

Лабораторная работа №1

Сетевые технологии

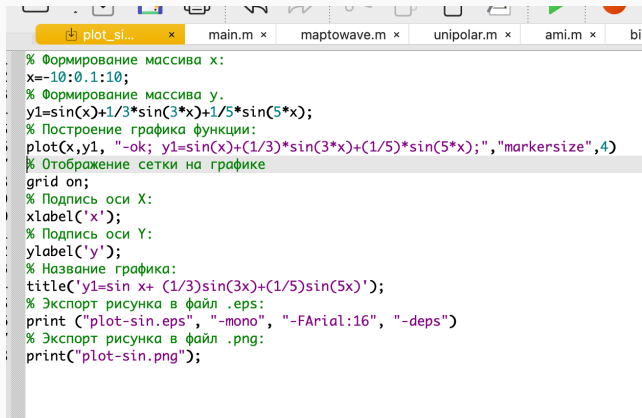
Мишина А. А.

13 сентября 2024

- Целью данной 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.

Выполнение лабораторной работы

Построение графиков в Octave



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

Рис. 1: Листинг файла plot_sin.m

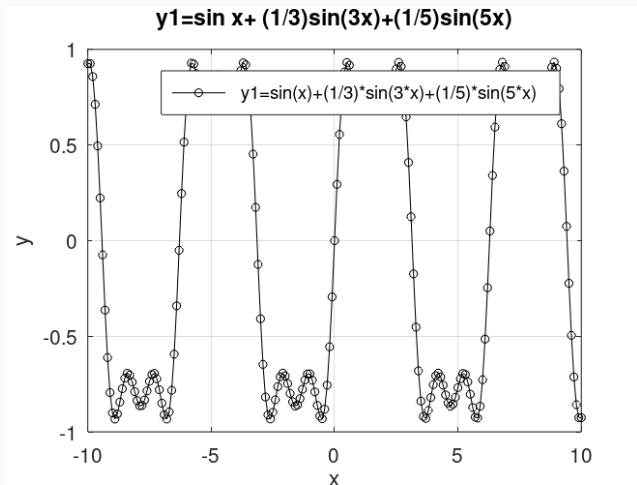


Рис. 2: График функций y_1 на интервале $-10; 10$

Проверка файлов

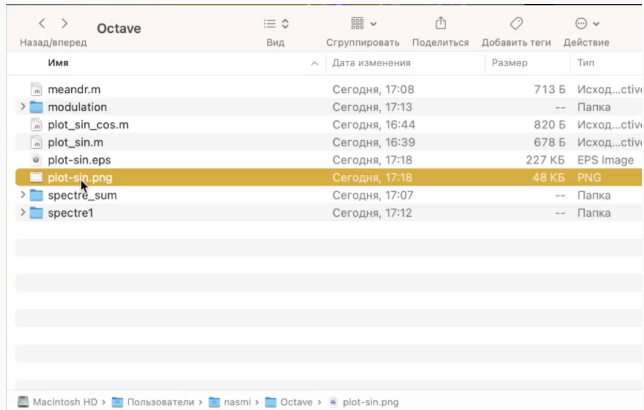


Рис. 3: Файлы .eps, .png

```
1 % Формирование массива x:
2 x=-10:0.1:10;
3 % Формирование массивов y1 и y2.
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 первого 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 7
10
11 % Построение второго графика функции:
12 plot(x,y2, "-; y2=cos(x)+(1/3)*cos(3*x)+(1/5)*cos(5*x);", "markersize", 4)
13 % Отображение сетки на графике
14 grid on;
15 % Подпись оси X:
16 xlabel('x');
17 % Подпись оси Y:
18 ylabel('y');
19 % Экспорт рисунка в файл .eps:
20 print ("plot-sin-cos.eps", "-mono", "-FArial:16", "-deps")
21 % Экспорт рисунка в файл .png:
22 print("plot-sin-cos.png");
23
```

Рис. 4: Листинг файла plot_sin_cos.m

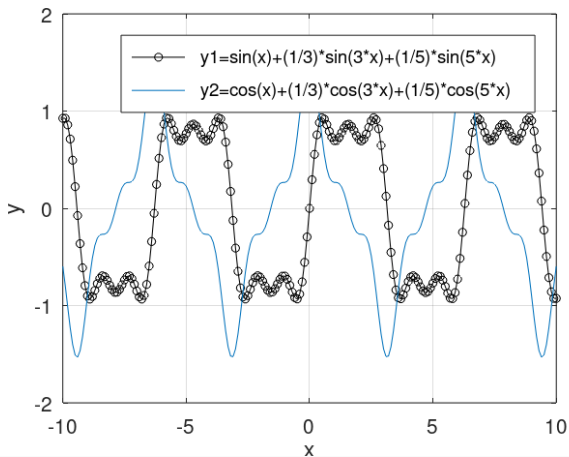


Рис. 5: График функций y_1 и y_2 на интервале $-10; 10$

Разложение импульсного сигнала в частичный ряд Фурье

- Создадим новый сценарий meandr.m. В коде зададим начальные значения. Вычислим амплитуду гармоник и заполним массивы гармоник и элементов ряда. Далее задаём массив значений гармоник массив элементов ряда. Также экспортируем полученный график в файл в формате .png

```
% meandr.m
% количество отсчетов (гармоник):
N=8;
% частота дискретизации:
t=-1:0.01:1;
% значение амплитуды:
A=1;
% период:
T=1;
% амплитуда гармоник
nh=(1:N)*2-1;
% массив коэффициентов для ряда, заданного через cos:
Am=2/pi ./ nh;
Am(2:2:end) = -Am(2:2:end);
% массив гармоник:
harmonics=cos(2 * pi * nh' * t/T);
% массив элементов ряда:
s1=harmonics.*repmat(Am',1,length(t));
% суммирование ряда:
s2=cumsum(s1);
% Построение графиков:
for k=1:N
    subplot(4,2,k)
    plot(t, s2(k,:))
end
print("plot-meandr-cos.png");
```

Рис. 6: Листинг файла meandr.m

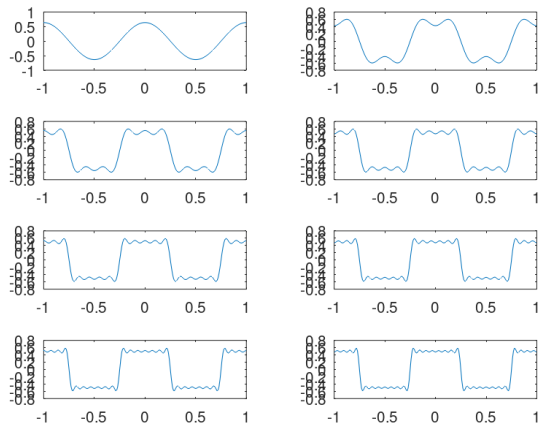


Рис. 7: Меандр через косинусы

```
% meandr.m
% количество отсчетов (гармоник):
N=8;
% частота дискретизации:
t=-1:0.01:1;
% значение амплитуды:
A=1;
% период:
T=1;
% амплитуда гармоник
nh=(1:N)*2-1;
% массив коэффициентов для ряда, заданного через sin:
Am=2/pi ./ nh;
Am(2:2:end) = -Am(2:2:end);
% массив гармоник:
harmonics=sin(2 * pi * nh' * t/T);
% массив элементов ряда:
s1=harmonics.*repmat(Am',1,length(t));
% Суммирование ряда:
s2=cumsum(s1);
% Построение графиков:
for k=1:N
    subplot(4,2,k)
    plot(t, s2(k,:))
end
print("plot-meandr-sin.png");
```

Рис. 8: Листинг файла meandr.m

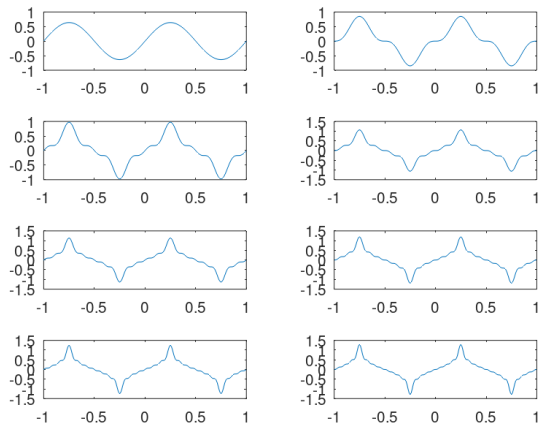


Рис. 9: Меандр через синусы

Определение спектра и параметров сигнала

```
spectre.m x main.m x maptowave.m x unipola
% 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';
```

Рис. 10: Листинг файла spectre.m

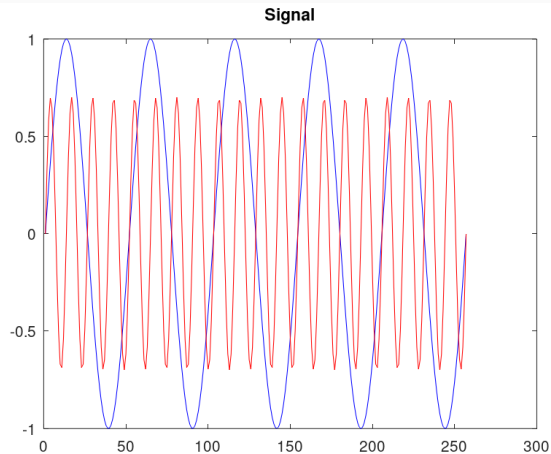


Рис. 11: Графики сигналов разной частоты

```
% 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';

% Исправление графика спектра
% Сетка частот:
f = 1000*(0:fd2)./(2*fd);
% Нормировка спектров по амплитуде:
spectre1 = 2*spectre1/fd2;
spectre2 = 2*spectre2/fd2;
% Построение графиков спектров сигналов:
plot(f,spectre1(1:fd2+1),'b');
hold on
plot(f,spectre2(1:fd2+1),'r');
hold off
xlim([0 100]);
title('Fixed spectre');
xlabel('Frequency (Hz)');
print 'spectre/spectre_fix.png';
```

Рис. 12: Листинг файла spectre.m

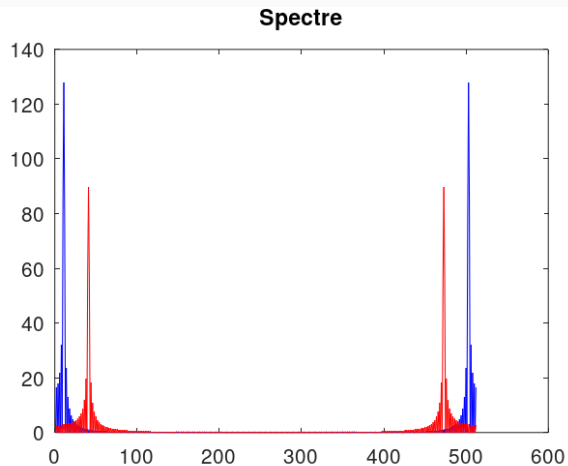


Рис. 13: График спектра синусоидальных сигналов

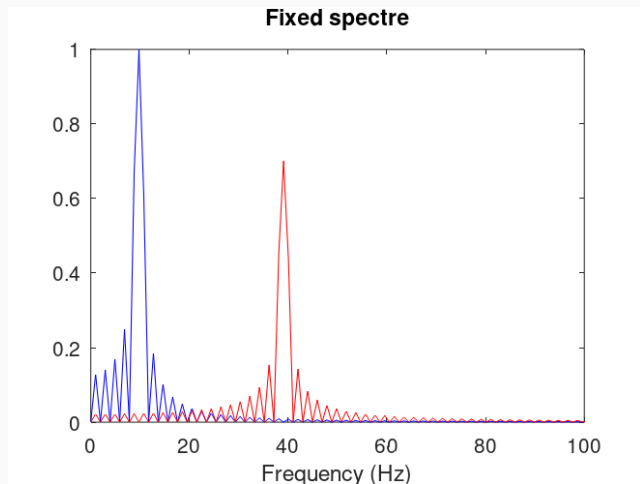


Рис. 14: Исправленный график спектров синусоидальных сигналов

```
spectre_sum.m  main.m  maptowave.m  unipc

% spectre_sum/spectre_sum.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;
% Спектр сигнала
fd2 = fd/2;
% Сумма двух сигналов (синусоиды) разной частоты:
% Массив отсчётов времени:
t = 0:1./fd:tmax;
signal1 = a1*sin(2*pi*t*f1);
signal2 = a2*sin(2*pi*t*f2);
signal = signal1 + signal2;
plot(signal);
title('Signal');
print 'signal/spectre_sum.png';
% Подсчет спектра:
% Амплитуды преобразования Фурье сигнала:
spectre = fft(signal,fd);
% Сетка частот
f = 1000*(0:fd2)/(2*fd);
% Нормировка спектра по амплитуде:
spectre = 2*sqrt(spectre.*conj(spectre))./fd2;
% Построение графика спектра сигнала:
plot(f, spectre(1:fd2+1))
xlim([0 100]);
title('Spectre');
xlabel('Frequency (Hz)');
print 'spectre/spectre_sum.png';
```

Рис. 15: Листинг файла spectre_sum.m

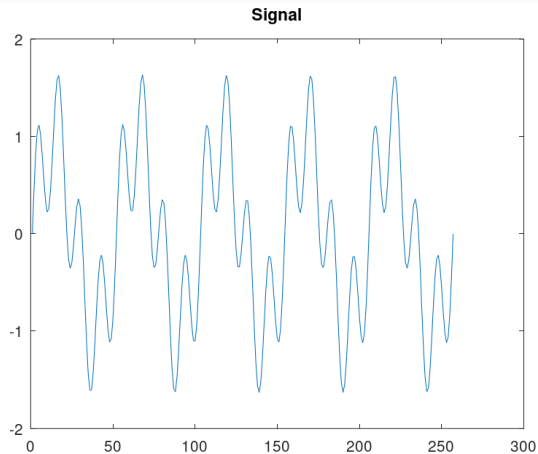


Рис. 16: Суммарный сигнал

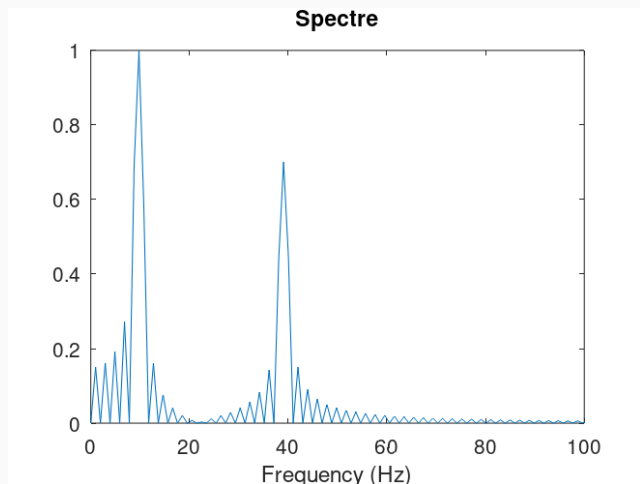


Рис. 17: Спектр суммарного сигнала

Амплитудная модуляция

```
am.m x main.m x maptowave.m x unipolar.r
% modulation/am.m
% Создание каталогов signal и spectre для размещения ч графиков:
mkdir 'signal';
mkdir 'spectre';
% Модуляция синусоид с частотами 50 и 5
% Длина сигнала (с)
tmax = 0.5;
% Частота дискретизации (Гц) (количество отсчетов)
fd = 512;
% Частота сигнала (Гц)
f1 = 5;
% Частота несущей (Гц)
f2 = 50;
% Спектр сигнала
fd2 = fd/2;
% Построение графиков двух сигналов (синусоиды)
% разной частоты
% Массив отсчетов времени:
t = 0:1./fd:tmax;
signal1 = sin(2*pi*t*f1);
signal2 = sin(2*pi*t*f2);
signal = signal1 .* signal2;
plot(signal, 'b');
hold on
% Построение огибающей:
plot(signal1, 'r');
plot(-signal1, 'r');
hold off
title('Signal');
print 'signal/am.png';
% Расчет спектра:
% Амплитуды преобразования Фурье-сигнала:
spectre = fft(signal,fd);
% Сетка частот:
f = 1000*(0:fd2)./(2*f2);
% Нормировка спектра по амплитуде:
spectre = 2*sqrt(spectre.*conj(spectre))./fd2;
% Построение спектра:
plot(f,spectre(1:fd2+1), 'b')
xlim([0 100]);
title('Spectre');
xlabel('Frequency (Hz)');
print 'spectre/am.png';
```

Рис. 18: Листинг файла am.m

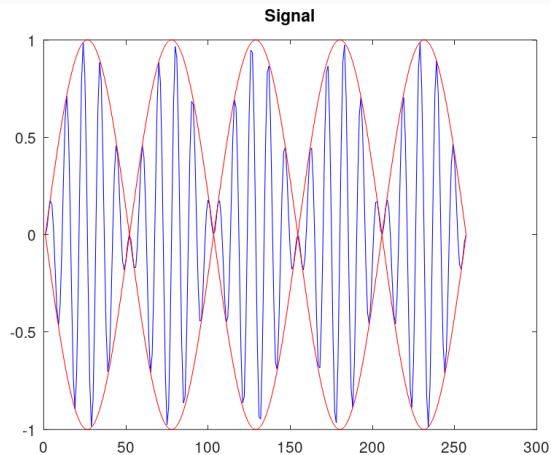


Рис. 19: Сигнал и огибающая при амплитудной модуляции

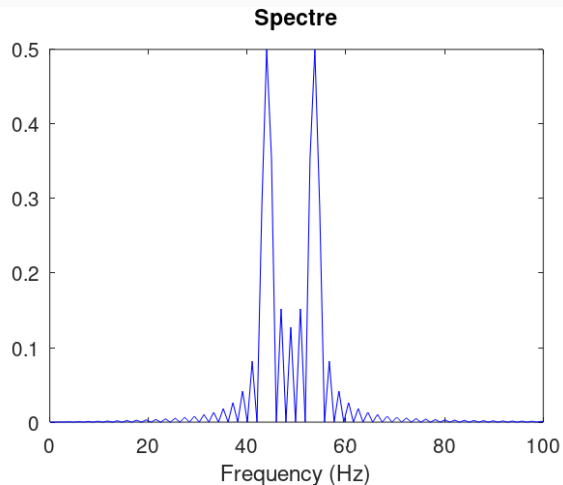


Рис. 20: Спектр сигнала при амплитудной модуляции

Кодирование сигнала.

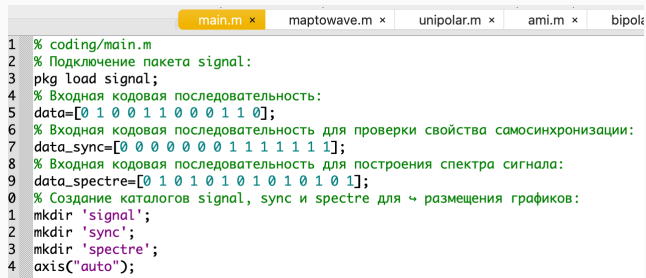
Исследование свойства

самосинхронизации сигнала

- В рабочем каталоге создадим каталог coding и в нём файлы main.m, maptowave.m, unipolar.m, ami.m, bipolarnrz.m, bipolarrrz.m, manchester.m, diffmanc.m, calcspectre.m.
- В окне интерпретатора команд проверяем, установлен ли пакет расширений signal: pkg list. Так как он не установлен, то устанавливаем его: pkg list -forge и pkg install control signal

```
>> pkg list
Package Name | Version | Installation directory
-----+-----+-----
control      | 4.0.1   | /Users/nasmi/.local/share/octave/api-v59/packages/control-4.0.1
signal *!    | 1.4.5   | /Users/nasmi/.local/share/octave/api-v59/packages/signal-1.4.5
>> main
```

Рис. 21: Проверка правильности установки пакета signal



```
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 свойства самосинхронизации:
7 data_sync=[0 0 0 0 0 0 0 1 1 1 1 1 1];
8 % Входная кодовая последовательность для построения спектра сигнала:
9 data_spectre=[0 1 0 1 0 1 0 1 0 1 0 1 0 1];
0 % Создание каталогов signal, sync и spectre для размещения графиков:
1 mkdir 'signal';
2 mkdir 'sync';
3 mkdir 'spectre';
4 axis("auto");
```

Рис. 22: Задаем входные кодовые последовательности

```
% Униполярное кодирование
wave=unipolar(data);
plot(wave);
ylim([-1 6]);
title('Unipolar');
print 'signal/unipolar.png';
% Кодирование ami
wave=ami(data);
plot(wave);
title('AMI');
print 'signal/ami.png';
% Кодирование NRZ
wave=bipolarnrz(data);
plot(wave);
title('Bipolar Non-Return to Zero');
print 'signal/bipolarnrz.png';
% Кодирование RZ
wave=bipolarrz(data);
plot(wave);
title('Bipolar Return to Zero');
print 'signal/bipolarrz.png';
% Манчестерское кодирование
wave=manchester(data);
plot(wave);
title('Manchester');
print 'signal/manchester.png';
% Дифференциальное манчестерское кодирование
wave=diffmanc(data);
plot(wave);
title('Differential Manchester');
print 'signal/diffmanc.png';
```

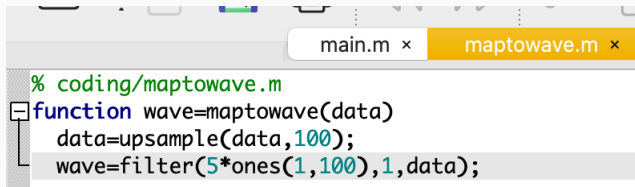
Рис. 23: Вызовы функций для построения модуляций кодированных сигналов кодовой последовательности data

```
% Униполярное кодирование
wave=unipolar(data_sync);
plot(wave);
ylim([-1 6]);
title('Unipolar');
print 'sync/unipolar.png';
% Кодирование AMI
wave=ami(data_sync);
plot(wave);
title('AMI');
print 'sync/ami.png';
% Кодирование NRZ
wave=bipolarnrz(data_sync);
plot(wave);
title('Bipolar Non-Return to Zero');
print 'sync/bipolarnrz.png';
% Кодирование RZ
wave=bipolarrz(data_sync);
plot(wave);
title('Bipolar Return to Zero');
print 'sync/bipolarrz.png';
% Манчестерское кодирование
wave=manchester(data_sync);
plot(wave);
title('Manchester');
print 'sync/manchester.png';
% Дифференциальное манчестерское кодирование
wave=diffmanc(data_sync);
plot(wave);
title('Differential Manchester');
print 'sync/diffmanc.png';
```

Рис. 24: Вызовы функций для построения модуляций кодированных сигналов кодовой последовательности data_sync

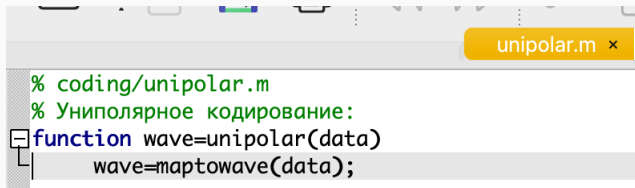

```
% Униполярное кодирование:
wave=unipolar(data_spectre);
spectre=calcspectre(wave);
title('Unipolar');
print 'spectre/unipolar.png';
% Кодирование AMI:
wave=ami(data_spectre);
spectre=calcspectre(wave);
title('AMI');
print 'spectre/ami.png';
% Кодирование NRZ:
wave=bipolarnrz(data_spectre);
spectre=calcspectre(wave);
title('Bipolar Non-Return to Zero');
print 'spectre/bipolarnrz.png';
% Кодирование RZ:
wave=bipolarrz(data_spectre);
spectre=calcspectre(wave);
title('Bipolar Return to Zero');
print 'spectre/bipolarrz.png';
% Манчестерское кодирование:
wave=manchester(data_spectre);
spectre=calcspectre(wave);
title('Manchester');
print 'spectre/manchester.png';
% Дифференциальное манчестерское кодирование:
wave=diffmanc(data_spectre);
spectre=calcspectre(wave);
title('Differential Manchester');
print 'spectre/diffmanc.png';
```

Рис. 25: Вызовы функций для построения графиков спектров



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

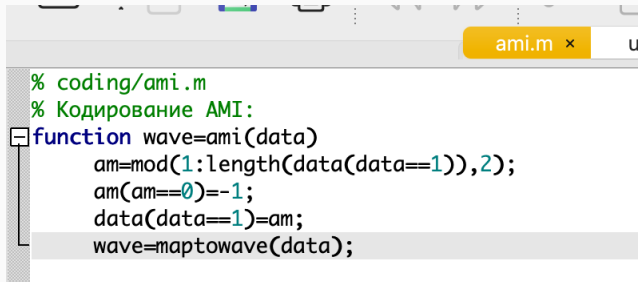
Рис. 26: Листинг файла maptowave.m



The image shows a screenshot of a MATLAB editor window. The title bar at the top right indicates the file is 'unipolar.m'. The code is as follows:

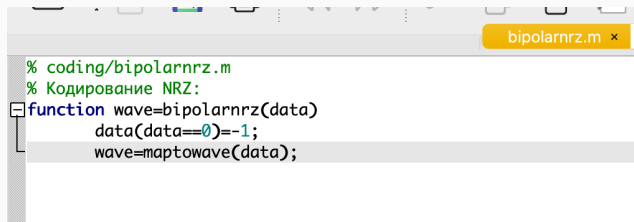
```
% coding/unipolar.m  
% Униполярное кодирование:  
function wave=unipolar(data)  
    wave=maptowave(data);
```

Рис. 27: Листинг файла unipolar.m



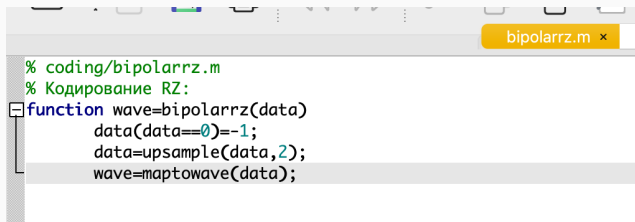
```
% coding/ami.m
% Кодирование AMI:
function wave=ami(data)
    am=mod(1:length(data(data==1)),2);
    am(am==0)=-1;
    data(data==1)=am;
    wave=maptowave(data);
```

Рис. 28: Листинг файла ami.m



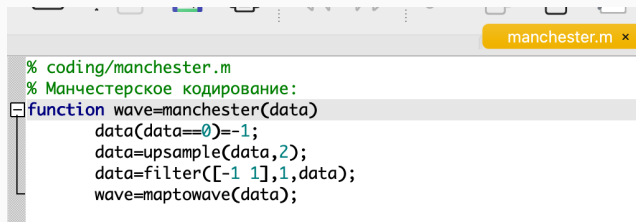
```
% coding/bipolarnrz.m
% Кодирование NRZ:
function wave=bipolarnrz(data)
    data(data==0)=-1;
    wave=maptowave(data);
```

Рис. 29: Листинг файла bipolarnrz.m



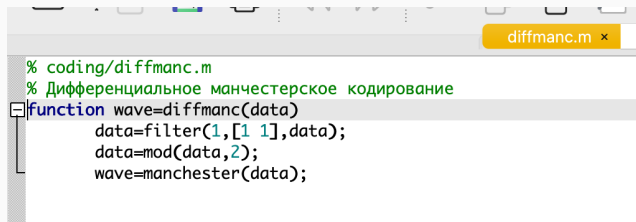
```
% coding/bipolarrrz.m
% Кодирование RZ:
function wave=bipolarrrz(data)
    data(data==0)=-1;
    data=upsample(data,2);
    wave=maptowave(data);
```

Рис. 30: Листинг файла bipolarrrz.m



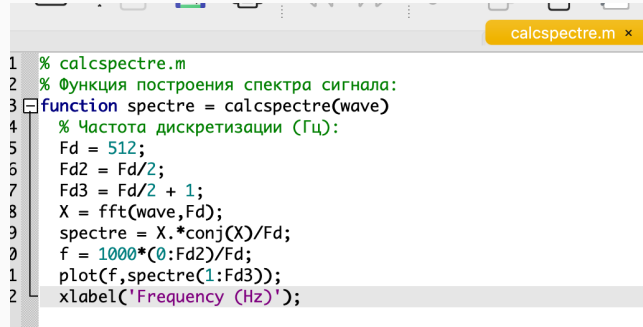
```
% coding/manchester.m
% Манчестерское кодирование:
function wave=manchester(data)
    data(data==0)=-1;
    data=upsample(data,2);
    data=filter([-1 1],1,data);
    wave=maptowave(data);
```

Рис. 31: Листинг файла manchester.m



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

Рис. 32: Листинг файла 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. 33: Листинг файла calcspectre.m

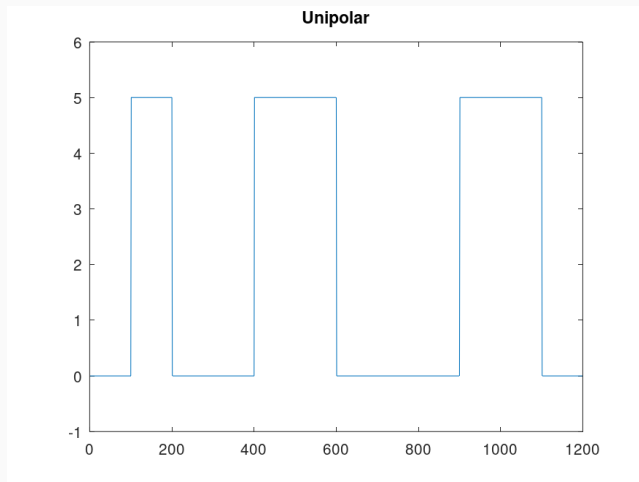


Рис. 34: Униполярное кодирование

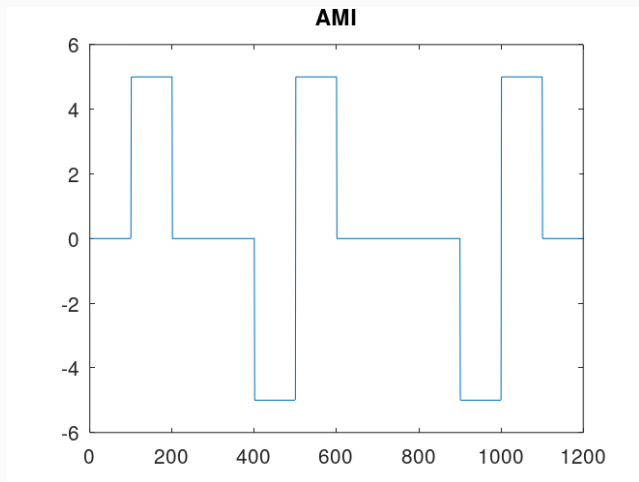


Рис. 35: Кодирование AMI

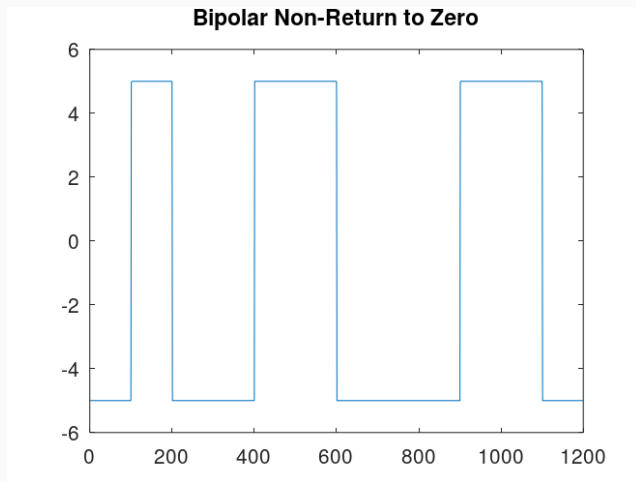


Рис. 36: Кодирование NRZ

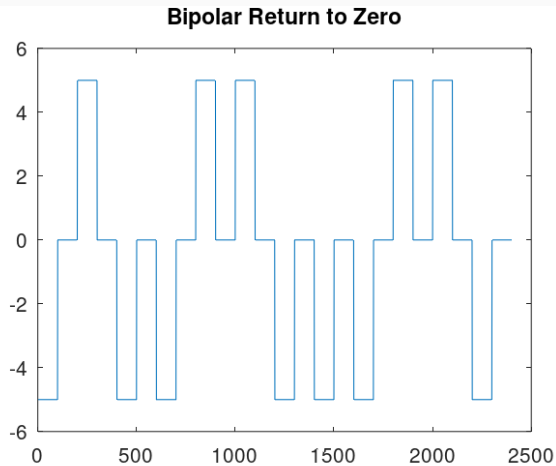


Рис. 37: Кодирование RZ

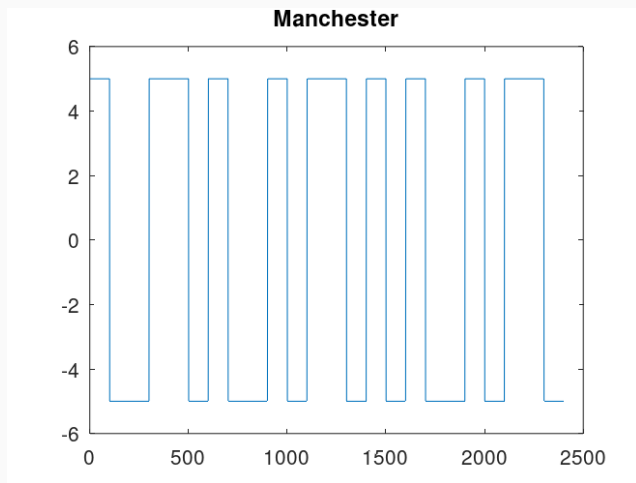


Рис. 38: Манчестерское кодирование

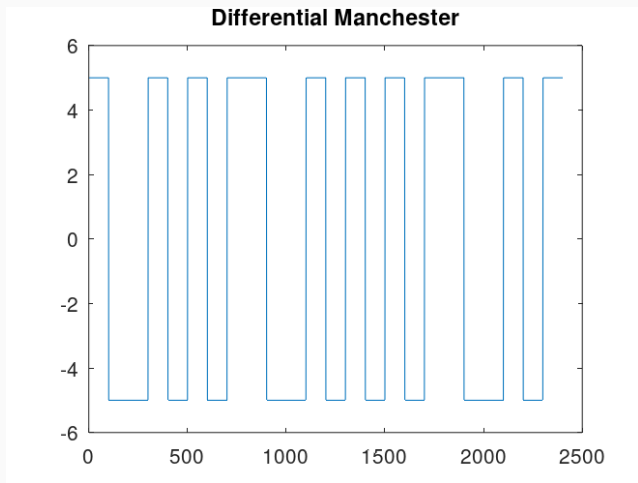


Рис. 39: Дифференциальное манчестерское кодирование

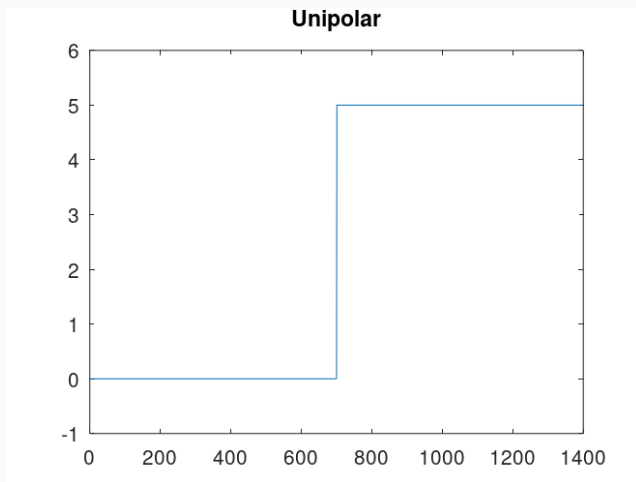


Рис. 40: Униполярное кодирование: нет самосинхронизации

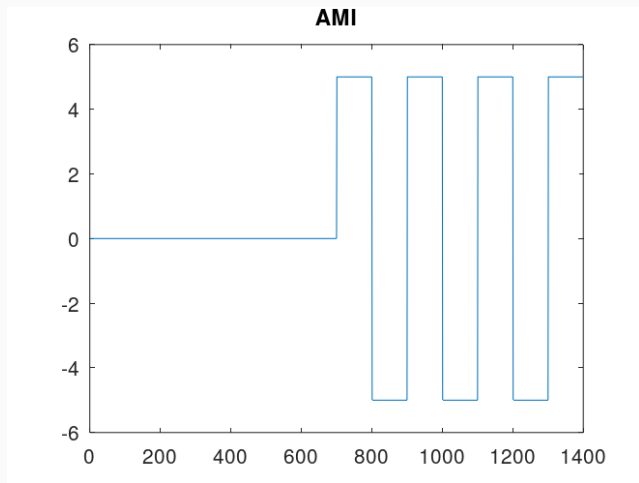


Рис. 41: Кодирование AMI: самосинхронизация при наличии сигнала

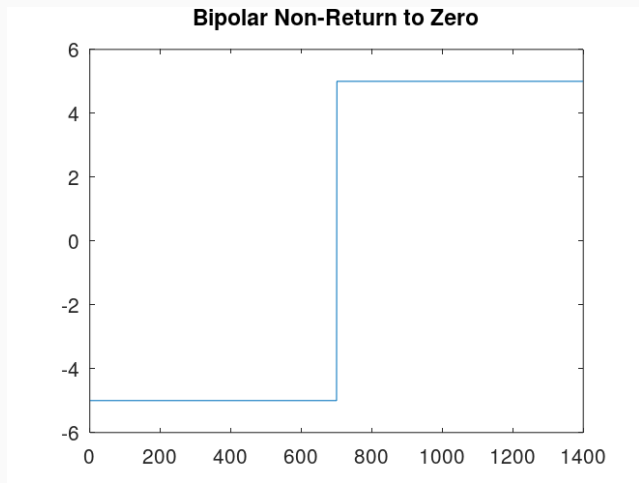


Рис. 42: Кодирование NRZ: нет самосинхронизации

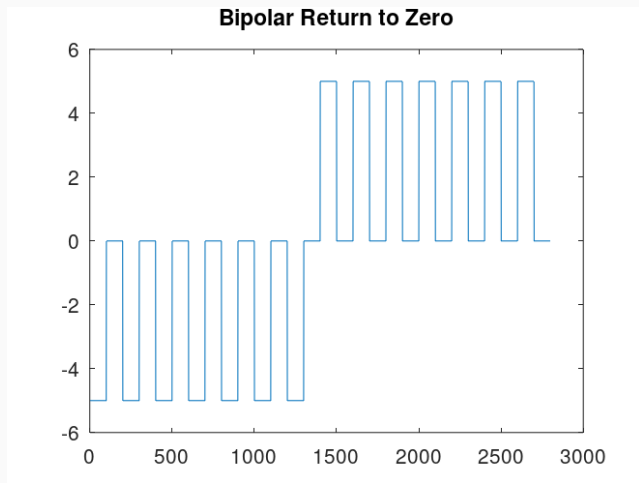


Рис. 43: Кодирование RZ: есть самосинхронизация

Иллюстрация свойства самосинхронизации

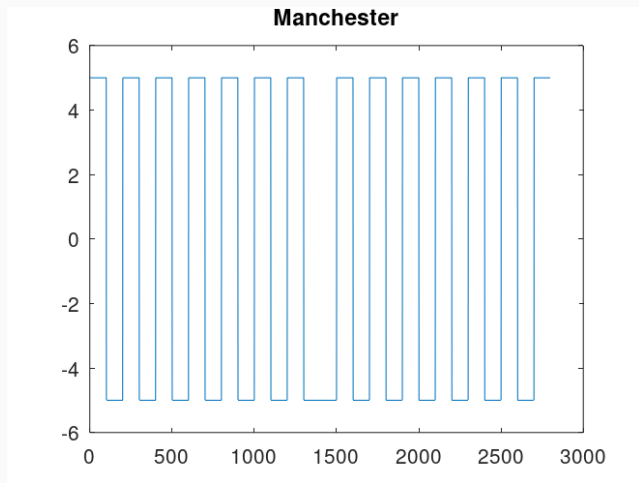


Рис. 44: Манчестерское кодирование: есть самосинхронизация

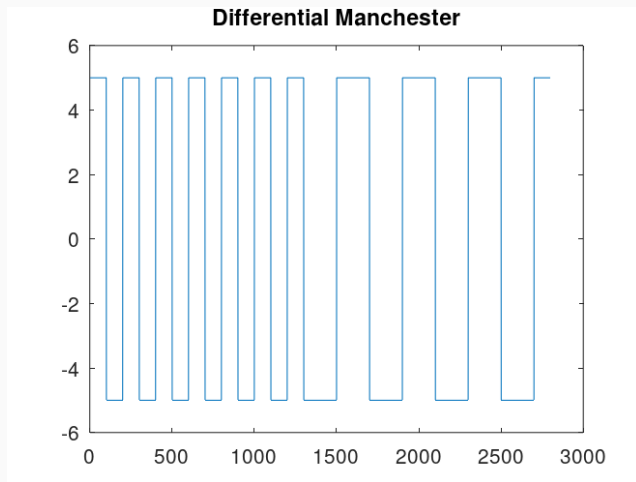


Рис. 45: Дифференциальное манчестерское кодирование: есть самосинхронизация

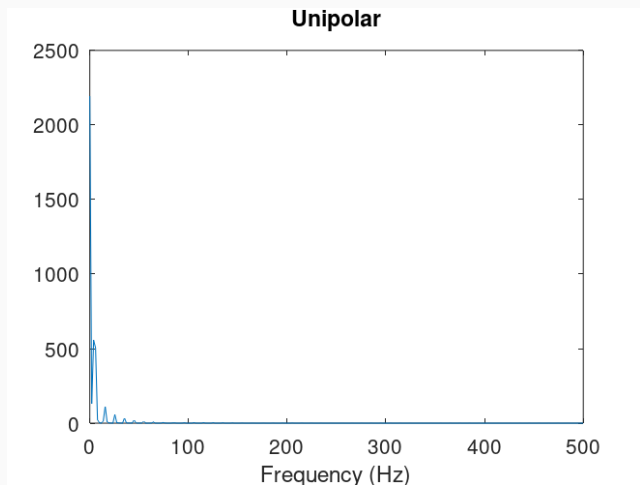


Рис. 46: Униполярное кодирование: спектр сигнала

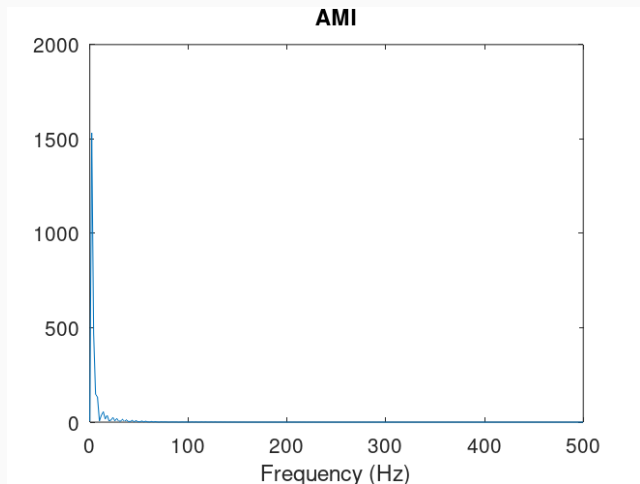


Рис. 47: Кодирование AMI: спектр сигнала

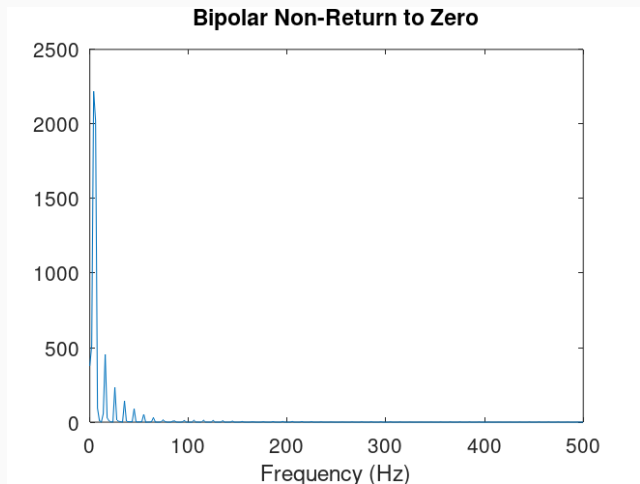


Рис. 48: Кодирование NRZ: спектр сигнала

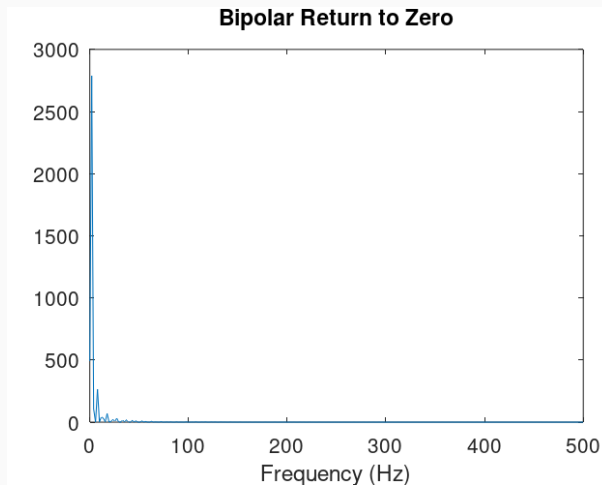


Рис. 49: Кодирование RZ: спектр сигнала

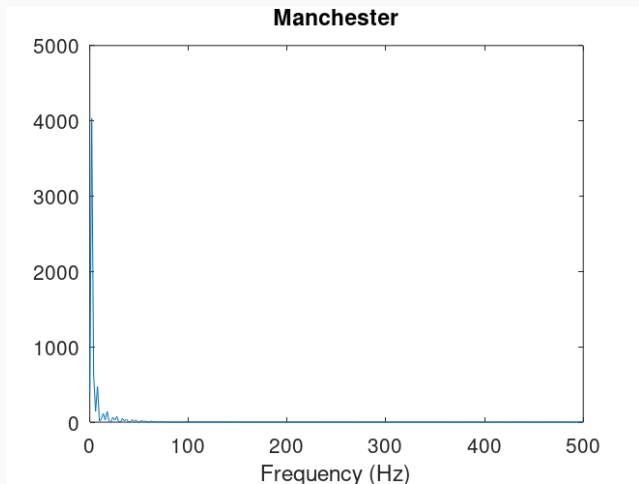


Рис. 50: Манчестерское кодирование: спектр сигнала

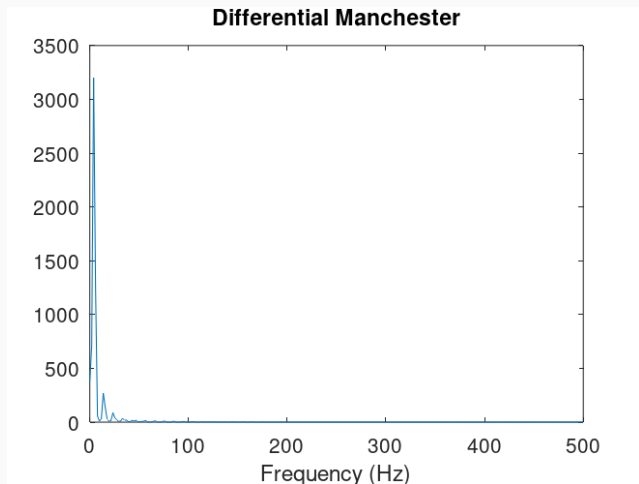


Рис. 51: Дифференциальное манчестерское кодирование: спектр сигнала

- В ходе выполнения данной лабораторной работы я изучила методы кодирования и модуляции сигналов с помощью высокоуровневого языка программирования Octave. Определила спектр и параметры сигнала. Продемонстрировала принципы модуляции сигнала на примере аналоговой амплитудной модуляции. Исследовала свойства самосинхронизации сигнала.