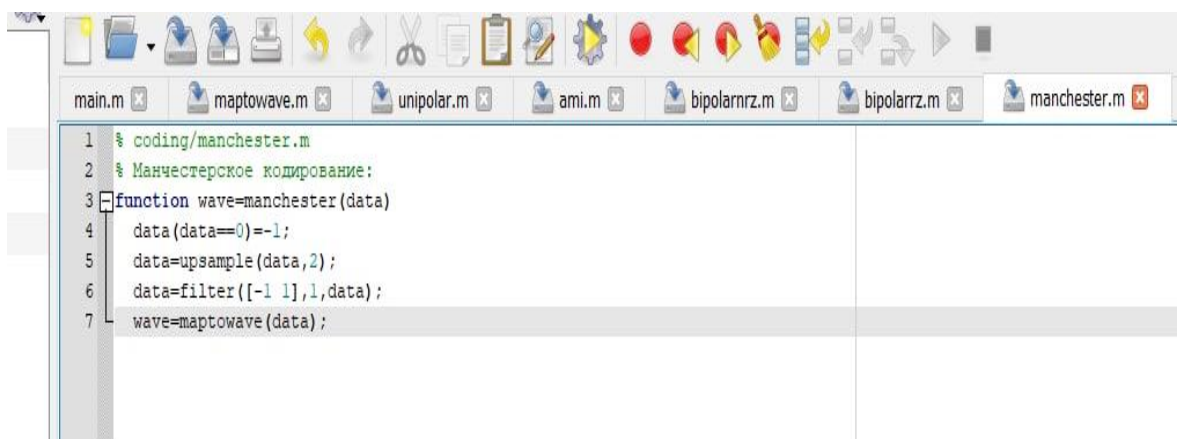
Кодирование RZ:



The image shows a MATLAB editor window with several tabs: main.m, maptowave.m, unipolar.m, ami.m, bipolarrrz.m, and bipolarrrz.m (highlighted). The code in the highlighted tab is as follows:

```
1 % coding/bipolarrrz.m
2 % Кодирование RZ:
3 function wave=bipolarrrz(data)
4     data(data==0)=-1;
5     data=upsample(data,2);
6     wave=maptowave(data);
7
8
```

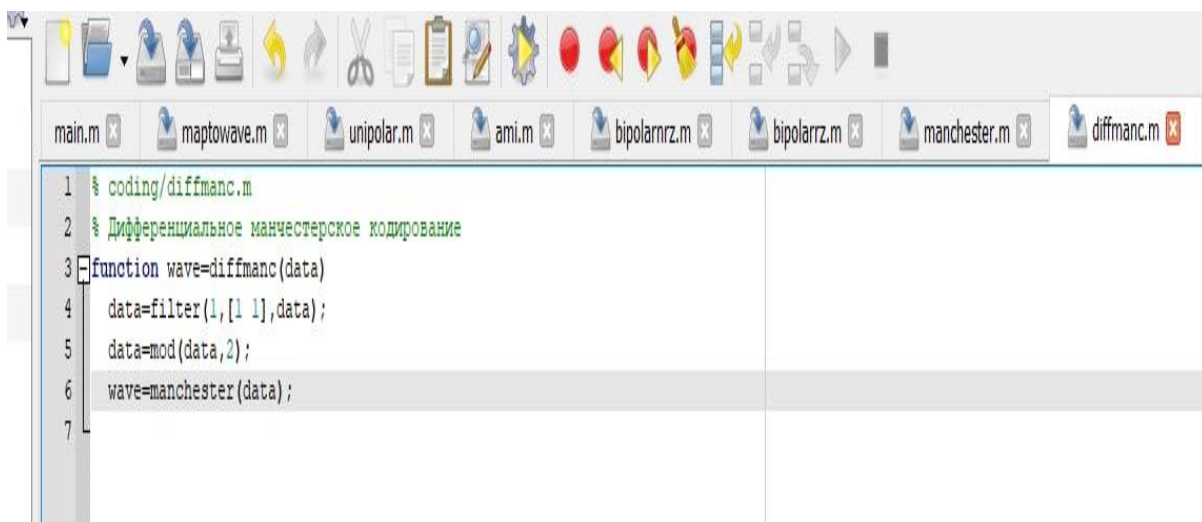
Манчестерское кодирование:



The image shows a MATLAB editor window with tabs: main.m, maptowave.m, unipolar.m, ami.m, bipolarrrz.m, bipolarrrz.m, and manchester.m (highlighted). The code in the highlighted tab is as follows:

```
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);
```

Дифференциальное манчестерское кодирование:

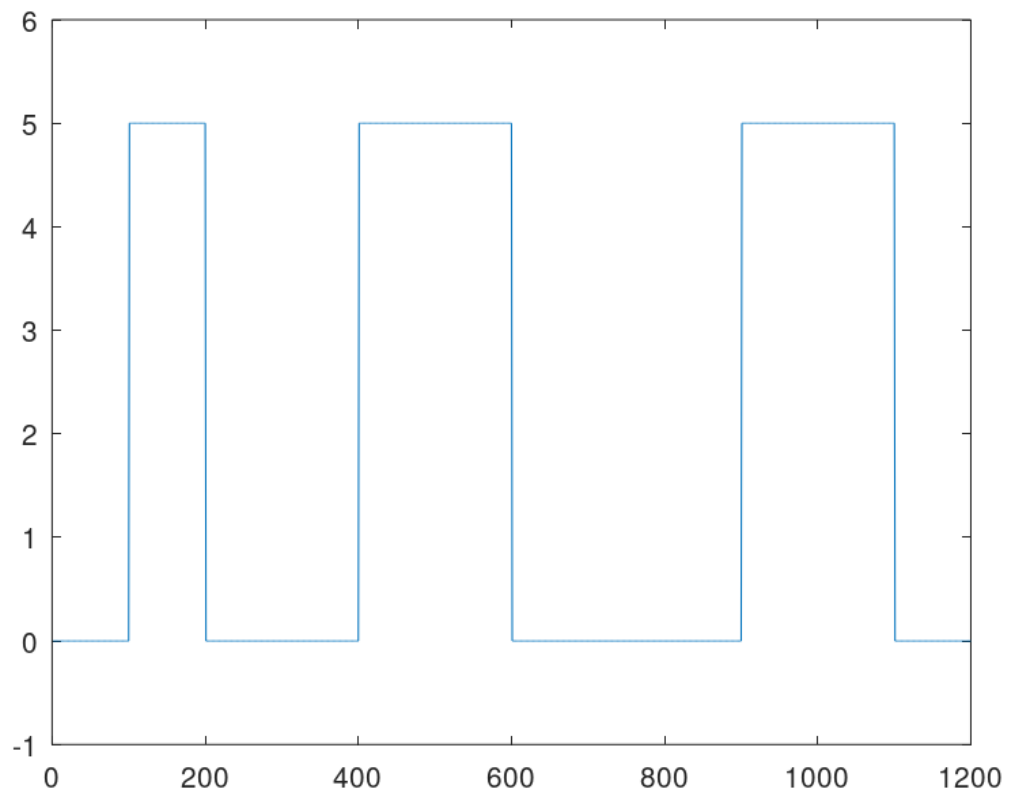


The image shows a MATLAB editor window with tabs: main.m, maptowave.m, unipolar.m, ami.m, bipolarrrz.m, bipolarrrz.m, manchester.m, and diffmanc.m (highlighted). The code in the highlighted tab is as follows:

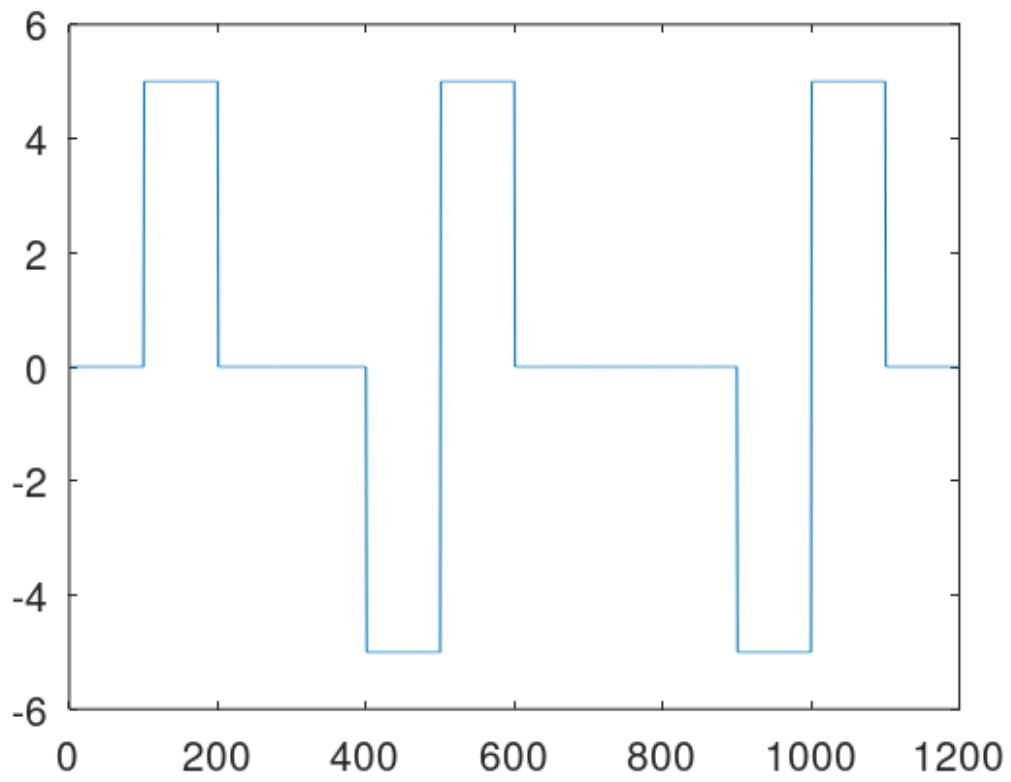
```
1 % coding/diffmanc.m
2 % Дифференциальное манчестерское кодирование
3 function wave=diffmanc(data)
4     data=filter(1,[1 1],data);
5     data=mod(data,2);
6     wave=manchester(data);
7
```

6. В файле calcspectre.m пропишите функцию построения спектра сигнала:

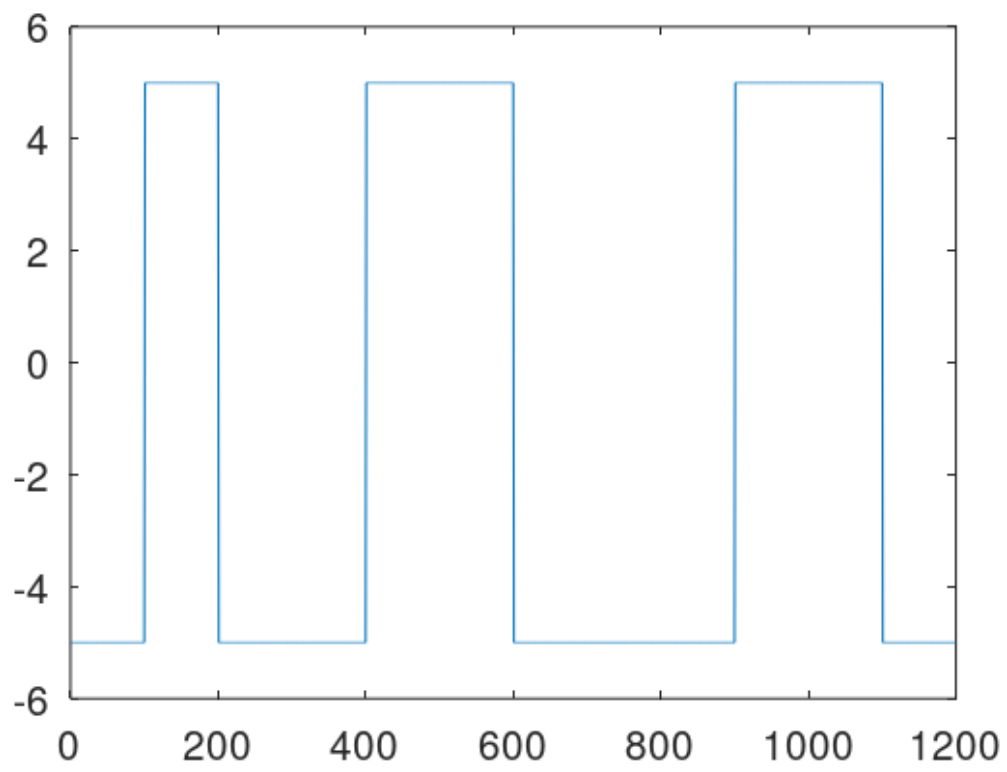
Unipolar



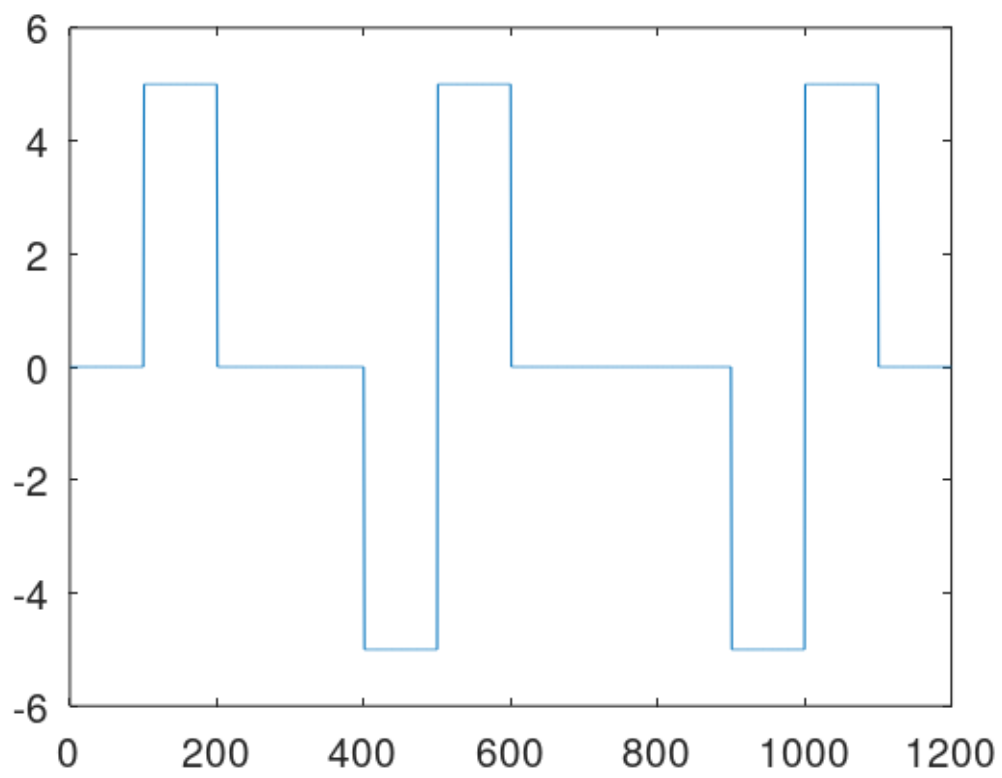
AMI



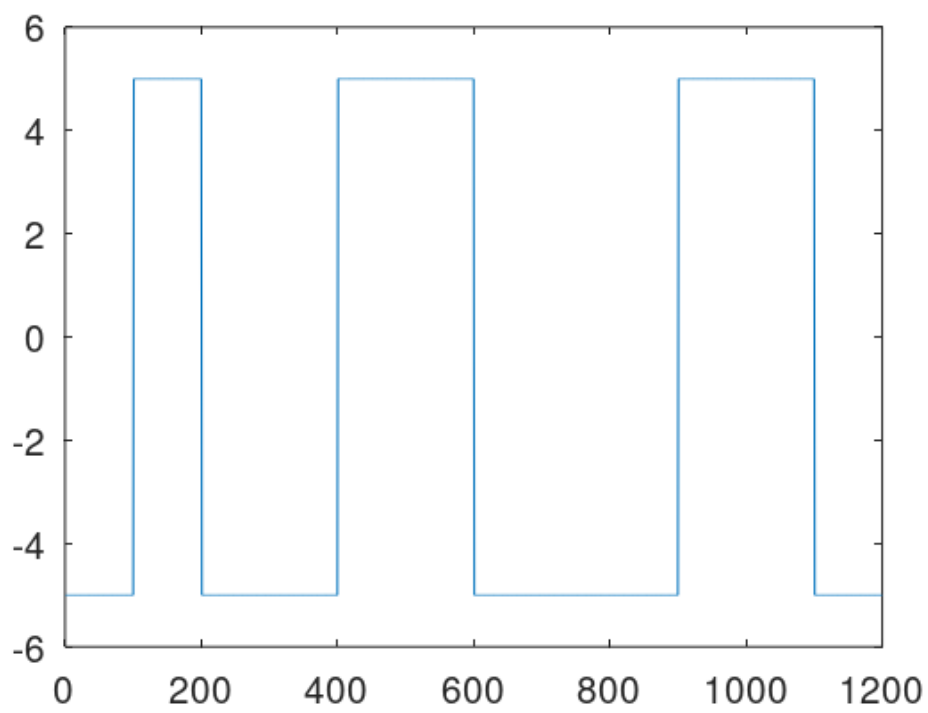
Bipolar Non-Return to Zero



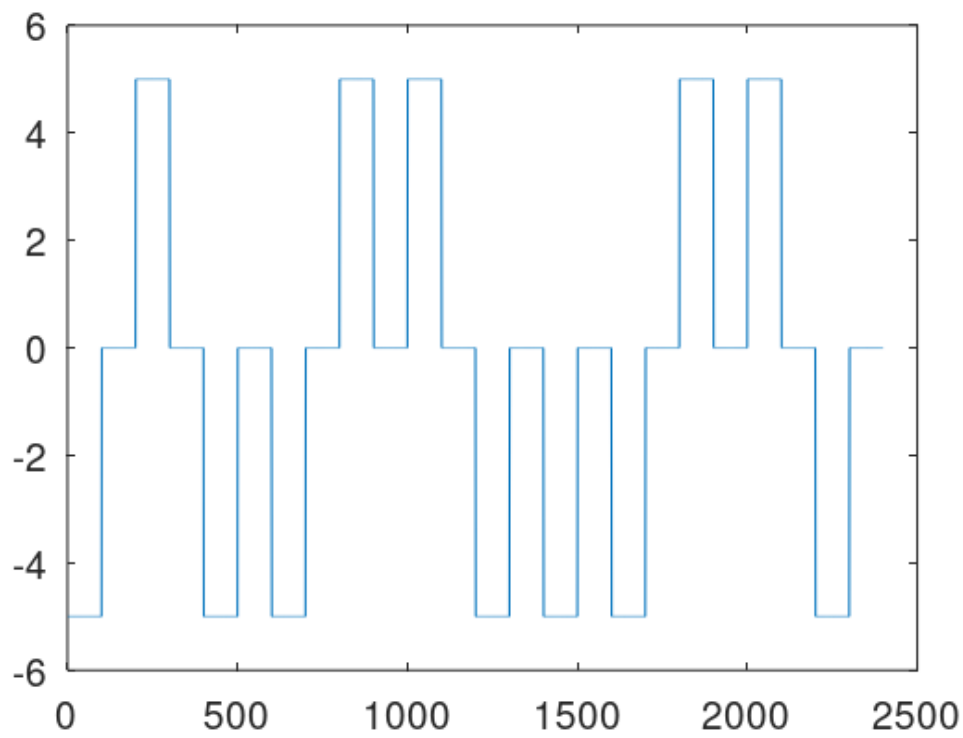
AMI



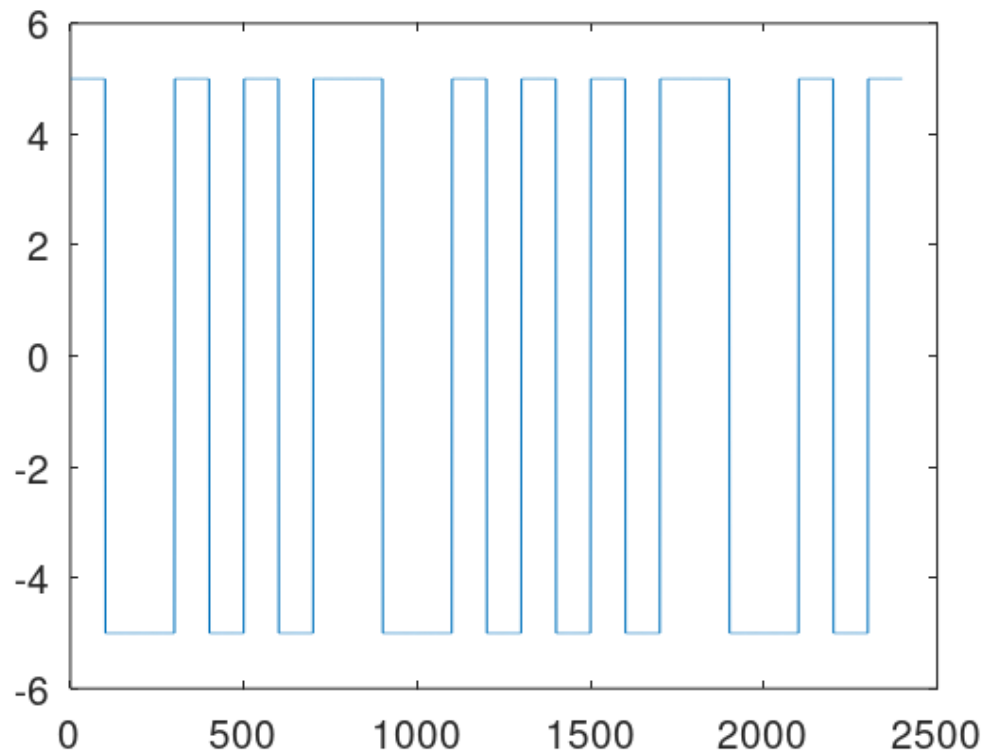
Bipolar Non-Return to Zero



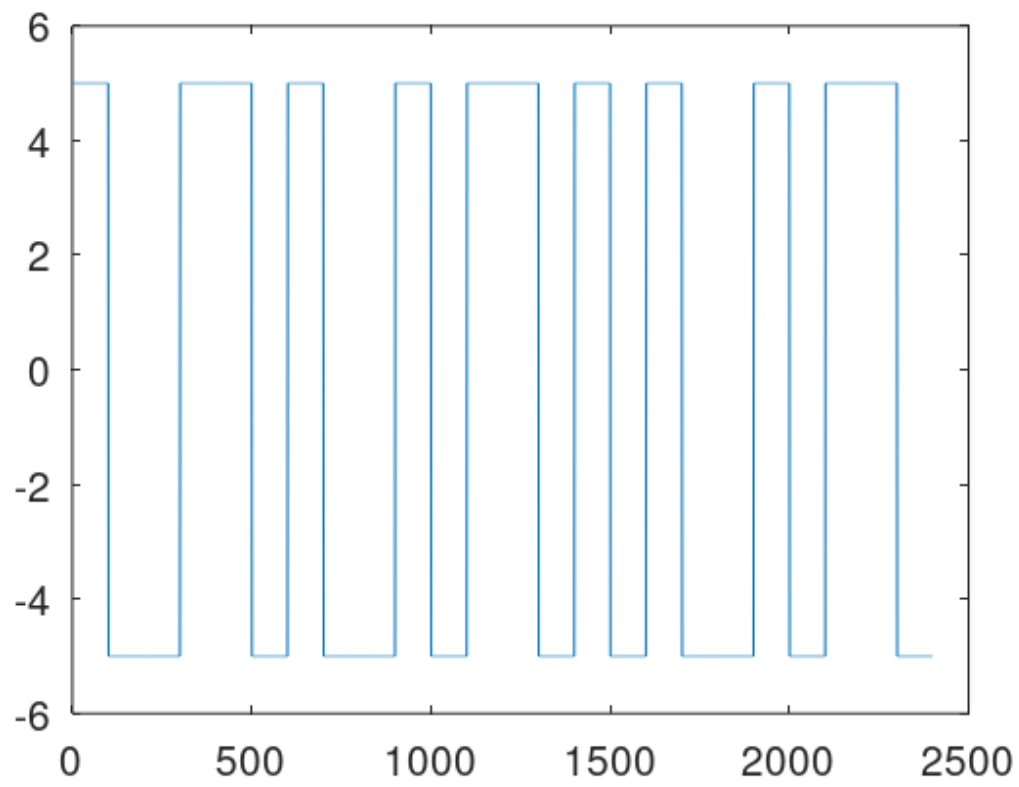
Bipolar Return to Zero



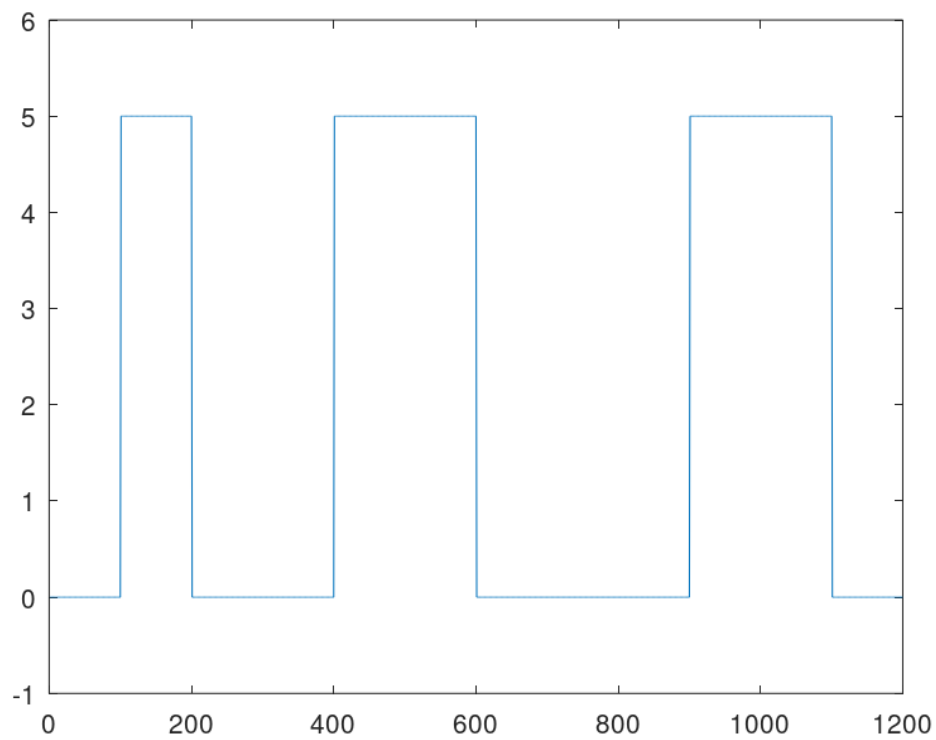
Differential Manchester



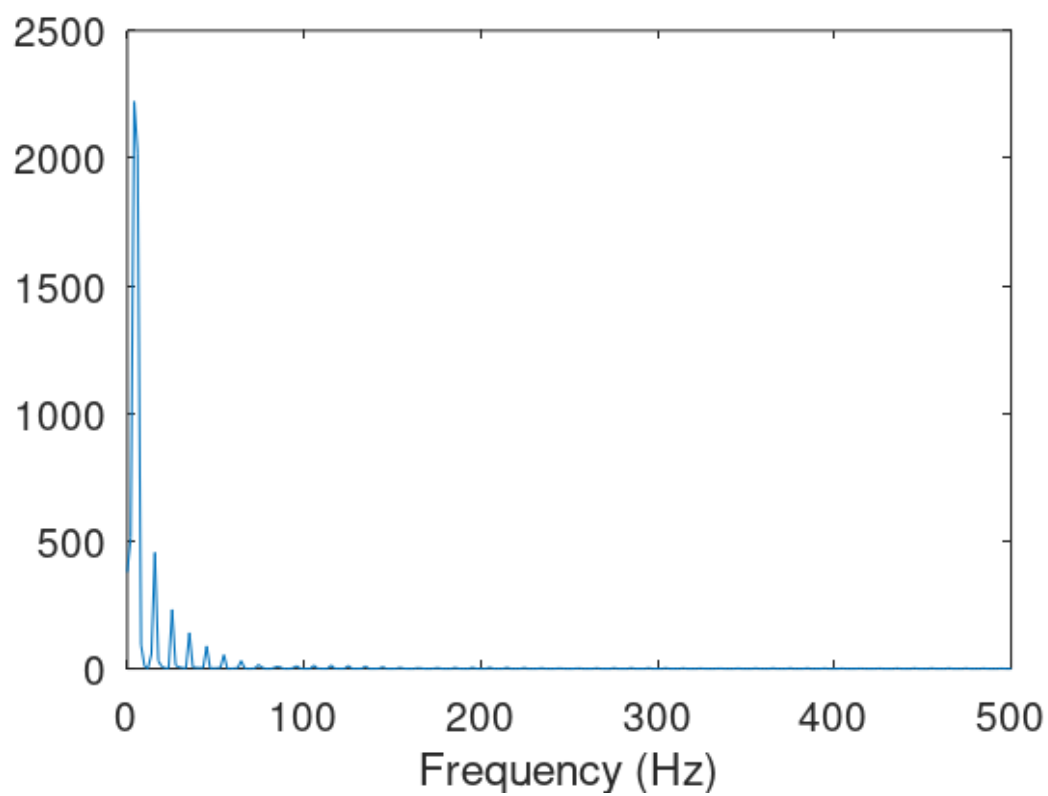
Manchester



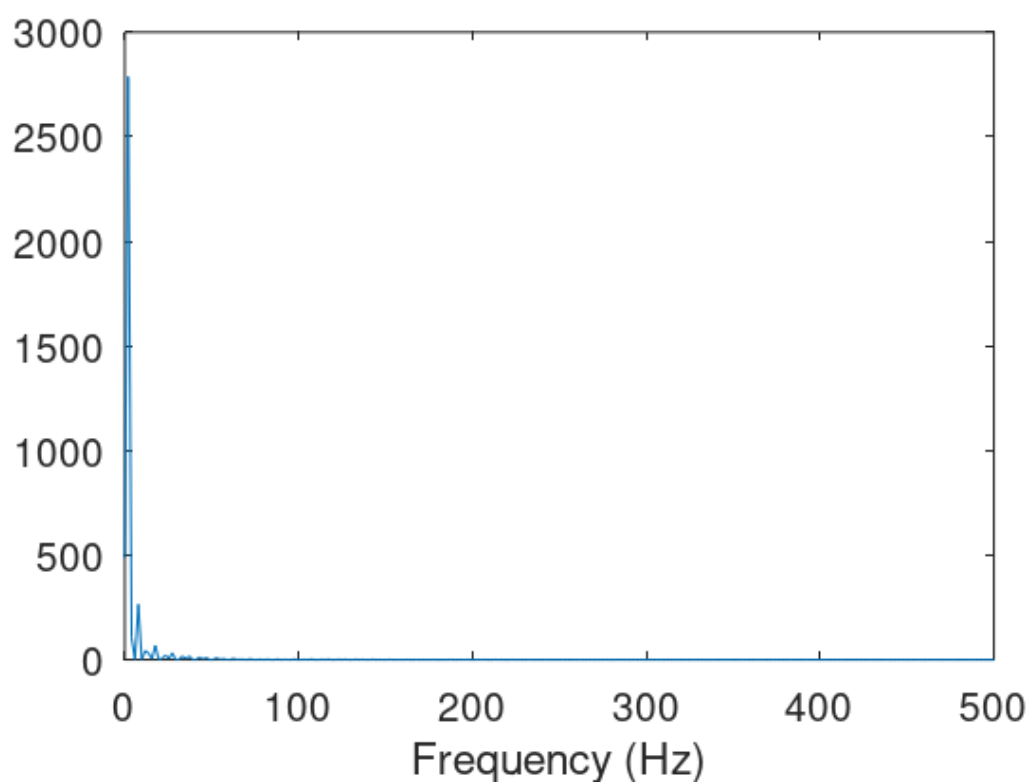
Unipolar



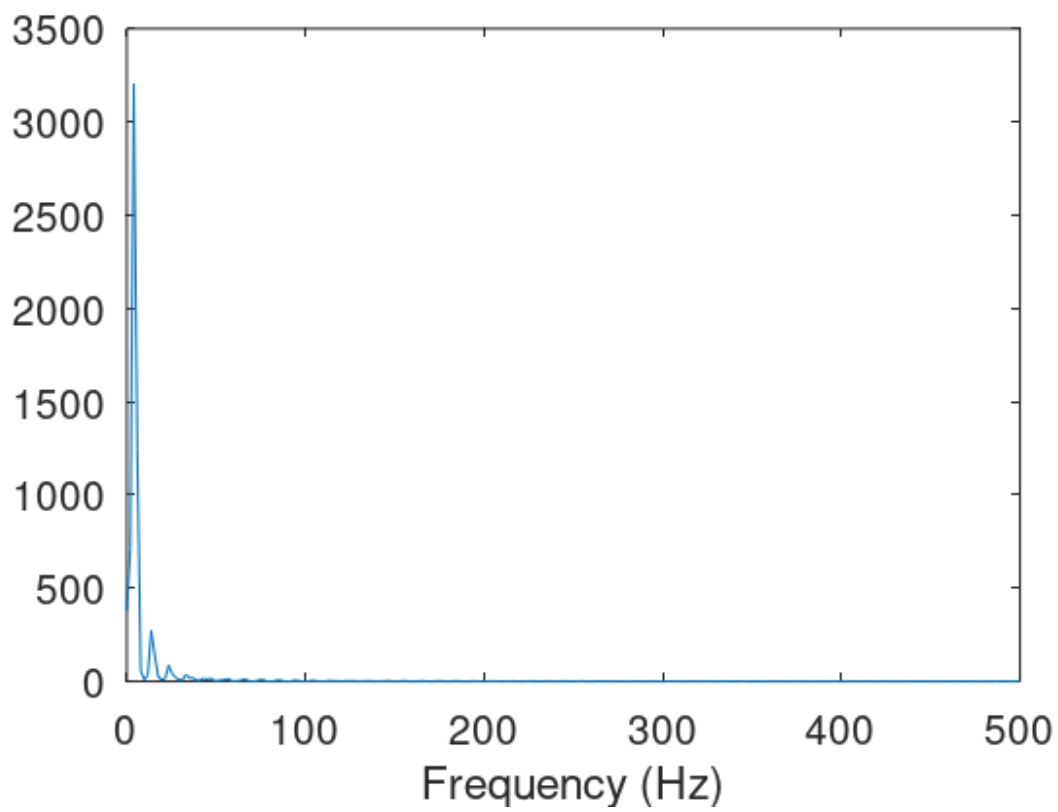
Bipolar Non-Return to Zero



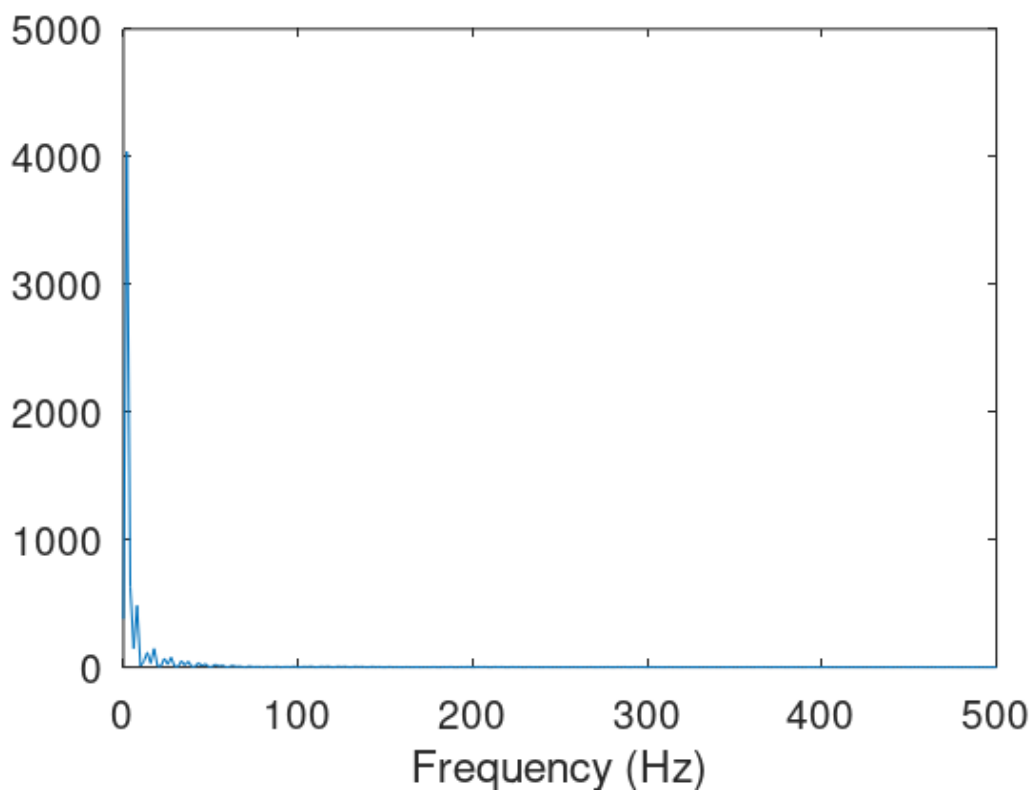
Bipolar Return to Zero



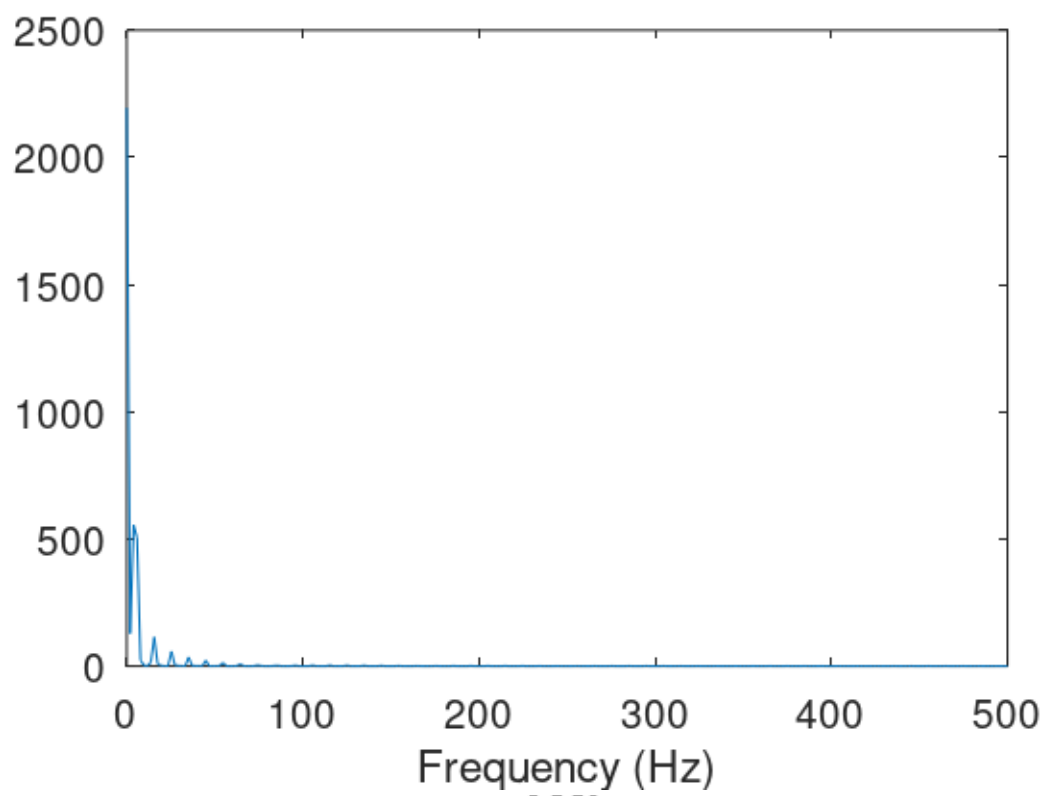
Differential Manchester



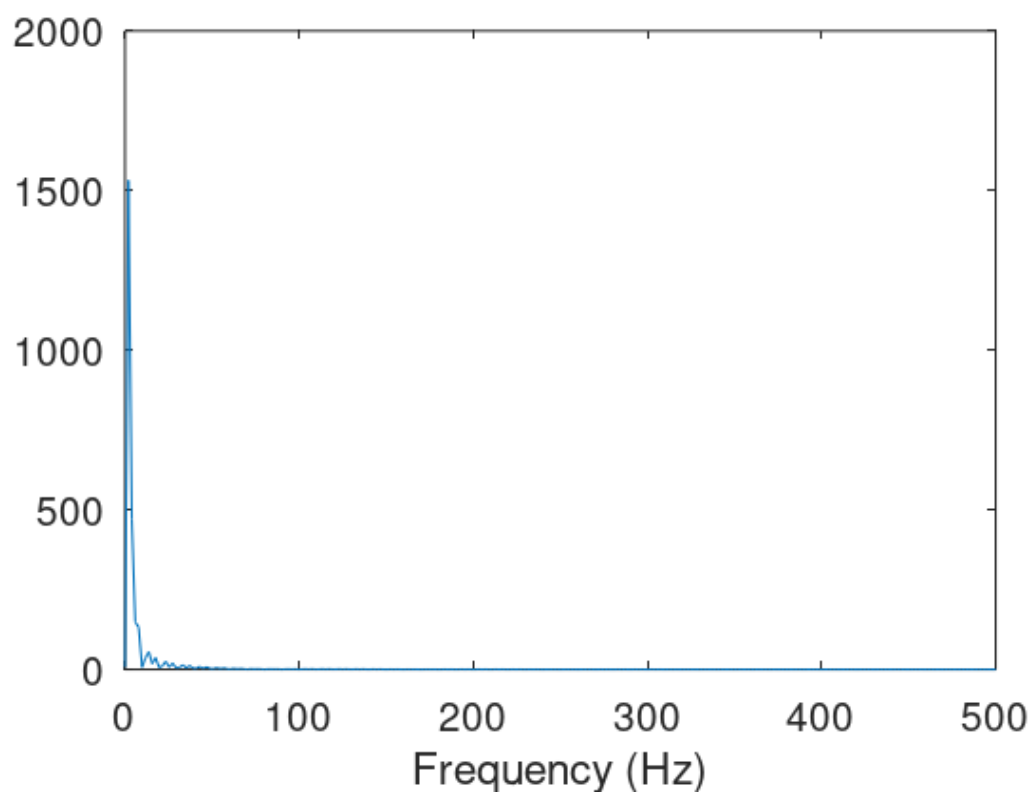
Manchester

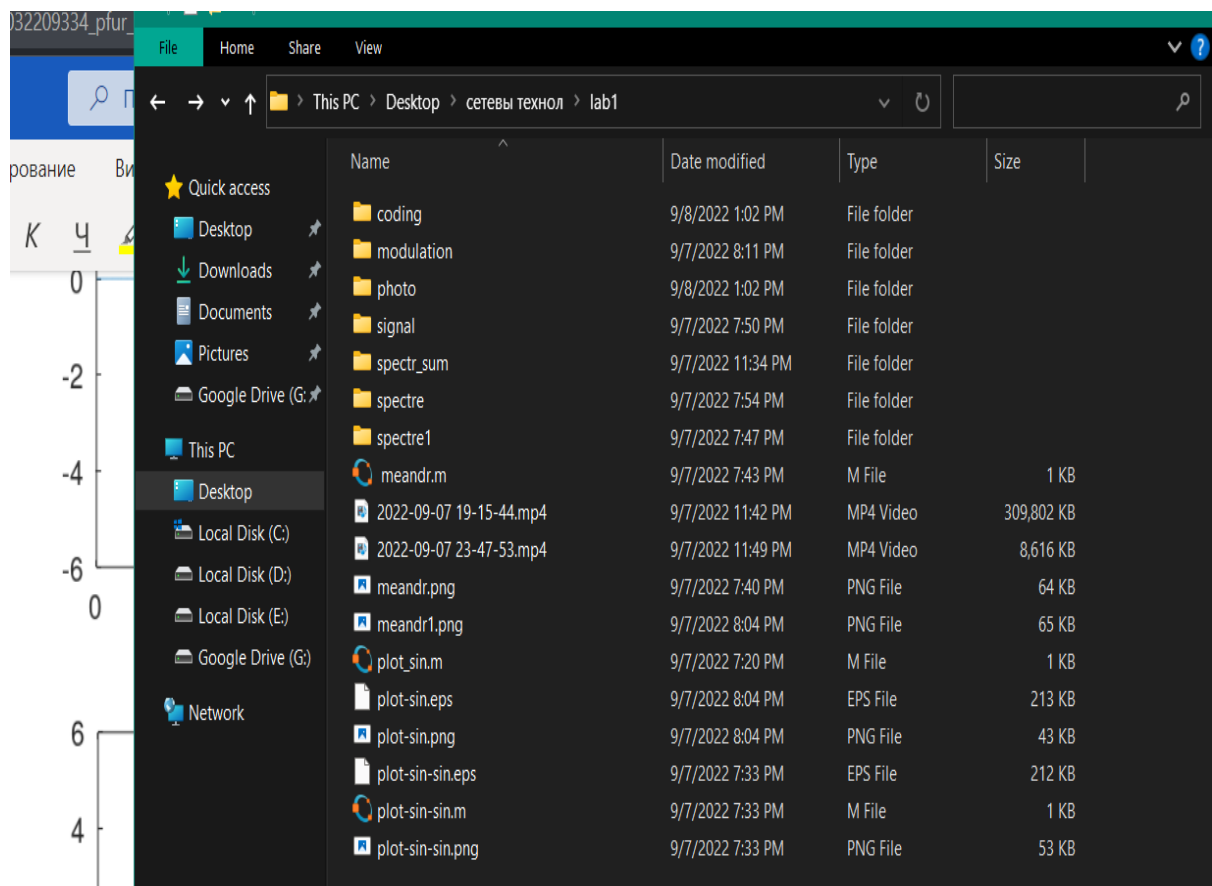


Unipolar



AMI





Вывод

Было действительно интересно узнать об Октаве. Теперь я знаю, как продемонстрировать конкретные графики с помощью Октавы. Задания не были сложными, вам просто нужно внимательно прочитать инструкцию