Отчет по лабораторным работам по дисциплине ТСС

Никитина А.В., Замотаева Ю.И.

2014

Содержание

1 Система верстки TEXи расширения LATEX

1.1 Цель работы

Ознакомление с программой для верстки отчетов TEX.

1.2 Задача работы

Научиться создавать титульный лист, разделы, списки и писать несложные формулы в среде верстки отчетов TEX.

1.3 Написание математических формул

1. $$lim_{n\to\infty}\sum_{k=1}^{n}\frac{25}{2*k^3}=\frac{\pi}{2}$$

2. $$x^3+\sum_{x=1}^{8}\frac{x}{2}=\sqrt{617}$$

3. $$x(t)=A*sin(w*t+f)$$

Формула интеграла:

$$\iint_{\frac{sin(x)}{x}+y^2=1} f(x,y)dxdy$$

1.4 Интерпретация результатов

Среда верстки отчетов TEXочень удобна для введения математических формул. Имеется множество команды, которые заметно облегчают ввод дробных выражений, констант, переменных, производных, интегралов, дифференциалов и других формул. Таким образом, можно без особы трудностей вводить формулы различной сложности.

1.5 Вывод

В данной лабораторной работе мы познакомились с системой создания и редактирования текстов TEXи расширением LATEX. Рассмотрели команды этой среды, которые помогают составлять красивые и аккуратные отчеты. Таким образом, теперь мы без труда может написать отчет, который будет иметь титульный лист, главы, заголовки и различные математические формулы. Нам очень понравилась эта система, потому что она помогает каждому составить отчет быстро и удобно. Система полностью автоматизирована, от нас требуется только освоить основные команды и ознакомиться с дополнительной литературой.

2 Визуализация сигналов во временной и частотной области

2.1 Цель работы

Познакомиться со средствами генерации сигналов и визуализации их спектров.

2.2 Постановка задачи

В командном окне MATLAB и в среде Simulink промоделировать чистый синусоидальный сигнал, так же синусоидальный сигнал с шумом. Получить их спектры.

2.3 Справочные материалы

В.С. Гутников Фильтрация измерительных сигналов пп.1-2, 11-12

2.4 Ход работы

В среде MATLAB вначале моделируем чистый синусоидальный сигнал по формуле:

$$A(t) = A_0 sin(2\pi ft + f_0)$$

Затем моделируем зашумленный сигнал по формуле:

$$A(t) = A_0 sin(2\pi ft + f_0) + A_1 rand()$$

Для выделения частот регулярных составляющих используем преобразование Фурье, которое реализовано встроенной в MATLAB функцией.

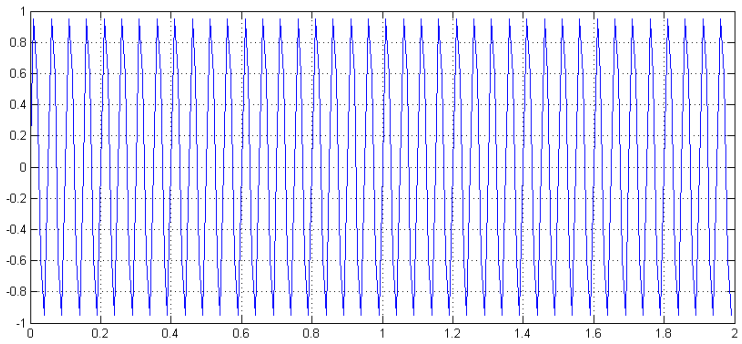
2.4.1 Код MATLAB для исходного сигнала

```
x=0:0.01:4*pi;
f0 = 20;
%исходный сигнал
y = sin(2*pi*f0*x);
figure(1)
plot(x(1:200),y(1:200))
grid
%спектр исходного сигнала
figure(2)
spectrum = fft(y,512);
normspectrum = spectrum.*conj(spectrum)/512;
f=100*(0:255)/512;
plot(f, normspectrum(1:256))
axis([0 max(F) 0 10])
grid
```

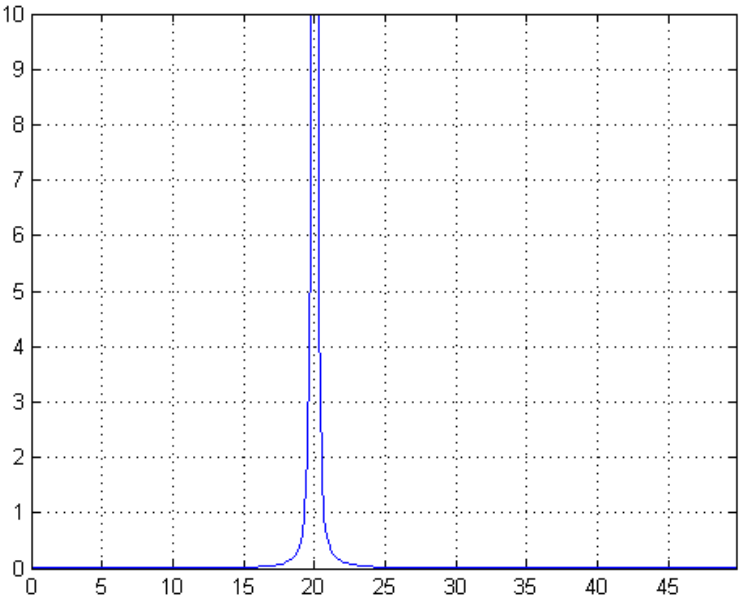
2.4.2 Результаты работы программы

В результате выполнения программы получились графики временной и частотной характеристик исходного синусоидальных сигнала.

- Исходный сигнал



- Спектр исходного сигнала



2.4.3 Код MATLAB для зашумленного сигнала

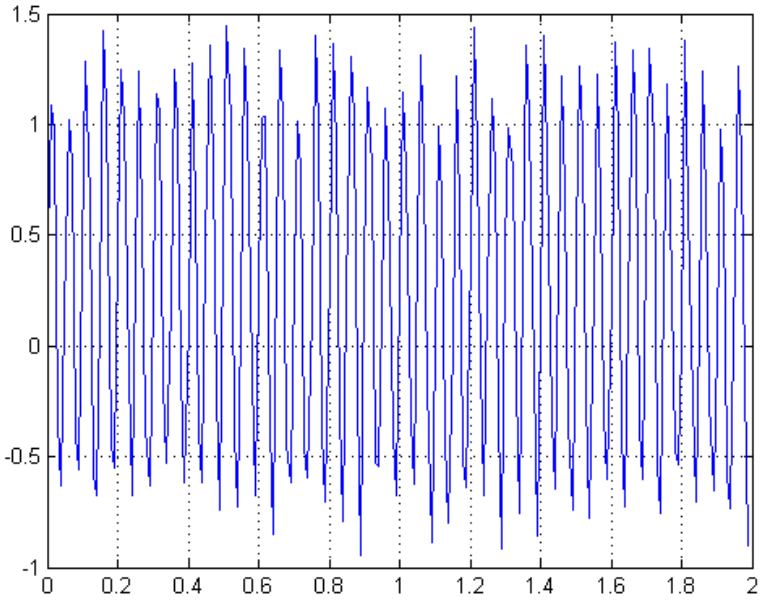
```
%зашумленный сигнал
ynoize = y+ 0.5*rand(size(x));
figure(3)
```

```
plot(x(1:200),ynoize(1:200));
grid
%спектр зашумленного сигнала
spectrum = fft(ynoize,512);
noizespectrum = spectrum.*conj(spectrum)/512;
figure(4)
plot(f, noizespectrum())
axis([0 max(f) 0 10])
grid
```

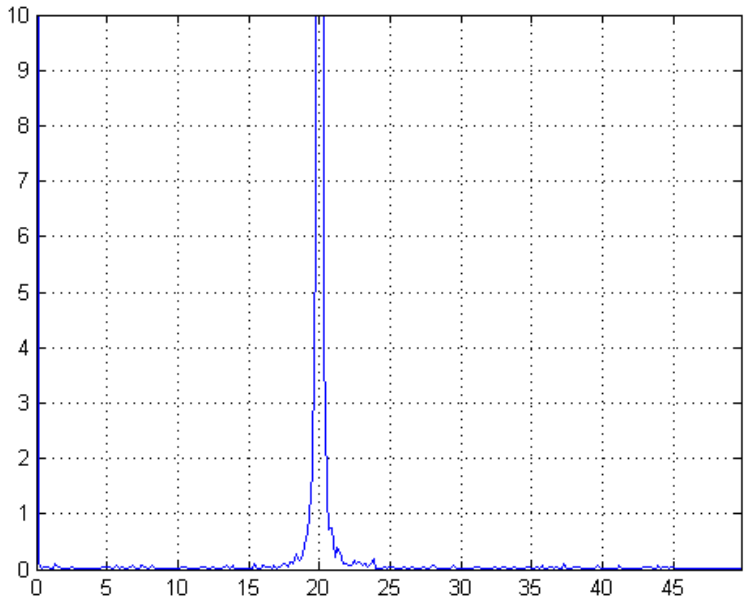
2.4.4 Результаты работы программы

В результате выполнения программы получились графики временной и частотной характеристик зашумленного синусоидальных сигнала.

- Зашумленный сигнал



- Спектр зашумленного сигнала



2.5 Вывод

В данной работе мы построили исходный синусоидальный сигнал, после чего получили его спектр. Спектр представляет из себя резко возрастающий график. Далее построили зашумленный сигнал и его спектр. По рисункам видно, что спектр зашумленного сигнала очень напонимает спектр исходного сигнала, но является более неровным. Стоит пояснить возможные расхождения между теоретически ожидаемым и полученным спектром сигнала вследствие дискретности представления сигналов, а также их конечности во времени: Любой сигнал с неограниченным спектром может быть представлен с некоторыми потерями как сигнал с ограниченным спектром. В силу дискретности представления сигналов мы теряем некоторое количество информации, таким образом, получаем искаженный спектр. Спектр дискретизированного сигнала получаем в результате сверки непрерывного сигнала с решеткой дельта-импульса.

$$S(k * f) = S(f) * \delta(k * f_n)$$

А так как мы строим конечный косинус путем умножения на прямоугольное окно, то получаем свертку sin(x)/x, т. е. спектр в виде нескольких(т.к.сигнал дискретизирован) sinc(x).

3 Спектры простых сигналов

3.1 Цель работы

Получить представление о тестовых сигналах во временной и частотной областях. Реализовать операцию свертки сигналов.

3.2 Постановка задачи

В командном окне MATLAB и в среде Simulink промоделировать следующие тестовые сигналы:

- Полигармонический сигнал

$$y(t) = \sum_{n=0}^{N-1} cos(nt) \tag{1}$$

- Прямоугольный импульсный сигнал

$$y(t) = \prod(t, T_i) \tag{2}$$

- Треугольный импульсный сигнал

$$y(t) = \triangle\ (t, T_i) \tag{3}$$

и получить их спектры.

3.3 Справочные материалы

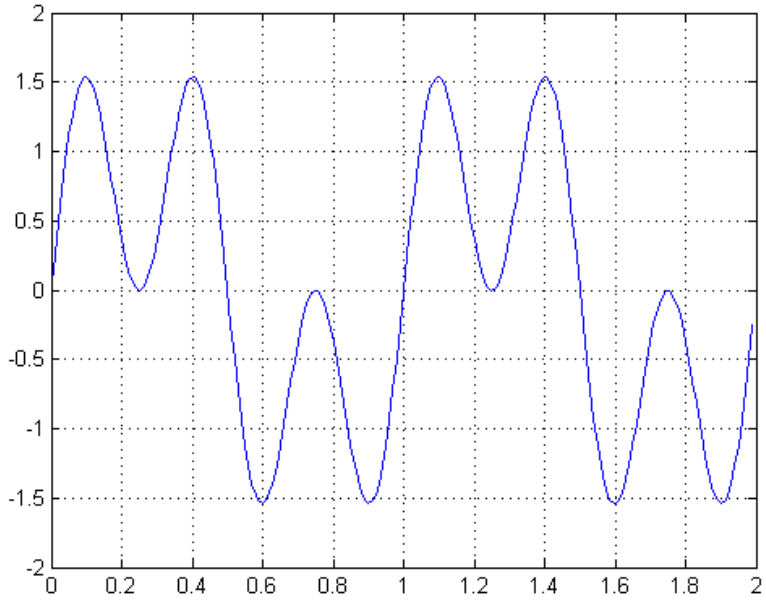
В.С. Гутников. Фильтрация измерительных сигналов пп.3-6, 13-14

3.4 Полигормонический сигнал

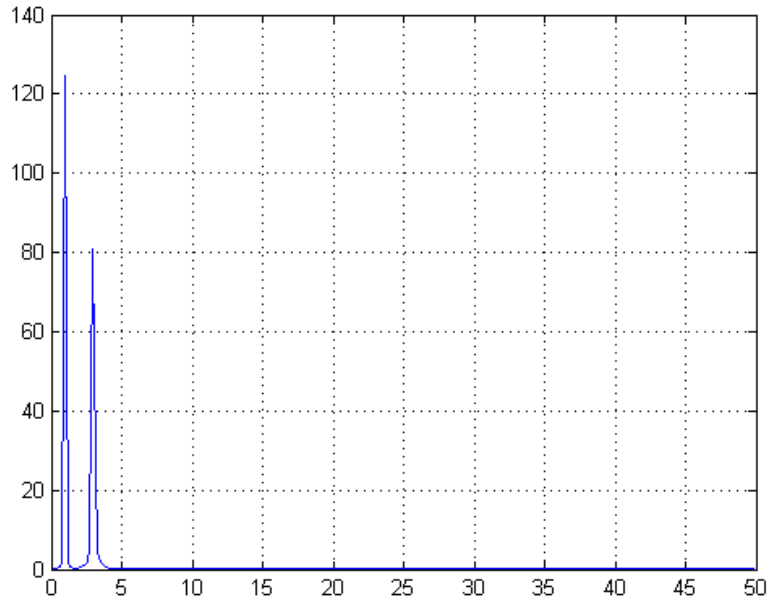
3.4.1 Код Matlab

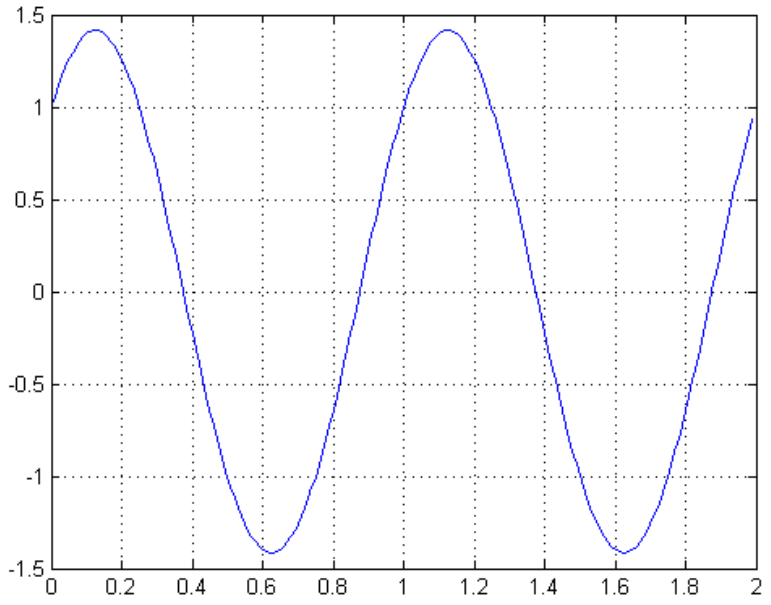
```
x = 0:0.01:4*pi;
f=100*(0:255)/512;
figure
y1 = sin(2*pi*x)+sin(2*pi*3*x);
plot(x(1:200),y1(1:200))
grid
figure
s1 = fft(y1,512);
ss1 = s1.*conj(s1)/512;
plot(f,ss1(1:256))
grid
figure
y2 = sin(2*pi*x)+cos(2*pi*x);
plot(x(1:200),y2(1:200))
grid
figure
s2 = fft(y2,512);
ss2 = s2.*conj(s2)/512;
plot(f,ss2(1:256))
grid
```

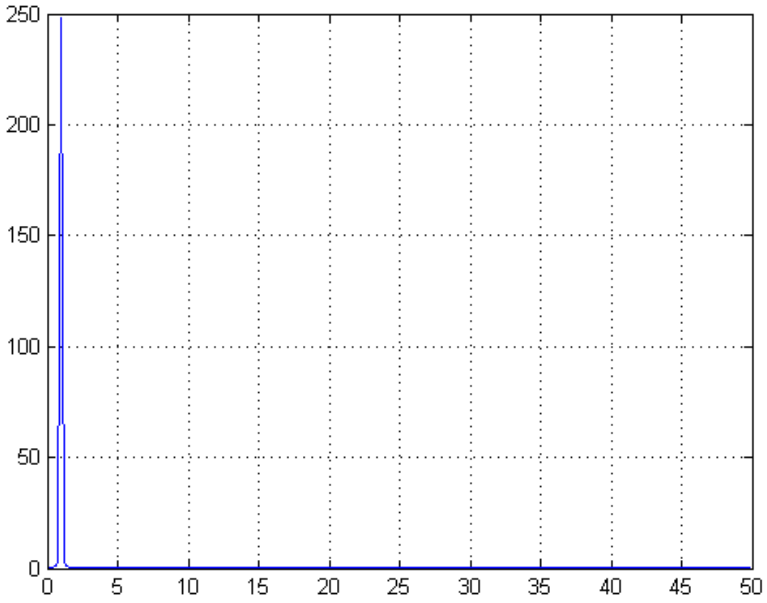
- полигармонический сигнал sin(x)+sin(3x)



- спектр данного полигармонического сигнала
- полигармонический сигнал sin(x)+cos(x)
- спектр данного полигармонического сигнала

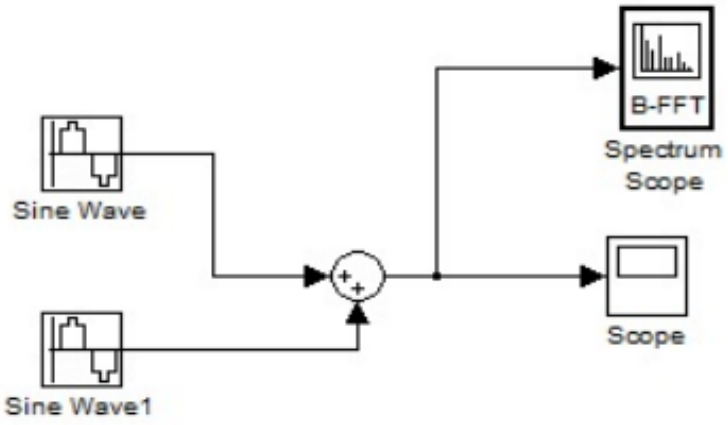

140
120
100
80
60
40
20
0
0
5
10
15
20
25
30
35
40
45
50


1.5
1
0.5
0
-0.5
-1
-1.5
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

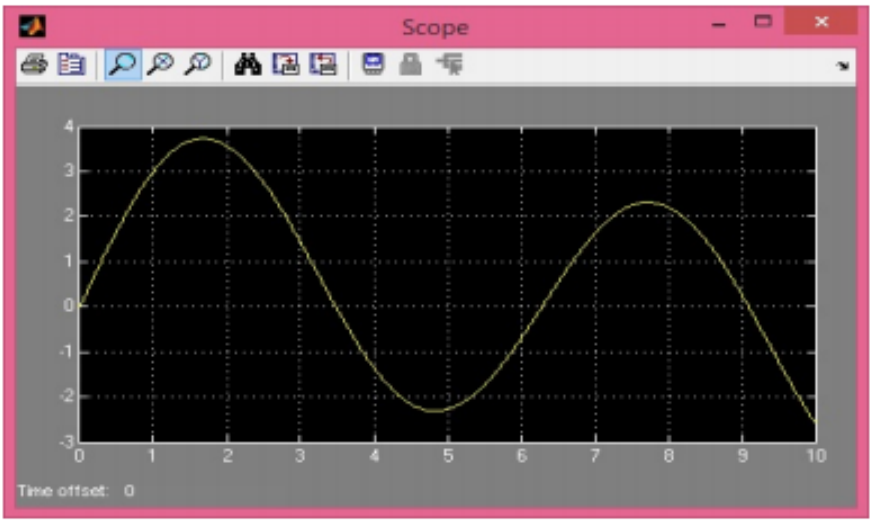

250
200
150
100
50
0
0
5
10
15
20
25
30
35
40
45
50

3.4.2 Модуляция в среде Simulink

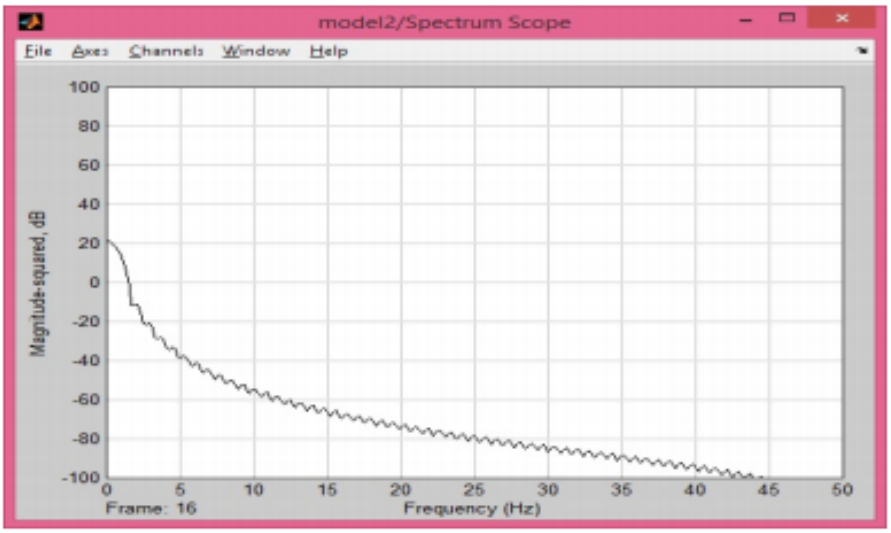
- модель



- сигнал



- спектр

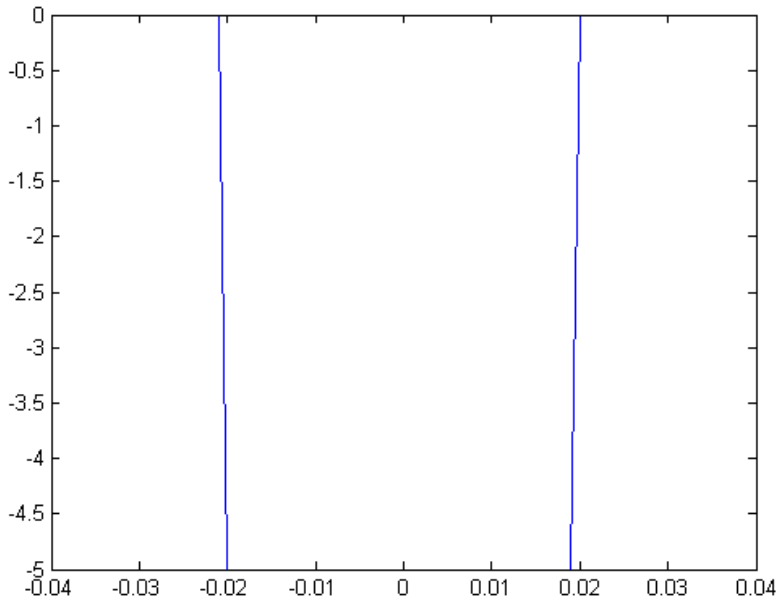


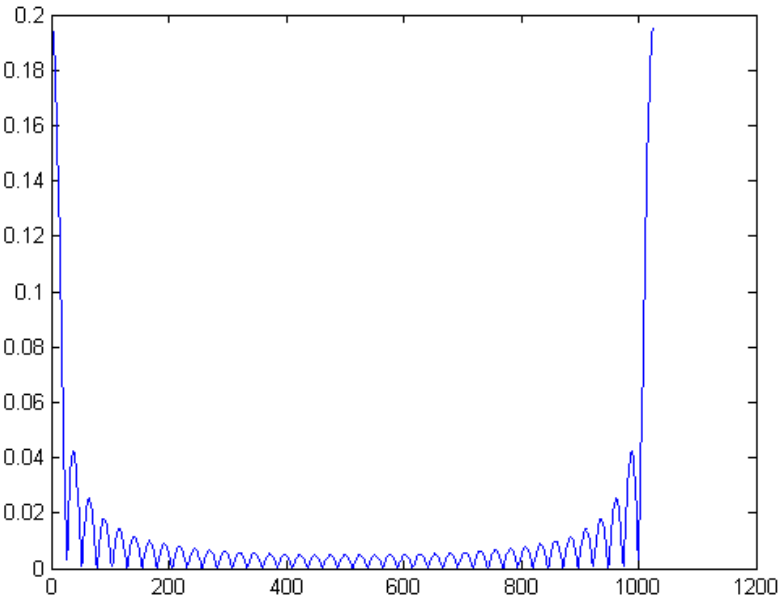
3.5 Прямоугольный импульсный сигнал

3.5.1 Код Matlab

```
t=-0.04:1/1000:0.04;
y4=-5*rectpuls(t,0.04);
plot(t,y4);
figure;
spectrum = abs(fft(y4,1024))/1024;
plot(spectrum);
```

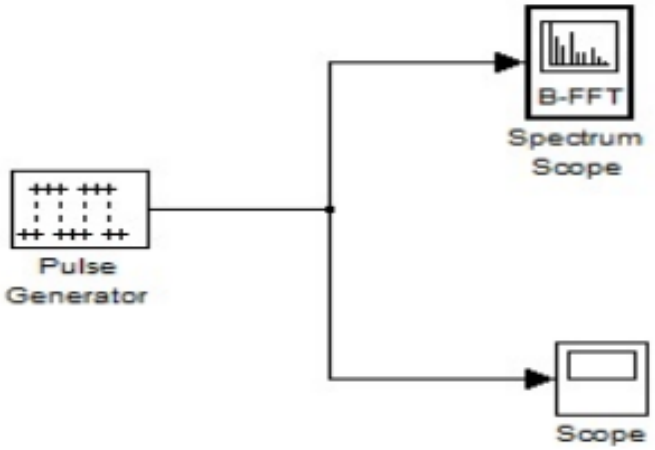
- Прямоугольный импульсный сигнал
- Спектр прямоугольного сигнала


0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
-4
-4.5
-5
-0.04
-0.03
-0.02
-0.01
0
0.01
0.02
0.03
0.04

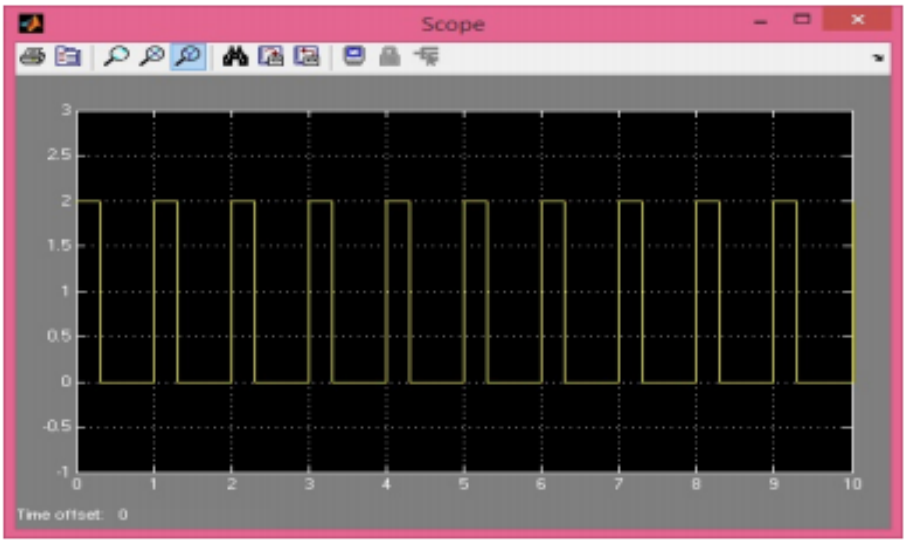

0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
200
400
600
800
1000
1200

3.5.2 Модуляция в среде Simulink

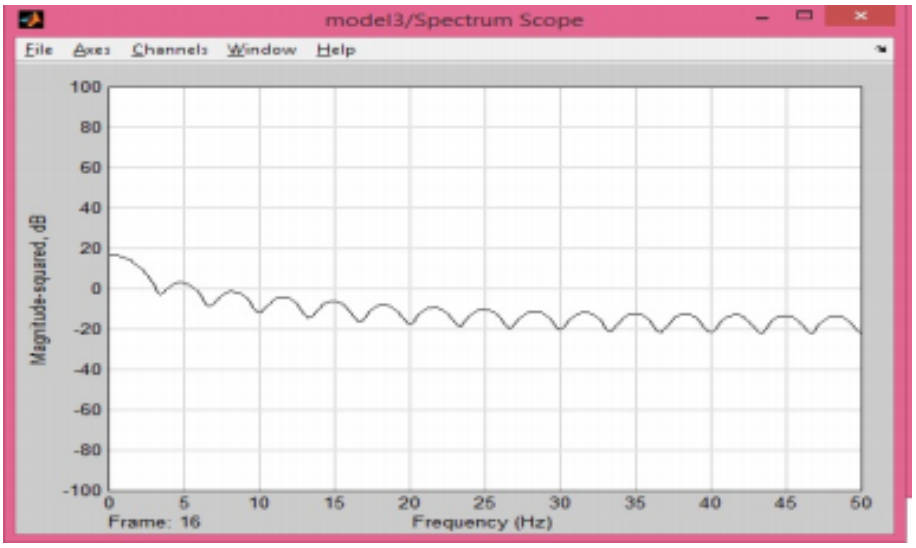
- модель



- сигнал



- спектр

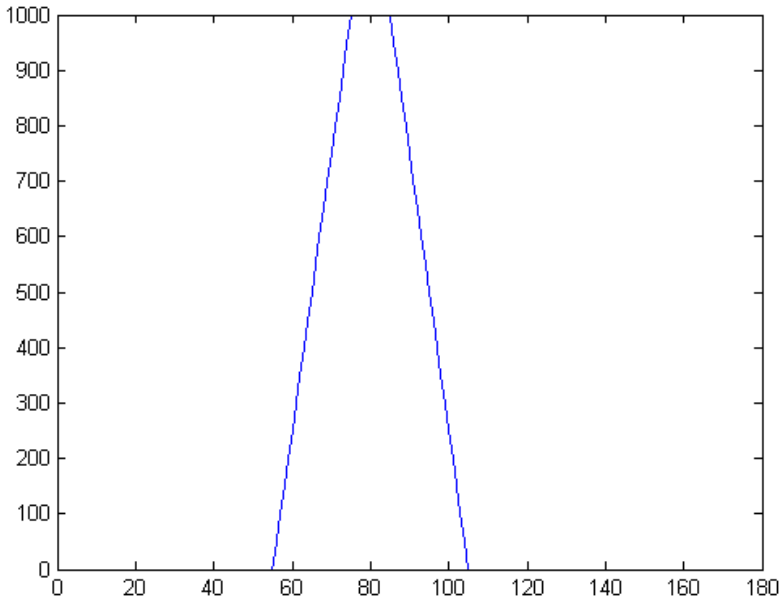


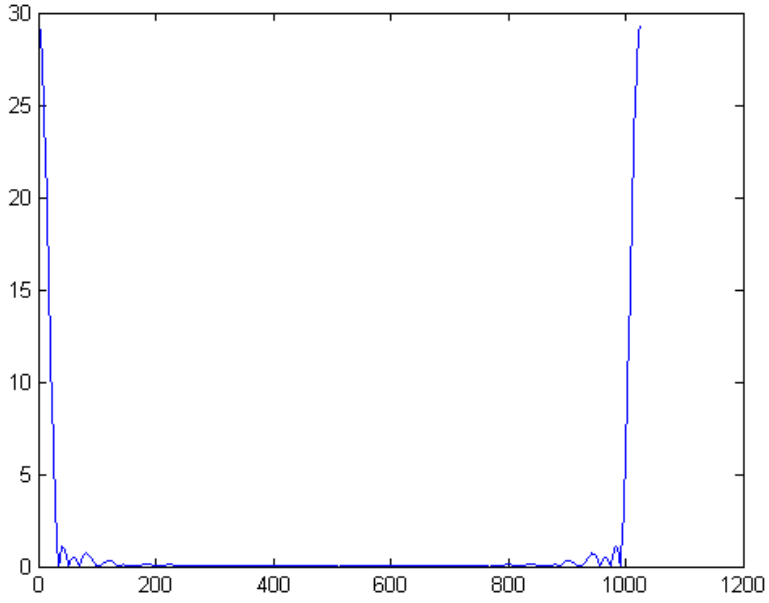
3.6 Трегольный импульсный сигнал

3.6.1 Код Matlab

```
t=-0.04:1/1000:0.04;
y4=-5*rectpuls(t,0.02);
y5=-10*rectpuls(t,0.03);
y6= conv(y4,y5);
plot(y6);
figure;
spectrum = abs(fft(y6,1024))/1024;
plot(spectrum);
```

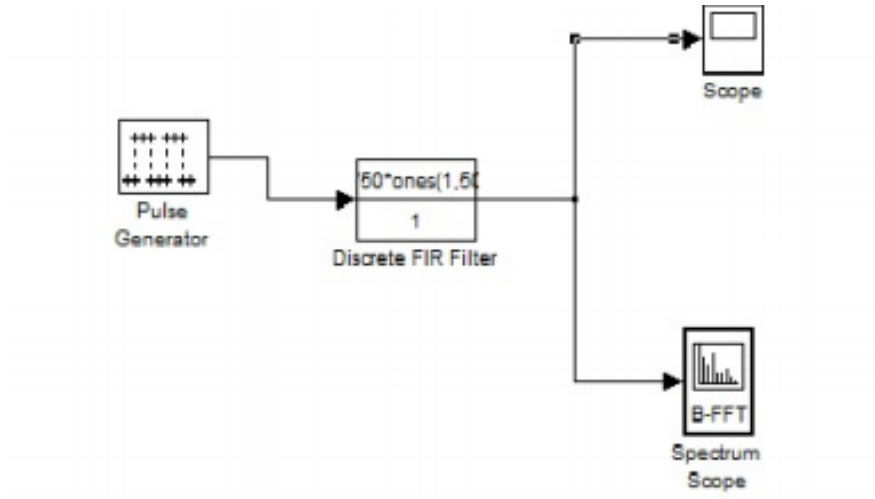
- Треугольный импульсный сигнал
- Спектр треугольного сигнала


1000
900
800
700
600
500
400
300
200
100
0
0
20
40
60
80
100
120
140
160
180

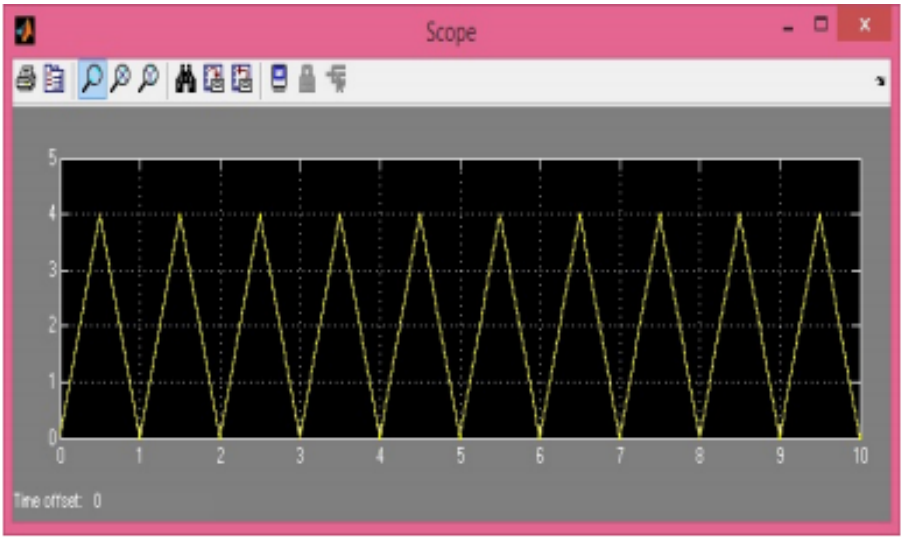

30
25
20
15
10
5
0
0
200
400
600
800
1000
1200

3.6.2 Модуляция в среде Simulink

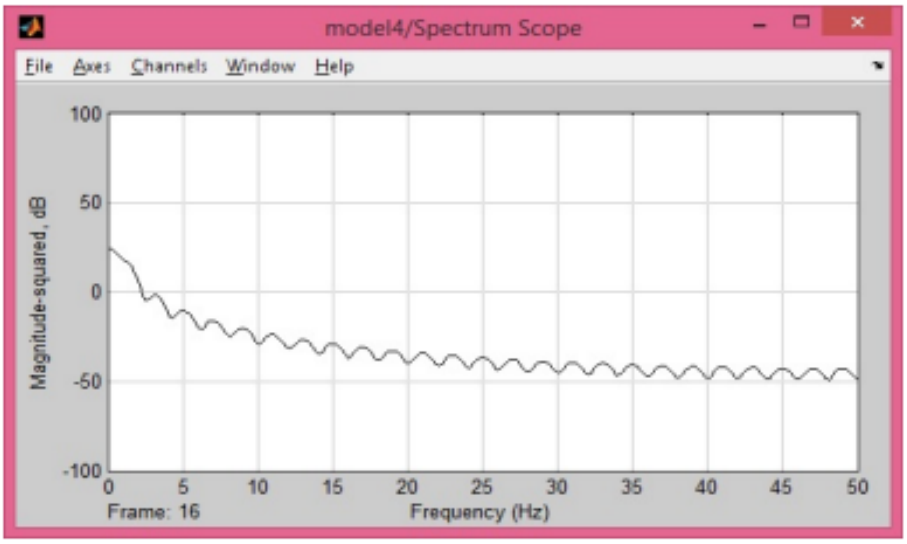
- модель



- сигнал



- спектр



3.7 Вывод

В лабораторной работе было проведено моделирование в среде MatLAB и Simulink полигармонического, прямоугольного и треугольного сигналаов. После чего получены их спектры. Для получения сигналов использовались математические формулы данных функций. Треугольный сигнал был получен путем применения операции свертки для двух прямоугольных сигналов. Данная операция осуществляется с помощью специальной функции Matlab. В Simulink для получения генератора треугольного сигнала используется генератор прямоугольных импульсов каскадно с фильтром с прямоугольным окном.
Обоснованием получения треугольного сигнала из свертки двух прямоугольных служит тот факт, что интеграл от произведения двух констант есть линейная функция. График такого преобразования будет представлять две линеные функции, одна из которых имеет положительный коэффициент наклона, а другая отрицательный.

3.8 Формулы

Ниже приведены формулы для вычисления спектров исходных импульсов. Так как треугольный импульс можно представить в виде свертки двух прямоугольных, спектр треугольного испульса выражается через квадрат спектра прямоугольного импульса.

$$S(t) = \sum_{k=0}^{K} (a_k * cos(2 * pi * k * f * t) + b_k * sin(2 * pi * k * f * t))$$

$$\Phi(f) = \int_{-\infty}^{\infty} \Pi(t,\ T_и)\, e^{-j2\pi ft}\, dt = \int_{-T_и/2}^{T_и/2} e^{-j2\pi ft}\, dt =$$

$$= \frac{\sin(\pi f T_и)}{\pi f}.$$

Рис. 1: Спектр прямоугольного импульса

$$\Delta(t,\ T_и) \rightleftarrows \frac{2}{T_и}\left[\frac{T_и}{2}\,\mathrm{sinc}\left(\pi f\,\frac{T_и}{2}\right)\right]^2 = \frac{\sin^2(\pi f T_и/2)}{\pi^2 f^2 T_и/2}$$

Рис. 2: Спектр треугольного импульса

4 Линейная фильтрация

4.1 Цель работы

Изучить воздействия ФНЧ на тестовый сигнал с шумом. Сгенерировать гармонический сигнал с шумом и синтезировать ФНЧ. Получить сигнал во временной и частотной областях до и после фильтрации.

4.2 Ход работы

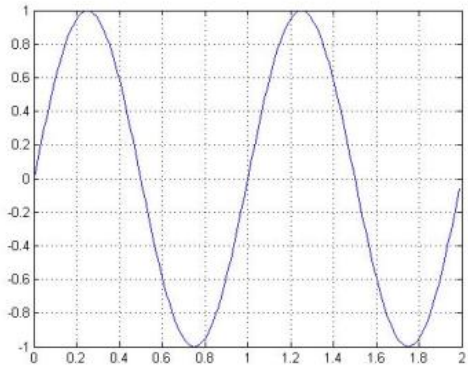
Линейный фильтр — динамическая система, применяющая некий линейный оператор ко входному сигналу для выделения или подавления определённых частот сигнала и других функций по обработке входного сигнала. Фильтр Баттерворта — один из типов электронных фильтров. Фильтры этого класса отличаются от других методом проектирования. Фильтр Баттерворта проектируется так, чтобы его амплитудно-частотная характеристика была максимально гладкой на частотах полосы пропускания.

В командном окне Matlab сгенерируем гармонический сигнал без шума и с шумом, а также спектры сигнала с шумом и без шума. В результате код Matlaba будет выглядеть так:

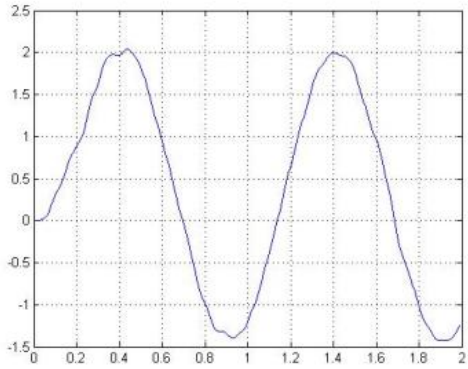
```
x = 0:0.01:4*pi;
f=100*(0:255)/512;
figure
noise=rand(size(x));
y = sin(2*pi*x);
y_noisy = y+0.3*noise;

%Построение сигнала без шума:
plot(x(1:200),y(1:200))
grid

%Построение сигнала с шумом:
```

```
plot(x(1:200),y_noisy(1:200))
grid

%синтез ФНЧ Баттерворта
[B,A] = butter(16,0.99);
B=B./sum(B);
A=A./sum(A);
%обработка сигнала ФНЧ
figure
y_filtered = conv(y_noisy,[B,A]);

%Построение сигнала после прохождения через фильтр:
plot(x(1:200),y_filtered(1:200))
grid
figure
%Построение спектра сигнала с шумом
noisy_spectrum = fft(y_noisy,512);
norm_noisy_spectrum = noisy_spectrum.*conj(noisy_spectrum)/512;

%Построение нормирова спектра сигнала с шумом:
plot(f,norm_noisy_spectrum(1:256))
axis([0 max(f) 0 2])
grid
figure

%Спектр отфильтрованного сигнала
spectrum = fft(y_filtered,512);
norm_filtered_spectrum=spectrum.*conj(spectrum)/512;
%Построение нормированого отфильтрованного спектра:
plot(f,norm_filtered_spectrum(1:256))
axis([0 max(f) 0 2])
grid
```

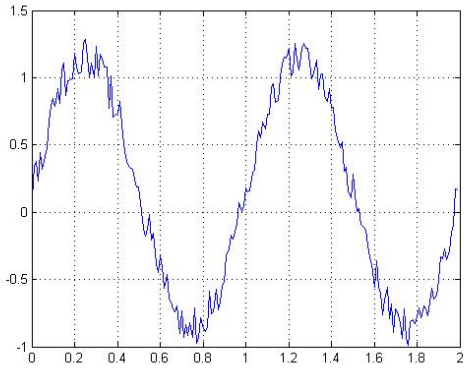
Результаты работы программы на рисунке.



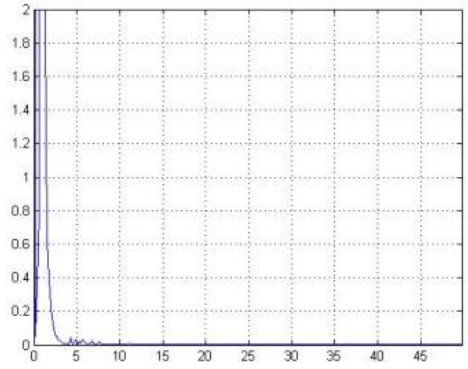
а)



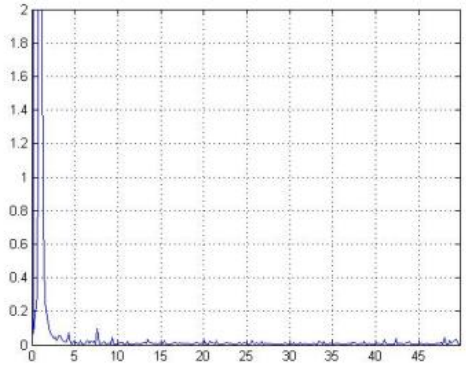
б)



в)



г)



д)

Рис. 3: Линейная фильтрация в Matlab: а) Сигнал без шума, б) Сигнал после фильтрации, в)Сигнал с шумом, г)Спектр сигнала после фильтрации, д)Спектр сигнала с шумом.

4.3 Моделирование в Simulink

Для создания модели в Simulink используем блок Discrete FIR Filter раздела Discrete главной библиотеки и блок Digital Filter design из Signal Processing Blockset/Filtering/Filter Designs. Результаты моделирование в Simulink на рисунке.

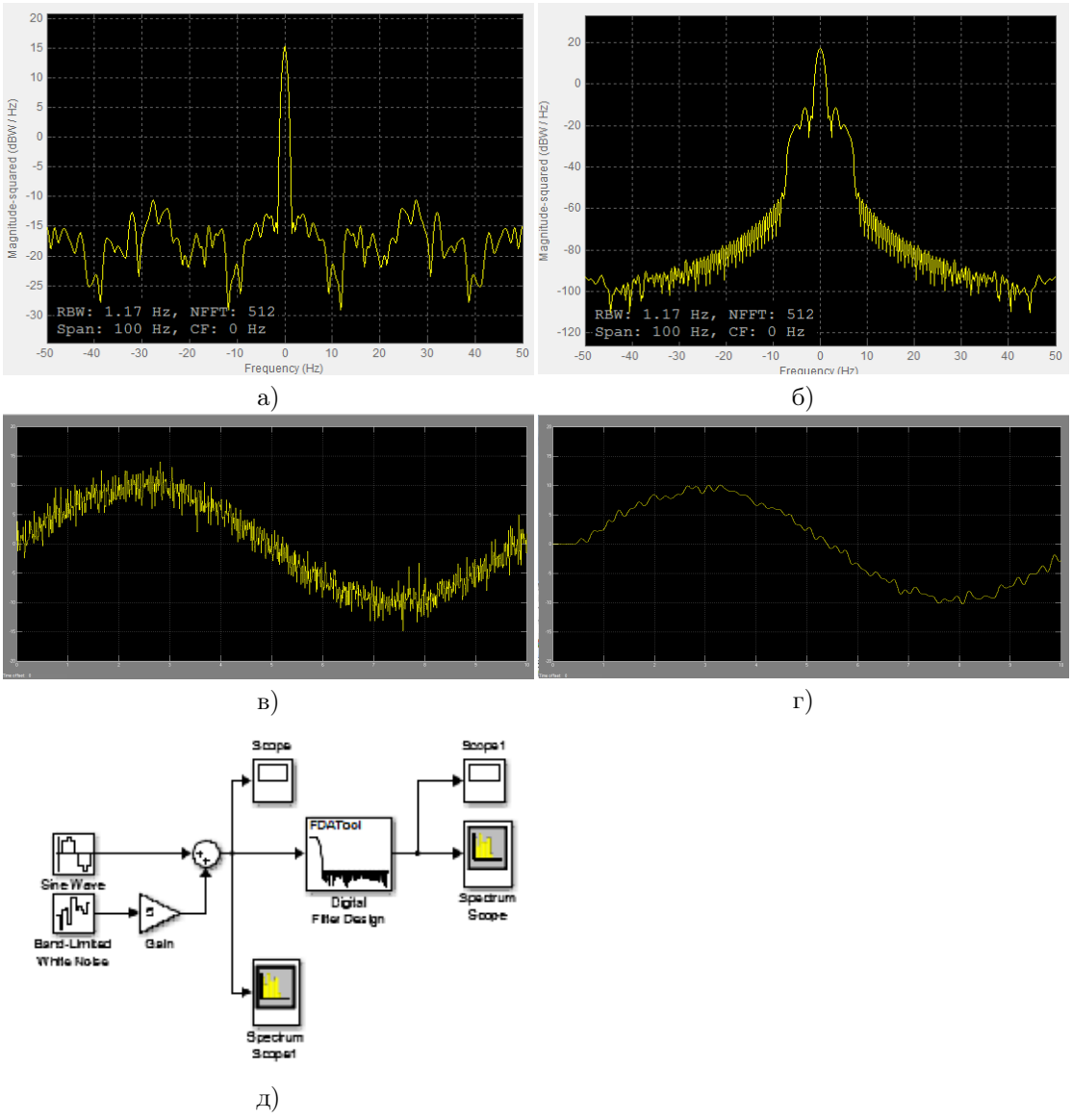


Рис. 4: Линейная фильтрация в Simulink: а) Спектр до фильтрации, б) Спектр после фильтрации, в)Сигнал до фильтрации, г)Сигнал после фильтрации, д) Схема модели.

4.4 Вывод

Фильтр нижних частот — один из видов аналоговых или электронных фильтров, эффективно пропускающий частотный спектр сигнала ниже некоторой частоты (частоты среза), и уменьшающий (подавляющий) частоты сигнала выше этой частоты. Степень подавления каждой частоты зависит от вида фильтра. В лабораторной работе мы произвели неполное удаление шума линейным фильтром. Для его полного удаления нам необходим идеальный фильтр (с прямоугольным окном). Линейная фильтрация не устраняет полностью шум, т.к. для полного удаления необходим идеальный фильтр с прямоугольным окном, которого на практике не существует. Так же, т. к. линейный фильтр подавляет все сигналы, находящиеся в его полосе задержания и не изменяет сигналы из полосы пропускания, то он не может удалить шум, который попал в его полосу пропускания. Соответственно удалить весь шум он не в силах.

5 Аналоговая модуляция

5.1 Цель работы

Изучить процесс амплитудной модуляции/демодуляции сигнала.

5.2 Алгоритм работы

- Сгенерировать однотональный сигнал низкой частоты.
- Выполнить амплитудную модуляцию (АМ) сигнала по закону

  $$u(t) = (1 + M * U_m * cos(\Omega * t)) * cos(\omega_0 * t + \phi_0)$$

  для различных значений глубины модуляции M.
- Получить спектр модулированного сигнала.
- Выполнить модуляцию с подавлением несущей:

  $$u(t) = MU_m cos(\omega t) cos(\omega_0 t + \phi_0). \quad (4)$$

  Получить спектр.
- Выполнить однополосную модуляцию:

  $$u(t) = U_m cos(\omega t) cos(\omega_0 t + \phi_0) + \frac{U_m}{2} \sum_{n=1}^{N} M_n (cos(\omega_0 + \Omega_n)t + \phi_0 + \Phi_n), \quad (5)$$

  положив n = 1.
- Выполить синхронное детектирование и получить исходный однополосный сигнал.
- Рассчитать КПД модуляции:

  $$\eta_A M = \frac{U_m^2 M^2 / 4}{P_U} = \frac{M^2}{M^2 + 2}. \quad (6)$$

5.3 Теоретические сведения

Перенос спектра сигналов из низкочастотной области в выделенную для их передачи область высоких частот выполняется операцией модуляции. АМ соответствует переносу информации s(t)=>U(t) при постоянных значениях параметров несущей частоты. АМ – сигнал представляет собой произведение информационной огибающей U(t) и гармонического колебания ее заполнения с более высокими частотами. Форма записи амплитудно-модулированного сигнала:

$$u(t) = (1 + M * U_m * cos(\Omega * t)) * cos(\omega_0 * t + \phi_0)$$

, где U – постоянная амплитуда несущего колебания при отсутствии входного (модулирующего) сигнала s(t), M – коэффициент амплитудной модуляции. Значение M характеризует глубину амплитудной модуляции. В зависимости от значения M различают нормальную модуляцию (M<1), глубокую модуляцию (M=1) и перемодуляцию (M>1). КПД амплитудной модуляции равен

$$\eta_A * M = \frac{M^2}{M^2 + 2}$$

Как видно, основная доля мощности АМ – сигнала приходится на несущую частоту. При балансной модуляции (с подавлением несущей) производится перемножение двух сигналов – модулирующего и несущего, при котором происходит подавление несущего колебания, соответственно, КПД модуляции становится равным 100

$$u(t) = M * cos(\Omega * t))$$

имеем

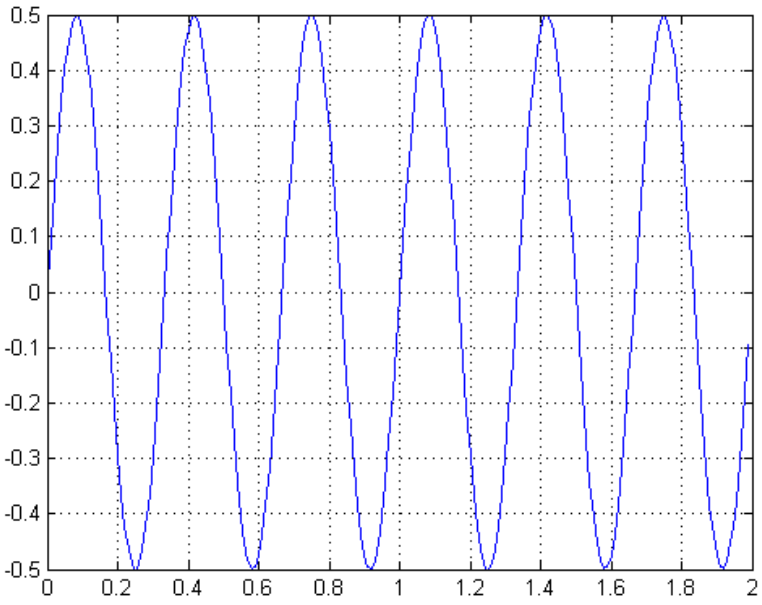
$$u(t) = M*U_m*cos(\Omega*t)*cos(\omega_0*t) = U_m*\frac{M}{2}*(cos((\omega_0+\Omega)*t)+cos((\omega_0-\Omega)*t)))$$

, т.е. два одинаковых по амплитуде гармонических сигнала с верхней и нижней боковыми частотами. По существу, однотональный модулирующий сигнал переносится на две высокие частоты.

5.4 Код MATLAB для п.1-3

```
f0=3; частота сигнала
fd=150; частота дискретизации
fc=30; частота несущего колебания
x=0:0.01:4*pi;
y=0.5*sin(2*pi*f0*x);
plot(x(1:200),y(1:200))
grid;
figure
M1=0.3;
M2=1;
M3=1.3;
t=0:0.001:10;
Um=0.5;
```

```
f=100*(0:255)/512;
u1=(1+Um*M1*cos(f0*t)).*cos(fc*t); модулированный сигнал
plot(t,u1)
figure
s1=fft(u1,512); спектр модулированного сигнала
ss1=s1.*conj(s1)/512;
plot(f,ss1(1:256));
figure
u2=Um*(1+M2*cos(f0*t)).*cos(fc*t);
plot(t,u2)
figure
s2=fft(u2,512);
ss2=s2.*conj(s2)/512;
plot(f,ss2(1:256));
figure
u3=Um*(1+M3*cos(f0*t)).*cos(fc*t);
plot(t,u3)
figure
s3=fft(u3,512);
ss3=s3.*conj(s3)/512;
plot(f,ss3(1:256));
```



Исходный низкочастотный сигнал

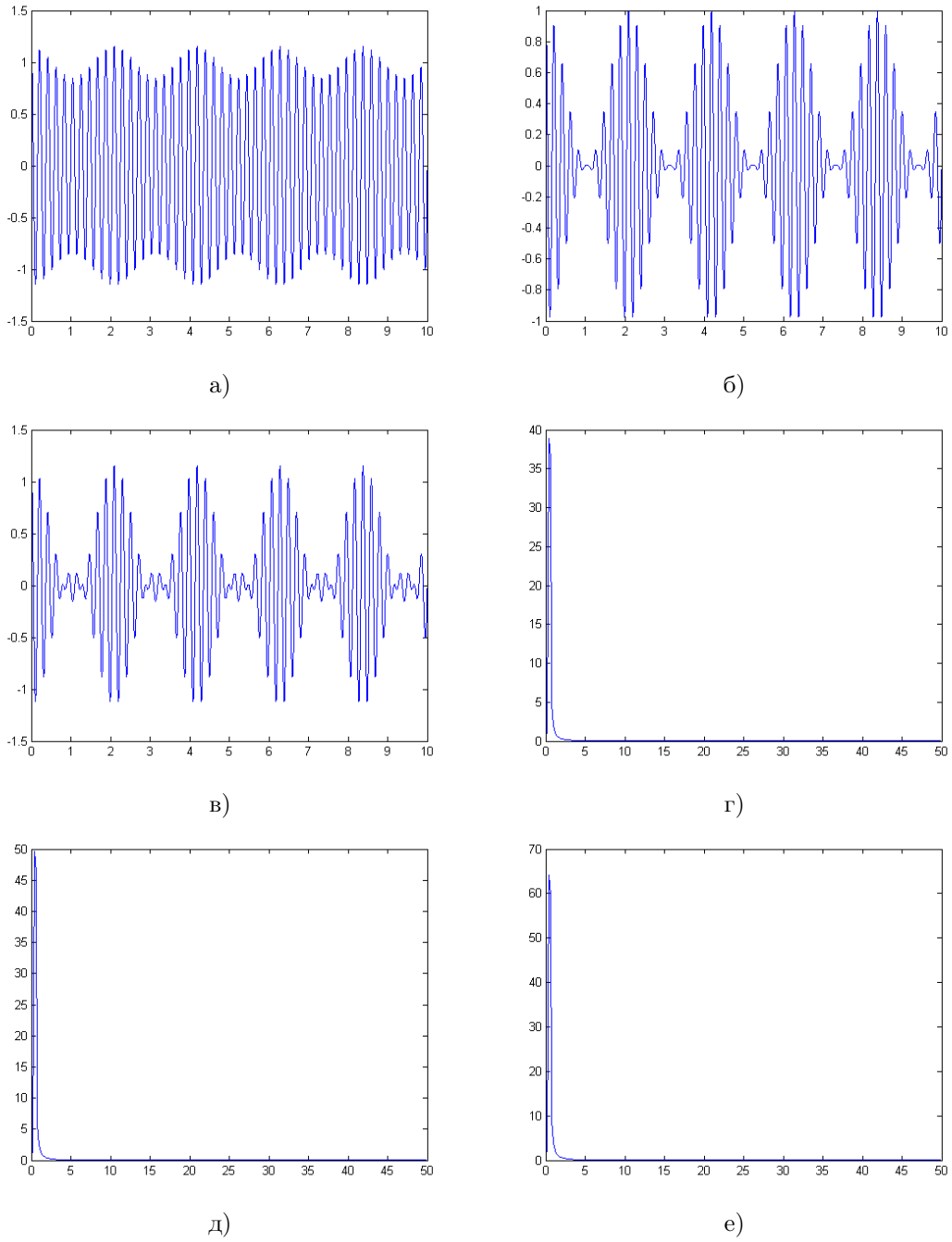
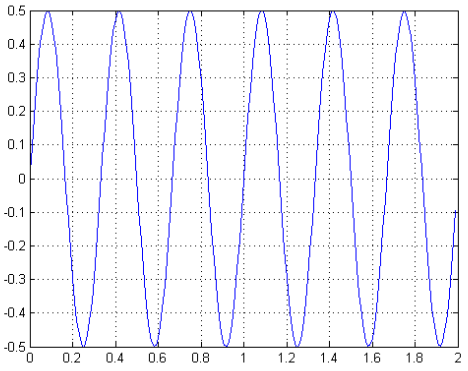


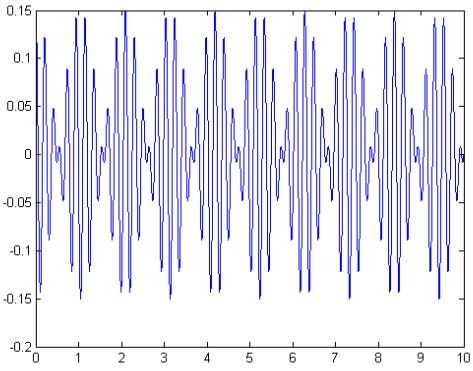
Рис. 5: Линейная фильтрация в Simulink: а) АМ при степени глубины=0.3, б)АМ при степени глубины=1, в)АМ при степени глубины=1.3, г) Спектр модулированного сигнала при степени глубины=0.3, д) Спектр модулированного сигнала при степени глубины=1, е) Спектр модулированного сигнала при степени глубины=1.3.

5.5 Код MATLAB для п.4

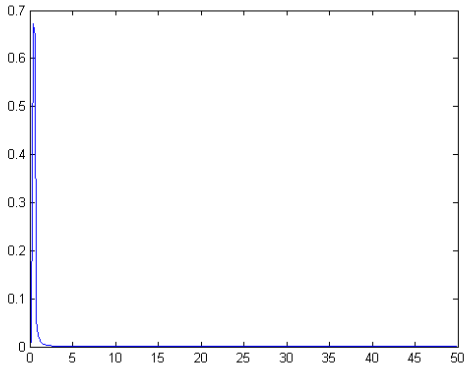
```
plot(x(1:200),y(1:200)) исходный сигнал
grid;
figure
u4=Um*M1*cos(f0*t).*cos(fc*t); модуляция с подавлением несущей
plot(t,u4)
figure
s4=fft(u4,512);
ss4=s4.*conj(s4)/512;
plot(f,ss4(1:256)); figure
plot(t,YY);
```



а)



б)

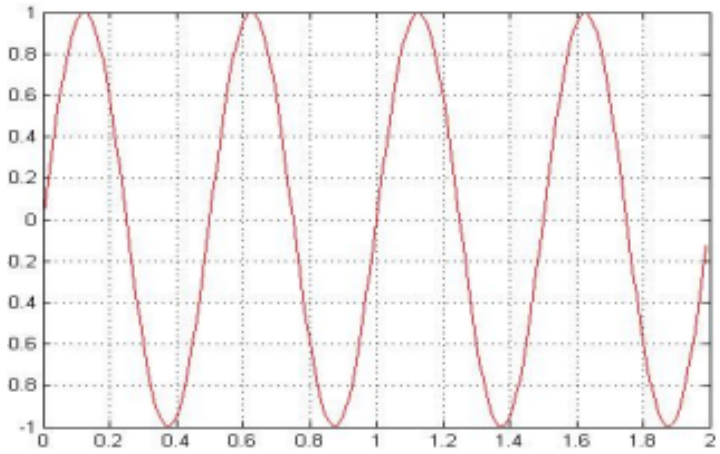


в)

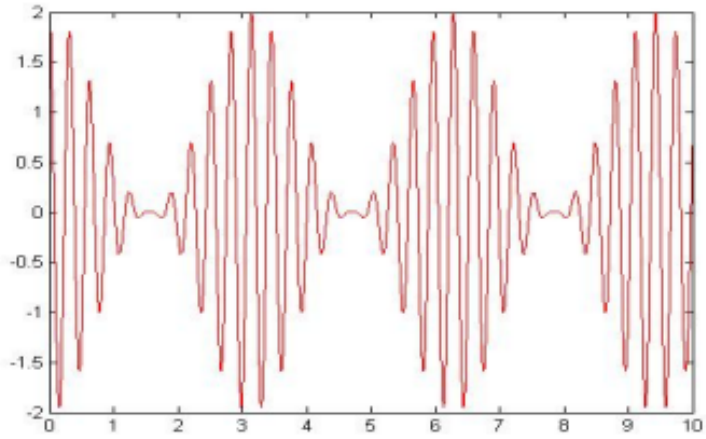
Рис. 6: Линейная фильтрация в Simulink: а) Исходный низкочастотный сигнал, б)Модулированный сигнал с подавлением несущей, в) Спектр модулированного сигнала с подавлением несущей.

5.6 Код MATLAB для п.5

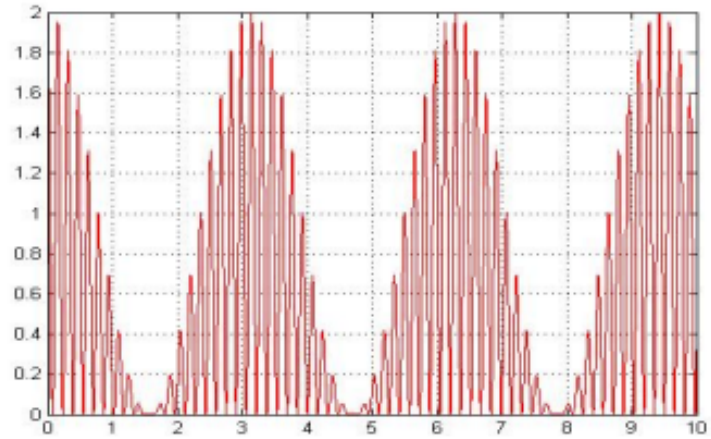
```
x=0:0.01:4*pi;
y=sin(2*pi*2*x);
plot(x(1:200),y(1:200),'r')
t=0:0.001:10;
u2=(1+1*cos(2*t)).*cos(20*t);
s2=fft(u2,512);
yy=u2.*cos(20*t);
plot(t,yy,'r');
ss=fft(yy,512);
sss=ss.*conj(ss)/512;
plot(f,sss(1:256),'r');
```



а)



б)



б)

Рис. 7: Линейная фильтрация в Simulink: а) Исходный сигнал, б)Модулированный сигнал, в)Сигнал после выполнения синхронного детектирования.

5.7 Код MATLAB для п.6

Найдем КПД модуляции по формуле

$$\eta_A * M = \frac{M^2}{M^2 + 2}$$

M=0.3; КПД=0.14
M=1; КПД=0.3(3)
M=1.3; КПД=0.35

5.8 Вывод

Для однотональной модуляции начальная фаза модулирующего колебания для верхней боковой частоты складывается с начальной фазой несущей, для нижней – вычитаются из фазы несущей. При балансной модуляции производится перемножение двух сигналов – модулирующего и несущего, при котором происходит подавление несущего колебания, соответственно, КПД модуляции становится равным 100

6 Частотная и фазовая модуляция

6.1 Цель работы

Изучение частотной и фазовой модуляции/демодуляции сигнала.

6.2 Постановка задачи

1. Сгенерировать однотоальный сигнал низкой частоты.
2. Выполнить фазовую модуляцию/демодуляцию сигнала по закону

$$u(t) = U_m cos(\Omega t + ks(t)), \tag{7}$$

   используюя встроенную функцию MatLab pmmod, pmdemod.
3. Получить спектр модулированного сигнала.
4. Выполнить частотню модуляцию/демодуляцию по закону

$$u(t) = U_m cos(\omega_0 t + k \int_0^t s(t)dt + \phi_0), \tag{8}$$

   используюя встроенные функции MatLab fmmod, fmdemod.

6.3 Справочные материалы

Н.В. Богач и др. Обработка сигналов в информационных системах, с. 118-125, 127-133.

6.4 Ход работы

Фазовая модуляция – процесс изменения мгновенной фазы несущего колебания пропорционально изменению непрерывного информационного сигнала. Фазомодулированный сигнал s(t) имеет следующий вид:

$$s(t) = g(t)\sin[2\pi f_c t + \varphi(t)], \tag{9}$$

где g(t) — огибающая сигнала; $\varphi(t)$ является модулирующим сигналом; f_c — частота несущего сигнала; t — время. В случае, когда информационный сигнал является дискретным, то говорят о фазовой манипуляции.

По характеристикам фазовая модуляция близка к частотной модуляции. В случае синусоидального модулирующего (информационного) сигнала, результаты частотной и фазовой модуляции совпадают.

Реализация фазовой модуляции/демодуляции с помощью MATLAB:

```
% Исходный сигнал
x = 0:0.01:4*pi;
y = sin(2*pi*x);

figure
subplot(4, 1, 1);
plot(x(1:500), y(1:500))
grid
title('Исходный сигнал')

% Фазовоя модуляция
Fs = 1000;                        % Частота дискретизации
Fc = 4;                           % Несущая частота
phasedev = pi/2;                  % Девиация фазы для фазовой модуляции
pm_y = pmmod(x,Fc,Fs,phasedev); % Фазовая модуляция
subplot(4, 1, 2);
plot(x(1:500), pm_y(1:500))
grid
title('Фазовая модуляция')

spectrum = fft(pm_y, 512);      % Спектр фазовой модуляции
norm_spectrum = spectrum.*conj(spectrum)/512;
f = 100*(0:255)/512;
subplot(4, 1, 3);
plot(f, norm_spectrum(1:256))
grid
title('Спектр фазовой модуляции')
axis([min(f) max(f) 0 max(norm_spectrum)]);

pdm_y = pmdemod(pm_y,Fc,Fs,phasedev); % Демодуляция
subplot(4, 1, 4);
plot(x, pdm_y)
grid
title('Фазовая демодуляция')
```

В результате выполнения приведенного кода были получены следующие характеристики:

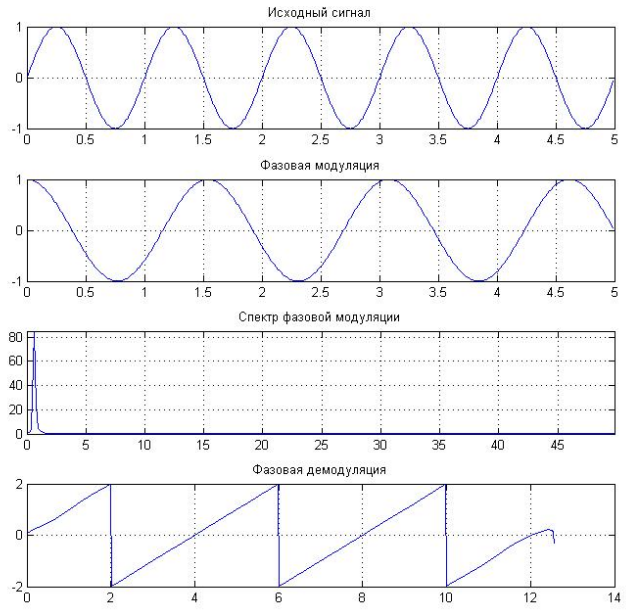


Рис. 8: Результаты фазовой модуляции/демодуляции

Частотная модуляция – процесс изменения мгновенной частоты несущего колебания в соответствии с изменением информационного сигнала.По сравнению с амплитудной модуляцией здесь амплитуда остаётся постоянной.

Реализация частотной модуляции/демодуляции с помощью MATLAB:

```
% Исходный сигнал
x = 0:0.01:4*pi;
y = sin(2*pi*x);

figure
subplot(4, 1, 1);
plot(x(1:500), y(1:500))
grid
title('Исходный сигнал')

% Частотная модуляция
freqdev = 20;                  % Девиация частоты для частотной модуляции
fm_y = fmmod(x,Fc,Fs,freqdev);  % Частотная модуляция
subplot(4, 1, 2);
plot(x(1:500), fm_y(1:500))
grid
```

```
title('Частотная модуляция')
spectrum = fft(fm_y, 512);      % Спектр частотной модуляции
norm_spectrum = spectrum.*conj(spectrum)/512;
f = 100*(0:255)/512;
subplot(4, 1, 3);
plot(f, norm_spectrum(1:256))
grid
title('Спектр частотной модуляции')
axis([min(f) max(f) 0 max(norm_spectrum)]);
fdm_y = fmdemod(fm_y,Fc,Fs,freqdev);  % Демодуляция
subplot(4, 1, 4);
plot(x, fdm_y)
grid
title('Частотная демодуляция')
```

В результате выполнения приведенного кода были получены следующие характеристики:

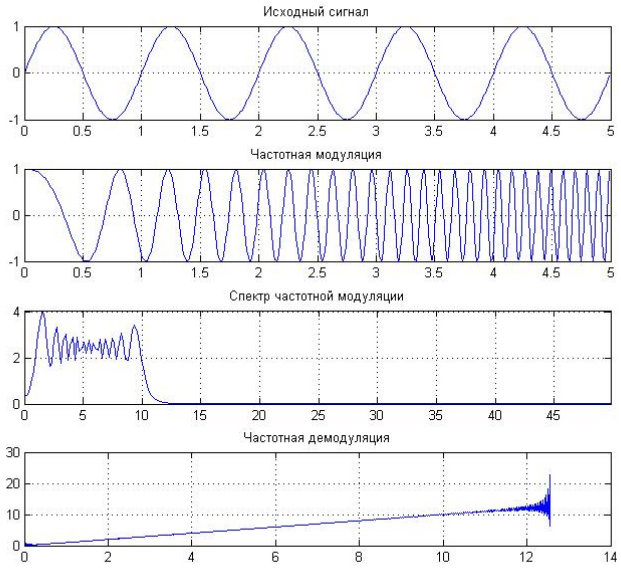


Рис. 9: Результаты частотной модуляции/демодуляции

Для создания содели в Simulinc использовался раздел Communication Blockset Simulink. Выполнена модуляция, в том числе с помощью блока захваты фазы Phase-Locked Loop.

Для моделирования использоовался исходный сигнал, представленный на Рис. 51.

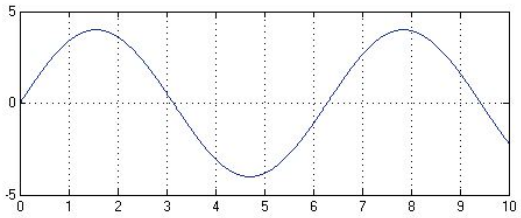


Рис. 10: Исходный сигнал

Фазовая модуляция/демодуляция в Simulink:

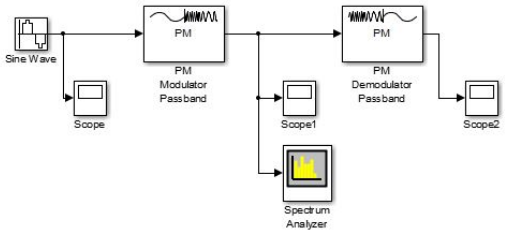


Рис. 11: Фазовая модуляция/демодуляция

В результате были получены следующие характеристики:

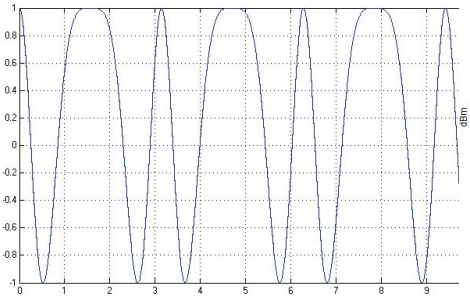


Рис. 12: ФМ-сигнал

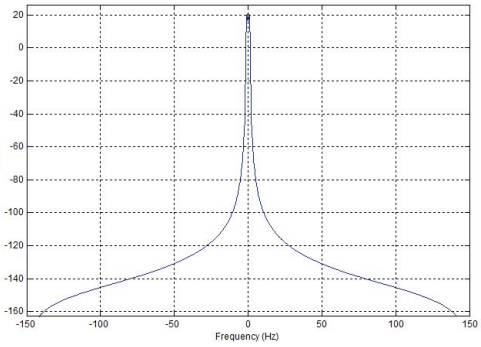


Рис. 13: Спектр ФМ-сигнала

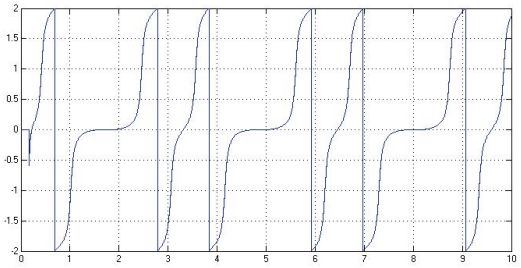


Рис. 14: Думодулированный ФМ-сигнал

6.5 Выводы

Частотная модуляция/демодуляция в Simulink:

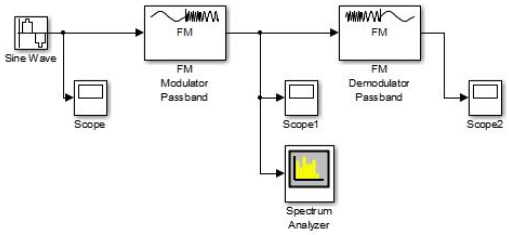


Рис. 15: Частотная модуляция/демодуляция

В результате были получены следующие характеристики:

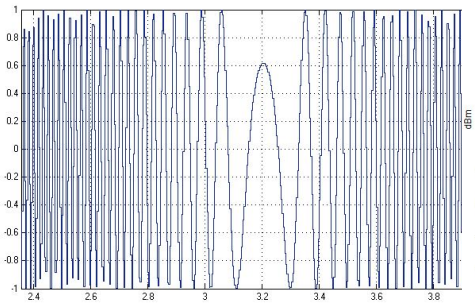


Рис. 16: ЧМ-сигнал

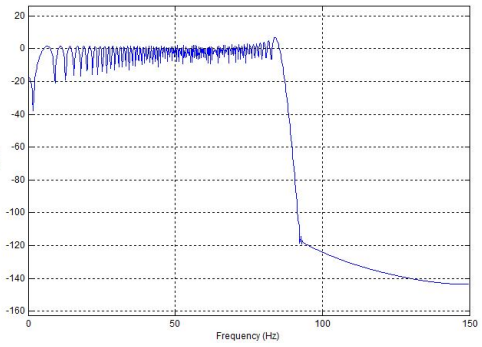


Рис. 17: Спектр ЧМ-сигнала

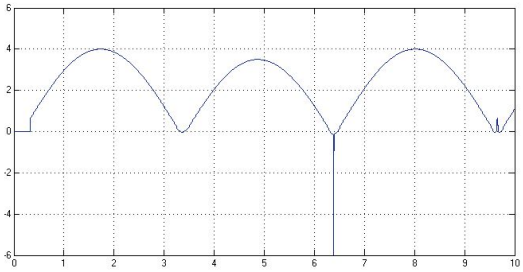


Рис. 18: Думодулированный ЧМ-сигнал

Фазовая автоподстройка частоты — система автоматического регулирования, подстраивающая фазу управляемого генератора так, чтобы она была равна фазе опорного сигнала, либо отличалась на известную функцию от времени. Регулировка осуществляется благодаря наличию отрицательной обратной связи. Выходной сигнал управляемого генератора сравнивается на фазовом детекторе с опорным сигналом, результат сравнения используется для подстройки управляемого генератора.

Система ФАПЧ используется для частотной модуляции и демодуляции, умножения и преобразования частоты, частотной фильтрации, выделения опорного колебания для когерентного детектирования и в других целях.

ФАПЧ сравнивает фазы входного и опорного сигналов и выводит сигнал ошибки, соответствующий разности между этими фазами. Сигнал ошибки проходит далее через фильтр низких частот и используется в качестве управляющего для генератора, управляемого напряжением (ГУН), обеспечивающего отрицательную обратную связь. Если выходная частота отклоняется от опорной, то сигнал ошибки увеличивается, воздействуя на ГУН в сторону уменьшения ошибки. В состоянии равновесия выходной сигнал фиксируется на частоте опорного.

Также в схеме ФАПЧ включен фазовый детекто, которыйсравнивает фазы двух входных сигналов. Обычно, один из них генерируется генератором сигнала, управляемым напряжением, а второй берется из внешнего источника. ФД имеет два входа, управляющих стоящей за ним схемой подстройки частоты, задача которой сделать фазы сигналов одинаковыми.

Для ФАПЧ сигнал ошибки из ФД (значение найденной разности фаз) подается на сглаживающий фильтр (фильтр нижних частот). Сигнал с фильтра подается на управляемый напряжением генератор, частота(фаза) выходного сигнала которого зависит от напряжения на входе. Сигнал с генератора по цепи обратной связи поступает назад в детектор, замыкая контур ФАПЧ.

Частотная модуляция/демодуляция с фазовой автоподстройкой частоты в Simulink:

В результате были получены следующие характеристики:

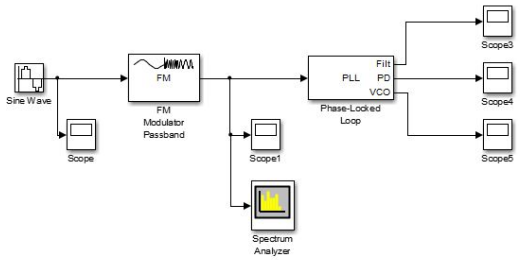


Рис. 19: Частотная модуляция/демодуляция

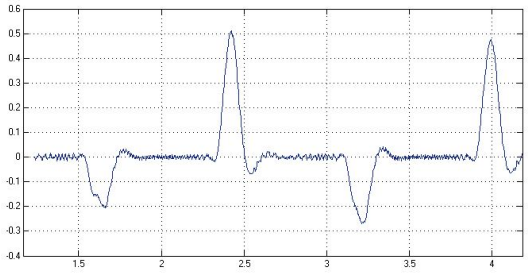


Рис. 20: Выход фльтра

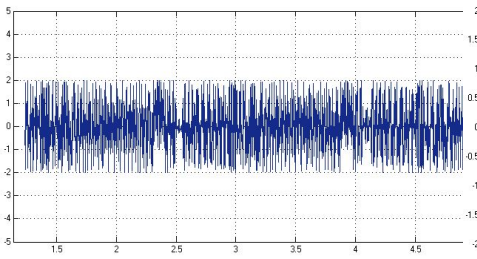


Рис. 21: Выход фазового детектора

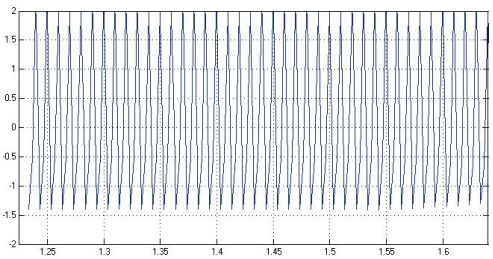


Рис. 22: Выход ГУН

6.6 Выводы

В результате выполнения данной работы были выполнены частотная и фазовая модуляция/демодуляция, а также чатотная демодуляция с помощью блока захвата фазы. Можно сделать вывод, что частотная и фазовая модуляция очень тесно взаимосвязаны, поскольку обе они влияют на аргумент функции cos. Поэтому эти два вида модуляции имеют общее название — угловая модуляция.Сигнал с угловой модуляцией имеет вид колебания, начальная фаза которого зависит от времени:

$$s_(t) = A_0 cos(\omega 0t + j(t)). \tag{10}$$

Различие между фазовой и частотной модуляцией заключается лишь в том, как именно начальная фаза $j(t)$ связана с модулирующим сигналом.

Для демодуляции использовалась петля ФАПЧ, состоящая из перемножителя (используемого в качестве фазового детектора), фильтра нижних частот и генератора, управляемого напряжением (ГУН). Получаемый на выходе петли ФАПЧ сигнал пропорционален отклонению мгновенной частоты модулированного сигнала от несущей частоты, поэтому при демодуляции ФМ этот сигнал можно дополнительно проинтегрировать, чтобы получить начальную фазу сигнала.

7 Цифровая модуляция

7.1 Цель работы

Изучение методов модуляции цифровых сигналов и сравнение их свойств.

7.2 Алгоритм работы

- Получить сигналы BPSK, PSK, OQPSK, genQAM, MSK, M-FSK модуляторов
- Построить их сигнальные созвездия.
- Провести сравнение изученных методов модуляции цифровых сигналов.

7.3 Теоретическая часть

В настоящее время все большая часть информации, передаваемой по разнообразным каналам связи, существует в цифровом виде. Это означает, что передаётся последовательность целых чисел, которые могут принимать значения из некоторого фиксированного конечного множества. Эти числа, называемые символами (symbol), поступают от источника информации с периодом T, а частота, соответствующая этому периоду, называется символьной скоростью (symbol rate): fT = 1/T. Последовательность передаваемых символов является дискретным сигналом. Поскольку символы принимают значения из конечного множества, этот сигнал является и квантованным, то есть его можно назвать цифровым сигналом. В данной работе рассматриваться вопросы, связанные с преобразованием этого цифрового сигнала в

аналоговый модулированный сигнал. Типичный подход при осуществлении передачи дискретной последовательности символов состоит в следующем. Каждому из возможных значений символа сопоставляется некоторый набор параметров несущего колебания. Эти параметры поддерживаются постоянными в течение интервала T, то есть до прихода следующего символа. Фактически это означает преобразование последовательности чисел nk в ступенчатый сигнал sn(t) : sn(t) = f(nk), kT <= t < (k + 1)T. Здесь f — некоторая функция преобразования. Полученный сигнал sn(t)далее используется в качестве модулирующего сигнала обычным способом. Такой способ модуляции, когда параметры несущего колебания меняются скачкообразно, называется манипуляцией (keying). В зависимости от того, какие именно параметры изменяются, различают амплитудную (АМн),фазовую (ФМн), частотную (ЧМн) и квадратурную (КАМн) манипуляцию. Цифровая модуляция и демодуляция включают в себя две стадии. При модуляции цифровое сообщение сначала преобразуется в аналоговый модулирующий сигнал с помощью функции modmap, а затем осуществляется аналоговая модуляция. При демодуляции сначала получается аналоговый демодулированный сигнал, а затем он преобразуется в цифровое сообщение с помощью функции demodmap. Функция randerr предназначена для формирования ошибок в цифровом сигнале. Она дает матрицу, в каждой строке которой имеется заданное число случайно расположенных ненулевых элементов. Для оценки помехоустойчивости системы связи необходимо произвести сравнение исходного (передаваемого) сообщения с сообщением, полученным в результате приема, и определить число ошибок, возникших в процессе передачи, а также вероятность ошибки. Эти действия выполняются функциями symerr и biterr, первая из которых подсчитывает число несовпадающих символах в двух сообщениях, а вторая — число несовпадающих битов в двоичных представлениях этих символов. Кроме числа ошибок, обе функции могут возвращать долю ошибок в общем числе символов (битов) и индикаторы мест возникновения ошибок. Последние две функции данной группы предназначены для графического отображения сигналов с квадратурной манипуляцией. Функция eyediagram выводит так называемую глазковую диаграмму, а функция scatterplot — диаграмму рассеяния. Аналоговый несущий сигнал модулируется цифровым битовым потоком. Существуют три фундаментальных типа цифровой модуляции (или шифтинга) и один гибридный:

1. ASK – Amplitude shift keying (Амплитудная двоичная модуляция).
2. FSK – Frequency shift keying (Частотая двоичная модуляция).
3. PSK – Phase shift keying (Фазовая двоичная модуляция).
4. ASK/PSK.

Одна из частных реализаций схемы ASK/PSK, которая называется QAM - Quadrature Amplitude Modulation (квадратурная амплитудная модуляция (КАМ). Это метод объединения двух АМ-сигналов в одном канале. Он позваляет удвоить эффективную пропускную способность. В QAM используется две несущих с одинаковой частотой но с разницей в фазе на четверть периода (отсюда и возникает слово квадратура). Частотная модуляция представляет логическую единицу интервалом с большей частотой, чем

ноль. Фазовый шифтинг представляет «0» как сигнал без сдвига, а «1» как сигнал со сдвигом.

BPSK : используется единственный сдвиг фазы между «0» и «1» — 180 градусов, половина периода. Существуют также QPSK: QPSK использует 4 различных сдвига фазы (по четверти периода) и может кодировать 2 бита в символе (01, 11, 00, 10).

7.4 Код MATLAB

7.4.1 BPSK modulation

```
h = modem.pskmod('M', 4);
g = modem.pskdemod('M', 4);
msg = randint(10,1,2);
modSignal = modulate(h,msg);
errSignal = (randerr(1,10, 3) ./ 30)';
modSignal = modSignal + errSignal;
demodSignal = demodulate(g,modSignal);
scatterplot(modSignal);
```

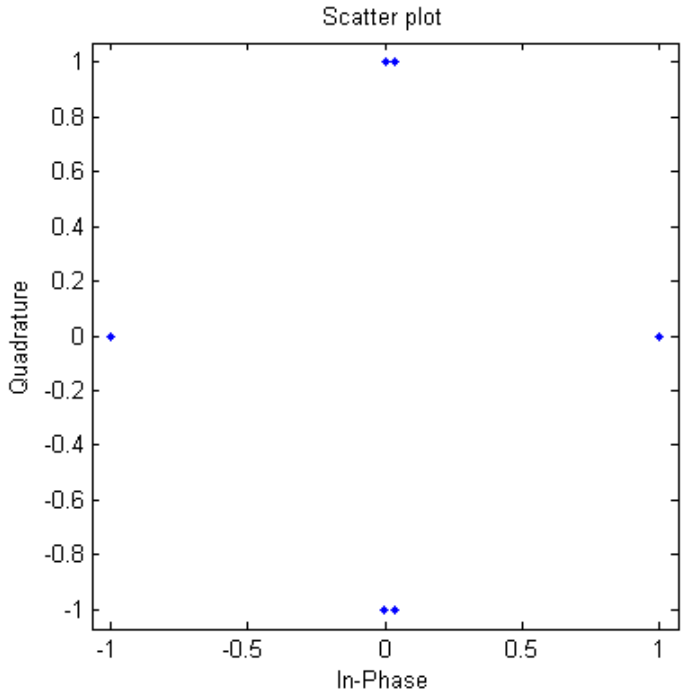


Рис. 23: Временная характеристика чистого синусоидального сигнала

7.4.2 PSK modulation

```
h = modem.pskmod('M', 8);
g = modem.pskdemod('M', 8);
msg = randint(10,1,8);
modSignal = modulate(h,msg);
errSignal = (randerr(1,10, 3) ./ 30)';
modSignal = modSignal + errSignal;
demodSignal = demodulate(g,modSignal);
scatterplot(modSignal);
```

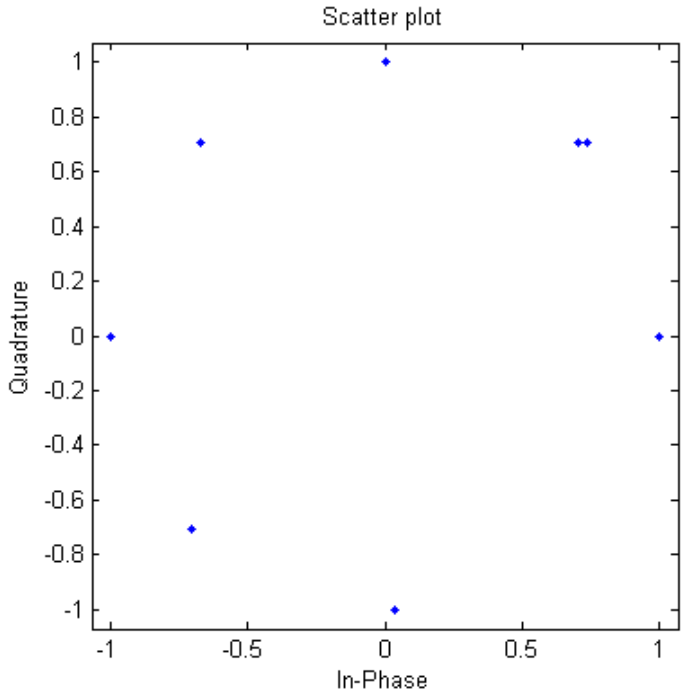


Рис. 24: Временная характеристика чистого синусоидального сигнала

7.4.3 OQPSK modulation

h = modem.oqpskmod;
g = modem.oqpskdemod;
msg = randint(200,1,4);
modSignal = modulate(h,msg);
errSignal = (randerr(1,400, 100) ./ 30)';
modSignal = modSignal + errSignal; 4
demodSignal = demodulate(g,modSignal);
scatterplot(modSignal);

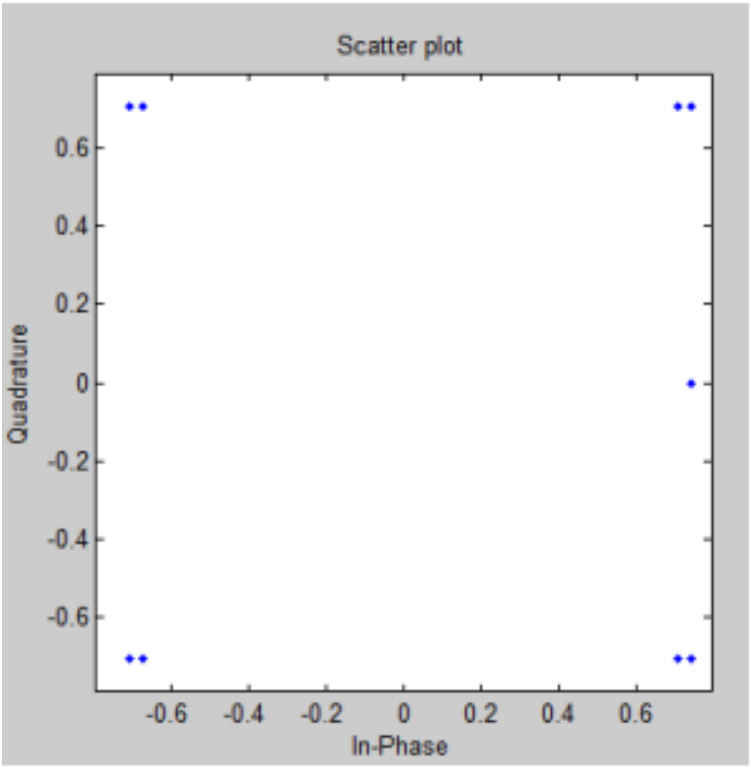


Рис. 25: Временная характеристика чистого синусоидального сигнала

7.4.4 GENQAM modulation

```
M = 8;
h = modem.genqammod('Constellation', exp(j*2*pi*[0:M-1]/M));
g = modem.genqamdemod('Constellation', exp(j*2*pi*[0:M-1]/M));
msg = randint(8,1,8);
modSignal = modulate(h,msg);
errSignal = (randerr(1,8, 3) ./ 30)';
modSignal = modSignal + errSignal;
demodSignal = demodulate(g,modSignal);
scatterplot(modSignal);
```

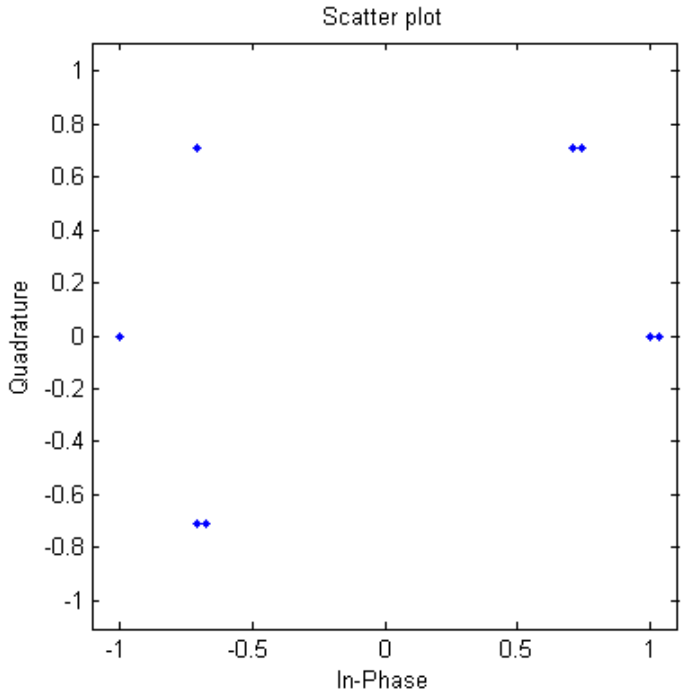


Рис. 26: Временная характеристика чистого синусоидального сигнала

7.4.5 MSK modulation

```
h = modem.mskmod('SamplesPerSymbol', 3);
g = modem.mskdemod('SamplesPerSymbol', 3);
msg = randint(3,1,2);
modSignal = modulate(h, msg);
errSignal = (randerr(1,9, 3) ./ 30)';
modSignal = modSignal + errSignal;
demodSignal = demodulate(g, modSignal);
scatterplot(modSignal);
```

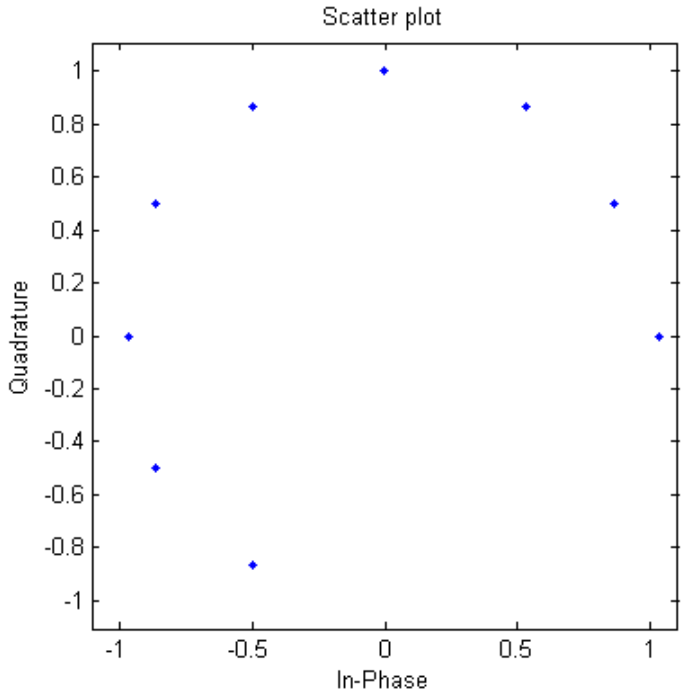


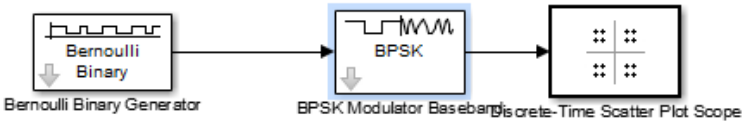
Рис. 27: Временная характеристика чистого синусоидального сигнала

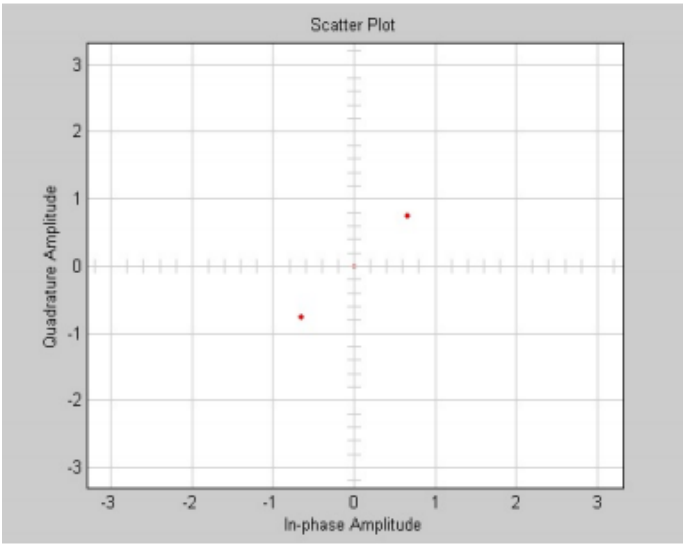
7.5 Работа в среде Simulink

7.5.1 BPSK modulation

- Получен сигнал BPSK и построено сигнальное созвездие в среде Simulink

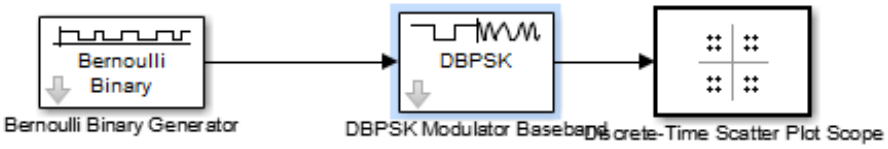
- Сигнальное созвездие



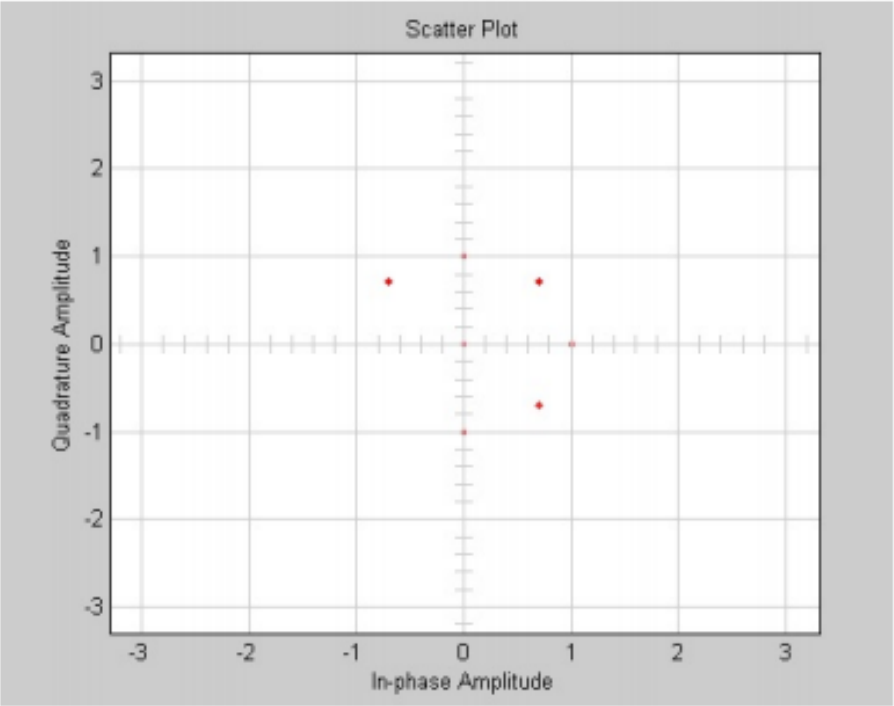


7.5.2 DBPSK modulation

- Получен сигнал DBPSK и построено сигнальное созвездие в среде Simulink

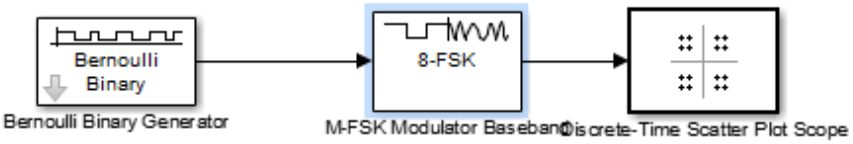


- Сигнальное созвездие



7.5.3 M-FSK modulation

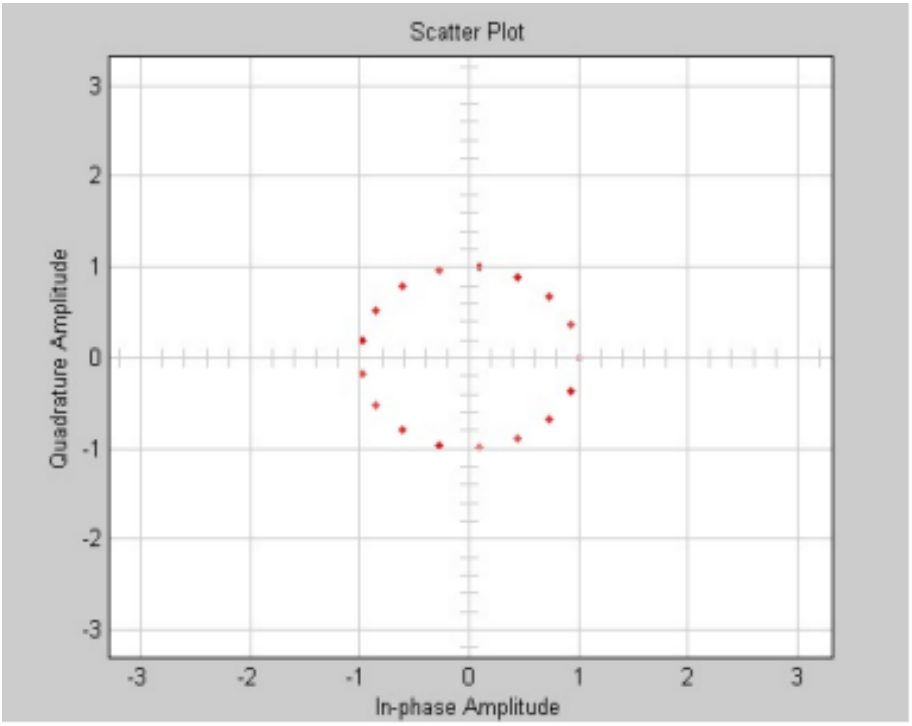
- Получен сигнал M-FSK и построено сигнальное созвездие в среде Simulink



- Сигнальное созвездие

7.6 Вывод

Уровень модуляции определяет количество состояний несущей, используемых для передачи информации. Чем выше этот уровень, тем большими скоростными возможностями и меньшей помехоустойчивостью модуляция обладает. Число бит, передаваемых одним состоянием, определяется как Log N, где N — уровень модуляции. Таким образом, чем выше уровень модуляции, тем больше данных мы можем передать (или потерять) за единицу


Scatter Plot
Quadrature Amplitude
In-phase Amplitude
3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3

времени.