

**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: Саргсян Арам Грачьевич

Группа: НПИбд 02-20

**МОСКВА**

2022 г.

## ЦЕЛЬ 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.

## ХОД РАБОТЫ

1. Запустил в моей ОС Octave с оконным интерфейсом. Перейдите в окно редактора. Открыл новый сценарий, сохранил его в рабочий каталог с именем plot\_sin.m. В окне редактора добавил листинг из файла. Запустите сценарий на выполнение. (Рис. 1-2)

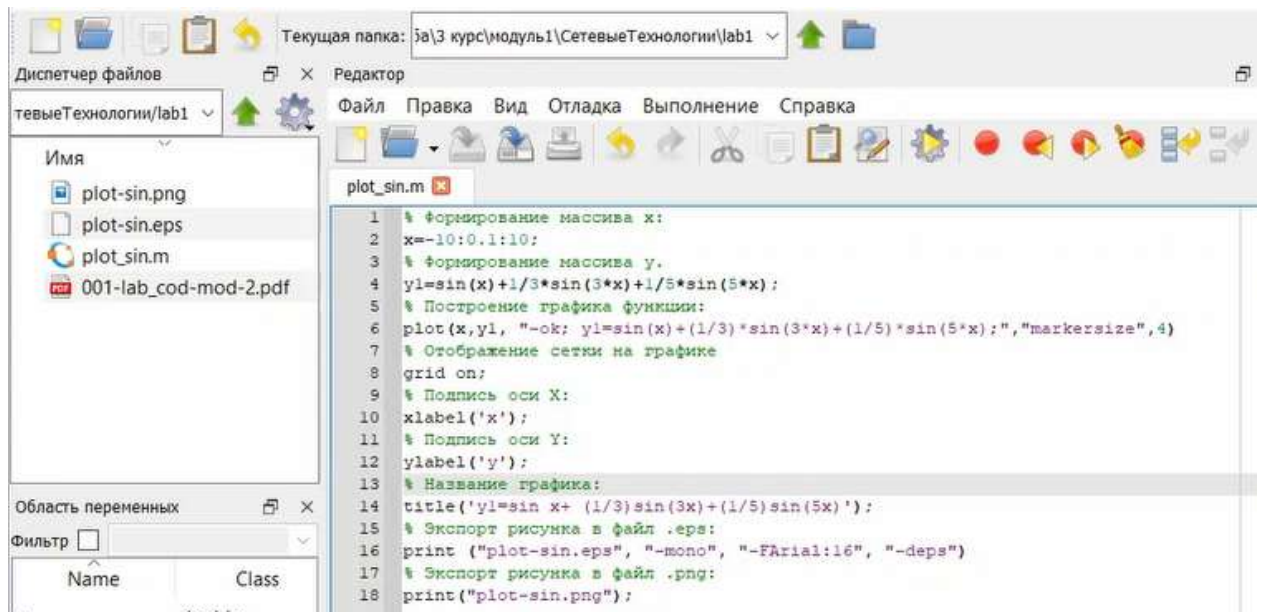


Рис. 1

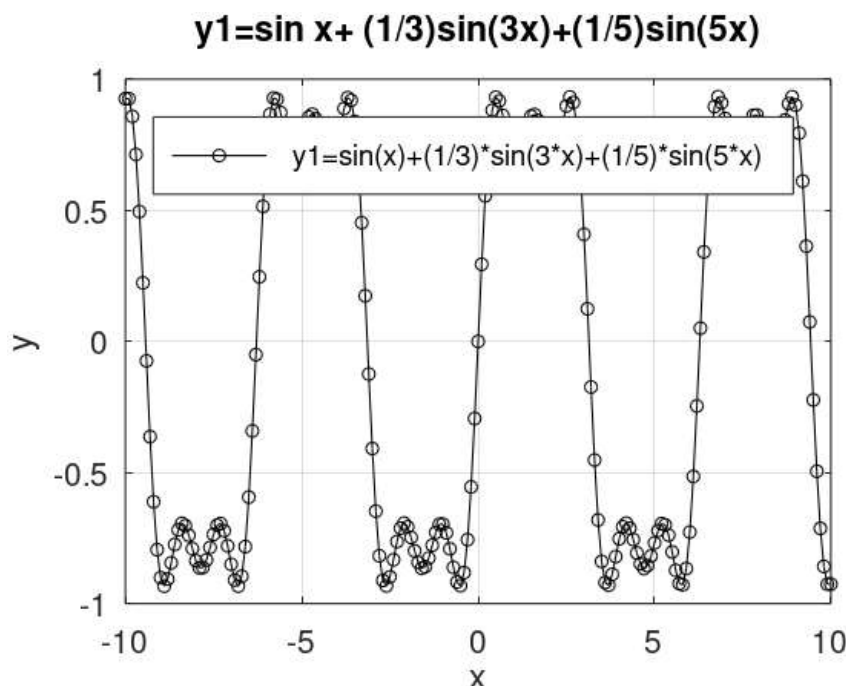


Рис. 2

2. Изменил код, чтоб на одной оси показывались два графика. Для наглядности выделил (Рис. 3-4).

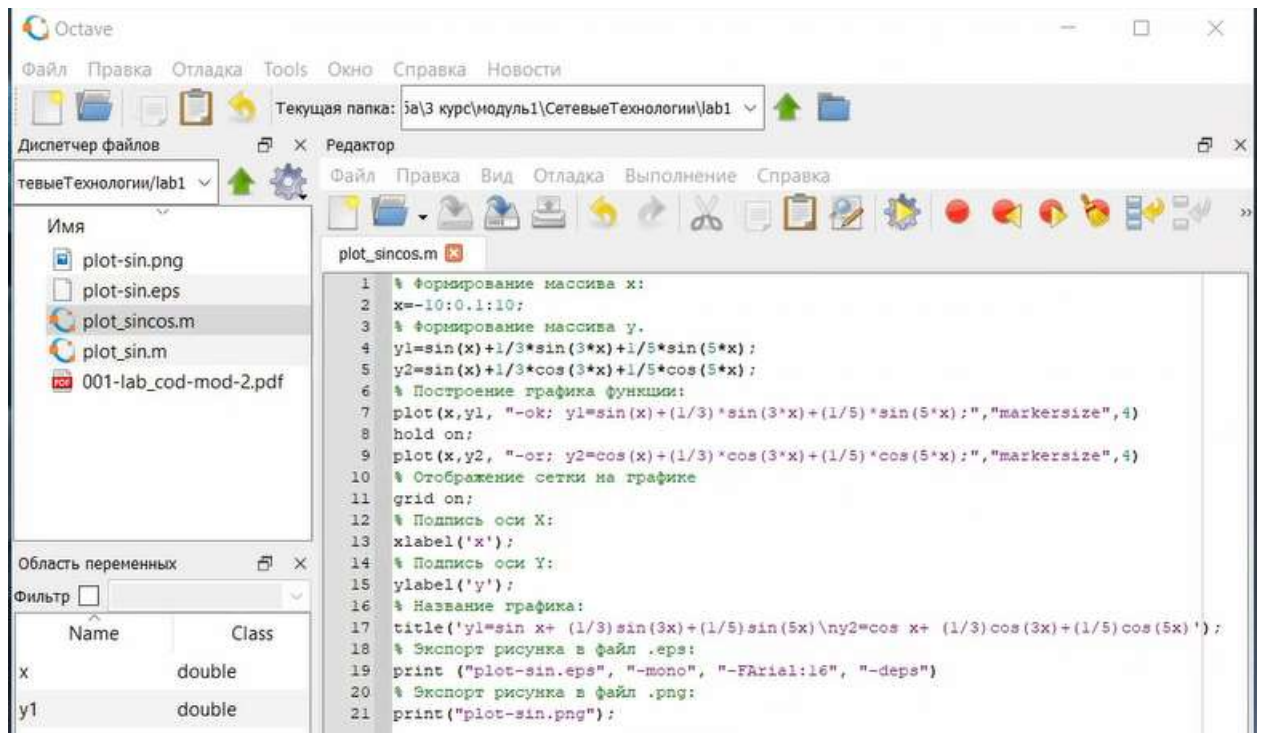


рис. 3

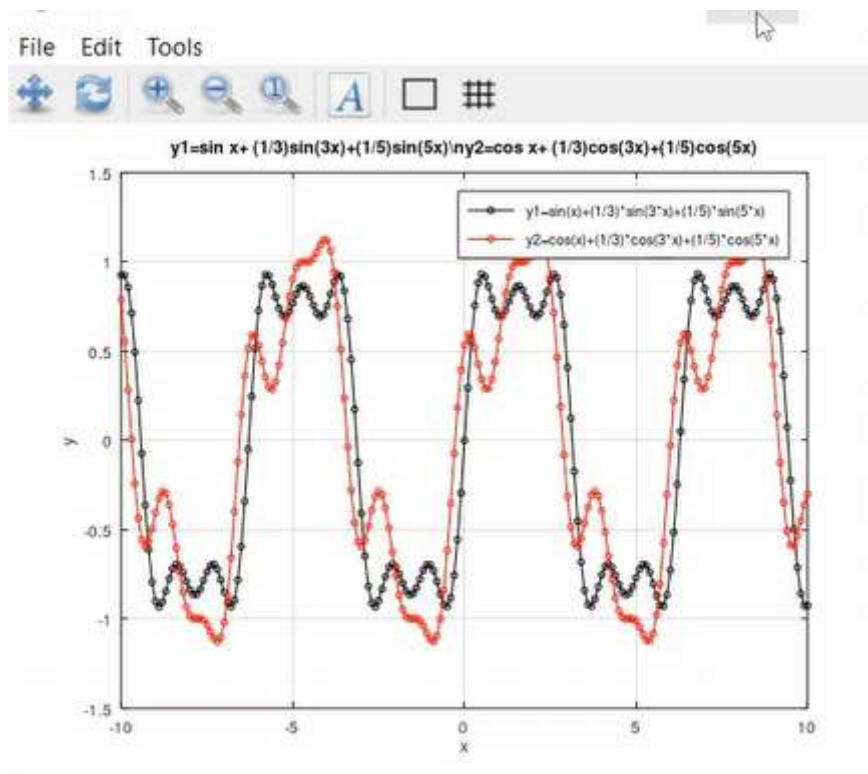


рис. 4

3. Создал новый сценарий и сохранил его в рабочий каталог с именем meandr.m. Скопировал в нём код из листинга и запустил её. Получил графики меандра с разным количеством гармоник и сравнил их между собой. (Рис. 5)

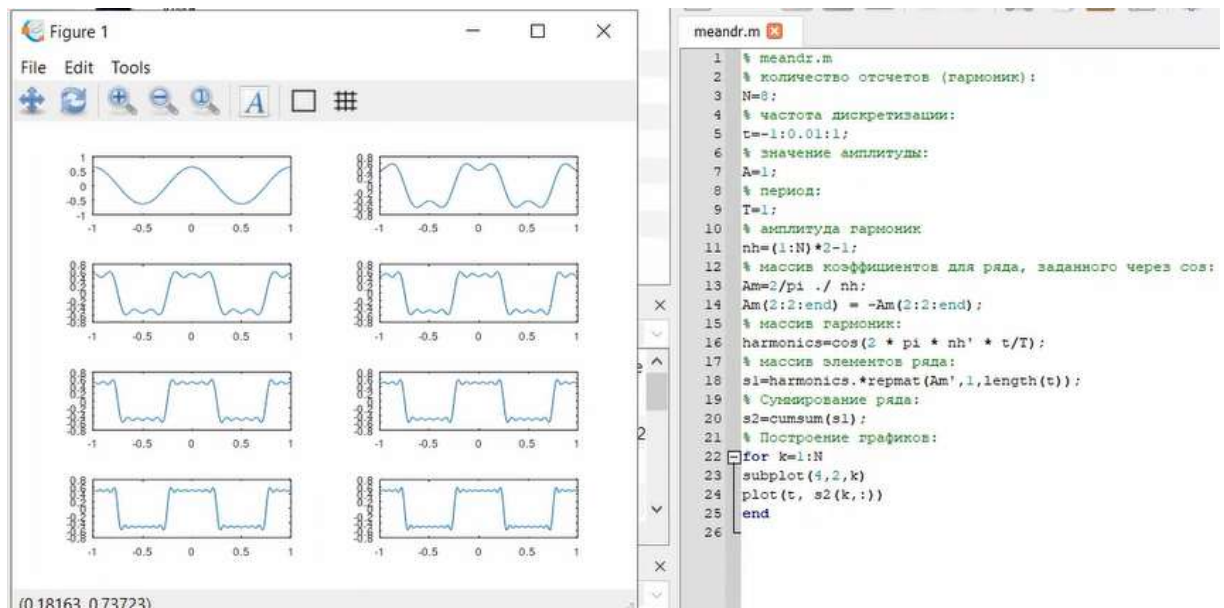


Рис. 5

4. Экспортировал файл в формат png. Потом выразил формулу через синусы и снова запустил программу. Учёл, что при использовании синусов все коэффициенты положительные. (Рис. 6)

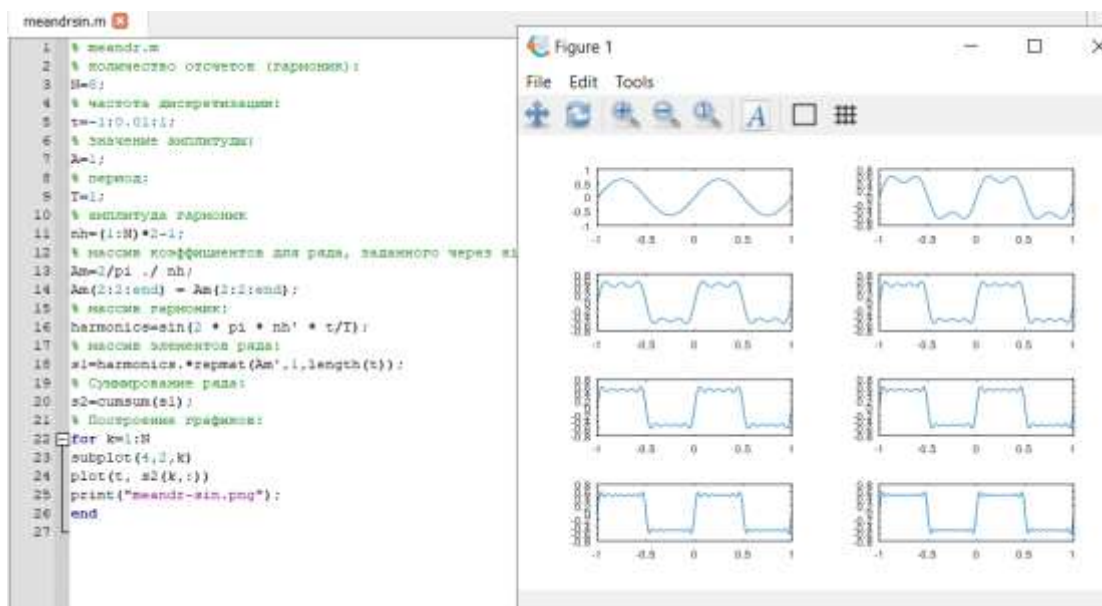


Рис. 6

5. В рабочем каталоге создал каталог spectre1 и в нём новый сценарий с именем, spectre.m. Прописал там код из листинга. (Рис. 7-8).

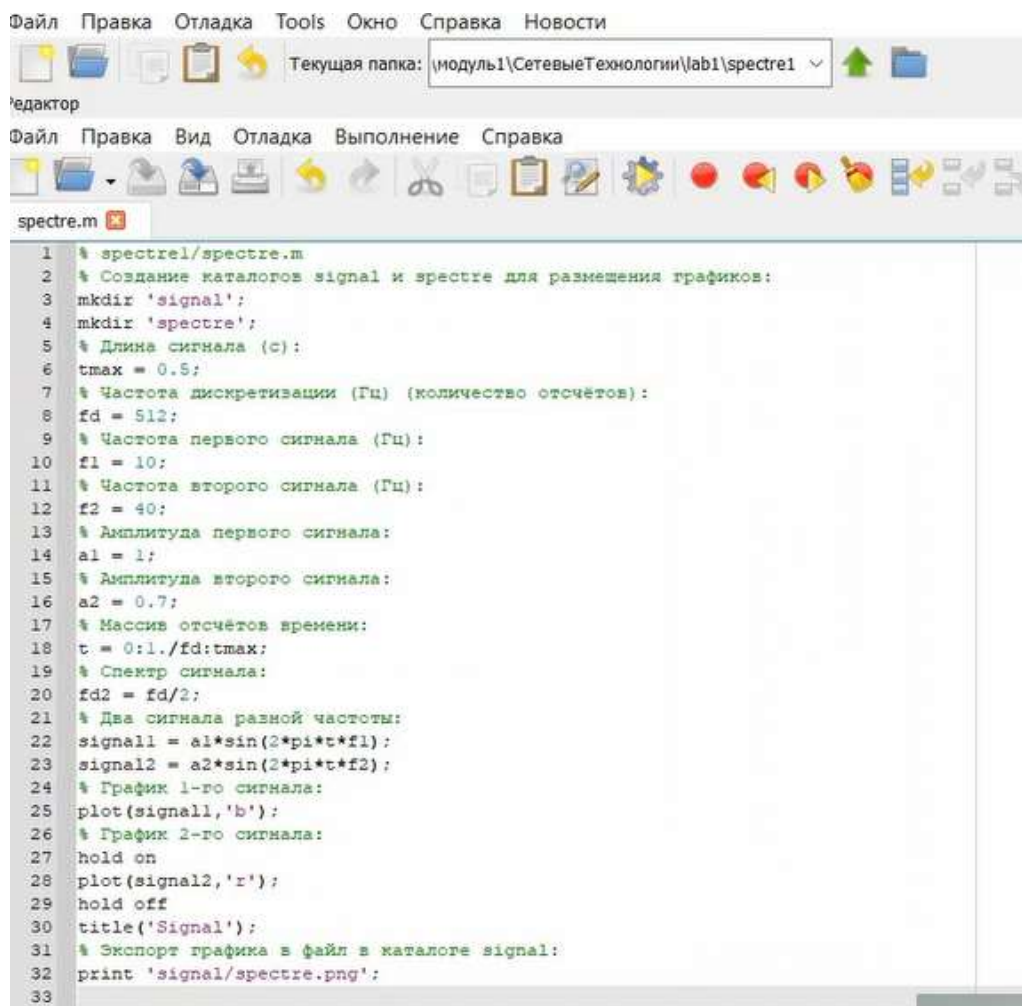


Рис. 7

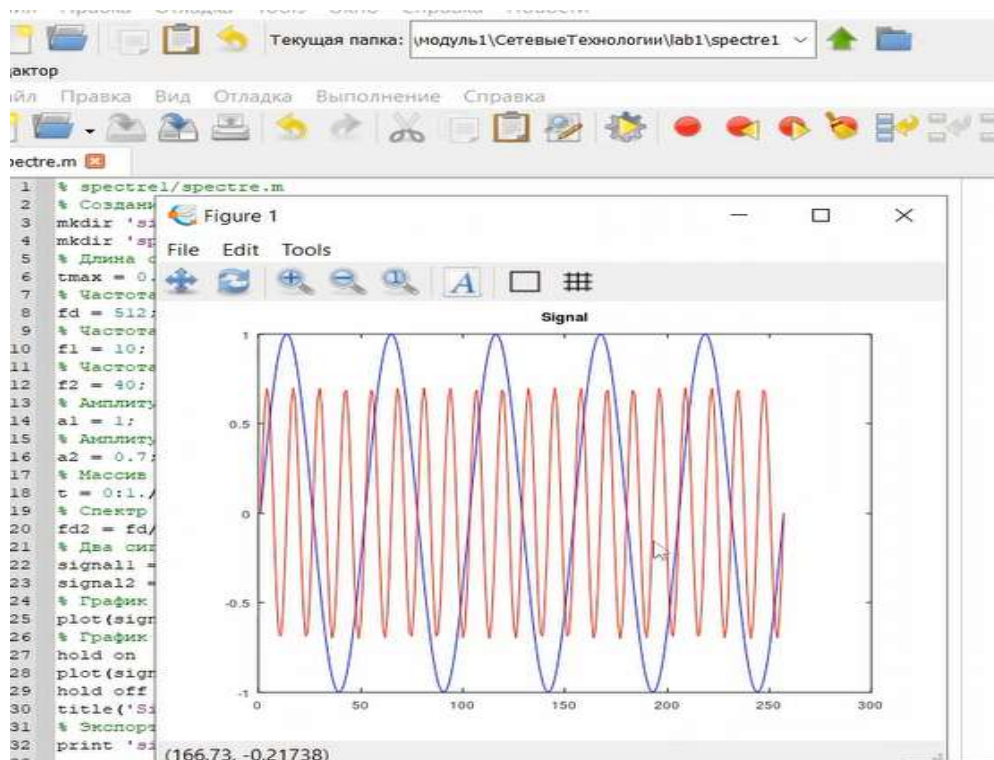
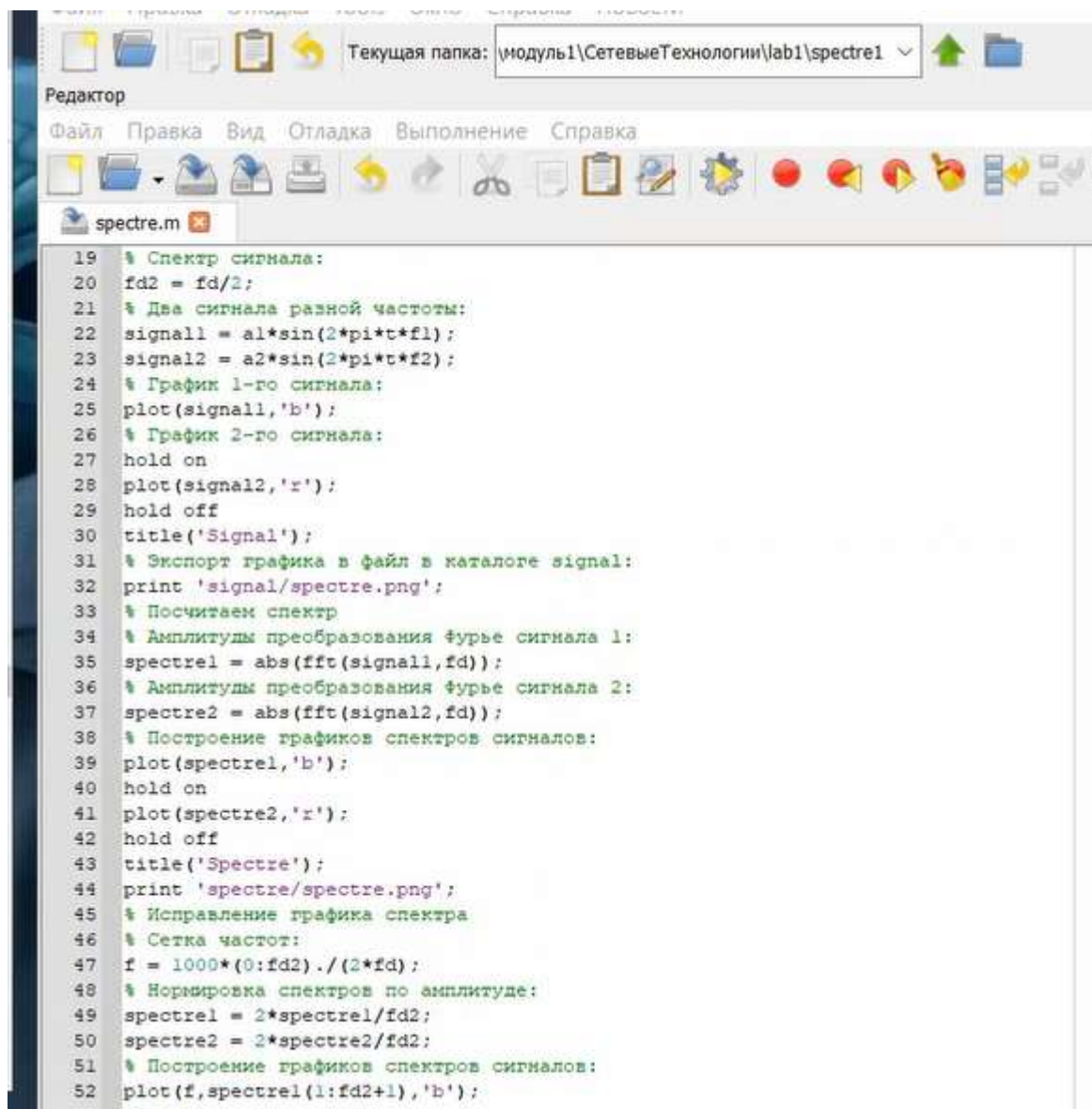


Рис. 8



6. С помощью быстрого преобразования Фурье нашел спектры сигналов добавив в файл spectre.m еще немного кода. Скорректировал график спектра: отбросил дублирующие отрицательные частоты, а также принял расчёт то, что на каждом шаге вычисления быстрого преобразования Фурье происходит суммирование амплитуд сигналов. Присравнении видно, что в исправленном графике отсутствуют отрицательные частоты, а также не суммируются амплитуды при каждом шаге. (Рис. 9-11).



```
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 % Посчитаем спектр
34 % Амплитуды преобразования Фурье сигнала 1:
35 spectre1 = abs(fft(signal1,fd));
36 % Амплитуды преобразования Фурье сигнала 2:
37 spectre2 = abs(fft(signal2,fd));
38 % Построение графиков спектров сигналов:
39 plot(spectre1,'b');
40 hold on
41 plot(spectre2,'r');
42 hold off
43 title('Spectre');
44 print 'spectre/spectre.png';
45 % Исправление графика спектра
46 % Сетка частот:
47 f = 1000*(0:fd2)./(2*fd);
48 % Нормировка спектров по амплитуде:
49 spectre1 = 2*spectre1/fd2;
50 spectre2 = 2*spectre2/fd2;
51 % Построение графиков спектров сигналов:
52 plot(f,spectre1(1:fd2+1),'b');
53 hold on
54 plot(f,spectre2(1:fd2+1),'r');
55 hold off
56 title('Spectre');
57 print 'spectre/spectre.png';
```

рис. 9

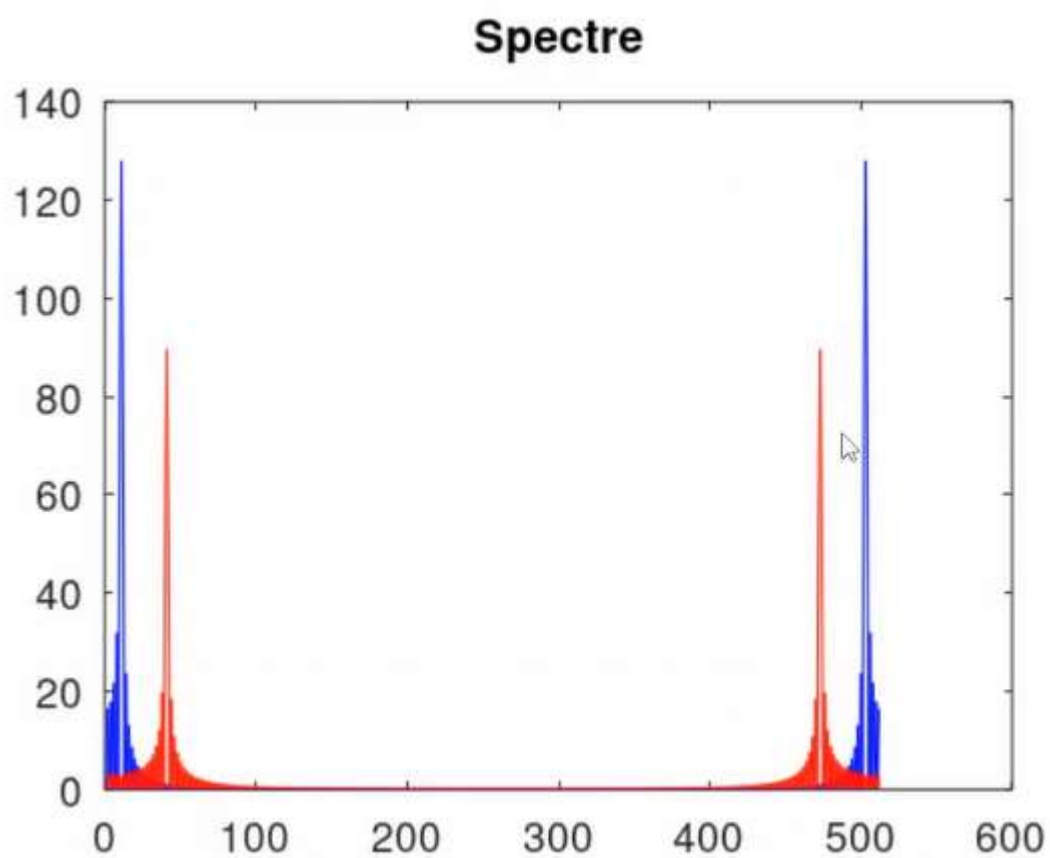


рис. 10

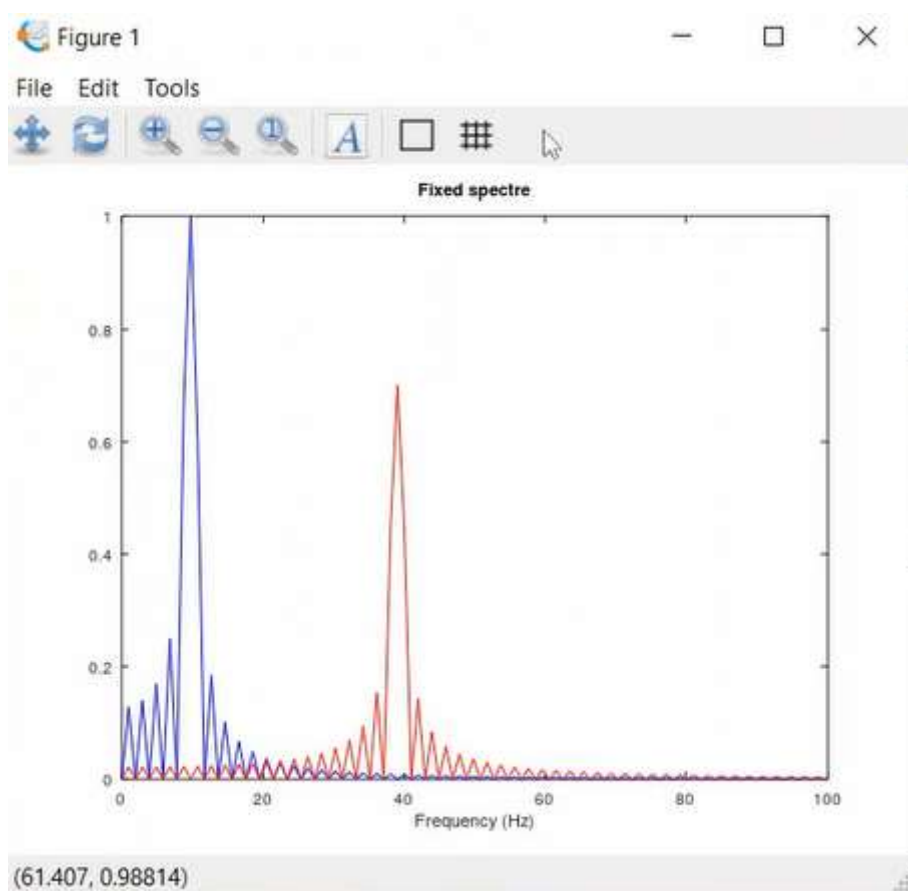
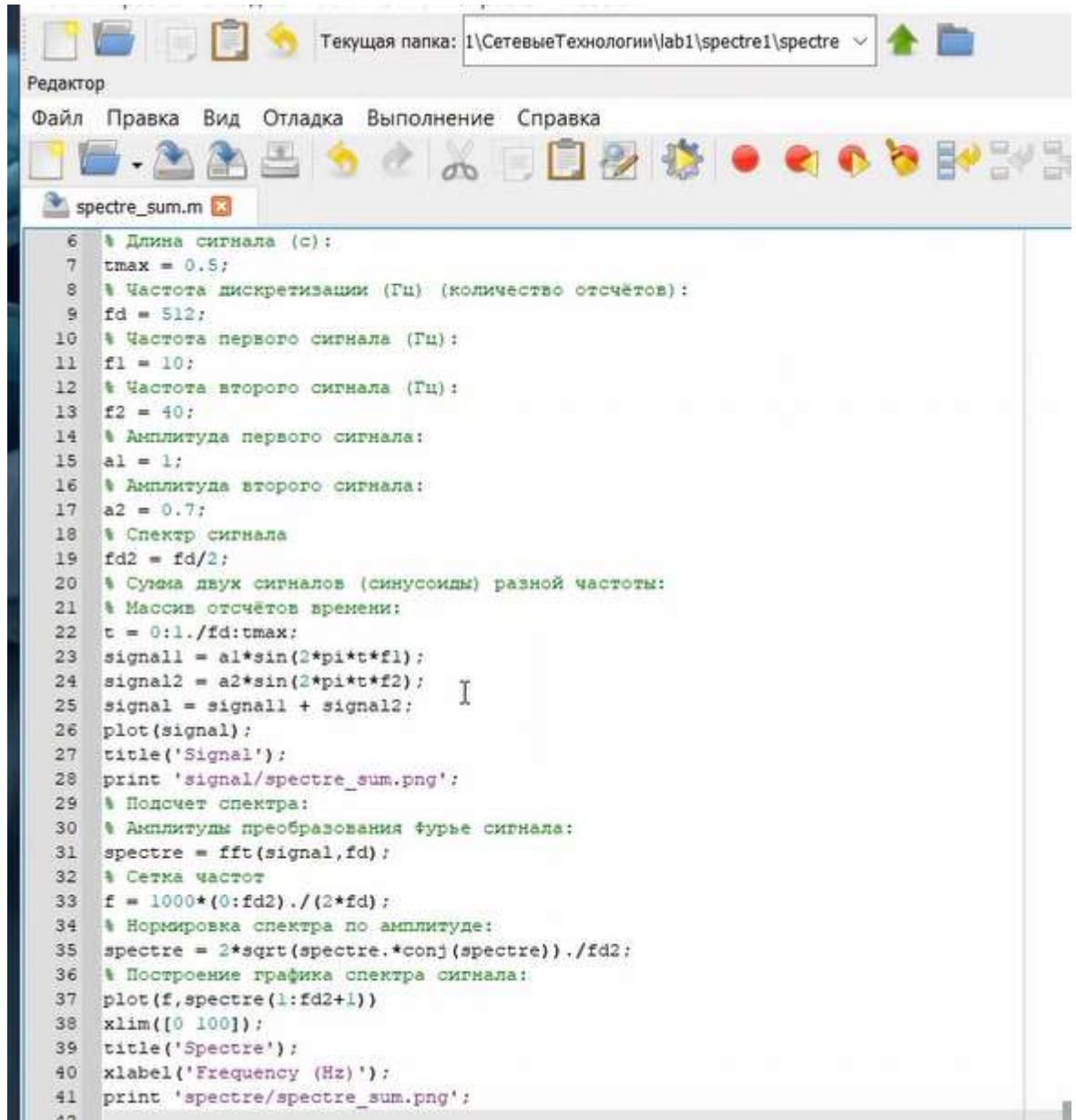


рис. 11

7. Найшел спектр суммы рассмотренных сигналов, создав каталог spectr\_sum и файл в нём spectre\_sum.m. В результате мы видим, что спектр суммы сигналов равен сумме спектров сигналов. (рис. 12-14)



```
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
33 f = 1000*(0:fd2)./(2*fd);
34 % Нормировка спектра по амплитуде:
35 spectre = 2*sqrt(spectre.*conj(spectre))./fd2;
36 % Построение графика спектра сигнала:
37 plot(f,spectre(1:fd2+1))
38 xlim([0 100]);
39 title('Spectre');
40 xlabel('Frequency (Hz)');
41 print 'spectre/spectre_sum.png';
42
```

рис. 12



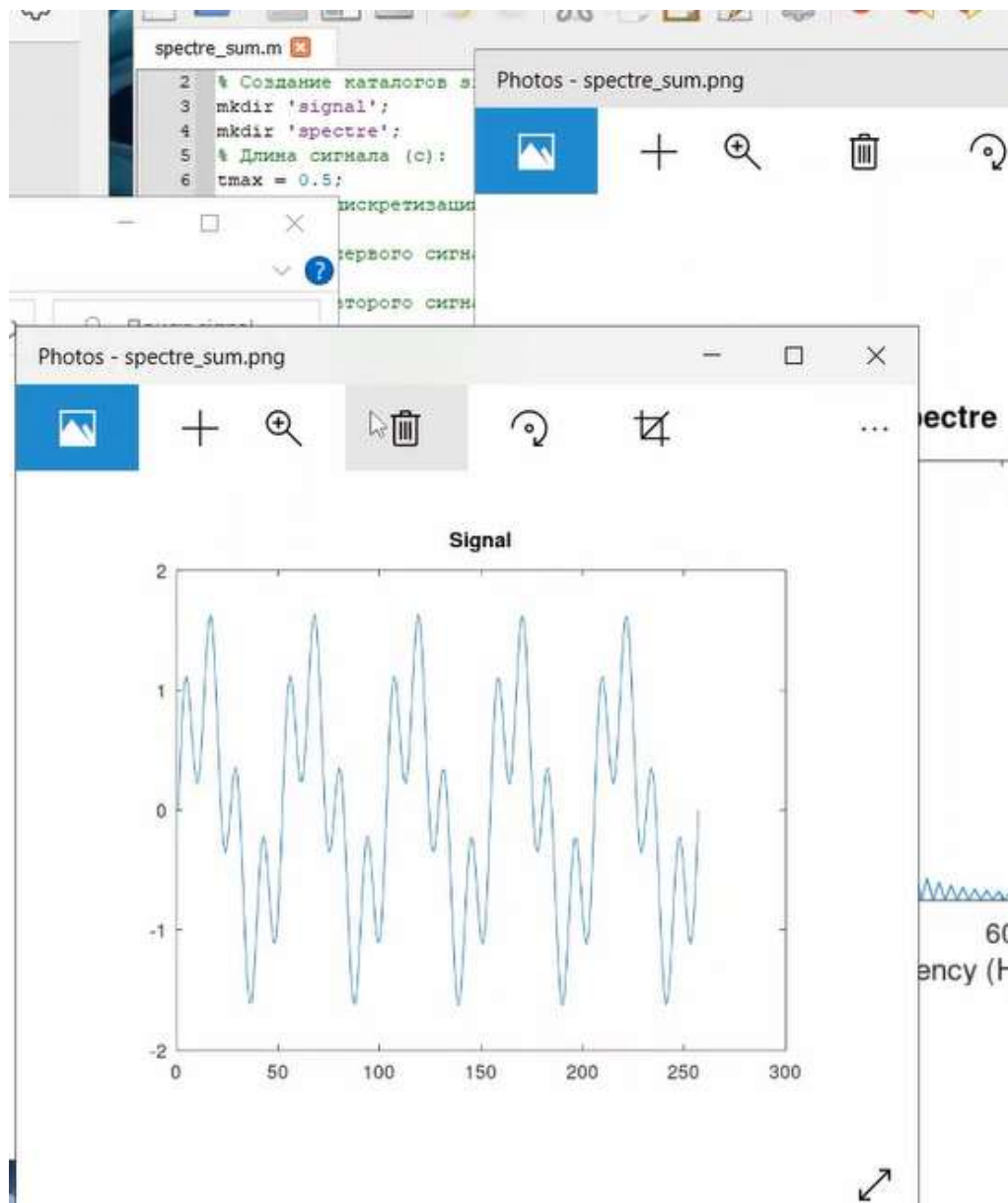


рис. 13

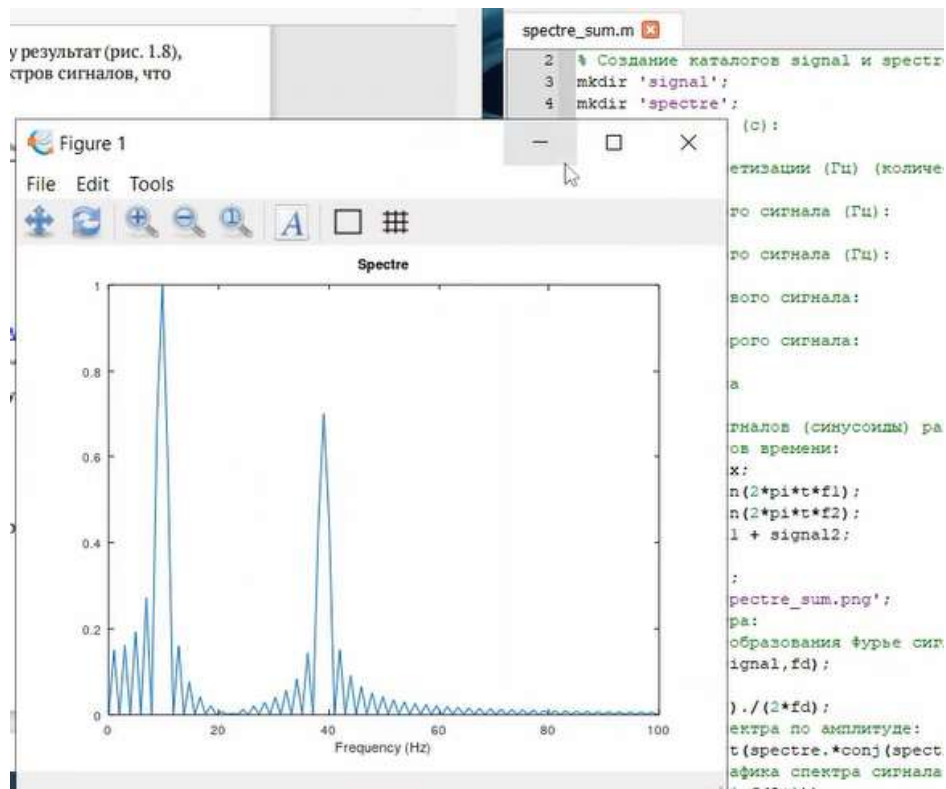


рис. 14

8. Нашел спектр суммы рассмотренных сигналов, создав каталог spectr\_sum и файл в нём spectre\_sum.m со следующим кодом. Попробовал с частотой дискретизации в 79, 256 и 512 Гц. Видно, что при частоте дискретизации меньше 80 Гц сигнал становится прерывистым. (рис. 15-17).

spectre\_sum.m

```

1 % spectr_sum/spectre_sum.m
2 % Создание каталогов signal и spectre для размещения графиков:
3 mkdir 'signal';
4 mkdir 'spectre';
5 % Длина сигнала (с):
6 tmax = 0.5;
7 % Частота дискретизации (Гц) (количество отсчётов):
8 fd = 512;
9 % Частота первого сигнала (Гц):
10 f1 = 10;
11 % Частота второго сигнала (Гц):
12 f2 = 40;
13 % Амплитуда первого сигнала:
14 a1 = 1;
15 % Амплитуда второго сигнала:
16 a2 = 0.7;
17 % Спектр сигнала
18 fd2 = fd/2;
19 % Сумма двух сигналов (синусоиды) разной частоты:
20 % Массив отсчётов времени:
21 t = 0:1./fd:tmax;
22 signal1 = a1*sin(2*pi*t*f1);
23 signal2 = a2*sin(2*pi*t*f2);
24 signal = signal1 + signal2;
25 plot(signal);
26 title('Signal');
27 print 'signal/spectre_sum.png';
28 % Подсчет спектра:
29 % Амплитуды преобразования Фурье сигнала:
30 spectre = fft(signal,fd);
31 % Сетка частот
32 f = 1000*(0:fd2)./(2*fd);
33 % Нормировка спектра по амплитуде:
34 spectre = 2*sqrt(spectre.*conj(spectre))./(fd2);
35 % Построение графика спектра сигнала:
36 plot(f,spectre(1:fd2+1));
37 xlim([0 100]);

```

рис. 15

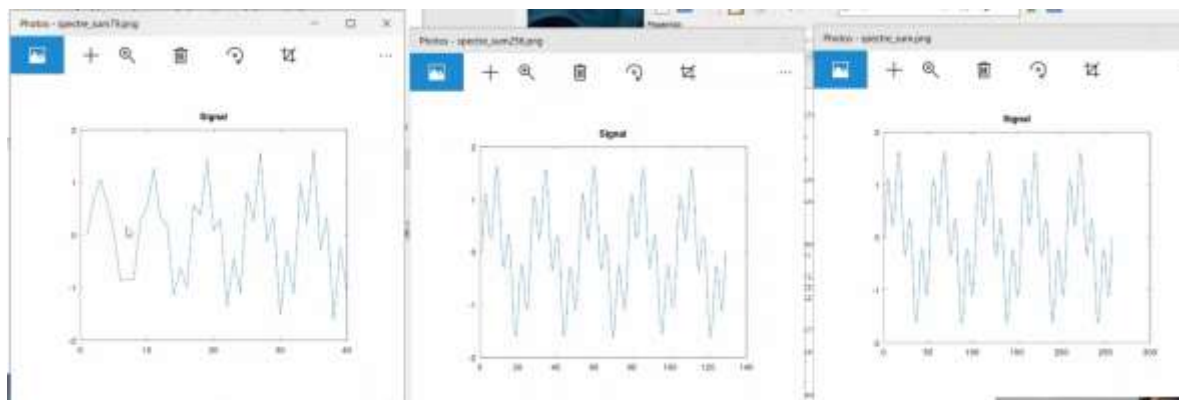


Рис. 16

9. В рабочем каталоге создал каталог modulation и в нём новый сценарий с именем am.m, куда прописал необходимый код. Запустил программу. В результате мы видим, что спектр произведения представляет собой свёртку спектров. (Рис. 17-18).

```

Текущая папка: d:\уль1\СетевыеТехнологии\lab1\modulation
Редактор
Файл Правка Вид Отладка Выполнение Справка
am.m
9 % Частота дискретизации (Гц) (количество отсчётов)
10 fd = 512;
11 % Частота сигнала (Гц)
12 f1 = 5;
13 % Частота несущей (Гц)
14 f2 = 50;
15 % Спектр сигнала
16 fd2 = fd/2;
17 % Построение графиков двух сигналов (синусоиды)
18 % разной частоты
19 % Массив отсчётов времени:
20 t = 0:1./fd:tmax;
21 signal1 = sin(2*pi*t*f1);
22 signal2 = sin(2*pi*t*f2);
23 signal = signal1 .* signal2;
24 plot(signal, 'b');
25 hold on
26 % Построение огибающей:
27 plot(signal1, 'r');
28 plot(-signal1, 'r');
29 hold off
30 title('Signal');
31 print 'signal/am.png';
32 % Расчет спектра:
33 % Амплитуды преобразования Фурье-сигнала:
34 spectre = fft(signal,fd);
35 % Сетка частот:
36 f = 1000*(0:fd2)./(2*fd);
37 % Нормировка спектра по амплитуде:
38 spectre = 2*sqrt(spectre.*conj(spectre))./fd2;
39 % Построение спектра:
40 plot(f,spectre(1:fd2+1), 'b')
41 xlim([0 100]);
42 title('Spectre');
43 xlabel('Frequency (Hz)');
44 print 'spectre/am.png';
45

```

рис. 17

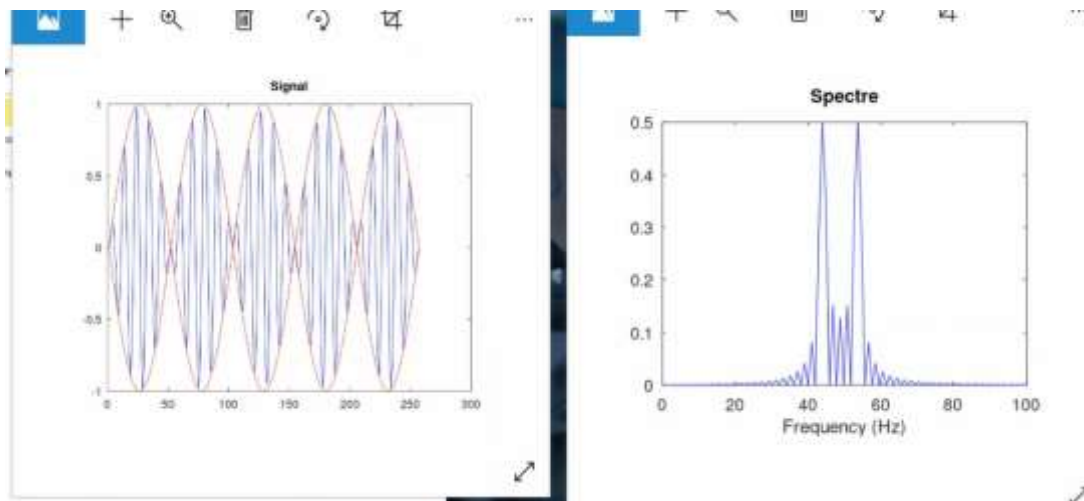


рис.18

10. Проверил, установлен ли у меня пакет расширений signal. В рабочем каталоге создал каталог coding и в нём постепенно создавал файлы main.m, maptowave.m, unipolar.m, ami.m, bipolarnrz.m, bipolarrrz.m, manchester.m, diffmanc.m, calcspectre.m и прописал во всех нужный там код. Убедился в самосинхронизуемости кодов и получил спектры (Рис. 19-22).

```

Текущая папка: c:\модуль1\СетевыеТехнологии\lab1\coding
Редактор
Файл Правка Вид Отладка Выполнение Справка
main.m maptowave.m unipolar.m ami.m bipolarnrz.m bipolarrrz.m manchester.m difmanc.m
40 print 'signal/manchester.png';
41 % Дифференциальное манчестерское кодирование
42 wave=diffmanc(data);
43 plot(wave)
44 title('Differential Manchester');
45 print 'signal/diffmanc.png';
46 % Униполярное кодирование
47 wave=unipolar(data_sync);
48 plot(wave);
49 ylim([-1 6]);
50 title('Unipolar');
51 print 'sync/unipolar.png';
52 % Кодирование AMI
53 wave=ami(data_sync);
54 plot(wave)
55 title('AMI');
56 print 'sync/ami.png';
57 % Кодирование NRZ
58 wave=bipolarnrz(data_sync);
59 plot(wave);
60 title('Bipolar Non-Return to Zero');
61 print 'sync/bipolarnrz.png';
62 % Кодирование RZ
63 wave=bipolarrrz(data_sync);
64 plot(wave)
65 title('Bipolar Return to Zero');
66 print 'sync/bipolarrrz.png';
67 % Манчестерское кодирование
68 wave=manchester(data_sync);
69 plot(wave)
70 title('Manchester');
71 print 'sync/manchester.png';
72 % Дифференциальное манчестерское кодирование
73 wave=diffmanc(data_sync);
74 plot(wave)
75 title('Differential Manchester');
76 print 'sync/diffmanc.png';

```

Рис. 19

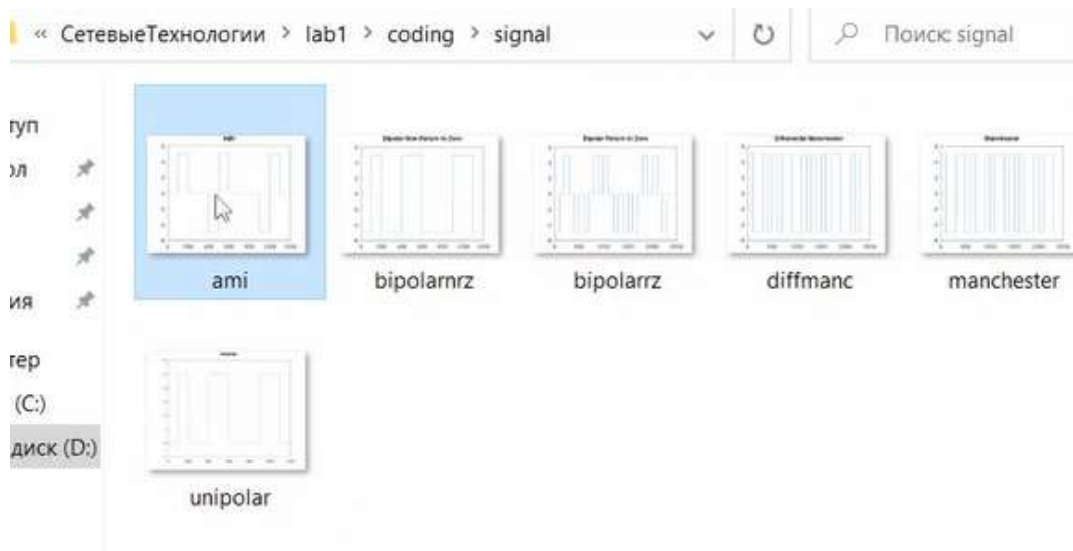


рис. 20

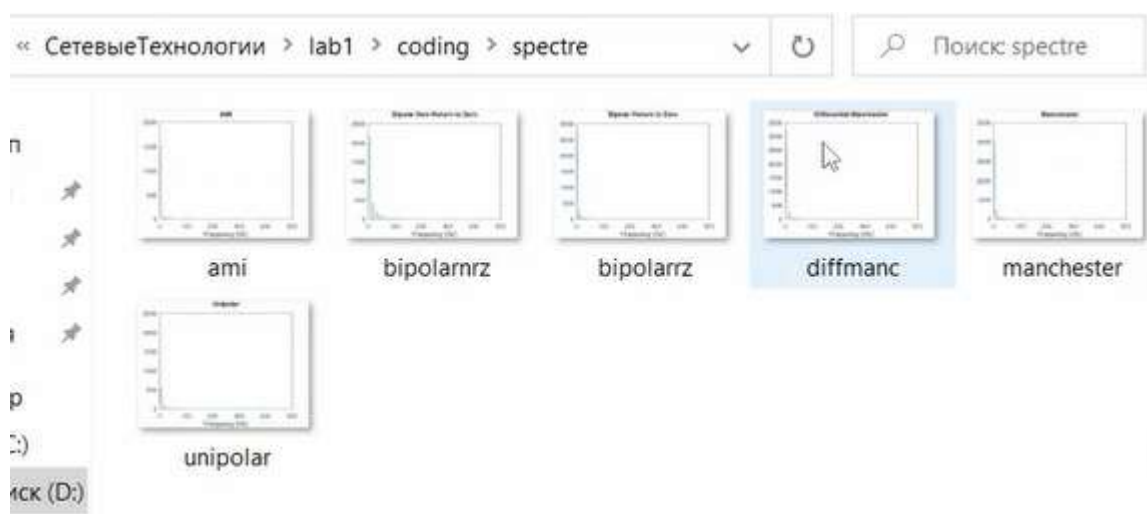


рис. 21

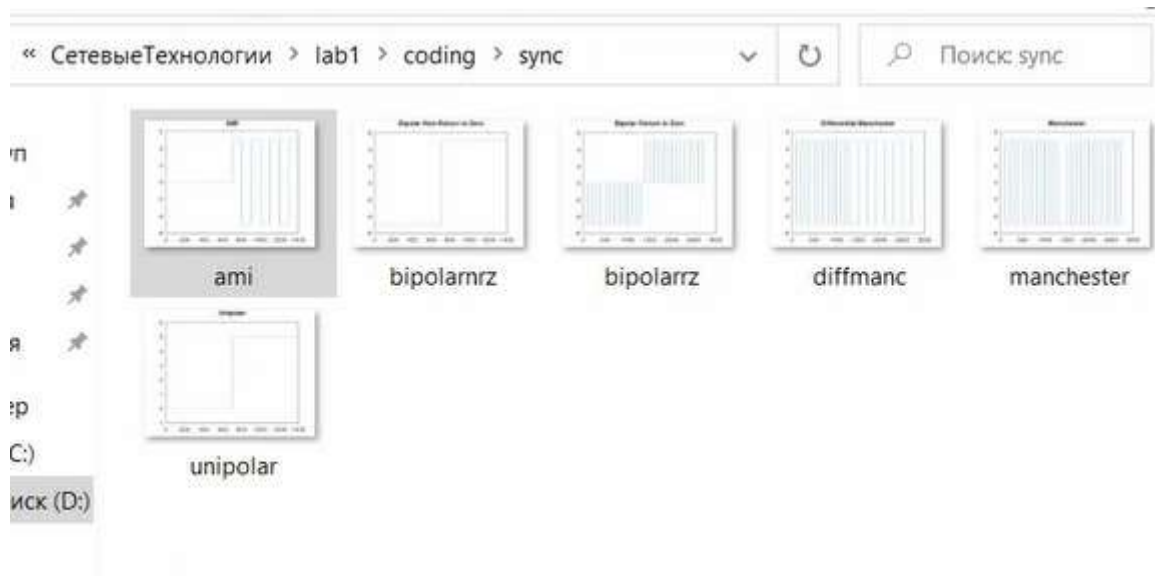


Рис. 22



## **ВЫВОД**

Я изучил методы кодирования и модуляции сигналов с помощью языка программирования Octave. Научился определять спектра и параметры сигнала. Освоил принципов модуляции сигнала на примере аналоговой амплитудной модуляции. Проверил свойства самосинхронизации сигнала и получил спектры.