

# **Отчет по лабораторной работе №1**

**Дисциплина: Сетевые технологии**

**Иванов Сергей Владимирович**

# **Содержание**

<b>1 Цель работы</b>	<b>5</b>
<b>2 Выполнение лабораторной работы</b>	<b>6</b>
2.1 Построение графиков в Octave . . . . .	6
2.2 Разложение импульсного сигнала в частичный ряд Фурье . . . . .	8
2.3 Определение спектра и параметров сигнала . . . . .	10
2.4 Амплитудная модуляция . . . . .	12
2.5 Кодирование сигнала. Исследование свойства самосинхронизации сигнала . . . . .	13
2.6 Задание с другой частотой дискретизации. . . . .	24
<b>3 Выводы</b>	<b>26</b>

# Список иллюстраций

2.1 Программа plot_sin.m . . . . .	6
2.2 Построенный график . . . . .	7
2.3 Форматы .eps и .png . . . . .	7
2.4 График с cos . . . . .	8
2.5 Код meandr.m . . . . .	9
2.6 Файл png . . . . .	9
2.7 Меандр через синусы . . . . .	10
2.8 График сигналов разной частоты . . . . .	10
2.9 График спектра сигнала . . . . .	11
2.10 Скорректированный график спектра . . . . .	11
2.11 Графики суммарного сигнала . . . . .	12
2.12 Каталог modulation . . . . .	12
2.13 Графики амплитудной модуляции . . . . .	13
2.14 Создание файлов . . . . .	13
2.15 Проверка пакета . . . . .	14
2.16 Код main.m . . . . .	14
2.17 Файл maptowave.m . . . . .	14
2.18 Прописывание функций . . . . .	15
2.19 Полученные графики . . . . .	15
2.20 Униполярное кодирование . . . . .	16
2.21 Кодирование AMI . . . . .	16
2.22 Кодирование NRZ . . . . .	17
2.23 Кодирование RZ . . . . .	17
2.24 Манчестерское кодирование . . . . .	18
2.25 Дифференциальное манчестерское кодирование . . . . .	18
2.26 Униполярное кодирование: нет самосинхронизации . . . . .	19
2.27 Кодирование AMI: самосинхронизация при наличии сигнала . . . . .	19
2.28 Кодирование NRZ: нет самосинхронизации . . . . .	20
2.29 Кодирование RZ: есть самосинхронизация . . . . .	20
2.30 Манчестерское кодирование: есть самосинхронизация . . . . .	21
2.31 Дифференциальное манчестерское кодирование: есть самосинхронизация . . . . .	21
2.32 Униполярное кодирование: спектр сигнала . . . . .	22
2.33 Кодирование AMI: спектр сигнала . . . . .	22
2.34 Кодирование NRZ: спектр сигнала . . . . .	23
2.35 Кодирование RZ: спектр сигнала . . . . .	23
2.36 Манчестерское кодирование: спектр сигнала . . . . .	24

2.37 Дифференциальное манчестерское кодирование: спектр сигнала	24
2.38 Задание с другой частотой дискретизации . . . . .	25

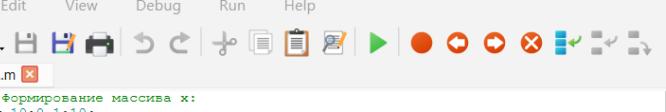
# **1 Цель работы**

Изучение методов кодирования и модуляции сигналов с помощью высокого-уровнего 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

## 2.1 Построение графиков в Octave

Запустим Octave с оконным интерфейсом. Перейдем в окно редактора. Воспользовавшись меню или комбинацией клавиш `ctrl + n` создадим новый сценарий. Сохраним его в рабочий каталог с именем, `plot_sin.m`. В окне редактора повторим листинг по построению графика функции  $y = \sin(x) + 1/3 * \sin(3x) + 1/5 * \sin(5x)$  на интервале  $[-10; 10]$ : (рис. 1).



The screenshot shows the MATLAB Editor interface with the following details:

- Editor Tab:** The top bar has tabs for "Editor", "File", "Edit", "View", "Debug", "Run", and "Help".
- Toolbar:** Below the tabs are various icons for file operations (New, Open, Save, Print, Find, Copy, Paste, etc.) and toolboxes.
- Code Area:** The main window contains the following MATLAB code:

```
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:
18 print("plot-sin.png");
```

Рис. 2.1: Программа plot sin.m

Запустим сценарий на выполнение. В качестве результата выполнения кода открылось окно с построенным графиком (рис. 2) и в рабочем каталоге появились файлы с графиками в форматах .eps, .png. (рис. 3).

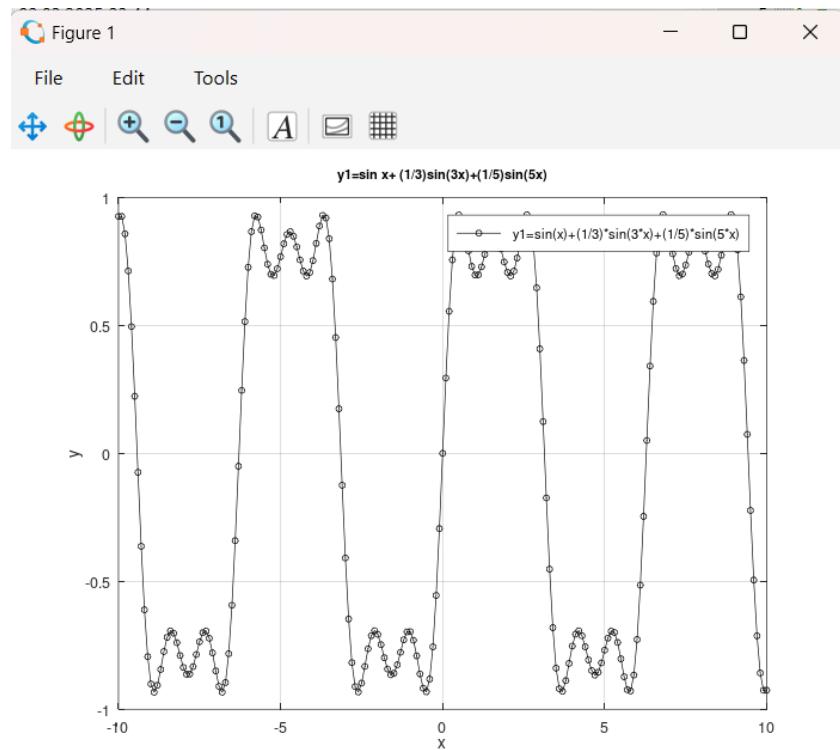


Рис. 2.2: Построенный график

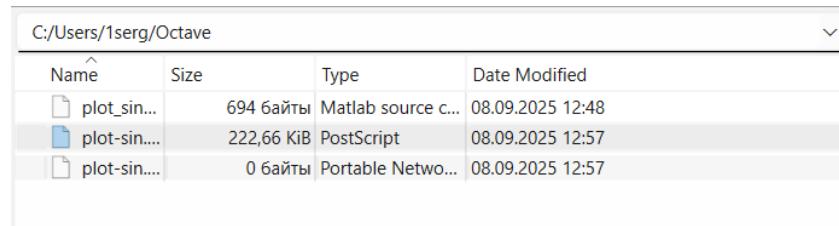


Рис. 2.3: Форматы .eps и .png

Сохраним сценарий под другим названием и изменим его так, чтобы на одном графике располагались отличающиеся по типу линий графики функций  $y_1 = \sin(x) + 1/3 * \sin(3x) + 1/5 * \sin(5x)$ ,  $y_2 = \cos(x) + 1/3 * \cos(3x) + 1/5 * \cos(5x)$  (рис. 4)

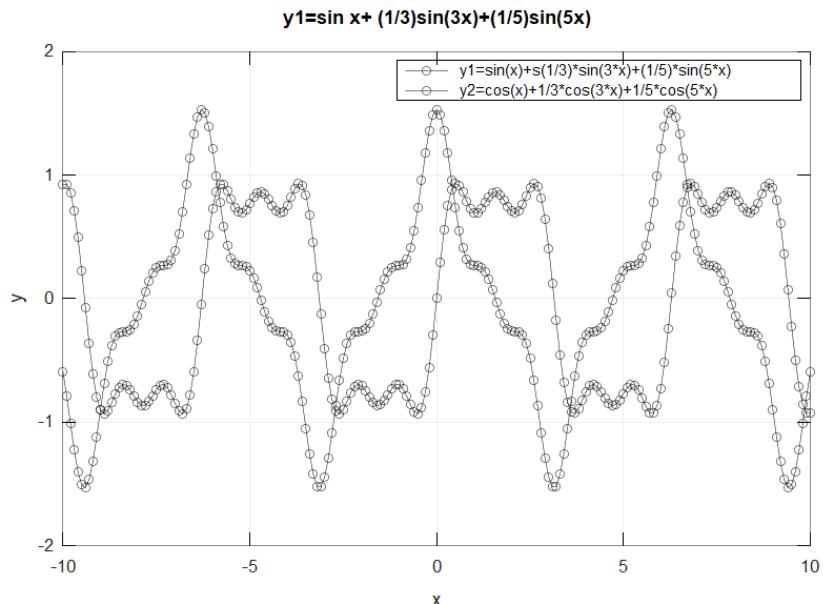


Рис. 2.4: График с cos

## 2.2 Разложение импульсного сигнала в частичный ряд

### Фурье

Создадим новый сценарий и сохраним его в рабочий каталог с именем meandr.m. В коде повторим листинг по построению графиков меандра. (рис. 5)

```

meandr.m
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
26 % Экспорт рисунка в файл .png:
27 print("meandr.png");

```

Рис. 2.5: Код meandr.m

Экспортируем полученный график в файл в формате .png. (рис. 6)

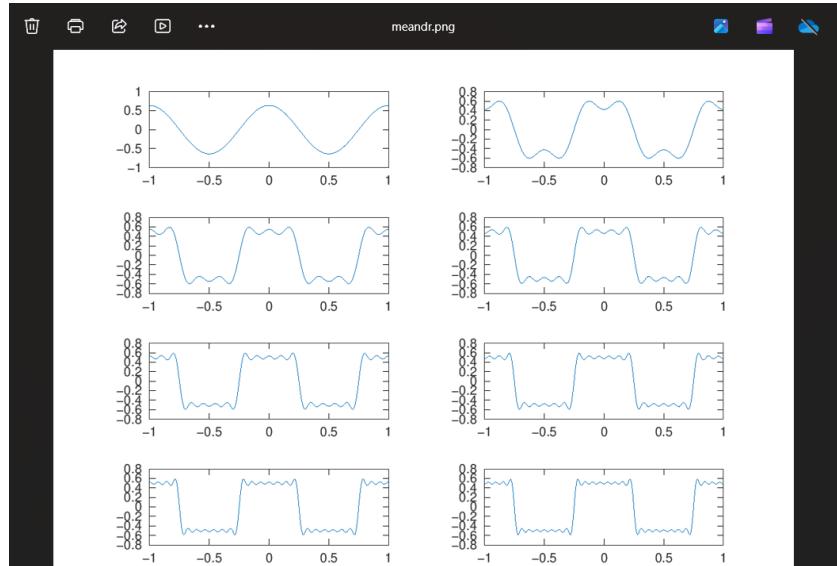


Рис. 2.6: Файл png

Скорректируем код для реализации меандра через синусы. Получим соответствующие графики. (рис. 7)

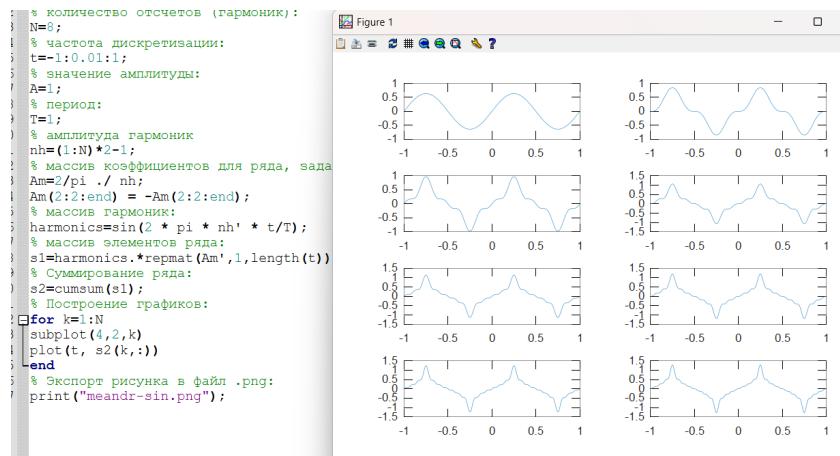


Рис. 2.7: Меандр через синусы

## 2.3 Определение спектра и параметров сигнала

В рабочем каталоге создадим каталог `spectre1` и в нём новый сценарий с именем, `spectre.m`. В коде повторим листинг по построению сигналов разной частоты. Получим график. (рис. 8)

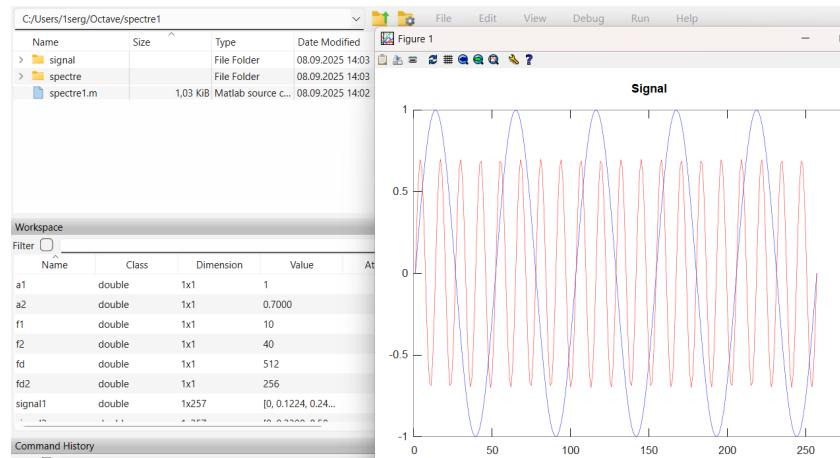


Рис. 2.8: График сигналов разной частоты

Далее добавим код для нахождения спектров сигналов с помощью быстрого преобразования Фурье и получим график. (рис. 9)

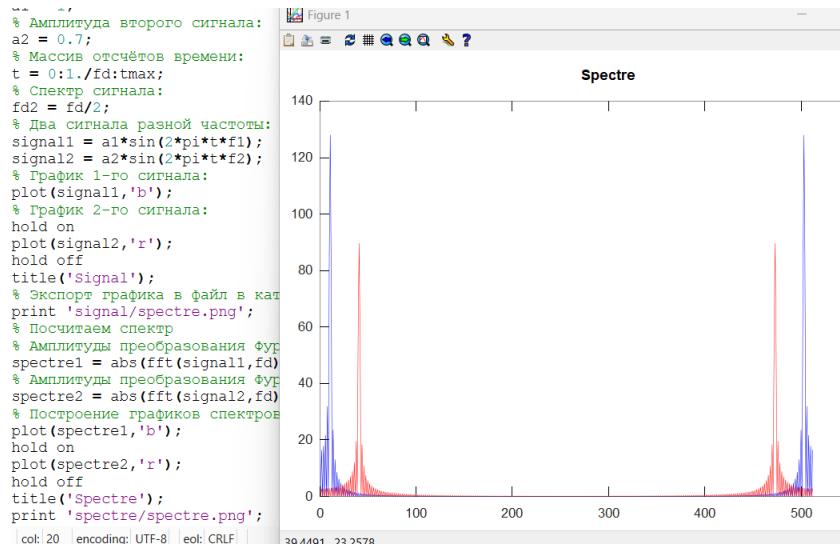


Рис. 2.9: График спектра сигнала

Скорректируем график спектра: отбросим дублирующие отрицательные частоты, а также примем в расчёт то, что на каждом шаге вычисления быстрого преобразования Фурье происходит суммирование амплитуд сигналов. (рис. 10)

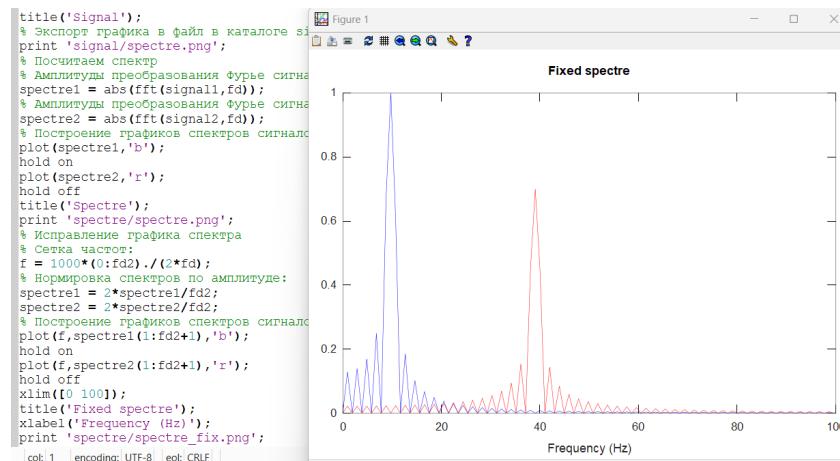


Рис. 2.10: Скорректированный график спектра

Найдем спектр суммы рассмотренных сигналов, создав каталог `spectr_sum` и файл в нём `spectre_sum.m`. В результате получился аналогичный предыдущему результат, т.е. спектр суммы сигналов равен сумме спектров сигналов, что вытекает из свойств преобразования Фурье. (рис. 11)

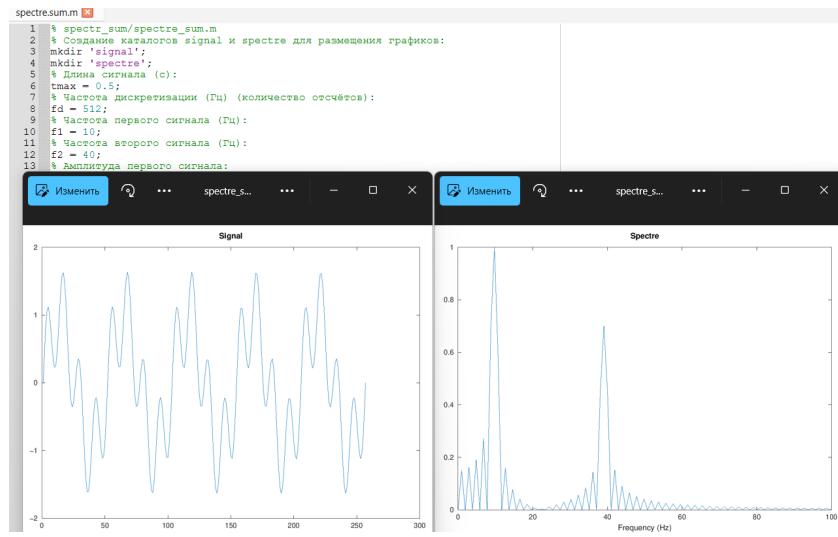


Рис. 2.11: Графики суммарного сигнала

## 2.4 Амплитудная модуляция

В рабочем каталоге создадим каталог `modulation` и в нём новый сценарий с именем `am.m`. Добавим в файле `am.m` код из листинга. (рис. 12)

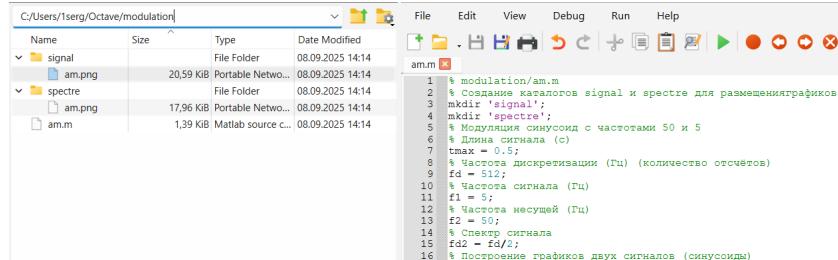


Рис. 2.12: Каталог `modulation`

В результате получаем, что спектр произведения представляет собой свёртку спектров (рис. 13)

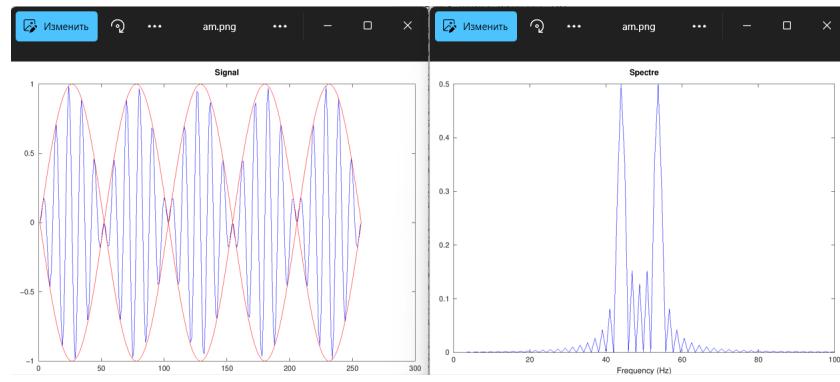


Рис. 2.13: Графики амплитудной модуляции

## 2.5 Кодирование сигнала. Исследование свойства самосинхронизации сигнала

В рабочем каталоге создадим каталог coding и в нём файлы main.m, maptowave.m, unipolar.m, ami.m, bipolarnrz.m, bipolarrz.m, manchester.m, diffmanc.m, calcspctre.m (рис. 14)

C:/Users/1serg/Octave/coding			
Name	Size	Type	Date Modified
ami.m	0 байты	Matlab source c...	08.09.2025 14:19
bipolarnrz.m	0 байты	Matlab source c...	08.09.2025 14:19
bipolarrz.m	0 байты	Matlab source c...	08.09.2025 14:19
calcspctre.m	0 байты	Matlab source c...	08.09.2025 14:20
diffmanc.m	0 байты	Matlab source c...	08.09.2025 14:20
main.m	0 байты	Matlab source c...	08.09.2025 14:18
manchester.m	0 байты	Matlab source c...	08.09.2025 14:19
maptowave.m	0 байты	Matlab source c...	08.09.2025 14:18
unipolar.m	0 байты	Matlab source c...	08.09.2025 14:18

Рис. 2.14: Создание файлов

В окне интерпретатора команд проверим, установлен ли пакет расширений signal: » pkg list. Видим, что он установлен. (рис. 15)

0	quaternion		2.4.0	C:\Program Files\GNU
	queueing		1.2.8	C:\Program Files\GNU
	signal		1.4.6	C:\Program Files\GNU

Рис. 2.15: Проверка пакета

В файле main.m подключим пакет signal и скопируем программу из листинга. (рис. 16)

```

main.m
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 data_sync=[0 0 0 0 0 0 0 1 1 1 1 1];
8 % Входная кодовая последовательность для построения
9 data_spectre=[0 1 0 1 0 1 0 1 0 1 0 1];
10 % Создание каталогов signal, sync и spectre для работы
11 mkdir 'signal';

```

Рис. 2.16: Код main.m

В файле maptowave.m пропишем функцию, которая по входному битовому потоку строит график сигнала. (рис. 17)

```

main.m
maptowave.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. 2.17: Файл maptowave.m

В файлах unipolar.m, ami.m, bipolarnrz.m, bipolarrz.m, manchester.m, diffmanc.m пропишите соответствующие функции преобразования кодовой последовательности data с вызовом функции maptowave для построения соответствующего графика. (рис. 18)

```

1 % calcspectre.m
2 % Функция построения спектра сигнала:
3 function spectre = calcspectre(wave)
4 % Королькова А. В., Кулабов Д. С. Сетевые технологии. Лабораторный практикум 31
5 % Частота дискретизации (Гц):
6 Fd = 512;
7 Fd2 = Fd/2;
8 Fd3 = Fd/2 + 1;
9 X = fft(wave,Fd);
10 spectre = X.*conj(X)/Fd;
11 f = 1000*(0:Fd2)/Fd;
12 plot(f,spectre(1:Fd3));

```

Рис. 2.18: Прописывание функций

Запустим главный скрипт main.m. В каталоге signal получены файлы с графиками кодированного сигнала, в каталоге sync — файлы с графиками, иллюстрирующими свойства самосинхронизации, в каталоге spectre — файлы с графиками спектров сигналов. (рис. 19)

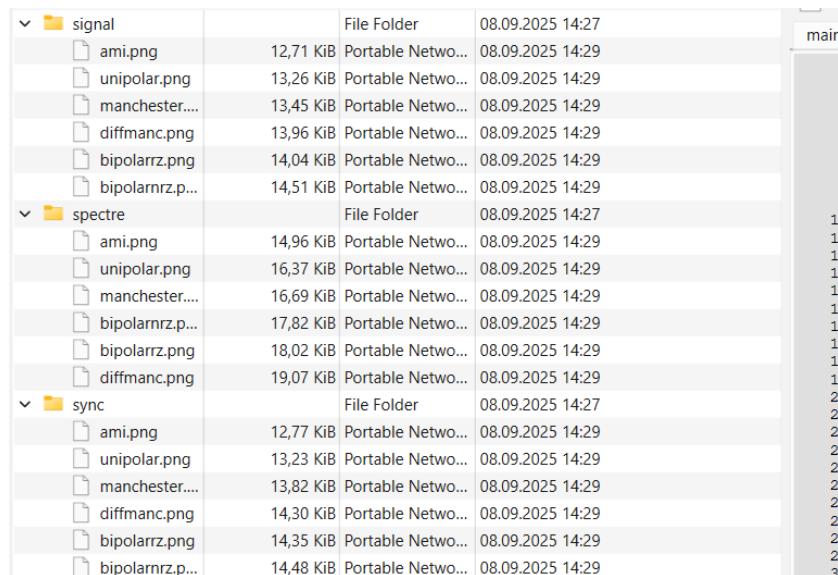


Рис. 2.19: Полученные графики

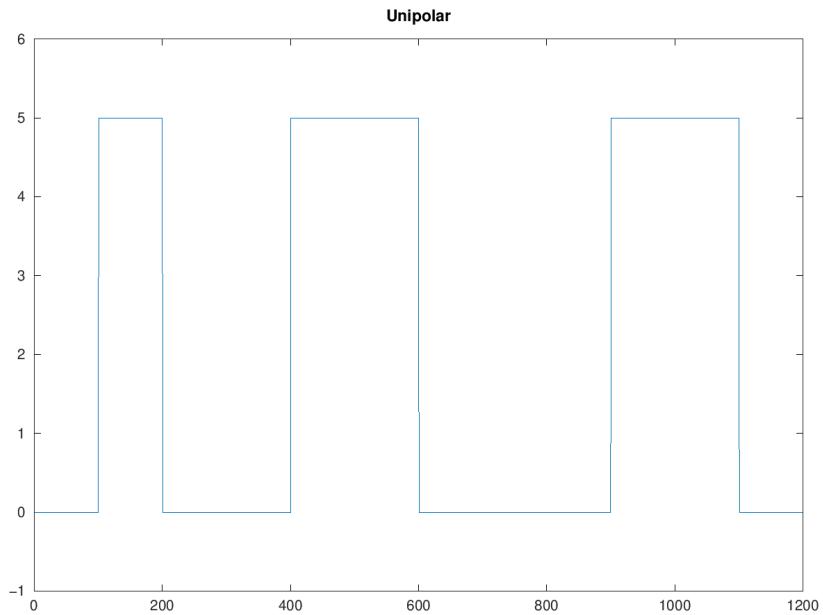


Рис. 2.20: Униполярное кодирование

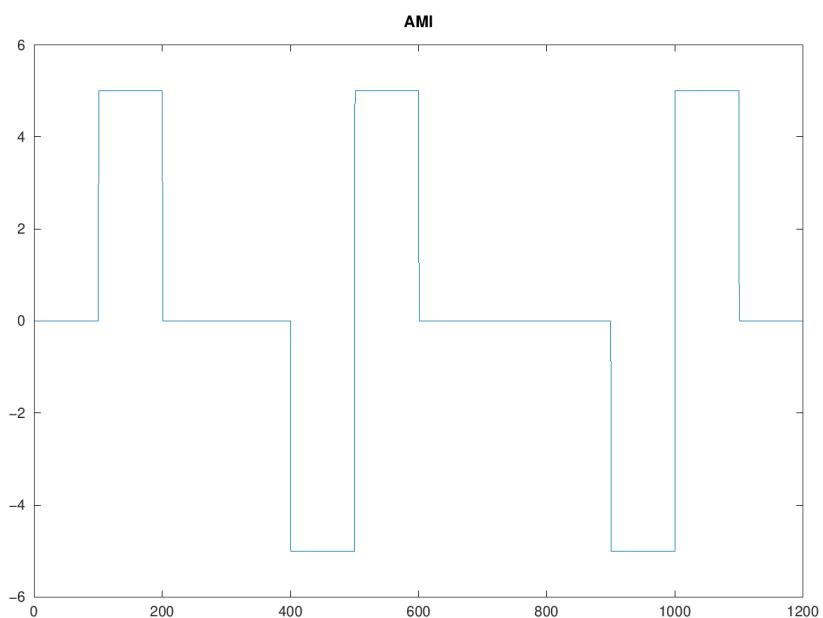


Рис. 2.21: Кодирование AMI

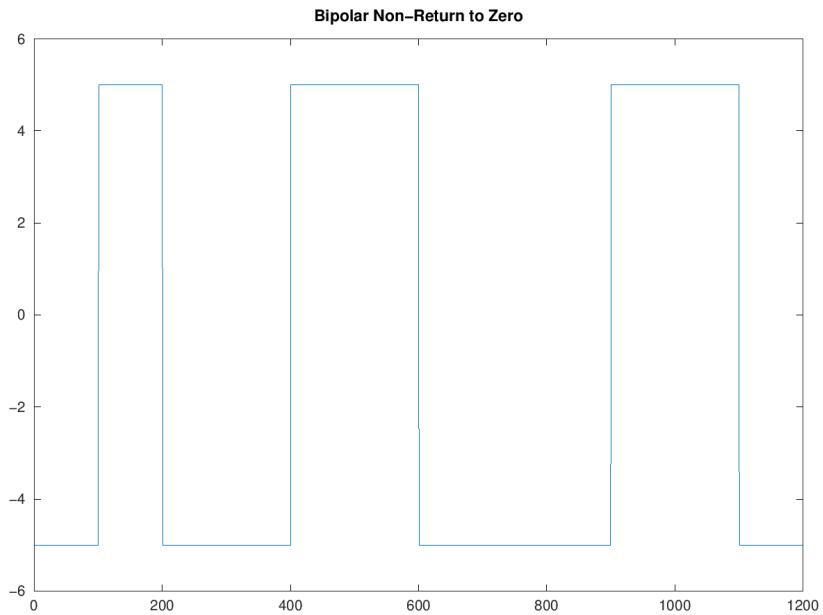


Рис. 2.22: Кодирование NRZ

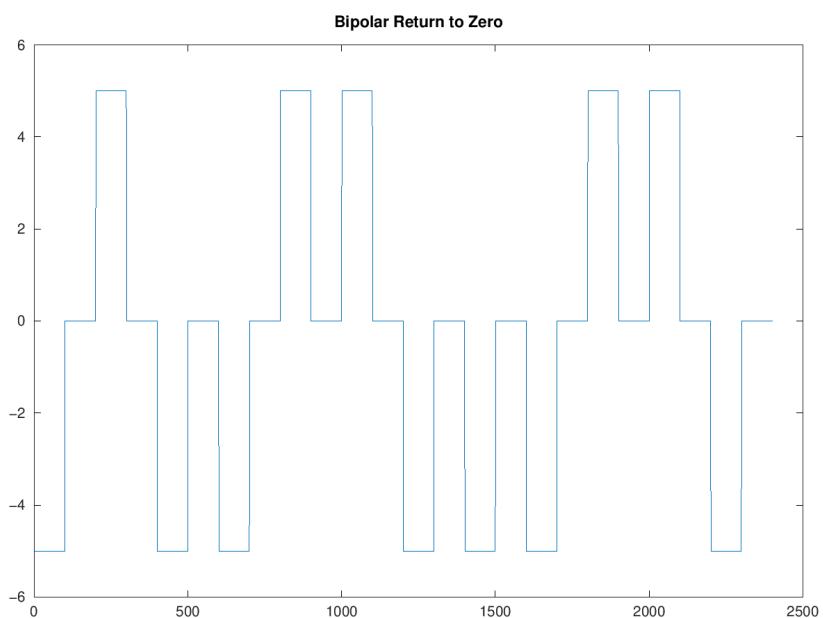


Рис. 2.23: Кодирование RZ

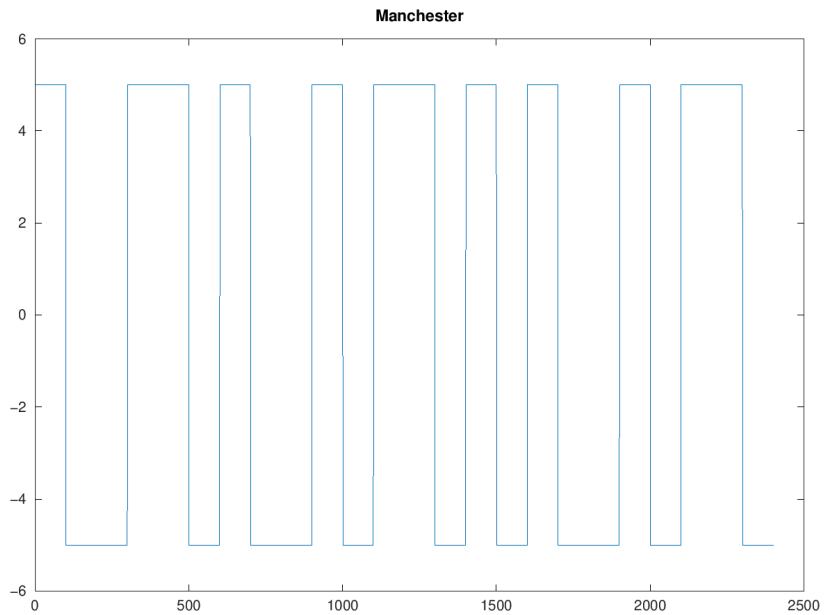


Рис. 2.24: Манчестерское кодирование

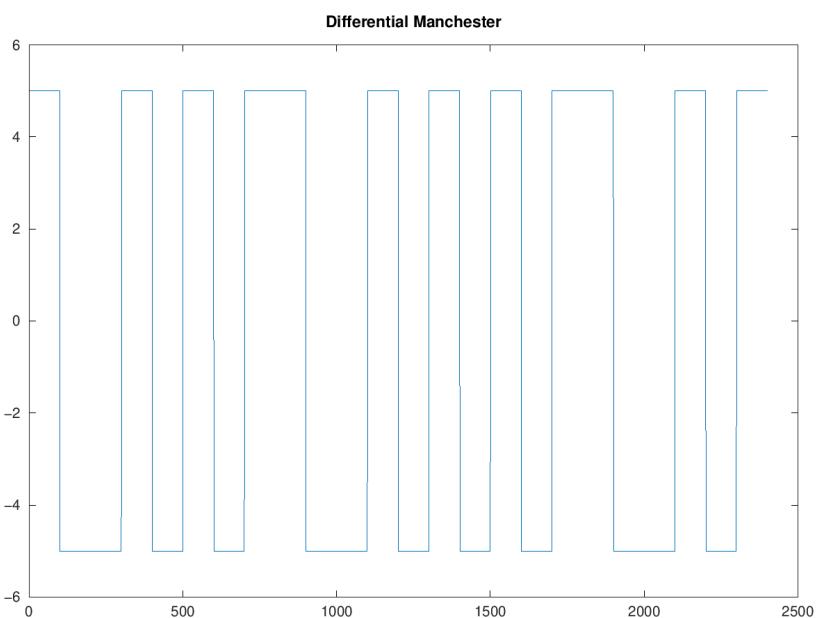


Рис. 2.25: Дифференциальное манчестерское кодирование

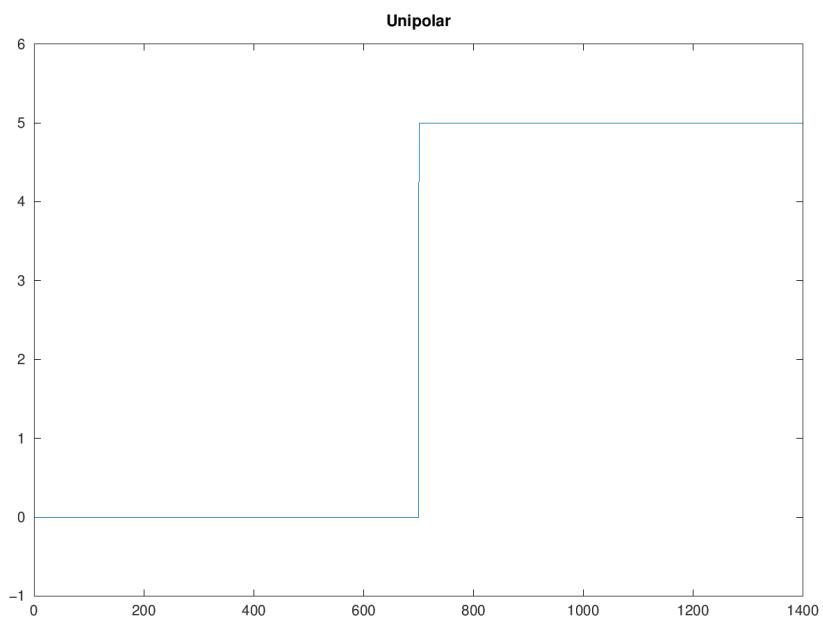


Рис. 2.26: Униполярное кодирование: нет самосинхронизации

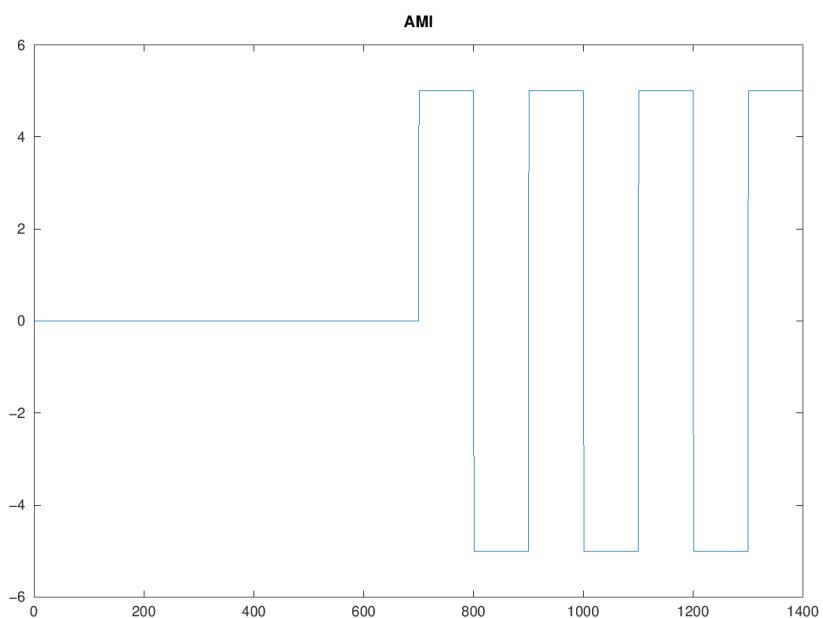


Рис. 2.27: Кодирование АМI: самосинхронизация при наличии сигнала

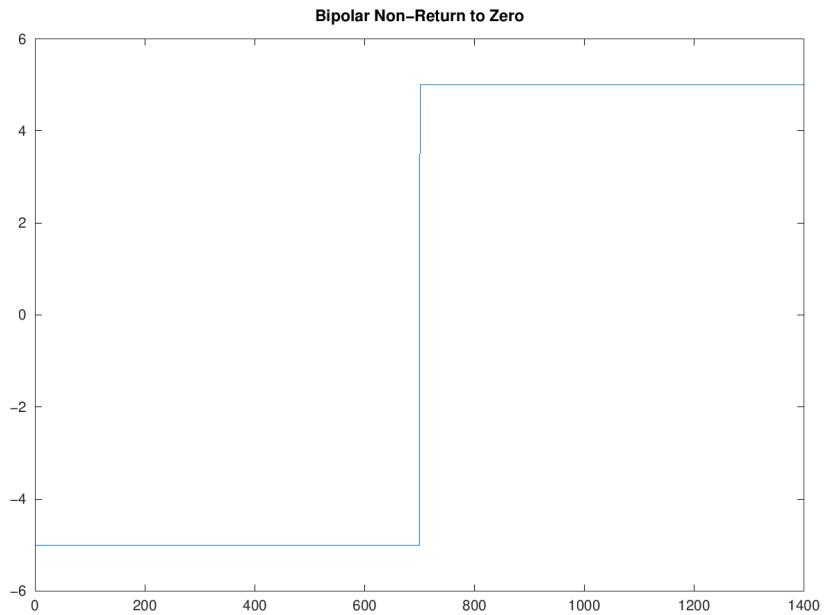


Рис. 2.28: Кодирование NRZ: нет самосинхронизации

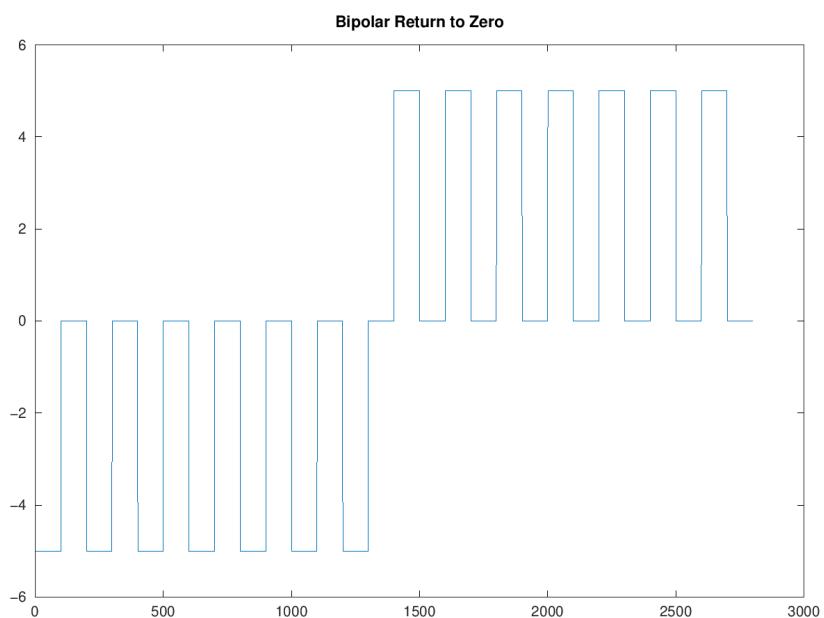


Рис. 2.29: Кодирование RZ: есть самосинхронизация

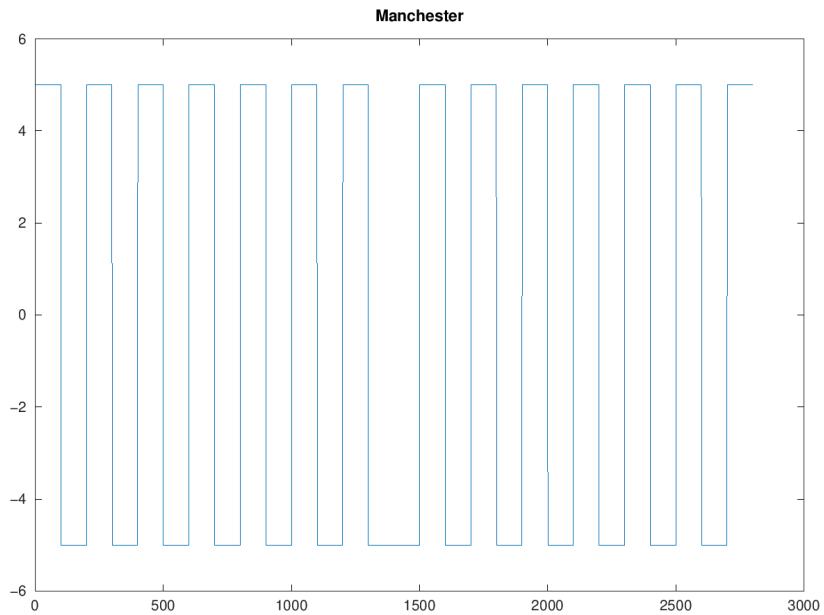


Рис. 2.30: Манчестерское кодирование: есть самосинхронизация

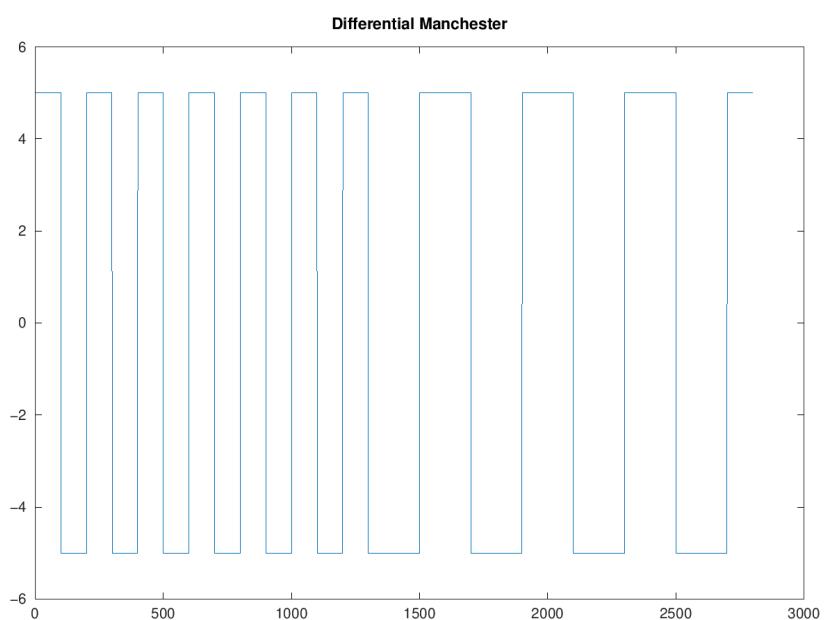


Рис. 2.31: Дифференциальное манчестерское кодирование: есть самосинхронизация

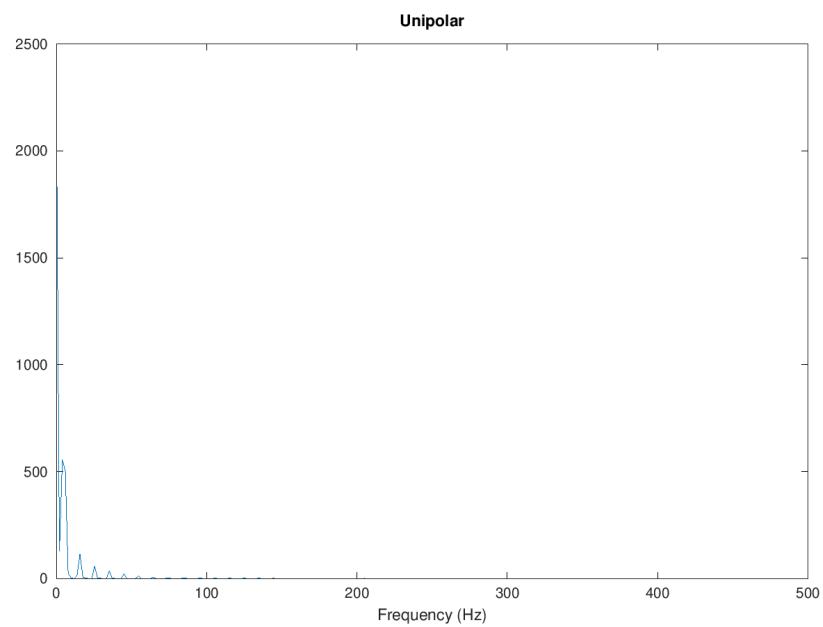


Рис. 2.32: Униполярное кодирование: спектр сигнала

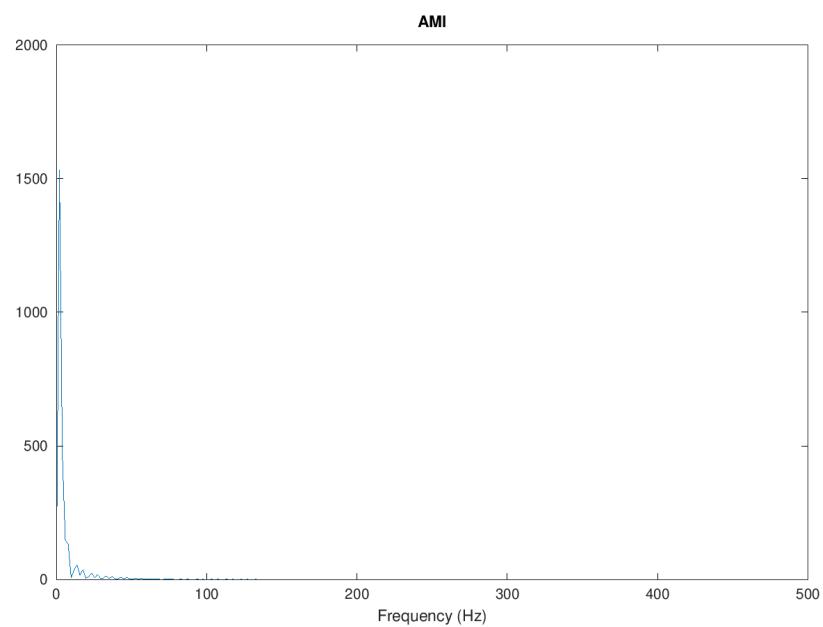


Рис. 2.33: Кодирование AMI: спектр сигнала

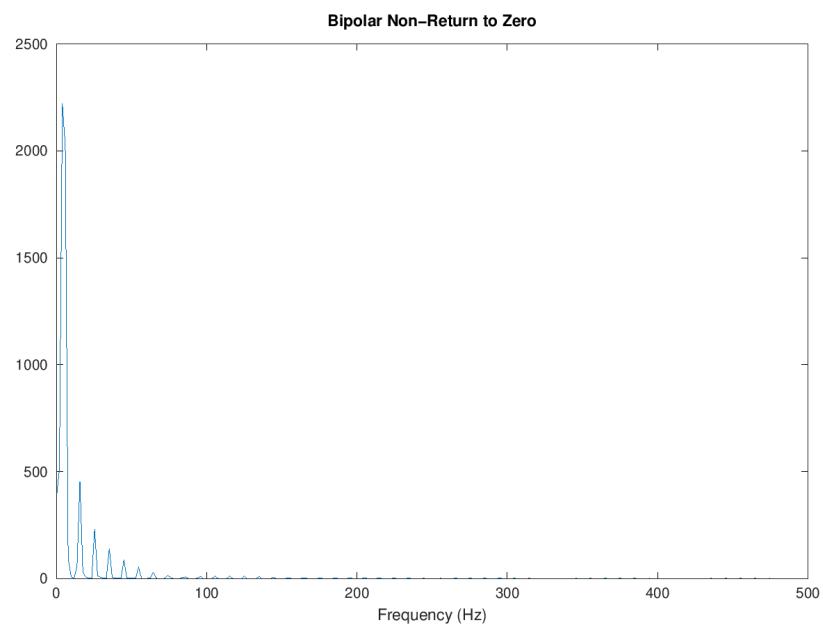


Рис. 2.34: Кодирование NRZ: спектр сигнала

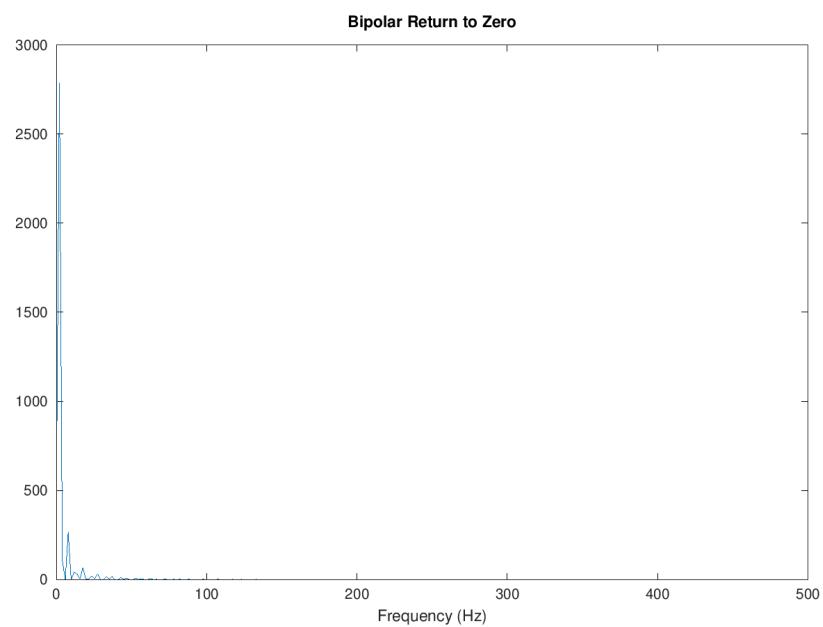


Рис. 2.35: Кодирование RZ: спектр сигнала

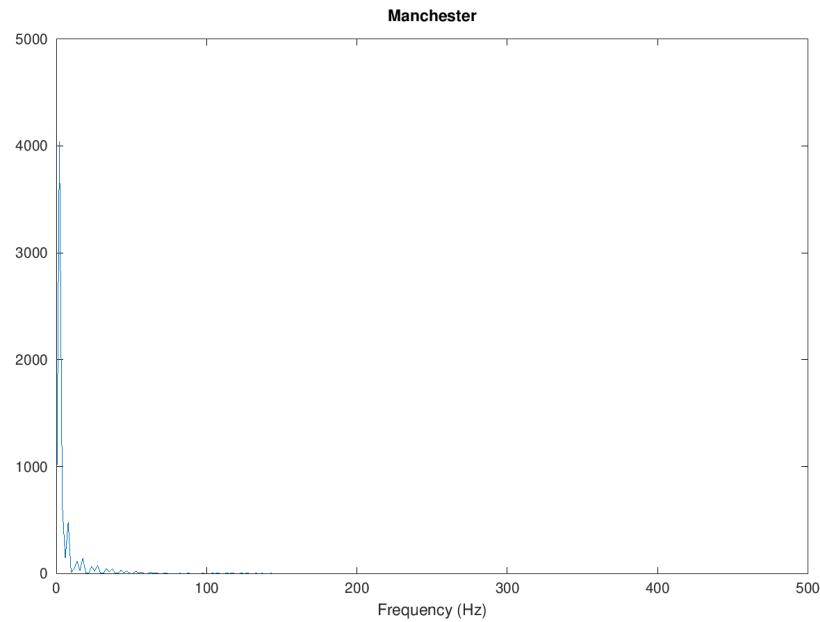


Рис. 2.36: Манчестерское кодирование: спектр сигнала

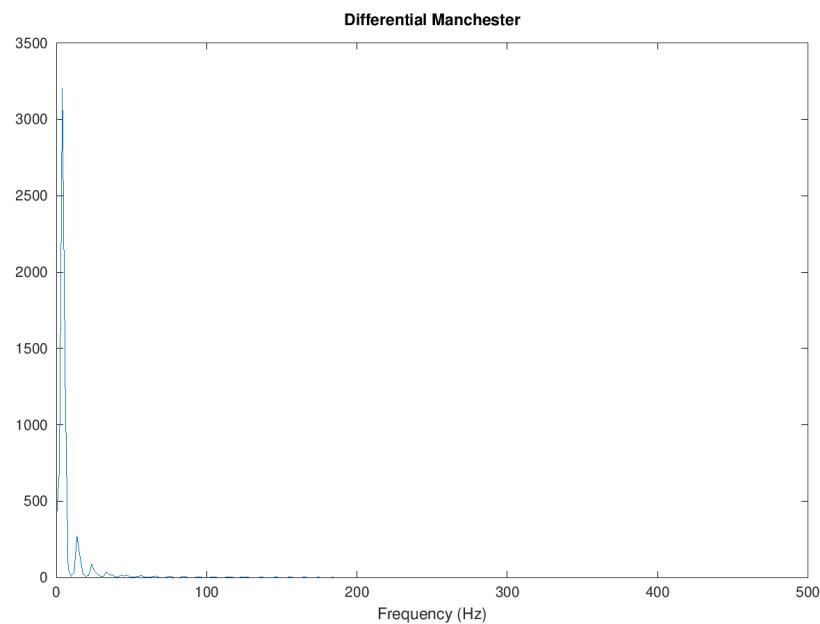


Рис. 2.37: Дифференциальное манчестерское кодирование: спектр сигнала

## 2.6 Задание с другой частотой дискретизации.

1) Что будет, если взять частоту дискретизации меньше 80 Гц?

При снижении частоты дискретизации ниже 80 Гц высокочастотные составляющие сигнала могут быть утеряны, что приведёт к искажению результатов спектрального анализа.

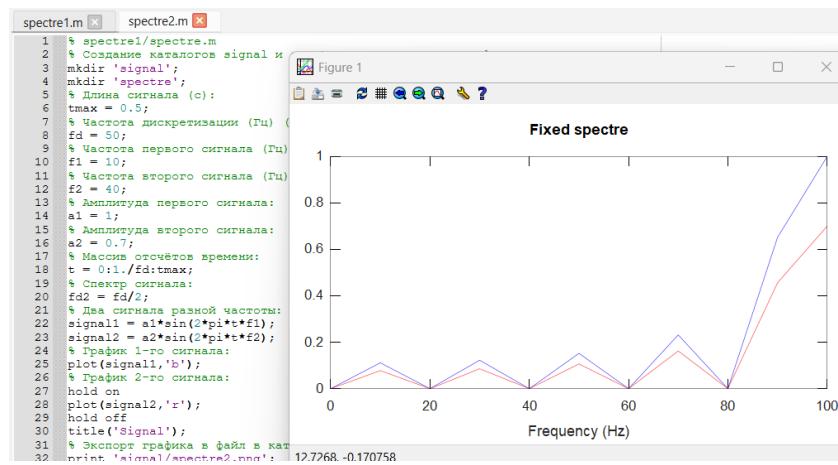


Рис. 2.38: Задание с другой частотой дискретизации

## **3 Выводы**

В рамках лабораторной работы мы изучили методов кодирования и модуляции сигналов с помощью высокоуровнного языка программирования Octave. Определили спектра и параметров сигнала. Продемонстрировали принципы модуляции сигнала на примере аналоговой амплитудной модуляции. Исследовали свойства самосинхронизации сигнала.