ГУАП

КАФЕДРА № 44

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

Старший преподаватель		Е.К.Григорьев
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ПРОЕКТЕ

МОДЕЛИРОВАНИЕ ФУНКЦИЙ НЕОПРЕДЕЛЁННОСТИ КОДОВЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ ДЛЯ ФАЗОВОЙ МОДУЛЯЦИИ СИГНАЛОВ

по курсу: МОДЕЛИРОВАНИЕ

РАБОТУ ВЫПОЛНИЛИ СТУДЕНТЫ ГР. №	4143		Е.Д.Тегай
		подпись, дата	инициалы, фамилия

Д.В.Пономарев
инициалы, фамилия

А.М.Гридин
инициалы, фамилия

Санкт-Петербург 2024

Теоретические сведения

Код (последовательность) Баркера – такая специальная последовательность двоичных символов (битов), которая используется в цифровой связи и обработке сигналов для улучшения качества передачи данных. Такие коды имеют определённые свойства, благодаря которым они обладают минимальными автокорреляционными боковыми лепестками. Длины последовательностей ограничены. Известны лишь длины из 2, 3, 4, 5, 7, 11 и 13 битов. Последняя длина, собственно, и будет рассмотрена далее.

Под автокорреляцией понимается математическая операция, которая используется для измерения сходства сигнала с его собственной копией, смещённой по времени. Проще говоря, это способ определить, насколько хорошо сигнал «коррелирует» с самим собой при различных временных задержках.

Лепестки – значения автокорреляционной функции, полученные на ненулевых задержках. Главный лепесток (пик) находится на нулевой задержке, его значение является максимальным. В идеальном случае, для хорошей кодовой последовательности, боковые лепестки автокорреляции должны быть как можно меньшими, чтобы минимизировать ложные срабатывания и помехи. Коды Баркера специально сконструированы так, чтобы иметь минимальные боковые лепестки автокорреляции.

В рамках данной работы необходимо провести моделирование функции неопределённости и взаимной неопределённости для кода Баркера и инвертированного кода Баркера длины 13.

Поэтому заключительным этапом данного блока будет подробное рассмотрение понятия функций неопределённости и взаимной неопределённости.

Функция неопределённости, также известная как функция неопределённости Уигнера-Вилла, используется в обработке сигналов для

описания, насколько хорошо можно одновременно локализовать сигнал по времени и частоте. Она помогает понять, насколько хорошо можно определить частоту сигнала в конкретный момент времени.

Отсюда идёт простой вывод о том, что чем точнее мы знаем частоту сигнала в конкретный момент времени, тем менее точно мы можем определить момент времени, и наоборот. Можно заметить сходство с аналогичным принципом в физике: точное определение положения частицы ухудшает знание об её импульсе.

Соответственно, основными характеристиками являются время и частота. Принцип неопределённости гласит, что существует ограничение на точность, с которой можно одновременно определить время и частоту сигнала.

Функция неопределённости определяется как:

$$A(t, f_d) = \frac{1}{E_x} \int_{-\infty}^{\infty} x(u) x^*(u - t) e^{j2\pi f_d u} du$$

$$E_x = \int_{-\infty}^{\infty} x(u) x^*(u) du$$

где $x^*(u)$ – комплексно-сопряжённая функция сигнала $x(u)$, t – время, f_d – доплеровский сдвиг , j – мнимая единица, $E_X -$ квадратная норма сигнала.

Функция взаимной неопределённости (иногда называемая кросс-функцией неопределённости) – это инструмент для анализа взаимного расположения двух сигналов во времени и частоте. Она показывает, насколько хорошо два сигнала совпадают или коррелируют друг с другом, когда один из них смещён по времени и частоте.

Функция взаимной неопределённости для двух сигналов x(t) и y(t) определяется как:

$$A_{xy}(t, \omega) = \frac{1}{\sqrt{E_x E_y}} \left| \int_{-\infty}^{\infty} x(u) y^*(u - t) e^{j2\pi f_d u} du \right|$$

$$E_x = \int_{-\infty}^{\infty} x(u)x^*(u)du$$

$$E_y = \int_{-\infty}^{\infty} y(u)y^*(u)du$$

где $y^*(u)$ – комплексно-сопряжённая функция сигнала $y(u)$, t – время, f_d – доплеровский сдвиг , j – мнимая единица, $E_{X,}E_y$ – квадратная норма сигнала.

Отсюда получаем, что если два сигнала очень похожи, то функция взаимной неопределённости будет иметь высокий пик в точке, где их временные и частотные сдвиги совпадают.

Разработка программы

Для начала необходимо указать в программе вид кода Баркера и инвертированного кода Баркера. Следует отметить, что инвертированный код Баркера строился по принципу «-1 заменяется на 1», и наоборот. Итого получаем:

Код Баркера: [1 1 1 1 1 − 1 − 1 1 1 − 1 1 − 1 1]

Инвертированный код Баркера: [-1 -1 -1 -1 -1 1 1 -1 -1 1-1 1-1]

Затем вычисляется функция неопределённости для обычного и инвертированного кода Баркера с помощью функции ***ambiguity_code.*** Также вычисляется функция взаимной неопределённости для этой пары сигналов с помощью функции ***cross_ambiguity_code.***

После этого создаются массивы частот, которые соответствуют осям частоты на графиках функции неопределённости и функции взаимной неопределённости. Для вычисления функции взаимной неопределённости в коде также обозначаются переменные, содержащие значения длин для каждого из кодов.

Далее обозначается блок переменных, которые отвечают за количество выборок интерполяции. Интерполяция важная для того, чтобы обеспечить достаточное разрешение при вычислении функции неопределённости и взаимной неопределённости с помощью быстрого преобразования Фурье (или FFT).

Затем идёт определение ближайшего большего значения, которое является степенью двойки для количества выборок, используемых в быстром преобразовании Фурье. Выбор размера FFT, равного степени двойки, значительно ускоряет вычисления, так как алгоритмы FTT оптимизированы для таких размеров.

После этого идёт определение диапазона задержек для каждого из кодов. Затем идёт блок кода, отвечающего за построение графиков для инвертированного и обычного кода Баркера, а также для функции взаимной неопределённости.

Далее идут вычисления самой функции неопределённости. Суть вычисления такова: сначала «подготавливаются» данные, а именно – увеличивается количество выборок исходного кода, чтобы улучшить разрешение функции неопределённости. Для этого исходный код повторяется с интерполяцией, чтобы достичь нужного количества выборок, которое округляется до следующей степени двойки для оптимизации FFT. После создаются массивы для временных задержек и частотных смещений и вычисляется FFT для интерполированного кода. Далее идёт блок с циклом по частотам: для каждой частоты вычисляется фазовый сдвиг, применяется полученный сдвиг к интерполированному коду, вычисляется FFT для фазосдвинутого сигнала и умножаются результаты FFT фазосдвинутого сигнала и исходного сигнала, а после выполняется обратное FFT. Результаты обратного FFT дают значения функции неопределённости для соответствующей частоты и задержки. Эти значения нормируются и сдвигаются для получения окончательной функции неопределённости.

Завершающим блоком является определение функции ***plot_figures_chap_6***, которая предназначена для визуализации функции неопределённости в виде различных графиков. На вход она принимает матрицу значений функции неопределённости, массив значений задержки, массив значений частоты, а также строку, содержащую заголовок для графиков.

Искомая функция строит 3 типа графиков: трёхмерную плоскость, нормализованный разрез и контурный график. Рассмотрим более подробно каждый из них.

Трёхмерная поверхность. Создаётся график трёхмерной поверхности, который показывает значения функции неопределённости в зависимости от задержки и частоты. Значения функции неопределённости нормализуются путём деления на максимальное значение. Параметры отображения устанавливаются для удобного просмотра, а именно: угол обзора, метки осей и заголовок графика.

Нормализованный разрез. Создаётся график разреза функции неопределённости при нулевой частоте. Выбирается центральная строка из матрицы функции неопределённости, которая соответствует нулевой частоте. Значения также нормализуются, и график строится по оси задержек, устанавливаются метки осей, сетка и заголовок.

Контурный график. Создаётся контурный график, который отображает изолинии функции неопределённости. Значения нормализуются, график строится в зависимости от задержки и частоты. Устанавливаются метки осей, сетка и заголовок.

Помимо вышеупомянутой функции также в коде программы определена функция ***ambiguity_code***, которая вычисляет функция неопределённости заданного кода. Суть её работы такова: сначала идёт инициализация данных, а именно – вычисляется длина входного кода и количество выборок для

интерполяции, увеличенное в 10 раз для улучшения разрешения. Определяется ближайшая степень двойки для количества выборок, чтобы оптимизировать вычисления с помощью FFT. Инициализируются массивы для интерполированного кода и временной задержки. Затем исходный код интерполируется путём повторения каждого элемента 10 раз, создавая новый массив с увеличенным числом выборок, а после вычисляется FFT для интерполированного кода. После этого определяется цикл по частотам, где для каждой частоты создаётся фазовый сдвиг, этот сдвиг применяется к интерполированному коду, вычисляется FFT для фазосдвинутого сигнала, а затем умножаются результаты FFT фазосдвинутого сигнала и исходного сигнала, и выполняется обратное FFT. Результаты обратного FFT нормализуются и сдвигаются, чтобы получить окончательную матрицу функции неопределённости.

Листинг кода

Основной файл project.m

```
clc
close all
clear all

% Входные данные - обычный код Баркера длины 13
uinput_barker = [+1 +1 +1 +1 +1 -1 -1 +1 +1 -1 +1 -1 +1];
% Входные данные - инвертированный код
uinput_inverted = -uinput_barker;

% Вычисление функции неопределенности для инвертированного кода
[ambig_inverted] = ambiguity_code1(uinput_inverted);
% Вычисление функции неопределенности для обычного кода Баркера
[ambig_barker] = ambiguity_code(uinput_barker);
% Вычисление функции взаимной неопределенности
[ambig_cross] = cross_ambiguity_code(uinput_barker, uinput_inverted);

% Частота
freq_inverted = linspace(-6, 6, size(ambig_inverted, 1));
freq_barker = linspace(-6, 6, size(ambig_barker, 1));
freq_cross = linspace(-6, 6, size(ambig_cross, 1));
```

```
% Длина кода
N_inverted = length(uinput_inverted);
N_barker = length(uinput_barker);
N_cross = length(uinput_barker); % Оба кода одинаковой длины

% Количество выборок для интерполяции
samp_num_inverted = N_inverted * 10;
samp_num_barker = N_barker * 10;
samp_num_cross = N_cross * 10;

% Следующая степень двойки для FFT
n_inverted = ceil(log(samp_num_inverted) / log(2));
n_barker = ceil(log(samp_num_barker) / log(2));
n_cross = ceil(log(samp_num_cross) / log(2));
nfft_inverted = 2^n_inverted;
nfft_barker = 2^n_barker;
nfft_cross = 2^n_cross;

% Диапазон задержек
delay_inverted = linspace(-N_inverted, N_inverted, nfft_inverted);
delay_barker = linspace(-N_barker, N_barker, nfft_barker);
delay_cross = linspace(-N_cross, N_cross, nfft_cross);

% Построение графиков для инвертированного кода
plot_figures_chap6(ambig_inverted, delay_inverted, freq_inverted,
'Inverted Barker Code');
% Построение графиков для обычного кода Баркера
plot_figures_chap6(ambig_barker, delay_barker, freq_barker, 'Barker
Code');
% Построение графиков для функции взаимной неопределенности
plot_figures_chap6(ambig_cross, delay_cross, freq_cross, 'Cross Ambiguity
Function');

function [ambig] = cross_ambiguity_code(uinput_barker, uinput_inverted)
N = length(uinput_barker);
tau = N;
code1 = uinput_barker;
code2 = uinput_inverted;
samp_num = length(code1) * 10;
n = ceil(log(samp_num) / log(2));
nfft = 2^n;
u1 = zeros(1, nfft);
u2 = zeros(1, nfft);
```

```
j = 0;
for index = 1:10:samp_num
j = j + 1;
u1(index:index+9) = code1(j);
u2(index:index+9) = code2(j);
end
delay1 = linspace(0, 5*tau, nfft); % Задержки для первого кода
freq_del2 = 12 / tau / 100; % Частота для второго кода
j = 0;
u1fft = fft(u1, nfft);
ambig = zeros(length(-6/tau:freq_del2:6/tau), nfft); % Инициализация переменной ambig
for freq = -6/tau:freq_del2:6/tau
j = j + 1;
exf = exp(1i * 2 * pi * freq * delay1); % Используем задержки и частоты для первого кода
u2_times_exf = u2 .* exf;
u2fft = fft(u2_times_exf, nfft);
prod = u1fft .* conj(u2fft);
ambig(j, :) = fftshift(abs(ifft(prod)));
end
end


% Функция построения графиков
function plot_figures_chap6(ambig, delay, freq, title_str)
figure;
mesh(delay, freq, (ambig / max(max(ambig))));
view(-30, 55);
axis tight;
ylabel('Frequency');
xlabel('Delay');
zlabel('Ambiguity Function');
title(title_str);

figure;
Nhalf_freq = (size(ambig, 1) - 1) / 2;
Nhalf_delay = (size(ambig, 2) - 1) / 2;
plot(freq, ambig(:, ceil(Nhalf_delay) + 1) / max(max(ambig)), 'k');
xlabel('Frequency');
ylabel('Normalized Ambiguity Cut for \tau=0');
grid on;
axis tight;
title([title_str ' - Normalized Ambiguity Cut']);
```

```
figure;
plot(delay, ambig(ceil(Nhalf_freq) + 1, :) / max(max(ambig)), 'r');
xlabel('Delay');
ylabel('Normalized Ambiguity Cut for f=0');
grid on;
axis tight;
title([title_str ' - Normalized Ambiguity Cut']);

figure;
contour(delay, freq, (ambig / max(max(ambig))));
axis tight;
ylabel('Frequency');
xlabel('Delay');
grid on;
title([title_str ' - Contour Plot']);
end
function [ambig] = ambiguity_code(uinput_barker)
% Compute and plot the ambiguity function for any give code u
% Compute the ambiguity function by utilizing the FFT
% through combining multiple range cuts
N = size(uinput_barker,2);
tau = N;
code = uinput_barker;
samp_num = size(code,2) * 10;
n = ceil(log(samp_num) / log(2));
nfft = 2^n;
u(1:nfft) = 0;
j = 0;
for index = 1:10:samp_num
index;
j = j+1;
u(index:index+10-1) = code(j);
end
% set-up the array v
v = u;
delay = linspace(0,5*tau,nfft);
freq_del = 12 / tau /100;
j = 0;
vfft = fft(v,nfft);
for freq = -6/tau:freq_del:6/tau;
j = j+1;
exf = exp(sqrt(-1) * 2. * pi * freq .* delay);
u_times_exf = u .* exf;
```

```
ufft = fft(u_times_exf,nfft);
prod = ufft .* conj(vfft);
ambig(j,:) = fftshift(abs(ifft(prod))');
end
function [ambig] = ambiguity_code1(uinput_inverted)
% Compute and plot the ambiguity function for any give code u
% Compute the ambiguity function by utilizing the FFT
% through combining multiple range cuts
N = size(uinput_inverted,2);
tau = N;
code = uinput_inverted;
samp_num = size(code,2) * 10;
n = ceil(log(samp_num) / log(2));
nfft = 2^n;
u(1:nfft) = 0;
j = 0;
for index = 1:10:samp_num
index;
j = j+1;
u(index:index+10-1) = code(j);
end
% set-up the array v
v = u;
delay = linspace(0,5*tau,nfft);
freq_del = 12 / tau /100;
j = 0;
vfft = fft(v,nfft);
for freq = -6/tau:freq_del:6/tau;
j = j+1;
exf = exp(sqrt(-1) * 2. * pi * freq .* delay);
u_times_exf = u .* exf;
ufft = fft(u_times_exf,nfft);
prod = ufft .* conj(vfft);
ambig(j,:) = fftshift(abs(ifft(prod))');
end
```

Вспомогательный файл ambiguity_code1.m

```
function [ambig] = ambiguity_code1(uinput_inverted)
% Compute and plot the ambiguity function for any give code u
% Compute the ambiguity function by utilizing the FFT
% through combining multiple range cuts
N = size(uinput_inverted,2);
tau = N;
code = uinput_inverted;
```

```
samp_num = size(code,2) * 10;
n = ceil(log(samp_num) / log(2));
nfft = 2^n;
u(1:nfft) = 0;
j = 0;
for index = 1:10:samp_num
index;
j = j+1;
u(index:index+10-1) = code(j);
end
% set-up the array v
v = u;
delay = linspace(0,5*tau,nfft);
freq_del = 12 / tau /100;
j = 0;
vfft = fft(v,nfft);
for freq = -6/tau:freq_del:6/tau;
j = j+1;
exf = exp(sqrt(-1) * 2. * pi * freq .* delay);
u_times_exf = u .* exf;
ufft = fft(u_times_exf,nfft);
prod = ufft .* conj(vfft);
ambig(j,:) = fftshift(abs(ifft(prod))');
end
```

Вспомогательный файл ambiguity_code.m

```
function [ambig] = ambiguity_code(uinput_barker)
% Compute and plot the ambiguity function for any give code u
% Compute the ambiguity function by utilizing the FFT
% through combining multiple range cuts
N = size(uinput_barker,2);
tau = N;
code = uinput_barker;
samp_num = size(code,2) * 10;
n = ceil(log(samp_num) / log(2));
nfft = 2^n;
u(1:nfft) = 0;
j = 0;
for index = 1:10:samp_num
index;
j = j+1;
u(index:index+10-1) = code(j);
end
```

```
% set-up the array v
v = u;
delay = linspace(0,5*tau,nfft);
freq_del = 12 / tau /100;
j = 0;
vfft = fft(v,nfft);
for freq = -6/tau:freq_del:6/tau;
j = j+1;
exf = exp(sqrt(-1) * 2. * pi * freq .* delay);
u_times_exf = u .* exf;
ufft = fft(u_times_exf,nfft);
prod = ufft .* conj(vfft);
ambig(j,:) = fftshift(abs(ifft(prod))');
end
```

Результаты работы программы

Рассмотрим первые полученные графики, построенные относительно обычного кода Баркера. Они показаны на рисунках 1 - 3.

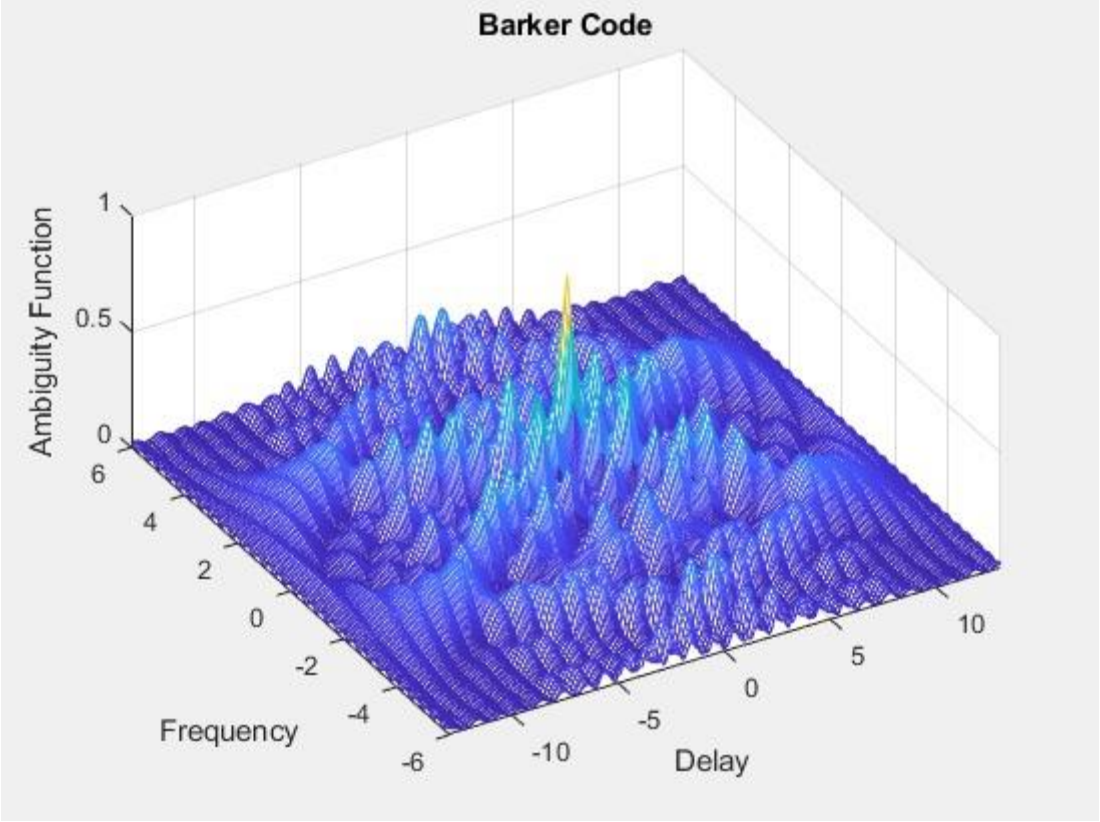

Рисунок 1 – Трёхмерный график кода Баркера

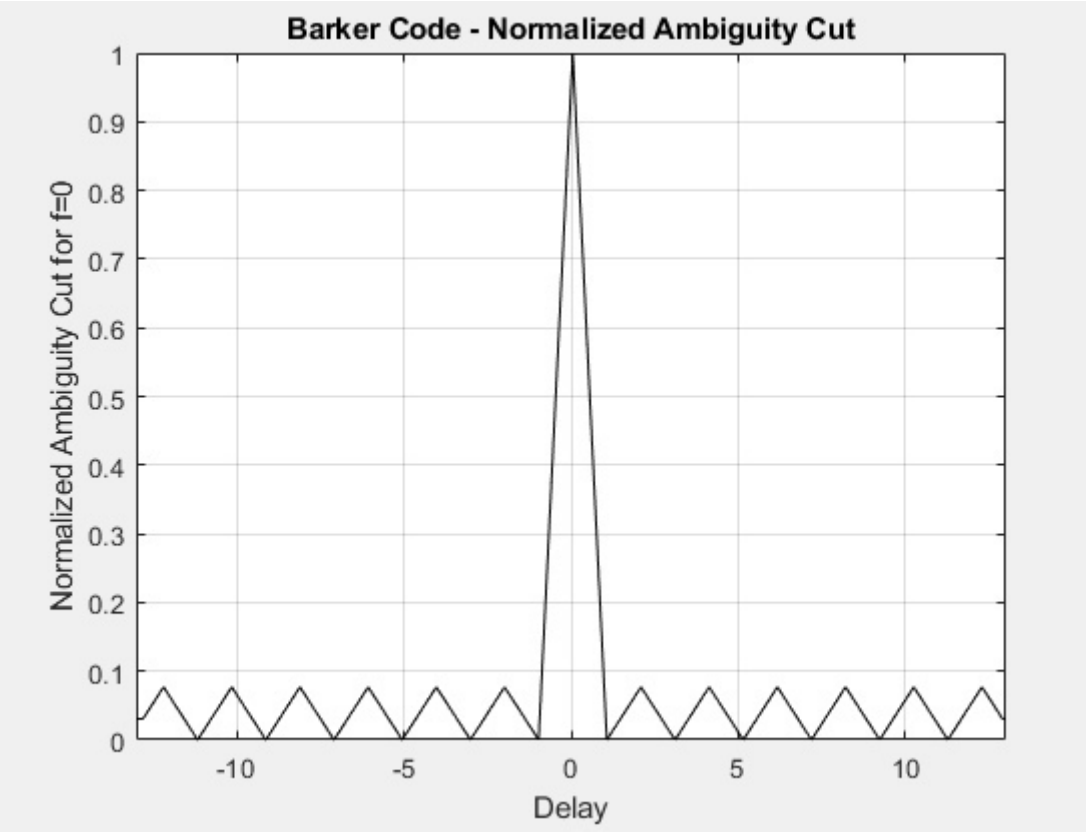

Рисунок 2 – Нормализованный разрез кода Баркера

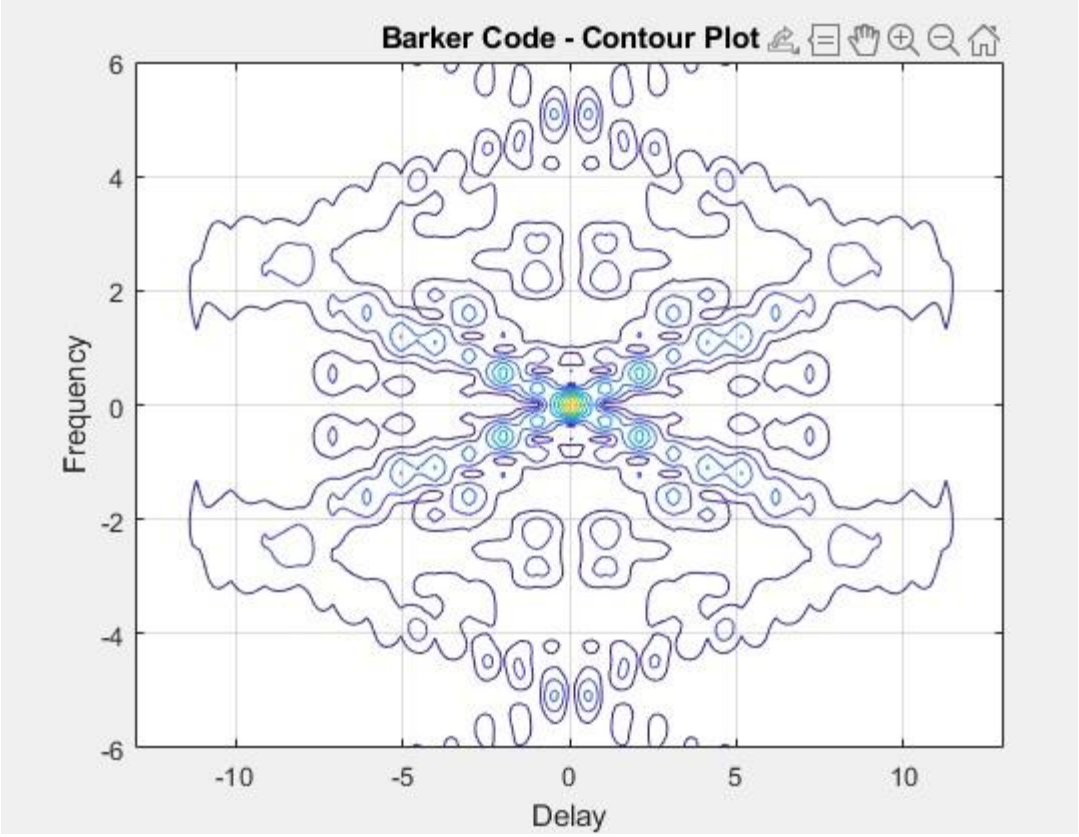

Рисунок 3 – Контурный график кода Баркера

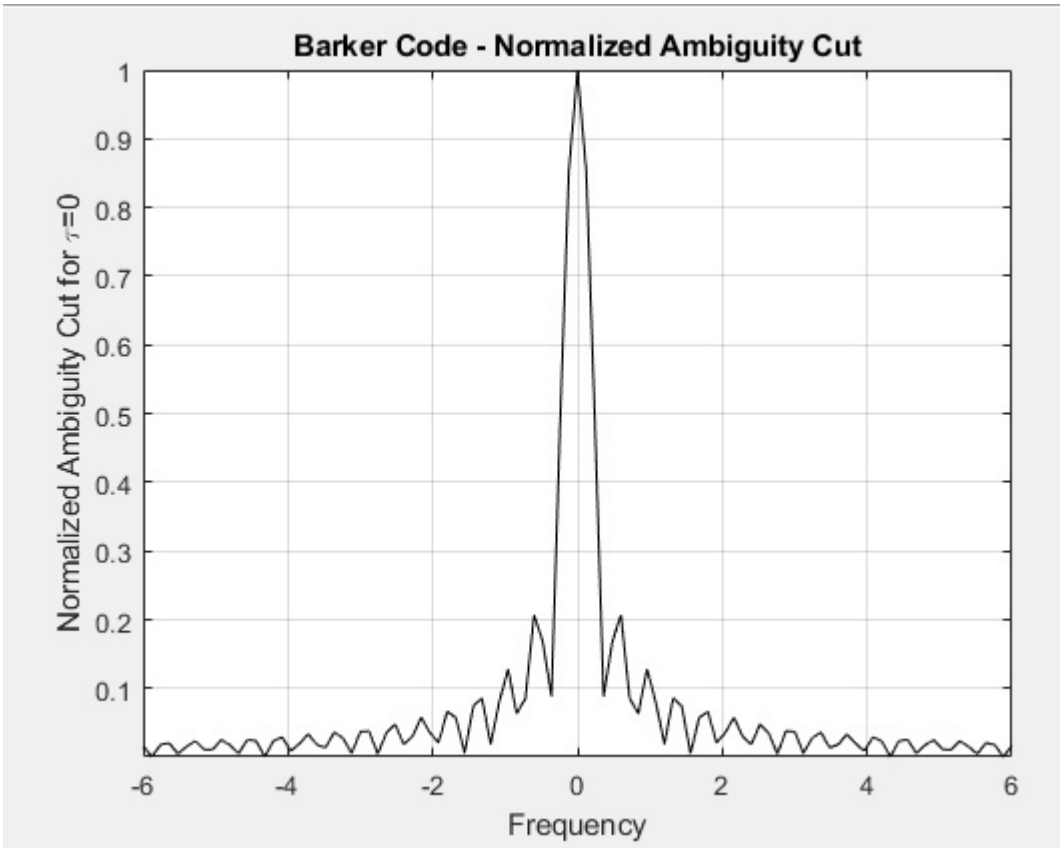

Рисунок 4 – Нормализованный разрез кода Баркера (частотная корреляционная функция)

Также рассмотрим графики из стороннего источника. Они продемонстрированы на рисунках 5 – 7.

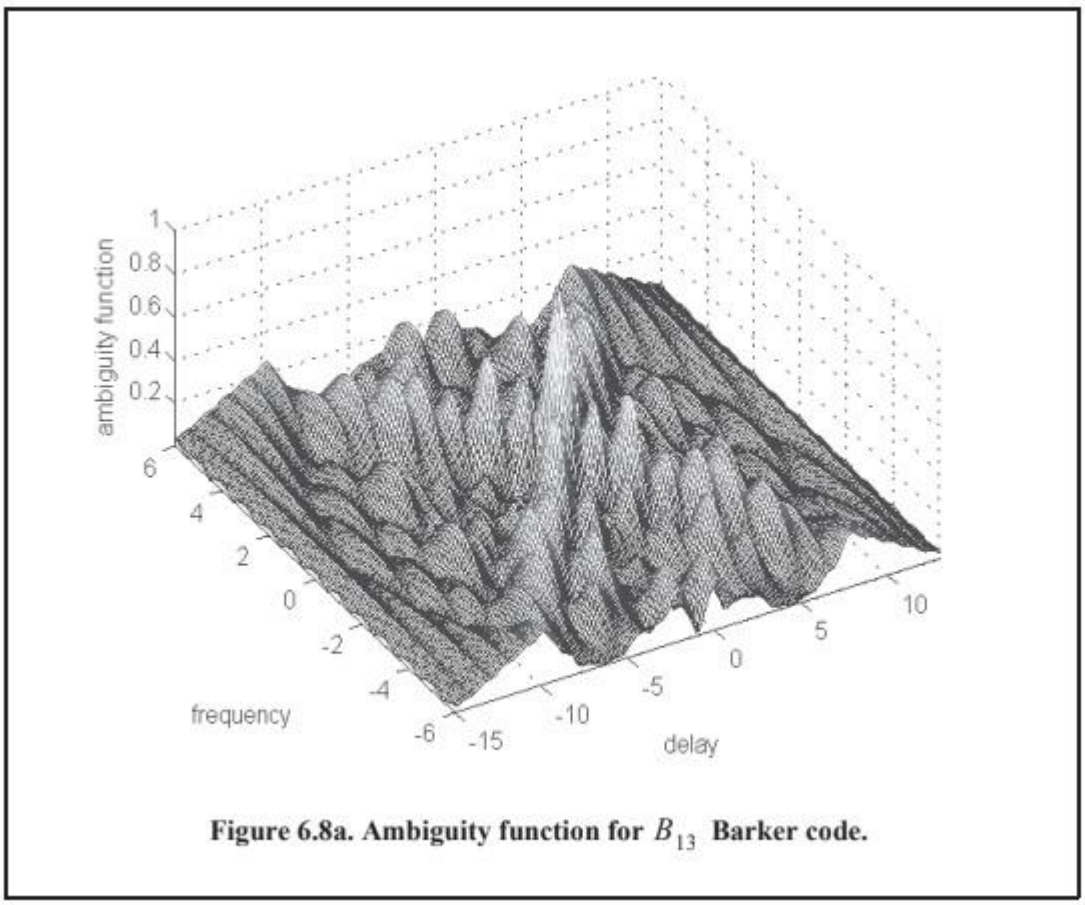

Figure 6.8a. Ambiguity function for B_{13} Barker code.

Рисунок 5 - Трёхмерный график кода Баркера (сторонний источник)

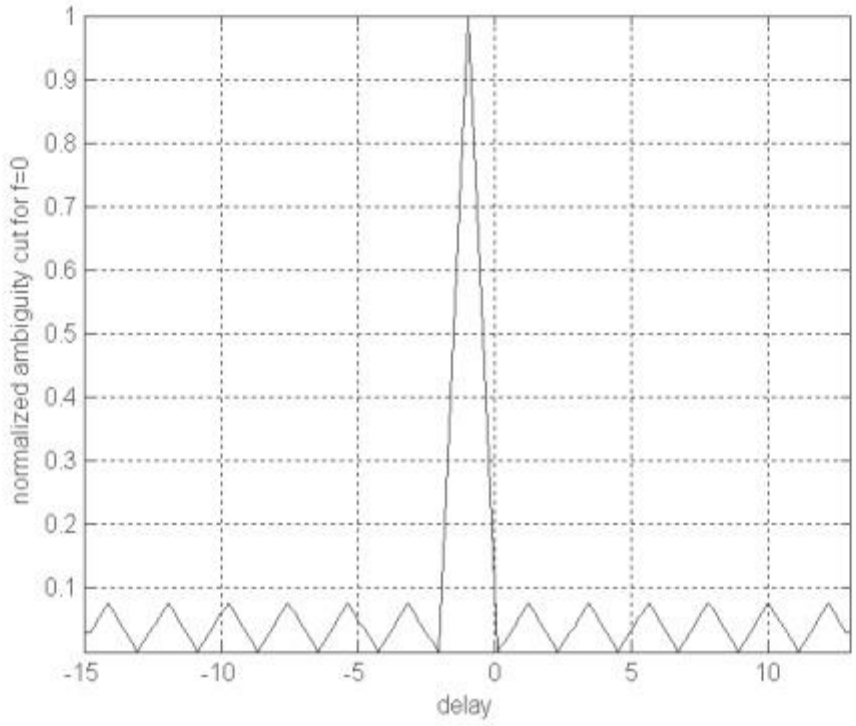

Figure 6.8b. Zero Doppler cut for the B_{13} ambiguity function.

Рисунок 6 – Нормализованный разрез кода Баркера (сторонний источник)

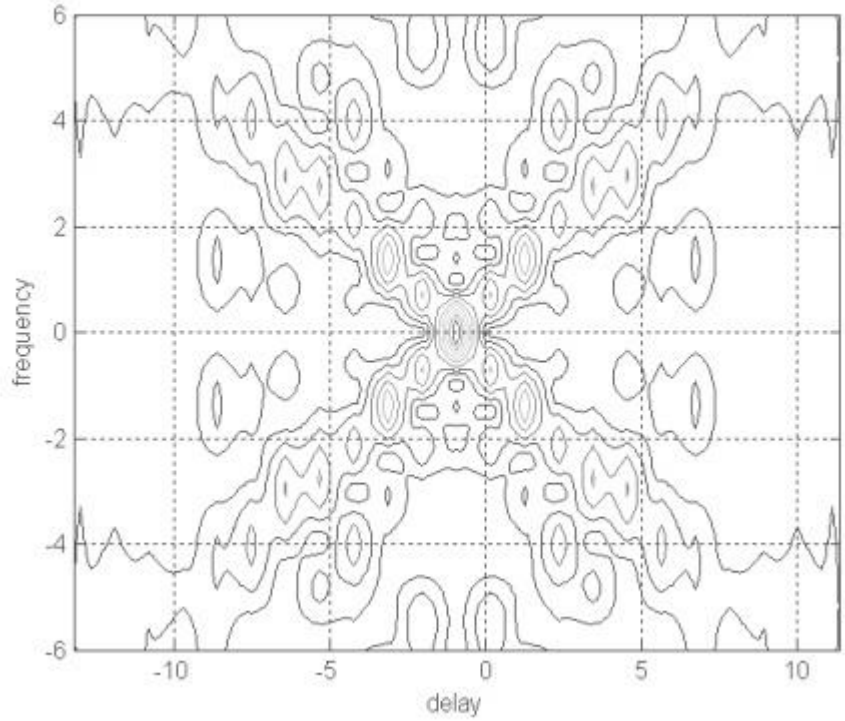

Рисунок 7 – Контурный график кода Баркера (сторонний источник)

Сразу можно заметить то, что полученные графики результате разработанной программы и графики, взятые из стороннего источника, имеют сильную, а в некоторых моментах и полную схожесть. В функции неопределённости основной пик находится в нуле задержки и частоты, а боковые лепестки – минимальны. Это связано с хорошими корреляционными свойствами кода Баркера (он имеет минимальные боковые лепестки в автокорреляции). Графики на рисунках 1 и 2 симметричны относительно центра (место нулевой задержки и частоты) и имеют один ярко выраженный центральный пик с относительно низкими боковыми лепестками.

Рассмотрим полученные графики относительно инвертированного кода Баркера. Они показаны на рисунках 8 – 11.

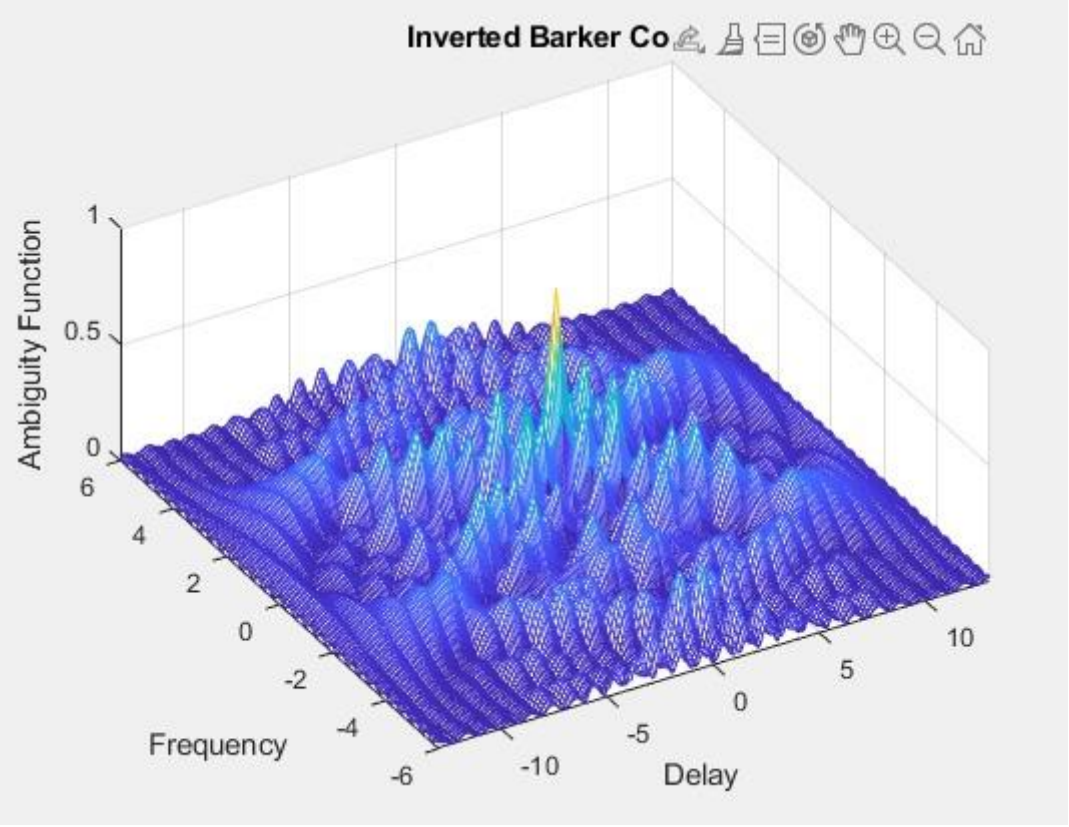

Рисунок 8 - Трёхмерный график инвертированного кода Баркера

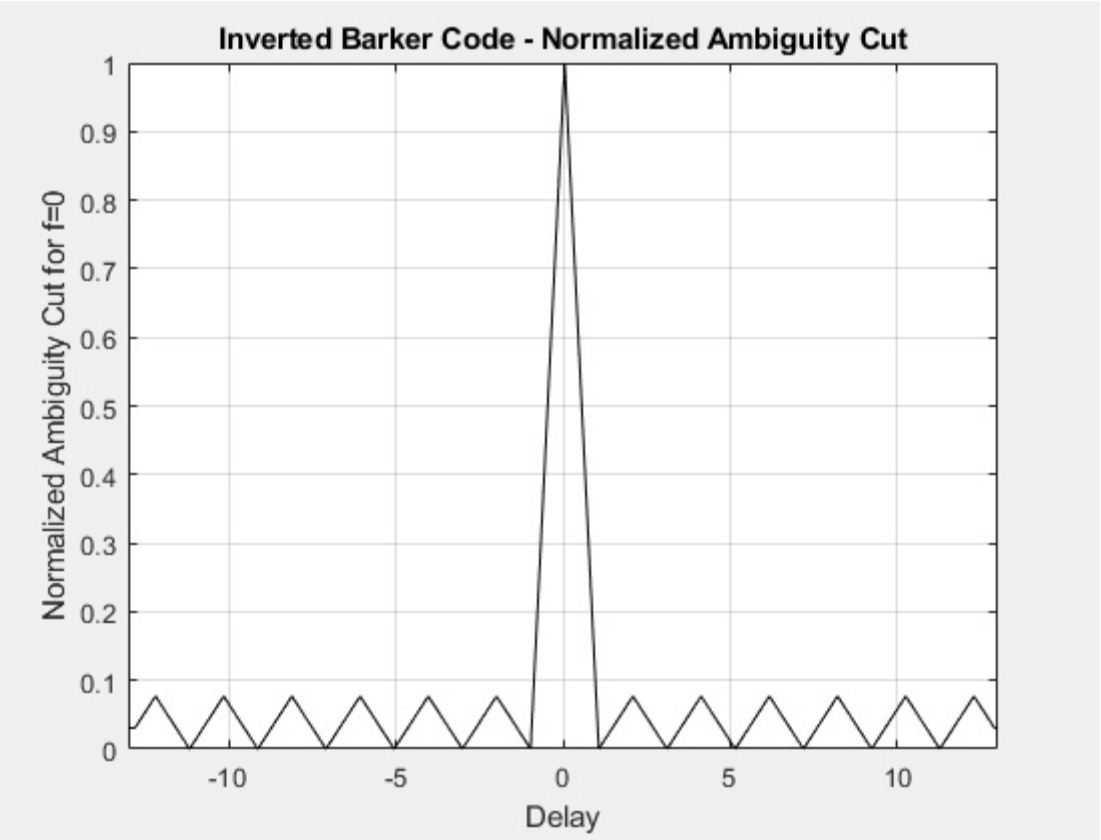

Рисунок 9 - Нормализованный разрез инвертированного кода Баркера

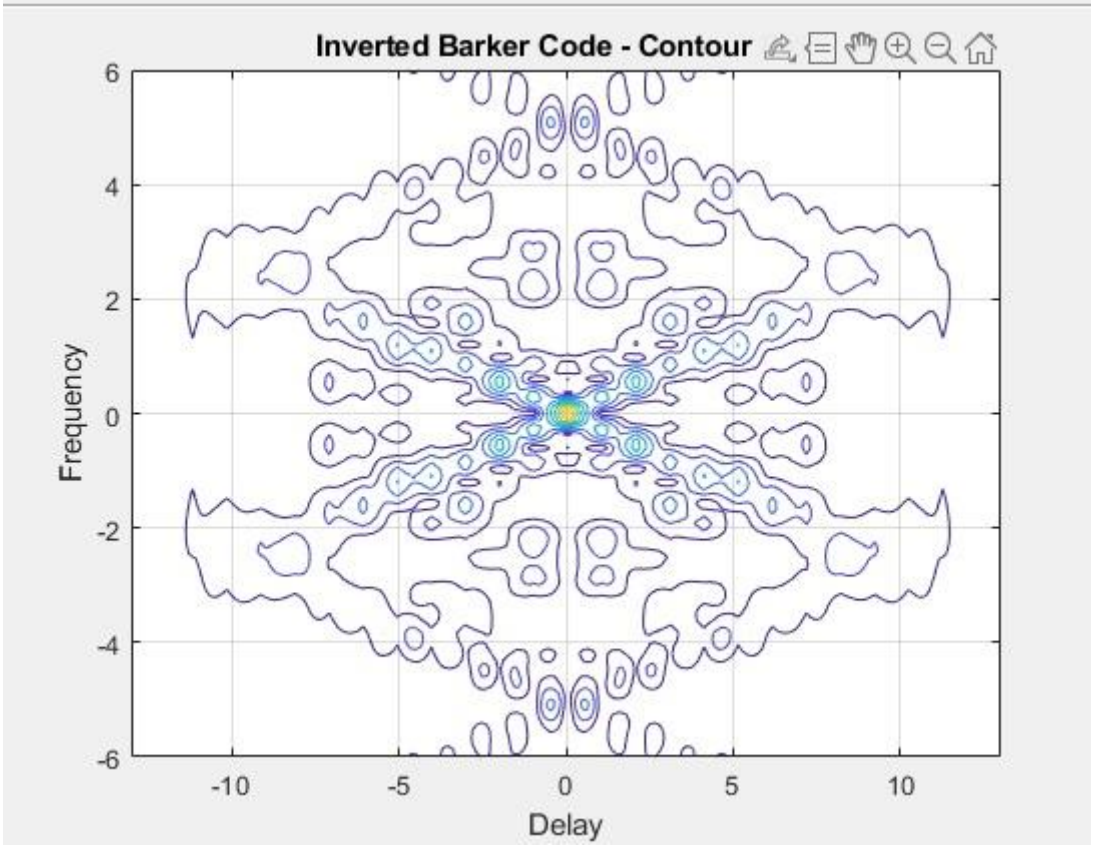

Рисунок 10 - Контурный график инвертированного кода Баркера

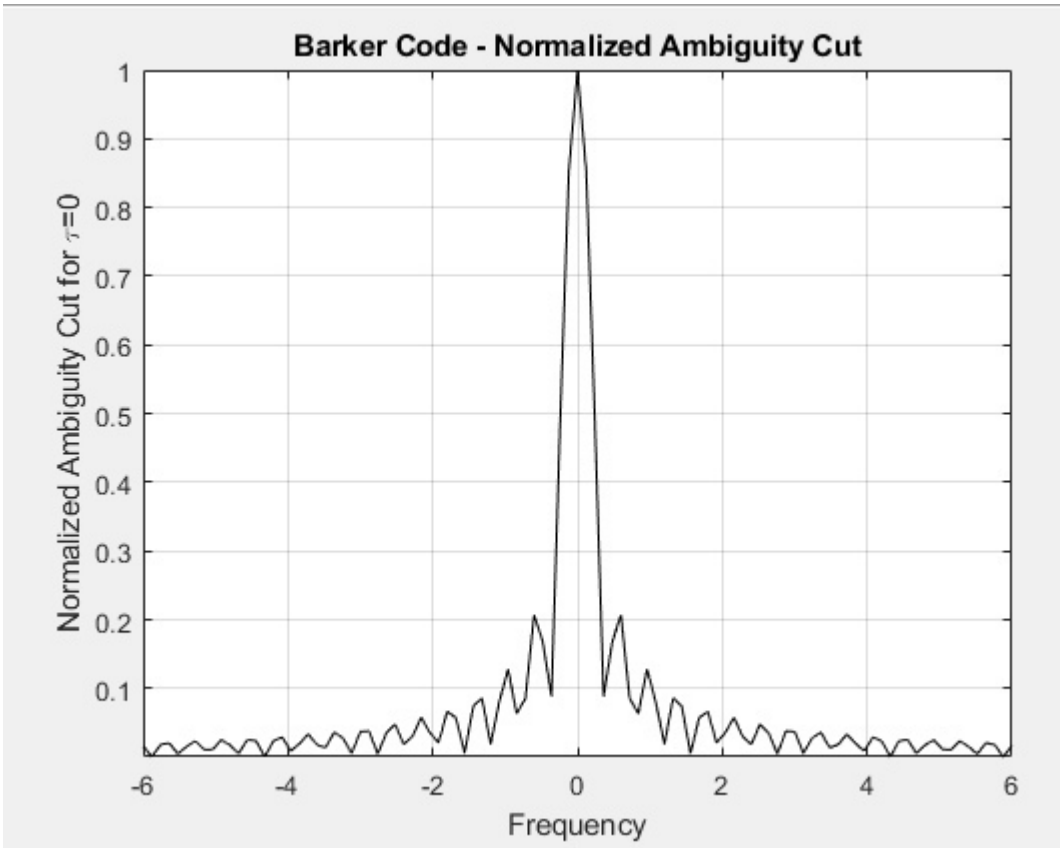

Рисунок 11 – Нормализованный разрез инвертированного кода Баркера (частотная корреляционная функция)

Функция неопределённости инвертированного кода Баркера идентична функции неопределённости для обычного кода Баркера, так как инверсия знаков не меняет автокорреляционные свойства. В силу идентичности графиков можно снова отчётливо заметить симметричность графика и центральный пик.

Рассмотрим последний набор полученный графиков, построенных относительно функции взаимной неопределённости. Графики продемонстрированы на рисунках 12 – 15.

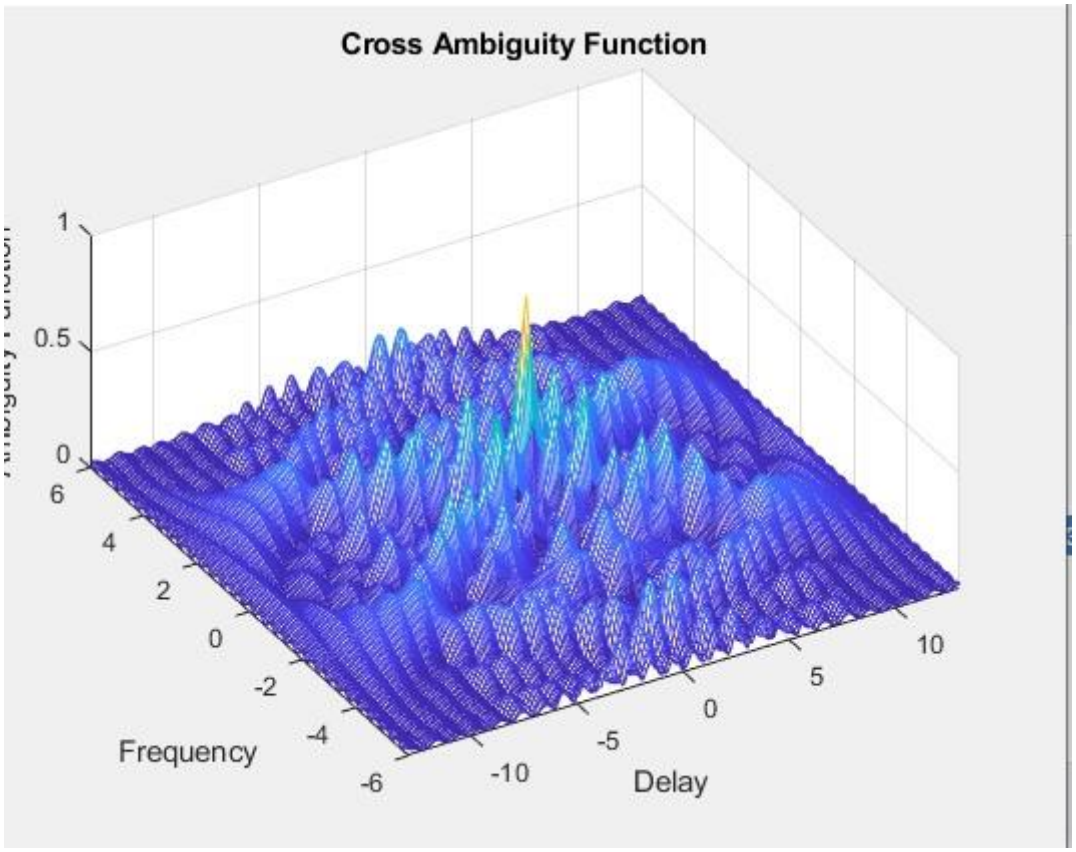

Рисунок 12 - Трёхмерный график функции взаимной неопределённости

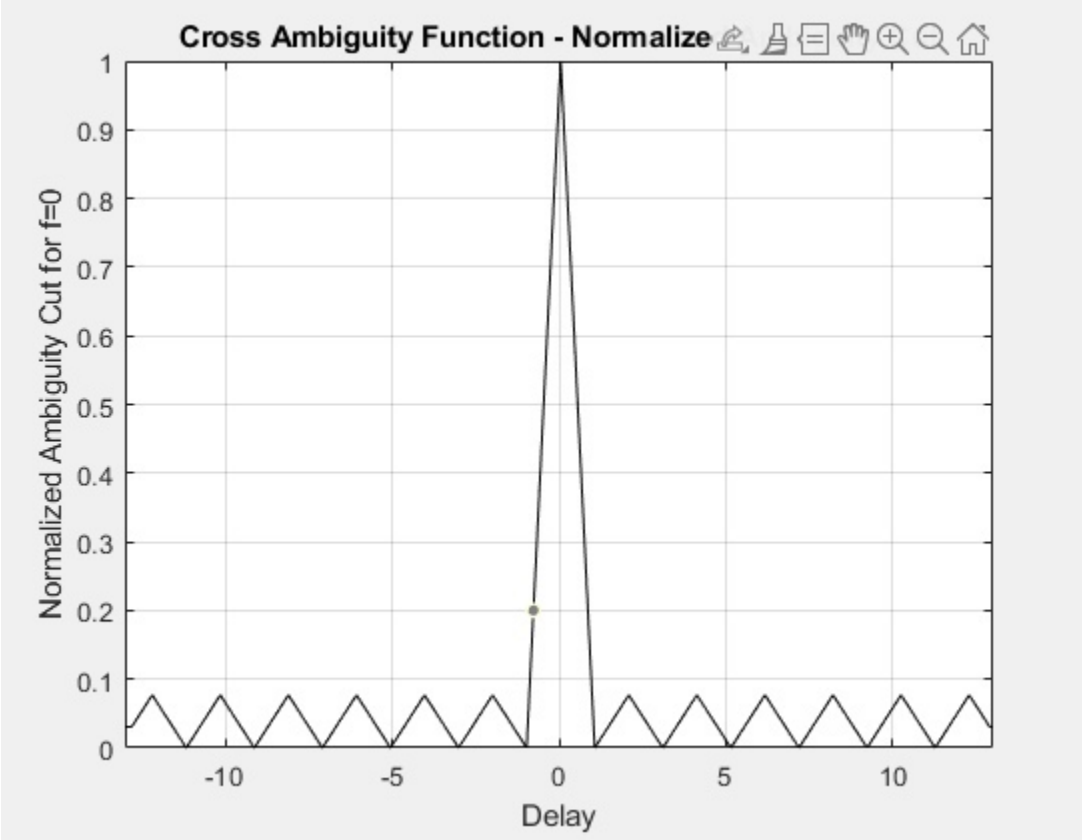

Рисунок 13 - Нормализованный разрез функции взаимной неопределённости

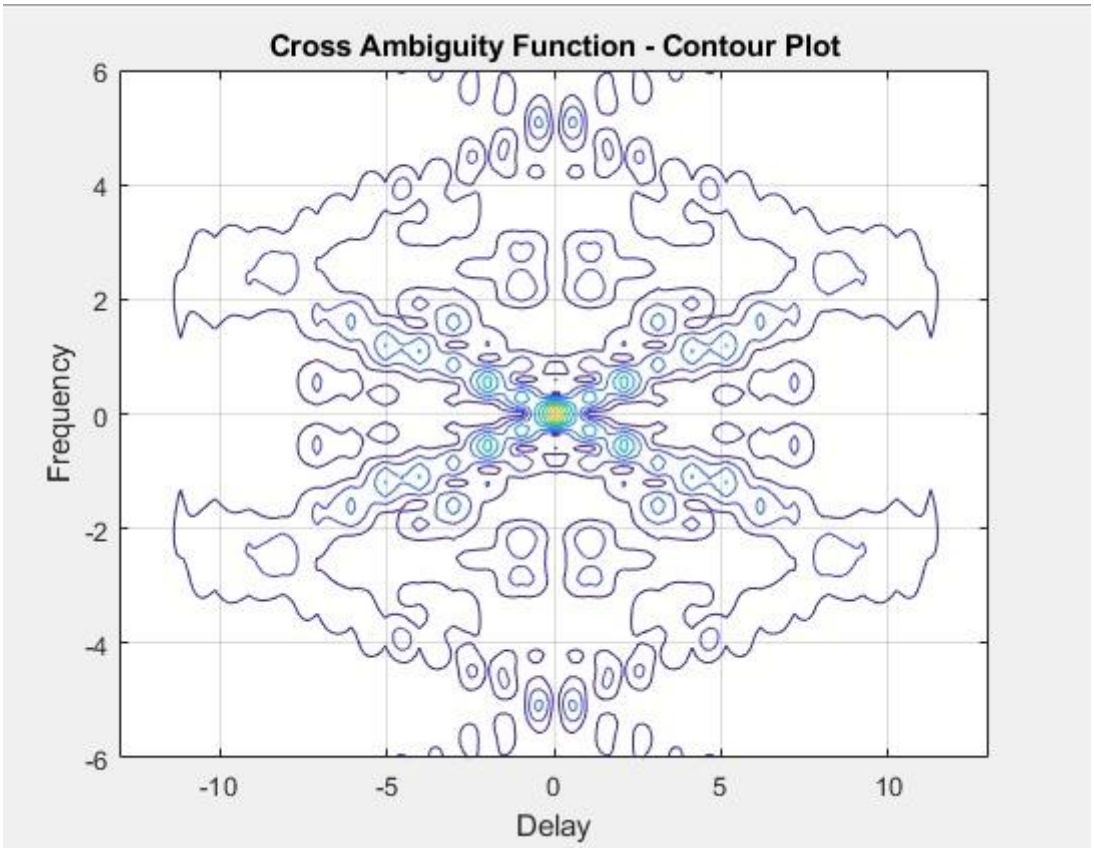

Рисунок 14 - Контурный график функции взаимной неопределённости

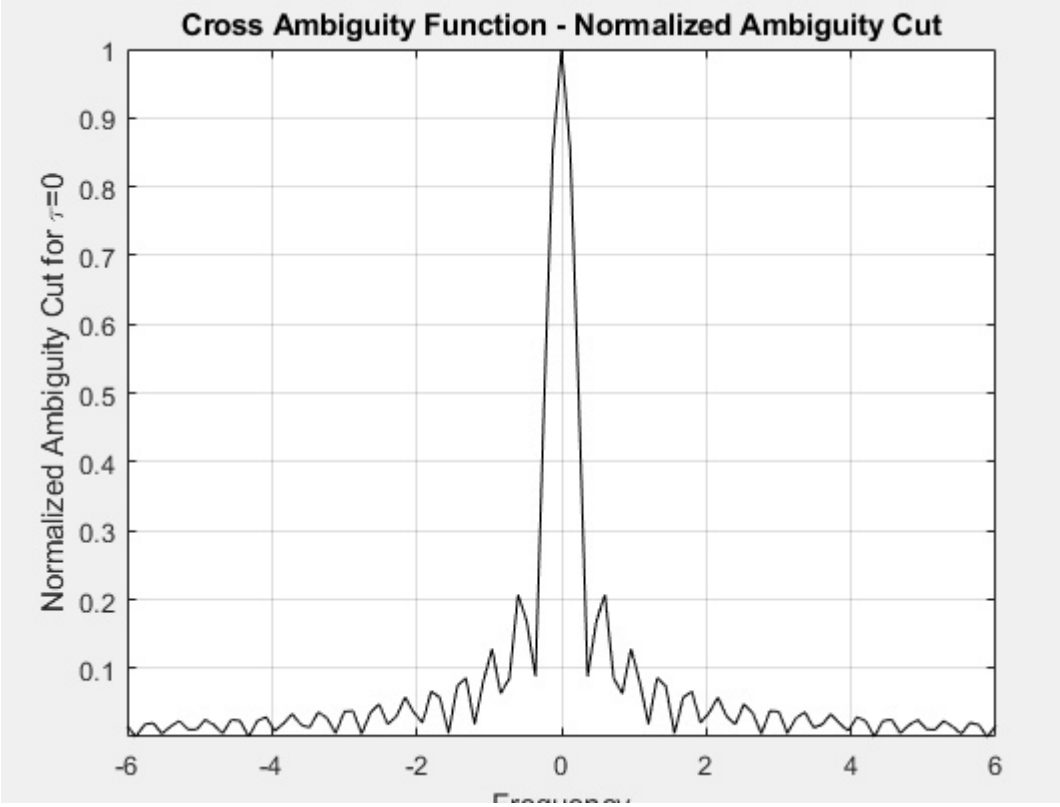

Рисунок 15 - Нормализованный разрез функции взаимной неопределённости (частотная корреляционная функция)

Функция взаимной неопределённости (между обычным и инвертированным кодом Баркера) показывает корреляцию между двумя сигналами. В силу того, что получилось так, что графики функции взаимной неопределённости идентичны полученным ранее. Есть предположение, что так сложилось в силу идеальной корреляции выбранных сигналов.

Проверим догадки. Для этого возьмём в качестве второго сигнала не просто инвертированный код Баркера, а случайную последовательность из 1 и -1 (главное, чтобы она её длина равнялась 13). Например, возьмём код вида:

$$[-1\ -1\ 1\ 1\ 1\ -1\ -1\ 1\ 1\ 1\ -1\ 1\ 1]$$

Получаем результаты, продемонстрированные на рисунках 16 – 19 относительно функции взаимной неопределённости и на рисунках 20 – 23 относительно второго кода.

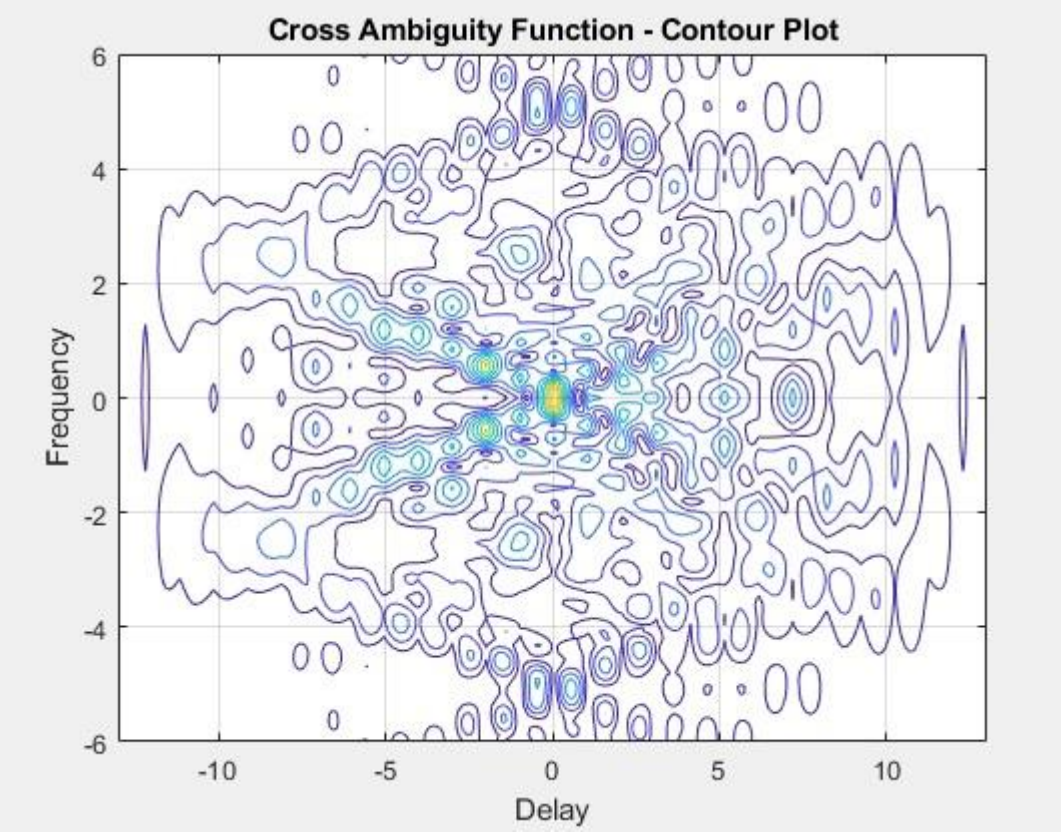

Рисунок 16 – Контурный график функции взаимной неопределённости

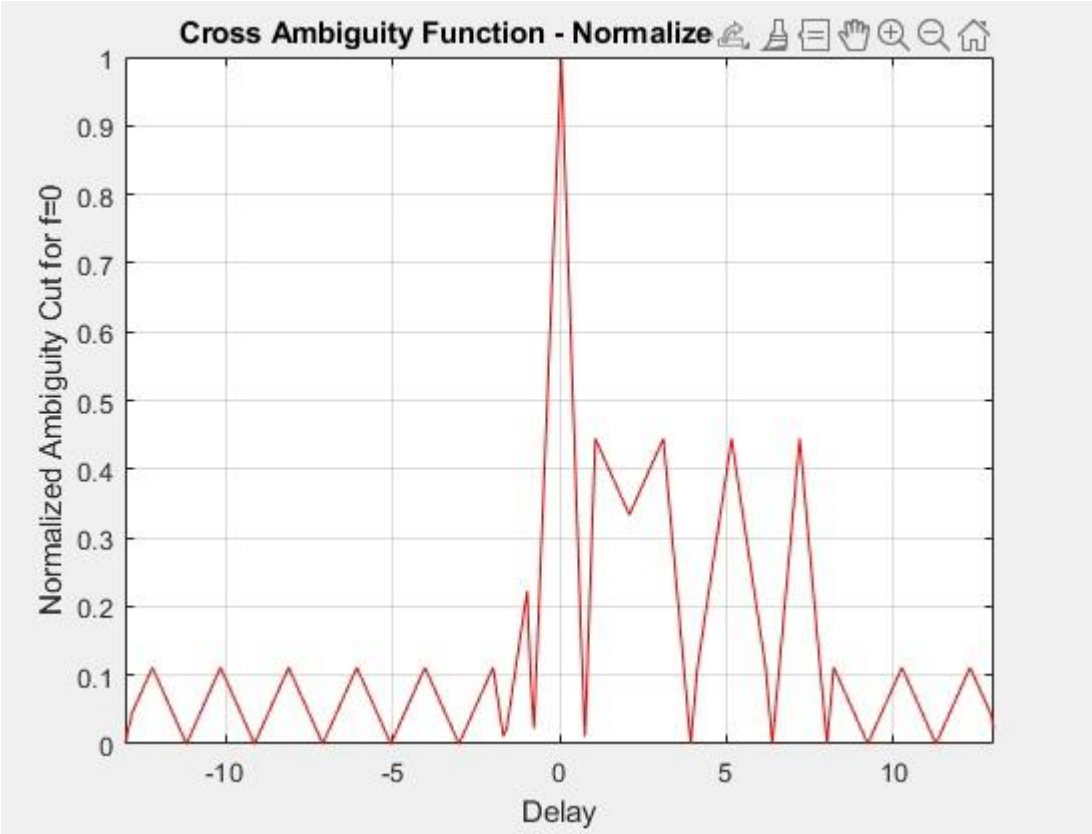

Рисунок 17 - Нормализованный разрез функции взаимной неопределённости

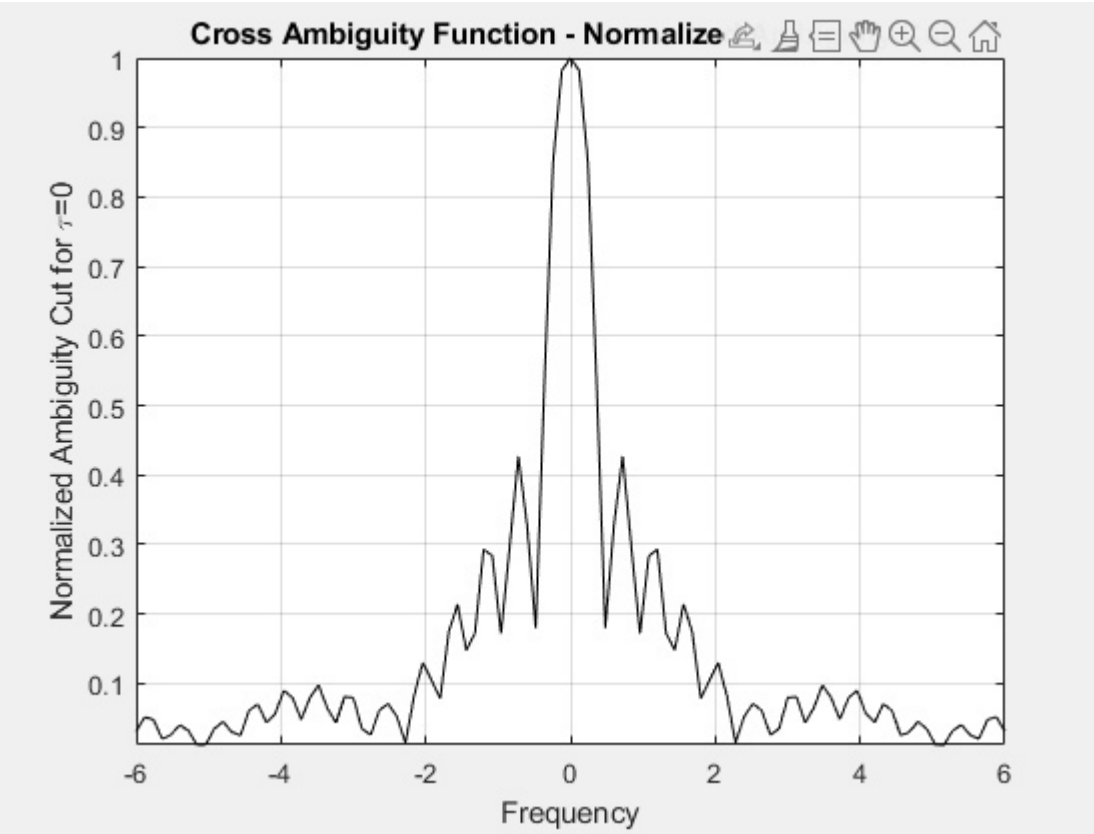

Рисунок 18 - Нормализованный разрез функции взаимной неопределённости (частотная корреляционная функция)

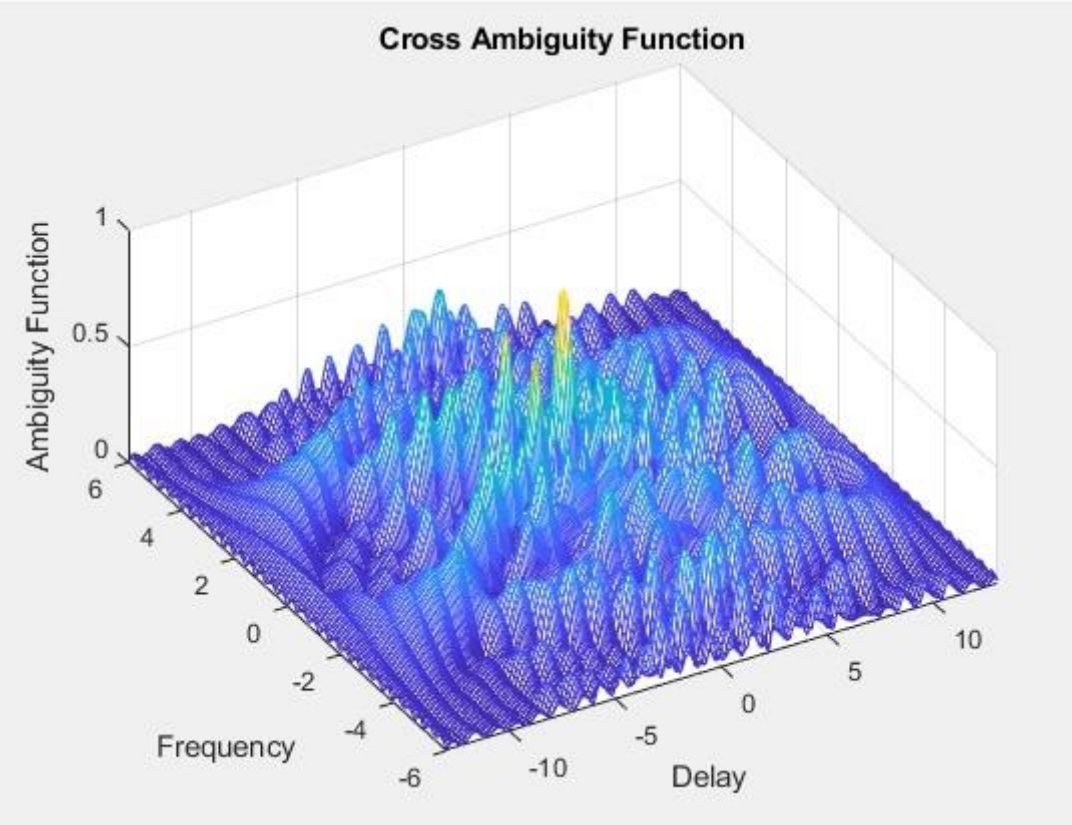

Рисунок 19 - Трёхмерный график функции взаимной неопределённости

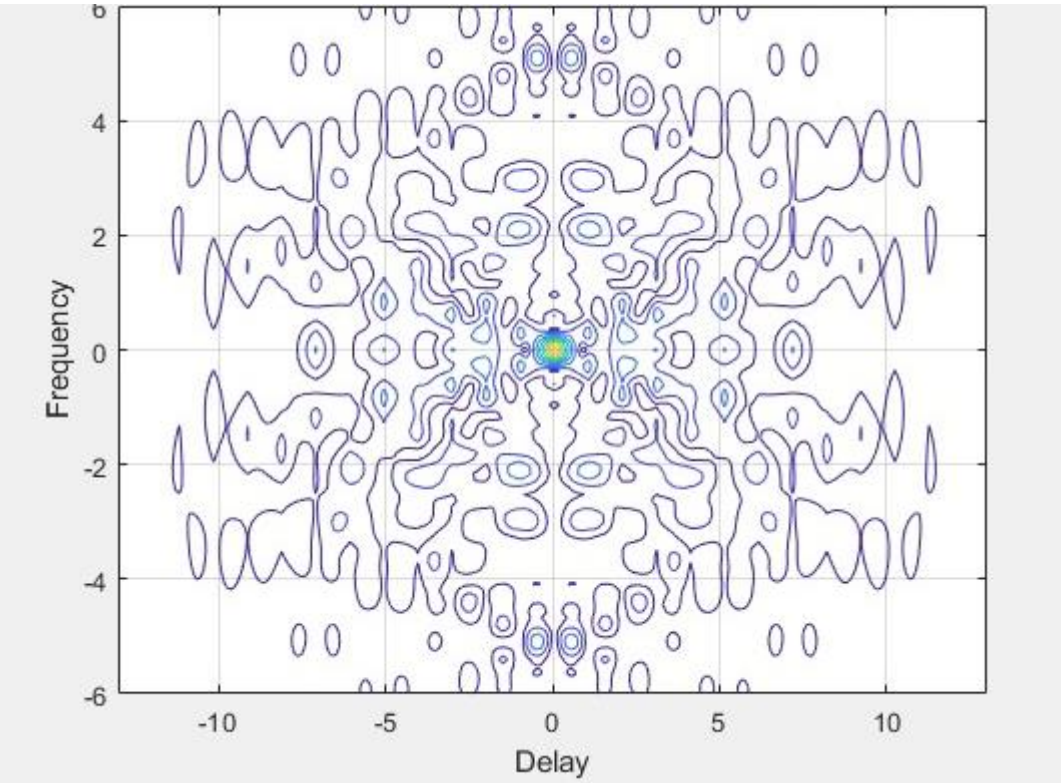

Рисунок 20 – Контурный график второго кода

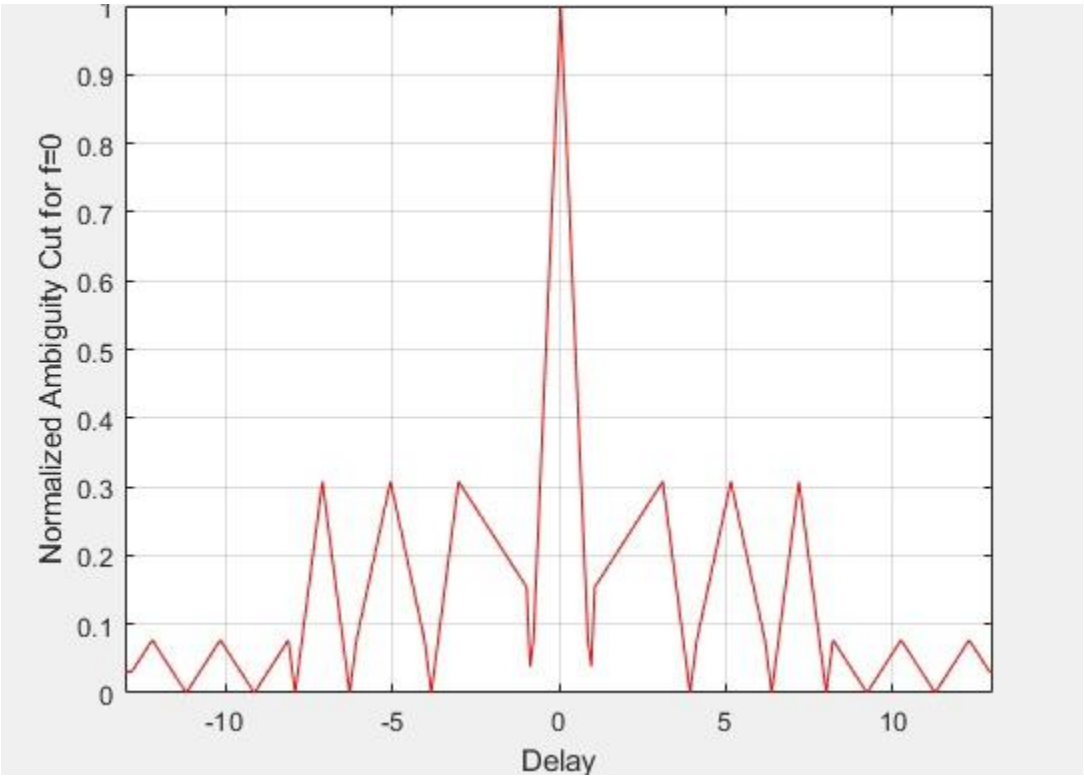

Рисунок 21 - Нормализованный разрез второго кода

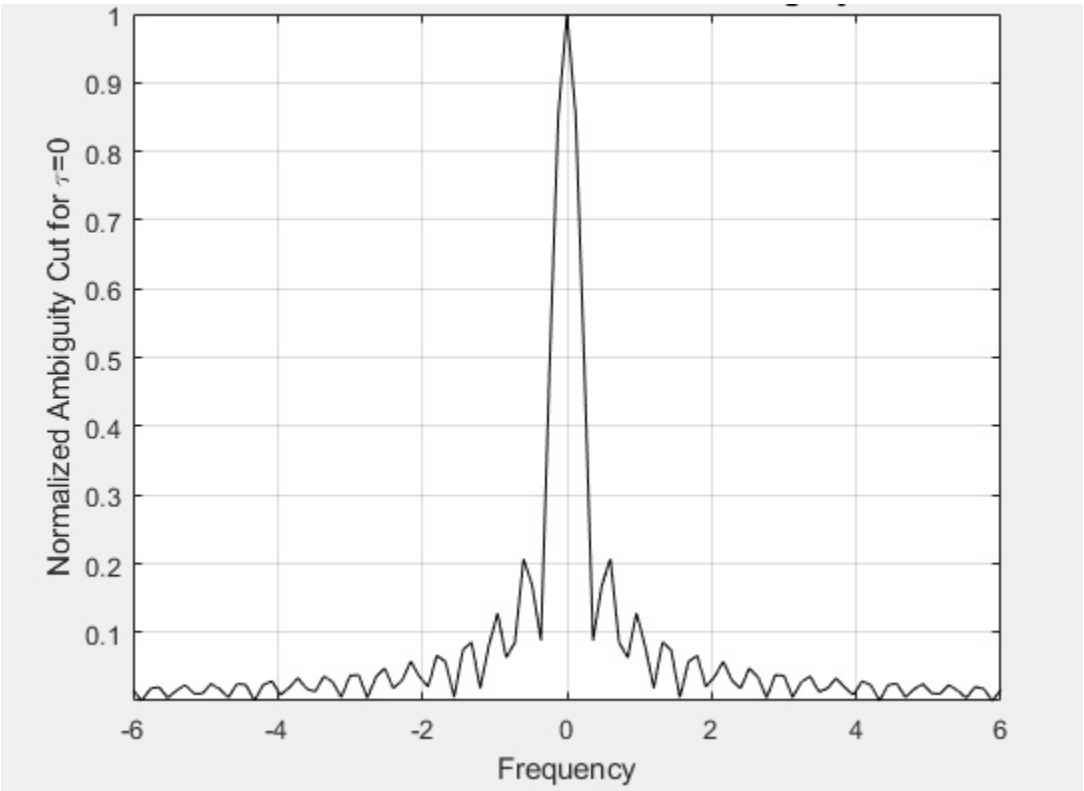

Рисунок 22 - Нормализованный разрез второго кода (частотная корреляционная функция)

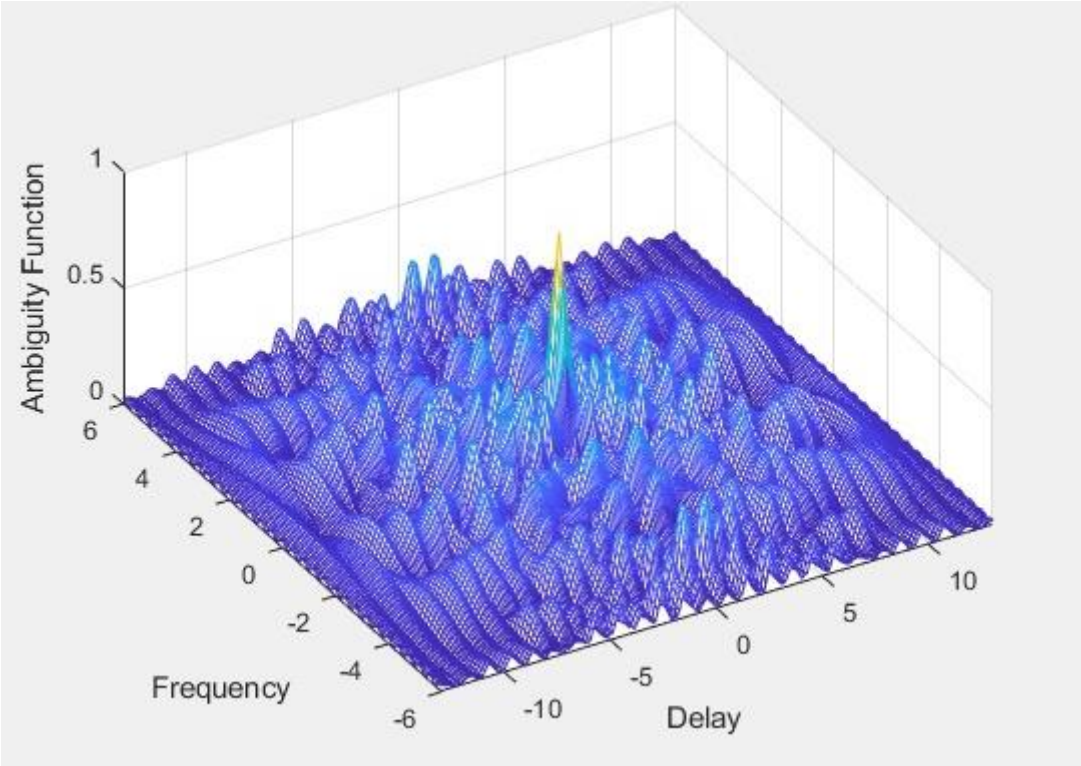

Рисунок 23 - Трёхмерный график второго кода

Проанализировав полученные графики можно сделать вывод о том, что наше предположение подтверждается. То есть, когда для построения функции взаимной неопределённости используется код Баркера и случайная последовательность, то графики будут отличаться в сравнении с предыдущими в случае использования кода Баркера и его инверсии. Это происходит из – за отсутствия высокой корреляции между выбранными сигналами (в отличие от первой ситуации). Это обуславливается «случайностью» значений кода, отчего на графиках наблюдаем видимые отличия. Эта ситуация помогает убедиться в том, что случайные последовательности не обладают свойствами кодов Баркера, обеспечивающими высокую корреляцию.

Отсюда и следует, что идентичность графиков в первом случае обуславливается высокой корреляцией, так как оба сигнала относятся к кодам Баркера, отличительной особенностью которых и является данное свойство. Функция взаимной неопределённости и показывает эту высокую корреляцию, то есть то, что они являются идеальными коррелирующими сигналами.

Выводы

В ходе выполнения данного проекта можно сделать следующие выводы.

Обнаружено, что функции неопределённости для кода Баркера и его инверсии идентичны. Это свидетельствует о том, что инверсия кода не оказывает никакого влияния на структуру автокорреляции. такое свойство является положительным, так как оно означает, что инвертированные кода могут быть использованы в тех же задачах, что и исходные, отчего эффективность не поменяется. При построении функции взаимной неопределённости в случае использования кодов Баркера (обычного и его инверсии) наблюдается идентичность графика с обычной функцией неопределённости. Как выяснилось, это связано с идеальной корреляцией между выбранными сигналами.

Также обнаружено, что у обоих кодов Баркера существует ярко выраженный пик корреляции при нулевой задержке. Эти пики соответствуют максимальному значению корреляции, равному длине последовательности.

Наблюдаются минимальные значения корреляции на ненулевых задержках для обоих кодов Баркера (боковые лепестки). Боковые лепестки могут иметь различную амплитуду, но как показывают результирующие графики, разница между основным пиком и боковыми лепестками по амплитуде довольно весома. Это подтверждает ключевое свойство кодов Баркера содержать в себе минимальные помехи, что полезно в задачах, требующих высокой точности временной синхронизации и малых уровней интерференции.

При проведении эксперимента с целью подтверждения догадки относительно идеальной корреляции кодов Баркера и его инверсии было обнаружено, что боковые лепестки в полученных новых графиках (наглядно это можно наблюдать даже на графике нормализованных срезов у второго кода, что так же даёт сразу понять о более плохой корреляции) имеют

существенно большие пики. Данная ситуация говорит о том, что сигналы хуже коррелируют между собой. Отсюда приходим к выводу о том, что при построении функции взаимной неопределённости случайные сигналы не имеют свойства идеально коррелировать в отличие от кодов Баркера.

Это ключевое свойство кодов Баркера идеально коррелировать может быть очень полезно, например, в сфере радиолокации. А именно, в силу этого свойства, устройство-приёмник, принимающее сигнал от передатчика, может точно определить, в какое время он принял сигнал. Также, помимо времени, приёмник в принципе может определить, какой это сигнал по заранее данным ему параметрам (некий шаблон), которые описывают сигнал. Уже отсюда следует, что это будет полезно при необходимости понимания расстояния между передатчиком и приёмником. Даже если сигнал зашумлён или сам по себе довольно слаб, его всё равно можно обнаружить.

Помимо этого, понятие корреляции используют в своей работе так называемые коррелирующие фильтры. Суть их работы заключается в том, что в вышеупомянутой системе «приёмник-передатчик» фильтр может помочь приёмнику точно определить изначально плохообнаруживаемый сигнал.

Список используемых источников

1. Радиолокационные системы авиационно-космического мониторинга земной поверхности и воздушного пространства. / Под ред. д.т.н.,проф. В. С. Вербы, д.т.н., проф. Б.Г. Татарского.: Москва, Радиотехника, 2014- 574с.

2. Теоретические основы радиолокации: [Электронный ресурс] // URL.: https://studizba.com/files/show/doc/228645-5-lekciya-15-16-konspekty-k-slaydam.html. (Дата обращения: 04.06.24).

3. Код Баркера: [Электронный ресурс] // URL.: https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%91%D0%B0%D1%80%D0%BA%D0%B5%D1%80%D0%B0. (Дата обращения: 04.06.24).

3. Синтез уникальных фазоманипулированных сигналов для интеллектуального обнаружения подвижных объектов: [Электронный ресурс] // URL.: http://jurnal.org/articles/2008/izmer9.html. (Дата обращения: 04.06.24).

4. Radar Systems Analysis and Design Using Matlab. / Bassem R. Mahafza.: Huntsville, Alabama, USA, CRC Press, 2013- 743с.