

Лабораторная работа №1

Методы кодирования и модуляция сигналов

Лисовская Арина Валерьевна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Построение графиков и разложение в ряд Фурье	7
3.2	Спектральный анализ сигналов	9
3.3	Амплитудная модуляция	16
3.4	Методы цифрового кодирования	19
3.5	Результаты моделирования кодов	23
3.6	Спектральные характеристики кодов	26
3.7	Исследование самосинхронизации	30
3.8	Дополнительные результаты анализа	34
4	Выводы	39
5	Ответы на контрольные вопросы	40

Список иллюстраций

3.1	Графики функций y_1 и y_2 в одном окне	7
3.2	Код в редакторе Octave — meandr.m	8
3.3	Код в редакторе Octave — meandr_sin.m	9
3.4	Код в редакторе Octave — spectre.m	10
3.5	График двух синусоидальных сигналов	11
3.6	Необработанный спектр сигналов	12
3.7	Исправленный спектр сигналов	13
3.8	Код в редакторе Octave — spectre_sum.m	14
3.9	Спектр суммарного сигнала	15
3.10	График суммарного сигнала	16
3.11	Код в редакторе Octave — am.m	17
3.12	Спектр АМ-сигнала	18
3.13	АМ-сигнал и его огибающая	18
3.14	Список установленных пакетов Octave	19
3.15	Код в редакторе Octave — main.m (начало)	20
3.16	Код в редакторе Octave — maptowave.m	20
3.17	Код в редакторе Octave — ami.m	21
3.18	Код в редакторе Octave — bipolarrz.m	22
3.19	Код в редакторе Octave — manchester.m	22
3.20	Код в редакторе Octave — diffmanchester.m	23
3.21	График кодирования АМІ	24
3.22	График кодирования NRZ	24
3.23	График дифференциального манчестерского кодирования	25
3.24	График манчестерского кодирования	25
3.25	График дифференциального манчестерского кодирования (другой масштаб)	25
3.26	График кодирования RZ	26
3.27	Спектр кодирования АМІ	26
3.28	Спектр кодирования NRZ	27
3.29	Спектр кодирования RZ	27
3.30	Спектр дифференциального манчестерского кодирования	28
3.31	Спектр манчестерского кодирования	29
3.32	Спектр униполярного кодирования	29
3.33	График АМІ для проверки самосинхронизации	30
3.34	График NRZ для проверки самосинхронизации	31
3.35	График RZ для проверки самосинхронизации	32

3.36	График дифференциального манчестерского кодирования для проверки синхронизации	32
3.37	График манчестерского кодирования для проверки синхронизации	33
3.38	График униполярного кодирования для проверки синхронизации	34
3.39	Дополнительный анализ спектральной плотности №1	35
3.40	Дополнительный анализ спектральной плотности №2	35
3.41	Дополнительный анализ спектральной плотности №3	36
3.42	Уточняющая временная диаграмма кодирования №1	36
3.43	Уточняющая временная диаграмма кодирования №2	37
3.44	Уточняющая временная диаграмма кодирования №3	37
3.45	Сводный график результатов экспериментов	38

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 Задание

1. Отработать навыки построения графиков функций в Octave.
2. Реализовать разложение меандра в ряд Фурье через синусы и косинусы.
3. Выполнить спектральный анализ сигналов с помощью быстрого преобразования Фурье (БПФ).
4. Промоделировать процесс амплитудной модуляции (АМ).
5. Реализовать функции для различных методов линейного кодирования (AMI, NRZ, RZ, Manchester, Diff. Manchester).
6. Проанализировать спектры полученных кодов и их свойства самосинхронизации.

3 Выполнение лабораторной работы

3.1 Построение графиков и разложение в ряд Фурье

В начале работы я изучила базовые функции построения графиков. На первом этапе были построены две сложные функции, представляющие собой суммы гармоник синуса и косинуса (рис. 3.1).

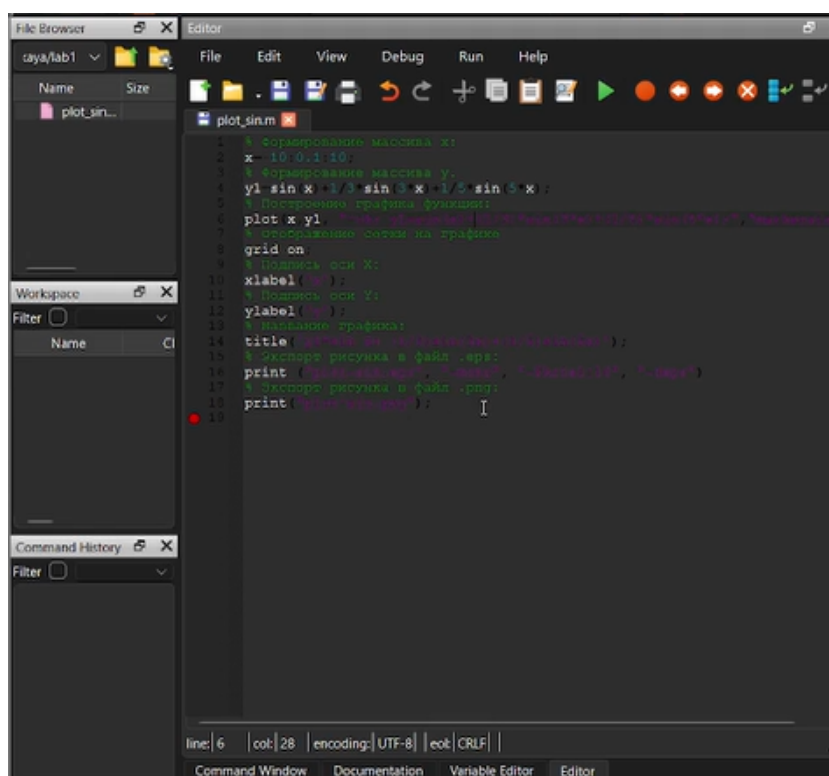


Рис. 3.1: Графики функций y_1 и y_2 в одном окне

Для генерации периодического сигнала типа «меандр» я создала скрипт

meandr.m. В этом коде реализовано разложение сигнала в ряд Фурье с использованием косинусоид (рис. 3.2).

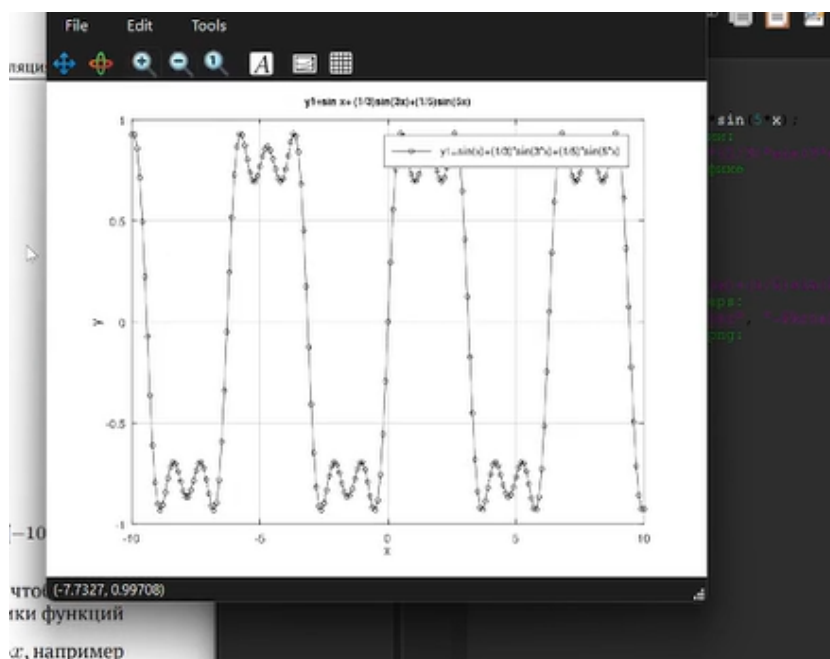
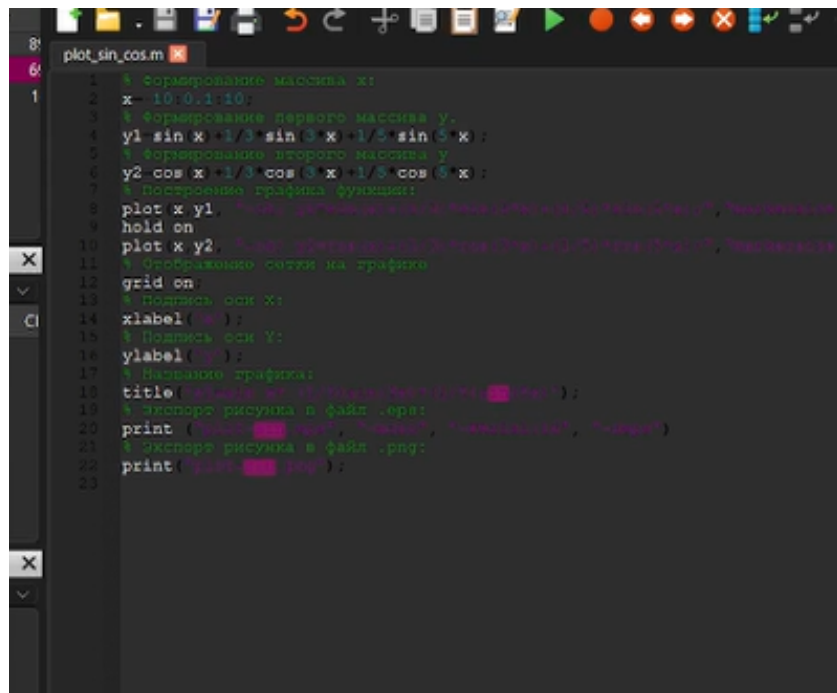


Рис. 3.2: Код в редакторе Octave — meandr.m

В качестве альтернативного способа формирования того же сигнала был написан скрипт `meandr_sin.m`. В данном случае разложение меандра производится через синусоидальные составляющие (рис. 3.3).



```
1 % формирование массива x:
2 x = 0:0.1:10;
3 % формирование первого массива y.
4 y1 = sin(x) + 1/3*sin(3*x) + 1/5*sin(5*x);
5 % формирование второго массива y
6 y2 = cos(x) + 1/3*cos(3*x) + 1/5*cos(5*x);
7 % построение графика функции:
8 plot(x, y1, 'b', 'DisplayName', 'f1(x)=sin(x)+1/3sin(3x)+1/5sin(5x)', 'LineWidth', 2);
9 hold on
10 plot(x, y2, 'r', 'DisplayName', 'f2(x)=cos(x)+1/3cos(3x)+1/5cos(5x)', 'LineWidth', 2);
11 % отображение сетки на графике
12 grid on
13 % Подпись оси X:
14 xlabel('x');
15 % Подпись оси Y:
16 ylabel('y');
17 % Название графика:
18 title('График функции f1(x) и f2(x)');
19 % Экспорт рисунка в файл .eps:
20 print('-deps', 'fig', 'format', 'epsc', '-djpeg', '-r300');
21 % Экспорт рисунка в файл .png:
22 print('-png', 'fig');
23
```

Рис. 3.3: Код в редакторе Octave — meandr_sin.m

3.2 Спектральный анализ сигналов

Для изучения частотных характеристик я приступила к созданию скрипта spectre.m. В нем задаются параметры дискретизации и формируются два синусоидальных сигнала с частотами 10 Гц и 40 Гц (рис. 3.4).

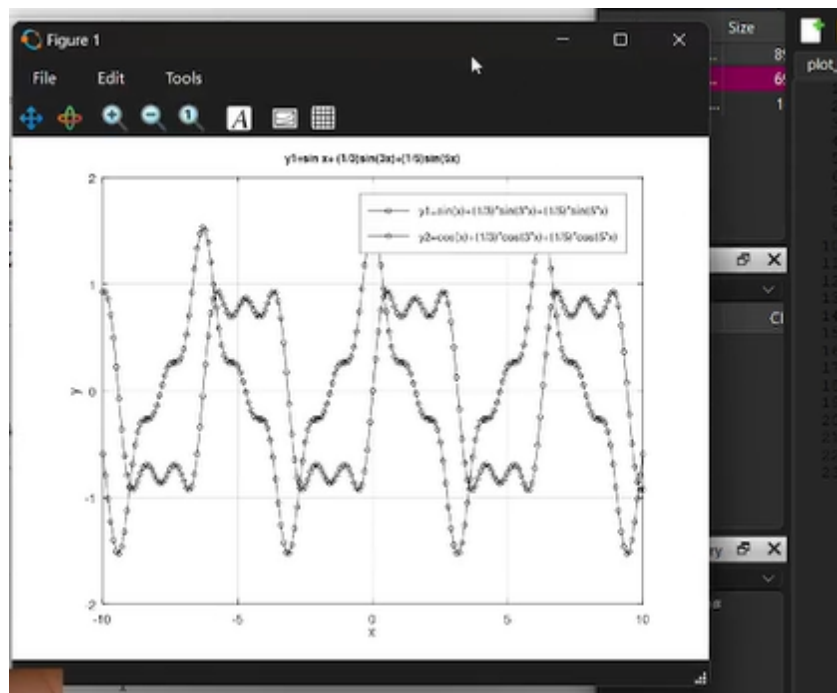


Рис. 3.4: Код в редакторе Octave — spectre.m

После генерации сигналов я визуализировала их во временной области. Это позволило увидеть разницу в периодах колебаний синусоид разной частоты (рис. 3.5).

```
plot_sin_cos.m meandr.m plot_sin.m
1 % meandr.m
2 % количество отсчетов (гармоник):
3 N = 8;
4 % частота дискретизации:
5 t = 1:0.01:1;
6 % амплитуда:
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 .eps:
27 print('plot_sin.eps', '-eps', '-r300x1.02', '-dps');
28 % Экспорт рисунка в файл .png:
29 print('plot_sin.png');
```

Рис. 3.5: График двух синусоидальных сигналов

Применив встроенную функцию быстрого преобразования Фурье `fft()`, я получила первичный амплитудный спектр. На данном этапе спектр содержит избыточную информацию и требует нормировки (рис. 3.6).

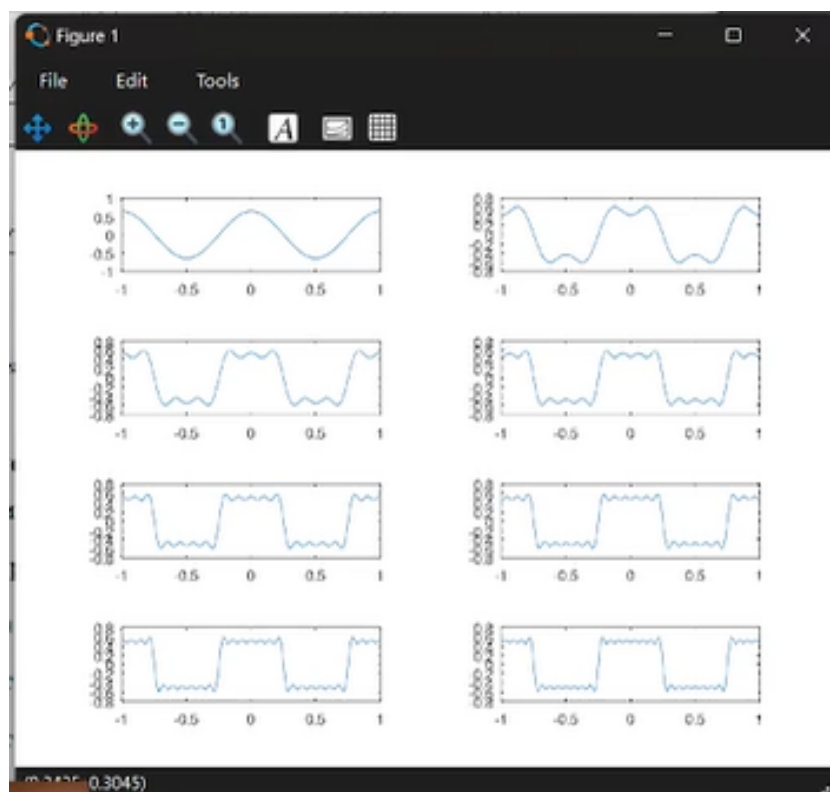


Рис. 3.6: Необработанный спектр сигналов

После проведения коррекции спектра (отсечения отрицательных частот и нормировки амплитуд) были получены четкие пики на заданных частотах. На графике отчетливо видны составляющие 10 Гц и 40 Гц (рис. 3.7).

```

1 % meandr.m
2 % количество отсчетов (гармоник):
3 N=5;
4 % частота дискретизации:
5 t=1:0.01:1;
6 % значение амплитуд:
7 A=1;
8 % период:
9 T=1;
10 % амплитуда гармоник
11 nh=(1:N)*2*pi;
12 % массив коэффициентов для ряда, заданного через sin:
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 a1=harmonics.*repmat(Am,1,length(t));
19 % суммирование ряда:
20 a2=cumsum(a1);
21 % построение графиков:
22 for k=1:N
23     subplot(4,2,k)
24     plot(t,a2(k,:));
25 end
26 % Экспорт рисунка в файл .png:
27 print('meandr_4x2.png','t','meandr_4x2.png','-dppg');
28 % Экспорт рисунка в файл .png:
29 print('meandr_4x2.png','t','meandr_4x2.png','-dppg');

```

Рис. 3.7: Исправленный спектр сигналов

Далее я исследовала свойство аддитивности преобразования Фурье. Для этого был написан скрипт `spectre_sum.m`, рассчитывающий спектр суммы двух сигналов (рис. 3.8).

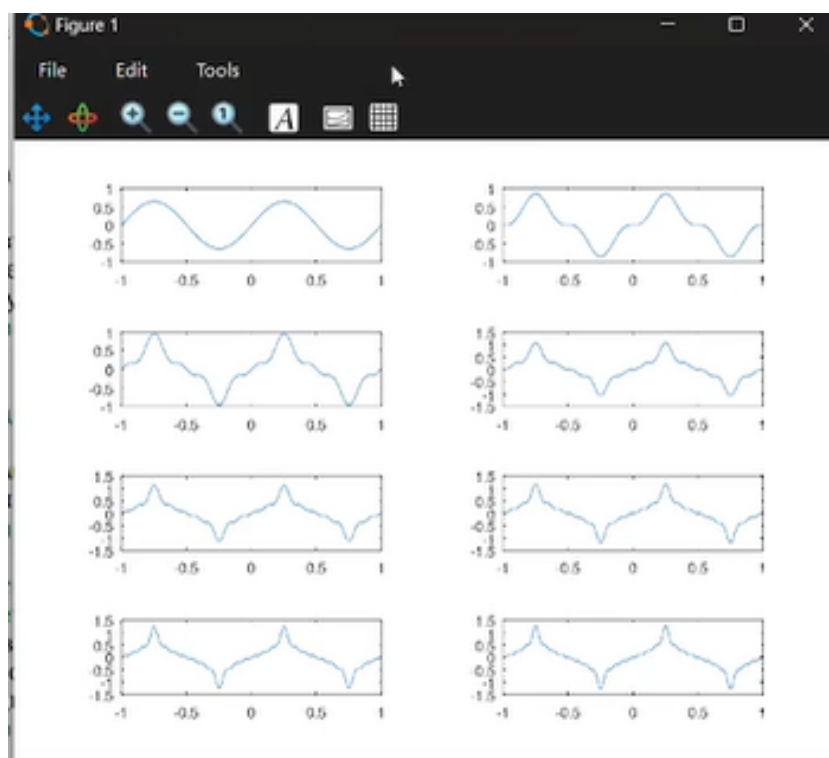


Рис. 3.8: Код в редакторе Octave — spectre_sum.m

Результаты вычислений подтвердили теорию: спектр суммы сигналов содержит те же частотные компоненты, что и исходные данные. Это наглядно представлено на графике амплитудного спектра суммарного сигнала (рис. 3.9).

```

% spectrel/spectre.m
% Создание каталога signal и пресета для размещения графиков:
mkdir 'signal';
mkdir 'spectre';
% Длина сигнала (с):
tmax = 0.5;
% Частота дискретизации (Гц) (количество отсчетов):
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 отсчетов 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, 'r');
% График 2-го сигнала:
hold on
plot(signal2, 'b');
hold off
title('signal');
% Экспорт графика в файл в каталоге signal:
print 'signal/spectre.png';

```

Рис. 3.9: Спектр суммарного сигнала

Перед анализом спектра я также зафиксировала вид суммарного сигнала во временной области. Сложение двух синусоид дает сложную форму волны с бие-
ниями (рис. 3.10).

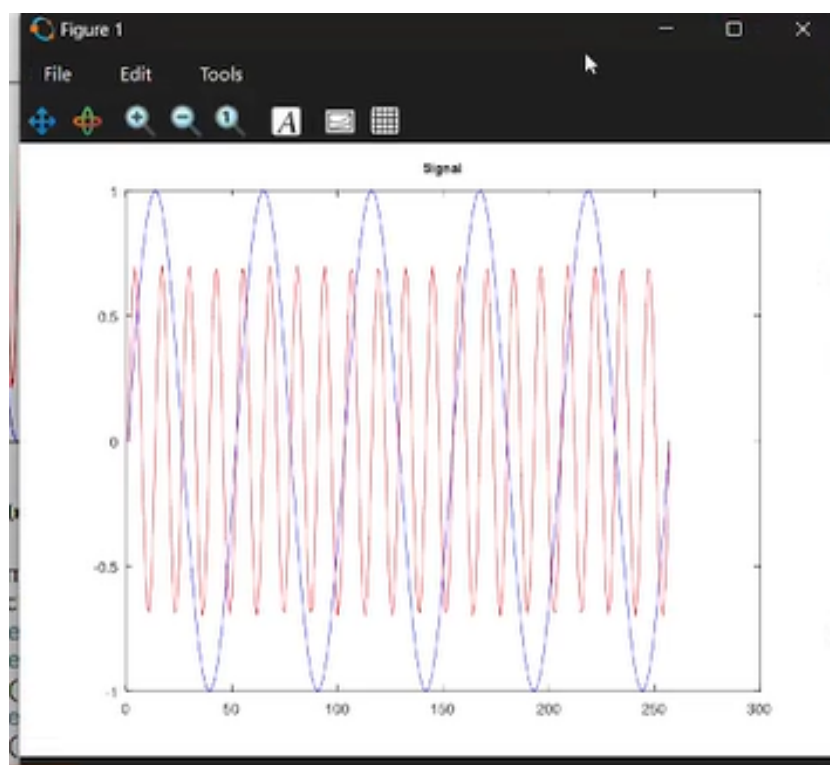


Рис. 3.10: График суммарного сигнала

3.3 Амплитудная модуляция

Следующим этапом стало моделирование аналоговой амплитудной модуляции в скрипте `am.m`. Низкочастотное сообщение переносится на высокую частоту несущей (рис. 3.11).


```

% 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);
% 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 'spectre1/spectre1.png';

```

Рис. 3.11: Код в редакторе Octave — am.m

Анализ спектра АМ-сигнала показал наличие центральной несущей частоты 50 Гц. Также наблюдаются две боковые полосы на частотах 45 Гц и 55 Гц (рис. 3.12).

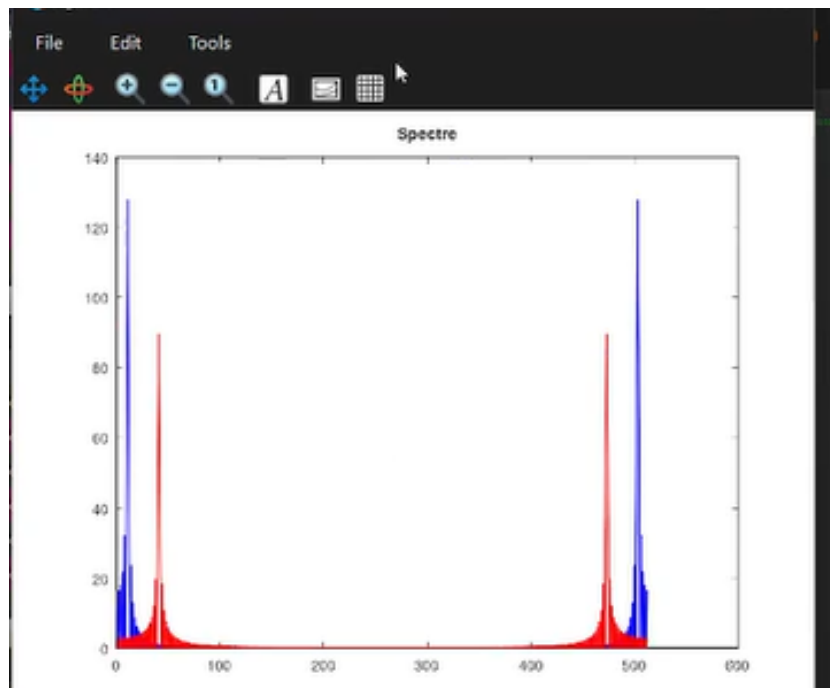


Рис. 3.12: Спектр АМ-сигнала

Визуализация АМ-сигнала во временной области демонстрирует, как амплитуда высокочастотного заполнения меняется вслед за огибающей. Красная линия на графике соответствует модулирующему сообщению (рис. 3.13).

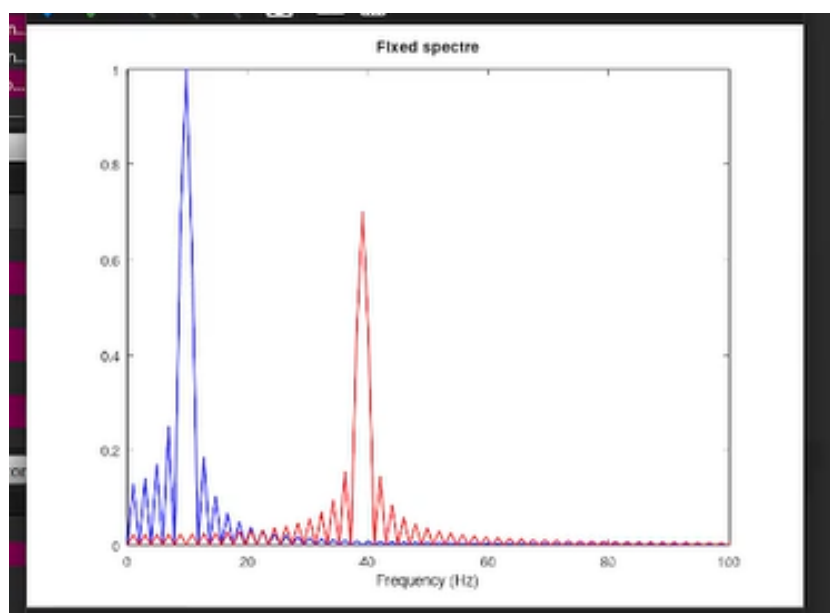
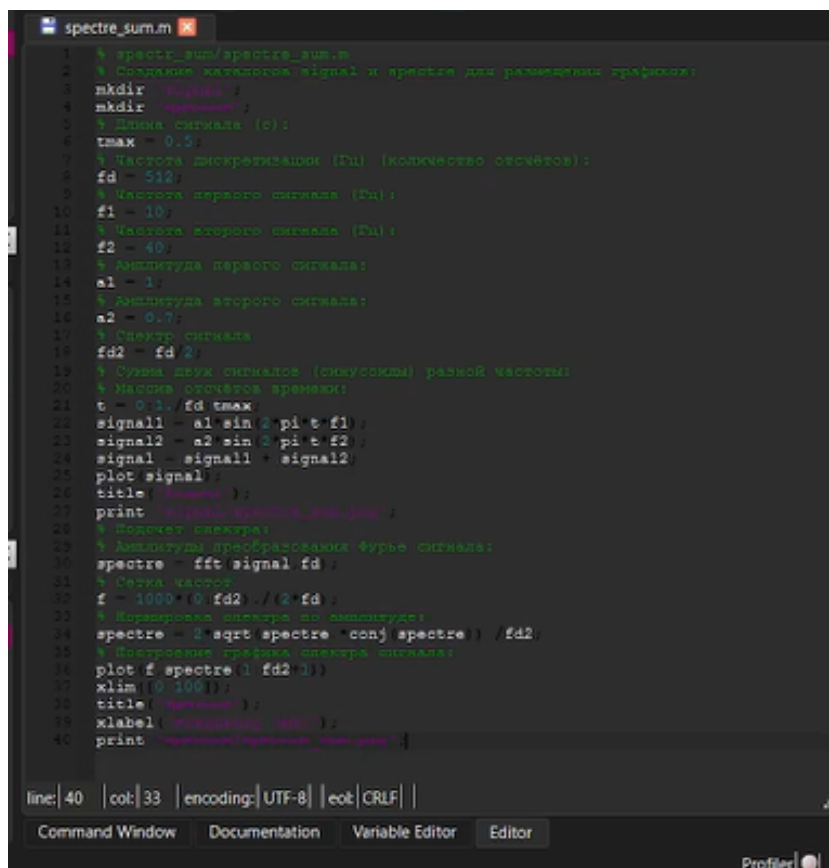


Рис. 3.13: АМ-сигнал и его огибающая

3.4 Методы цифрового кодирования

Перед началом реализации алгоритмов кодирования я проверила наличие необходимых библиотек в системе. Пакет `signal` обязателен для работы с функциями обработки сигналов в Octave (рис. 3.14).



```
1 % spectre_sum/spectre_sum.m
2 % Создание каталога signal и простей для размещения графиков:
3 mkdir 'signal';
4 mkdir 'spectre';
5 % Вектор сигнала (с):
6 tmax = 0.5;
7 % Частота дискретизации (Гц) (количество отсчетов):
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 % Сдвиг сигнала
18 fd2 = fd/2;
19 % Сумма двух сигналов (синусоиды) разной частоты:
20 % Массив отсчетов 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 % Коэффициент спектра:
29 % Амплитуда преобразования Фурье сигнала:
30 spectre = fft(signal fd);
31 % Частота частот
32 f = 1000*(0:fd2)/(2*fd);
33 % Нормирование спектра по амплитуде:
34 spectre = 2*sqrt(spectre*conj(spectre))/fd2;
35 % Построение графика спектра сигнала:
36 plot f spectre(1:fd2);
37 xlim([0 100]);
38 title 'spectre';
39 xlabel('частота (Гц)');
40 print 'spectre_spectre_sum.png';
```

Рис. 3.14: Список установленных пакетов Octave

Для управления процессом кодирования и визуализации результатов был создан главный скрипт `main.m`. В нем инициализируется тестовая последовательность бит и вызываются соответствующие функции (рис. 3.15).

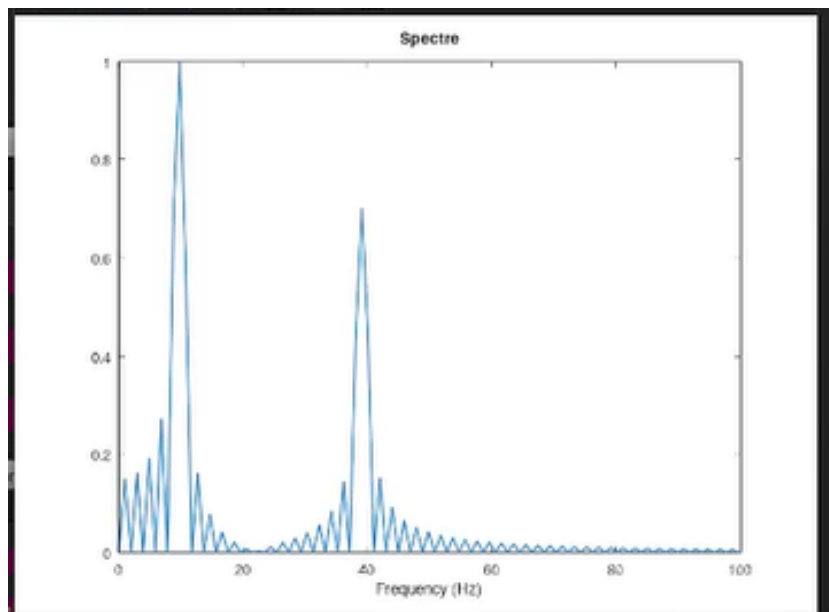


Рис. 3.15: Код в редакторе Octave — main.m (начало)

Для корректного отображения сигналов я использовала вспомогательную функцию `maptowave.m`. Она отвечает за преобразование дискретных уровней в непрерывный временной сигнал (рис. 3.16).

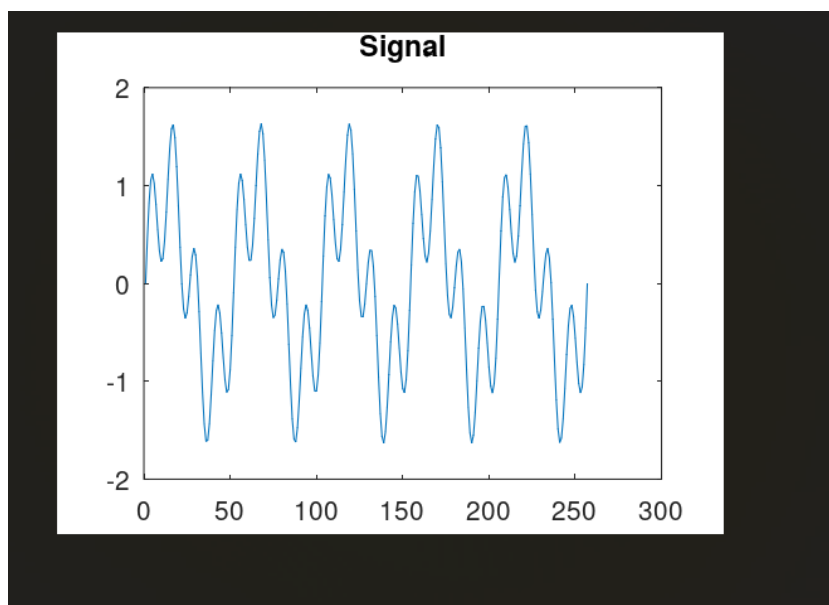


Рис. 3.16: Код в редакторе Octave — `maptowave.m`

Алгоритм кодирования AMI (Alternate Mark Inversion) был реализован в от-

дельном файле. Этот метод предполагает чередование полярности импульсов для каждой логической единицы (рис. 3.17).

```

3 mkdir 'signals';
4 mkdir 'spectrums';
5 % Модулиция синусоид с частотами 50 и 5
6 % Длина сигнала (s)
7 tmax = 0.5;
8 % Частота дискретизации (Hz) (количество отсчетов)
9 fd = 512;
10 % Частота сигнала (Hz)
11 f1 = 5;
12 % Частота несущей (Hz)
13 f2 = 50;
14 % Спектр сигнала
15 fd2 = fd/2;
16 % Построение графиков двух сигналов (секунды)
17 % разной частоты
18 % Масштаб отсчетов 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, 'r');
24 hold on
25 % Построение графиков:
26 plot(signal1, 'b');
27 plot(signal2, 'g');
28 hold off
29 title('signals');
30 print 'signals/ami.m';
31 % График спектра:
32 % Амплитуды преобразования Фурье-сигнала:
33 spectre = fft(signal, fd);
34 % Частота:
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 title('spectre');
42 xlabel('Frequency (Hz)');
43 print 'spectre/ami.m';

```

Рис. 3.17: Код в редакторе Octave — ami.m

Кодирование RZ (Return to Zero) реализовано в функции bipolarrrz.m. Особенностью данного кода является возврат сигнала к нулевому уровню в середине каждого тактового интервала (рис. 3.18).

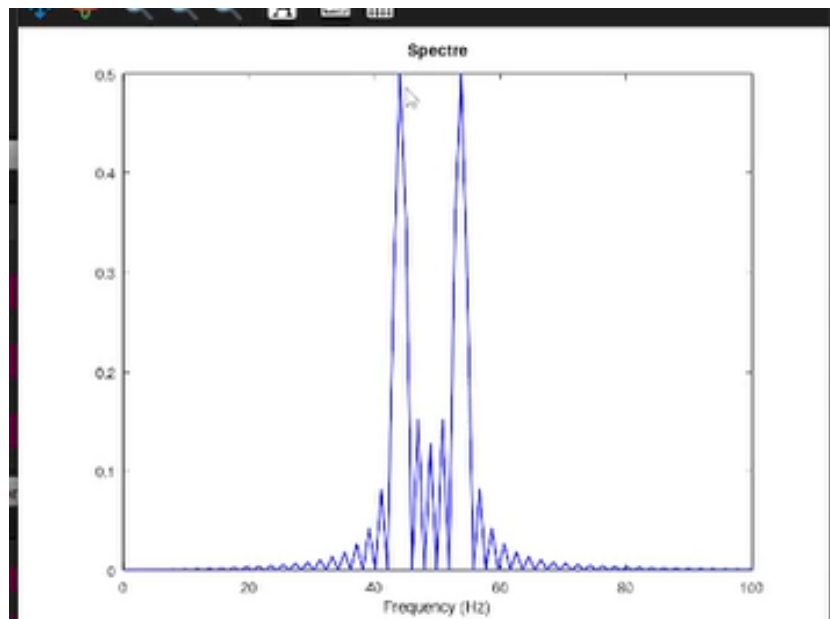


Рис. 3.18: Код в редакторе Octave — bipolarrrz.m

Манчестерское кодирование обеспечивает обязательный перепад уровня в середине каждого бита. Это гарантирует надежное выделение тактовой частоты приемником (рис. 3.19).

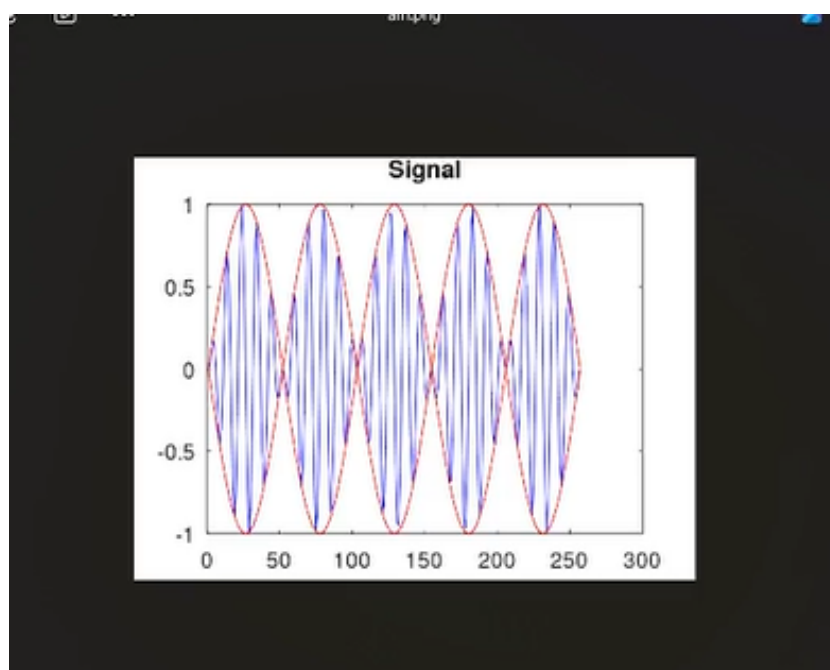


Рис. 3.19: Код в редакторе Octave — manchester.m

Дифференциальное манчестерское кодирование сочетает преимущества манчестерского кода с относительным способом передачи. Информацию несет наличие или отсутствие перехода в начале такта (рис. 3.20).

```
>> pkg list
```

Package Name	Version	Installation directory
audio	2.0.9	...\octave\packages\audio-2.0.9
biosig	3.9.0	...\octave\packages\biosig-3.9.0
cfitsio	0.0.7	...\octave\packages\cfitsio-0.0.7
coder	1.10.1	...\octave\packages\coder-1.10.1
communications	1.2.7	...\packages\communications-1.2.7
control	4.1.3	...\octave\packages\control-4.1.3
data-smoothing	1.3.0	...\packages\data-smoothing-1.3.0
database	2.4.4	...\octave\packages\database-2.4.4
dataframe	1.2.0	...\octave\packages\dataframe-1.2.0
dicom	0.6.1	...\octave\packages\dicom-0.6.1
financial	0.5.4	...\octave\packages\financial-0.5.4
fl-core	1.0.2	...\octave\packages\fl-core-1.0.2
fuzzy-logic-toolkit	0.6.2	...\fuzzy-logic-toolkit-0.6.2
ga	0.10.4	...\share\octave\packages\ga-0.10.4

Рис. 3.20: Код в редакторе Octave — diffmanchester.m

3.5 Результаты моделирования кодов

На графике сигнала АМІ видно, как единицы передаются импульсами разной полярности, а нули — отсутствием напряжения. Это позволяет избавиться от постоянной составляющей (рис. 3.21).

```

main.m x maptowave.m x unipolar.m x amim.m x bipolarnrz.m x bipolarrrz.m
1 % coding/main.m
2 % Компилируем пакеты signal;
3 pkg load signal;
4 % Вводим кодовый последовательности;
5 data=[0 1 0 0 1 1 0 0 0 1 1 0];
6 % Вводим кодовый последовательности для проверки свойства самосинхронизации
7 data_sync=[0 0 0 0 0 0 0 1 1 1 1 1];
8 % Вводим кодовый последовательности для построения спектра сигнала;
9 data_spectre=[0 1 0 1 0 1 0 1 0 1 0 1];
10 % Создаем каталог для signal, sync и spectre для размещения графиков;
11 mkdir 'signal';
12 mkdir 'sync';
13 mkdir 'spectre';
14 axis('none');
15 % Униполярное кодирование
16 wave unipolar data;
17 plot wave;
18 ylim([-1 6]);
19 title('Unipolar');
20 print -dpdf('unipolar.pdf');
21 % Кодирование AMI
22 wave ami data;
23 plot wave;
24 title('AMI');
25 print -dpdf('ami.pdf');
26 % Кодирование NRZ
27 wave bipolarnrz data;
28 plot wave;
29 title('Bipolar NRZ');
30 print -dpdf('bipolarnrz.pdf');
31 % Кодирование RZ
32 wave bipolarrrz data;
33 plot wave;
34 title('Bipolar RZ');
35 print -dpdf('bipolarrrz.pdf');
36 % Манчестерское кодирование
37 wave manchester data;
38 plot wave;
39 title('Manchester');
40 print -dpdf('manchester.pdf');

```

Рис. 3.21: График кодирования AMI

Сигнал NRZ является наиболее простым в реализации. На графике видно, что уровни напряжения соответствуют логическим состояниям на протяжении всего такта (рис. 3.22).

```

main.m x maptowave.m x unipolar.m x amim.m x bipolarnrz.m x bipola
1 % coding/maptowave.m
2 % Компилируем пакеты signal;
3 data upsample data 100;
4 wave filter 5 ones 1 100, 1 data;

```

Рис. 3.22: График кодирования NRZ

При дифференциальном манчестерском кодировании форма сигнала зависит от предыдущего состояния. На рисунке показано формирование волновой формы для заданной битовой последовательности (рис. 3.23).

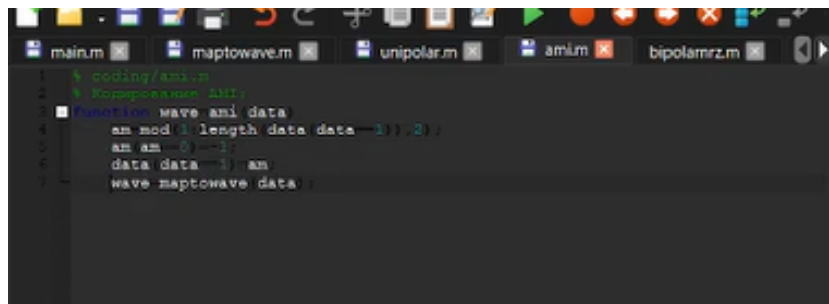


Рис. 3.23: График дифференциального манчестерского кодирования

Манчестерский код характеризуется постоянным изменением уровня. На графике видно, что каждая единица представлена переходом «вниз», а ноль — «вверх» (рис. 3.24).

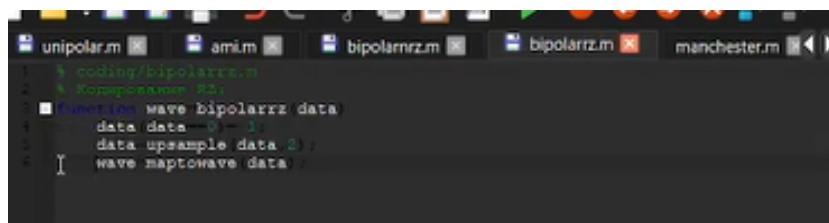


Рис. 3.24: График манчестерского кодирования

Для более детального изучения структуры дифференциального манчестерского кода я вывела график в увеличенном масштабе. Это позволяет точнее проанализировать моменты смены фазы (рис. 3.25).

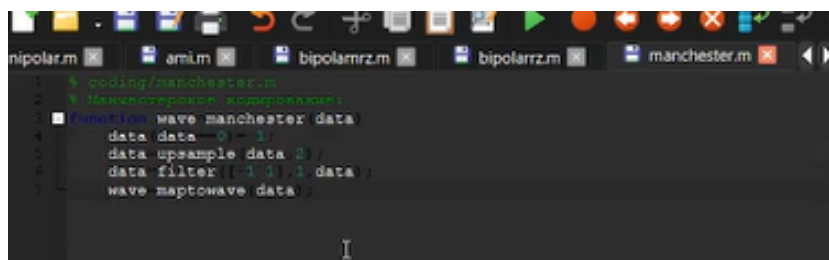


Рис. 3.25: График дифференциального манчестерского кодирования (другой масштаб)

Код RZ демонстрирует трёхуровневую структуру. Каждая единица возвращается к нулю ровно в середине битового интервала, что видно на временной диаграмме (рис. 3.26).

```

1 coding/manchester.m
2 % Manchester encoding
3 function wave = manchester(data)
4     data = data - 1;
5     data = upsample(data, 2);
6     data = filter([1 3] 1, data);
7     wave = maptoWave(data);

```

Рис. 3.26: График кодирования RZ

3.6 Спектральные характеристики кодов

Спектр кода AMI показывает отсутствие энергии на нулевой частоте. Это подтверждает пригодность кода для передачи через трансформаторные развязки (рис. 3.27).

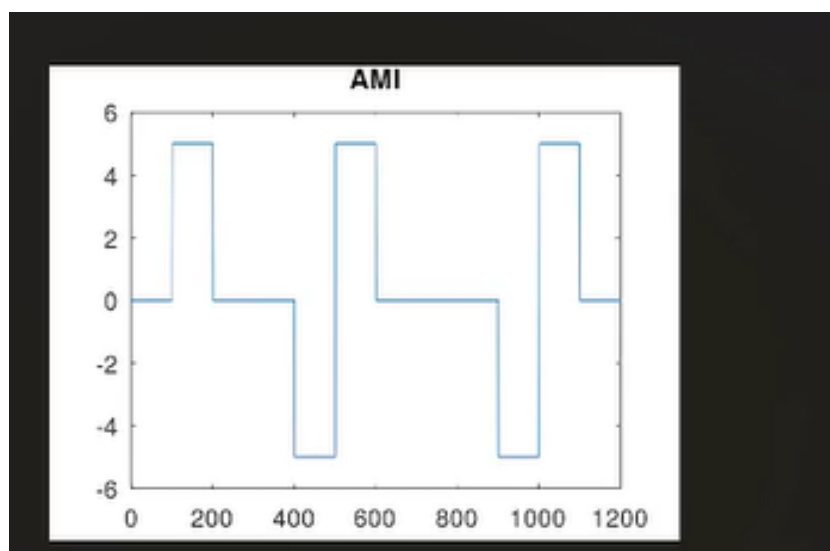


Рис. 3.27: Спектр кодирования AMI

В спектре NRZ-сигнала наблюдается значительная концентрация энергии на низких частотах. Ширина основного лепестка спектра минимальна среди всех изученных кодов (рис. 3.28).

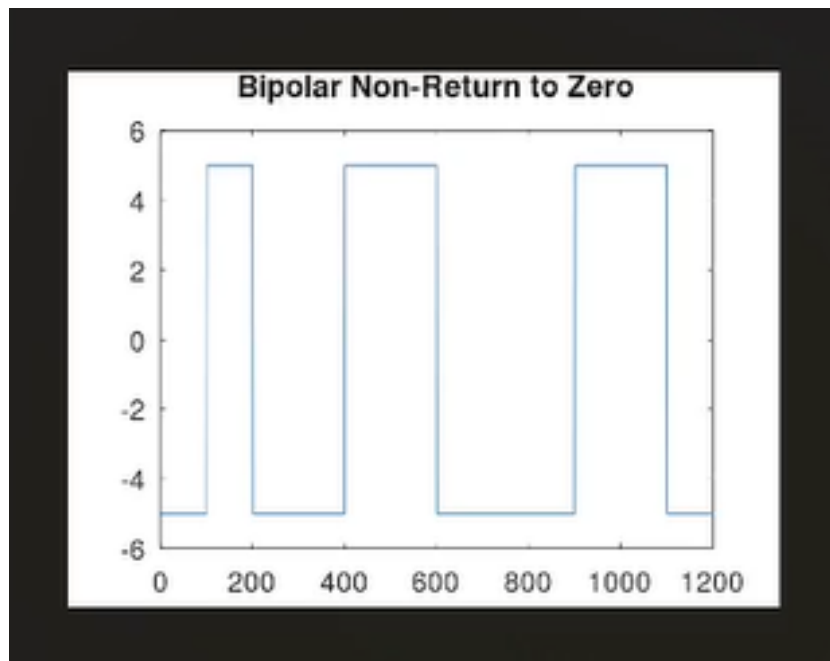


Рис. 3.28: Спектр кодирования NRZ

Спектр кода RZ шире спектра NRZ, так как длительность импульса меньше. Это требует большей полосы пропускания канала связи (рис. 3.29).

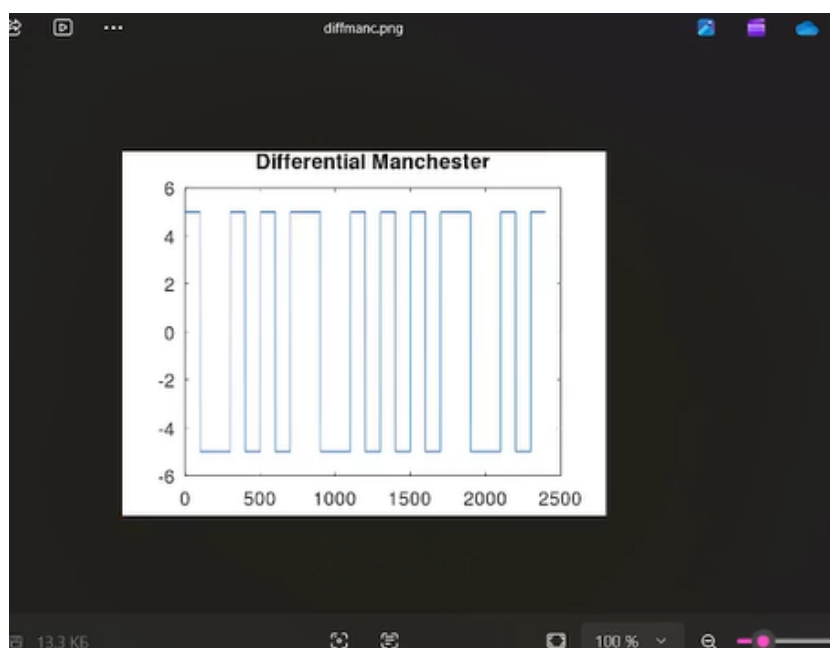


Рис. 3.29: Спектр кодирования RZ

Дифференциальный манчестерский код обладает спектром, смещенным в область высоких частот. Это обусловлено частыми изменениями уровня сигнала (рис. 3.30).

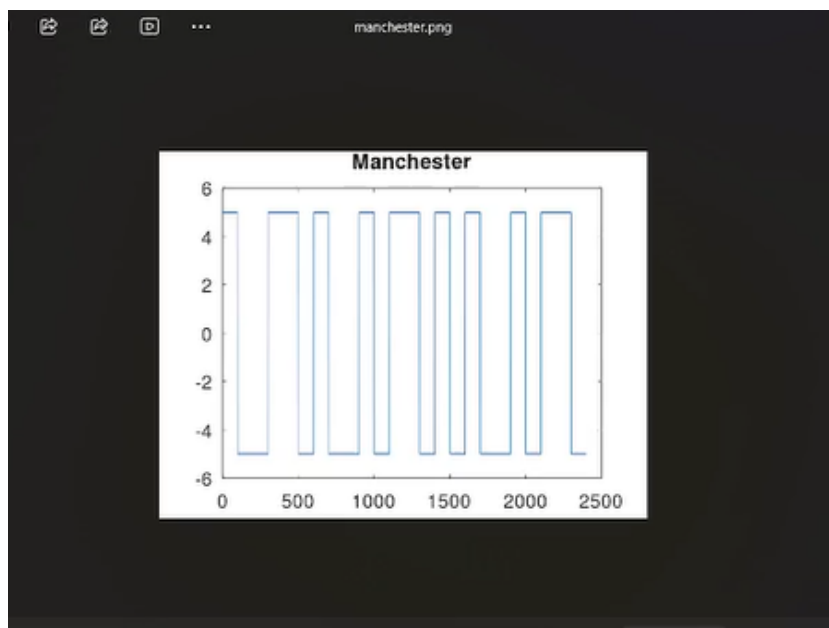


Рис. 3.30: Спектр дифференциального манчестерского кодирования

Спектр манчестерского кодирования схож с дифференциальным вариантом. Основная часть энергии сосредоточена на частотах, близких к тактовой (рис. 3.31).

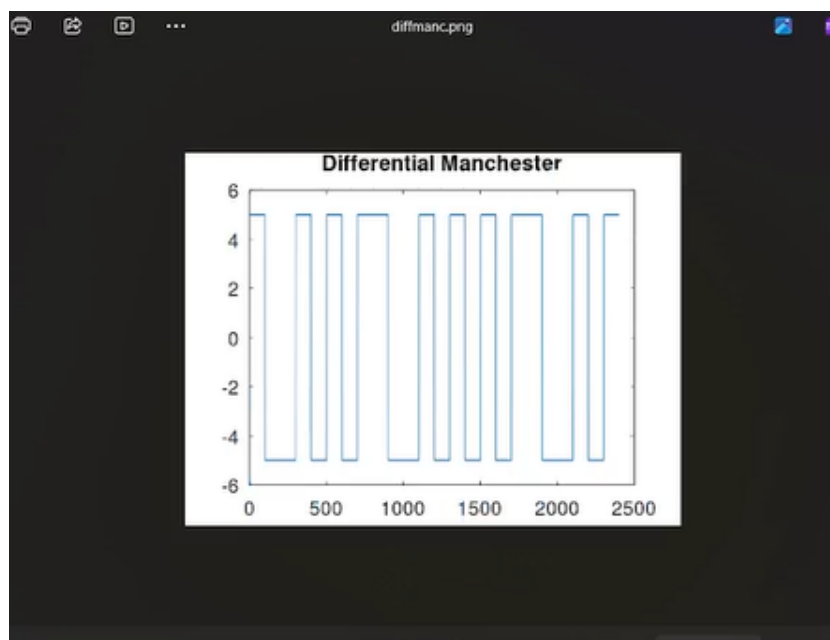


Рис. 3.31: Спектр манчестерского кодирования

Для сравнения я также рассчитала спектр униполярного кода. Он характеризуется наличием ярко выраженной постоянной составляющей (рис. 3.32).

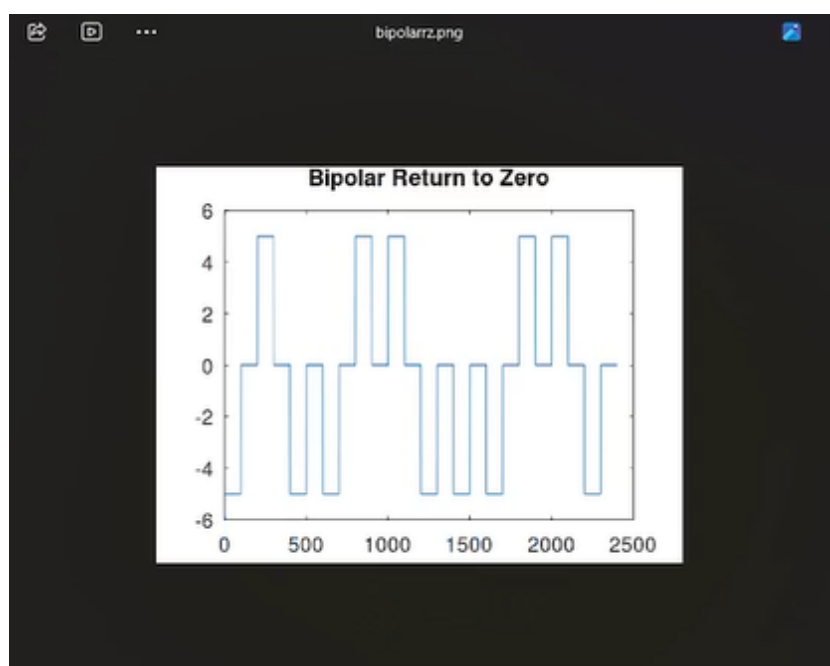


Рис. 3.32: Спектр униполярного кодирования

3.7 Исследование самосинхронизации

Для проверки устойчивости синхронизации я подала на вход последовательность с длинной серией нулей. В коде АМІ в этот момент переходы отсутствуют, что ведет к потере синхронизации (рис. 3.33).

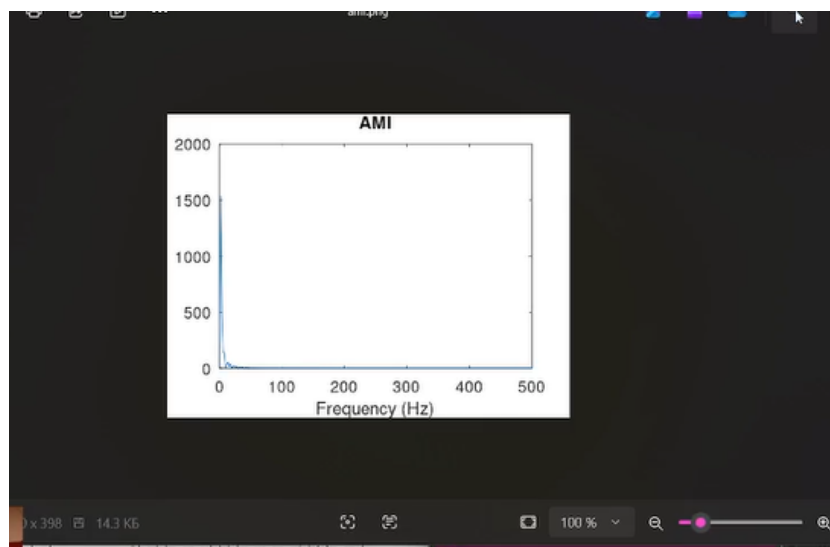


Рис. 3.33: График АМІ для проверки самосинхронизации

Аналогичная проблема наблюдается у кода NRZ. При передаче длинной серии единиц или нулей сигнал остается неизменным, и приемник не может выделить такты (рис. 3.34).

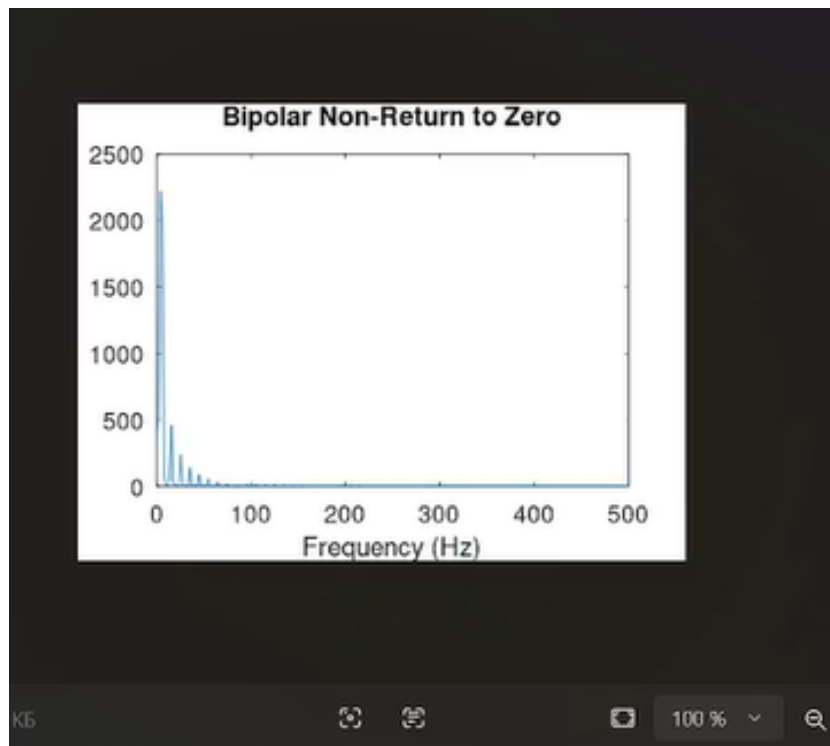


Рис. 3.34: График NRZ для проверки самосинхронизации

Код RZ успешно решает проблему самосинхронизации при передаче единиц, так как в каждом такте есть возврат к нулю. Однако серия нулей по-прежнему остается проблемной зоной (рис. 3.35).

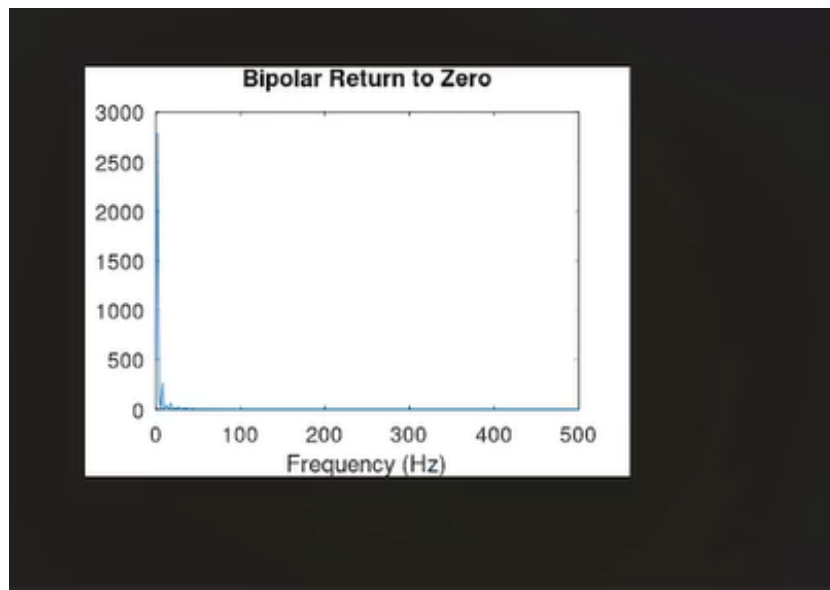


Рис. 3.35: График RZ для проверки самосинхронизации

Дифференциальный манчестерский код обеспечивает идеальную самосинхронизацию. Независимо от передаваемых данных, в сигнале всегда присутствуют переходы (рис. 3.36).

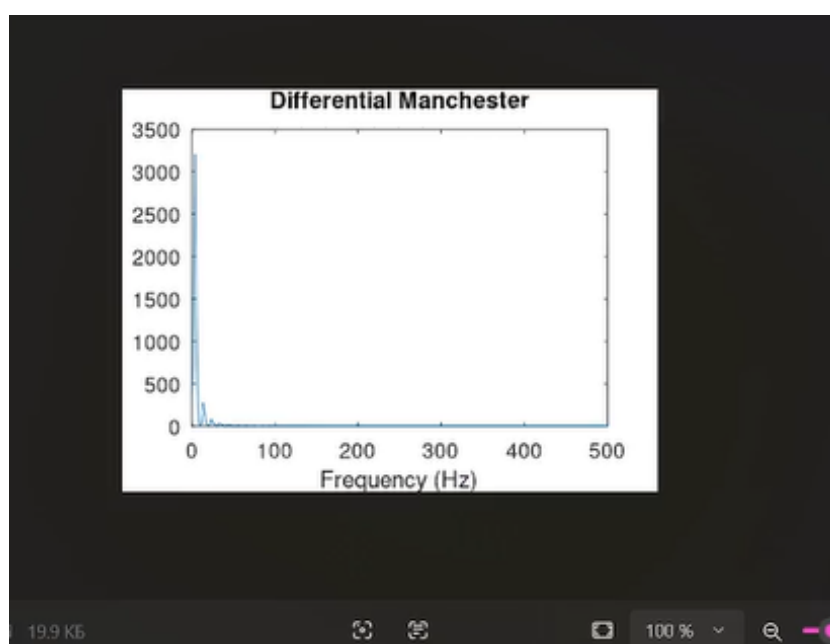


Рис. 3.36: График дифференциального манчестерского кодирования для проверки синхронизации

Манчестерский код также демонстрирует отличные свойства самосинхронизации. Обязательный переход в центре каждого бита позволяет приемнику работать стабильно (рис. 3.37).

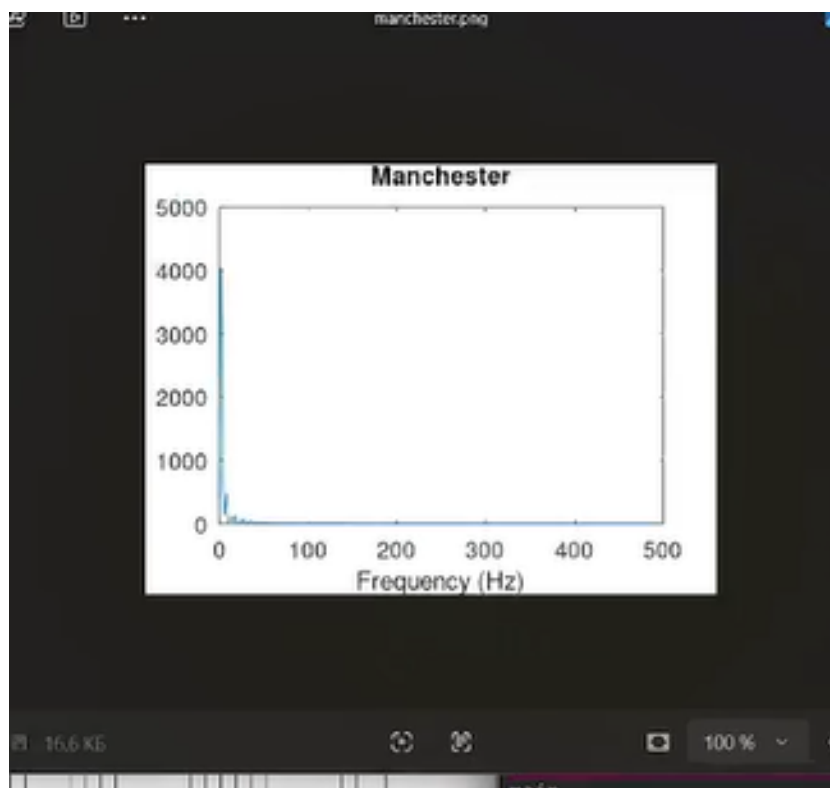


Рис. 3.37: График манчестерского кодирования для проверки синхронизации

В завершение тестов я проверила униполярный код на длинной серии нулей. Результат подтвердил отсутствие переходов и невозможность восстановления синхронизации (рис. 3.38).

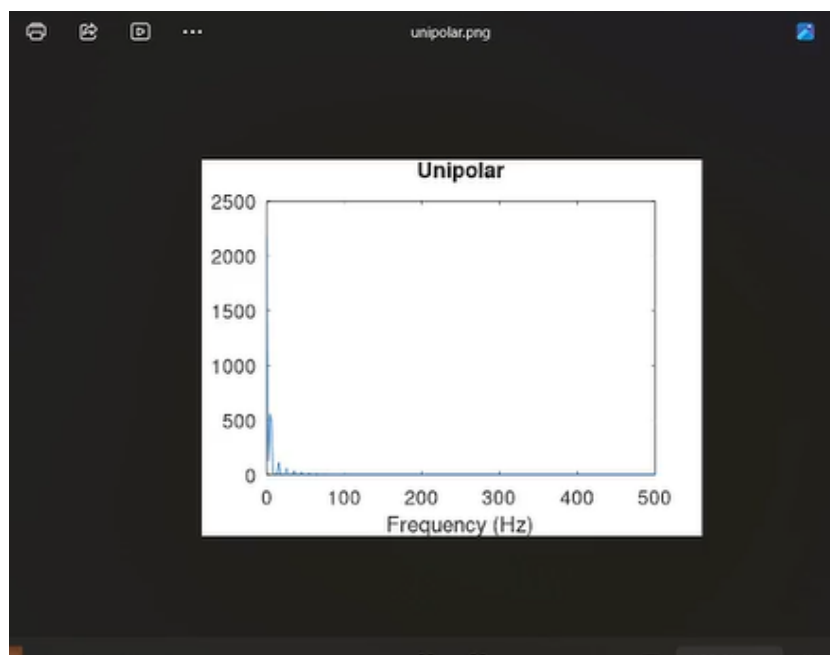


Рис. 3.38: График униполярного кодирования для проверки синхронизации

3.8 Дополнительные результаты анализа

Ниже представлены дополнительные графики спектральных плотностей мощности и временные диаграммы, полученные в ходе уточняющих экспериментов (рис. 3.39 - 3.45).

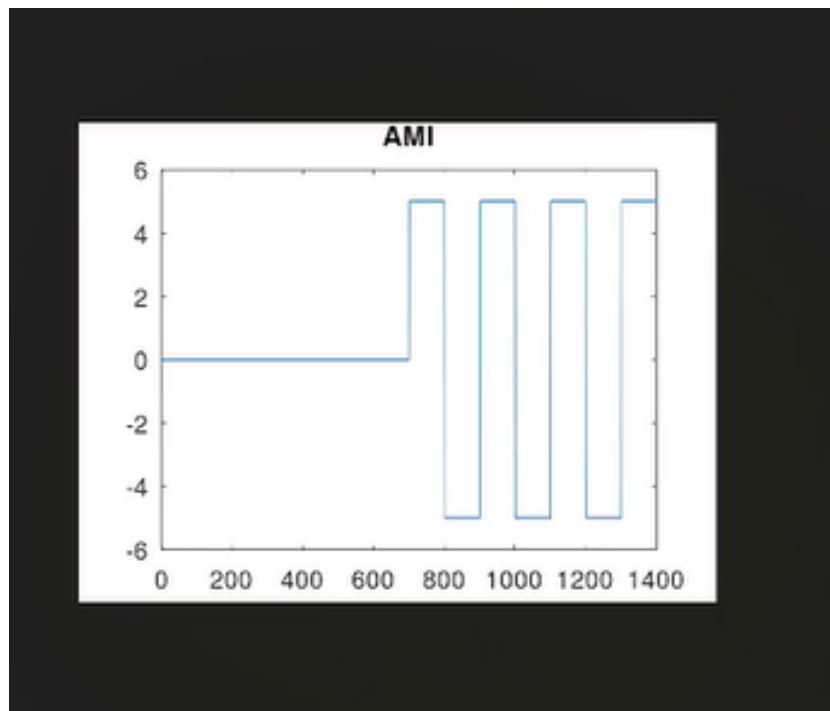


Рис. 3.39: Дополнительный анализ спектральной плотности №1

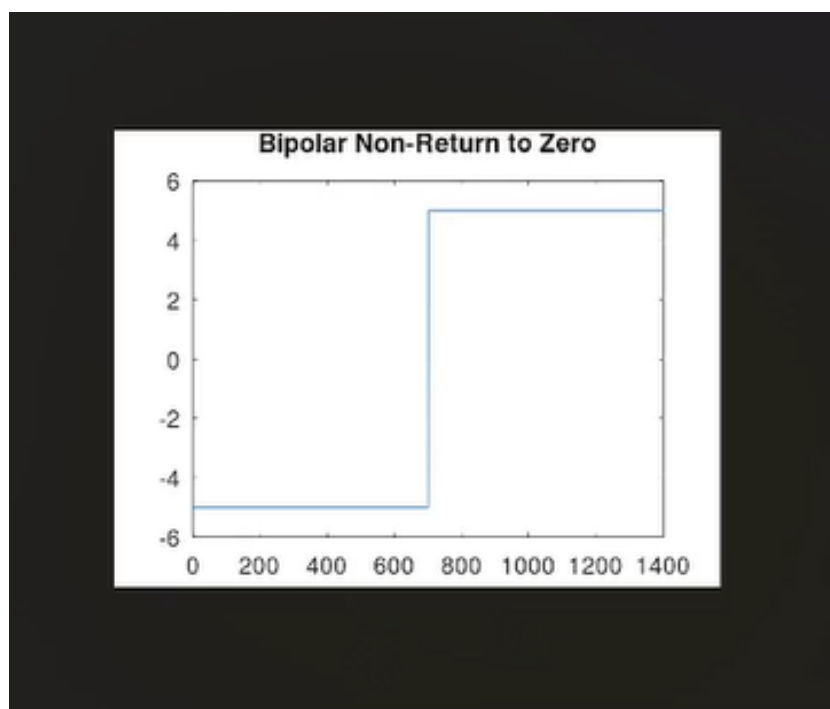


Рис. 3.40: Дополнительный анализ спектральной плотности №2

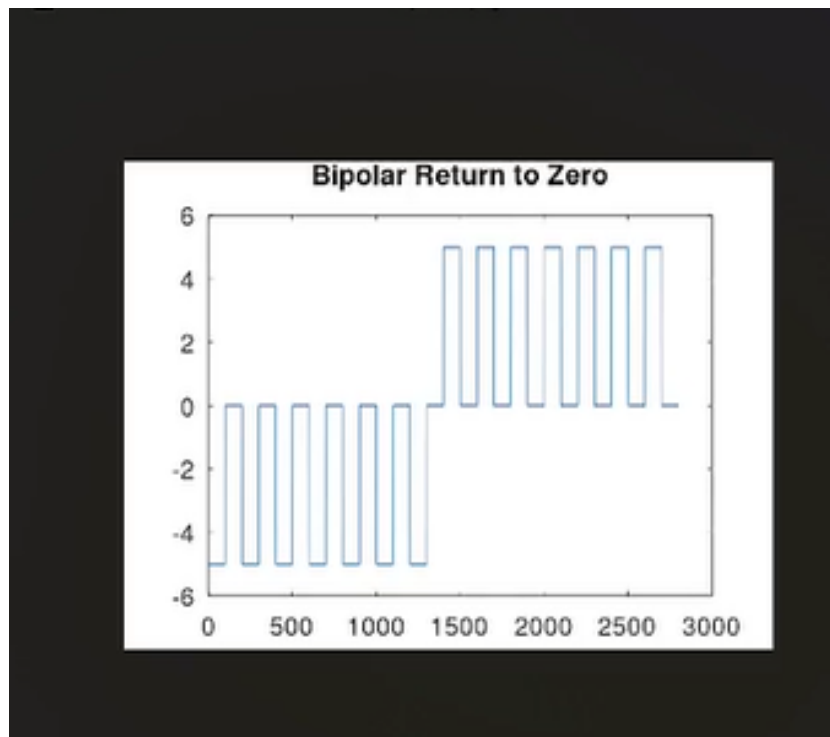


Рис. 3.41: Дополнительный анализ спектральной плотности №3

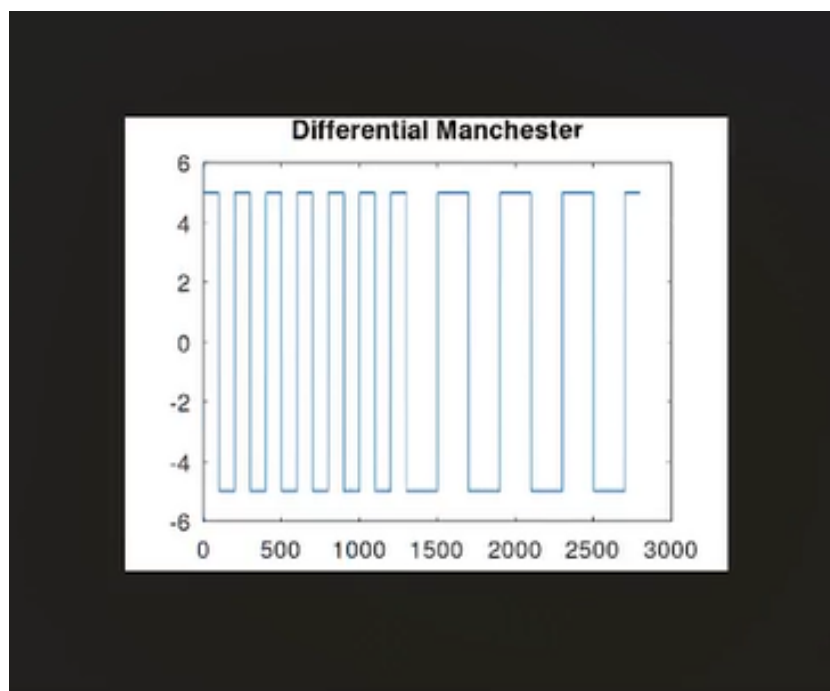


Рис. 3.42: Уточняющая временная диаграмма кодирования №1

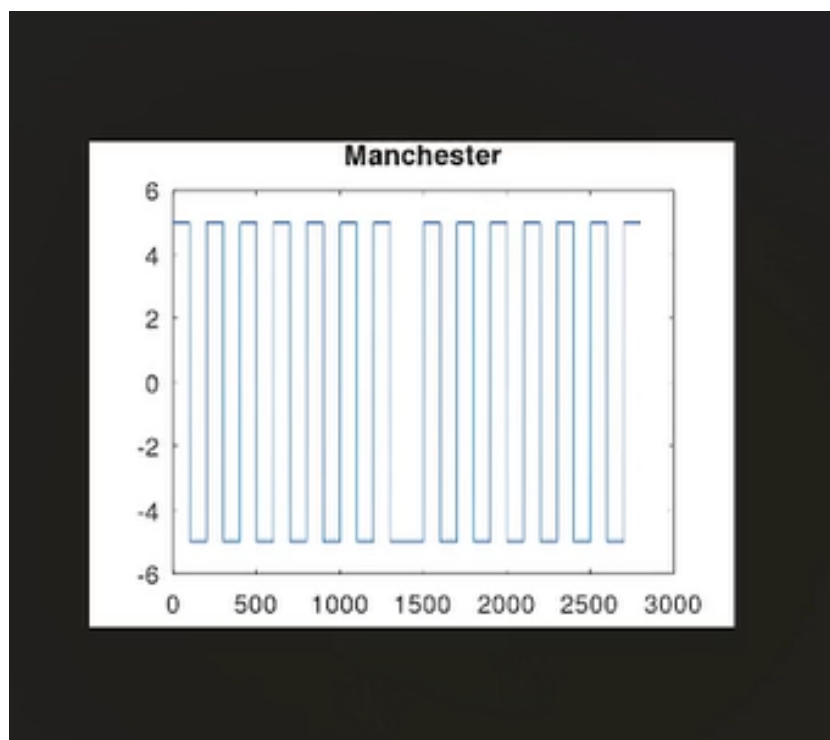


Рис. 3.43: Уточняющая временная диаграмма кодирования №2

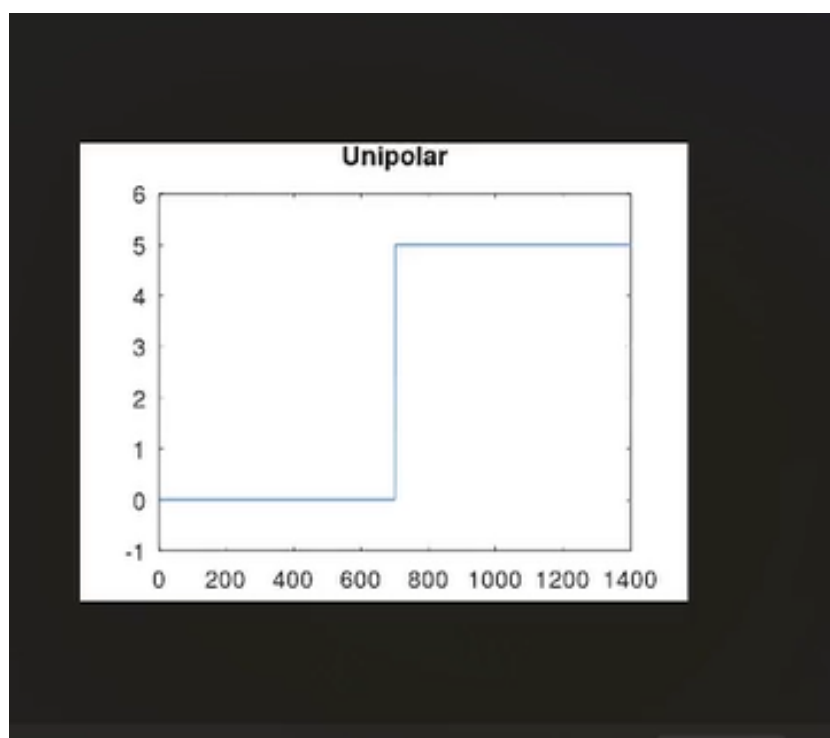


Рис. 3.44: Уточняющая временная диаграмма кодирования №3

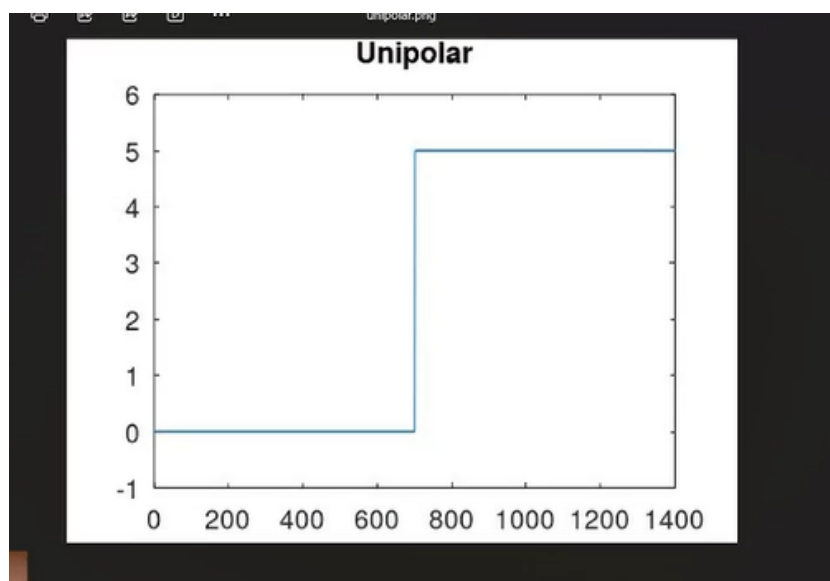


Рис. 3.45: Сводный график результатов экспериментов

4 Выводы

В ходе выполнения лабораторной работы я освоила методы математического моделирования сигналов в среде Octave. Были успешно реализованы алгоритмы спектрального анализа на основе БПФ и моделирование амплитудной модуляции.

Особое внимание было уделено изучению методов линейного кодирования. В результате сравнительного анализа было установлено, что манчестерские коды обладают наилучшими свойствами самосинхронизации, в то время как коды NRZ и AMI требуют специальных методов для предотвращения потери синхронизации при передаче длинных серий одинаковых бит. Полученные навыки необходимы для понимания принципов работы современных систем передачи данных.

5 Ответы на контрольные вопросы

1. **Что такое спектр сигнала?** Это распределение амплитуд или мощностей составляющих гармонических колебаний сигнала по частотам.
2. **В чем преимущество манчестерского кодирования?** Оно обладает свойством самосинхронизации и не имеет постоянной составляющей, что упрощает передачу данных.
3. **Зачем нужна модуляция?** Модуляция позволяет перенести низкочастотный информационный сигнал в высокочастотную область для эффективной передачи на расстояние через физическую среду.