

Лабораторная работа 1

Методы кодирования и модуляция сигналов

Лушин Артём Андреевич

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Вывод	27

Список иллюстраций

2.1	Код plot_sin.m	5
2.2	График plot_sin.m	6
2.3	Код plot_combo.m	6
2.4	Рисунок plot_combo.m	7
2.5	Код meandr.m	7
2.6	Рисунок meandr.m	8
2.7	Код изменённого meandr.m	8
2.8	Рисунок изменённого meandr.m	9
2.9	Код spectre.m	10
2.10	Рисунок spectre.m	11
2.11	Код spectre.m с Фурье	12
2.12	Рисунок spectre.m с Фурье	13
2.13	Код spectre.m с улучшенным Фурье	14
2.14	Рисунок spectre.m с улучшенным Фурье	15
2.15	Код spectre_sum.m	16
2.16	Рисунок spectre_sum.m	17
2.17	Рисунок2 spectre_sum.m	18
2.18	Код am.m	19
2.19	Рисунок am.m	20
2.20	Рисунок2 am.m	21
2.21	Проверка пакета расширений	21
2.22	Код main.m	22
2.23	Код maptowave.m	23
2.24	Код unipolar.m	23
2.25	Код ami.m	23
2.26	Код bipolarnrz.m	23
2.27	Код bipolarrrz.m	24
2.28	Код manchester.m	24
2.29	Код diffmanc.m	24
2.30	Код calcspectre.m	25
2.31	Рисунки в каталоге signal	25
2.32	Рисунки в каталоге sync	26
2.33	Рисунки в каталоге spectre	26

1 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.

2 Выполнение лабораторной работы

- 1) Я создал и сохранил файл plot_sin.m. В этом файле написал код, чтобы построить функцию. После исполнения кода вывелся график.

```
x=-10:0.1:10;  
y1=sin(x)+1/3*sin(3*x)+1/5*sin(5*x);  
plot(x,y1, "-ok; y1=sin(x)+(1/3)*sin(3*x)+(1/5)*sin(5*x);", "markersize", 4)  
grid on;  
xlabel('x');  
ylabel('y');  
title('y1=sin x+ (1/3)sin(3x)+(1/5)sin(5x)');  
print ("plot-sin.eps", "-mono", "-FArial:16", "-deps")  
print("plot-sin.png");
```

Рис. 2.1: Код plot_sin.m

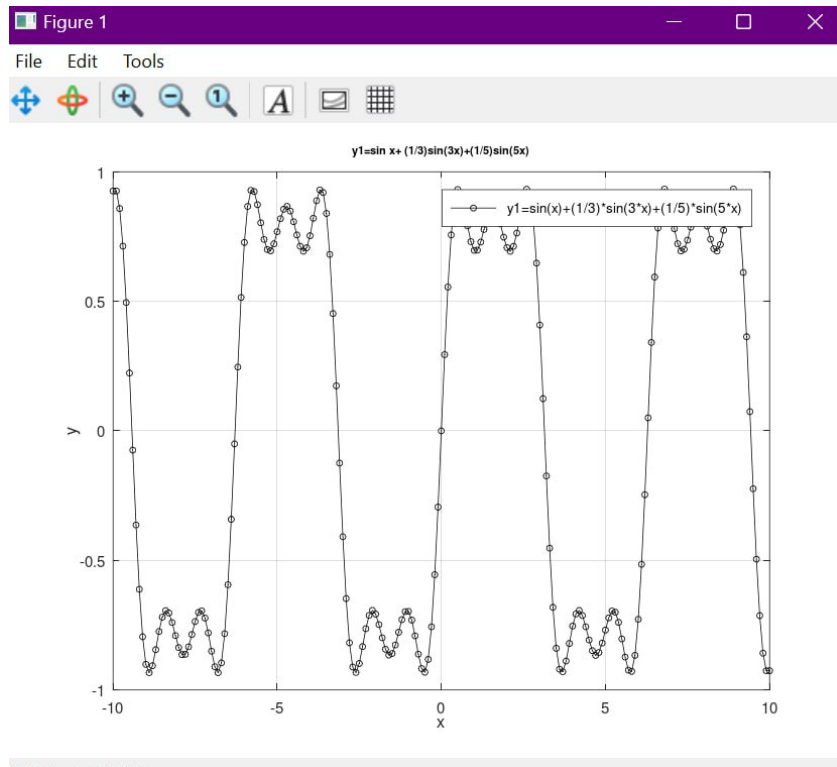


Рис. 2.2: График plot_sin.m

- 2) Создали новый файл и написал код, который основывается на коде из прошлого пункта, но добавляет и функцию с косинусом. Сгенерировал рисунок.

```
x=-10:0.1:10;
y1=sin(x)+1/3*sin(3*x)+1/5*sin(5*x);
y2 = cos(x)+1/3*cos(3*x)+1/5*cos(5*x);
plot(x,y1, "-ok; y1=sin(x)+(1/3)*sin(3*x)+(1/5)*sin(5*x);", "markersize", 4)
hold on;
plot(x,y2, "-k; y2=cos(x)+1/3*cos(3*x)+1/5*cos(5*x);", "markersize", 4)
grid on;
xlabel('x');
ylabel('y');
title('y1=sin x+ (1/3)sin(3x)+(1/5)sin(5x)');
title('y2=cos(x)+1/3*cos(3*x)+1/5*cos(5*x)');
print ("plot-sin.eps", "-mono", "-FArial:16", "-deps")
print("plot-sin.png");
```

Рис. 2.3: Код plot_combo.m

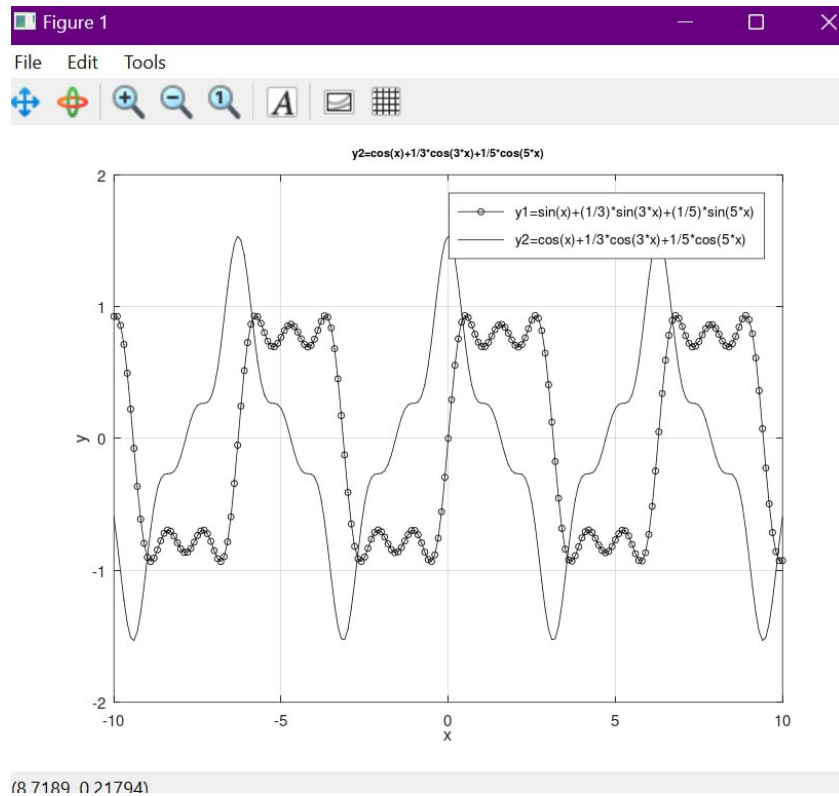


Рис. 2.4: Рисунок plot_combo.m

- 3) Создал новый файл meandr для реализации графики меандра, реализованный с различным количеством гармоник. Реализовывал через косинусы.

```

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
23     subplot(4,2,k)
24     plot(t, s2(k,:))
25 end

```

Рис. 2.5: Код meandr.m

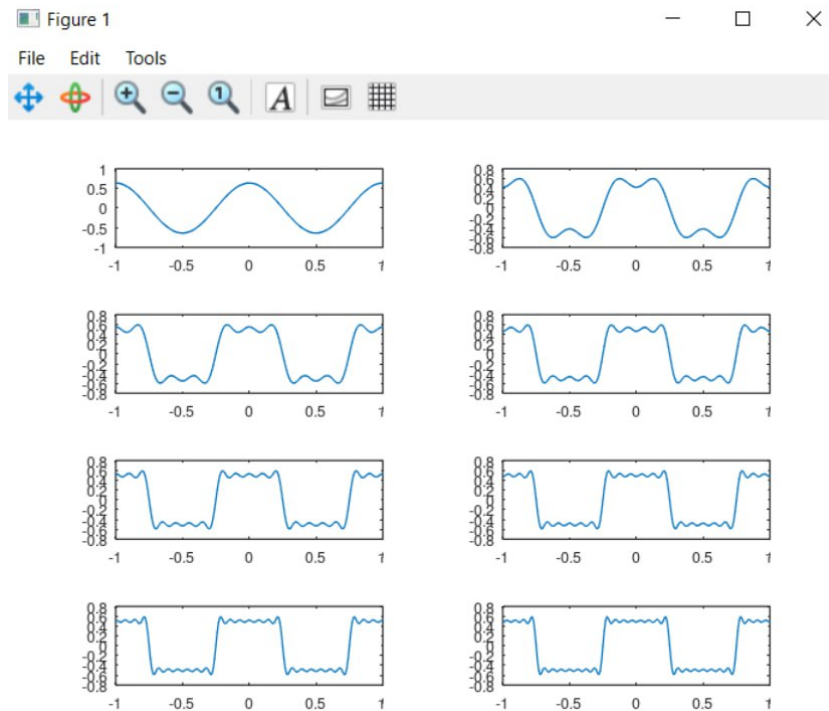


Рис. 2.6: Рисунок meandr.m

4) Изменил код файла meandr, чтобы он реализовывался через синусы.

```

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=sin(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
23     subplot(4,2,k)
24     plot(t, s2(k,:))
25 end

```

Рис. 2.7: Код изменённого meandr.m

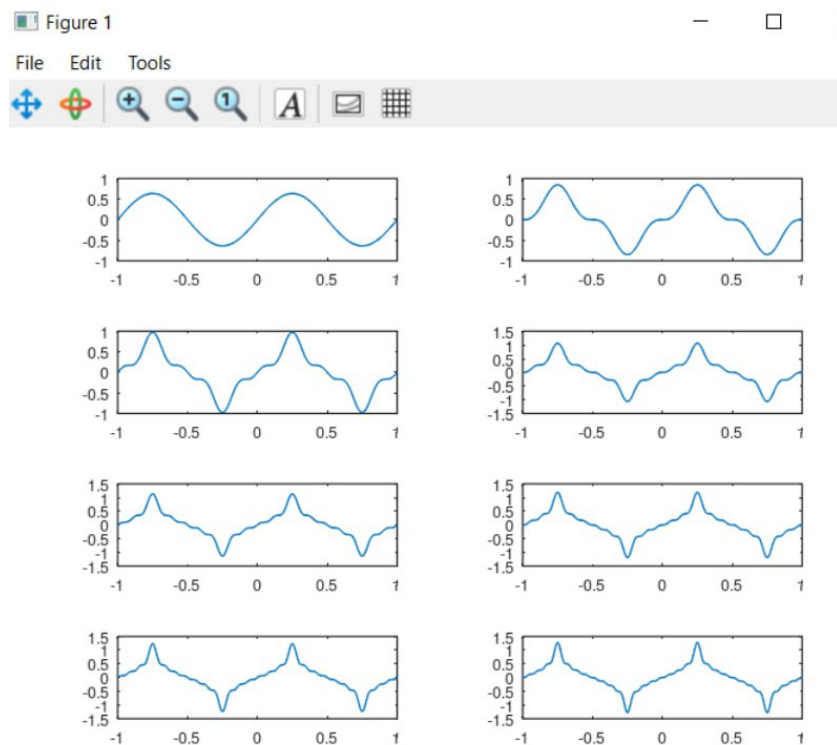


Рис. 2.8: Рисунок изменённого meandr.m

- 5) Создал новый каталог spectre1, а в нём файл spectre.m. Файл будет определять спектр двух отдельных сигналов и их сумму.

```

1 % spectre1/spectre.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 % 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
25 % График 1-го сигнала:
26 plot(signal1,'b');
27 % График 2-го сигнала:
28 hold on
29 plot(signal2,'r');
30 hold off
31 title('Signal');
32 % Экспорт графика в файл в каталоге signal:
33 print 'signal/spectre.png';
34

```

Рис. 2.9: Код spectre.m

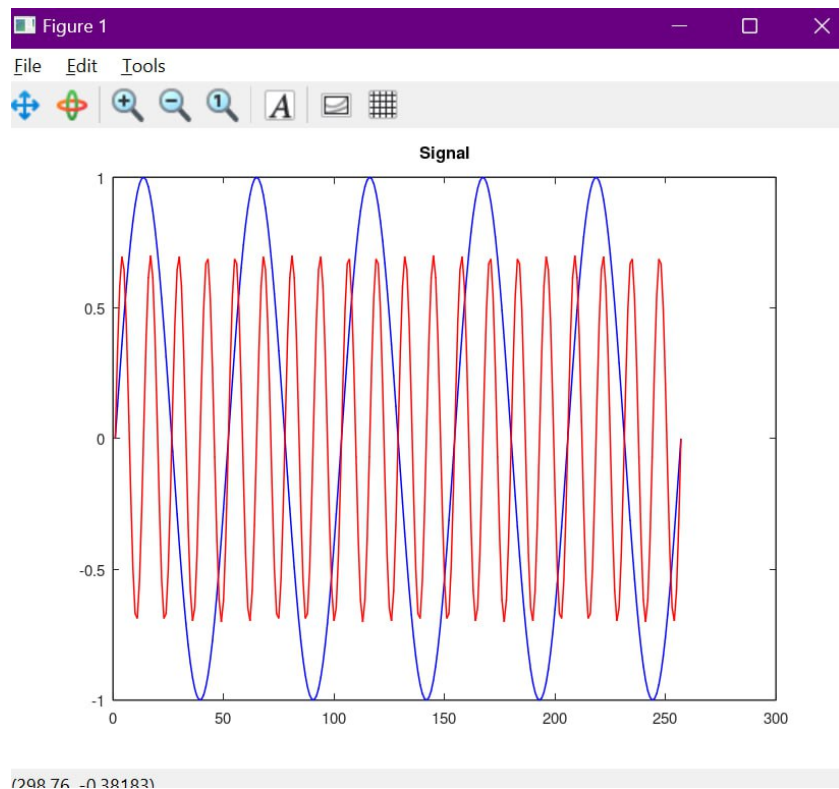


Рис. 2.10: Рисунок spectre.m

- 6) Затем изменил код файла spectre, чтобы он использовал преобразования Фурье.

```

% spectre1/spectre.m
% Создание каталогов signal и spectre для размещения графиков:
mkdir 'signal';
mkdir 'spectre';
% Длина сигнала (с):
tmax = 0.5;
% Частота дискретизации (Гц) (количество отсчётов):
fd = 512;
% Частота первого сигнала (Гц):
f1 = 10;
% Частота второго сигнала (Гц):
f2 = 40;
% Амплитуда первого сигнала:
a1 = 1;
% Амплитуда второго сигнала:
a2 = 0.7;
% Массив отсчётов времени:
t = 0:1./fd:tmax;
% Спектр сигнала:
fd2 = fd/2;
% Два сигнала разной частоты:
signal1 = a1*sin(2*pi*t*f1);
signal2 = a2*sin(2*pi*t*f2);

% График 1-го сигнала:
plot(signal1,'b');
% График 2-го сигнала:
hold on
plot(signal2,'r');
hold off
title('Signal');
% Экспорт графика в файл в каталоге signal:
print 'signal/spectre.png';
% Посчитаем спектр
% Амплитуды преобразования Фурье сигнала 1:
spectre1 = abs(fft(signal1,fd));
% Амплитуды преобразования Фурье сигнала 2:
spectre2 = abs(fft(signal2,fd));
% Построение графиков спектров сигналов:
plot(spectre1,'b');
hold on
plot(spectre2,'r');
hold off
title('Spectre');
print 'spectre/spectre.png';

```

Рис. 2.11: Код spectre.m с Фурье

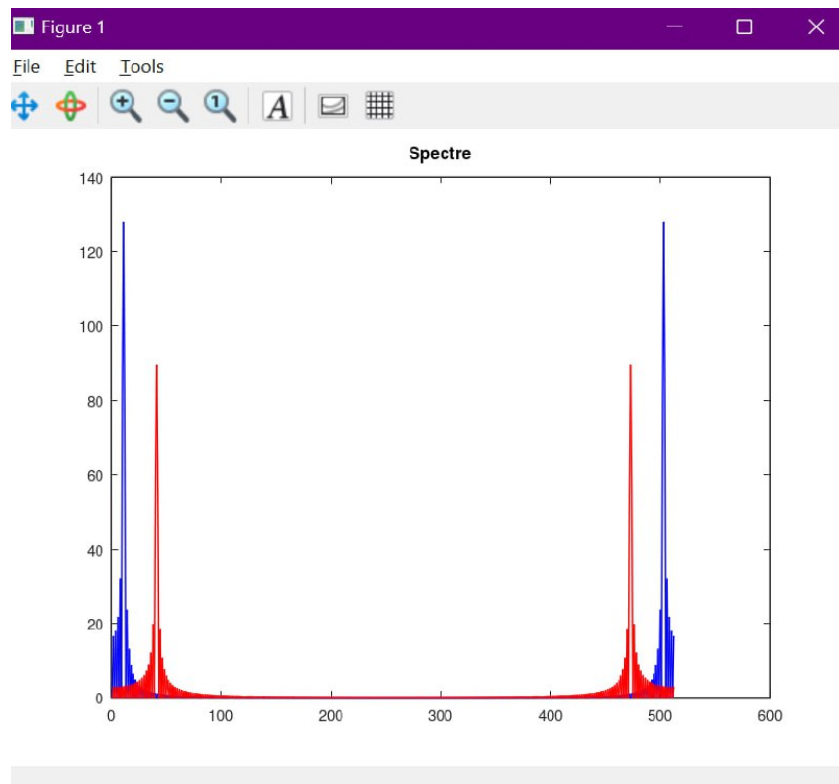


Рис. 2.12: Рисунок spectre.m с Фурье

- 7) Я улучшил график преобразования Фурье, чтоб он отбрасывал дублирующие отрицательные частоты.

```

1 % spectre1/spectre.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 % 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
25 % График 1-го сигнала:
26 plot(signal1,'b');
27 % График 2-го сигнала:
28 hold on
29 plot(signal2,'r');
30 hold off
31 title('Signal');
32 % Экспорт графика в файл в каталоге signal:
33 print 'signal/spectre.png';
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');
43 hold off
44 title('Spectre');
45 print 'spectre/spectre.png';
46 % Исправление графика спектра
47 % Сетка частот:
48 f = 1000*(0:fd2)./(2*fd);
49 % Нормировка спектров по амплитуде:
50 spectre1 = 2*spectre1/fd2;
51 spectre2 = 2*spectre2/fd2;
52 % Построение графиков спектров сигналов:
53 plot(f,spectre1(1:fd2+1),'b');
54 hold on
55 plot(f,spectre2(1:fd2+1),'r');
56 hold off
57 xlim([0 100]);
58 title('Fixed spectre');
59 xlabel('Frequency (Hz)');
60 print 'spectre/spectre_fix.png';
61

```

Рис. 2.13: Код spectre.m с улучшенным Фурье

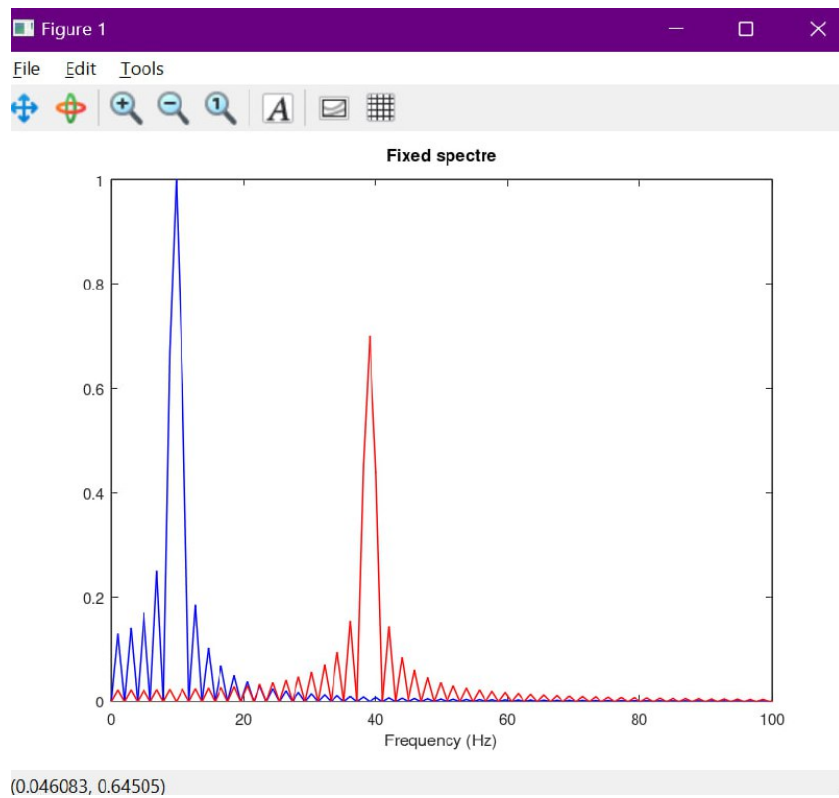


Рис. 2.14: Рисунок spectre.m с улучшенным Фурье

- 8) Я создал каталог spectr_sum и в нём создал файл spectre_sum.m. Вписал следующий код и на выходе доказал, что спектр сумм сигналов равен сумме спектров сигнала.

```

1  % spectr_sum/spectre_sum.m
2  % Создание каталогов signal и spectre для размещения dграфиков:
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]);
38 title('Spectre');
39 xlabel('Frequency (Hz)');
40 print 'spectre/spectre_sum.png';

```

Рис. 2.15: Код spectre_sum.m

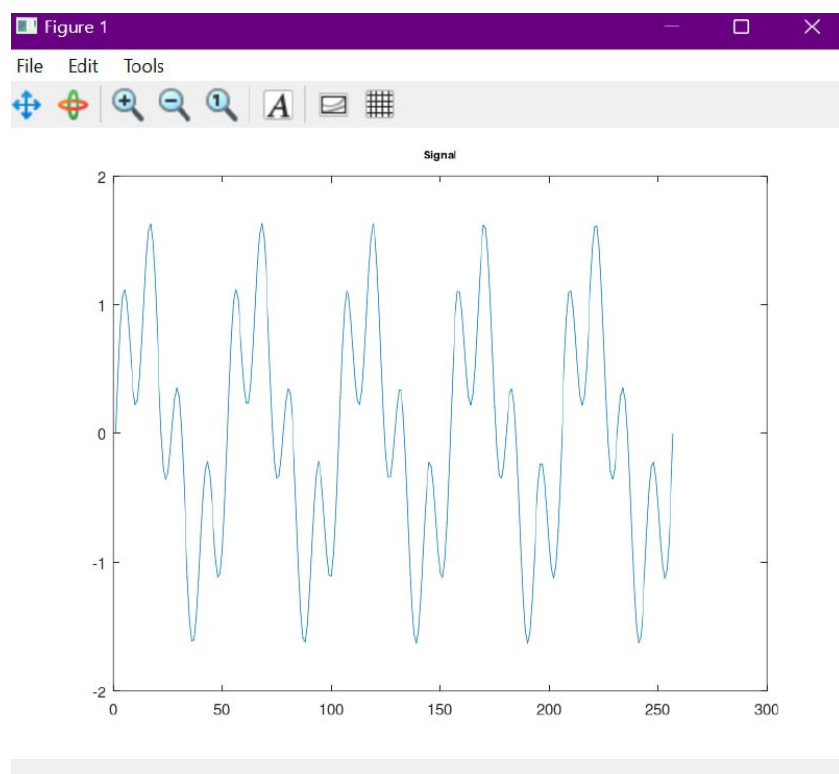


Рис. 2.16: Рисунок spectre_sum.m

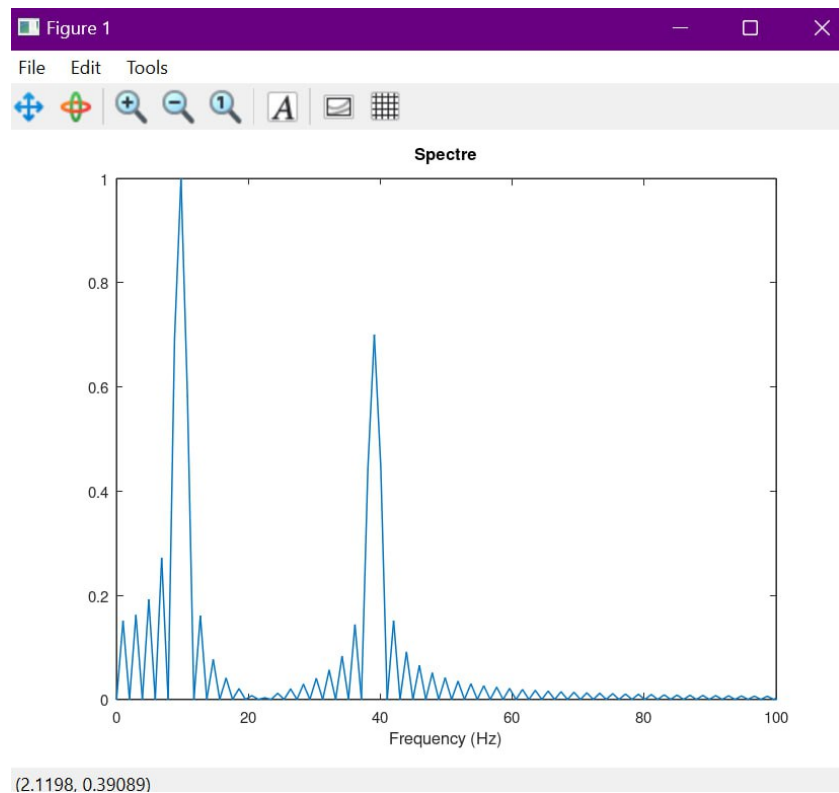


Рис. 2.17: Рисунок2 spectre_sum.m

- 9) Создал каталог modulation, а внутри создал файл am. Этот код нам показывает, что спектр произведения составляет свёртку спектров.

```

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;
38 % Построение спектра:
39 plot(f,spectre(1:fd2+1), 'b')
40 xlim([0 100]);
41 plot(signal, 'b');
42 hold on
43 % Построение огибающей:
44 plot(signal1, 'r');
45 plot(-signal1, 'r');
46 hold off
47 title('Signal');
48 print 'signal/am.png';
49 % Расчет спектра:
50 % Амплитуды преобразования Фурье-сигнала:
51 spectre = fft(signal,fd);
52 % Сетка частот:
53 f = 1000*(0:fd2)/(2*fd);
54 % Нормировка спектра по амплитуде:
55 spectre = 2*sqrt(spectre.*conj(spectre))./fd2;
56 % Построение спектра:
57 plot(f,spectre(1:fd2+1), 'b')
58 xlim([0 100]);

```

Рис. 2.18: Код am.m

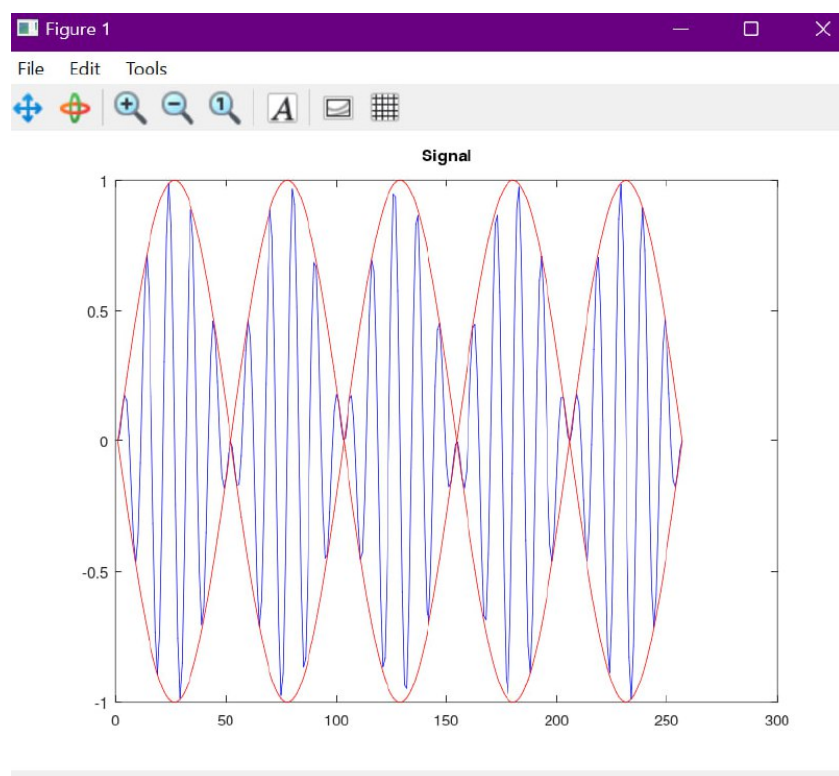


Рис. 2.19: Рисунок am.m

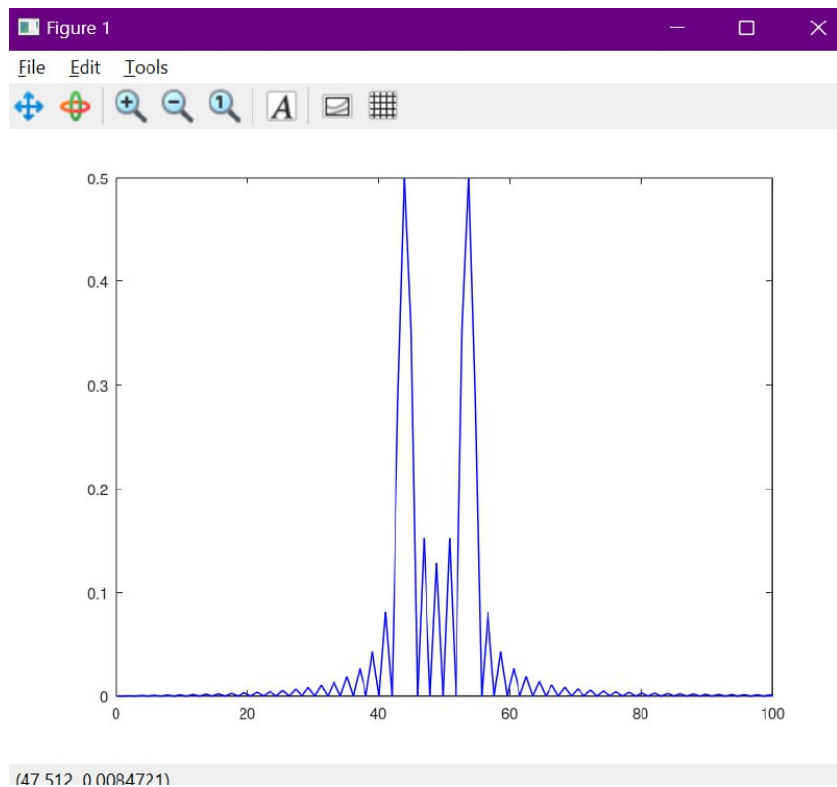


Рис. 2.20: Рисунок2 am.m

10) Я создал каталог coding. В нём создал файлы main.m, mартowave.m, unipolar.m, ami.m, bipolar.m, diffmanс.m, calcspectre.m. Проверил установлен ли пакет расширений.

Комбинированное окно			
miscellaneous	1.3.0	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\miscellaneous-1.3.0	
mqtt	0.0.5	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\mqtt-0.0.5	
nan	3.7.0	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\nan-3.7.0	
netcdf	1.0.17	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\netcdf-1.0.17	
nurbs	1.4.3	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\nurbs-1.4.3	
ocs	0.1.5	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\ocs-0.1.5	
octproj	3.0.2	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\octproj-3.0.2	
optim	1.6.2	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\optim-1.6.2	
optiminterp	0.3.7	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\optiminterp-0.3.7	
quaternion	2.4.0	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\quaternion-2.4.0	
queueing	1.2.8	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\queueing-1.2.8	
signal	1.4.5	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\signal-1.4.5	
sockets	1.4.1	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\sockets-1.4.1	
sparsersb	1.0.9	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\sparsersb-1.0.9	
splines	1.3.5	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\splines-1.3.5	
statistics	1.6.6	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\statistics-1.6.6	
stk	2.8.1	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\stk-2.8.1	
strings	1.3.1	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\strings-1.3.1	
struct	1.0.18	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\struct-1.0.18	
symbolic	3.2.1	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\symbolic-3.2.1	
tablicious	0.4.2	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\tablicious-0.4.2	
tsa	4.6.3	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\tsa-4.6.3	
video	2.1.1	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\video-2.1.1	
windows	1.6.4	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\windows-1.6.4	
zeromq	1.5.6	C:\Program Files\GNU Octave\Octave-9.2.0\mingw64\share\octave\packages\zeromq-1.5.6	

Рис. 2.21: Проверка пакета расширений

11) Вписал код в каждый файл. Основной файл - main. С помощью него запускаются все остальные файлы.

```

1  % coding/main.m
2  % Коммуникация данных signal:
3  pkg load signal;
4  data=[0 1 0 0 1 1 0 0 0 1 1 0];
5  data_sync=[0 0 0 0 0 0 0 0 1 1 1 1 1 1];
6  data_spectre=[0 1 0 1 0 1 0 1 0 1 0 1 0 1];
7  mkdir 'signal';
8  mkdir 'sync';
9  mkdir 'spectre';
10 axis("auto");
11
12 % Вычисления компонентов
13 wave=unipolar(data);
14 plot(wave);
15 ylim([-1 6]);
16 title('Unipolar');
17 print 'signal/unipolar.png';
18 % Коммуникация AMI
19 wave=ami(data);
20 plot(wave);
21 title('AMI');
22 print 'signal/ami.png';
23 % Коммуникация NRZ
24 wave=bipolarnrz(data);
25 plot(wave);
26 title('Bipolar Non-Return to Zero');
27 print 'signal/bipolarnrz.png';
28 % Коммуникация RZ
29 wave=bipolarrrz(data);
30 plot(wave);
31 title('Bipolar Return to Zero');
32 print 'signal/bipolarrrz.png';
33 % Математическое преобразование
34 wave=manchester(data);
35 plot(wave);
36 title('Manchester');
37 print 'signal/manchester.png';
38 % Интегрирование математического преобразования
39 wave=diffmanc(data);
40 plot(wave);
41 title('Differential Manchester');
42 print 'signal/diffmanc.png';
43 % Вычисления компонентов
44 wave=unipolar(data_sync);
45 plot(wave);
46 ylim([-1 6]);
47 title('Unipolar');
48 print 'sync/unipolar.png';
49 % Коммуникация AMI
50 wave=ami(data_sync);
51 plot(wave);
52 title('AMI');
53 print 'sync/ami.png';
54 % Коммуникация NRZ
55 wave=bipolarnrz(data_sync);
56 plot(wave);
57 title('Bipolar Non-Return to Zero');
58 print 'sync/bipolarnrz.png';
59 % Коммуникация RZ
60 wave=bipolarrrz(data_sync);
61 plot(wave);
62 title('Bipolar Return to Zero');
63 print 'sync/bipolarrrz.png';
64 % Математическое преобразование
65 wave=manchester(data_sync);
66 plot(wave);
67 title('Manchester');
68 print 'sync/manchester.png';
69 % Интегрирование математического преобразования
70 wave=diffmanc(data_sync);
71 plot(wave);
72 title('Differential Manchester');
73 print 'sync/diffmanc.png';
74 % Вычисления компонентов
75 wave=unipolar(data_spectre);
76 spectre=calcspectre(wave);
77 title('Unipolar');
78 print 'spectre/unipolar.png';
79 % Коммуникация AMI
80 wave=ami(data_spectre);
81 spectre=calcspectre(wave);
82 title('AMI');
83 print 'spectre/ami.png';

```

Рис. 2.22: Код main.m


```

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

```

Рис. 2.23: Код maptowave.m

```

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

```

Рис. 2.24: Код unipolar.m

```

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

```

Рис. 2.25: Код ami.m

```

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

```

Рис. 2.26: Код bipolarnrz.m

```

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);

```

Рис. 2.27: Код bipolarrz.m

```

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);

```

Рис. 2.28: Код manchester.m

```

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);

```

Рис. 2.29: Код diffmanc.m


```

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;
0 f = 1000*(0:Fd2)/Fd;
1 plot(f,spectre(1:Fd3));
2 xlabel('Frequency (Hz)');

```

Рис. 2.30: Код calcspectre.m

- 12) После запуска файла main у меня создались 3 каталога и создались несколько графиков, которые соответствуют каждому файлу.

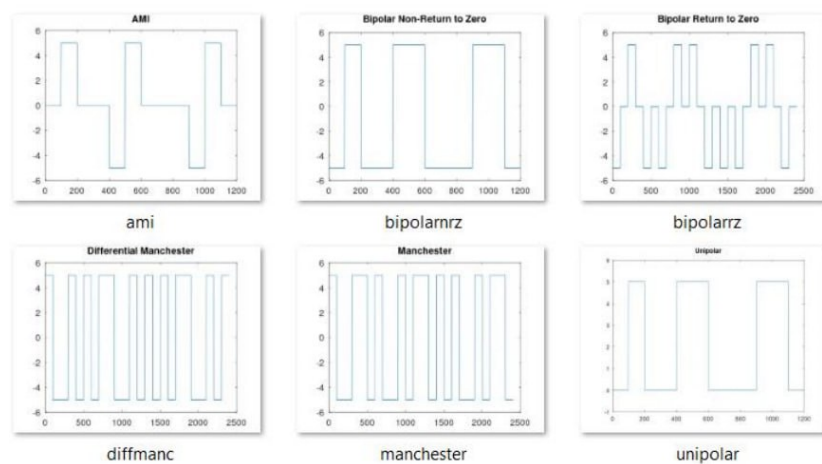


Рис. 2.31: Рисунки в каталоге signal

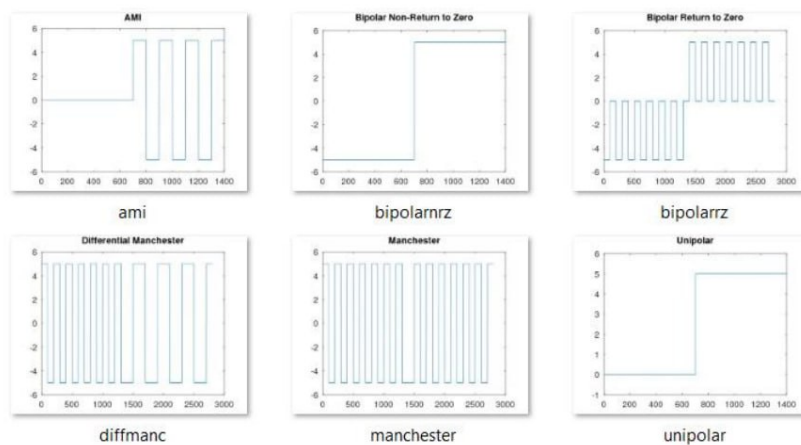


Рис. 2.32: Рисунки в каталоге sync

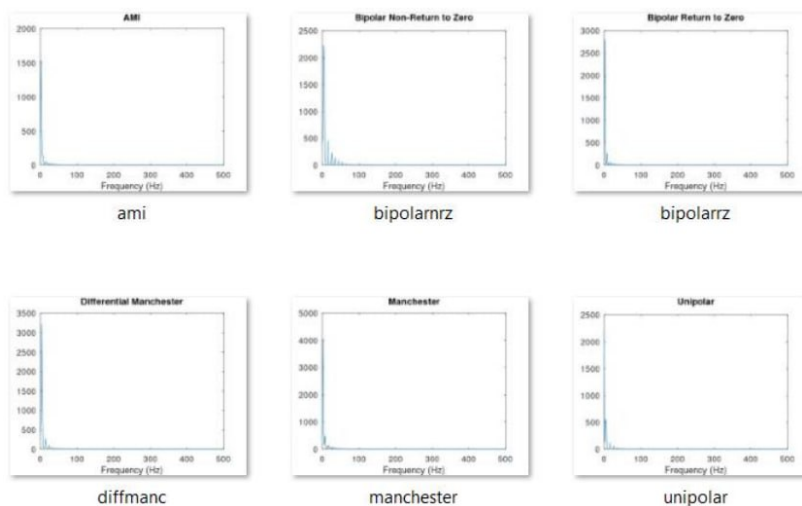


Рис. 2.33: Рисунки в каталоге spectre

3 Вывод

Я изучил методы кодирования и модуляции сигналов с помощью языка Octave. Определил спектры и параметры сигнала. Продемонстрировал принципы модуляции сигналов на примере аналоговой амплитудной модуляции. Исследовал свойства самосинхронизации сигнала.