Лабораторная работа №1

Сетевые технологии

Бансимба Клодели Дьегра 1032215651

НПИбд02-22 .

12 сентября 2024

Российский университет дружбы народов имени Патриса Лумумбы, Москва, Россия

Цель работы

Изучить методы кодирования и модуляции сигналов с помощью высокоуровнего языка программирования octave. Определить спектр и параметры сигнала. Продемонстрировать принципы модуляции сигнала на примере аналоговой амплитудной модуляции. Исследовать свойства самосинхронизации сигнала.

Задание

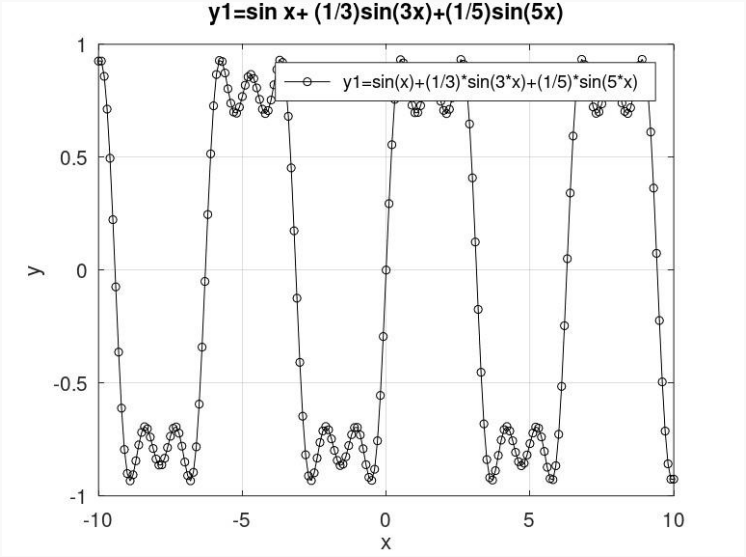
1. Построить графики в octave;
2. Разложить импульсный сигнал в частичный ряд Фурье;
3. Определить спектр и параметры сигнала;
4. Продемонстрировать принципы модуляции сигнала на примере аналоговой амплитудной модуляции;
5. По заданным битовым последовательностям требуется получить кодированные сигналы для нескольких кодов, проверить свойства самосинхронизуемости кодов,получить спектры.

Задание №1

Построение графика функции:

```
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)
```

Задание №1

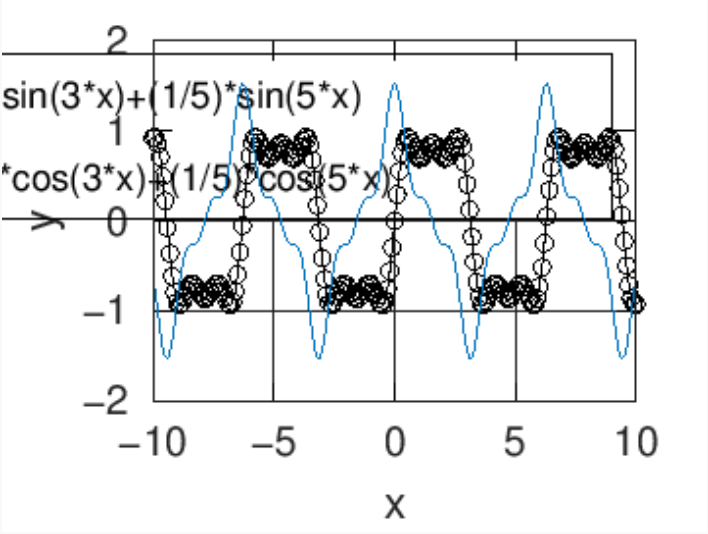

y1=sin x+ (1/3)sin(3x)+(1/5)sin(5x)
y1=sin(x)+(1/3)*sin(3*x)+(1/5)*sin(5*x)
y
x
1
0.5
0
-0.5
-1
-10
-5
0
5
10

Задание №1

Далее я изменила сценарий так, чтобы на одном графике располагались отличающиеся по типу линий графики функций $y1 = \sin x + (1/3)\sin 3x + (1/5)\sin 5x$
$y2 = \cos x + (1/3)\cos 3x + (1/5)\cos 5x$

```
plot(x,y1, "-ok; y1=sin(x)+(1/3)*sin(3*x)+(1/5)*sin(5*x);","markersize",4)
hold on
plot(x,y2, "-; y2=cos(x)+(1/3)*cos(3*x)+(1/5)*cos(5*x);","markersize",4)
```

Задание №1



Задание №2

Разложение импульсного сигнала в форме меандра в частичный ряд Фурье можно задать формулой:

$$s(t) = \frac{A}{2} + \frac{2A}{\pi}cos(\frac{2\pi}{T}t) - \frac{1}{3}cos(s\frac{2\pi}{T}t) + \frac{1}{5}cos(5\frac{2\pi}{T}t) - \dots$$

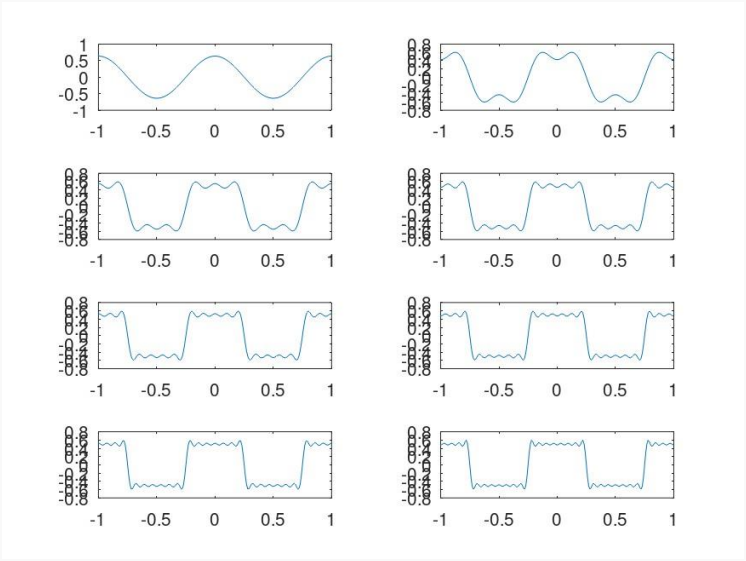
или формулой:

$$s(t) = \frac{A}{2} + \frac{2A}{\pi}sin(\frac{2\pi}{T}t) - \frac{1}{3}sin(s\frac{2\pi}{T}t) + \frac{1}{5}sin(5\frac{2\pi}{T}t) - \dots$$

Задание №2

```
nh=(1:N)*2-1;
Am=2/pi ./ nh;
Am(2:2:end) = -Am(2:2:end);
harmonics=cos(2 * pi * nh' * t/T);
s1=harmonics.*repmat(Am',1,length(t));
s2=cumsum(s1);
for k=1:N
subplot(4,2,k)
plot(t, s2(k,:))
end
```

Задание №2

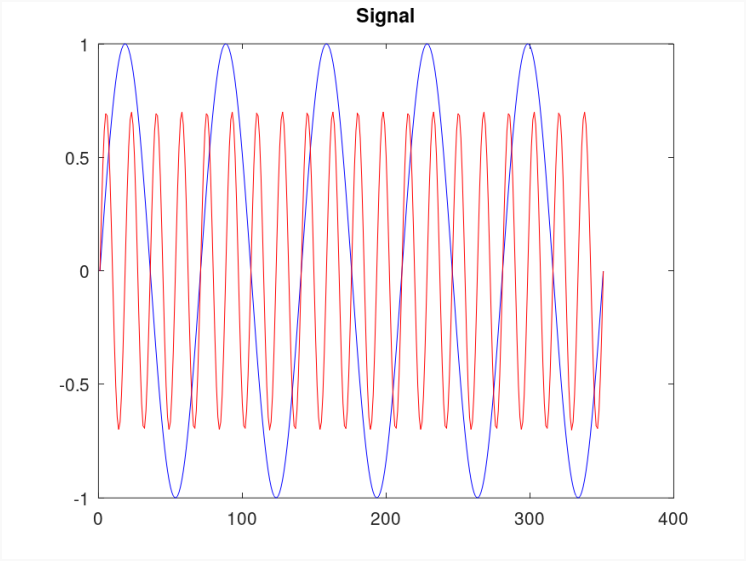


Задание №3

Определим спектр двух отдельных сигналов и их суммы.

```
tmax = 0.5;
fd = 512;
f1 = 10;
f2 = 40;
a1 = 1;
a2 = 0.7;
t = 0:1./fd:tmax;
fd2 = fd/2;
signal1 = a1*sin(2*pi*t*f1);
signal2 = a2*sin(2*pi*t*f2);
```

Задание №3

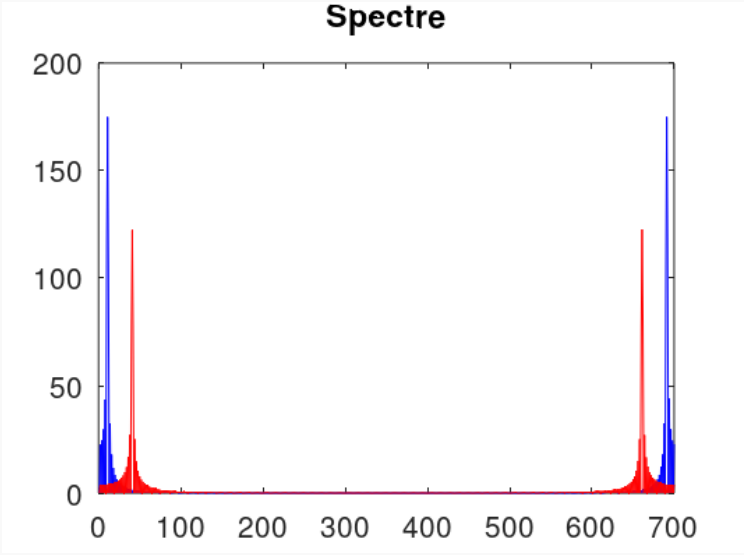


Задание №3

Используем быстрое преобразование Фурье, чтобы найти спектры сигналов.

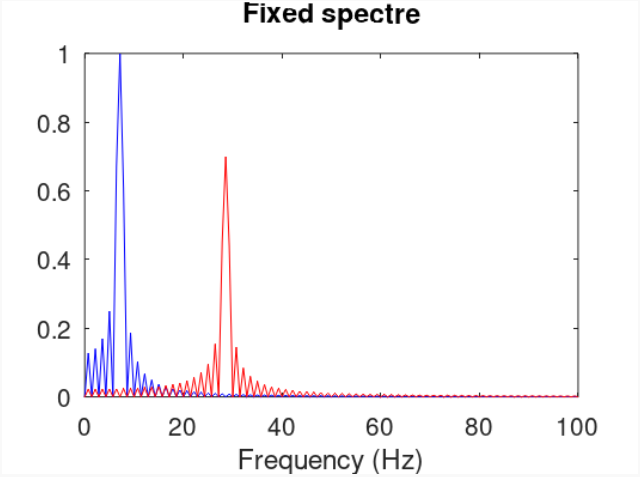
```
spectre1 = abs(fft(signal1,fd));
spectre2 = abs(fft(signal2,fd));
plot(spectre1,'b');
hold              on
plot(spectre2,'r');
```

Задание №3



Задание №3

Учитывая некоторые неточности преобразования Фурье, нужно скорректировать график спектра.

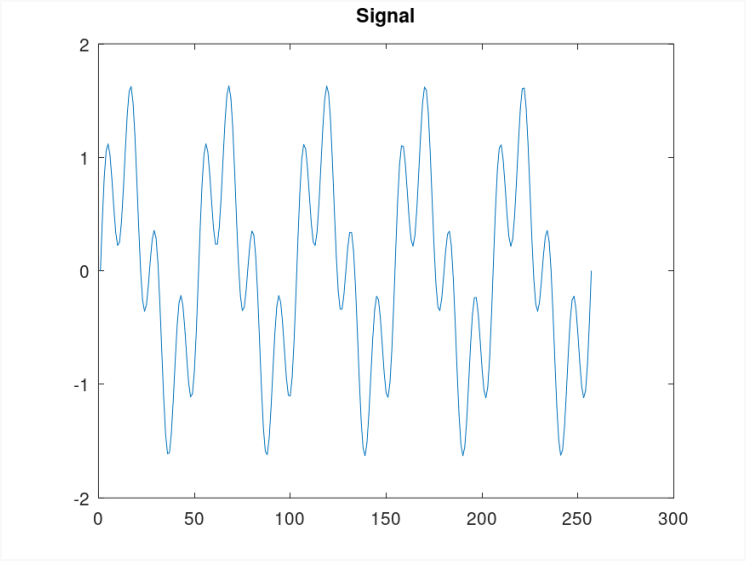


Задание №3

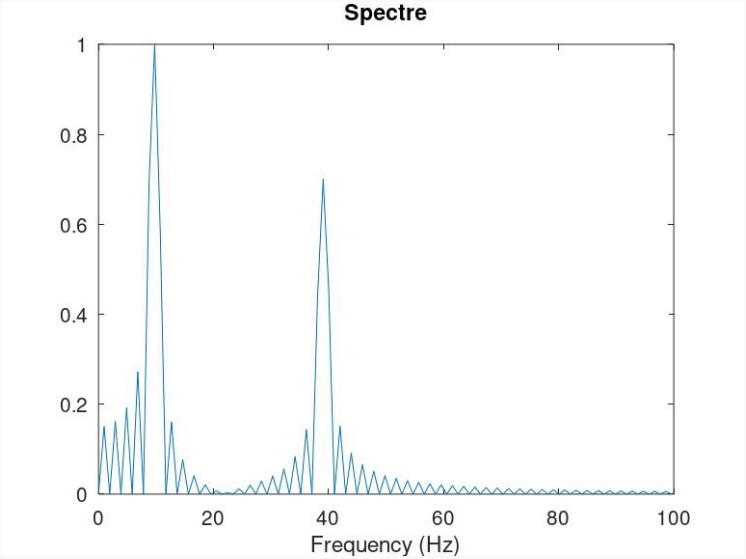
Спектр суммы рассмотренных сигналов:

```
% Сумма двух сигналов (синусоиды) разной частоты:
t = 0:1./fd:tmax;
signal1 = a1*sin(2*pi*t*f1);
signal2 = a2*sin(2*pi*t*f2);
signal = signal1 + signal2;
% Подсчет спектра:
spectre = fft(signal,fd);
f = 1000*(0:fd2)./(2*fd);
spectre = 2*sqrt(spectre.*conj(spectre))./fd2;
```

Задание №3

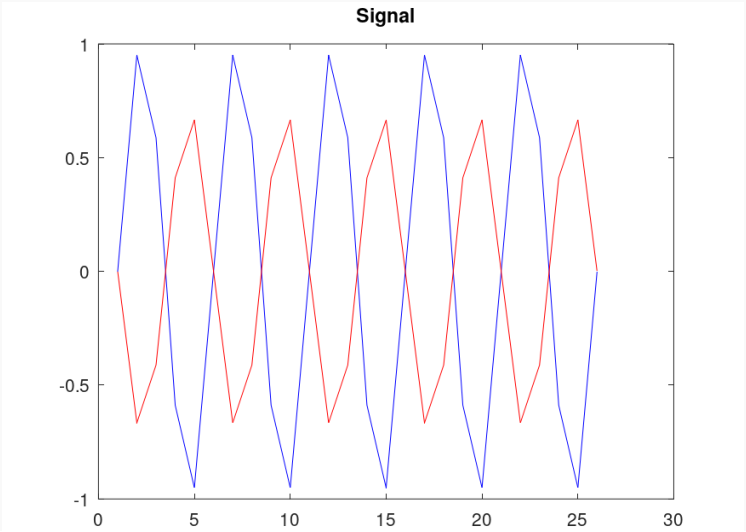


Задание №3



Задание №3

Пример графика с частотой дискретизации меньше 80 Гц:

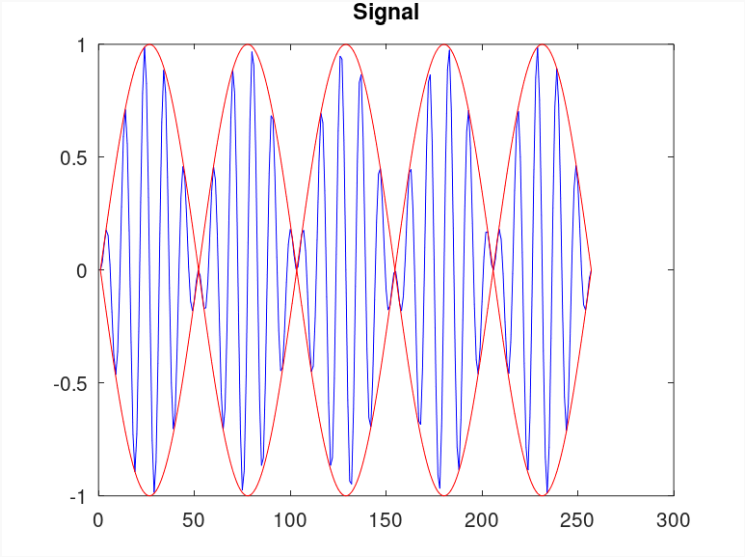


Задание №4

Продемонстрируем аналоговую амплитудную модуляцию:

```
tmax = 0.5;
fd = 512;
f1 = 5;
f2 = 512;
fd2 = fd/2;
t  =  0:1./fd:tmax;
signal1 = sin(2*pi*t*f1);
signal2 = sin(2*pi*t*f2);
signal = signal1 .* signal2;
plot(signal, 'b');
plot(signal1, 'r');
plot(-signal1, 'r');
```

Задание №4

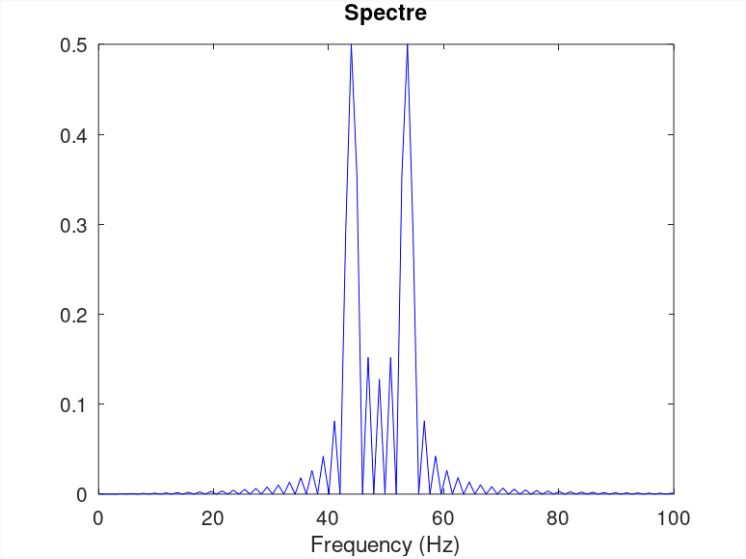


Задание №4

Далее построим спектр произведения, который представляет собой свертку спекторв.

```
spectre = fft(signal,fd);
f = 1000*(0:fd2)./(2*fd);
spectre = 2*sqrt(spectre.*conj(spectre))./fd2;
plot(f,spectre(1:fd2+1), 'b')
```

Задание №4



Задание №5

Содержание файла main.m :

```
% Входная кодовая последовательность:
data=[0 1 0 0 1 1 0 0 0 1 1 0];
% Входная кодовая последовательность для проверки свойства самосинхронизации:
data_sync=[0 0 0 0 0 0 0 1 1 1 1 1 1 1 1];
% Входная кодовая последовательность для построения спектра сигнала:
data_spectre=[0 1 0 1 0 1 0 1 0 1 0 1 0 1];
% Униполярное кодирование
wave=unipolar(data);
plot(wave);
% Кодирование ami
wave=ami(data);
plot(wave)
```

Задание №5

В файле maptowave.m прописала функцию, которая по входному битовому потоку строит график сигнала:

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

Задание №5

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

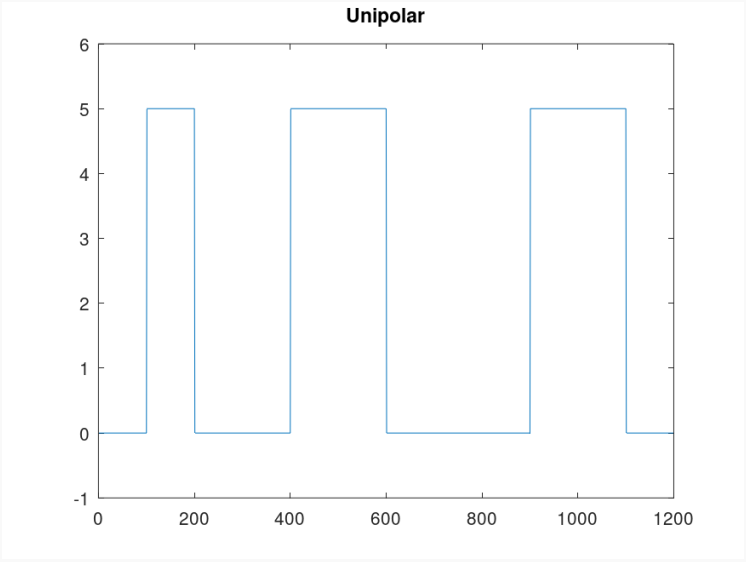
```
% Униполярное кодирование:
function wave=unipolar(data)
wave=maptowave(data);
% Кодирование AMI:
function wave=ami(data)
am=mod(1:length(data(data==1)),2);
am(am==0)=-1;
data(data==1)=am;
wave=maptowave(data);
```

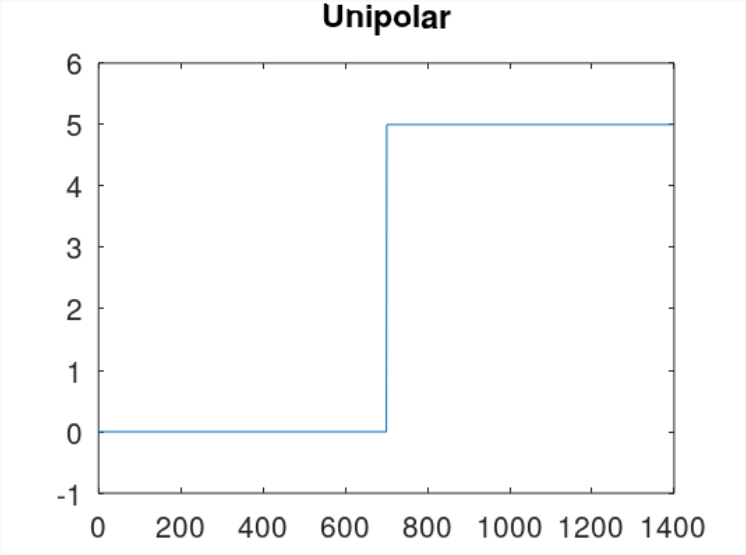
Задание №5

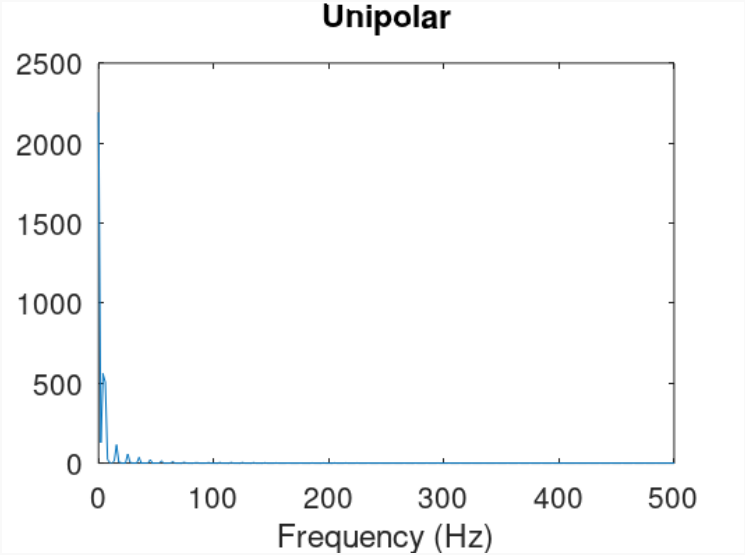
В файле calcspectre.m пропишим функцию построения спектра сигнала:

```
% Функция построения спектра сигнала:
function spectre = calcspectre(wave)
Fd = 512;
Fd2  =  Fd/2;
Fd3 = Fd/2 + 1;
X = fft(wave,Fd);
spectre = X.*conj(X)/Fd;
f = 1000*(0:Fd2)/Fd;
plot(f,spectre(1:Fd3));
```

Задание №5




Unipolar
6
5
4
3
2
1
0
-1
0
200
400
600
800
1000
1200
1400


Unipolar
2500
2000
1500
1000
500
0
0
100
200
300
400
500
Frequency (Hz)

Выводы

В процессе выполнения данной лабораторной работы я изучил методы кодирования и модуляции сигналов с помощью высокоуровнего языка программи- рования octave.

Определил спектр и параметры сигнала.

Показал принципы модуляции сигнала на примере аналоговой амплитудной модуляции. А также исследовала свойства самосинхронизации сигнала