Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

Е. В. Рогожников
С. М. Мухамадиев

ПРОСТРАНСТВЕННАЯ ОБРАБОТКА СИГНАЛОВ

Методические указания для выполнения
практических работ и самостоятельной работы

Томск
2024

УДК 621.396.2
ББК 32.84
Р 598

Рецензент:
Крюков Я.В., доцент кафедры телекоммуникаций и основ радиотехники ТУСУРа, кандидат технических наук

Р 598 Пространственная обработка сигналов: Методические указания для выполнения практических работ и самостоятельной работы / Е. В. Рогожников, С. М. Мухамадиев – Томск: Томск. гос. ун-т систем управления и радиоэлектроники, 2024. – 41 с.

Настоящие учебно-методическое пособие содержит указания по выполнению практических работ и самостоятельной работы. Данный практикум имеет целью закрепить и расширить теоретические знания студентов в области многоантенных систем беспроводной связи путем представления студентам основной информации об алгоритмах обработки и формирования сигналов в многоантенных системах.

Одобрено на заседании кафедры ТОР, протокол № 1 от 31 августа 2023 г.

УДК 621.396.2
ББК 32.84

Оглавление

ВВЕДЕНИЕ

Практикум имеет целью закрепить и расширить теоретические знания студентов в области многоантенных систем, познакомить с основами пространственной обработки сигналов и пространственного разделения каналов в режиме многопользовательского доступа.

Практикум содержит описание следующих работ:

1) Начало работы с Octave;

2) Формирование модели системы беспроводной связи для конфигурации SISO;

3) Пространственно-временное кодирование;

4) Формирование модели системы беспроводной связи для конфигурации Single User MIMO;

5) Цифровое диаграммобразование для многоантенных систем связи;

6) Формирование модели системы беспроводной связи для конфигурации Multi User MIMO;

7) Формирование модели системы беспроводной связи для конфигурации Multi User Massive MIMO;

Работы данного перечня выполняются в среде Matlab.

Работа № 1

«Начало работы с Octave»

Цель работы: изучить основные функции и блоки Octave и составить тестовую программу.

Задачи лабораторной работы:

- Изучить основные функции и блоки Octave.
- Произвести формирование синусоидального сигнала в Octave.
- Произвести сложение и умножение гармонических сигналов.

1. Теоретический материал

Octave – высокоуровневый интерпретируемый язык программирования, предназначенный для решения задач вычислительной математики. После запуска Octave пользователь видит рабочую область Octave, как на рисунке 1.1.

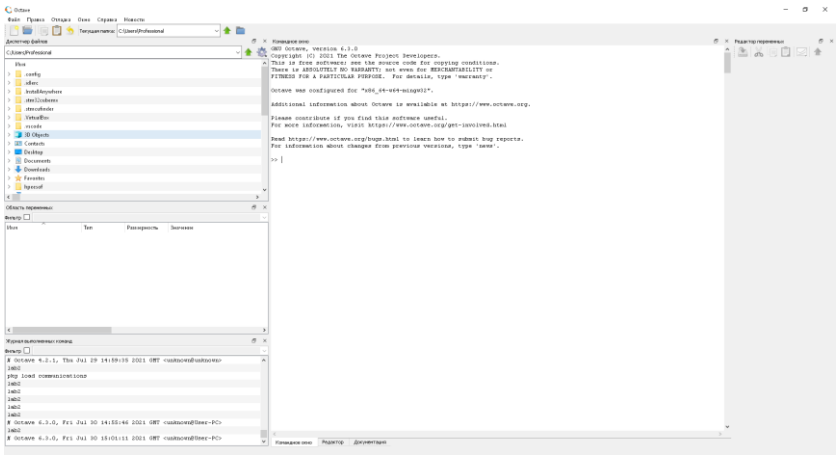

Рисунок 1.1 – Рабочая область Octave

Для создания нового проекта кликните на пустой лист в левом верхнем углу, как это показано на рисунке 1.2.

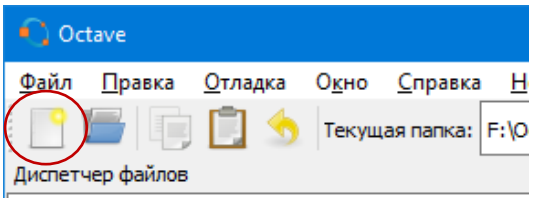

Рисунок 1.2 – Иконка «Создать сценарий»

После создания проекта вы можете начать писать свой код, по завершению скомпилируйте его, нажав на кнопку компиляции, изображенную на рисунке 1.3, или используя горячую клавишу F5.

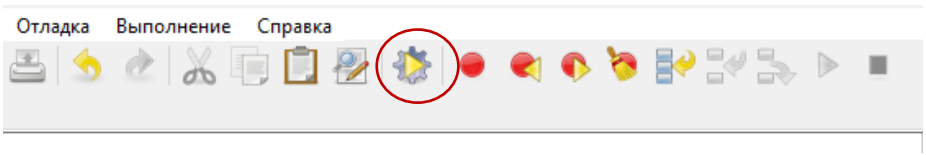

Рисунок 1.3 – Компиляция проекта

Для работы в среде Octave с функциями, относящимися к области телекоммуникаций, необходимо загрузить нужную библиотеку. Для этого перейдите в командное окно и введите «pkg load communications», как это показано на рисунке 1.4.

```
The 'awgn' function belongs to the communications package from Octave
Forge which you have installed but not loaded.  To load the package, run
'pkg load communications' from the Octave prompt.

Please read <https://www.octave.org/missing.html> to learn how you can
contribute missing functionality.
error: called from
    lab2 at line 26 column 14
>> pkg load communications
```

Рисунок 1.4 – Загрузка библиотек

В работе будут реализованы следующие этапы:

Рисунок 1.5 – Этапы выполнения работы

2. Ход выполнения работы

1. Первые три строки программы как правило такие:

```
clc
clear all
close all
```

clc – очищает «командное окно»;
clear all – удаляет все переменные из «область переменных», очищает память;
close all – закрывает все открытые фигуры.

2. Постройте гармонический сигнал во временной и частотной области со следующими параметрами:

F0 = 10 кГц – несущая частота,
Fs = 250 кГц – частота дискретизации,
N = 100 – количество отсчетов.
Ниже представлена реализация в Octave.

```
clc
clear all
close all
F0 = ...;
```

```
Fs = ...;
N = ...;
t = (0:N-1)/Fs; % временные отсчеты
sig = sin (2*pi*F0*t);% гармонический сигнал

% Построение временной формы сигнала
figure
plot(t,sig);
grid on
xlabel('Time, s','fontsize',16);
ylabel('Amplitude','fontsize',16);
```

Результат выполнения кода показана на рисунке 2.1.

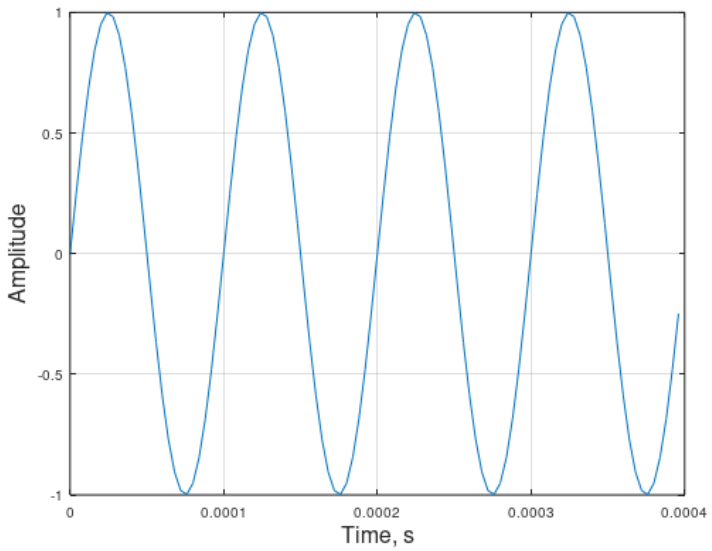

Рисунок 2.1 – Временная форма гармонического сигнала

3. Для того чтобы рассчитать спектр сгенерированного гармонического сигнала sig воспользуйтесь функцией fft(), которая выполняет операцию быстрого или дискретного преобразования Фурье в зависимости от длины массива.

```
spec = fft(...);
```

4. Выполните построение спектра сигнала. Для начала необходимо правильно задать ось частот, которая будет занимать диапазон от 0 Гц до Fs с шагом Fs/N.

```
f = (0:N-1)*(Fs/N);
```

5. Далее по аналогии с пунктом 2 выполните построение переменной spec на частотной оси f. Подпишите ось ординат как 'Amplitude' и ось абсцисс – 'Frequency, Hz'.

```
% Построение временной формы сигнала
figure
plot(..., ...);
grid on
xlabel('...','fontsize',16);
```

```
ylabel('...','fontsize',16);
```

В результате должны получить спектр гармонического сигнала как на рисунке 2.2.

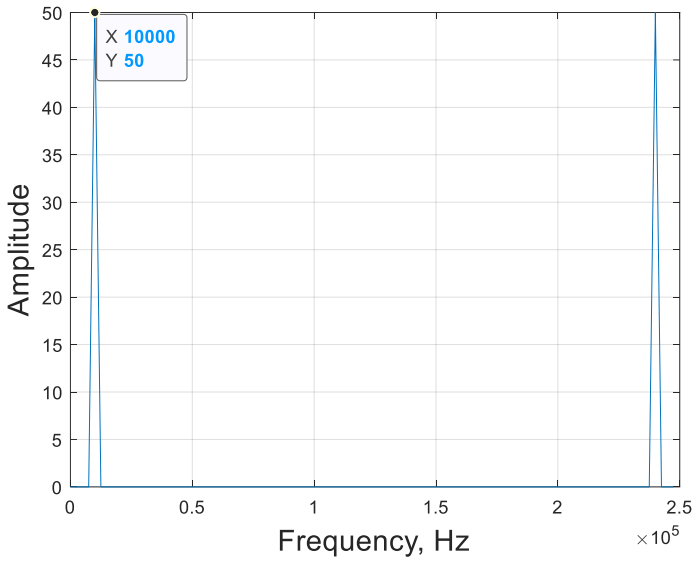

Рисунок 2.2 – Спектр гармонического сигнала

3. Самостоятельные задачи

Задача 1. Постройте спектр синусоидального сигнала, используя функцию fft(). Задайте ось частот аналогично временным отсчетам. Сформируйте еще один синусоидальный сигнал частотой 30 кГц. Используя цикл for, произведите перемножение и суммирование 2 синусоидальных сигналов.

Задача 2. Создайте две битовые последовательности (A и B) состоящие из 100 элементов ($N=100$), используя функцию randi. Используя цикл for и функцию if, произведите суммирование по модулю 2. Таблица истинности XOR представлена в таблице 3.1.

Таблица 3.1 – Таблица истинности XOR

ВХОД		ВЫХОД
0	0	0
0	1	1
1	0	1
1	1	0

4. Контрольные вопросы к работе

1. Для чего может использоваться среда разработки Octave?
2. Как вывести рисунок на экран?
3. Как рассчитать спектр сигнала?
4. Как задать псевдослучайную битовую последовательность?

5. Требования к оформлению отчета

Содержание отчета:
1. Введение.
2. Описание работы.
3. Рисунки:

- Гармонический сигнал.
- Спектр гармонического сигнала.
- Временная форма произведения гармонических сигналов.
- Спектр произведения гармонических сигналов.
- Временная форма суммы гармонических сигналов.
- Спектр суммы гармонических сигналов.

4. Выводы.
5. Полный листинг программы.

Работа № 2

«Формирование модели системы беспроводной связи для конфигурации SISO»

Цель работы: произвести цифровую модуляцию и демодуляцию сигнала. Смоделировать передачу сигнала через канал с аддитивным белым гауссовским шумом. Построить зависимость вероятности битовой ошибки (Bit Error Ratio, BER) от отношения сигнал-шум (Signal-to-Noise Ratio, SNR).

Задачи лабораторной работы:

- Произвести модуляцию и демодуляцию сигнала в среде Octave.
- Имитировать передачу сигнала через канал с аддитивным белым гауссовским шумом.
- Рассчитать значения BER.
- Построить созвездия сигнала и график зависимости BER от SNR.

1. Теоретический материал

Модуляция – процесс изменения одного или нескольких параметров модулируемого несущего сигнала при помощи модулирующего информационного сигнала.

Виды модуляции: амплитудная и фазовая.

В данной лабораторной работе будет рассмотрена квадратурная фазовая модуляция (QPSK).

QPSK модуляция строится на основе кодирования двух бит передаваемой информации одним символом модуляции (рисунок 1.1). Символ модуляции графически отображается точкой созвездия, а математически – комплексным числом I+iQ.

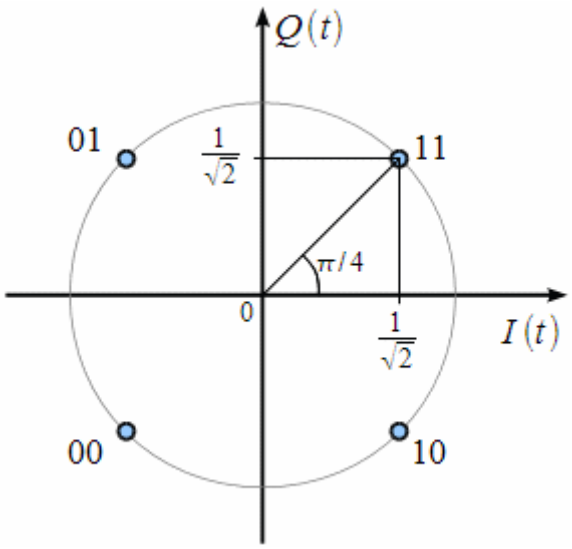

Рисунок 1.1 – Векторная диаграмма QPSK сигнала (Сигнальное созвездие)

То есть при модуляции сигнала каждому возможному значение дибита ставится в соответствие определенный символ модуляции, определяющий амплитуду и фазу несущего колебания. В таблице 1.1 представлены комплексные значения для квадратурной амплитудной модуляции.

Таблица 1.1 – Пример QPSK модуляции

Биты (до модуляции)	I и Q компоненты (после модуляции)
11	1+1i

Окончание таблицы 1.1

01	-1+1i
10	1-1i
00	-1-1i

Для демодуляции сигнальное созвездие делится на четверти. В зависимости от того, к какой области относится принятый символ модуляции, определяется значение битовой последовательности. Например, на рисунке 1.2 все точки созвездия, находящиеся в закрашенной области, вероятнее всего при передаче имели значение 1+1i, что соответствует битовому значению 11 до модуляции.

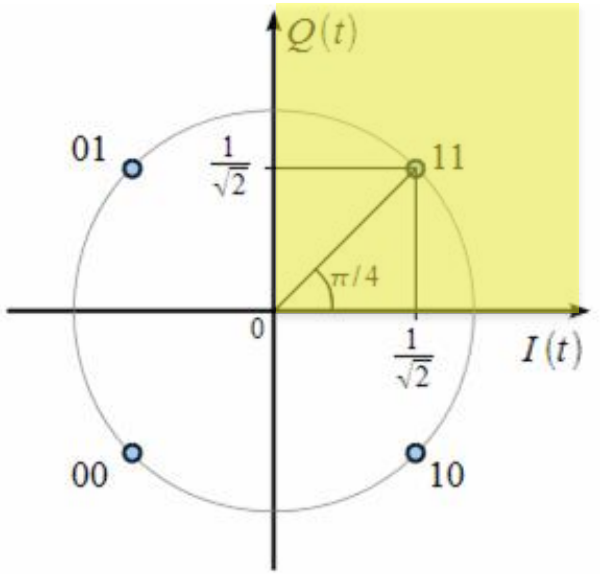

Рисунок 1.2 – Демодуляция

В системах связи возможно возникновение ошибок из-за шума, помех, искажений или битовой рассинхронизации. Вероятность битовой ошибки, BER (Bit Error Rate) – это количество битовых ошибок, деленное на общее число переданных битов в течение периода времени. Коэффициент ошибок является безразмерной величиной и часто выражается в процентах.

2. Ход выполнения работы

1. Создайте битовую последовательность из 5000 символов, используя функцию randi().

```
clc
clear all
close all
N = ...;
bits = ...;
```

2. Произведите QPSK модуляцию, используя цикл for и условие if. Ниже приведено одно условие из четырёх, остальные допишите, опираясь на таблицу 1.1.

```
k=1;
for i = 1:2:N
```

```
  if bits(i) == 1 && bits (i+1) == 1
    mod_data (k) = 1+1i;
  end
  ...
  k=k+1;
end
```

Постройте созвездие модулированного сигнала, используя функцию scatterplot(). Для использования этой функции необходимо загрузить ее, написав «pkg load communications» в командное окно, и выполнить. Если автомасштабирование расположило точки на краях отображаемой области или за ней, задайте оси самостоятельно при помощи функций ylim([min max]), xlim([min max]). Получение созвездие должно соответствовать рисунку 2.1.

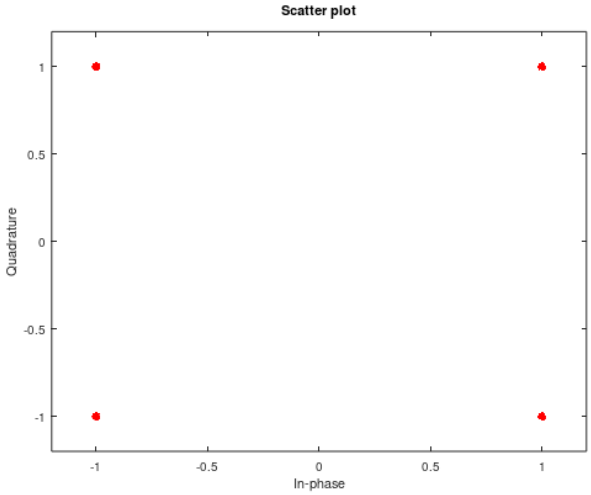

Рисунок 2.1 – Созвездие QPSK сигнала

3. Используя функцию awgn(), добавьте шум к модулированному сигналу. Задайте значение SNR, равным 25.

```
SNR=...;
data_noise = awgn(mod_data,SNR,'measured');
```

Постройте созвездие сигнала с шумом, используя функцию plot(). Сделайте точки толще и измените их цвет на красный, добавив «”r.”». Созвездие QPSK сигнала под воздействием шума представлено на рисунке 2.2.

```
plot(data_noise, "r.");
```

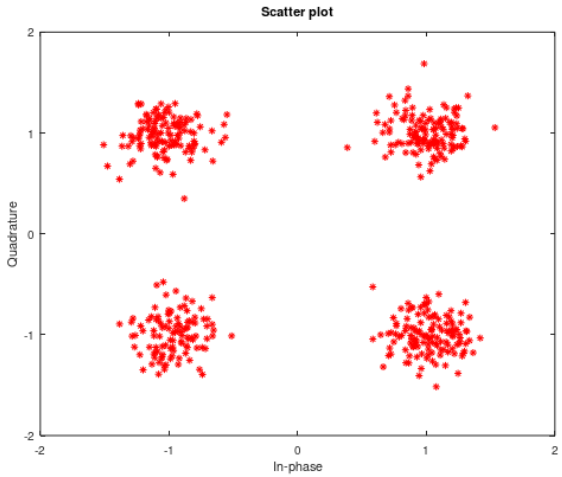

Рисунок 2.2 – Созвездие QPSK сигнала под воздействием шума

Отобразите сигнальные созвездия для этих значений в отчете.

4. Используя условный оператор if и цикл for, осуществите демодуляцию. Ниже приведены только одно условие из четырёх. Добавьте еще три условия, необходимых для демодуляции.

```
k=1;
for i = 1:length(mod_data_noise)
   if real(mod_data_noise (i))  >0 && imag(mod_data_noise (i)) >0
     bits_demod (k) = 1; bits_demod (k+1) = 1;
  end
  ...
  k=k+2;
end
```

5. Для того чтобы посчитать BER необходимо сравнить массивы bits и bits_demod. Битовое сравнение выполняет функция biterr. Функция возвращает количество несовпадений, то есть в данном случае ошибок.

```
BER_1 = biterr(...,...);
```

6. Чтобы построить зависимость BER от SNR, необходимо увеличить количество передаваемых бит N, указанное в начале кода, со значения $N = 5000$ на $N = 10000$.

```
N = ...;
```

Создайте в начале кода два пустых безразмерных массива, они будут использоваться для записи значений BER при разных значениях SNR. Пустой безразмерный массив задаётся посредством присвоения двух квадратных скобок [].

```
BER_B = [];
SNR_S = ...;
```

В предыдущих пунктах было указано фиксированное значение $SNR = 25$, теперь для того, чтобы построить зависимость BER от SNR, значение SNR необходимо изменять в диапазоне от –10 дБ до 25 дБ с шагом 1 дБ. Для этого измените код, создав цикл, в котором в качестве итерационной переменной будет использоваться переменная SNR.

```
for SNR = -10:1:25
```

В пункте 5 был рассчитан BER для одного значения SNR. Создайте переменную BER, в которую будут записываться значения вероятности для текущего SNR. Используйте ранее созданные два накопителя, в которые будут записываться все значения BER и SNR посредством их конкатенации. После пропишите окончание цикла.

```
BER = BER_1/N;
BER_B = [BER_B BER];
SNR_S = [SNR_S SNR];
end;
```

Постройте график зависимости BER от SNR, используя команды figure и semiology. «”-ok”» – делает график чёрно-белым и меняет маркер точек. Добавьте подписи к осям и дайте название графику. В результате изображение должно получиться, как на рисунке 2.3.

```
figure
semilogy(SNR_S, BER_B,'-ok')
grid;
title('OFDM Bit Error Rate .VS. Signal To Noise Ratio');
ylabel('BER');
xlabel('SNR [dB]');
```

Чтобы наглядно увидеть, как SNR влияет на созвездие, напишите перед циклом for команду figure. А после построения графика с шумом добавьте следующий код.

```
grid on
ylim([-2,2]);
pause(0.1)
```

grid on – позволяет строить несколько графиков в одном окне, ylim – задает лимит по оси Oy, pause – делает паузу между построениями.

В результате вы получите график зависимости BER от SNR, как на рисунке 2.3.

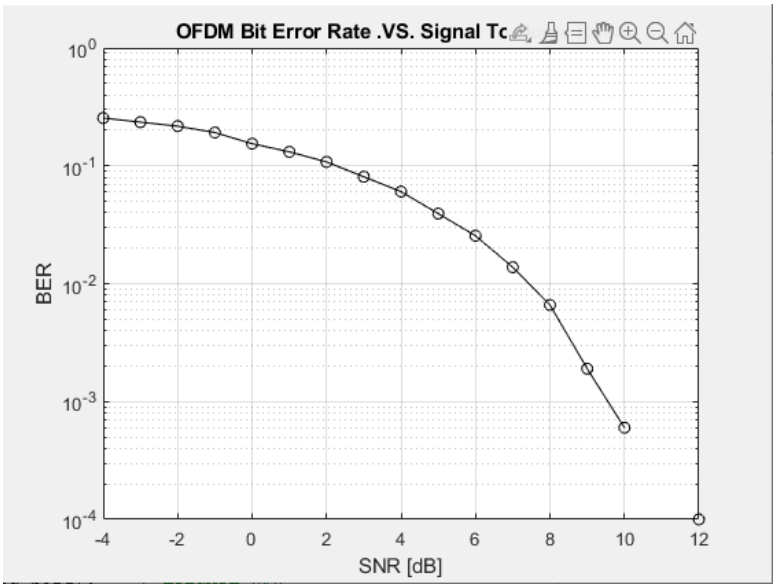

Рисунок 2.3 – Зависимость вероятности битовой ошибки от отношения сигнал/шум

3. Контрольные вопросы к работе:

1. Что такое QPSK модуляция?
2. Что такое BER, SNR?

4. Требования к оформлению отчета:

Содержание отчета:
1. Введение;
2. Блок схема алгоритма, который вы применили в работе;
3. Параметры OFDM символа;
4. Рисунки:
   - Созвездие QPSK сигнала при разном отношении сигнал шум;
   - График зависимости BER от SNR;
5. Выводы;
6. Листинг программы.

Работа № 3

«Пространственно-временное кодирование»

Цель работы: формирование и обработка OFDM сигналов в MISO системе с пространственно-временным блочным кодированием по схеме Аламоути.

Задачи лабораторной работы:

- Формирование сигналов OFDM сигналов;
- Моделирование канала связи сигнала с такими эффектами как шум и многолучевое распространение;
- Кодирование сигналов по схеме Аламоути;
- Обработка принятого сигнала и извлечение полезной информации.

Space Time Block Codes (STBC) или же пространственно-временное блочное кодирование – это способ передачи информации через несколько антенн, обеспечивающий повышенную помехозащищенность сигнала. В данной работе рассматривается простейший случай STBC, представляющий собой схему Аламоути с 2 передающими антеннами и 1 приемной.

1. Ход выполнения работы

Создайте новый проект в Matlab. Запишите следующие команды для очистки предыдущих значений:

```
clc
close all
clear all
```

Первая строка очищает командное окно, вторая закрывает все открытые окна и графики, третья – очищает область переменных. Данные команды рекомендуется писать в начале каждого проекта.

Далее нужно задать основные параметры системы: размерность OFDM символа, размер защитных интервалов, индекс модуляции, количество передаваемых бит. Данный фрагмент программы выглядит следующим образом:

```
%% ================================= Переменные
NFFT = 1024;    % Размерность OFDM символа
index_mod1 = 4; % индекс модуляции первого символа
index_mod2 = 4; % индекс модуляции второго символа
gi = 199;       % Размер защитных интервалов OFDM символа
numbits = (NFFT-gi-1); % Число передаваемых бит
```

Используя заданные параметры, необходимо сформировать вектор информационных бит, выполнить его модуляцию, сформировать из полученных данных два OFDM символа и выполнить их кодирование по схеме Аламоути. Для этой цели используйте следующую конструкцию:

```
%% ================================= Модуляция
data_bit1 = randi([0, index_mod1-1],1, numbits);
data_bit2 = randi([0, index_mod2-1],1, numbits);

%Осуществите QPSK модуляцию data_bit.
```

```
modqpsk1 = qammod(data_bit1.', index_mod1).';
modqpsk2 = qammod(data_bit2.', index_mod2).';

% Сформируйте два OFDM символа.
sym1 = [zeros(1,100), modqpsk1, zeros(1,100)];
sym2 = [zeros(1,100), modqpsk2, zeros(1,100)];

%Задайте еще два OFDM символа следующим образом.
sym1z = conj(sym1);
sym2z = -1.*conj(sym2);
```

Разберем представленный код.

Переменные data_bit1 и data_bit2 представляют собой вектора 0 и 1. Функция qammod применяется для модуляции этих двух векторов. Из модулированных данных формируется два OFDM символа – sym1 и sym2. После чего формируются еще два OFDM символа – sym1z и sym2z – для выполнения кодирования по схеме Аламоути.

Код формирования многолучевого канала представлен ниже:

```
%% ================================ Channel
%Сформируйте 4 передаточные функции для канальной матрицы.
h11 = zeros(1,NFFT);
h11(1) = 1;
h11(2) = 0.5;
h11(5) = 0.3;
H11 = fft(h11);

h22 = zeros(1,NFFT);
h22(1) = 1;
h22(8) = 0.5;
h22(9) = 0.3;
H22 = fft(h22);
```

Переменные H11 и H22 представляют собой искажения в частотной области – вместе они формируют канальную матрицу. Для внесения данных искажений в сигнал, необходимо спектр обоих OFDM символов перемножить с этими переменными. После чего сигналы переносятся во временную область с помощью обратного преобразования Фурье и суммируются с шумом. Но помимо сигналов, кодированных с помощью Аламоути, также передадим сигналы sym1 и sym2 по обычной SISO схеме. Это необходимо, чтобы оценить преимущество MISO Аламоути перед SISO.

```
%% ================================ Receiver
power = -40;
noise = wgn(1,NFFT,power);

% Многолучевые искажения
y1 =  ifft( sym1.*H11) + ifft( sym2.*H22) ;
y2 =  ifft(sym2z.*H11) + ifft(sym1z.*H22) ;

y1_siso =  ifft( sym1.*H11);
y2_siso =  ifft( sym2.*H22);
```

```
%Добавляем каждому символу шум и переносим их в область частот
n1 = fft( y1+noise );
n2 = fft( y2+noise );

n1_siso = fft( y1_siso+noise );
n2_siso = fft( y2_siso+noise );
```

y1 и y2 это OFDM символы во временной области с добавленным эффектом многолучевости. Далее к ним добавляется шум, который создается с помощью функции wgn, после чего осуществляется перенос в частотную область – переменные n1 и n2. Соответственно y1_siso, y1_siso, n1_siso и n2_siso – это сигналы, переданные без кодирования Аламоути. Мощность шума задается переменной power и измеряется в дБ. Из принятых сигналов n1, n2, n1_siso и n2_siso необходимо устранить влияние многолучевого канала, а из n1 и n2 также и влияние одного сигнала на другой. Воспользуемся формулой из лекции.

```
%% ================================ Estimation
x1_ocen = (conj(H11).*n1+H22.*conj(n2))./(abs(H11).^2+abs(H22).^2);
x2_ocen = (conj(H22).*n1-H11.*conj(n2))./(abs(H11).^2+abs(H22).^2);

%Демодулируйте принятые символы
rxData1 = qamdemod(x1_ocen(101:924).', index_mod1).';
rxData2 = qamdemod(..., index_mod2).';

x1_ocen_siso = n1_siso./H11;
x2_ocen_siso = ...;

%Демодулируйте принятые символы
rxData1_siso = qamdemod(..., index_mod1).';
rxData2_siso = qamdemod(..., index_mod2).';
```

Переменные x1_ocen и x2_ocen содержат восстановленные OFDM символы, извлеките из них полезную информацию игнорируя защитные интервалы, после чего осуществите демодуляцию – переменные rxData1 и rxData2. Тоже проделайте с переменными x1_ocen_siso и x2_ocen_siso.

Посчитайте количество ошибок для Аламоути и SISO. Выведите результат в консоль. При заданном уровне шума, в Аламоути их количество должно быть равно 0, а в случае SISO отлично от нуля.

```
%% ================================ BER
Errors_MIMO1 = sum(abs(rxData1-data_bit1));
Errors_MIMO2 = sum(abs(rxData2-data_bit2));

fprintf('Alamouti num errors = %d\n', Errors_MIMO1+Errors_MIMO2)

Errors_SISO1 = sum(abs(rxData1_siso-data_bit1));
Errors_SISO2 = sum(abs(rxData2_siso-data_bit2));
```

```
fprintf('SISO num errors = %d\n', Errors_SISO1+Errors_SISO2)
```

2. Контрольные вопросы к работе

1) Для чего используется кодирование Аламоути?

2) У какой системы связи помехозащищенность больше: SISO или MISO 2 на 1 с кодированием Аламоути?

3) В чем преимущество схемы MISO 2 на 1 с кодированием Аламоути перед классической схемой MISO 2 на 1 без кодирования.

Работа № 4

«Формирование модели системы беспроводной связи для конфигурации Single User MIMO»

Цель работы: Формирование и обработка OFDM сигналов в SU-MIMO системе с пространственным мультиплексированием.

Задачи лабораторной работы:

- Формирование и передача сигналов по схеме MIMO 2 на 2 с одним пользователем;
- Реализовать канал связи для MIMO;
- Осуществить эквалайзирование принятого сигнала. Восстановить сигнал.

За счет использования нескольких передающих и приемных антенн в MIMO системах возможно добиться повышенной скорости передачи – данный подход называется пространственным мультиплексированием, когда каждая антенна излучает свой сигнал. В данной работе рассматривается формирование и обработка OFDM сигнала в системе MIMO 2 на 2.

1. Ход выполнения работы

Передающая часть представлена ниже:

```
clc
clear all
close all

%% ================================== Переменные
NFFT = 1024;    % Размерность OFDM символа
index_mod1 = 4; % индекс модуляции первого символа
index_mod2 = 4; % индекс модуляции второго символа
gi = 199;       % Размер защитных интервалов OFDM символа
numbits = (NFFT-gi-1); % Число передаваемых бит
snr = 35; % Отношение сигнал/шум

%% ================================== Модуляция
data_bit1 = randi([0, index_mod1-1],1, numbits);
data_bit2 = randi([0, index_mod2-1],1, numbits);

%Осуществите QPSK модуляцию переменной data_bit.
modqpsk1 = qammod(data_bit1.', index_mod1).';
modqpsk2 = qammod(data_bit2.', index_mod2).';

% Сформируйте два OFDM символа.
sym1 = [zeros(1,100), modqpsk1, zeros(1,100)];
sym2 = [zeros(1,100), modqpsk2, zeros(1,100)];
```

sym1 и sym2 – это передаваемые OFDM символы. Так как в используемой MIMO системе 4 пути распространения, тогда необходимо задать 4 передаточные характеристики:

```
%% ================================== Channel
%Формируем 4 передаточные функции для канальной матрицы.
h11 = zeros(1,NFFT);
```

```
h11(1) = 1;
h11(2) = 0.5;
h11(5) = 0.3;
H11 = fft(h11);

h12 = zeros(1,NFFT);
h12(1) = 1;
h12(2) = 0.5;
h12(4) = 0.3;
H12 = fft(h12);

h21 = zeros(1,NFFT);
h21(1) = 1;
h21(3) = 0.5;
h21(6) = 0.3;
H21 = fft(h21);

h22 = zeros(1,NFFT);
h22(1) = 1;
h22(8) = 0.5;
h22(9) = 0.3;
H22 = fft(h22);
```

В приемнике в каждой из антенн будет сумма двух сигналов, каждый сигнал будет подвержен влиянию соответствующего ему канала распространения:

```
%% ================================ Receiver
%Записываем сигналы принимаемые первой и второй приемной антеннами MIMO 2x2
системы
y1 = ifft(sym1.*H11) + ifft(sym2.*H12);
y2 = ifft(sym1.*H21) + ifft(sym2.*H22);

n1 = fft(awgn(y1, snr, 'measured'));
n2 = fft(awgn(y2, snr, 'measured'));
```

Переменные y1 и y2 это сигналы, принятые первой и второй антенной соответственно. Каждый сигнал умножается на соответствующую ему передаточную характеристику. Переменные n1 и n2 – это принятые сигналы в частотно области. Постройте диаграмму созвездия переменной n1 или n2.

```
figure(1)
plot(n1,'.')
title('Принятый сигнал')
```

Результат построения показан на рисунке 1.1.

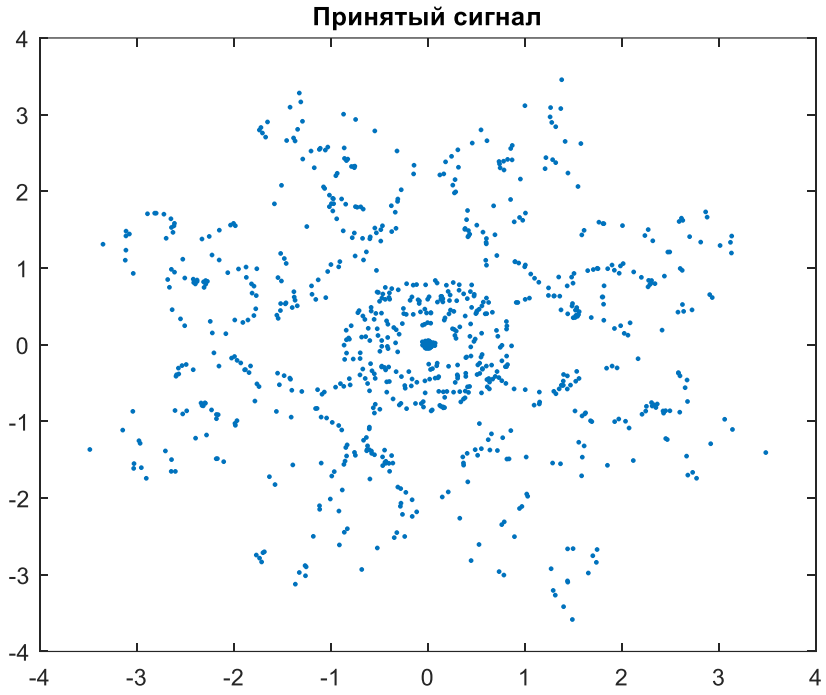

Рисунок 1.1 – Диаграммы созвездия принятого сигнала

Из такого созвездия не удастся извлечь полезную информацию, так как принятый сигнал – это сумма двух сигналов, прошедших через многолучевой канал. Воспользуйтесь формулой из лекций, чтобы вычислить обратную канальную матрицу, после чего выполните восстановление переданных сигналов:

```
%% ================================ Estimation
%Найдем определитель матрицы
H_det = ...; % Детерминант

% Вычислим оценку
x1_ocen = ...;
x2_ocen = ...;

% Демодулируйте принятые символы
rxData1 = qamdemod(x1_ocen(101:924).', index_mod1).';
rxData2 = qamdemod(x2_ocen(101:924).', index_mod2).';
```

Постройте созвездие восстановленных сигналов, переменная – x1_ocen или x2_ocen. Сравните с созвездием до выполнения операции восстановления сигнала.

```
figure(2)
plot(x1_ocen,'.')
title('Восстановленный сигнал')
```

На рисунке 1.2 приведён пример созвездия x1_ocen.

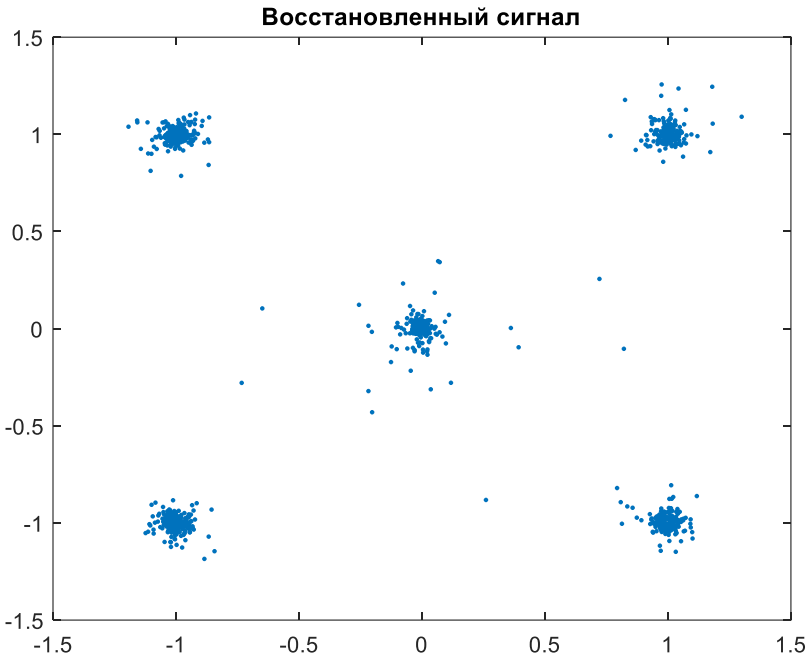

Рисунок 1.2 – Диаграммы созвездия восстановленного сигнала

Посчитайте количество ошибок и выведите результат в консоль, при заданном SNR их количество должно быть равно 0.

```
%% ================================ BER
Errors_MIMO1 = sum(abs(rxData1-data_bit1));
Errors_MIMO2 = sum(abs(rxData2-data_bit2));
fprintf('MIMO num errors = %d\n', Errors_MIMO1+Errors_MIMO2)
```

2. Контрольные вопросы

1. Для чего используется пространственное мультиплексирование MIMO?
2. У какой системы связи помехозащищенность больше: MIMO или MIMO с кодированием Аламоути?
3. У какой системы связи скорость передачи данных больше: MIMO или SISO?

Работа № 5

«Цифровое диаграммобразование для многоантенных систем связи»

Цель работы: Формирование и обработка OFDM сигналов в MIMO системе с использованием технологии Beamforming.

Задачи лабораторной работы:

- Формирование сигналов по схеме MIMO 2 на 2.
- Осуществление передачи с использованием Beamforming.
- Обработка принятого сигнала и извлечение полезной информации.

Beamforming это технология, которая позволяет максимально эффективно осуществлять передачу сигналов при заданном канале. На основании известной канальной матрицы вычисляется усиление в каждом канале, что позволяет оптимально распределить сигналы по каждому каналу для максимизации скорости передачи. В данной работе рассматривается формирование и обработка КАМ сигналов в системе MIMO 2 на 2.

1. Ход выполнения работы

Зададим параметры модели:

```
clc;
clear all;  %#ok<CLALL>
close all;

%% ================================== %% Параметры модели
Nt = 2; % Число передающих антенн
Nr = 2; % Число приемных антенн
N_sym = 20000; % Число бит
snr = 20;
```

Создадим вектор бит и выполним КАМ модуляцию:

```
%% ================================ Signal formatting
msg_bits = randi([0, 1], 2, N_sym);            % Bit generation
x = 0.707*qammod(msg_bits.', 4, 'InputType', 'bit').'; % Modulation
```

Выполним сингулярное разложение канальной матрицы и осуществим прекодирование сигналов. Также выведите в консоль усиление в каждом канале:

```
%% ================================ Precoding
H = randn(Nr,Nt) + 1i*randn(Nr,Nt);
[U, S, V] = svd(H);
W_tx_1 = V(:,1);
W_rx_1 = U(:,1)';

W_tx_2 = V(:,2);
W_rx_2 = U(:,2)';

x1 = [x(1,:); x(1,:)];
x2 = [x(2,:); x(2,:)];
X_Tx = W_tx_1.*x1 + W_tx_2.*x2;
```

```
fprintf('Channel Gain 1 = %.2f\n', S(1,1));
fprintf('Channel Gain 2 = %.2f\n', S(2,2));
```

В консоле Вы должны увидеть, что усиление первого канала всегда будет больше усиления второго канала, как это показано на рисунке 1.1.

```
Channel Gain 1 = 2.09
Channel Gain 2 = 0.47
```

Рисунок 1.1 – Усиление первого и второго канала

Так как канальная матрица формируется случайным образом при каждом запуске, то усиление будет каждый раз принимать другие значения. Но в первом канале усиление всегда будет больше чем во втором.

Постройте созвездие прекодированного сигнала, пример представлен на рисунке 1.2.

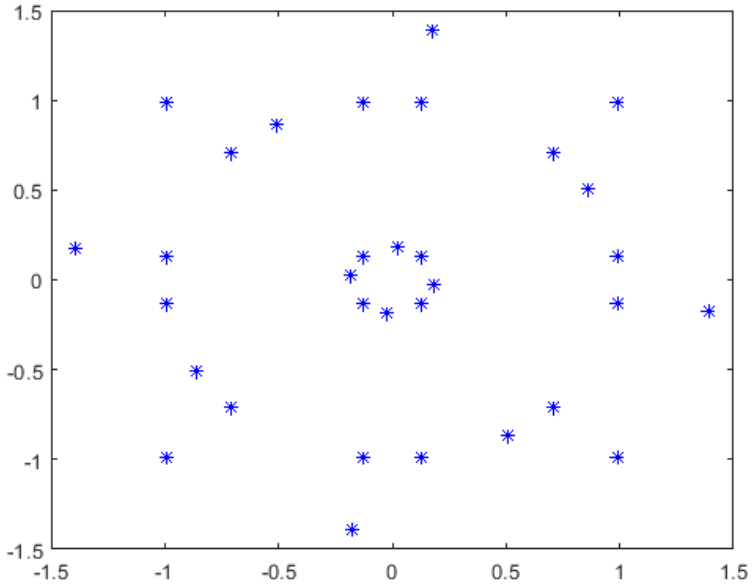

Рисунок 1.2 – Прекодированный сигнал

Добавьте к сигналу шум и постройте созвездие принятого сигнала.

```
%% ================================ Rx signal
Y = awgn(H*X_Tx, snr, 'measured');
```

Пример созвездия принятого сигнала представлен на рисунке 1.3.

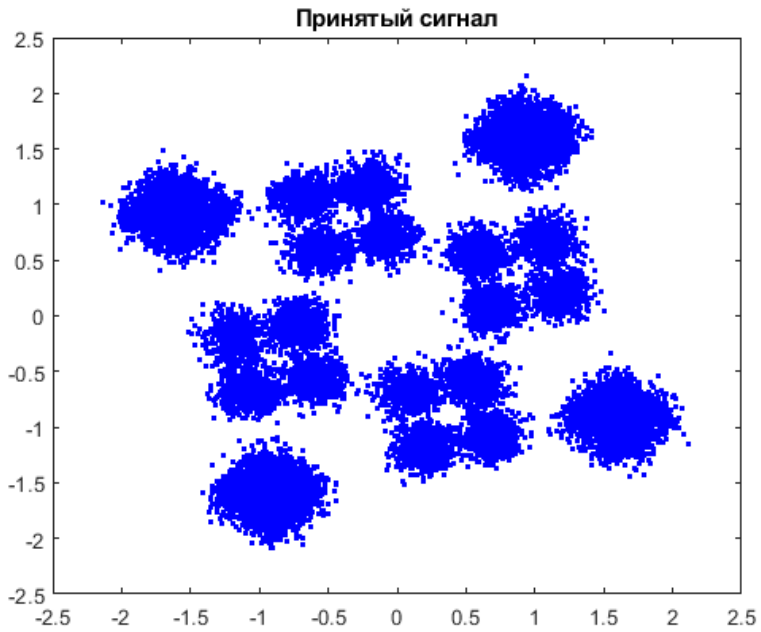

Рисунок 1.3 – Принятый сигнал

Выполните восстановление переданных сигналов используя левые сингулярные вектора и матрицу коэффициентов усиления канала, а также вычислите количество ошибок в обоих сигналах.

```
%% ================================= Channel estimation
X_hat = [W_rx_1*Y./S(1,1); W_rx_2*Y./S(2,2)];
demapped  = qamdemod(X_hat.', 4, 'OutputType', 'bit').';        % Bit modulation
N_ebits_1   = sum(msg_bits(1,:)~=demapped(1,:));
N_ebits_2   = sum(msg_bits(2,:)~=demapped(2,:));
fprintf('Num Errors in the first channel = %d\n', N_ebits_1);
fprintf('Num Errors in the second channel = %d\n', N_ebits_2);
```

Постройте диаграммы созвездия обоих сигналов. Пример показан на рисунке 1.4.

```
%% ================================= Plot constellation
figure(3); hold on; grid on;
plot( X_hat(2,:), 'b.');
plot( X_hat(1,:), 'r.');
legend('Принятый сигнал из второго канала', ...
     'Принятый сигнал из первого канала')
```

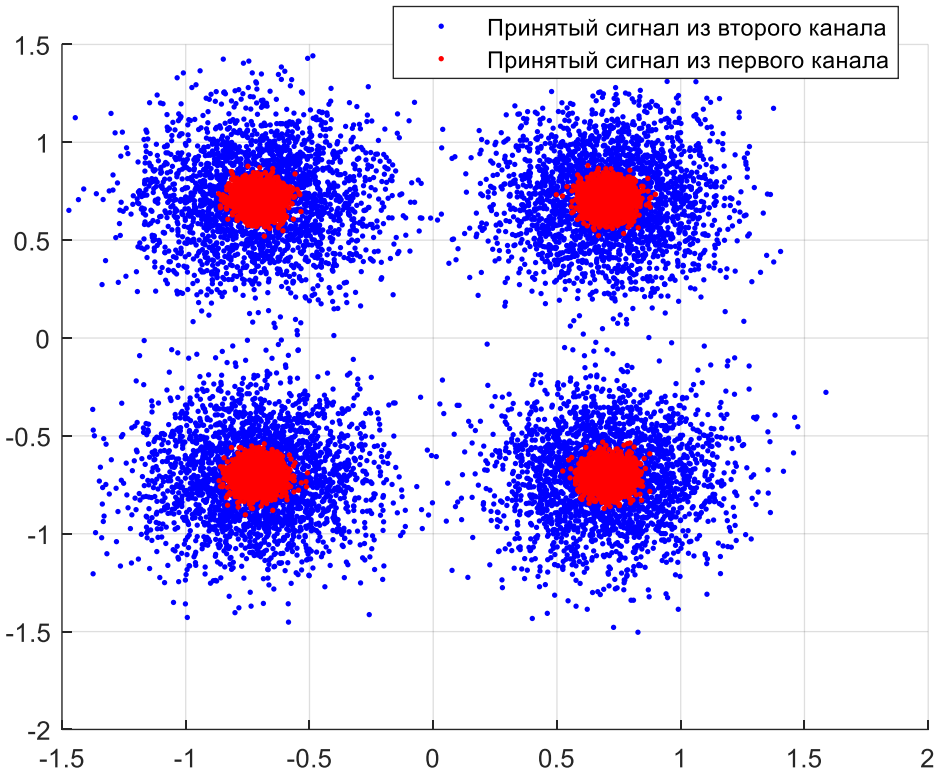

Рисунок 1.4 – Диаграммы созвездия принятых сигналов

Из графиков можно видеть, что отношение сигнал/шум в первом сигнале значительно выше чем во втором. Это обусловлено тем, что усиление в первом канале всегда превышает усиление второго канала.

2. Контрольные вопросы

1. Для чего используется beamforming MIMO?
2. У какой системы связи помехозащищенность больше: beamforming MIMO или MIMO с кодированием Аламоути?
3. Возможно ли повысить скорость передачи с помощью beamforming MIMO или данная технология используется только для увеличения помехозащищенности?

Работа № 6

«Формирование модели системы беспроводной связи для конфигурации Multi User MIMO»

Цель работы: Формирование и обработка сигналов в MU MIMO системе.

Задачи лабораторной работы:

- Формирование сигналов по схеме MU MIMO с двумя пользователями.
- Выполнение прекодирования сигналов.
- Извлечение сигналов каждого пользователя. Эквалайзирование принятого сигнала и извлечение полезной информации.

Ранее были рассмотрены MIMO системы с одним пользователем. В данной работе в системе MIMO будет несколько пользователей, у каждого пользователя несколько антенн. Сигналы пользователей суммируются в передатчике и прекодируются. Каждый пользователь принимает данную сумму и извлекает сигнал, предназначенный для него, отбрасывая сигнал другого пользователя.

1. Ход выполнения работы

Данная работа является продолжением работы «Формирование сигналов первичной и вторичной синхронизации». Из предыдущей работы вам необходимо взять код по формированию сигналов синхронизации PSS и SSS, а также загрузить сформированный кадр.

Установите следующие параметры модели.

```
clc;
clear all;  %#ok<CLALL>
close all;

%% ================================ Параметры модели
N_frame    = 1000;  % Number of frames
b1         = 2;  % Number of bits per QPSK symbol
b2         = 2;  % Number of bits per QPSK symbol
NT         = 4;  % Number of Tx antennas
N_user     = 2;  % Number of users
NR         = 2;  % Number of Rx antennas
N_pbits1 = N_frame*NT*b1;     % Number of bits in a packet
N_pbits2 = N_frame*NT*b2;     % Number of bits in a packet
SNRdB    = 35;
```

Количество передающих антенн NT равно 4, количество пользователей N_user 2, количество антенн у каждого пользователя NR 2.

Выполните QAM модуляцию. Чтобы визуально отличить на диаграмме созвездия сигнал одного пользователя от другого, введем в сигнал второго пользователя сдвиг в 45 градусов:

```
%% ================================ Signal formatting
msg_bit1 = randi([0, 1], 1, N_pbits1);                % Bit generation
msg_bit2 = randi([0, 1], 1, N_pbits2);                % Bit generation
symbol1 = 0.707.*qammod(msg_bit1.',4,'InputType', 'bit').'; % 1 user
symbol2 = 0.707.*qammod(msg_bit2.',4,'InputType', 'bit').'; % 2 user
```

```
symbol2 = symbol2.*exp(1i*45*pi/180);
x1 = reshape(symbol1, NR, []);
x2 = reshape(symbol2, NR, []);
x = [x1; x2];
```

Размерность сформированных сигналов изменяется в соответствие с количеством приемных антенн. Так как в работе у пользователей по 2 антенны, размерность сформированных сигналов symbol1 и symbol2 изменяется с 2 x N на 2 x N/2 – переменные x1, x2. После чего они конкатенируются в один сигнал – x.

Создайте канальную матрицу для каждого пользователя. Выполните сингулярное разложение канальных матриц и осуществите прекодирование сигналов по методу Block Diagonalization.

```
%% ================================ Precoding
% Channel Function Generator and Precoding
H1 = (randn(NR,NT)+1i*randn(NR,NT)); % 1 user channel
H2 = (randn(NR,NT)+1i*randn(NR,NT)); % 2 user channel

[U1,S1,V1] = svd(H1);
[U2,S2,V2] = svd(H2);

W2 = V1(:,3:4);
W1 = V2(:,3:4);

Tx_Data = W1*x(1:2,:) + W2*x(3:4,:);
```

Переменные W2 и W1 – это матрицы прекодирования. W2 перемножается на сигнал x1, а W1 на x2. Далее эти произведения суммируются – переменная Tx_Data. Полученный сигнал представляет собой сумму прекодированных сигналов двух пользователей. Сигнал содержит 4 потока, следовательно, сигнал передается одновременно со всех 4 передающих антенн. Сигнал принятый первыми пользователем будет умножен на канальную матрицу первого пользователя, соответственно сигнал, принятый вторым пользователем будет умножен на канальную матрицу второго пользователя:

```
%% ================================ Rx signal
% Transmitter and Receiver
Rx1     = awgn(H1*Tx_Data, SNRdB,'measured');
Rx2     = awgn(H2*Tx_Data, SNRdB,'measured');
```

Согласно формулам из лекций, принятый сигнал будет содержать в себе как канальную матрицу, так и матрицу прекодирования. Следовательно, оценка канала должна учитывать матрицу прекодирования.

Выполните оценку канальной матрицы используя правые сингулярные вектора, после чего восстановите переданные сигналы, выполните демодуляцию и посчитайте количество ошибок. Перед выполнением демодуляции не забудьте, что сигнал второго пользователя сдвинут на 45 градусов. Устраните этот сдвиг.

```
%% ================================ Channel estimation
```

```
W1_H1   = H1*W1;
W2_H2   = H2*W2;

EQ1     = W1_H1'*inv(...); % Equalizer for the 1st user
EQ2     = W2_H2'*inv(...); % Equalizer for the 2nd user

y       = [EQ1*...; EQ2*...];

symbol_hat1 = reshape(y(1:2,:),[],1).';
symbol_hat2 = reshape(y(3:4,:),[],1).';
symbol_hat2_recover = symbol_hat2.*exp(-1i*45*pi/180);
demapped1 = qamdemod(symbol_hat1.', 4, 'OutputType', 'bit').';
demapped2 = qamdemod(symbol_hat2_recover.',4,'OutputType','bit').'
N_ebits1 = sum( msg_bit1~=demapped1, 'all');
N_ebits2 = sum( msg_bit2~=demapped2, 'all');
fprintf('Num first user Errors: %d\n', N_ebits1);
fprintf('Num second user Errors: %d\n', N_ebits2)
```

Переменные W1_H1 и W2_H2 представляют собой оценку канала, с учетом прекодирующих матриц. EQ1 и EQ2 – восстановленные сигналы. Так как перед отправкой было выполнено изменение размерности сигналов, требуется вернуть им исходный размер – переменные symbol_hat1, symbol_hat2. Число ошибок каждого пользователя должно быть равно нулю. Постройте диаграммы созвездия сигналов обоих пользователей:

```
%% ================================ Plot constellation
figure(1); grid on; hold on;
plot( symbol_hat1, 'bx');
plot( symbol_hat2, 'r+');
legend('First user', 'Second user');
```

На рисунке 1.1 представлены восстановленные сигналы обоих пользователей.

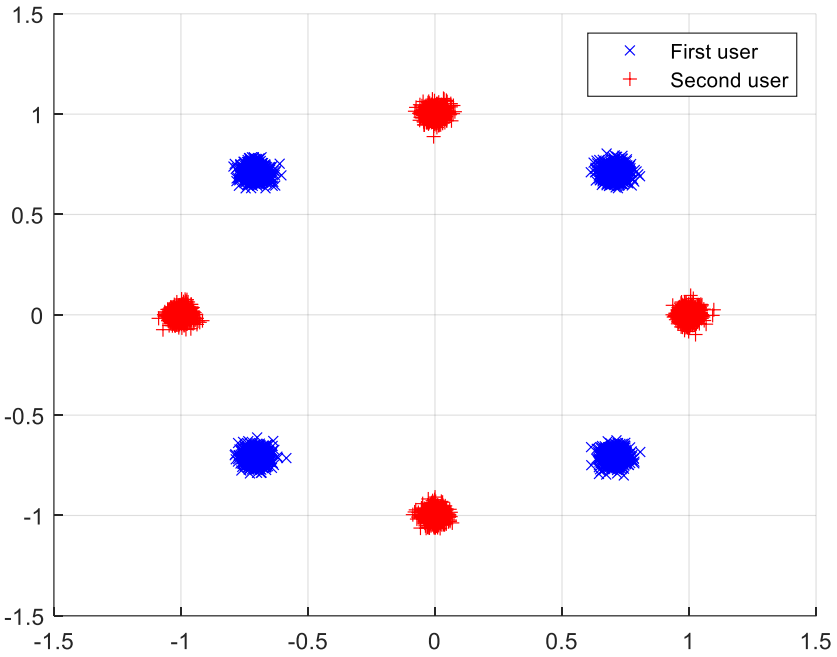

Рисунок 1.1 – Диаграммы созвездия обоих пользователей

Из графиков видно, что сигналы пользователей были успешно выделены из передаваемой суммы сигналов и приняты без искажений.

2. Контрольные вопросы

1. Для чего используется MU MIMO?
2. Чем отличается система MU MIMO 4 на 4 от классической MIMO 4 на 4?
3. Возможно ли повысить скорость передачи с помощью MU MIMO?

Работа № 7

«Формирование модели системы беспроводной связи для конфигурации Malti User Massive MIMO»

Цель работы: Ознакомление с принципами оценки канала и передачи сигналов в MU – mMIMO системе.

Задачи лабораторной работы:

- Сформировать MIMO канал с учетом затухания в свободном пространстве и направления распространения сигналов;
- Выполнить формирование сигнала для оценки MIMO канала;
- Выполнить оценки MIMO канала по методам LS и MMSE.

1. Теоретический материал

Самыми эффективными антенными системами в настоящее время являются MU – mMIMO системы. Для эффективного использования всех преимуществ, обеспечиваемых применением данных систем, необходимо знание канальной матрицы. Одним из способов получить оценку канальной матрицы является ортогональное кодирование символов преамбулы. В MU – mMIMO из-за наличия пространственной корреляции, для получения оценки канала необходимо знать корреляционную матрицу.

В данной работе будет рассмотрена полноценная система передачи данных, представленная на рисунке 1.1. Сперва пользователи передают символы преамбулы БС. В БС выполняется оценка канала, после чего она используется для прекодирования пилотных и информационных, которые БС в свою очередь передает пользователю.

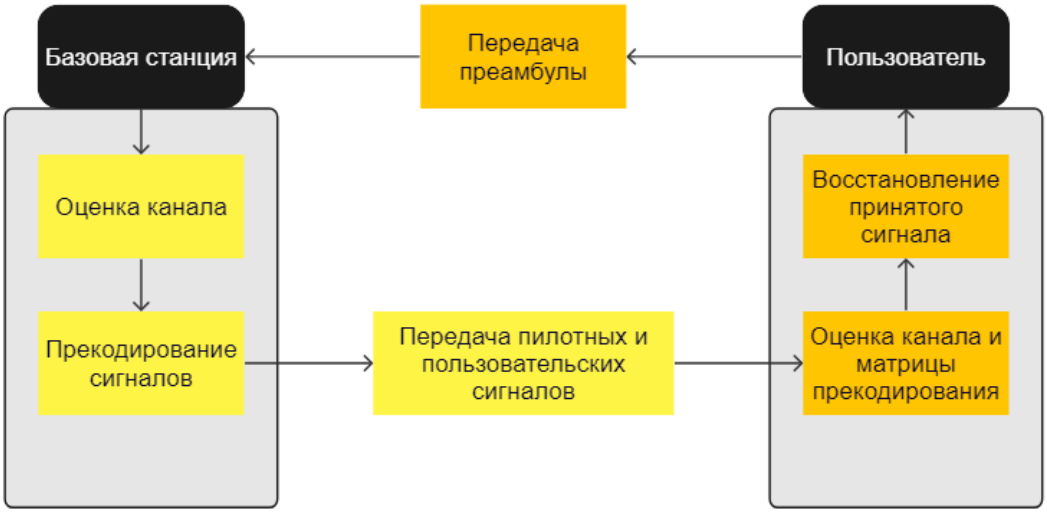

Рисунок 1.1 – Схема проекта

2. Ход выполнения работы

Установите следующие параметры модели.

```
clc
clear all
close all

%% ================================== Переменные
```

```
K = 4; % Число пользователей
M = 100; % Число передающих антенн
eyeM = eye(M);
distances = randi([35 500],K,1); % Расстояние между приемником и передатчиком
theta_r = randi([0,360],K,1)*pi/180; % Угол между пользователями и базовой станцией
ASDdeg = 10;
ASD = ASDdeg*pi/180; % Отклонение угла прихода
antennaSpacing = 1/2; % Расстояние между антеннами
bitsPerSCar = 4; % Количество бит на символ модуляции
B = 20e6; % Полоса сигнала
noiseFigure = 7; % Коэффициент шума
constantTerm = -35.3; % Ослабление распространения
sigma_sf = 10; % Коэффициент затухания
alpha = 3.76; % Постоянная потери распространения
FFT = 256; % Размер Фурье преобразования
nDCar = 242; % Количество информационных поднесущих
nCarI = [1:7 129 256-5:256]'; % Положение защитных интервалов
CarLoc = setdiff((1:FFT)',sort(nCarI)); % Положение информационных поднесущих
modMode = 2^bitsPerSCar; % Индекс модуляции
```

Основными параметрами являются характеристики канала связи:
Сформируйте преамбулу согласно стандарту 802.11n.

```
%% ================================= Преамбула
ltfLeft = [1; 1;-1;-1; 1; 1;-1; 1; -1; 1; 1; 1; ...
          1; 1; 1;-1; -1; 1; 1;-1; 1;-1; 1; 1; 1; 1];
ltfRight = [1; -1;-1; 1; 1; -1; 1;-1; 1; -1;-1;-1;-1; ...
          -1; 1; 1;-1; -1; 1;-1; 1; -1; 1; 1; 1; 1];

ltfSC = [zeros(7,1); ltfLeft; 1; ltfRight;-1;-1;-1; 1; 1;-1; 1; ...
        -1; 1; 1;-1; ltfLeft; 1; ltfRight; 1;-1; 1;-1; 0; ...
         1;-1;-1; 1; ltfLeft; 1; ltfRight;-1;-1;-1; 1; 1;-1; 1; ...
        -1; 1; 1;-1; ltfLeft; 1; ltfRight; zeros(6,1)];
```

Преамбулы всех 4 пользователей будут передаваться одновременно, поэтому, чтобы оценить канал каждого отдельного пользователя, необходимо в приемнике разделить эти 4 преамбулы. Для этого преамбулы кодируются ортогональными кодами. Сформируйте матрицу ортогонального кодирования.

```
%% ================================= Ортогональное кодирование
if floor(log2(K)) == log2(K)
   preamMatrix = hadamard(K);
else
   w = exp(-1i*2*pi/K);
   preamMatrix = zeros(K,K);
   for i = 1:K
      for j = 1:K
         preamMatrix(i,j) = w^((i-1)*(j-1));
      end
   end
```

```
end
```

Сформируйте корреляционную матрицу, канальную матрицу, шум, вычислите значение шума на входе приемника и обратную корреляционную матрицу:

```
%% ================================ Канал
% Вычисление корреляционной матрицы
firstRow = zeros(M,1);
R = zeros(M,M,K);
for theta = 1:length(theta_r)
   for column = 1:M
      distance = antennaSpacing*(column-1);
      F = @(Delta)exp(1i*2*pi*distance*sin(theta_r(theta)+ Delta)).*exp(-
Delta.^2/(2*ASD^2))/(sqrt(2*pi)*ASD);
      firstRow(column) = integral(F,-20*ASD,20*ASD);
   end
   R(:,:,theta) = toeplitz(firstRow);
end

% Канальная матрицы
h = zeros(M,K,FFT);
h(:,:,1) = 1*randn(M,K);
h(:,:,3) = 0.5*randn(M,K);
h(:,:,5) = 0.2*randn(M,K);
for i = 1:K
   for j = 1:M
      H(j,i,:) = fft(h(j,i,:));
   end
end

% Вектор шума
Np = sqrt(0.5)*(randn(M,K,FFT)+1i*randn(M,K,FFT));

% Ослабление в канале
betas = zeros(K,1);
channelGaindB = constantTerm - alpha*10*log10(distances) + sigma_sf*randn(1,1);
noiseVariancedBm = -174 + 10*log10(B) + noiseFigure;
channelGainOverNoise = channelGaindB - noiseVariancedBm;

normF = FFT/nDCar;
ltfTx = complex(zeros(FFT,K));
y_f = complex(zeros(FFT,K,K));
PsiInv = zeros(M,M,K);
Rpsi = zeros(M,M,K);
for i = 1:K
   ltfTx(CarLoc,:) = bsxfun(@times,ltfSC(CarLoc),preamMatrix(:,i)');
   betas(i) = 10^(channelGainOverNoise(i)/10);
   y_f(:,:,i) = ltfTx*normF;
   R(:,:,i) = betas(i)*R(:,:,i);
   Rsqrt = sqrtm(R(:,:,i));
```

```
    H(:,i,:) = Rsqrt*squeeze(H(:,i,:));
    PsiInv(:,:,i) = (K*R(:,:,i)+betas(i)*eyeM);
    Rpsi(:,:,i) = R(:,:,i) / PsiInv(:,:,i);
end
```

Переменная R – это корреляционная матрица, H – канальная матрица, channelGainOverNoise – шумовая полка в приемнике, y_f – принятые символы преамбулы, Rpsi –корреляционная матрица, используемая для оценки канала по методу MMSE. Перемножьте канальную матрицу с переданными символами преамбулы:

```
%% ================================ Принятый опорный сигнал
y_f_re = permute(y_f,[2 3 1]);
rec = pagemtimes(H, y_f_re) + 0.001*Np;
```

Перемножьте принятые символы с ортогональной матрицей для выполнения декодирования. Выполните оценку канальной матрицы используя методы LS и MMSE.

```
%% ================================ Оценка канальной матрицы
rec_w = pagemtimes(rec,preamMatrix);

% LS
Hhat_LS = rec_w/K;

% MMSE
rec_w_re = permute(rec_w,[1 3 2]);
rec_MMSE = pagemtimes(Rpsi,rec_w_re);

Hhat = zeros(M,K,FFT);
for i =1:K
    for j = 1:M
        Hhat(j,i,:) = ltfSC.'.*squeeze(rec_MMSE(j,:,i));
    end
end
```

После получения оценки, она вновь умножается на переданную преамбулу. Так как преамбула содержит в себе -1, то фаза оценки канала в этих местах будет инвертирована по знаку. Для устранения этого эффекта, оценка повторно умножается на переданную преамбулу. Переменные Hhat_LS и rec_MMSE – оценки, выполненные по методам LS и MMSE соответственно. Hhat – оценка канала по методу MMSE с устранением инвертирования фазы.

Постройте графики модуля канальной матрицы и оценки канальной матрицы, полученные разными методами, пример представлен на рисунке 2.1.

```
figure(1); hold on;
plot(abs(squeeze(H(1,2,:))),'b','LineWidth',2)
plot(abs(squeeze(Hhat_LS(1,2,:))),'k')
plot(abs(squeeze(rec_MMSE(1,:,2))),'c+-')
plot(abs(squeeze(Hhat(1,2,:))),'r','LineWidth',1)
legend('Канальная матрица', ...
```

```
    'LS оценка канала', ...
    'MMSE оценка канала с инвертированной фазой',...
    'MMSE оценка канала без инвертированной фазы')
```

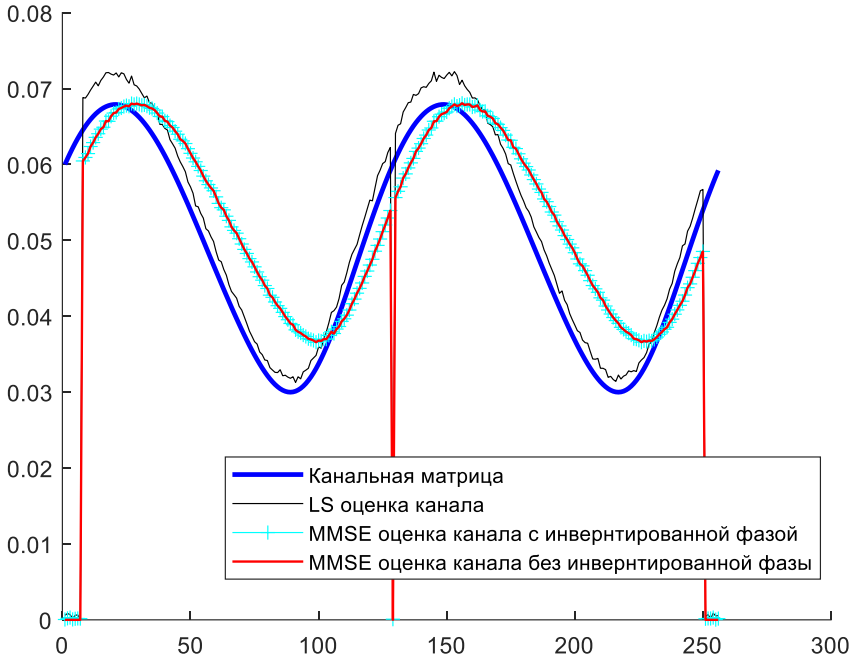

Рисунок 2.1 – Модуль канальной матрицы и оценок канальной матрицы

Модуль оценки до повторного умножения на преамбулу и после полностью совпадают, а также все оценки повторяют по форме канальную матрицу. Чтобы увидеть влияние на фазу, постройте график мнимой части канальной матрицы и оценок, пример представлен на рисунке 2.2.

```
figure(2); hold on;
plot(imag(squeeze(H(1,2,:))),'b','LineWidth',3)
plot(imag(squeeze(Hhat_LS(1,2,:))),'k','LineWidth',2)
plot(imag(squeeze(rec_MMSE(1,:,2))),'c')
plot(imag(squeeze(Hhat(1,2,:))),'r','LineWidth',1)
legend('Канальная матрица', ...
    'LS оценка канала', ...
    'MMSE оценка канала с инвертированной фазой',...
    'MMSE оценка канала без инвертированной фазы')
```

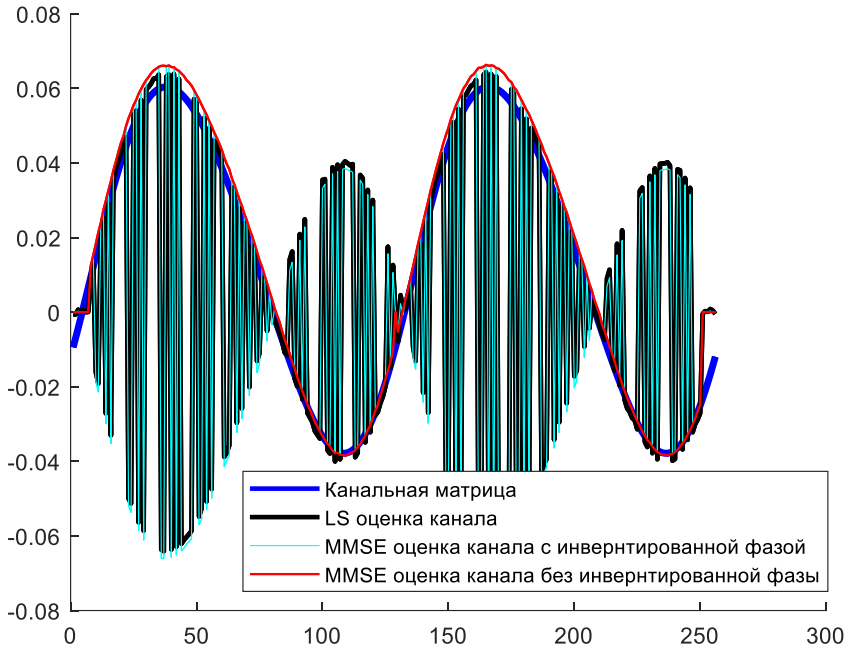

Рисунок 2.2 – Мнимая часть канальной матрицы и оценок канальной матрицы

В оценке, не умноженной повторно на переданную преамбулу, видны частые инвертирования фазы, что будет оказывать негативное влияние при восстановлении сигнала. При умножении оценки на переданную преамбулу, подобные изменения фазы полностью устраняются.

Сформируйте пилотные сигналы и сигналы с полезной информацией. Измените размерность сигналов в соответствие с количеством пользователей.

```
%% ================================ Формирование полезных сигналов
DSymbole = zeros(K,FFT);
PSymbole = zeros(K,FFT);

data = randi([0 modMode-1], K,length(CarLoc));
pilot = randi([0 modMode-1], K,length(CarLoc));

data_mod = qammod(data,modMode);
pilot_mod = qammod (pilot,modMode);

DSymbole(:,CarLoc) = data_mod;
PSymbole(:,CarLoc) = pilot_mod;

DSymbole = reshape(DSymbole,[K,1,FFT]);
PSymbole = reshape(PSymbole,[K,1,FFT]);
```

Выполните прекодирование, используя полученную оценку канальной матрицы. Для получения матрицы прекодирования по методу MMSE необходимо вычислить корреляционную матрицу ошибки оценки канальной матрицы – переменная C.

```
%% ================================ MMSE Прекодинг
% Матрица ошибки
C_MMSE = R - K*pagemtimes(Rpsi,R);
```

```
C = sum(C_MMSE,3);

% Матрица прекодинга
V_MMSE = pagemrdivide(conj(pagetranspose(Hhat)),...
   (pagemtimes(Hhat,conj(pagetranspose(Hhat)))) + C + norm(betas)*eye(M));
% w_MMSE = conj(pagetranspose( V_MMSE ));
w_MMSE = conj(pagetranspose(pagemrdivide(V_MMSE,pagenorm(V_MMSE))));

% Прекодированные сигналы
preCodedPilot_MMSE = pagemtimes(w_MMSE,PSymbole);
preCodedData_MMSE = pagemtimes(w_MMSE,DSymbole);
```

Принятые сигналы будут иметь следующий вид.

```
%% ================================ Принятые сигналы
Ns = 0.001*sqrt(0.5)*(randn(K,FFT)+1i*randn(K,FFT));
YP_MMSE=squeeze(pagemtimes(conj(pagetranspose(H)),preCodedPilot_MMSE)) + Ns;
YD_MMSE=squeeze(pagemtimes(conj(pagetranspose(H)),preCodedData_MMSE)) + Ns;
```

Выполните оценку канальной матрицы используя пилотные сигналы, а затем восстановите полезные сигналы по методу LS и MMSE. Постройте график созвездия и выведите в консоль количество ошибок до восстановления сигналов и после для разных методов эквалайзирования.

```
%% ================================ Восстановление сигналов
H_eq = YP_MMSE./squeeze(PSymbole);
Y_MMSE_eq = YD_MMSE./H_eq;

figure(3); hold on;
plot( YD_MMSE(1,:), 'r*')
plot( Y_MMSE_eq(1,:), 'b*')
legend('Принятый сигнал','Принятый восстановленный сигнал')

YD_MMSE(isnan(YD_MMSE)) = 0;
data_r_MMSE = qamdemod(YD_MMSE, modMode);
er_MMSE = biterr(data, data_r_MMSE(:,CarLoc));
er_MMSE_u = zeros(1,K);
for i =1:K
   er_MMSE_u(i) = biterr(data(i,:),data_r_MMSE(i,CarLoc));
end
fprintf('Количество ошибок до восстановления: %d; %d; %d; %d\n', er_MMSE_u);

Y_MMSE_eq(isnan(Y_MMSE_eq)) = 0;
data_r_MMSE_eq = qamdemod(Y_MMSE_eq,modMode);
er_MMSE_eq = biterr(data , data_r_MMSE_eq(:,CarLoc));
er_MMSE_eq_u = zeros(1,K);
for i =1:K
   er_MMSE_eq_u(i) = biterr(data(i,:),data_r_MMSE_eq(i,CarLoc));
```

```
end
fprintf('Количество ошибок после восстановления: %d; %d; %d; %d\n', er_MMSE_eq_u);
```

Пример диаграммы созвездия принятого и восстановленного сигнала представлен на рисунке 2.3.

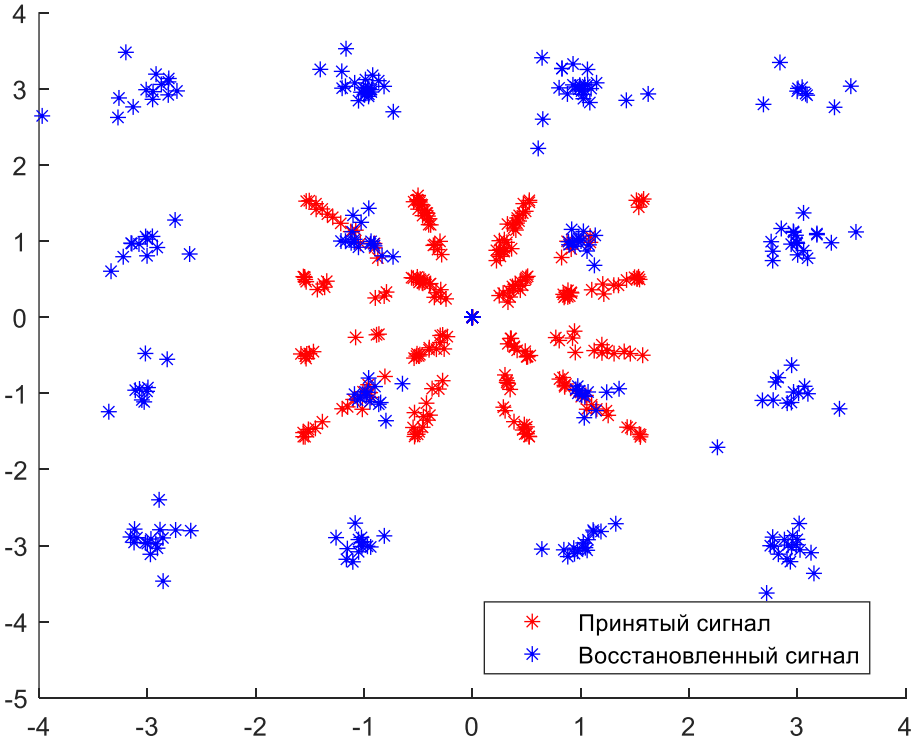

Рисунок 2.3 – Диаграмма созвездия принятого и восстановленного сигнала

3. Контрольные вопросы к работе

1. В чем принципиальное отличие MU – mMIMO от MU – MIMO?
2. Почему в MU – mMIMO наблюдается пространственная корреляция элементов канальной матрицы?
3. Какие вы видите отличия при выполнении прекодирования в MU – mMIMO и MU – MIMO?

ЗАКЛЮЧЕНИЕ

При успешном освоении курса студенты будут уметь применять знания основ пространственной обработки сигналов при разработке и построении беспроводных систем передачи данных, а также владеть основными методами пространственного разделения каналов в режиме многопользовательского доступа и пространственного мультиплексирования для систем связи на основе MIMO.

СПИСОК ЛИТЕРАТУРЫ

1. Björnson E. Introduction to Multiple Antenna Communications and Reconfigurable Surfaces / E. Björnson, Ö. T. Demir. – Hanover : now Publishers Inc, 2024. – 679 p.

2. Бакулин, М.Г. Технология MIMO: принципы и алгоритмы / М.Г. Бакулин, Л.А. Варукина, В.Б. Крейнделин. – Москва : Горячая линия-Телеком, 2014. – 244 с.

3. Clerckx, B. MIMO wireless networks: channels, techniques and standards for multi-antenna, multi-user and multi-cell systems / B. Clerckx B, C. Oestges. : Academic Press, 2013. – 1541 p.

4. Paulraj, A. Introduction to space-time wireless communications / A. Paulraj, R. Nabar, D. Gore. – Cambridge : Cambridge university press, 2003. – 308 p.

5. Brown, T. Practical guide to MIMO radio channel: With MATLAB examples. – Hoboken : John Wiley & Sons, 2012. – 284 p.

6. Björnson E. Massive MIMO networks: Spectral, energy, and hardware efficiency / E. Björnson, J. Hoydis, L. Sanguinetti // Foundations and Trends® in Signal Processing. – 2017. – Т. 11. – №. 3-4. – С. 154-655.

7. Fundamentals of massive MIMO / T. L. Marzetta, E.G. Larsson, H. Yang, H.Q. Ngo. – Cambridge : Cambridge University Press, 2016. – 220 p.