Федеральное государственное бюджетное образовательное учреждение высшего образования «Сибирский государственный университет телекоммуникаций и информатики»

Лабораторная работа №4

**по дисциплине «Основы систем мобильной связи»**

**«Изучение корреляционных свойств последовательностей, используемых для синхронизации в сетях мобильной связи»**

Выполнил: Шаповал Н.О. Группа: ИА-232

Проверил: Дроздова В.Г.

Вариант: 16 GitHub: [https://github.com/](https://github.com/IvanNoritsin/OSMS)nikin1/osms



Новосибирск 2024

**Содержание**

[Цель работы 3](#_bookmark0)

[Краткие теоретические сведения 4](#_bookmark1)

[Этапы выполнения работы 10](#_bookmark2)

[Контрольные вопросы 18](#_bookmark3)

[Заключение 20](#_bookmark4)

## Цель работы

Получить представление о том, какие существуют псевдослучайные двоичные последовательности, какими корреляционными свойствами они обладают и как используются для синхронизации приемников и передатчиков в сетях мобильной связи.

**Задачи работы:**

* Написать программу на языке С/С++ для генерации последовательности Голда;
* Вывести получившуюся последовательность на экран;
* Сделать поэлементный циклический сдвиг последовательности и посчитать автокорреляцию исходной последовательности и сдвинутой;
* Сформировать таблицу с битовыми значениями последовательностей, в последнем столбце которой будет вычисленное значение автокорреляции;
* Сформировать еще одну последовательность Голда, используя свою схему, такую, что x=x+1, а y=у-5;
* Вычислить значение взаимной корреляции исходной и новой последовательностей;
* Проделать предыдущие шаги в Matlab, используя функции xcorr() и autocorr() для вычисления соответствующих корреляций;
* Выведите на график в Matlab функцию автокорреляции в зависимости от величины задержки (lag).

# Краткие теоретические сведения

### Псевдослучайные двоичные последовательности

Псевдослучайные двоичные последовательности (PN-sequences – PseudoNoise) – это частный случай псевдослучайных последовательностей, элементами которой являются только 2 возможных значения (1 и 0 или -1 и

+1). Такие последовательности очень часто используются в сетях мобильной связи. Возможные области применения:

* оценка вероятности битовой ошибки (BER – Bit Error Rate). В этом случае передатчик передает приемнику заранее известную PNпоследовательность бит, а приемник анализируя значения бит на конкретных позициях, вычисляет количество искаженных бит и вероятность битовой ошибки в текущих радиоусловиях, что затем может быть использовано для работы алгоритмов, обеспечивающих помехозащищенность системы;
* временная синхронизация между приемником и передатчиком.

Включаясь, абонентский терминал начинает записывать сигнал, дискретизируя его с требуемой частотой, в результате чего формируется массив временных отсчетов и требуется понять, начиная с какого элемента в этом массиве собственно содержатся какие-либо данные, как именно структурирована ось времени, где начинаются временные слоты. Используя заранее известную синхронизирующую PN-последовательность (синхросигнал), приемник сравнивает полученный сигнал с этой последовательностью на предмет «сходства» - корреляции. И если фиксируется корреляционный пик, то на стороне приема можно корректно разметить буфер с отсчетами на символы, слоты, кадры и пр.

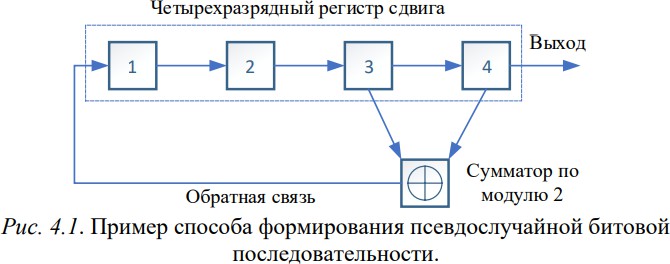
* расширение спектра. Используется для повышения эффективности передачи информации с помощью модулированных сигналов через канал с сильными линейными искажениями (замираниями), делая систему устойчивой к узкополосным помехам (например, в 3G WCDMA).

Псевдослучайная битовая последовательность должна обладать следующими свойствами, чтобы казаться почти случайной:

1. Сбалансированность (balance), то есть число единиц и число нулей на любом интервале последовательности должно отличаться не более чем на одну.
2. Цикличность. Циклом в данном случае является последовательность бит с одинаковыми значениями. В каждом фрагменте псевдослучайной битовой последовательности примерно половину составляли циклы длиной 1, одну четверть – длиной 2, одну восьмую – длиной 3 и т.д.
3. Корреляция. Корреляция оригинальной битовой последовательности с ее сдвинутой копией должна быть минимальной. Автокорреляция этих последовательностей – это практически дельта-функция во временной области, как для аддитивного белого гауссовский шума AWGN (Additive white Gaussian noise), а в частотной области – это константа.

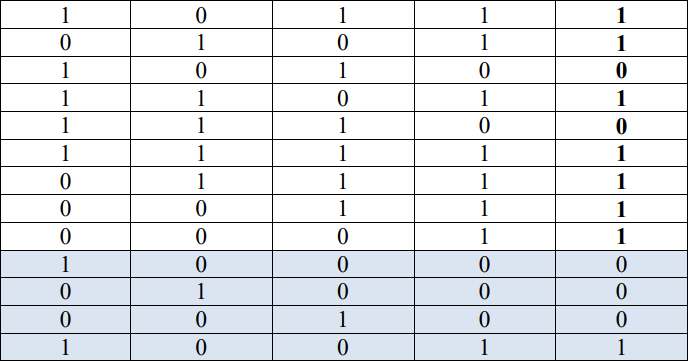
Как можно сгенерировать последовательность, обладающую вышеперечисленными свойствами?

Для этого можно использовать, например, линейный четырехразрядный регистр сдвига с обратной связью, сумматора по модулю 2 и контуром обратной связи со входом регистра [3]. Работа регистра тактируется синхроимпульсами и с каждым новым тактом осуществляется сдвиг битовой последовательности вправо, а содержимое регистров 3 и 4 суммируется по модулю два, при этом результат суммирования подается на вход регистра 1, как показано на рисунке 4.1



Рассмотрим пример формирования псевдослучайной битовой последовательности с помощью схемы, показанной на рисунке 4.1, при условии, что регистр проинициализирован последовательностью 1 0 0 0. На каждом такте эта последовательность будет сдвигаться на одну позицию вправо, при этом на выходе будут появляться биты псевдослучайной последовательности. В таблице 4.1 показаны состояния разрядов регистра на каждом такте и выходные биты.

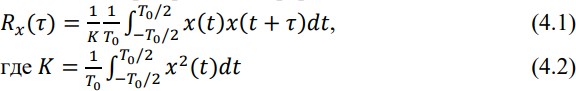




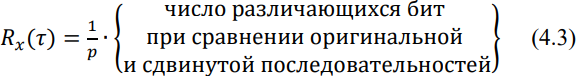
На выход всегда идут биты из 4-го разряда регистра. Очевидно, что длина полученной последовательности равна 2 m -1=15 – максимальное число различных состояний нашего регистра, где m=4 – число разрядов в сдвиговом регистре, используемом для формирования последовательности, а затем, начиная с 16-го бита, значения на выходе начинают циклически повторяться. Такие последовательности еще называются m- последовательностями (от англ.слова maximum – последовательности максимальной длины). Важно заметить, что инициализирующая битовая последовательность (или полином) не может быть нулевой, так как из всех нулей невозможно создать последовательность, содержащую единицы, данным способом.

Проанализируем последовательность, полученную в таблице 4.1 с точки зрения наличия свойств псевдослучайных битовых последовательностей:

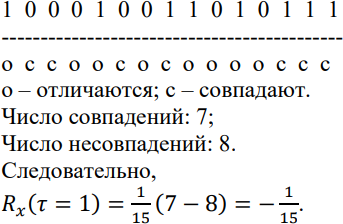
1. Сбалансированность: 8 единиц и 7 нулей.
2. Цикличность: нет циклов длиннее 4х (1 цикл из 4-х единиц, 1 цикл из 3-х нулей, 2 цикла из нулей и единиц, и 4 цикла длиной, равной одному).
3. Корреляция: автокорреляционная функция периодического сигнала x(t) с периодом T0 в нормированной форме (4.1) - (4.2)



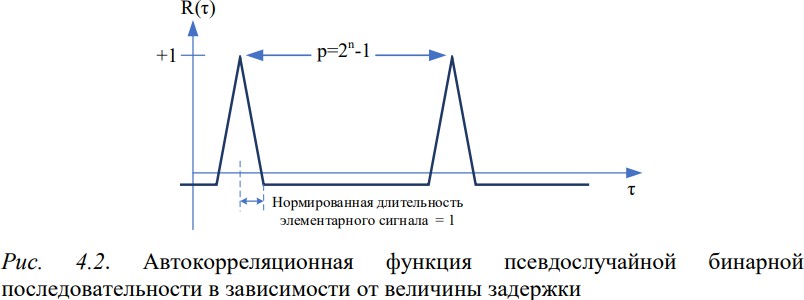
Нормированная автокорреляционная функция псевдослучайного сигнала с длительностью символа, равной единице и периодом p=2m -1 может быть определена как (4.3):



Для примера, определим значение автокорреляции последовательности из таблицы 4.1 со сдвигом на 1 элемент.



Автокорреляция для любого сдвига будет равна -1/15, и лишь в момент полного совпадения всех элементов будет наблюдаться пик корреляционной функции 𝑅𝑥 (𝜏 = 0) = +1. На рисунке 4.2 показана автокорреляционная функция псевдослучайной бинарной последовательности.



Чем длиннее последовательность, тем выше пик ее автокорреляционной функции, и тем больше напоминает дельта-функцию. Такого типа автокорреляцией характеризуется и белый гауссовский шум, поэтому в англоязычной литературе такие последовательности называют pseudo noise sequences.

Чем острее автокорреляционный пик (то есть чем длиннее последовательность), тем удобней использовать данные последовательности для решения проблем синхронизации в сетях мобильной связи.

Действительно, абонентский терминал при начальном включении должен засинхронизировать начало своих временных слотов на временной оси приемника и передатчика. Поэтому обычно базовые станции периодически отправляют специальные синхронизирующие последовательности, в качестве которых часто используются именно m-последовательности, и терминал вычисляет автокорреляцию этой заранее известной последовательности с полученным записанным сигналом, и в тот момент, когда фиксируется автокорреляционный пик, абонент отмечает начало слота на своей оси времени (а точнее номер отсчета в буфере, начиная с которого идет передаваемый базовой станцией слот с данными).

Стоит отметить, что даже в случае наличия ошибок в принятой синхропоследовательности, возникших вследствие помех, присутствующих в канале связи, приемник все равно достаточно легко обнаружит явный корреляционный пик.

На рисунке 4.3 представлены варианты реализации схемы синхронизации с помощью последовательного и параллельного поиска

Разновидности псевдо-шумовых битовых последовательностей

M-последовательности – не единственные PN-последовательности, используемые в системах мобильной связи. Существуют также коды Баркера, коды Голда, коды Касами, коды Уолша-Адамара.

Коды Голда формируются путем суммирования по модулю 2 двух Mпоследовательностей одинаковой длины. Коды Касами также формируются из М-последовательностей путем взятия периодических выборок из этих последовательностей и суммированием их по модулю два. Данные коды обладают очень хорошими взаимокорреляционными свойствами.

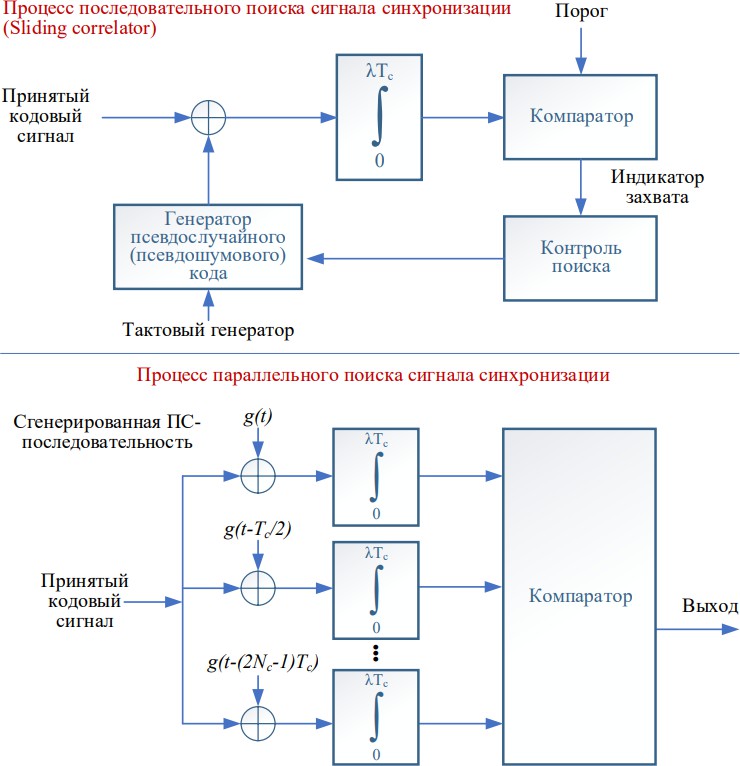
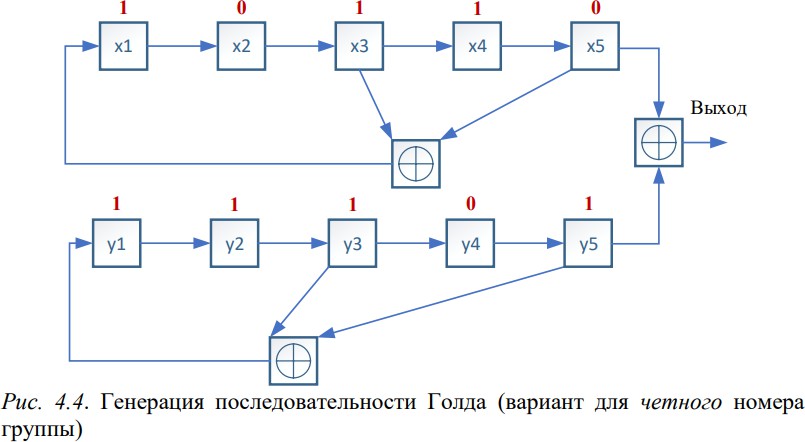


Рис. 4.3. Синхронизация с помощью последовательного и параллельного поиска

# Этапы выполнения работы

### Исходные данные:



x = 16 = 10000

y = 23 = 10111

1. Сначала требуется написать программу на языке C/С++ для генерации последовательности Голда, используя схему, изображенную на рисунке 4.4. Для выполнения задания будем использовать язык C++. Создадим функцию create\_gold\_sequence, которая будет принимать в себя два вектора:

vector<int> create\_gold\_sequence(vector<int>& x, vector<int>& y){ vector<int> gold\_sequence;

int xor\_shift\_x, xor\_shift\_y; int last\_bit\_x, last\_bit\_y;

for (int i = 0; i < LENGTH\_SEQUENCE; i++){

xor\_shift\_x = x[2] ^ x[4]; xor\_shift\_y = y[2] ^ y[4];

gold\_sequence.push\_back(x.back() ^ y.back()); x.pop\_back();

y.pop\_back();

x.insert(x.begin(), xor\_shift\_x);

y.insert(y.begin(), xor\_shift\_y);

}

return gold\_sequence;

}

Функция составляет последовательность Голда согласно схеме изображѐнной на рисунке 4.4.

1. Создадим два вектора, которые соответствуют значениям x и y и вызывем функцию создания последовательности Голда.

vector<int> x = {0, 1, 1, 0, 0};

vector<int> y = {1, 0, 0, 1, 1};

cout << "Последовательность Голда: ";

vector<int> gold\_sequence = create\_gold\_sequence(x, y); for (int i : gold\_sequence){

cout << i;

}

cout << endl;

Получаем следующий результат:



1. Далее необходимо сделать поэлементный сдвиг последовательности и посчитать автокорреляцию исходной последовательности и сдвинутой. Реализуем соответствующие функции.

Функция циклического сдвига последовательности:

vector<int> shift\_sequence(vector<int>& original\_sequence, int shift){ int N = original\_sequence.size();

vector<int> shifted\_sequence(N);

for (int i = 0; i < N; i++){

shifted\_sequence[i] = original\_sequence[(i - shift + N) % N];

}

return shifted\_sequence;

}

Функция автокорреляции последовательности:

double autoCorrelation(vector<int>& x, vector<int>& y){ int equal = 0, different = 0;

double auto\_corr;

for (int i = 0; i < LENGTH\_SEQUENCE; i++){

if (x[i] == y[i]){ equal += 1;

}

else{

different += 1;

}

}

auto\_corr = (1.0 / LENGTH\_SEQUENCE) \* (equal - different); return auto\_corr;

}

Сформируем таблицу с битовыми значениями последовательности, в последнем столбце которой будет находиться результат вычисления автокорреляции. Для этого также напишем функцию, на вход которой будет поступать последовательность:

void print\_table(vector<int>& sequence){ int N = sequence.size();

cout << "Сдвиг\tПоследовательность\t\t\tАвтокорреляция" << endl; for (int i = 0; i < N + 1; i++){

vector<int> shifted\_sequence = shift\_sequence(sequence, i); double corr = autoCorrelation(sequence, shifted\_sequence); cout << i << "\t";

for (int i : shifted\_sequence){ cout << i << " ";

}

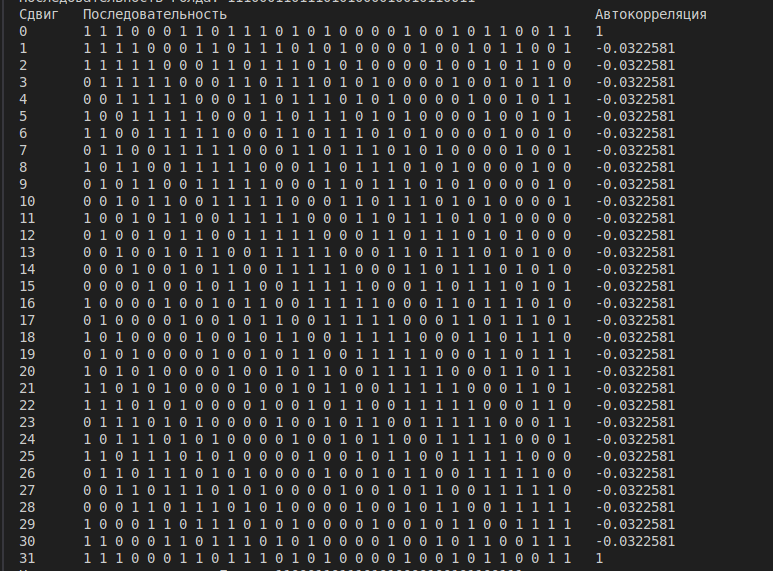
cout << "\t";

cout << corr << endl;

}

}

После вызова функции получаем следующий результат:



1. Сформируем ещѐ одну последовательность Голда, где x = 17 (10001) и y = 18 (10010).



1. Вычислим взаимную корреляцию исходной и новой последовательности. Для этого воспользуемся функцией для расчѐта нормализованной корреляции из лабораторной работы 3:

double NormalizedCorrelation(vector<int>& x, vector<int>& y){ int N = x.size();

double corr = 0; double sumXY = 0; double sumX2 = 0;

double sumY2 = 0;

for (int i = 0; i < N; i++){ sumXY += x[i] \* y[i]; sumX2 += x[i] \* x[i]; sumY2 += y[i] \* y[i];

}

corr = sumXY / sqrt(sumX2 \* sumY2); return corr;

}

Получим результат:



1. Следующим шагом является повторение всех действий, но только в среде Matlab.

Функция для формирования последовательности Голда:

Сформированная последовательность Голда для исходных данных:

function gold\_sequence = create\_gold\_sequence(x, y, length\_sequence) gold\_sequence = zeros(1, length\_sequence);

for i = 1:length\_sequence xor\_shift\_x = bitxor(x(3), x(5)); xor\_shift\_y = bitxor(y(3), y(5));

gold\_sequence(i) = bitxor(x(end), y(end)); x = [xor\_shift\_x, x(1:end-1)];

y = [xor\_shift\_y, y(1:end-1)];

end

end



Далее сформируем таблицу с битовыми значениями последовательности и результатом вычисления автокорреляции. Для вычисления автокорреляции используется функция autocorr, которая сразу вернѐт значения задержки и результат автокорреляции. Для отображения циклического сдвига последовательности используется функция circshift().

function print\_table(sequence)

N = length(sequence);

fprintf('Сдвиг | Последовательность Автокорреляция\n');

|

fprintf('

\n');

|

|-

[corr, lags] = autocorr(sequence, 'NumLags', N - 1);

for i = 1:N

shifted\_sequence = circshift(sequence, i - 1);

fprintf('%5d | ', lags(i)); fprintf('%d ', shifted\_sequence); fprintf('| %10.4f\n', corr(i));

end

fprintf('\n'); figure;

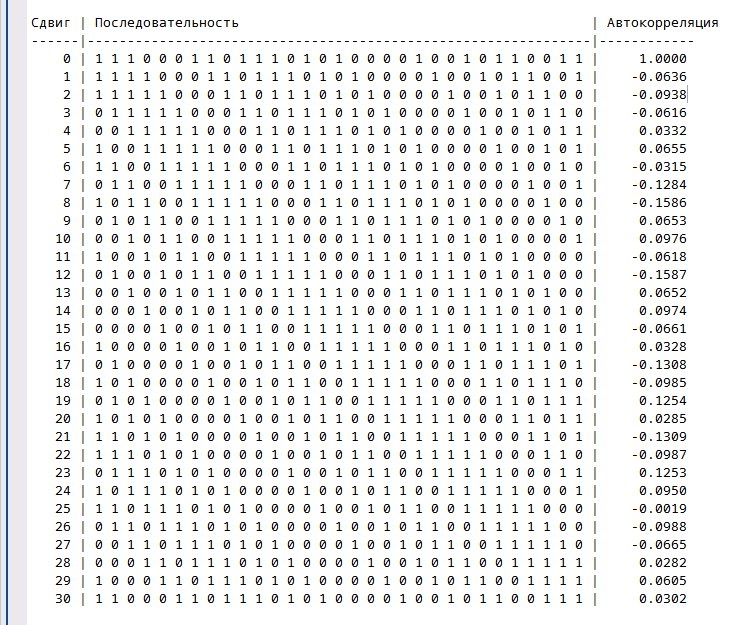
plot(0:N - 1, corr, '-o'); xlabel('Задержка (lag)'); ylabel('Автокорреляция');

title('Функция автокорреляции в зависимости от задержки'); xlim([0 N])

grid on;

end

В итоге выводится таблица:



Как можно заметить, результат вычисления автокорреляции в Matlab не сходится с результатом на C++. Это связано с тем, что мы использовали функцию autocorr(), которая вычисляет автокорреляцию по другой формуле, нежели мы используем при расчѐте на C++.

Сформируем новую последовательность Голда и вычислим взаимную корреляцию новой и исходной последовательности при помощи функции xcorr():

x = [0, 1, 1, 0, 1];

y = [0, 1, 1, 1, 0];

new\_gold\_sequence = create\_gold\_sequence(x, y, length\_sequence); fprintf('Новая последовательность Голда: ');

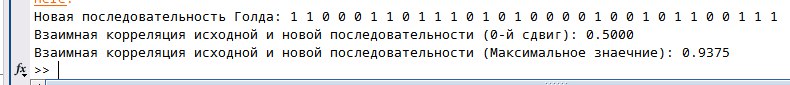
fprintf('%d ', new\_gold\_sequence); fprintf('\n');

[corr, lags] = xcorr(gold\_sequence, new\_gold\_sequence, 'normalized'); fprintf('Взаимная корреляция исходной и новой последовательности (0-й сдвиг):

%.4f\n', corr(length\_sequence));

fprintf('Взаимная корреляция исходной и новой последовательности (Максимальное знаечние): %.4f\n', max(corr));

Так как xcorr() возвращает взаимные корреляции, сдвигая последовательности, выведем результат на 0-м сдвиге (что считалось на C++) и максимально возможную корреляцию:



Как видно, результат на 0-м сдвиге абсолютно идентичен результату, полученному на C++.

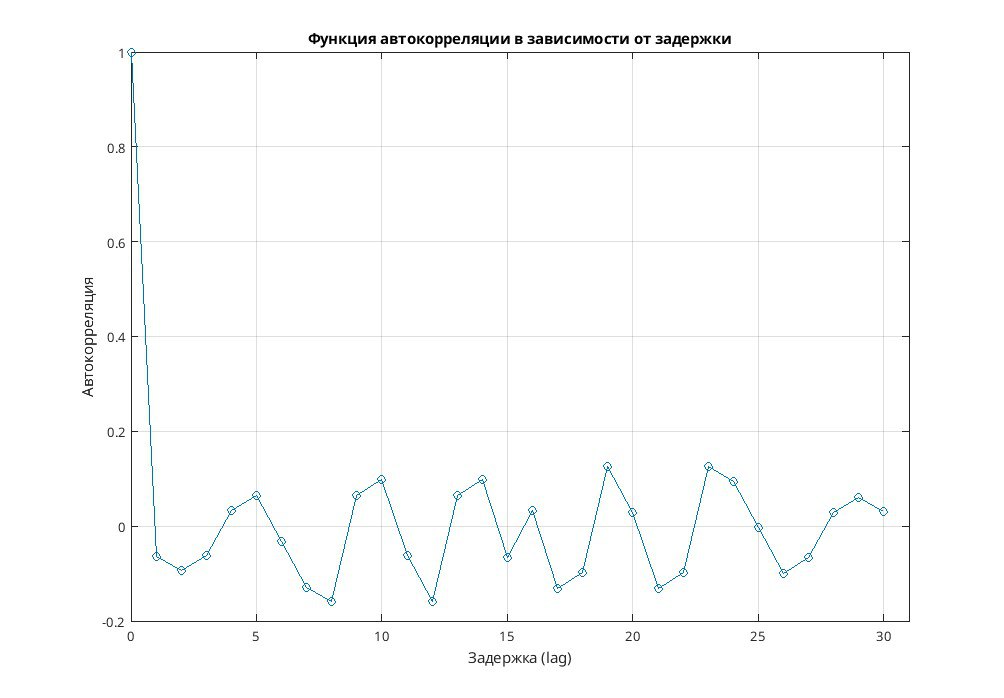
1. Выведем на график в Matlab функцию автокорреляции в зависимости от величины задержки (lag):

figure;

plot(0:N - 1, corr, '-o'); xlabel('Задержка (lag)'); ylabel('Автокорреляция');

title('Функция автокорреляции в зависимости от задержки'); xlim([0 N])

grid on;



# Контрольные вопросы

1. **Для чего в мобильных сетях могут использоваться псевдослучайные последовательности?**

Возможные области применения:

* + оценка вероятности битовой ошибки (BER – Bit Error Rate). В этом случае передатчик передает приемнику заранее известную PNпоследовательность бит, а приемник анализируя значения бит на конкретных позициях, вычисляет количество искаженных бит и вероятность битовой ошибки в текущих радиоусловиях, что затем может быть использовано для работы алгоритмов, обеспечивающих помехозащищенность системы;
  + временная синхронизация между приемником и передатчиком.

Включаясь, абонентский терминал начинает записывать сигнал, дискретизируя его с требуемой частотой, в результате чего формируется массив временных отсчетов и требуется понять, начиная с какого элемента в этом массиве собственно содержатся какие-либо данные, как именно структурирована ось времени, где начинаются временные слоты. Используя заранее известную синхронизирующую PN-последовательность (синхросигнал), приемник сравнивает полученный сигнал с этой последовательностью на предмет «сходства» - корреляции. И если фиксируется корреляционный пик, то на стороне приема можно корректно разметить буфер с отсчетами на символы, слоты, кадры и пр.

* + расширение спектра. Используется для повышения эффективности передачи информации с помощью модулированных сигналов через канал с сильными линейными искажениями (замираниями), делая систему устойчивой к узкополосным помехам (например, в 3G WCDMA).

1. **Что значит положительная корреляция сигналов?**

Положительная корреляция сигналов означает, что при увеличении амплитуды одного сигнала увеличивается амплитуда второго сигнала, и наоборот. Это свидетельствует о схожести их временных характеристик и позволяет предполагать, что сигналы имеют общую или взаимозависимую природу.

1. **Что такое корреляционный прием сигналов?**
2. **Что такое корреляционный прием сигналов?**

Корреляционный прием — это метод, который использует свойства корреляции для определения наличия или отсутствия полезного сигнала в принятом шумовом сигнале. Применяя к принимаемому сигналу корреляционную функцию, можно выделить моменты, когда входной сигнал совпадает с эталоном, что позволяет извлекать полезный сигнал из шума.

1. **Как вычисление корреляционных функций помогает синхронизироваться приемнику и передатчику в сетях мобильной связи?**

Вычисление корреляционных функций помогает приемнику распознавать начало и конец передаваемого сигнала, сравнивая его с известным шаблоном. В сетях мобильной связи это необходимо для синхронизации тактовой частоты передатчика и приемника. Используя корреляционные функции, приемник может точно определить временное положение сигнала, что улучшает точность временной синхронизации между узлами сети и обеспечивает стабильность связи.

1. **Какими свойствами обладают псевдослучайные битовые последовательности?**
   1. Сбалансированность (balance), то есть число единиц и число нулей на любом интервале последовательности должно отличаться не более чем на одну.
   2. Цикличность. Циклом в данном случае является последовательность бит с одинаковыми значениями. В каждом фрагменте псевдослучайной битовой последовательности примерно половину составляли циклы длиной 1, одну четверть – длиной 2, одну восьмую – длиной 3 и т.д.
   3. Корреляция. Корреляция оригинальной битовой последовательности с ее сдвинутой копией должна быть минимальной. Автокорреляция этих последовательностей – это практически дельта-функция во временной области, как для аддитивного белого гауссовский шума AWGN (Additive white Gaussian noise), а в частотной области – это константа.
2. **Какие разновидности PN-последовательностей вам известны?**

M-последовательности, коды Баркера, коды Голда, коды Касами, коды Уолша-Адамара.

# Заключение

В этой лабораторной работе мы получили представление о том, какие существуют псевдослучайные двоичные последовательности, какими корреляционными свойствами они обладают и как используются для синхронизации приемников и передатчиков в сетях мобильной связи.

Мы разработали функции на языке программирования C++ для создания последовательности Голда и, используя циклический сдвиг, провели автокорреляцию этой последовательности.

Также мы воспользовались MATLAB для сравнения результатов работы встроенной функции автокорреляции с функцией, которая рассчитывает автокорреляцию по специальной формуле. Кроме того, мы построили график зависимости автокорреляции от величины задержки.