МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное
учреждение высшего образования
«Национальный исследовательский университет «МЭИ»

Институт **ИРЭ**
Кафедра **РТС**

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(магистерская диссертация)

Направление **11.04.01 - Радиотехника**
(код и наименование)

Направленность (программа) **Радиотехнические системы**

Форма обучения **очная**
(очная/очно-заочная/заочная)

Тема: **Обработка современных сигналов ГНСС с модуляцией цифровой поднесущей традиционными корреляторами**

Студент	**ЭР-12м-19**			**Михайлова О.К.**
	группа		подпись	фамилия и инициалы
Научный руководитель	**к.т.н.**	**доцент**		**Корогодин И.В.**
	уч. степень	должность	подпись	фамилия и инициалы
Консультант				
	уч. степень	должность	подпись	фамилия и инициалы
Консультант				
	уч. степень	должность	подпись	фамилия и инициалы

«Работа допущена к защите»

Зав. кафедрой	**к.т.н.**	**доцент**		**Куликов Р.С.**
	уч. степень	звание	подпись	фамилия и инициалы

Дата **21.06.2021**

Москва, 2021

МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное
учреждение высшего образования
«Национальный исследовательский университет «МЭИ»

Институт ИРЭ
Кафедра РТС

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
(магистерскую диссертацию)

Направление **11.04.01 - Радиотехника**
(код и наименование)

Направленность (профиль) **Радиотехнические системы**

Форма обучения **очная**
(очная/очно-заочная/заочная)

Тема: **Обработка современных сигналов ГНСС с модуляцией цифровой поднесущей традиционными корреляторами**

Студент	**ЭР-12м-19**			**Михайлова О.К.**
	группа		подпись	фамилия и инициалы
Научный руководитель	**к.т.н.**	**доцент**		**Корогодин И.В.**
	уч. степень	должность	подпись	фамилия и инициалы
Консультант				
	уч. степень	должность	подпись	фамилия и инициалы
Консультант				
	уч. степень	должность	подпись	фамилия и инициалы
Зав. кафедрой	**к.т.н.**	**доцент**		**Куликов Р.С.**
	уч. степень	звание	подпись	фамилия и инициалы

Место выполнения работы **Кафедра РТС**

1.Обоснование выбора темы выпускной квалификационной работы

Современные спутниковые радионавигационные системы используют навигационные сигналы с BOC модуляцией. Применение данного типа модуляции позволяет повысить точность оценивания координат потребителя, а также снижает межсистемные и внутрисистемные помехи. Обработка таких сигналов усложняется наличием модуляции поднесущей, что может потребовать изменения аппаратной структуры навигационных приемников, что зачастую приводит к необходимости разработки новых.

Актуальной является задача осуществления когерентной обработки BOC сигналов с помощью двух традицонных корреляторов, что в дальнейшем позволит обрабатывать BOC и AltBOC сигналы без усложнения аппаратной части коррелятора, при этом, не теряя в точности оценки задержки.
Исследование в данном направлении актуальны и перспективны.

Научный руководитель Корогодин И.В. **дата** 10.09.2019

Студент Михайлова О.К. **дата** 10.09.2019

2. Консультации по разделу

Подпись консультанта ____________ **дата** ____________

3.Консультации по разделу

Подпись консультанта ____________ **дата** ____________

4. План выполнения выпускной квалификационной работы

№ п\п	Содержание разделов	Срок выпол-нения	Трудоём-кость, %
I.	Теоретическая часть 1 Постановка задачи и синтез алгоритма дискриминатора задержки ВОС сигнала с использованием традиционных BPSK корреляторов; 2 Аналитический расчет статистических характеристик дискриминатора задержки: дискриминационной и флуктуационной характеристик;	апрель 2020	30%
II.	Экспериментальная часть 1 Имитационное моделирование: 1.1 Проверка характеристик дискриминатора задержки; 1.2 Получение характеристик слежения за задержкой ВОС сигналов; 1.3 Получение характеристик захвата; 2 Экспериментальное исследование возможности обработки сигнала Galileo E5 с помощью двух радиотрактов;	май 2020 апрель 2021 май 2021	20%: 7%
III.	Публикации 1 Статья в журнал РИНЦ; 2 Статья в журнал Scopus; 3 Тезисы докладов на НТК студентов и	октябрь 2019 ноябрь 2020г.	3% 10%

	аспирантов 2шт.	март 2021г.	5%
IV.	Оформление диссертации	май 2021г.	25%

5. Рекомендуемая литература

1. ГЛОНАСС. Модернизация и перспективы развития. Монография / Под. ред. Перова А.И. — М.: Радиотехника, 2020. — 1072 с.
2. E.D. Kaplan, C.J. Hegarty. Understanding GPS: Principles and Applications. —second edition. —Artech House, 2005. —ISBN: 1580538940.
3. Перов А. И., В.Н. Замолодчиков, В.М. Чиликин. Радиоавтоматика. — М.: Радиотехника, 2014.
4. Шатилов А. Ю. Характеристики радиосигналов глобальных спутниковых радионавигационных систем ГЛОНАСС, GPS, GALILEO, BEIDOU и функциональных дополнений SBAS. —Изд-во МЭИ, 2015.

Примечания:

1. Задание брошюруется вместе с выпускной работой после титульного листа (страницы задания имеют номера 2, 3, 4, 5).
2. Отзыв руководителя, рецензия(и), отчет о проверке на объем заимствований и согласие студента на размещение работы в открытом доступе вкладываются в конверт (файловую папку) под обложкой работы.

Оглавление

Перечень сокращений

AltBOC	Alternate Binary Offset Carrier
BOC	от англ. Binary offset carrier
BPSK	от англ. Binary phase-shift keying
LSB	от англ. lower side band
NCO	от англ. numerically-controlled oscillator
NELP	от англ. Non-coherent Early minus Late Power
SCPC	от англ. Sub Carrier Phase Cancellation technique
USB	от англ. upper side band
АКФ	Автокорреляционная функция
АЦП	Аналого-цифровой преобразователь
ГНСС	Глобальные навигационные спутниковые системы
ДЗ	Дискриминатор задержки
ДХ	Дискриминационная характеристика
КСШВ	Кодовая сигнальная шкала времени
КФ	Корреляционная функция
НАП	Навигационная аппаратура потребителя
ПК	Персональный компьютер
ПЛИС	Программируемая логическая интегральная схема
РЧБ	Радиочастотный блок
СПМ	Спектральная плотность мощности
ССЗ	Система слежения за задержкой

ССФ	Система слежения за фазой
ССЧ	Система слежения за частотой
СнК	Система на кристалле, от англ. System-on-a-Chip, SoC
ФД	Фазовый дискриминатор
ФСШВ	Фазовая сигнальная шкала времени
ФХ	Флуктуационная характеристика
ЧД	Частотный дискриминатор

ВВЕДЕНИЕ

Спутниковые навигационные системы используются для определения местоположения потребителей посредством измерения задержки навигационных сигналов. Постепенно навигационные системы модернизируются, в них расширяется список используемых сигналов. Если первые сигналы имели модуляцию BPSK, то некоторые новые сигналы имеют модуляцию типа BOC (с модуляцией цифровой поднесущей). Такие сигналы позволяют повысить точность оценки задержки, а также позволяют лучше использовать предоставленные частоты, что снижает внутрисистемные и межсистемные помехи.

BOC сигналы помимо модуляции дальномерным кодом дополнительно модулируются цифровой поднесущей. Цифровая поднесущая – это меандровое колебание: $B(t) = \mathrm{sign}(\sin(2\pi f_s t))$, где f_s – частота поднесущей.

Модуляция цифровой поднесущей приводит к переносу сигнала на суммарную и разностную частоты, так называемые поднесущие частоты. Так как частота цифровой поднесущей немногим больше темпа следования символов дальномерного кода, то спектр сигнала принимает характерную форму с двумя лепестками. Это усложняет обработку таких сигналов, так как структура BOC сигналов отличается от традиционных, что может потребовать изменение аппаратных средств, а зачастую приводит к необходимости разработки новых [1], [2].

Мощность BOC сигнала сосредоточена в двух лепестках, однако, вся информация для приема и декодирования такого сигнала содержится в каждом из лепестков. Поэтому часто для обработки BOC сигналов используют, так называемые, BPSK-like методы [3], которые позволяют использовать аппаратные модули, изначально предназначенные для BPSK сигналов. При применении этих методов либо обрабатывается только один лепесток, что ведёт к потере 3 дБ мощности, либо лепестки обрабатываются независимо. Но и в одном, и во втором случае расширяется корреляционный пик, из-за чего деградирует точность оценки задержки. Прямой прием BOC сигнала обеспечивает потенциальную точность, но требует добавления цифровой поднесущей и в опорный сигнал коррелятора.

Наибольшая точность достигается при обработке обоих лепестков спектра когерентно. В работе предлагается использовать для обработки сигналов с BOC модуляцией два традиционных BPSK канала коррелятора. Такой подход не требует изменения аппаратной части навигационного приемника, и может быть легко адаптирован для обработки AltBOC сигналов ввиду возможности отдельной настройки каждого канала коррелятора. Также благодаря этому возможно пропускать лепестки спектра сигнала с BOC модуляцией через различные каналы радиочастотного блока (РЧБ), что снижает требование к ширине полосы пропускания РЧБ.

Цель работы – синтез и анализ алгоритма обработки BOC сигнала с помощью двух BPSK корреляторов, сравнимого по точности с алгоритмом, использующем специализированные корреляторы.

1 Постановка задачи

Рассмотрим прием навигационного сигнала, модулированного дальномерным кодом $C\left(\ \right)$, цифровой поднесущей $B\left(\ \right)$ и наблюдаемого на выходе АЦП приемника:

$$S\left(t^{cs}\left(t_{k,l}\right),t^{cp}\left(t_{k,l}\right),t_{k,l}\right) = AC\left(t^{cp}_{k,l}\right)B\left(t^{cp}_{k,l}\right)\cos\left(2\pi f_0 t^{ps}_{k,l} - 2\pi\left(f_0 - f_{if}\right)t_{k,l}\right), \tag{1.1}$$

где $t^{cs}\left(t_{k,l}\right),t^{cp}\left(t_{k,l}\right)$ - кодовое и фазовое сигнальное время на момент $t_{k,l}$ по шкале времени приемника, для которой используется двойная индексация:

$$t_{k,l} = t_{1,1} + (k-1)\,T + (l-1)\,T_d, \tag{1.2}$$

где $T_d = {}^1/_{F_d}$ - интервал дискретизации АЦП, $T = LT_d$, $l = 1..L$, f_{if} - номинал промежуточной частоты, f_0 - номинал несущей частоты, A- амплитуда сигнала, принимается известной. Выразим (1.1) через задержку огибающей, частоту и начальную фазу:

$$S\left(\tau_k,\omega_k,\varphi_k\right) = AC\left(t_{k,l} - \tau_k\right)B\left(t_{k,l} - \tau_k\right)\cos\left(\omega_{if}t_{k,l} + \omega_k\left(l-1\right)T_d + \varphi_k\right), \tag{1.3}$$

где $\omega_0 = 2\pi f_0$, $\omega_{if} = 2\pi f_{if}$

$$\omega_0 t^{ps}_{k,l} = \omega_0 t_{k,l} + \omega_k\left(l-1\right)T_d + \varphi_k, \tag{1.4}$$

$$\tau_k = t_{k,l} - t^{cs}_{k,l}. \tag{1.5}$$

Здесь параметры задержки огибающей τ_k, частоты ω_k и начальной фазы φ_k приняты неизменными на k-м интервале. Например, они могут описываться моделями диффузионных марковских дискретных случайных процессов:

$$\tau_k = \tau_{k-1} + \xi_{k-1}, \tag{1.6}$$

где ξ_{k-1} - формирующий дискретный белый гауссовский шум с дисперсией σ^2_ξ и нулевым математическим ожиданием.

Положим, что сигнал $S_{k,l}$ наблюдается на фоне аддитивного белого гауссовского дискретного шума $n_{k,l}$ с дисперсией σ^2_n и нулевым математическим ожиданием:

$$y_{k,l} = S_{k,l}\left(\tau_k,\omega_k,\varphi_k\right) + n_{k,l}. \tag{1.7}$$

Рассмотрим задачу фильтрации задержки сигнала. При определенных допущениях квазиоптимальным по критерию минимизации среднеквадратической ошибки оценивания является алгоритм из двух систем слежения: системы слежения за задержкой (ССЗ) и системы слежения за фазой (ССФ) или системы слежения за частотой (ССЧ). Система слежения за частотой возникает в том случае, когда фаза относится к неинформативным параметрам и не подлежит оцениванию. В этом случае слежение называется некогерентным, в противном – когерентным.

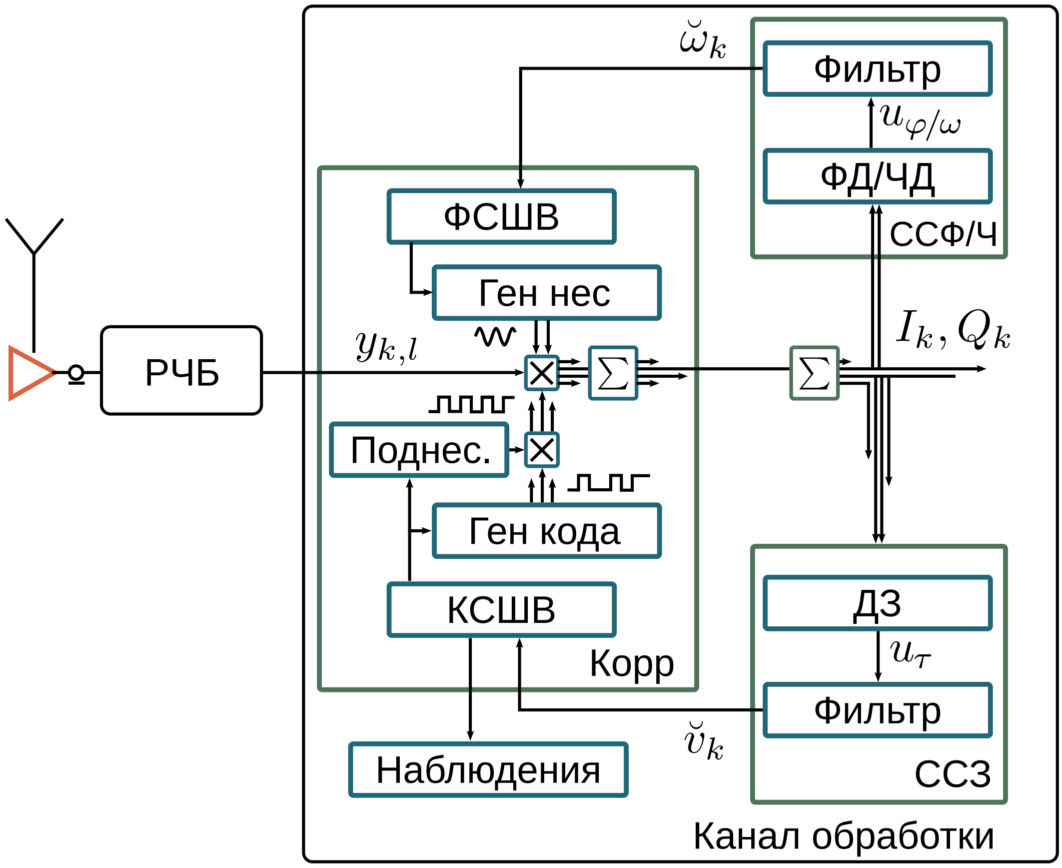

Рисунок 1.1 — Слежение за параметрами навигационного сигнала

На рисунке РЧБ – радиочастотный блок, ФСШВ – фазовая сигнальная шкала времени, КСШВ – кодовая сигнальная шкала времени, ФД – фазовый дискриминатор, ЧД – частотный дискриминатор, ДЗ – дискриминатор задержки.

2 Синтез алгоритма слежения за задержкой BOC сигналов с использованием сплит-коррелятора

2.1 Синтез дискриминатора задержки

Используемые системы слежения содержат дискриминатор и фильтр, выходной сигнал которого замыкается на общий коррелятор, участвующий в алгоритме дискриминатора. Коррелятор осуществляет перемножение отсчетов наблюдения (1.7) на опорный сигнал с последующим накоплением на интервале T. Опорные сигналы формируются с учетом оценок поступающих от систем слежения. Дискриминатор системы синтезируется как производная функции правдоподобия по оцениваемому параметру. Например, в случае некогерентного слежения алгоритм дискриминатора задержки принимает вид (англ. NELP, Non-coherent Early minus Late Power):

$$u_\tau = -\left(I_e^2 + Q_e^2\right) + \left(I_l^2 + Q_l^2\right) \tag{2.1}$$

где $I_{e/l}$, $Q_{e/l}$ - синфазная и квадратурная корреляционные суммы с опережающим/запаздывающим дальномерным кодом в опорном сигнале:

$$\begin{aligned}
I_{e,k} = I_{e,k}^{direct} &= \sum_{l=1}^{L} y_{k,l} C_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k - \Delta\right)\right) B_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k - \Delta\right)\right)\times \\
&\times \cos\left(\omega_{if} t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right), \\
Q_{e,k} = Q_{e,k}^{direct} &= \sum_{l=1}^{L} y_{k,l} C_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k - \Delta\right)\right) B_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k - \Delta\right)\right)\times \\
&\times \sin\left(\omega_{if} t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right), \\
I_{l,k} = I_{l,k}^{direct} &= \sum_{l=1}^{L} y_{k,l} C_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k + \Delta\right)\right) B_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k + \Delta\right)\right)\times \\
&\times \cos\left(\omega_{if} t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right), \\
Q_{l,k} = I_{l,k}^{direct} &= \sum_{l=1}^{L} y_{k,l} C_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k + \Delta\right)\right) B_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k + \Delta\right)\right)\times \\
&\times \sin\left(\omega_{if} t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right).
\end{aligned} \tag{2.2}$$

Здесь и далее символ "$\smile$"используется для обозначения параметров опорного сигнала.

Реализация дискриминатора (2.1) в прямой форме требуется создания специальных каналов коррелятора (2.2), в которых бы опорный сигнал содержал не только дальномерный код $C\left(\right)$, но и цифровую поднесущую $B\left(\right)$ (см. рис.1.1).

Проведем ряд преобразований, чтобы снизить требования к каналу коррелятора. Аппроксимируем цифровую поднесущую опорного сигнала её первой гармоникой:

$$B_{k,l}\left(x\right) = \operatorname{sign}\left(\sin\left(2\pi f_B x\right)\right) \approx \sin\left(2\pi f_B x\right) = \sin\left(\omega_B x\right), \tag{2.3}$$

где f_B - частота поднесущей, $\omega_B = 2\pi f_B$.

Разделим параметры задержки огибающей для дальномерного кода $\tau_k \to \tau_k^C$, $\Delta \to \Delta^C$ и поднесущей $\tau_k \to \tau_k^B$, $\Delta \to \Delta^B$. Тогда, например, синфазная опережающая корреляционная сумма с учетом (2.3) принимает вид:

$$\begin{gathered}
I_{e,k}^{split} = \sum_{l=1}^{L} y_{k,l} C_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) B_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k^B - \Delta^B\right)\right)\times \\
\times\cos\left(\omega_{if} t_{k,l} + \breve{\omega}_k\left(l-1\right)T_d + \breve{\varphi}_k\right) \approx \frac{1}{2}\sum_{l=1}^{L} y_{k,l} C_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right)\times \\
\times\sin\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k\left(l-1\right)T_d + \breve{\varphi}_k - \omega_B\left(\breve{\tau}_k^B - \Delta^B\right)\right) - \\
-\frac{1}{2}\sum_{l=1}^{L} y_{k,l} C_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right)\times \\
\times\sin\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k\left(l-1\right)T_d + \breve{\varphi}_k + \omega_B\left(\breve{\tau}_k^B - \Delta^B\right)\right) = \\
= \frac{-\ Q_{1e,k}^{ps} + Q_{2e,k}^{ps}}{2}.
\end{gathered} \tag{2.4}$$

Выполняя аналогичные преобразования для квадратурной опережающей квадратурной суммы получаем:

$$
\begin{gathered}
Q_{e,k}^{split} = \sum_{l=1}^{L} y_{k,l} C_{k,l} \left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) B_{k,l} \left(t_{k,l} - \left(\breve{\tau}_k^B - \Delta^B\right)\right) \times \\
\times \sin\left(\omega_{if} t_{k,l} + \breve{\omega}_k (l-1) T_d + \breve{\varphi}_k\right) \approx \frac{1}{2} \sum_{l=1}^{L} y_{k,l} C_{k,l} \left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) \times \\
\times \cos\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k (l-1) T_d + \breve{\varphi}_k + \omega_B \left(\breve{\tau}_k^B - \Delta^B\right)\right) - \\
- \frac{1}{2} \sum_{l=1}^{L} y_{k,l} C_{k,l} \left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) \times \\
\times \cos\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k (l-1) T_d + \breve{\varphi}_k - \omega_B \left(\breve{\tau}_k^B - \Delta^B\right)\right) = \\
= \frac{I_{1e,k}^{ps} - I_{2e,k}^{ps}}{2}.
\end{gathered} \tag{2.5}
$$

Аналогично другие компоненты представимы в виде сумм:

$$
\begin{gathered}
I_{e,k}^{split} = \frac{Q_{2e,k}^{ps} - \ Q_{1e,k}^{ps}}{2}, \quad I_{l,k}^{split} = \frac{Q_{2l,k}^{ps} - \ Q_{1l,k}^{ps}}{2}, \\
Q_{e,k}^{split} = \frac{I_{1e,k}^{ps} - \ I_{2e,k}^{ps}}{2}, \quad Q_{l,k}^{split} = \frac{I_{1l,k}^{ps} - \ I_{2l,k}^{ps}}{2},
\end{gathered} \tag{2.6}
$$

где синфазные квадратурные суммы (“+” при Δ для later компонент, “-” – для early):

$$
\begin{gathered}
I_{1^l/_e,k}^{ps} = \sum_{l=1}^{L} y_{k,l} C \left(t_{k,l} - \left(\breve{\tau}_k^C {}^+/_- \Delta^C\right)\right) \times \\
\times \cos\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k (l-1) T_d + \breve{\varphi}_k + \omega_B \left(\breve{\tau}_k^B {}^+/_- \Delta^B\right)\right),
\end{gathered}
$$

$$
\begin{gathered}
I_{2^l/_e,k}^{ps} = \sum_{l=1}^{L} y_{k,l} C \left(t_{k,l} - \left(\breve{\tau}_k^C {}^+/_- \Delta^C\right)\right) \times \\
\times \cos\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k (l-1) T_d + \breve{\varphi}_k - \omega_B \left(\breve{\tau}_k^B {}^+/_- \Delta^B\right)\right),
\end{gathered} \tag{2.7}
$$

а в квадратурных суммах отличается лишь тригонометрическая функция:

$$
\begin{gathered}
Q_{1^l/_e,k}^{ps} = \sum_{l=1}^{L} y_{k,l} C \left(t_{k,l} - \left(\breve{\tau}_k^C {}^+/_- \Delta^C\right)\right) \times \\
\times \sin\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k (l-1) T_d + \breve{\varphi}_k + \omega_B \left(\breve{\tau}_k^B {}^+/_- \Delta^B\right)\right),
\end{gathered}
$$

$$Q^{ps}_{2^{l}/_{e},k} = \sum_{l=1}^{L} y_{k,l} C \left(t_{k,l} - \left(\breve{\tau}_k^{C} {}^{+}/_{-} \Delta^{C} \right) \right) \times$$
$$\times \sin \left(\left(\omega_{if} + \omega_B \right) t_{k,l} + \breve{\omega}_k \left(l - 1 \right) T_d + \breve{\varphi}_k - \omega_B \left(\breve{\tau}_k^{B} {}^{+}/_{-} \Delta^{B} \right) \right). \quad (2.8)$$

Здесь и далее для краткости записи выражений для опережающей (early – e) и запаздывающей (late – l) компонент будет использоваться символ дробной черты. Так, обозначению " ${}^{l}/_{e}$ " в нижнем индексе соответствуют обозначения в выражении " ${}^{+}/_{-}$ ". Такая запись означает, что для запаздывающей компоненты l в выражении используются знаки "$+$"или "$-$ находящиеся над чертой, в данном случае "$+$ а опережающей компоненте соответствует знак, находящийся под чертой, в данном примере "$-$".

Выражения (2.6) - (2.8) позволяют рассчитать корреляционные суммы, не используя цифровую поднесущую в опорном сигнале. Но они всё ещё трудно реализуемы аппаратно – в опорном сигнале для опережающей и запаздывающей компонент используются разные фазы гармонических колебаний, что потребует реализации для них разных цифровых генераторов синусоиды (от англ. NCO, numerically-controlled oscillator). Слагаемые $\pm\omega_B\Delta^B$, отличающие фазы этих компонент, постоянны на интервале коррелирования, поэтому необходимый фазовый сдвиг может быть легко внесен с помощью программных фазовращателей уже на выходе коррелятора. В эти же фазовращатели может быть вынесен сдвиг $\pm\omega_B \breve{\tau}_k^{B}$, что позволит разомкнуть обратную связь от системы слежения за задержкой к NCO коррелятора.

С учетом описанных выше модификаций запишем корреляционные суммы на выходе фазовращателей:

$$\begin{gathered}
I^{ps}_{1^l/_e,k} = \sum_{l=1}^{L} y_{k,l} C\left(t_{k,l} - \left(\breve{\tau}_k^{C} {}^{+}/_{-}\Delta^C\right)\right) \times \\
\times \cos\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \left(\breve{\tau}_k^{B} {}^{+}/_{-}\Delta^B\right)\right) = \\
= \sum_{l=1}^{L} y_{k,l} C\left(t_{k,l} - \left(\breve{\tau}_k^{C} {}^{+}/_{-}\Delta^C\right)\right) [\, \cos\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right) \times \\
\times \cos\left(\omega_B \left(\breve{\tau}_k^{B} {}^{+}/_{-}\Delta^B\right)\right) - \sin\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right) \times \\
\times \sin\left(\omega_B \left(\breve{\tau}_k^{B} {}^{+}/_{-}\Delta^B\right)\right) \,] = \\
= \cos\left(\omega_B \left(\breve{\tau}_k^{B} {}^{+}/_{-}\Delta^B\right)\right) I^{reg}_{1^l/_e,k} - \sin\left(\omega_B \left(\breve{\tau}_k^{B} {}^{+}/_{-}\Delta^B\right)\right) Q^{reg}_{1^l/_e,k}, \quad (2.9)
\end{gathered}$$

где $I^{reg}_{1l/e,k}$, $Q^{reg}_{1l/e,k}$ - корреляционные суммы, рассчитанные аппаратным традиционным BPSK коррелятором:

$$\begin{gathered}
I^{reg}_{1^l/_e,k} = \sum_{l=1}^{L} y_{k,l} C\left(t_{k,l} - \left(\breve{\tau}_k^{C} {}^{+}/_{-}\Delta^C\right)\right) \cos\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right), \\
Q^{reg}_{1^l/_e,k} = \sum_{l=1}^{L} y_{k,l} C\left(t_{k,l} - \left(\breve{\tau}_k^{C} {}^{+}/_{-}\Delta^C\right)\right) \sin\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right), \\
I^{reg}_{2^l/_e,k} = \sum_{l=1}^{L} y_{k,l} C\left(t_{k,l} - \left(\breve{\tau}_k^{C} {}^{+}/_{-}\Delta^C\right)\right) \cos\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right), \\
Q^{reg}_{2^l/_e,k} = \sum_{l=1}^{L} y_{k,l} C\left(t_{k,l} - \left(\breve{\tau}_k^{C} {}^{+}/_{-}\Delta^C\right)\right) \sin\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k\right). \\
(2.10)
\end{gathered}$$

Аналогично запишем квадратурную сумму:

$$\begin{gathered}
Q^{ps}_{1^l/_e,k} = \sum_{l=1}^{L} y_{k,l} C\left(t_{k,l} - \left(\breve{\tau}_k^{C} {}^{+}/_{-}\Delta^C\right)\right) \times \\
\times \sin\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \left(\breve{\tau}_k^{B} {}^{+}/_{-}\Delta^B\right)\right) = \\
= \sin\left(\omega_B \left(\breve{\tau}_k^{B} {}^{+}/_{-}\Delta^B\right)\right) I^{reg}_{1l/e,k} + \cos\left(\omega_B \left(\breve{\tau}_k^{B} {}^{+}/_{-}\Delta^B\right)\right) Q^{reg}_{1l/e,k}. \quad (2.11)
\end{gathered}$$

Обобщим на остальные корреляционные суммы:

$$\begin{aligned}
I^{ps}_{1^l/_e,k} &= +\cos\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right) I^{reg}_{1^l/_e,k} - \sin\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right) Q^{reg}_{1^l/_e,k},\\
Q^{ps}_{1^l/_e,k} &= +\sin\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right) I^{reg}_{1^l/_e,k} + \cos\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right) Q^{reg}_{1^l/_e,k},\\
I^{ps}_{2^l/_e,k} &= +\cos\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right) I^{reg}_{2^l/_e,k} + \sin\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right) Q^{reg}_{2^l/_e,k},\\
Q^{ps}_{2^l/_e,k} &= -\sin\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right) I^{reg}_{2^l/_e,k} + \cos\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right) Q^{reg}_{2^l/_e,k}.
\end{aligned} \tag{2.12}$$

Посмотрим на полученные выражения для расчета корреляционных сумм на выходе фазовращателей. Обратим внимание на то, что теперь корреляцонные суммы включают в себя корреляционные суммы традиционных BPSK корреляторов $I^{reg}_{l/_e,k}$, а также вносимый фазовый сдвиг. Что позволяет при обработке сигналов с BOC модуляцией отказаться от создания специализированных каналов коррелятора.

С учетом модификаций запишем выражения сплит-корреляционных сумм (2.6) приобретают форму:

$$\begin{aligned}
I^{split}_{l/_e,k} = \frac{1}{2}\Bigg(&-\sin\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right)\left(I^{reg}_{1^l/_e,k} + I^{reg}_{2^l/_e,k}\right) - \\
&- \cos\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right)\left(Q^{reg}_{1^l/_e,k} - Q^{reg}_{2^l/_e,k}\right)\Bigg)
\end{aligned} \tag{2.13}$$

$$\begin{aligned}
Q^{split}_{l/_e,k} = \frac{1}{2}\Bigg(&+\cos\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right)\left(I^{reg}_{1^l/_e,k} - I^{reg}_{2^l/_e,k}\right) - \\
&- \sin\left(\omega_B\left(\breve{\tau}^{B}_{k}{}^{+}/_{-}\Delta^B\right)\right)\left(Q^{reg}_{1^l/_e,k} + Q^{reg}_{2^l/_e,k}\right)\Bigg)
\end{aligned}$$

Выражения (2.13) позволяют рассчитать выходной сигнал NELP дискриминатора в приближении (2.3), используя пару традиционных корреляционных каналов, рассчитанных на обработку только BPSK, но не BOC сигналов (см. рис. 2.1).

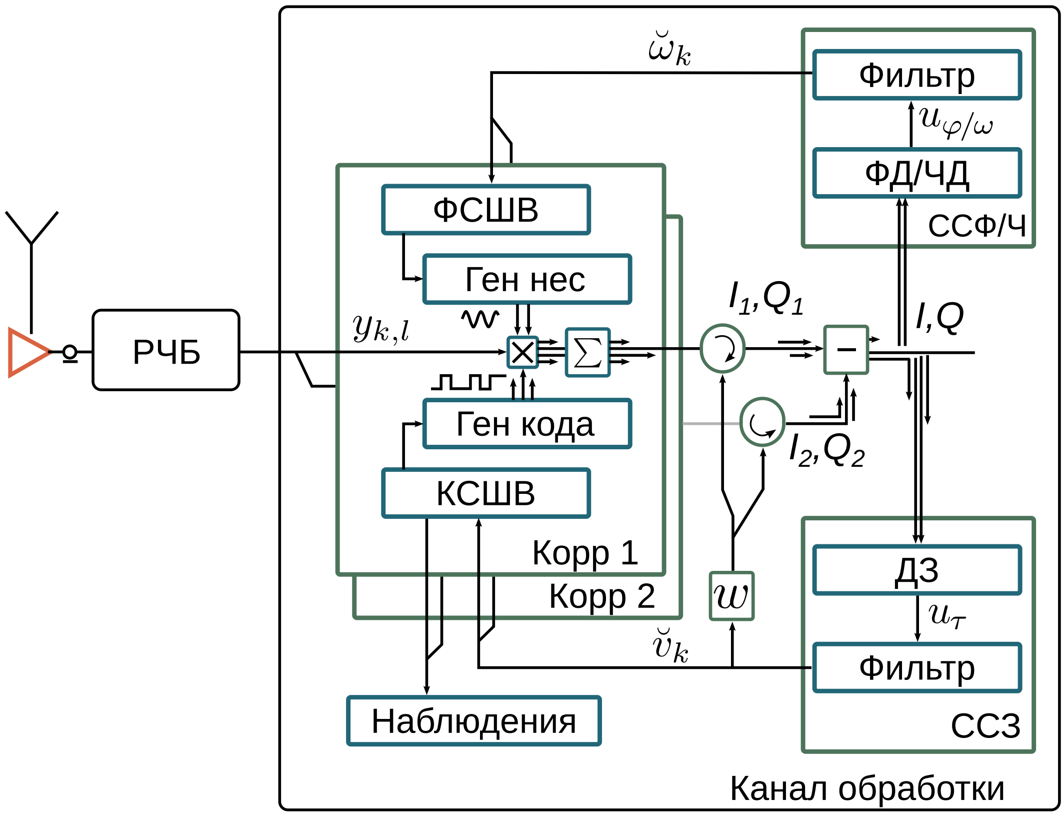

Рисунок 2.1 — Схема обработки сигнала с цифровой поднесущей с использованием двух каналов коррелятора

На рисунке 2.1 РЧБ – радиочастотный блок, ФСШВ – фазовая сигнальная шкала времени, КСШВ – кодовая сигнальная шкала времени, ФД – фазовый дискриминатор, ЧД – частотный дискриминатор, ДЗ – дискриминатор задержки, ССФ, ССЗ, ССЧ – системы слежения за фазой, задержкой, частотой соответственно.

3 Аналитический расчет характеристик дискриминатора задержки

Найдем аналитические выражения для дискриминационной и флуктуационных характеристик дискриминатора задержки и рассмотрим, как на них повлияло приближение (2.3). Для расчета характеристик требуется найти статистические эквиваленты традиционных корреляторов при обработке BOC сигнала.

3.1 Статистический эквивалент коррелятора

Сигнал $S_{k,l}$ наблюдается на фоне аддитивного белого гауссовского дискретного шума $n_{k,l}$ с дисперсией σ_n^2 и нулевым математическим ожиданием:

$$y_{k,l} = S_{k,l}\left(\tau_k,\omega_k,\varphi_k\right) + n_{k,l} \tag{3.1}$$

Тогда синфазные и квадратурные корреляционные суммы распадаются на систематические и флуктуационные составляющие:

$$I_{l/e,k} = M\left[I_{l/e,k}\right] + n_{Ie/l,k},\;\; Q_{l/e,k} = M\left[Q_{l/e,k}\right] + n_{Ie/l,k} \tag{3.2}$$

3.1.1 Статистический эквивалент сигналов на выходе фазовращателя

Рассмотрим систематическую составляющую опережающей синфазной корреляционной суммы первого канала коррелятора:

$$\begin{aligned}
&M\left[I_{1e,k}^{ps}\right] = M\left[\sum_{l=1}^{L} y_{k,l} C\left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) \times\right.\\
&\left.\times \cos\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B\left(\breve{\tau}_k^B - \Delta^B\right)\right)\right] =\\
&= M\left[\sum_{l=1}^{L} AC\left(t_{k,l} - \tau_k\right) C_{k,l}\left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) \times\right.\\
&\times B\left(t_{k,l} - \tau_k\right) \cos\left(\omega_{if} t_{k,l} + \omega_k \left(l-1\right) T_d + \varphi_k\right) \times\\
&\left.\times \cos\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B\left(\breve{\tau}_k^B - \Delta^B\right)\right)\right].
\end{aligned}$$

Пренебрегая гармоникой с удвоенной промежуточной частотой, получаем

$$M\left[I_{1e,k}^{ps}\right] \approx \frac{1}{2} M\left[\sum_{l=1}^{L} AC\left(t_{k,l}-\tau_k\right) C_{k,l}\left(t_{k,l}-\left(\breve{\tau}_k^C-\Delta^C\right)\right) \times \right.$$
$$\left. \times B\left(t_{k,l}-\tau_k\right) \cos\left(\omega_B t_{k,l}+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k-\omega_B\left(\breve{\tau}_k^B-\Delta^B\right)\right)\right].$$

Представим цифровую поднесущую сигнала в виде ряда Фурье:

$$B\left(t_{k,l}-\tau_k\right) = \frac{4}{\pi}\sum_{n=1}^{\infty}\frac{\sin\left(\left(2n-1\right)\omega_B\left(t_{k,l}-\tau_k\right)\right)}{2n-1} \tag{3.3}$$

Рассмотрим отдельный член ряда:

$$\begin{aligned}
&\sin\left(\left(2n-1\right)\omega_B\left(t_{k,l}-\tau_k\right)\right)\cos\left(\omega_B t_{k,l}+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k-\omega_B\left(\breve{\tau}_k^B-\Delta^B\right)\right) = \\
&= \frac{1}{2}\sin\left(\omega_B t_{k,l}+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k-\omega_B\left(\breve{\tau}_k^B-\Delta^B\right)+\left(2n-1\right)\omega_B\left(t_{k,l}-\tau_k\right)\right) - \\
&- \frac{1}{2}\sin\left(\omega_B t_{k,l}+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k-\omega_B\left(\breve{\tau}_k^B-\Delta^B\right)-\left(2n-1\right)\omega_B\left(t_{k,l}-\tau_k\right)\right) = \\
&= \frac{1}{2}\sin\left(2n\omega_B\left(t_{k,l}-\tau_k\right)+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right) - \\
&- \frac{1}{2}\sin\left(-\left(2n-2\right)\omega_B\left(t_{k,l}-\tau_k\right)+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right) = \\
&= \frac{1}{2}\sin\left(\left(\delta\omega_k+2n\omega_B\right)\left(l-1\right)T_d+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)+2n\omega_B\left(t_{k,1}-\tau_k\right)\right) - \\
&- \frac{1}{2}\sin\left(\left(\delta\omega_k-\left(2n-2\right)\omega_B\right)\left(l-1\right)T_d+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right) - \right. \\
&\left. -\left(2n-2\right)\omega_B\left(t_{k,1}-\tau_k\right)\right).
\end{aligned}$$

Тогда

$$M\left[I_{1e,k}^{ps}\right] \approx \frac{1}{\pi} M\left[\sum_{l=1}^{L} AC\left(t_{k,l}-\tau_k\right) C_{k,l}\left(t_{k,l}-\left(\breve{\tau}_k^C-\Delta^C\right)\right) \times \right. \tag{3.4}$$
$$\sum_{n=1}^{\infty}\frac{1}{2n-1}\ \sin\left(\left(2n-2\right)\omega_B\left(t_{k,l}-\tau_k\right)-\delta\omega_k\left(l-1\right)T_d-\delta\varphi_k-\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right) +$$
$$\left. +\frac{1}{2n-1}\sin\left(2n\omega_B\left(t_{k,l}-\tau_k\right)+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)\right].$$

Члены ряда для $n > 1$ являются гармоническими колебаниями с кратными ω_B частотами. Так как период частоты ω_B значительно меньше интервала накопления в корреляторе, то при интегрировании они подавляются. Тогда систематическая составляющая опережающей синфазной корреляционной суммы первого канала коррелятора принимает вид:

$$M\left[I_{1e,k}^{ps}\right] \approx -\frac{1}{\pi} M\left[\sum_{l=1}^{L} AC\left(t_{k,l}-\tau_k\right) C_{k,l}\left(t_{k,l}-\left(\breve{\tau}_k^C-\Delta^C\right)\right) \times\right. \tag{3.5}$$
$$\left.\times \sin\left(\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)\right] \approx$$
$$\approx -\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C+\Delta^C\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\sin\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right).$$

Проведем аналогичные вычисления для систематической составляющей опережающей синфазной корреляционной суммы для второго канала коррелятора:

$$M\left[I_{2e,k}^{ps}\right] \approx \frac{1}{2} M\left[\sum_{l=1}^{L} AC\left(t_{k,l}-\tau_k\right) C_{k,l}\left(t_{k,l}-\left(\breve{\tau}_k^C-\Delta^C\right)\right) \times\right. \tag{3.6}$$
$$\left.\times B\left(t_{k,l}-\tau_k\right)\cos\left(-\omega_B t_{k,l}+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k+\omega_B\left(\breve{\tau}_k^B-\Delta^B\right)\right)\right].$$

Рассмотрим отдельный член ряда:

$$\begin{aligned}
&\sin\left(\left(2n-1\right)\omega_B\left(t_{k,l}-\tau_k\right)\right)\cos\left(-\omega_B\left(t_{k,l}-\left(\breve{\tau}_k^B-\Delta^B\right)\right)+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k\right)=\\
&=\frac{1}{2}\sin\left(\left(2n-1\right)\omega_B\left(t_{k,l}-\tau_k\right)+\omega_B\left(t_{k,l}-\left(\breve{\tau}_k^B-\Delta^B\right)\right)-\delta\omega_k\left(l-1\right)T_d-\delta\varphi_k\right)+\\
&+\frac{1}{2}\sin\left(\left(2n-1\right)\omega_B\left(t_{k,l}-\tau_k\right)-\omega_B\left(t_{k,l}-\left(\breve{\tau}_k^B-\Delta^B\right)\right)+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k\right)=\\
&=\frac{1}{2}\sin\left(2n\omega_B\left(t_{k,l}-\tau_k\right)-\delta\omega_k\left(l-1\right)T_d-\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)+\\
&+\frac{1}{2}\sin\left(\left(2n-2\right)\omega_B\left(t_{k,l}-\tau_k\right)+\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k-\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right).
\end{aligned}$$

Тогда систематическая составляющая опережающей синфазной корреляционной суммы для второго канала коррелятора принимает вид:

$$M\left[I_{2e,k}^{ps}\right] \approx \frac{1}{\pi} M\left[\sum_{l=1}^{L} AC\left(t_{k,l}-\tau_k\right) C_{k,l}\left(t_{k,l}-\left(\breve{\tau}_k^C-\Delta^C\right)\right) \times\right. \tag{3.7}$$
$$\left.\times \sin\left(\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k-\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)\right] \approx$$
$$\approx \frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C+\Delta^C\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\sin\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k-\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right).$$

Рассмотрим систематическую составляющую опережающей квадратурной корреляционной суммы первого канала коррелятора:

$$M\left[Q_{1e,k}^{ps}\right] \approx -\frac{1}{2} M\left[\sum_{l=1}^{L} AC\left(t_{k,l}-\tau_k\right) C_{k,l}\left(t_{k,l}-\left(\breve{\tau}_k^C-\Delta^C\right)\right) \times\right. \tag{3.8}$$
$$\left.\times B\left(t_{k,l}-\tau_k\right) \sin\left(\omega_B\left(t_{k,l}-\left(\breve{\tau}_k^B-\Delta^B\right)\right)+\delta\omega_k(l-1)T_d+\delta\varphi_k\right)\right].$$

Рассмотрим отдельный член ряда:

$$\sin\left((2n-1)\omega_B\left(t_{k,l}-\tau_k\right)\right)\sin\left(\omega_B\left(t_{k,l}-\left(\breve{\tau}_k^B-\Delta^B\right)\right)+\delta\omega_k(l-1)T_d+\delta\varphi_k\right)=$$
$$=\frac{1}{2}\cos\left(\omega_B\left(\delta\tau_k^B+\Delta^B\right)+\delta\omega_k(l-1)T_d+\delta\varphi_k-(2n-2)\omega_B\left(t_{k,l}-\tau_k\right)\right)$$
$$-\frac{1}{2}\cos\left(\delta\omega_k(l-1)T_d+\delta\varphi_k+2n\omega_B\left(t_{k,l}-\tau_k\right)+\omega_B\left(\delta\tau_k+\Delta^B\right)\right)$$

Учитывая это систематическая составляющая опережающей квадратурной корреляционной суммы первого канала коррелятора:

$$M\left[Q_{1e,k}^{ps}\right] \approx -\frac{1}{\pi} M\left[\sum_{l=1}^{L} AC\left(t_{k,l}-\tau_k\right) C_{k,l}\left(t_{k,l}-\left(\breve{\tau}_k^C-\Delta^C\right)\right) \times\right. \tag{3.9}$$
$$\left.\times\cos\left(\delta\omega_k(l-1)T_d+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)\right] \approx$$
$$\approx -\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C+\Delta^C\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\cos\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right).$$

Проведем расчет систематической составляющей опережающей квадратурной корреляционной суммы второго канала коррелятора:

$$M\left[Q_{2e,k}^{ps}\right] \approx -\frac{1}{2} M\left[\sum_{l=1}^{L} AC\left(t_{k,l}-\tau_k\right) C_{k,l}\left(t_{k,l}-\left(\breve{\tau}_k^C-\Delta^C\right)\right) \times\right. \tag{3.10}$$
$$\left.\times B\left(t_{k,l}-\tau_k\right) \sin\left(-\omega_B\left(t_{k,l}-\left(\breve{\tau}_k^B-\Delta^B\right)\right)+\delta\omega_k(l-1)T_d+\delta\varphi_k\right)\right].$$

Рассмотрим отдельный член ряда:

$$\sin\left((2n-1)\omega_B\left(t_{k,l}-\tau_k\right)\right)\sin\left(-\omega_B\left(t_{k,l}-\left(\breve{\tau}_k^B-\Delta^B\right)\right)+\delta\omega_k(l-1)T_d+\delta\varphi_k\right)=$$
$$=\frac{1}{2}\cos\left(\delta\omega_k(l-1)T_d+\delta\varphi_k-2n\omega_B\left(t_{k,l}-\tau_k\right)-\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)-$$
$$-\frac{1}{2}\cos\left((2n-2)\omega_B\left(t_{k,l}-\tau_k\right)-\omega_B\left(\delta\tau_k^B+\Delta^B\right)+\delta\omega_k(l-1)T_d+\delta\varphi_k\right).$$

Тогда систематическая составляющая опережающей квадратурной суммы второго канала коррелятора принимает вид:

$$M\left[Q_{2e,k}^{ps}\right] \approx +\frac{1}{\pi}M\left[\sum_{l=1}^{L} AC\left(t_{k,l}-\tau_k\right)C_{k,l}\left(t_{k,l}-\left(\breve{\tau}_k^C-\Delta^C\right)\right)\times\right.$$
$$\left.\times\cos\left(\delta\omega_k\left(l-1\right)T_d+\delta\varphi_k-\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)\right] \approx$$
$$\approx +\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C+\Delta^C\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\cos\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k-\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right). \tag{3.11}$$

Проведем аналогичные преобразования и запишем выражения, описывающие систематические составляющие запаздывающих синфазных и квадратурных корреляционных сумм для двух каналов корреляторов:

$$M\left[I_{1l,k}^{ps}\right] = -\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C-\Delta^C\right)\times$$
$$\times\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\sin\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k+\omega_B\left(\delta\tau_k^B-\Delta^B\right)\right), \tag{3.12}$$
$$M\left[I_{2l,k}^{ps}\right] = +\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C-\Delta^C\right)\times$$
$$\times\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\sin\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k-\omega_B\left(\delta\tau_k^B-\Delta^B\right)\right),$$
$$M\left[Q_{1l,k}^{ps}\right] = -\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C-\Delta^C\right)\times$$
$$\times\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\cos\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k+\omega_B\left(\delta\tau_k^B-\Delta^B\right)\right),$$
$$M\left[Q_{2l,k}^{ps}\right] = +\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C-\Delta^C\right)\times$$
$$\times\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\cos\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k-\omega_B\left(\delta\tau_k^B-\Delta^B\right)\right).$$

3.1.2 Статистический эквивалент сплит-коррелятора

Найдем статистический эквивалент рассматриваемого в работе сплит-коррелятора. Аналогично 3.2 синфазные и квадратурные корреляционные суммы распадаются на систематические и флуктуационные составляющие.

$$I_{l/e,k}^{split} = M\left[I_{l/e,k}^{split}\right]+n_{Ie/l,k}^{split},\;\; Q_{l/e,k}^{split} = M\left[Q_{l/e,k}^{split}\right]+n_{Ie/l,k}^{split}, \tag{3.13}$$

Сначала рассмотрим систематические составляющие сплит корреляционных сумм. На основе (3.5)-(3.12) найдем математическое ожидание комбинированной запаздывающей синфазной корреляционной суммы:

$$M\left[I_{l,k}^{split}\right] = \frac{M\left[Q_{2l,k}^{ps}\right] - M\left[Q_{1l,k}^{ps}\right]}{2} = \frac{1}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C - \Delta^C\right)\mathrm{sinc}\left(\frac{\delta\omega_k T}{2}\right)\times$$
$$\times\left[\cos\left(\frac{\delta\omega_k T}{2} + \delta\varphi_k + \omega_B\left(\delta\tau_k^B - \Delta^B\right)\right) + \right.$$
$$\left. + \cos\left(\frac{\delta\omega_k T}{2} + \delta\varphi_k - \omega_B\left(\delta\tau_k^B - \Delta^B\right)\right)\right] =$$
$$= \frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C - \Delta^C\right)\cos\left(\omega_B\left(\delta\tau_k^B - \Delta^B\right)\right)\mathrm{sinc}\left(\frac{\delta\omega_k T}{2}\right)\cos\left(\frac{\delta\omega_k T}{2} + \delta\varphi_k\right) \quad (3.14)$$

Систематическая составляющая комбинированной запаздывающей квадратурной корреляционной суммы:

$$M\left[Q_{l,k}^{split}\right] = \frac{M\left[I_{1l,k}^{ps}\right] - M\left[I_{2l,k}^{ps}\right]}{2} = -\frac{1}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C - \Delta^C\right)\mathrm{sinc}\left(\frac{\delta\omega_k T}{2}\right)\times$$
$$\times\left[\sin\left(\frac{\delta\omega_k T}{2} + \delta\varphi_k + \omega_B\left(\delta\tau_k^B - \Delta^B\right)\right) + \right.$$
$$\left. + \sin\left(\frac{\delta\omega_k T}{2} + \delta\varphi_k - \omega_B\left(\delta\tau_k^B - \Delta^B\right)\right)\right] =$$
$$= -\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^C - \Delta^C\right)\cos\left(\omega_B\left(\delta\tau_k^B - \Delta^B\right)\right)\mathrm{sinc}\left(\frac{\delta\omega_k T}{2}\right)\sin\left(\frac{\delta\omega_k T}{2} + \delta\varphi_k\right). \quad (3.15)$$

Аналогично найдем выражения для опережающих компонент:

$$M\left[I_{e,k}^{split}\right] = \frac{M\left[Q_{2e,k}^{ps}\right] - M\left[Q_{1e,k}^{ps}\right]}{2} = \frac{AL}{2\pi}\rho\left(\delta\tau^C + \Delta^C\right)\mathrm{sinc}\left(\frac{\delta\omega_k T}{2}\right)\times \quad (3.16)$$
$$\times\left[\cos\left(\frac{\delta\omega_k T}{2} + \delta\varphi_k + \omega_B\left(\delta\tau_k^B + \Delta^B\right)\right) + \right.$$
$$\left. + \cos\left(\frac{\delta\omega_k T}{2} + \delta\varphi_k - \omega_B\left(\delta\tau_k^B + \Delta^B\right)\right)\right] =$$
$$= \frac{AL}{\pi}\rho\left(\delta\tau^C + \Delta^C\right)\cos\left(\omega_B\left(\delta\tau_k^B + \Delta^B\right)\right)\mathrm{sinc}\left(\frac{\delta\omega_k T}{2}\right)\cos\left(\frac{\delta\omega_k T}{2} + \delta\varphi_k\right)$$

$$M\left[Q_{e,k}^{split}\right]=\frac{M\left[I_{1e,k}^{ps}\right]-M\left[I_{2e,k}^{ps}\right]}{2}=\frac{AL}{2\pi}\rho\left(\delta\tau^{C}+\Delta^{C}\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\times$$
$$\times\left[\sin\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k+\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)+\right.$$
$$\left.+\sin\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k-\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)\right]=$$
$$=-\frac{AL}{\pi}\rho\left(\delta\tau^{C}+\Delta^{C}\right)\cos\left(\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\sin\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k\right) \quad (3.17)$$

Теперь перейдем к расчету флуктуационых составляющих корреляционных сумм.

Рассчитаем дисперсию флуктуационной составляющей запаздывающей синфазной компоненты:

$$M\left[n_{I_{l,k}}^{split^2}\right]=M\left[\frac{1}{4}\left(n_{Q_{2l,k}}^{ps}-\ n_{Q_{1l,k}}^{ps}\right)^2\right]=\frac{1}{4}M\left[\sum_{l=1}^{L}n_l^2\left(\breve{S}_{Q_{2l,l}^{ps}}-\ \breve{S}_{Q_{1l,l}^{ps}}\right)^2\right]$$

где $\breve{S}_{Q_{2l,l}^{ps}}$,$\breve{S}_{Q_{1l,l}^{ps}}$ - опорные сигналы корреляторов, рассчитывающих суммы Q_{2l} и Q_{1l} соответственно.

Рассчитаем сначала разность опорных сигналов корреляторов:

$$\breve{S}_{Q_{2l,l}^{ps}}-\ \breve{S}_{Q_{1l,l}^{ps}}=C\left(t_{k,l}-\left(\breve{\tau}_k^C+\Delta^C\right)\right)\times$$
$$\times\left(\sin\left(\left(\omega_{if}+\omega_B\right)t_{k,l}+\breve{\omega}_k\left(l-1\right)T_d+\breve{\varphi}_k-\omega_B\left(\breve{\tau}_k^B+\Delta^B\right)\right)-\right.$$
$$\left.-\sin\left(\left(\omega_{if}-\omega_B\right)t_{k,l}+\breve{\omega}_k\left(l-1\right)T_d+\breve{\varphi}_k+\omega_B\left(\breve{\tau}_k^B+\Delta^B\right)\right)\right)=$$
$$=2C\left(t_{k,l}-\left(\breve{\tau}_k^C+\Delta^C\right)\right)\times$$
$$\times\cos\left(\omega_{if}t_{k,l}+\breve{\omega}_k\left(l-1\right)T_d+\breve{\varphi}_k\right)\cos\left(\omega_B\left(t_{k,l}-\left(\breve{\tau}_k^B+\Delta^B\right)\right)\right)$$

Тогда квадрат разности опорных сигналов корреляторов образует:

$$\left(\breve{S}_{Q_{2l,l}^{ps}}-\ \breve{S}_{Q_{1l,l}^{ps}}\right)^2=$$
$$=4\cos^2\left(\omega_{if}t_{k,l}+\breve{\omega}_k\left(l-1\right)T_d+\breve{\varphi}_k\right)\cos^2\left(\omega_B\left(t_{k,l}-\left(\breve{\tau}_k^B+\Delta^B\right)\right)\right)$$

Получим, что дисперсия флуктуационной составляющей запаздывающей синфазной компоненты принимает вид:

$$M\left[n_{I_{l,k}}^{spl2}\right] =$$
$$= \frac{1}{4}\sigma_n^2 \sum_{l=1}^{L} 4\cos^2\left(\omega_{if} t_{k,l} + \breve{\omega}_k (l-1) T_d + \breve{\varphi}_k\right) \cos^2\left(\omega_B \left(t_{k,l} - \left(\breve{\tau}_k^B + \Delta^B\right)\right)\right) =$$
$$= \frac{\sigma_n^2 L}{4}$$

Дисперсии остальных флуктуационных составляющих рассчитываются аналогично. Получаем, что флуктуационные составляющие $n_{Ie,k}^{split}$, $n_{Qe,k}^{split}$, $n_{Il,k}^{split}$, $n_{Ql,k}^{split}$ - нормальные случайные величины с нулевым математическим ожиданием и дисперсией $\sigma_{IQ}^{split^2} = \sigma_{IQ}^2/2 = \sigma_n^2 L/4$. Каждая из величин образует белый шум для разных k. Взаимные дисперсии флуктуационных составляющих синфазных и квадратурных клмпонент:

$$M\left[n_{Ie,k}^{split} n_{Qe,k}^{split}\right] = M\left[n_{Il,k}^{split} n_{Ql,k}^{split}\right] = M\left[n_{Ie,k}^{split} n_{Ql,k}^{split}\right] = M\left[n_{Il,k}^{split} n_{Qe,k}^{split}\right] = 0 \quad (3.18)$$

Рассчитаем взаимную дисперсию синфазных опережающей и запаздывающей компонент:

$$M\left[n_{Ie,k}^{split} n_{Il,k}^{split}\right] = \frac{1}{4} M\left[\left(n_{Q_{2l,k}}^{ps} - n_{Q_{1l,k}}^{ps}\right)\left(n_{Q_{2e,k}}^{ps} - n_{Q_{1e,k}}^{ps}\right)\right] = \quad (3.19)$$
$$= \frac{1}{4} M\left[n_{Q_{2l,k}}^{ps} n_{Q_{2e,k}}^{ps} - n_{Q_{2l,k}}^{ps} n_{Q_{1e,k}}^{ps} - n_{Q_{1l,k}}^{ps} n_{Q_{2e,k}}^{ps} + n_{Q_{1l,k}}^{ps} n_{Q_{1e,k}}^{ps}\right] =$$
$$= \frac{1}{4} M\left[\sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{2l,i}} n_j \breve{S}_{Q_{2e,j}} - \sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{2l,i}} n_j \breve{S}_{Q_{1e,j}} - + \right.$$
$$\left. - \sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{1l,i}} n_j \breve{S}_{Q_{2e,j}} + \sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{1l,i}} n_j \breve{S}_{Q_{1e,j}}\right] =$$
$$= \frac{1}{4}\left(M\left[\sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{2l,i}} n_j \breve{S}_{Q_{2e,j}}\right] - M\left[\breve{S}_{Q_{2l,i}} n_j \breve{S}_{Q_{1e,j}}\right] - \right.$$
$$\left. M\left[\sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{1l,i}} n_j \breve{S}_{Q_{2e,j}}\right] + +M\left[\sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{1l,i}} n_j \breve{S}_{Q_{1e,j}}\right]\right)$$

Найдем каждое математическое ожидание отдельно.

$$M\left[\sum_{i=1}^{L} n_i \breve{S}_{Q_{2l,i}} \sum_{j=1}^{L} n_j \breve{S}_{Q_{2e,j}}\right] = M\left[\sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{2l,i}} n_j \breve{S}_{Q_{2e,j}}\right] = M\left[\sum_{l=1}^{L} n_l^2 \breve{S}_{Q_{2l,l}} \breve{S}_{Q_{2e,l}}\right] = \quad (3.20)$$

$$= \sigma_n^2 \left(\sum_{l=1}^{L} C\left(t_{k,l} - \left(\breve{\tau}_k^C + \Delta^C\right)\right) \sin\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k - \omega_B \left(\breve{\tau}_k^B + \Delta^B\right)\right) \times\right.$$

$$\left.\times\, C\left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) \sin\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k - \omega_B \left(\breve{\tau}_k^B - \Delta^B\right)\right)\right) =$$

$$= \frac{1}{2}\sigma_n^2 \rho\left(2\Delta^C\right) \sum_{l=1}^{L} \left(\cos\left(2\omega_B \Delta^B\right) - \cos\left(2\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k - \omega_B \breve{\tau}_k^B\right)\right)\right) =$$

$$= \frac{1}{2}\sigma_n^2 \rho\left(2\Delta^C\right) \sum_{l=1}^{L} \cos\left(2\omega_B \Delta^B\right) - \sum_{l=1}^{L} \left(\cos^2\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k - \omega_B \breve{\tau}_k^B\right) -\right.$$

$$\left.-\, \sin^2\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k - \omega_B \breve{\tau}_k^B\right)\right) = \frac{\sigma_n^2 L}{2} \rho\left(2\Delta^C\right) \cos\left(2\omega_B \Delta^B\right)$$

$$M\left[\sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{2l,i}} n_j \breve{S}_{Q_{1e,j}}\right] = M\left[\sum_{l=1}^{L} n_l^2 \breve{S}_{Q_{2l,l}} \breve{S}_{Q_{1e,l}}\right] = \quad (3.21)$$

$$= \sigma_n^2 \left(\sum_{l=1}^{L} C\left(t_{k,l} - \left(\breve{\tau}_k^C + \Delta^C\right)\right) \sin\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k - \omega_B \left(\breve{\tau}_k^B + \Delta^B\right)\right) \times\right.$$

$$\left.\times\, C\left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) \sin\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \left(\breve{\tau}_k^B - \Delta^B\right)\right)\right) =$$

$$= \frac{1}{2}\sigma_n^2 \rho\left(2\Delta^C\right) \sum_{l=1}^{L} \cos\left(2\omega_B \left(t_{k,l} - \breve{\tau}_k^B\right)\right) - \sum \cos\left(2\left(\omega_{if} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k - \omega_B \Delta^B\right)\right) = 0$$

$$M\left[\sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{1l,i}} n_j \breve{S}_{Q_{2e,j}}\right] = M\left[\sum_{l=1}^{L} n_l^2 \breve{S}_{Q_{1l,l}} \breve{S}_{Q_{2e,l}}\right] = \quad (3.22)$$

$$= \sigma_n^2 \left(\sum_{l=1}^{L} C\left(t_{k,l} - \left(\breve{\tau}_k^C + \Delta^C\right)\right) \sin\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \left(\breve{\tau}_k^B + \Delta^B\right)\right) \times\right.$$

$$\left.\times\, C\left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) \sin\left(\left(\omega_{if} + \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k - \omega_B \left(\breve{\tau}_k^B - \Delta^B\right)\right)\right) =$$

$$= \frac{1}{2}\sigma_n^2 \rho\left(2\Delta^C\right) \sum_{l=1}^{L} \cos\left(2\omega_B \left(t_{k,l} - \breve{\tau}_k^B\right)\right) - \sum \cos\left(2\left(\omega_{if} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \Delta^B\right)\right) = 0$$

$$M\left[\sum_{i=1}^{L}\sum_{j=1}^{L} n_i \breve{S}_{Q_{1l,i}} n_j \breve{S}_{Q_{1e,j}}\right] = M\left[\sum_{l=1}^{L} n_l^2 \breve{S}_{Q_{1l,l}} \breve{S}_{Q_{1e,l}}\right] = \tag{3.23}$$

$$= \sigma_n^2 \left(\sum_{l=1}^{L} C\left(t_{k,l} - \left(\breve{\tau}_k^C + \Delta^C\right)\right) \sin\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \left(\breve{\tau}_k^B + \Delta^B\right)\right) \times\right.$$

$$\left.\times\, C\left(t_{k,l} - \left(\breve{\tau}_k^C - \Delta^C\right)\right) \sin\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \left(\breve{\tau}_k^B - \Delta^B\right)\right)\right) =$$

$$= \frac{1}{2}\sigma_n^2 \rho\left(2\Delta^C\right) \sum_{l=1}^{L} \left(\cos\left(2\omega_B \Delta^B\right) - \cos\left(2\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \breve{\tau}_k^B\right)\right)\right) =$$

$$= \frac{1}{2}\sigma_n^2 \rho\left(2\Delta^C\right) \sum_{l=1}^{L} \cos\left(2\omega_B \Delta^B\right) - \sum_{l=1}^{L} \left(\cos^2\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \breve{\tau}_k^B\right) -\right.$$

$$\left.-\sin^2\left(\left(\omega_{if} - \omega_B\right) t_{k,l} + \breve{\omega}_k \left(l-1\right) T_d + \breve{\varphi}_k + \omega_B \breve{\tau}_k^B\right)\right) = \frac{\sigma_n^2 L}{2} \rho\left(2\Delta^C\right) \cos\left(2\omega_B \Delta^B\right)$$

Подставляя 3.20-3.23 в 3.19 получаем, что взаимная дисперсия опережающей и запаздывающей компонент принимает вид:

$$M\left[n_{Ie,k}^{split} n_{Il,k}^{split}\right] = \frac{\sigma_n^2 L}{4} \rho\left(2\Delta^C\right) \cos\left(2\omega_B \Delta^B\right) = \frac{\sigma_{IQ}^2}{2} \rho\left(2\Delta^C\right) \cos\left(2\omega_B \Delta^B\right) =$$
$$= \sigma_{IQ}^{spl2} \rho\left(2\Delta^C\right) \cos\left(2\omega_B \Delta^B\right)$$

Взаимные дисперсии одноименных (либо синфазных, либо квадратурных) компонент рассчитываются аналогично:

$$M\left[n_{Ie,k}^{split} n_{Il,k}^{split}\right] = M\left[n_{Qe,k}^{split} n_{Ql,k}^{split}\right] = \rho_C\left(2\Delta^C\right) \cos\left(2\omega_B \Delta^B\right) \sigma_{IQ}^{spl2}$$

Множитель при взаимной дисперсии далее часто будет встречаться в формулах, поэтому для сокращения записи введем для него обозначение:

$$r_{2\Delta} = \rho\left(2\Delta^C\right) \cos\left(2\omega_B \Delta^B\right).$$

3.1.3 Статистический эквивалент сигналов на входе фазовращателя

Для нужд имитационного моделирования может потребоваться статистическое описание сигналов на входе фазовращателей, т.е. корреляционных сумм, получаемых от каналов коррелятора. Математические ожидания могут быть легко записаны на основе (3.5 - 3.12):

$$
\begin{aligned}
M\left[I^{reg}_{1^e/_l,k}\right] &= -\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^{C+}/_{-}\Delta^{C}\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\sin\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k+\omega_B\tau^B_k\right),\\
M\left[Q^{reg}_{1^e/_l,k}\right] &= -\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^{C+}/_{-}\Delta^{C}\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\cos\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k+\omega_B\tau^B_k\right),\\
M\left[I^{reg}_{2^e/_l,k}\right] &= +\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^{C+}/_{-}\Delta^{C}\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\sin\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k-\omega_B\tau^B_k\right),\\
M\left[Q^{reg}_{2^e/_l,k}\right] &= +\frac{2}{\pi}\frac{AL}{2}\rho\left(\delta\tau^{C+}/_{-}\Delta^{C}\right)\operatorname{sinc}\left(\frac{\delta\omega_k T}{2}\right)\cos\left(\frac{\delta\omega_k T}{2}+\delta\varphi_k-\omega_B\tau^B_k\right).
\end{aligned}
\tag{3.24}
$$

Выражения (3.24) наглядно демонстрируют: информация о задержке сигнала содержится в разности фаз сигналов первого и второго каналов коррелятора, т.е. в разности фаз лепестков BOC сигнала. Чем больше частота поднесущей, тем интенсивнее изменяется разность фаз при изменении задержки. Но ввиду неоднозначности оценок разностей фаз, всё ещё полезна модуляция дальномерным кодом для её разрешения. При актуальных для ГНСС значениях поднесущей частоты, взаимным влиянием шумов в первом и втором каналах коррелятора можно пренебречь. Тогда дисперсии шумов корреляционных сумм:

$$
\begin{aligned}
&M\left[n^{reg}_{Iie,k}n^{reg}_{Iie,k}\right] = M\left[n^{reg}_{Iil,k}n^{reg}_{Iil,k}\right] = \sigma^2_{IQ} = \frac{\sigma^2_n L}{2},\\
&M\left[n^{reg}_{Iie,k}n^{reg}_{Iil,k}\right] = \rho_C\left(2\Delta\right)\sigma^2_{IQ} = \rho_C\left(2\Delta\right)\frac{\sigma^2_n L}{2},\\
&M\left[n^{reg}_{I1e,k}n^{reg}_{I2e,k}\right] = M\left[n^{reg}_{I1l,k}n^{reg}_{I2l,k}\right] = M\left[n^{reg}_{I1e,k}n^{reg}_{I2l,k}\right] = 0.
\end{aligned}
\tag{3.25}
$$

Шумы некоррелированы для разных временных отчетов k.

3.2 Дискриминационная характеристика и её крутизна

Найдем дискриминационную характеристику при применении split-компонент – математическое ожидание выходного сигнала дискриминатора как функции ошибки по параметру слежения (примечание “s” временно опустим, как и индекс времени “k”)

$$U_\tau\left(\delta\tau\right) = M\left[u^s_\tau\left(\delta\tau\right)\right] - ? \tag{3.26}$$

Для расчетов воспользуемся статистическим эквивалентом корреляционных сумм (3.14 - 3.17). Для обозначения операции взятия математического ожидания для краткости будем использовать символ верхней черты.

$$
\begin{gathered}
U_\tau^{split}(\delta\tau) = M\left[u_\tau^s\right] = -M\left[\left(I_e^2 + Q_e^2\right)\right] + M\left[\left(I_l^2 + Q_l^2\right)\right] = \\
= -M\left[\left(\bar{I}_e + n_{Ie}\right)^2\right] - M\left[\left(\bar{Q}_e + n_{Qe}\right)^2\right] + M\left[\left(\bar{I}_l + n_{Il}\right)^2\right] + M\left[\left(\bar{Q}_l + n_{Ql}\right)^2\right] = \\
= \bar{I}_l^2 + \bar{Q}_l^2 - (\bar{I}_e^2 + \bar{Q}_e^2) = \left(\frac{2}{\pi}\frac{AL}{2}\right)^2 \operatorname{sinc}^2\left(\frac{\delta\omega_k T}{2}\right) \times \\
\times \left[\rho_C^2\left(\delta\tau^C - \Delta^C\right)\cos^2\left(\omega_B\left(\delta\tau_k^B - \Delta^B\right)\right) - \rho_C^2\left(\delta\tau^C + \Delta^C\right)\cos^2\left(\omega_B\left(\delta\tau_k^B + \Delta^B\right)\right)\right].
\end{gathered} \tag{3.27}
$$

Выражение (3.27) можно записать через отношение сигнал/шум

$$
\begin{gathered}
U_\tau^{split}(\delta\tau) = \ 2\sigma_{IQ}^2\left(\frac{2}{\pi}\right)^2 q_{c/n0} T \operatorname{sinc}^2\left(\frac{\delta\omega_k T}{2}\right) \times \\
\times \left[\rho_C^2\left(\delta\tau^C - \Delta^C\right)\cos^2\left(\omega_B\left(\delta\tau_k^B - \Delta^B\right)\right) - \rho_C^2\left(\delta\tau^C + \Delta^C\right)\cos^2\left(\omega_B\left(\delta\tau_k^B + \Delta^B\right)\right)\right].
\end{gathered} \tag{3.28}
$$

Определим крутизну дискриминационной характеристики при отсутствии рассогласования по задержке, и полагая рассогласование остальных параметров нулевым.

$$
S_\tau^{split} = \left.\frac{\partial U_\tau^s(\delta\tau)}{\partial\delta\tau}\right|_{\delta\tau=0} -? \tag{3.29}
$$

Представим выражение для дискриминационной характеристики в удобной для дифференцирования форме. Первый квадрат косинуса:

$$
\begin{gathered}
\cos^2\left(\omega_B\left(\delta\tau_k^B - \Delta^B\right)\right) = \frac{1 + \cos\left(2\omega_B\left(\delta\tau_k^B - \Delta^B\right)\right)}{2} = \\
= \frac{1 + \cos\left(2\omega_B\delta\tau_k^B\right)\cos\left(2\omega_B\Delta^B\right) + \sin\left(2\omega_B\delta\tau_k^B\right)\sin\left(2\omega_B\Delta^B\right)}{2}
\end{gathered} \tag{3.30}
$$

Квадрат второго косинуса:

$$
\begin{gathered}
\cos^2\left(\omega_B\left(\delta\tau_k^B + \Delta^B\right)\right) = \frac{1 + \cos\left(2\omega_B\left(\delta\tau_k^B + \Delta^B\right)\right)}{2} = \\
= \frac{1 + \cos\left(2\omega_B\delta\tau_k^B\right)\cos\left(2\omega_B\Delta^B\right) - \sin\left(2\omega_B\delta\tau_k^B\right)\sin\left(2\omega_B\Delta^B\right)}{2},
\end{gathered} \tag{3.31}
$$

тогда выражение для дискриминационной характеристики принимает вид

$$U_\tau^{split}(\delta\tau) = \left(\frac{2}{\pi}\right)^2 \left(\frac{AL}{2}\right)^2 \operatorname{sinc}^2\left(\frac{\delta\omega_k T}{2}\right) \times \frac{1}{2}\left[\left(\rho_C^2\left(\delta\tau^C - \Delta^C\right) - \rho_C^2\left(\delta\tau^C + \Delta^C\right)\right)\right.$$
$$+\left(\rho_C^2\left(\delta\tau^C - \Delta^C\right) - \rho_C^2\left(\delta\tau^C + \Delta^C\right)\right)\cos\left(2\omega_B\delta\tau_k^B\right)\cos\left(2\omega_B\Delta^B\right) +$$
$$\left. + \left(\rho_C^2\left(\delta\tau^C - \Delta^C\right) + \rho_C^2\left(\delta\tau^C + \Delta^C\right)\right)\sin\left(2\omega_B\delta\tau_k^B\right)\sin\left(2\omega_B\Delta^B\right)\right] \tag{3.32}$$

Найдем частные производные

$$\frac{\partial}{\partial\delta\tau}\left[\rho_C^2\left(\delta\tau^C - \Delta^C\right) - \rho_C^2\left(\delta\tau^C + \Delta^C\right)\right] = \tag{3.33}$$
$$= 2\rho_C\left(\delta\tau^C - \Delta^C\right)\rho'_C\left(\delta\tau^C - \Delta^C\right) - 2\rho_C\left(\delta\tau^C + \Delta^C\right)\rho'_C\left(\delta\tau^C + \Delta^C\right). \tag{3.34}$$

Если $\Delta^C < \tau_c$, где τ_c - длительность символа дальномерного кода, то

$$\left.\frac{\partial}{\partial\delta\tau}\left[\rho_C^2\left(\delta\tau^C - \Delta^C\right) - \rho_C^2\left(\delta\tau^C + \Delta^C\right)\right]\right|_{\delta\tau=0} =$$
$$= 2\left(1 - \frac{\Delta^C}{\tau_c}\right)\frac{1}{\tau_c} - 2\left(1 - \frac{\Delta^C}{\tau_c}\right)\left(-\frac{1}{\tau_c}\right) = 4\left(1 - \frac{\Delta^C}{\tau_c}\right)\left(\frac{1}{\tau_c}\right). \tag{3.35}$$

Второе слагаемое в аналогичных условиях

$$\cos\left(2\omega_B\Delta^B\right)\frac{\partial}{\partial\delta\tau}\left(\rho_C^2\left(\delta\tau^C - \Delta^C\right) - \rho_C^2\left(\delta\tau^C + \Delta^C\right)\right)\cos\left(2\omega_B\delta\tau_k^B\right)\Big|_{\delta\tau=0} =$$
$$= 4\left(1 - \frac{\Delta^C}{\tau_c}\right)\left(\frac{1}{\tau_c}\right)\cos\left(2\omega_B\Delta^B\right), \tag{3.36}$$

третье

$$\sin\left(2\omega_B\Delta^B\right)\frac{\partial}{\partial\delta\tau}\left(\rho_C^2\left(\delta\tau^C - \Delta^C\right) + \rho_C^2\left(\delta\tau^C + \Delta^C\right)\right)\sin\left(2\omega_B\delta\tau_k^B\right)\Big|_{\delta\tau=0} =$$
$$= 4\omega_B\sin\left(2\omega_B\Delta^B\right)\left(1 - \frac{\Delta^C}{\tau_c}\right)^2 \tag{3.37}$$

Тогда выражение для крутизны дискриминационной характеристики принимает вид:

$$S_\tau^{split} = \left.\frac{\partial U_\tau^s(\delta\tau)}{\partial\delta\tau}\right|_{\delta\tau=0} = \left(\frac{2}{\pi}\right)^2\left(\frac{AL}{2}\right)^2\operatorname{sinc}^2\left(\frac{\delta\omega_k T}{2}\right) \times 2\left(1 - \frac{\Delta^C}{\tau_c}\right)\left(\frac{1}{\tau_c}\right)$$
$$\times\left[1 + \cos\left(2\omega_B\Delta^B\right) + \omega_B\tau_c\sin\left(2\omega_B\Delta^B\right)\left(1 - \frac{\Delta^C}{\tau_c}\right)\right] =$$
$$= 4\left(\frac{2}{\pi}\right)^2 q_{c/n0}T\sigma_{IQ}^2\left(\frac{\tau_c - \Delta^C}{\tau_c^2}\right)\operatorname{sinc}^2\left(\frac{\delta\omega_k T}{2}\right)$$
$$\times\left[1 + \cos\left(2\omega_B\Delta^B\right) + \omega_B\left(\tau_c - \Delta^C\right)\sin\left(2\omega_B\Delta^B\right)\right]. \tag{3.38}$$

Для сравнения дискриминационная характеристика дискриминатора при использовании корреляционных сумм в прямой форме:

$$U_{\tau}^{direct}\left(\delta\tau\right) = \ 2\sigma_{IQ}^{2} q_{c/n0} T \operatorname{sinc}^{2}\left(\frac{\delta\omega_{k}T}{2}\right)\left[\rho_{BC}^{2}\left(\delta\tau-\Delta\right)-\rho_{BC}^{2}\left(\delta\tau+\Delta\right)\right]. \quad (3.39)$$

Крутизна дискриминационной характеристики дискриминатора при использовании корреляционных сумм в прямой форме имеет вид:

$$S_{\tau}^{direct} = \left.\frac{\partial U_{\tau}^{direct}\left(\delta\tau\right)}{\partial\delta\tau}\right|_{\delta\tau=0} = \ 8\sigma_{IQ}^{2} q_{c/n0} T \operatorname{sinc}^{2}\left(\frac{\delta\omega_{k}T}{2}\right)\left(1-\frac{\Delta}{\tau_{zero}}\right)\left(\frac{1}{\tau_{zero}}\right), \quad (3.40)$$

где τ_{zero} - половина ширины первого пика АКФ ВОС сигнала по первым нулям.

3.3 Флуктуационная характеристика

Найдем выходную флуктуационную характеристику дискриминатора при применении split-компонент – дисперсию его выходного процесса при нулевой ошибке по параметрам слежения (примечание “split” временно опустим, как и индекс времени “k”)

$$D_{u\tau} = M\left[\left(u_{\tau}^{s} - M\left[u_{\tau}^{s}\right]\right)^{2}\right] - ? \quad (3.41)$$

Выражение под квадратом содержит сумму слагаемых вида (чертой сверху для краткости выводов обозначим символ взятия математического ожидания):

$$I_{e}^{2} - \overline{I_{e}^{2}} = \bar{I}_{e}^{2} + 2\bar{I}_{e} n_{Ie} + n_{Ie}^{2} - \left(\bar{I}_{e}^{2} + \sigma_{IQ}^{s2}\right) = 2\bar{I}_{e} n_{Ie} + n_{Ie}^{2} - \sigma_{IQ}^{s2}, \quad (3.42)$$

тогда флуктуационная характеристика может быть представлена в виде

$$D_{u\tau} = \sum_{i}\sum_{j} M\left[\left(-1\right)^{i}\left(-1\right)^{j}\left(2X_{i}n_{i} + n_{i}^{2} - \sigma_{IQ}^{s2}\right)\left(2X_{j}n_{j} + n_{j}^{2} - \sigma_{IQ}^{s2}\right)\right], \quad (3.43)$$

где $X_1 = I_e,\ X_2 = I_l,\ X_3 = Q_e\,, X_4 = Q_l,$

$n_1 = n_{Ie},\ n_2 = n_{Il},\ n_3 = n_{Qe},\ n_4 = n_{Il}.$

Рассмотрим одно из i-e j-e слагаемое суммы (без учета знакового множителя)

$$\begin{aligned} &M\left[\left(2X_{i}n_{i} + n_{i}^{2} - \sigma_{IQ}^{s2}\right)\left(2X_{j}n_{j} + n_{j}^{2} - \sigma_{IQ}^{s2}\right)\right] = \\ &= M\left[4X_{i}n_{i}X_{j}n_{j} + n_{i}^{2}n_{j}^{2} - n_{i}^{2}\sigma_{IQ}^{s2} - n_{j}^{2}\sigma_{IQ}^{s2} + \sigma_{IQ}^{s4}\right] = \\ &= M\left[4X_{i}X_{j}n_{i}n_{j}\right] + M\left[n_{i}^{2}n_{j}^{2}\right] - \sigma_{IQ}^{s4} \end{aligned} \quad (3.44)$$

Всего в сумме 163 шестнадцать слагаемых: перебираются сочетания запаздывающих/опережающих компонент, а так же синфазных/квадратурных сумм. Рассмотрим соответствующие этим комбинациям слагаемые:

$$\begin{vmatrix} M[] & -\bar{I}_e n_{Ie} & -\bar{Q}_e n_{Qe} & \bar{I}_l n_{Il} & \bar{Q}_l n_{Ql} \\ -\bar{I}_e n_{Ie} & \bar{I}_e^2 \sigma_{IQ}^{s2} & 0 & -\bar{I}_e \bar{I}_l r_{2\Delta} \sigma_{IQ}^{s2} & 0 \\ -\bar{Q}_e n_{Qe} & 0 & \bar{Q}_e^2 \sigma_{IQ}^{s2} & 0 & -\bar{Q}_e \bar{Q}_l r_{2\Delta} \sigma_{IQ}^{s2} \\ \bar{I}_l n_{Il} & -\bar{I}_e \bar{I}_l r_{2\Delta} \sigma_{IQ}^{s2} & 0 & \bar{I}_l^2 \sigma_{IQ}^{s2} & 0 \\ \bar{Q}_l n_{Ql} & 0 & -\bar{Q}_e \bar{Q}_l r_{2\Delta} \sigma_{IQ}^{s2} & 0 & \bar{Q}_l^2 \sigma_{IQ}^{s2} \end{vmatrix},$$

откуда

$$\sum_i \sum_j (-1)^i (-1)^j M\left[4X_i X_j n_i n_j\right] = \\ = 4\sigma_{IQ}^{s2}\left(\bar{I}_e^2 + \bar{Q}_e^2 + \bar{I}_l^2 + \bar{Q}_l^2\right) - 8r_{2\Delta}\sigma_{IQ}^{s2}\left(\bar{I}_e\bar{I}_l + \bar{Q}_e\bar{Q}_l\right) \tag{3.45}$$

Аналогично

$$\begin{vmatrix} M[] & -n_{Ie}^2 & -n_{Qe}^2 & n_{Il}^2 & n_{Ql}^2 \\ -n_{Ie}^2 & 3\sigma_{IQ}^{s4} & \sigma_{IQ}^{s4} & -\left(1+2r_{2\Delta}^2\right)\sigma_{IQ}^{s4} & -\sigma_{IQ}^{s4} \\ -n_{Qe}^2 & \sigma_{IQ}^{s4} & 3\sigma_{IQ}^{s4} & -\sigma_{IQ}^{s4} & -\left(1+2r_{2\Delta}^2\right)\sigma_{IQ}^{s4} \\ n_{Il}^2 & -\left(1+2r_{2\Delta}^2\right)\sigma_{IQ}^{s4} & -\sigma_{IQ}^{s4} & 3\sigma_{IQ}^{s4} & \sigma_{IQ}^{s4} \\ n_{Ql}^2 & -\sigma_{IQ}^{s4} & -\left(1+2r_{2\Delta}^2\right)\sigma_{IQ}^{s4} & \sigma_{IQ}^{s4} & 3\sigma_{IQ}^{s4} \end{vmatrix}$$

что дает

$$\sum_i \sum_j (-1)^i (-1)^j M\left[n_i^2 n_j^2\right] = \\ = 4\left(3\sigma_{IQ}^{s4} - \left(1+2r_{2\Delta}^2\right)\sigma_{IQ}^{s4}\right) = 8\left(1-r_{2\Delta}^2\right)\sigma_{IQ}^{s4}. \tag{3.46}$$

Третья составляющая самоуничтожается:

$$\sum_i \sum_j (-1)^i (-1)^j \sigma_{IQ}^{s4} = 0.$$

Объединяя все слагаемые, получаем

$$D_{u\tau} = 4\sigma_{IQ}^{s2}\left(\bar{I}_e^2 + \bar{Q}_e^2 + \bar{I}_l^2 + \bar{Q}_l^2\right) \\ - 8r_{2\Delta}\sigma_{IQ}^{s2}\left(\bar{I}_e\bar{I}_l + \bar{Q}_e\bar{Q}_l\right) + 8\left(1-r_{2\Delta}^2\right)\sigma_{IQ}^{s4}. \tag{3.47}$$

Воспользуемся выражениями для статистических эквивалентов

$$\bar{I}_e^2+\bar{Q}_e^2=\left(\frac{2}{\pi}\frac{AL}{2}\right)^2\operatorname{sinc}^2\left(\frac{\delta\omega_k T}{2}\right)\rho_C^2\left(\delta\tau^C+\Delta^C\right)\cos^2\left(\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right), \tag{3.48}$$

$$\bar{I}_l^2+\bar{Q}_l^2=\left(\frac{2}{\pi}\frac{AL}{2}\right)^2\operatorname{sinc}^2\left(\frac{\delta\omega_k T}{2}\right)\rho_C^2\left(\delta\tau^C-\Delta^C\right)\cos^2\left(\omega_B\left(\delta\tau_k^B-\Delta^B\right)\right),$$

$$\bar{I}_e\bar{I}_l+\bar{Q}_e\bar{Q}_l=\left(\frac{2}{\pi}\frac{AL}{2}\right)^2\operatorname{sinc}^2\left(\frac{\delta\omega_k T}{2}\right)\rho_C\left(\delta\tau^C+\Delta^C\right)\rho_C\left(\delta\tau^C-\Delta^C\right)\times$$
$$\times\cos\left(\omega_B\left(\delta\tau_k^B-\Delta^B\right)\right)\cos\left(\omega_B\left(\delta\tau_k^B+\Delta^B\right)\right).$$

Для упрощения выражений рассмотрим флуктуационную характеристику при нулевых ошибках слежения, тогда:

$$\bar{I}_e^2+\bar{Q}_e^2=\bar{I}_e^2+\bar{Q}_e^2=\bar{I}_e\bar{I}_l+\bar{Q}_e\bar{Q}_l=\left(\frac{2}{\pi}\frac{AL}{2}\right)^2\times$$
$$\times\rho_C^2\left(\Delta^C\right)\cos^2\left(\omega_B\Delta^B\right)=\left(\frac{2}{\pi}\right)^2\left(\frac{AL}{2}\right)^2 r_\Delta^2, \tag{3.49}$$

где

$$r_\Delta=\rho_C\left(\Delta^C\right)\cos\left(\omega_B\Delta^B\right). \tag{3.50}$$

В этом случае выражение для дисперсии выходного процесса дискриминатора принимает вид

$$D_{u\tau}=8\sigma_{IQ}^{s2}\left(\frac{2}{\pi}\right)^2\left(\frac{AL}{2}\right)^2 r_\Delta^2-8r_{2\Delta}\sigma_{IQ}^{s2}\left(\frac{2}{\pi}\right)^2\left(\frac{AL}{2}\right)^2\times$$
$$\times r_\Delta^2+8\left(1-r_{2\Delta}^2\right)\sigma_{IQ}^{s4}=$$
$$=\left(1-r_{2\Delta}\right)8\sigma_{IQ}^{s2}\left(\frac{2}{\pi}\right)^2\left(\frac{AL}{2}\right)^2 r_\Delta^2+8\left(1-r_{2\Delta}^2\right)\sigma_{IQ}^{s4}=$$
$$=(1-r_{2\Delta})8\sigma_{IQ}^{s4}\left(4\left(\frac{2}{\pi}\right)^2\frac{\left(\frac{AL}{2}\right)^2}{4\sigma_{IQ}^{s2}}r_\Delta^2+1+r_{2\Delta}\right). \tag{3.51}$$

Вспомним связь отношения сигнал/шум с параметрами сигнала:

$$\frac{\left(\frac{AL}{2}\right)^2}{4\sigma_{IQ}^{s2}T}=q_{c/no}\to q_{c/no}T=\frac{\left(\frac{AL}{2}\right)^2}{4\sigma_{IQ}^{s2}}=\frac{\left(\frac{AL}{2}\right)^2}{2\sigma_{IQ}^{2}} \tag{3.52}$$

Тогда выражение для дисперсии выходного процесса дискриминатора принимает вид:

$$D_{u\tau}^{split} = (1 - r_{2\Delta})\, 8\sigma_{IQ}^{s4} \left(4\left(\frac{2}{\pi}\right)^2 q_{c/n0} T r_{\Delta}^2 + 1 + r_{2\Delta}\right) = \tag{3.53}$$

$$= (1 - r_{2\Delta})\, 8 q_{c/n0} T \sigma_{IQ}^{s4} \left(4\left(\frac{2}{\pi}\right)^2 r_{\Delta}^2 + \frac{1 + r_{2\Delta}}{q_{c/n0} T}\right).$$

Аналогичная формула при использовании корреляционных сумм в прямой форме в дискриминаторе:

$$D_{u\tau}^{direct} = (1 - \rho_{BC}(2\Delta))\, 8\sigma_{IQ}^4 \left(2 q_{c/n0} T \rho_{BC}^2(\Delta) + 1 + \rho_{BC}^2(2\Delta)\right) = \tag{3.54}$$

$$= (1 - \rho_{BC}(2\Delta))\, 16 q_{c/n0} T \sigma_{IQ}^4 \left(\rho_{BC}^2(\Delta) + \frac{1 + \rho_{BC}(2\Delta)}{2 q_{c/n0} T}\right).$$

Завершим преобразования для (3.53):

$$D_{u\tau}^{split} = \left(1 - \rho_C\left(2\Delta^C\right)\cos\left(2\omega_B \Delta^B\right)\right) 4 q_{c/n0} T \sigma_{IQ}^4 \times \tag{3.55}$$

$$\left(2\left(\frac{2}{\pi}\right)^2 \rho_C^2\left(\Delta^C\right)\cos^2\left(\omega_B \Delta^B\right) + \frac{1 + \rho_C\left(2\Delta^C\right)\cos\left(2\omega_B \Delta^B\right)}{2 q_{c/n0} T}\right).$$

Найдем выражение для дисперсии эквивалентных наблюдений:

$$D_{\tilde{u}\tau}^{split} = \frac{D_{u\tau}^{split}}{{S_\tau}^2} = \frac{\left(1 - \rho_C\left(2\Delta^C\right)\cos\left(2\omega_B \Delta^B\right)\right)}{\left(\frac{2}{\pi}\right)^4 4 q_{c/n0} T \left(\frac{\tau_c - \Delta^C}{\tau_c^2}\right)^2} \times \tag{3.56}$$

$$\times \frac{2\left(\frac{2}{\pi}\right)^2 \rho_C^2\left(\Delta^C\right)\cos^2\left(\omega_B \Delta^B\right) + \frac{1 + \rho_C\left(2\Delta^C\right)\cos\left(2\omega_B \Delta^B\right)}{2 q_{c/n0} T}}{\left[1 + \cos\left(2\omega_B \Delta^B\right) + \omega_B\left(\tau_c - \Delta^C\right)\sin\left(2\omega_B \Delta^B\right)\right]^2}$$

Тогда выражение дисперсии эквивалентных наблюдений при использовании корреляционных сумм в прямой форме выглядит следующим образом:

$$D_{\tilde{u}\tau}^{direct} = \frac{D_{u\tau}^{direct}}{{S_\tau}^2} = \frac{(1 - \rho_{BC}(2\Delta))\left(\rho_{BC}^2(\Delta) + \frac{1 + \rho_{BC}(2\Delta)}{2 q_{c/n0} T}\right)}{4 q_{c/n0} T \left(\frac{1}{\tau_{zero}} - \frac{\Delta}{\tau_{zero}^2}\right)^2} \tag{3.57}$$

4 Особенности обработки сигналов с модуляцией BOC и AltBOC

4.1 Прием AltBOC сигналов

В главе 2 был рассмотрен способ приема BOC сигнала при помощи традиционных BPSK корреляторов, при этом для обработки каждого из главных лепестков спектра выделяется один канал коррелятора (2.2). Обобщим полученный алгоритм для случая обработки сигнала с AltBOC модуляцией.

В современной спутниковой радионавигационной системе Galileo используется сигнал E5 с модуляцией AltBOC(15,10). В таком сигнале уплотнено 4 радиосигнала: E5aI, E5aQ, E5bI, E5bQ. В результате такого уплотнения формируется радиосигнал, спектральная плотность мощности (СПМ) которого имеет два главных лепестка, аналогично спектру радиосигнала с BOC модуляцией [4]. Причем левый лепесток спектра содержит только радиосигналы E5a-I и E5a-Q, а правый только сигналы E5b-I и E5b-Q, которые можно принимать как квадратурные радиосигналы с модуляцией BPSK(10). Благодаря этому предложенный в работе метод приема BOC сигнала на два независимых BPSK коррелятора можно адаптировать и для обработки AltBOC сигнала. Обобщим выражения корреляционных сумм (2.2) для случая обработки сигнала с AltBOC модуляцией:

$$
\begin{aligned}
I^{reg}_{1^l/_e,k} &= \sum_{l=1}^{L} y_{1,k,l} C_1\left(t_{k,l} - \left(\breve{\tau}^{C}_{k} {}^{+}/_{-}\Delta^C\right)\right)\cos\left(\left(\omega_{if} - \omega_B\right)t_{k,l} + \breve{\omega}_k\left(l-1\right)T_d + \breve{\varphi}_k\right),\\
Q^{reg}_{1^l/_e,k} &= \sum_{l=1}^{L} y_{1,k,l} C_1\left(t_{k,l} - \left(\breve{\tau}^{C}_{k} {}^{+}/_{-}\Delta^C\right)\right)\sin\left(\left(\omega_{if} - \omega_B\right)t_{k,l} + \breve{\omega}_k\left(l-1\right)T_d + \breve{\varphi}_k\right),\\
I^{reg}_{2^l/_e,k} &= \sum_{l=1}^{L} y_{2,k,l} C_2\left(t_{k,l} - \left(\breve{\tau}^{C}_{k} {}^{+}/_{-}\Delta^C\right)\right)\cos\left(\left(\omega_{if} + \omega_B\right)t_{k,l} + \breve{\omega}_k\left(l-1\right)T_d + \breve{\varphi}_k\right),\\
Q^{reg}_{2^l/_e,k} &= \sum_{l=1}^{L} y_{2,k,l} C_2\left(t_{k,l} - \left(\breve{\tau}^{C}_{k} {}^{+}/_{-}\Delta^C\right)\right)\sin\left(\left(\omega_{if} + \omega_B\right)t_{k,l} + \breve{\omega}_k\left(l-1\right)T_d + \breve{\varphi}_k\right),
\end{aligned}
\tag{4.1}
$$

где y_1, y_2 – входные сигналы нижнего и верхнего лепестков спектра, $C_1\left(\right)$, $C_2\left(\right)$ – дальномерные коды левого и правого лепестков спектра соответственно. Такая возможность позволяет с помощью описанного сплит-метода принимать сигнал E5 Galileo, имеющий AltBOC модуляцию.

4.2 Возможность обработки BOC и AltBOC сигналов с помощью двух радиотрактов

Сигналы с BOC и AltBOC модуляцией имеют относительно широкую занимаемую полосу, например сигнал Galileo E5 имеет полосу $\Delta f \approx 51$ МГц. Однако радиочастотные блоки, которые позволяли бы обрабатывать такие сигналы целиком имеют высокую стоимость. Поэтому часто при обработке сигналов с BOC и AltBOC модуляцией используют BPSK-like метод. При этом обрабатывается только один из главных лепестков спектра, что влечет за собой потери в точности, и нивелирует приемущества BOC сигналов.

При использовании сплит алгоритма, рассмотренного в главе 2 разные лепестки спектра могут пропускаться через разные радиотракты, как это возникает, например, при использовании популярной микросхемы NT1065 “Nomada”. В случае обработки сигнала Galileo E5I микросхемой NT1065 могут быть использованы два канала, с одинаковой частотой гетеродина $f = 5 \cdot 238 = 1190$ МГц: RF1_IN и RF2_IN, с настройками LSB и USB соответственно. Тогда компонента сигнала E5a будет пропускаться через канал RF1_IN, а компонента E5b через RF2_IN (см. рис. 4.1), формируя два разных цифровых сигнала. Эти сигналы могут быть распределены между двумя BPSK корреляторами и обработаны, что недоступно при использовании специализированного BOC коррелятора.

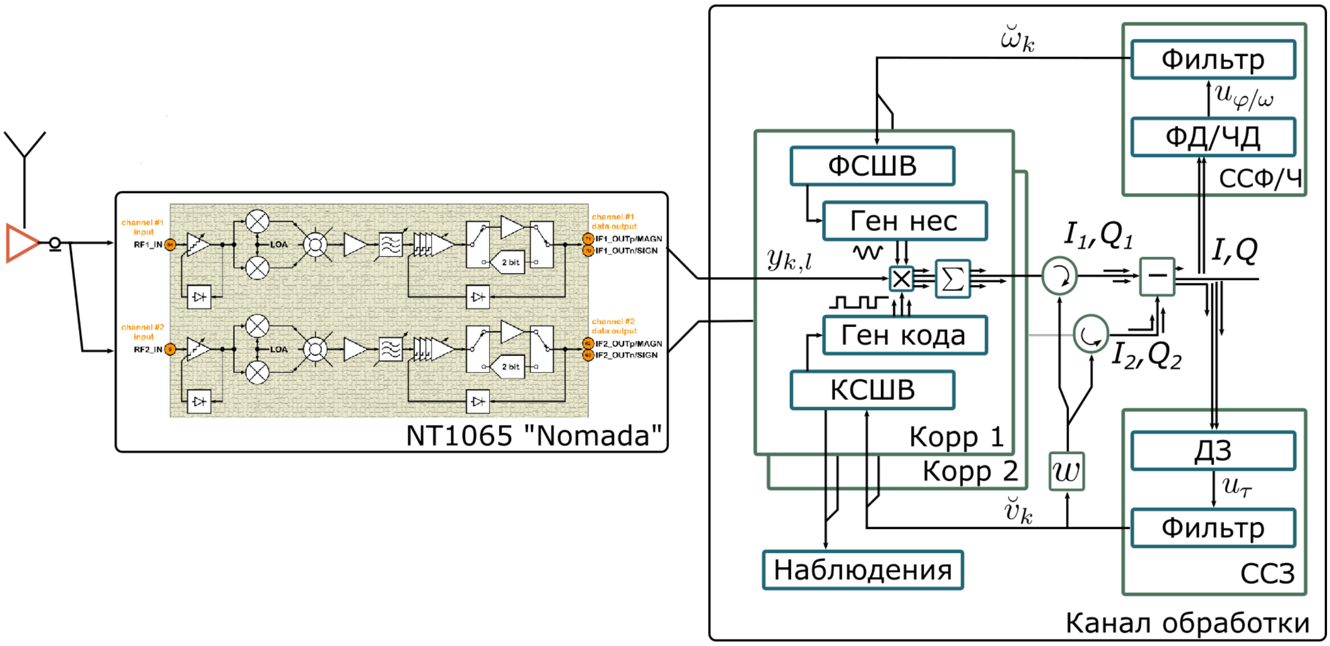

Рисунок 4.1 — Обработка сигнала Galileo E5I с использованием микросхемы NT1065 "Nomada"

В УИЦ "ЛНС"(Учебно-исследовательский центр "Лаборатория Навигационных Систем") МЭИ разработан навигационный приемник "Clonicus". Радиочастотные блоки данного приемника реализованы с помощью двух микросхем NT1065 "Nomada что позволяет принимать сигналы всех частотных диапазонов существующих ГНСС. Ядром цифровой обработки сигналов в НАП "Clonicus"является СнК ZYNQ XC7Z030. Данная СнК содержит как двухядерный процессор ARM Cortex-A9, так и ПЛИС. Это позволяет реализовывать в НАП необходимые алгоритмы обработки сигналов, а также различные вспомогательные функции.

В частности в ПЛИС НАП "Clonicus"реализован модуль Datacollector. Данный модуль представляет собой набор ячеек памяти и позволяет запоминать отсчеты процессов (сигналов) в ПЛИС для дальнейшего анализа и отладки. Для отображения и анализа данных блока Datacollector в УИЦ "ЛНС"была разработана одноименная программа для ПК на языке Python. Программа позволяет строить осциллограммы, спектры, гистограммы сигналов, выборки которых записаны в модуль Datacollector ПЛИС.

Проведен эксперимент по приему сигнала Galileo E5. Схема экспериментальной установки представлена на рисунке 4.2.

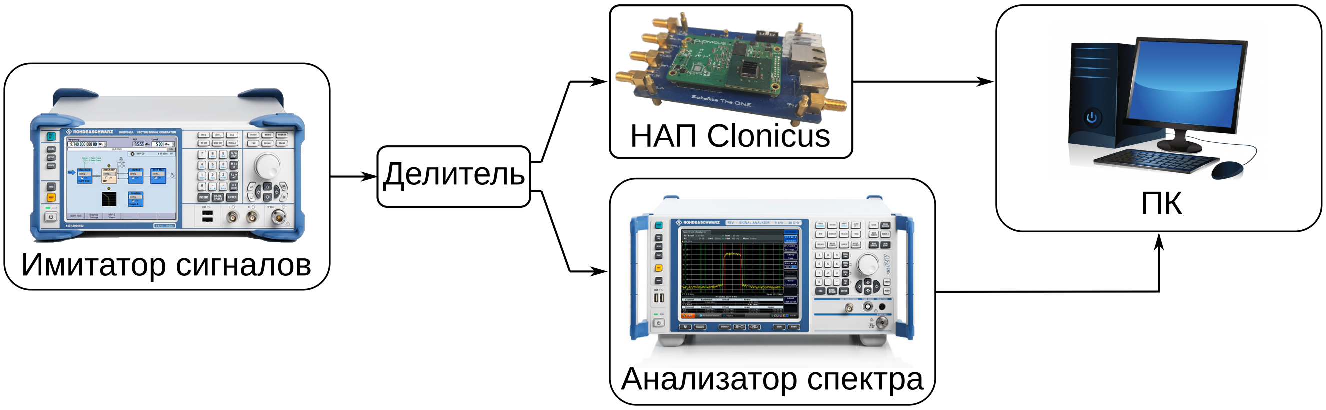

Рисунок 4.2 — Схема установки

Сигнал Galileo E5 формируется с помощью имитатора сигналов ГНСС Rohde&Schwarz SMBV 100B. Через делитель мощности навигационный сигнал поступает на НАП "Clonicus"и на анализатор спектра Rohde&Schwarz FSV. ПО НАП настраивает микросхему РЧБ "Nomada"NT1065 на прием компонент E5a и E5b сигнала Galileo E5 как описано выше в начале раздела.

В блок Datacollector ПЛИС записываются отсчеты двухразрядных АЦП (sig/mag) с каналов USB и LSB РЧБ Nomada.

Спектр сигнала Galileo E5, формируемого имитатором сигналов ГНСС на несущей частоте $f = 1191.795$ МГц, полученный с помощью анализатора спектра, представлен на рисунке 4.3.

Спектр сигнала Galileo E5a (левый лепесток СПМ сигнала E5) полученный с помощью модуля Datacollector на выходе канала 1 (LSB) РЧБ "Nomada"показан синим цветом на рисунке 4.4. Аналогичный спектр сигнала Galileo E5b (правый лепесток СПМ сигнала E5) на выходе канала 2 (USB) РЧБ "Nomada"показан красным цветом.

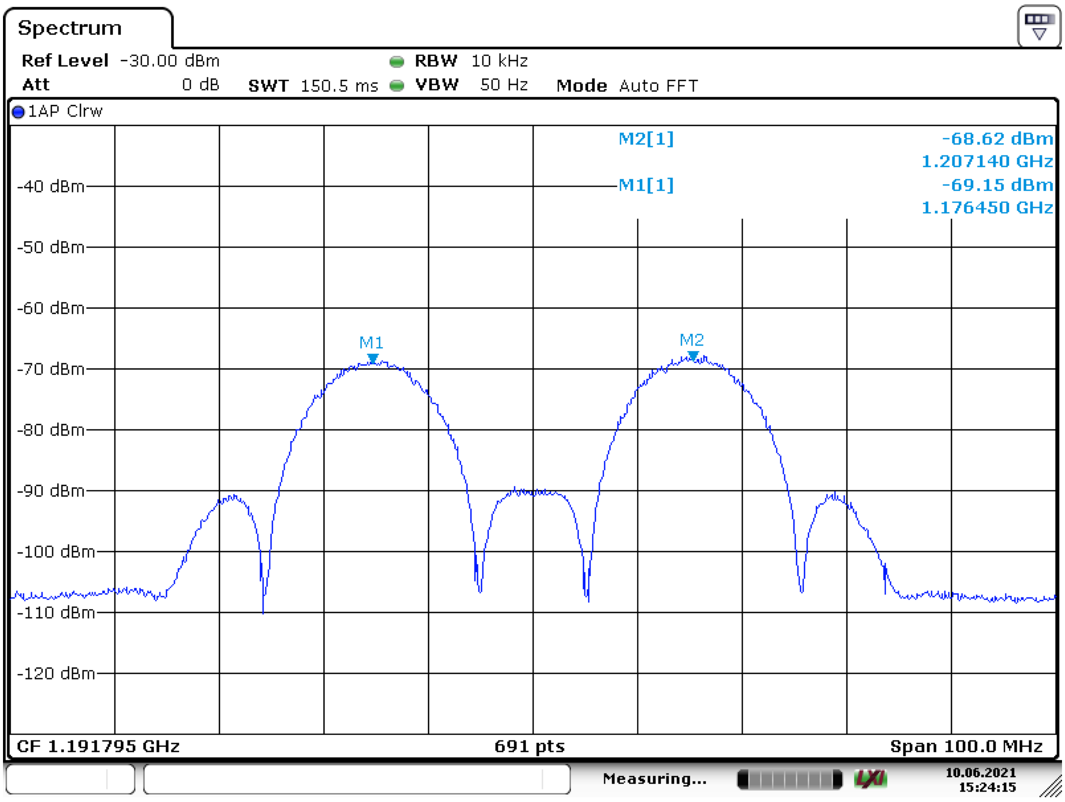

Рисунок 4.3 — Спектр сигнала Galileo E5

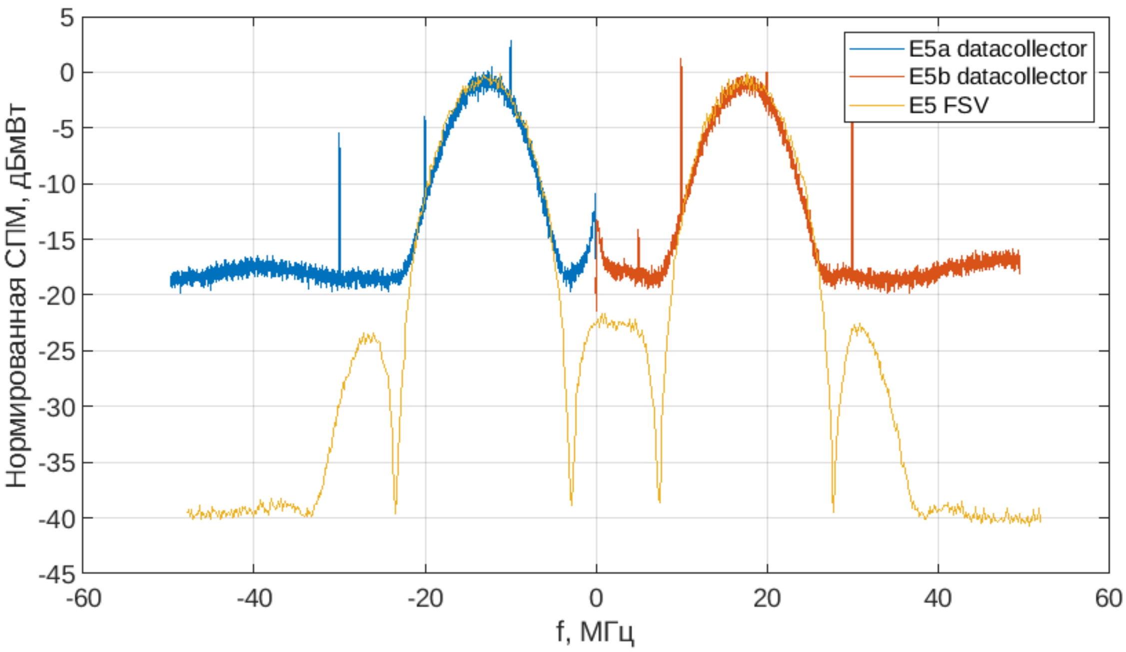

Рисунок 4.4 — Спектр сигнала Galileo E5 после РЧБ

Гармоники на спектрах обусловлены паразитными составляющими опорного сигнала при формировании колебаний гетеродина в РЧБ. Отсутсвие симметрии спектра компонент E5a и E5b относительно нуля обусловлено применением в РЧБ "Nomada"гетеродина с частотой $f_г = 1190$ МГц, не равной центральной частоте сигнала Galileo E5 $f = 1191.795$ МГц.

Для наглядности на рисунке 4.4 также приведен спектр сигнала с анализатора спектра FSV, перенесенный на промежуточную частоту с помощью гетеродина той же частоты, что и в РЧБ $f_г = 1190$ МГц. Все спектры на рисунке 4.4 нормированы на свой максимум.

4.3 Использование алгоритма при втягивании систем слежения

Использование модуляции поднесущей в большинстве случаев повышает точность оценки задержки, однако она имеет некоторые недостатки, которые связаны с видом автокорреляционной функции. Автокорреляционная функция BOC сигнала имеет многопиковый характер, что делает приемник, предназначенный для обработки таких сигналов, более чувствительным к динамическим ошибкам, повышая вероятность ложного захвата. Однако, если система слежения настроена на главный пик АКФ, то точность оценки задержки лучше, чем для соответствующего сигнала BPSK, за счет более узкого главного пика АКФ.

В литературе рассматривается проблема ложного захвата при обработке BOC сигналов и описываются некоторые методики для решения этой проблемы [5, 6, 7, 8, 9, 10, 11].

Такие методики как Bump-jump или, например, [7, 11], основанные на сравнении соседних пиков АКФ сигнала, предполагают использование дополнительных каналов коррелятора. Помимо запаздывающих (late) и опережающих (early) компонент корреляционных сумм необходимы также Very Late и Very Early компоненты корреляционных сумм. Такие методики требуют модификации аппаратной части навигационного приемника.

Метод устранения неоднозначности SCPC предполагает осуществлять умножение входного сигнала на дальномерный код и на две копии меандра сдвинутые относительно друг друга на 90 градусов. Соответственно данный метод также требует модификации аппаратной части приемника, так как требует реализовывать генератор поднесущей для формирования опорного сигнала.

Рассмотрим далее способ уменьшения вероятности ложного захвата слежения за задержкой BOC сигнала с использованием полученных в Главе 2 сплит корреляционных сумм (2.13). Данный способ по сути определяет алгоритм втягивания следящей системы за задержкой сигнала и не требует изменения аппаратной части навигационного приемника.

При использовании такого подхода разделение параметров задержки огибающей для дальномерного кода $\tau_k \to \tau_k^C$, $\Delta \to \Delta^C$ и поднесущей $\tau_k \to \tau_k^B$, $\Delta \to \Delta^B$ позволяет по-отдельности управлять этими параметрами в опорном сигнале. Для наглядности представим корреляционную сумму сигнала с модуляцией поднесущей в виде поверхности (см. рис. 4.5).

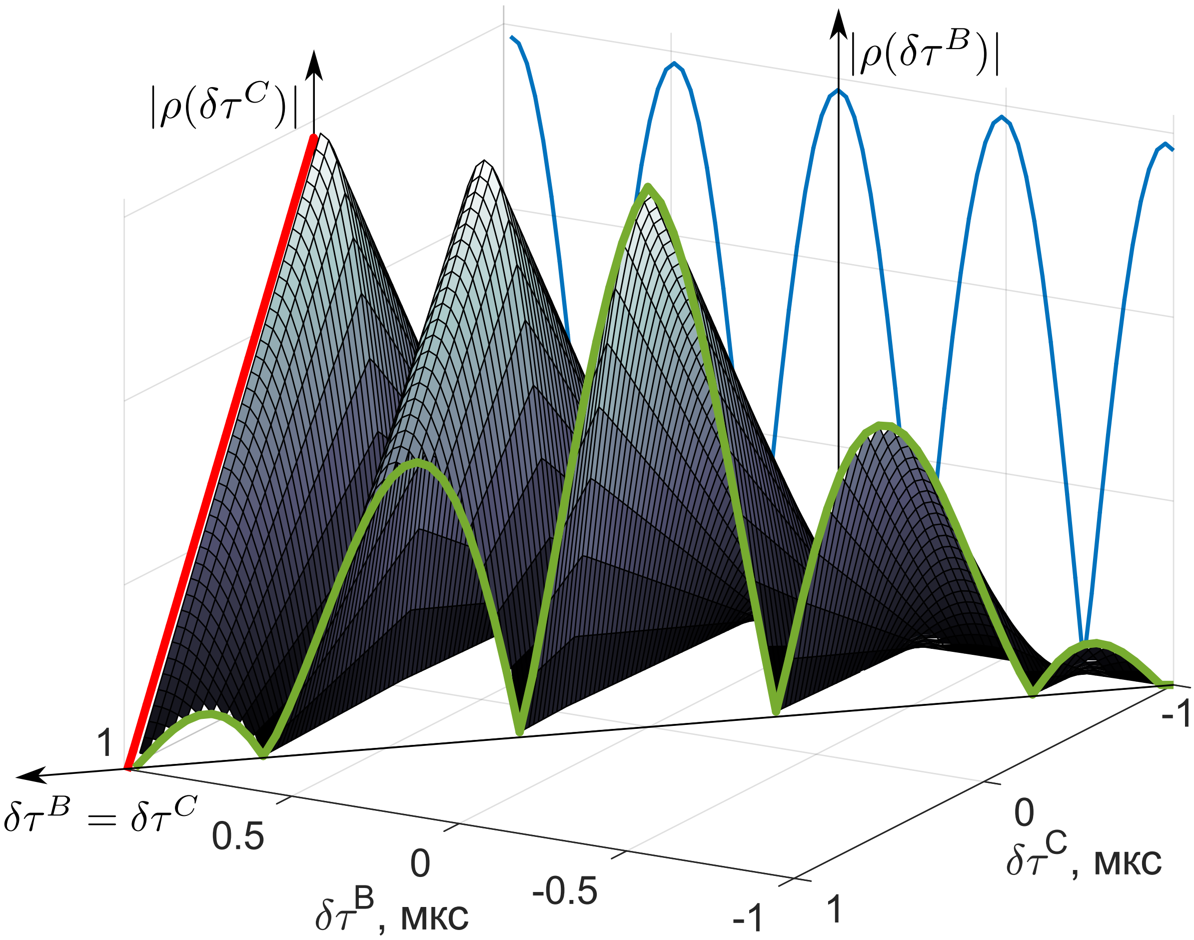

Рисунок 4.5 — Двумерная АКФ сигнала BOC(1,1)

Как было сказано ранее, автокорреляционная функция BOC сигналов многопиковая, и при слежении за параметрами сигнала важно отличить главный корреляционный пик от ложных/побочных. Для этого при запуске систем слежения можно перестать управлять параметром τ^B из ССЗ, при этом дискриминационная характеристика будет иметь более широкую апертуру и не будет иметь ложных нулей. На рис. 4.6 показаны: синей линией – дискриминационная характеристика сплит-дискриминатора без управления τ^B, красной линией нормированная на крутизну дискриминационная характеристика с управлением τ^B.

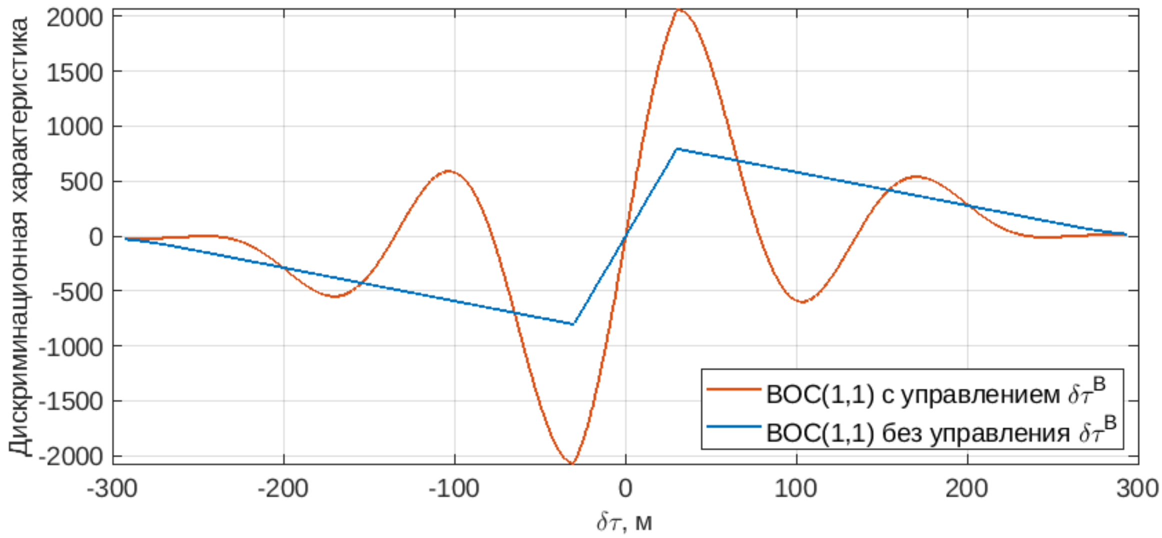

Рисунок 4.6 — Дискриминационная характеристика с управлением τ^B и без управления τ^B для сплит-алгоритма

Аналогично корреляционная функция будет более широкая и не будет иметь ложных максимумов (красный график на рисунке 4.7). В таком режиме по сигналу рассогласования на выходе дискриминатора, подстраивая τ^C, можно свести ошибку почти к нулю.

После того, как закончился переходной процесс, режим раздельного втягивания выключается. Включается управление задержкой поднесущей ($\delta\tau^B = \delta\tau^C$). При этом мы получаем многопиковую корреляционную функцию с узким главным пиком (зеленый график на рисунке 4.7). Полученная корреляционная функция соответствует срезу двумерной корреляционной функции $\delta\tau^B = \delta\tau^C$ (см. рис. 4.5).

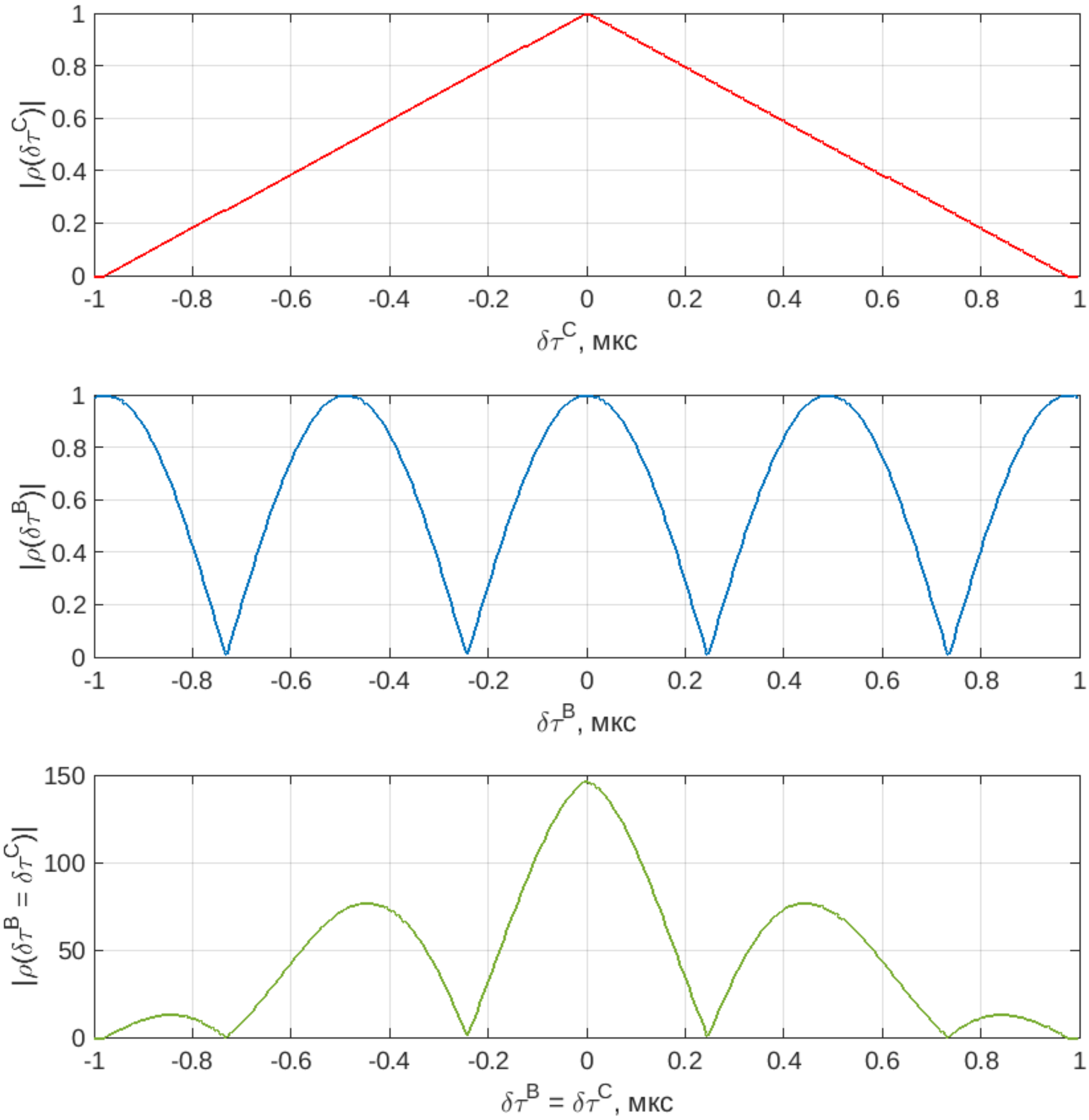

Рисунок 4.7 — Проекции корреляционных функций

Таким образом алгоритм позволяет раздельно подстраивать задержку дальномерного кода и задержку поднесущей, что может быть использовано для разрешения неоднозначности задержки огибающей BOC сигнала, вызванной многопиковым характером его корреляционной функции.

Особенностью данного алгоритма является то, что начальное значение задержки поднесущей является случайной величиной и может попадать в минимумы корреляционной функции поднесущей (синий график на рисунке 4.7). В таком случае реальная крутизна ДХ в режиме раздельного втягивания будет меньше, чем рассчитанная. Под термином рассчитанная крутизна, в данном случае, понимается крутизна, рассчитанная при условии, что задержка поднесущей попала в максимум своей корреляционной функции. В связи с этом реальная полоса системы уменьшается, переходной процесс занимает больше времени. Для уменьшения этого влияния предлагается увеличивать полосу примерно до 2-3 Гц.

5 Имитационное моделирование

5.1 Проверка характеристик дискриминатора задержки

В результате синтеза алгоритма дискриминатора задержки (2.1) с применением сплит-корелляционных сумм (2.13) были получены аналитические выражения, определяющие его дискриминационную (3.27) и флуктуационную (3.55) характеристики. Кроме того в работе получены аналогичные аналитические выражения, определяющие ДХ (3.39) и ФХ (3.54) для дискриминатора задержки с применением корреляционных сумм в прямой форме (2.2). Для проверки аналитических выражений дискриминационных характеристик проведено имитационное моделирование методом статистических эквивалентов двух алгоритмов дискриминатора задержки. Составлена модель, где корреляторы моделировались своими статистическими эквивалентами. Листинг программы на языке matlab приведен в Приложении А.

Моделирование проводилось при следующих параметрах: отношение сигнал шум $q_{c/n0} = 35$ дБГц, время накопления в корреляторах $Tc = 5$ мс, усреднение проводилось по 1000 реализациям. При построении характеристик ошибки слежения по фазе $\delta\varphi_k$ и частоте $\delta\omega_k$ полагались равными нулю.

В дальнейшем все результаты моделирования будут представлены для двух алгоритмов дискриминатора задержки: с использованием сплит-корреляционных сумм (2.13) (обозначение split) и с использованием корреляционных сумм в прямой форме (2.2) (обозначение direct).

На рисунках 5.1 – 5.3 представлены полученные в результате моделирования (назовем их экспериментальными), а также рассчитанные по аналитическим выражениям ДХ для сигналов с модуляцией BOC(1,1), BOC(5,2.5), BOC(15,10).

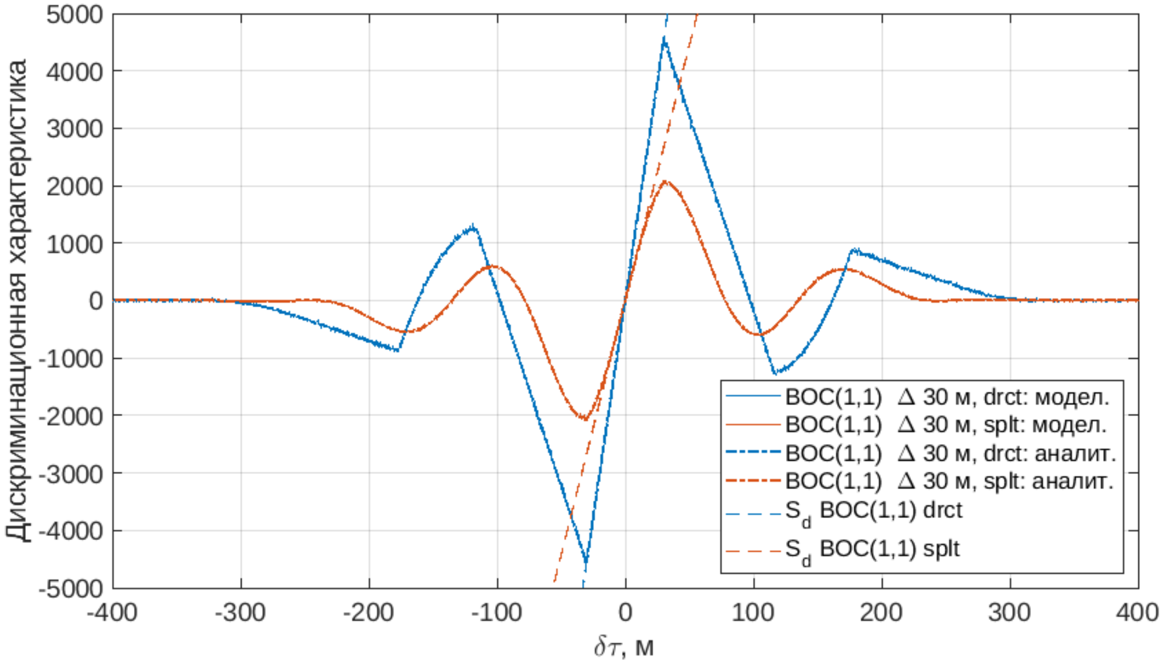

Рисунок 5.1 — Аналитические и экспериментальные ДХ дискриминатора задержки для двух случаев: split и direct. Сигнал BOC(1,1)

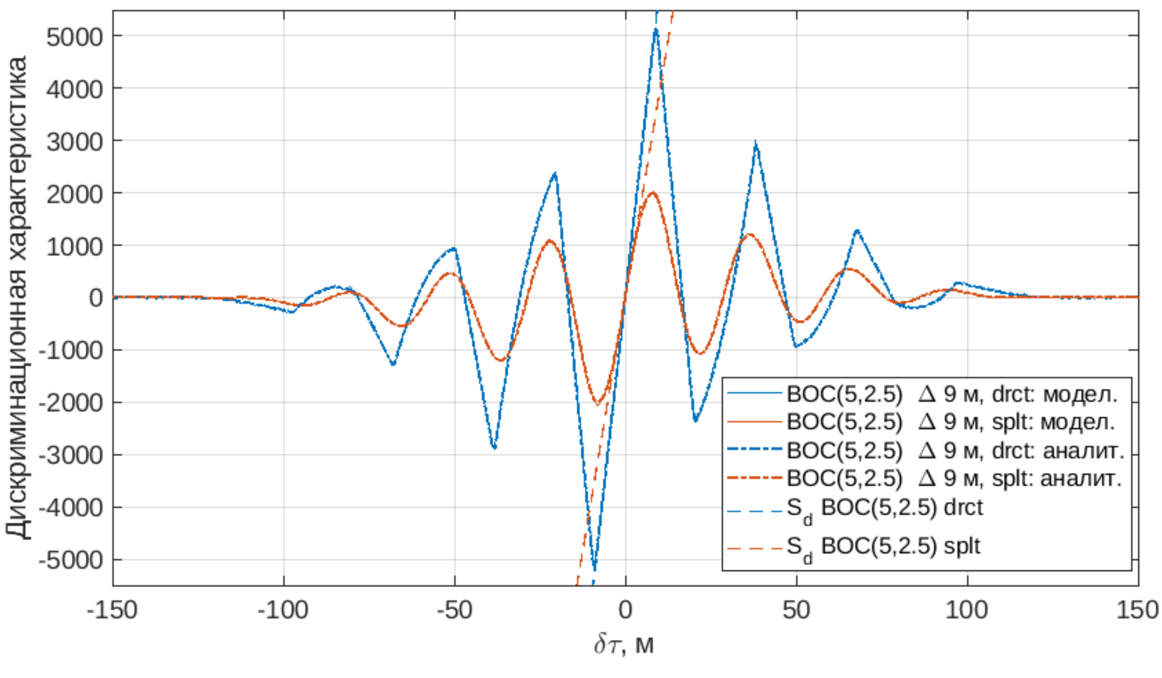

Рисунок 5.2 — Аналитические и экспериментальные ДХ дискриминатора задержки для двух случаев: split и direct. Сигнал BOC(5,2.5)

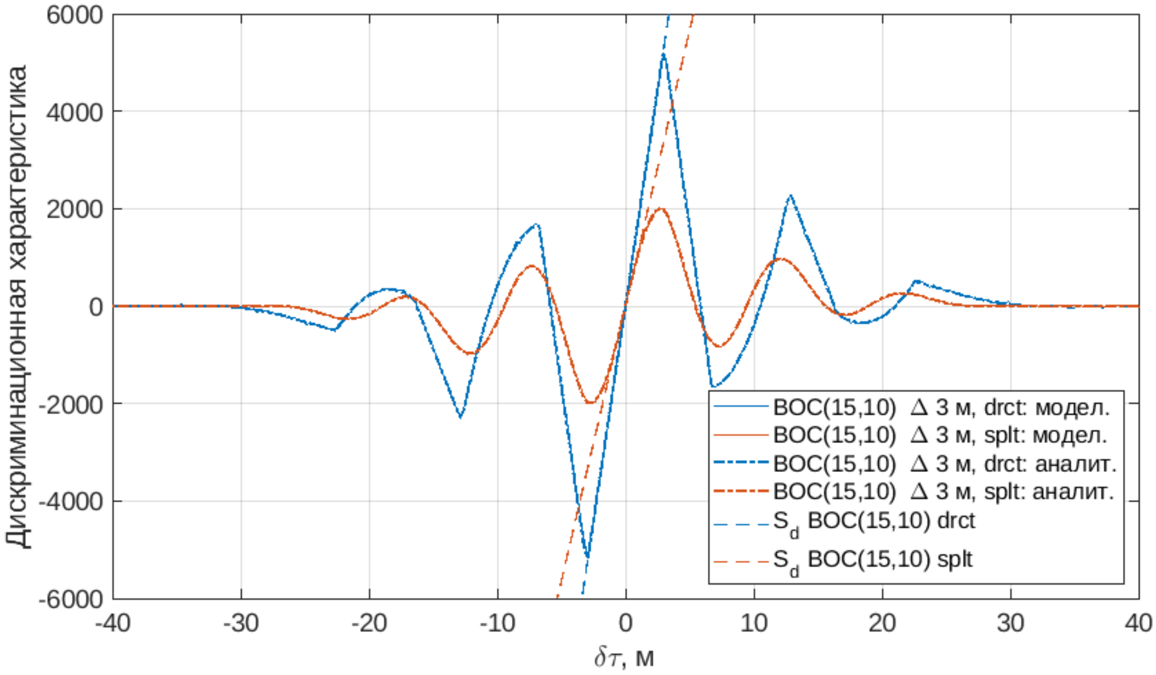

Рисунок 5.3 — Аналитические и экспериментальные ДХ дискриминатора задержки для двух случаев: split и direct. Сигнал BOC(15,10) aka AltBOC

Расстройка между компонентами early и promt Δ^C отличается для каждого типа BOC. Для модуляции BOC(1,1) $\Delta^C = 30$ м, для BOC(5,2.5) $\Delta^C = 9$ м, для BOC(15,10) $\Delta^C = 3$ м.

Также на графиках построены зависимости вида $y(\delta\tau) = S_d \delta\tau$, где S_d – рассчитывается по аналитической формуле: при использовании сплит компонент (3.38), при использовании корреляционных сумм в прямой форме (3.40).

Как видно из рисунков 5.1 – 5.3 аналитические и экспериментальные ДХ совпадают как при использовании сплит компонент коррелятора, так и при использовании корреляционных сумм в прямой форме.

Отметим, что как и ожидалось, дискриминационные характеристики имеют многопиковый характер ввиду использования модуляции цифровой поднесущей. ДХ дискриминатора задержки при применении сплит компонент оказывается сглаженной. Это обусловлено аппроксимацией цифровой поднесущей ее первой гармоникой (2.3).

На рисунках 5.4 – 5.6 представлены корреляционные функции (promt) для двух типов корреляционных сумм: сплит-компонент и корреляционных сумм в прямой форме. Моделирование производилось в отсутствии шума. В данном контексте термин "корреляционная функция" использован для обозначения зависимости выходного сигнала от ошибки по задержке или, говоря строже, как $\rho(\delta\tau) \sim \sqrt{I_p^2 + Qp^2}$. Также на рисунка отображена расстройка, выбранная для построения 5.1-5.3.

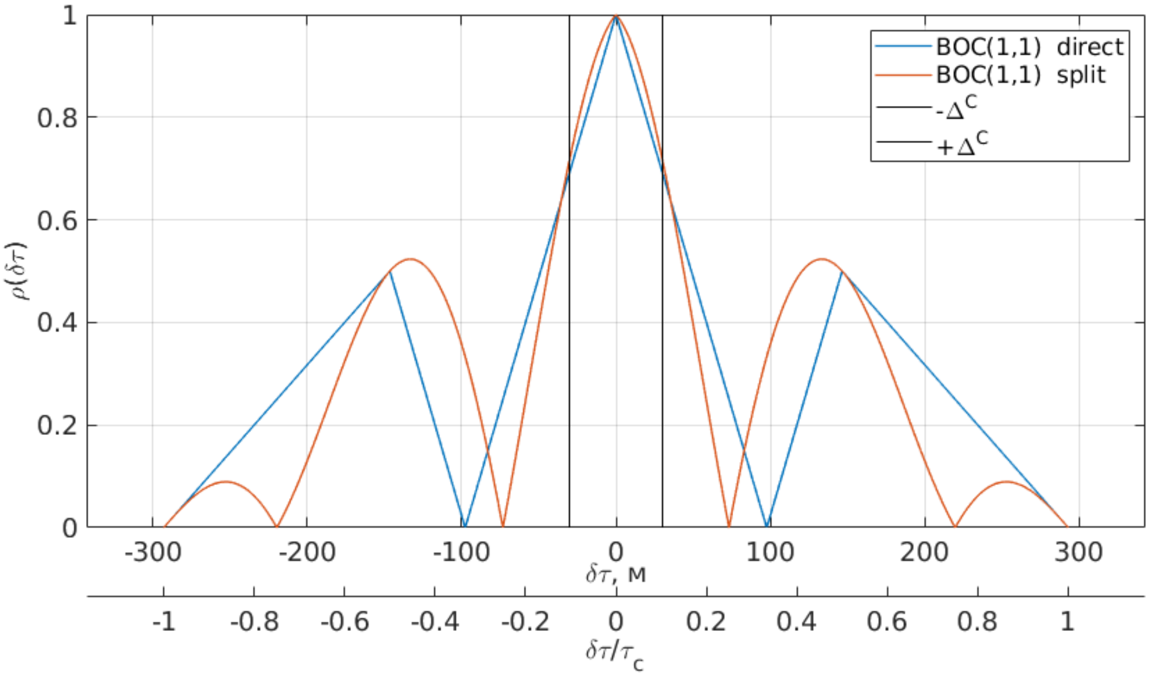

Рисунок 5.4 — Корреляционные функции для двух случаев: split и direct. Сигнал BOC(1,1)

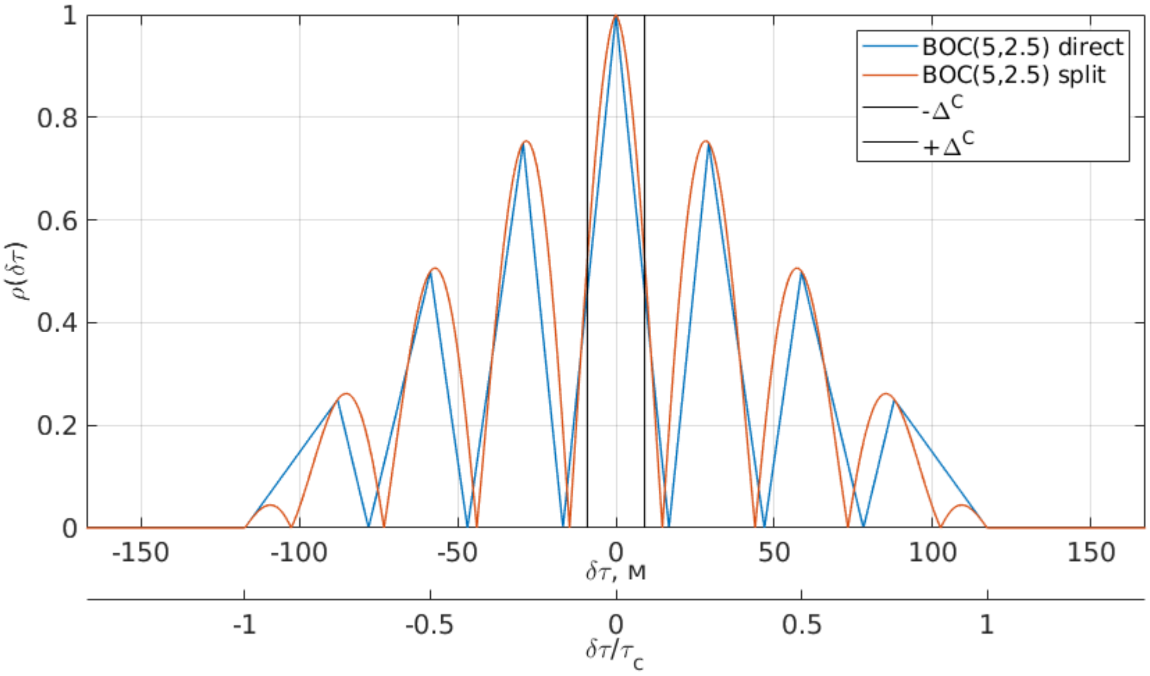

Рисунок 5.5 — Корреляционные функции для двух случаев: split и direct. Сигнал BOC(5,2.5)

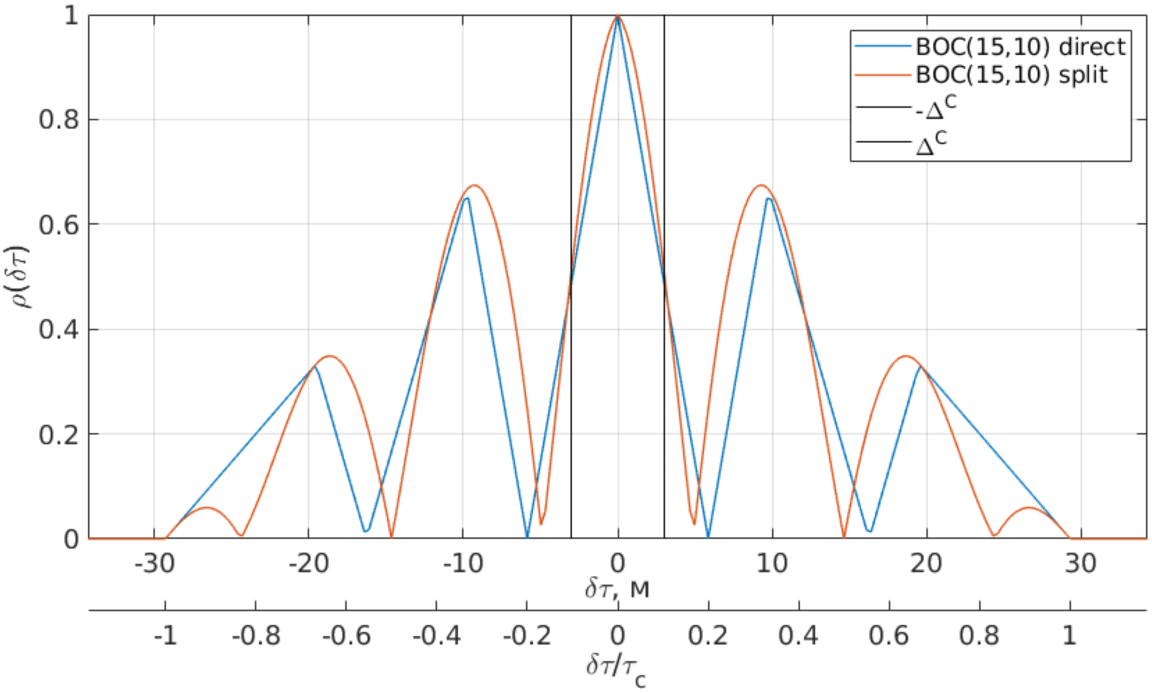

Рисунок 5.6 — Корреляционные функции для двух случаев: split и direct. Сигнал BOC(15,10)

Для проверки аналитических выражений флуктуационных характеристик составлена модель, где корреляторы моделировались своими статистическими эквивалентами. Листинг программы на языке matlab приведен в Приложении Б.

На рисунках 5.7 – 5.9 представлены полученные в результате моделирования, а также рассчитанные по аналитическим выражениям зависимости приведенной ко входу дискриминатора флуктуационной характеристики от отношения сигнал/шум для трех типов модуляции BOC.

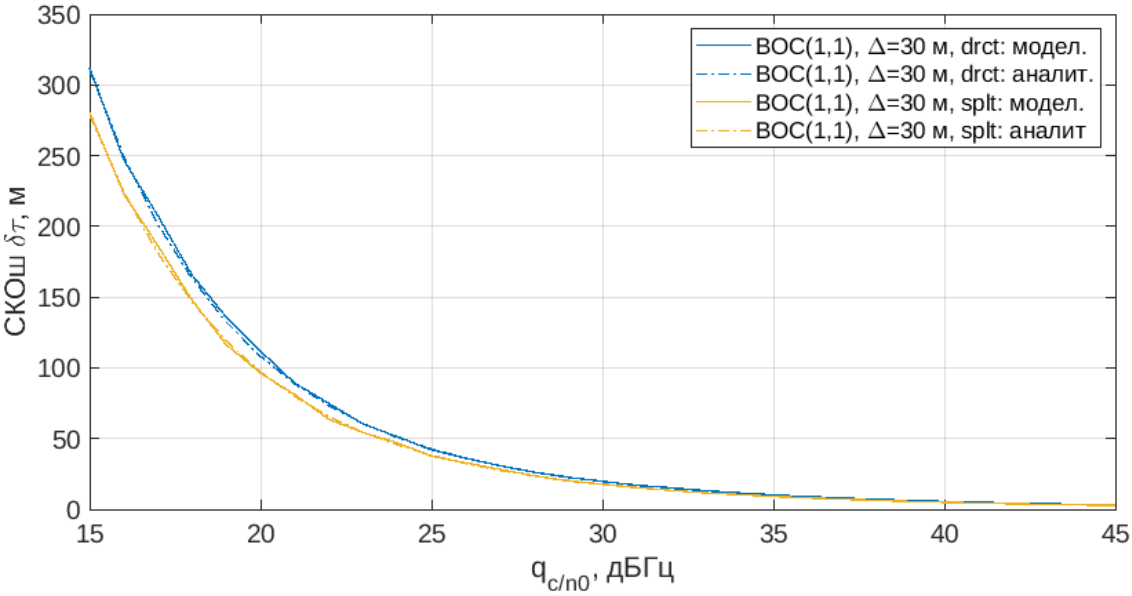

Рисунок 5.7 — Зависимость среднеквадратической ошибки эквивалентных наблюдений от отношения сигнал/шум для двух случаев: split и direct. Сигнал BOC(1,1)

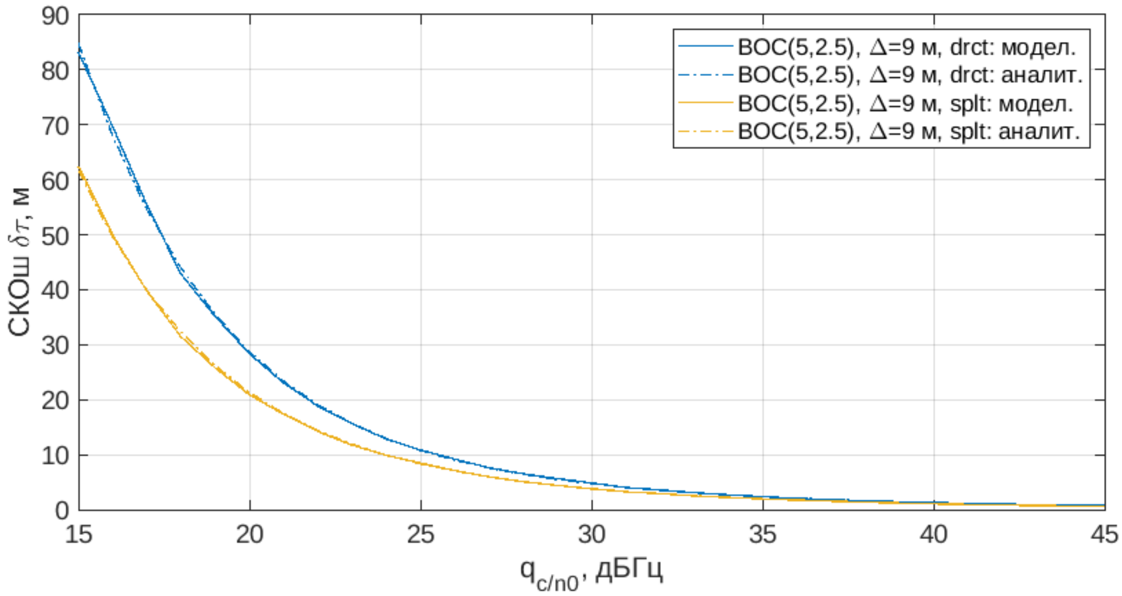

Рисунок 5.8 — Зависимость среднеквадратической ошибки эквивалентных наблюдений от отношения сигнал/шум для двух случаев: split и direct. Сигнал BOC(5,2.5)

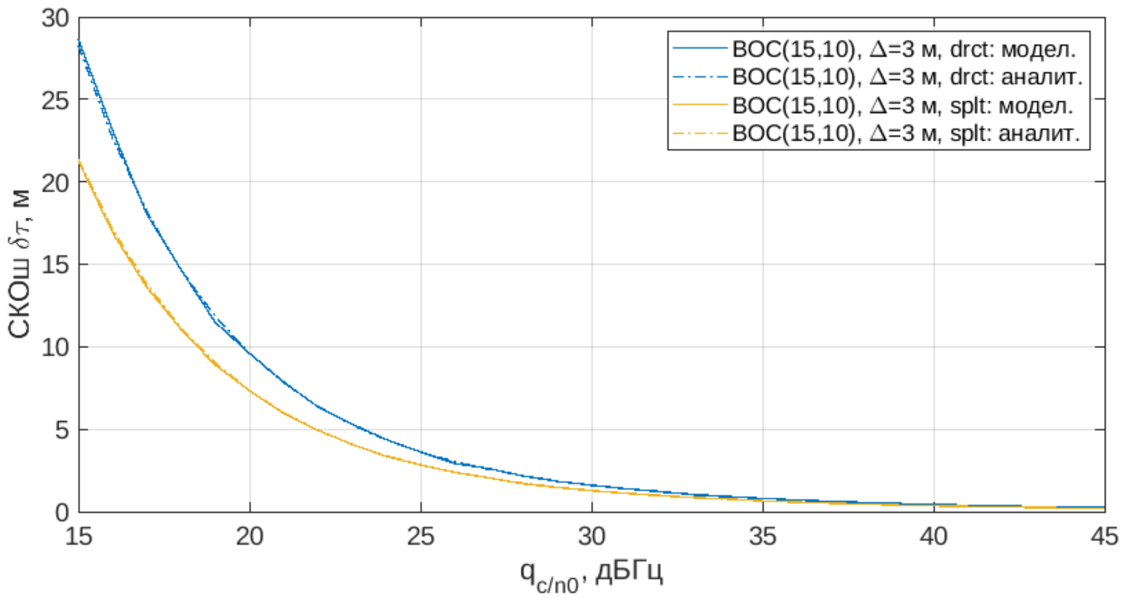

Рисунок 5.9 — Зависимость среднеквадратической ошибки эквивалентных наблюдений от отношения сигнал/шум для двух случаев: split и direct. Сигнал BOC(15,10)

Зависимости получены при времени накопления в корреляторах $T = 5$ мс, усреднение проводилось по 5000 реализациям. Выбор расстройки Δ между компонентами early и promt зависит от типа модуляции сигнала. Величина расстройки указана в легенде для каждого графика.

Как видно из рисунков 5.7 – 5.9 аналитические и экспериментальные зависимости совпадают как при использовании сплит компонент коррелятора, так и при использовании корреляционных сумм в прямой форме.

5.2 Характеристики слежения за задержкой сигнала

С целью сравнения точности и чувствительности (помехоустойчивости) алгоритмов дискриминатора задержки при использовании сплит корреляционных сумм и корреляционных сумм в прямой форме проведено моделирование системы слежения за задержкой сигнала с модуляцией цифровой поднесущей. Листинг программы на языке Matlab приведен в Приложении В.

Модель включает в себя только систему слежения за задержкой, ошибки от других систем не учитываются. Модель выполнена на статистических эквивалентах дискриминаторов. Динамика задержки сигнала в модели обусловлена только нестабильностью опорного генератора. В фильтрационном алгоритме используются коэффициенты фильтра в установившемся режиме.

5.2.1 Математическая модель

Динамика задержки сигнала в модели обусловлена только дрейфом частоты опорного генератора [12]. В следствии этого вектор состояния введен как:

$$\mathbf{x}_k = \begin{vmatrix} \varphi_k \\ \omega_k \\ \nu_k \end{vmatrix} \tag{5.1}$$

Динамика вектора состояния определяется выражением:

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{G}_{\text{ог}} n_{\text{ог}_{k-1}}, \tag{5.2}$$

где $\mathbf{F} = \begin{vmatrix} 1 & T & 0 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{vmatrix}$, $\mathbf{G} = \begin{vmatrix} 0 \\ T \\ 0 \end{vmatrix}$, $n_{\text{ог}}$ – ДБГШ с дисперсией $\sigma^2_{n_{\text{ог}}}$.

Дисперсия определяется как: $\sigma^2_{n_{\text{ог}}} = \frac{N_{\text{ог}}}{2T}$, где $N_{\text{ог}} = N_{\xi_\nu}\omega_0^2$.

Параметр N_{ξ_ν} был взят из статьи [12] для опорного генератора среднего качества аналогичный GTXO-83 $N_{\xi_\nu} = 1.1e^{-9}\ c^{-1}$

Процесс изменения задержки пересчитывается из вектора состояния:

$$\tau_k = -\frac{\varphi_k}{\omega_0} \tag{5.3}$$

Моделируется система слежения за задержкой 2-го порядка с вектором состояния:

$$\mathbf{x}_{dll_k} = \begin{vmatrix} \tau_k \\ \upsilon_k \end{vmatrix} \tag{5.4}$$

Алгоритм фильтрации:

$$\tilde{\mathbf{x}}_{dll_k} = \mathbf{F_{dll}}\hat{\mathbf{x}}_{dll_{k-1}}, \tag{5.5}$$

где вектор $\tilde{\mathbf{x}}_{dll}$ – экстраполяция вектора состояния, $\mathbf{F_{dll}} = \begin{vmatrix} 1 & T \\ 0 & 1 \end{vmatrix}$, вектор $\hat{\mathbf{x}}_{dll}$ – оценка вектора состояния.

$$\hat{\mathbf{x}}_{dll_k} = \tilde{\mathbf{x}}_{dll_k} + \mathbf{K}\frac{u_{d_k}}{S_d}, \tag{5.6}$$

В модели используется статистический эквивалент дискриминатора. Выходной сигнал дискриминатора представляется как сумма:

$$u_d = U_d + n_d. \tag{5.7}$$

Здесь $U_d\left(\delta\tau\right) = M\left[u_d\left(\delta\tau\right)\right]$ – дискриминационная характеристика, при нуле ошибок по фазе $\delta\varphi_k$ и частоте $\delta\omega_k$. Для расчета дискриминационной характеристики используются аналитические выражения полученные в 2 главе. Для дискриминатора при использовании корреляционных сумм в прямой форме – выражение (3.39), при использовании сплит компонент (3.27). Крутизна дискриминационной характеристики рассчитывается по формуле (3.40) при использовании корреляционных сумм в прямой форме и (3.38) при использовании сплит компонент; n_d – шум с дисперсией D_{u_d}. D_{u_d} – флуктуационная характеристика рассчитывается по формулам (3.54) и (3.55), для дискриминатора при использовании корреляционных сумм в прямой форме и при использовании сплит компонент соответственно.

При моделировании системы слежения за задержкой используются коэффициенты фильтра в установившемся режиме. Значения коэффициентов фильтра могут быть рассчитаны через эквивалентную шумовую полосу Δf следящей системы [3]:

$$\mathbf{K} = \begin{vmatrix} K_1 \cdot T \\ K_2 \cdot T \end{vmatrix}, \tag{5.8}$$

где

$$K_1 = 2 \cdot \frac{16}{9} \cdot \Delta f^2,$$
$$K_2 = \sqrt{2 \cdot K_1}.$$

5.2.2 Верификация имитационной модели

Имитационная модель следящей системы за задержкой сигнала была верифицирована с помощью тестового воздействия. Отклик следящей системы на тестовое воздействие должен быть известен до начала моделирования. В качестве тестового воздействия был выбран скачок истинного процесса изменения задержки на 10 нс. Скачок формируется в модели с 20 секунды моделирования, иная динамика истинного процесса отсутствует.

Ожидаемый результат:

Моделируемая следящая система имеет второй порядок астатизма, а тестовое воздействие постоянно. Следовательно, ошибка оценивания задержки в следящей системе будет равняться нулю (в отсутствии шумов наблюдения) в установившемся режиме [13]. Характерное время переходного процесса – несколько обратных значений полосы следящей системы.

Результат работы модели на тестовом воздействии приведен на рисунках 5.10 и 5.11. Моделирование проведено для сигнала с модуляцией BOC(1,1) при отношении сигнал/шум $q_{c/n0} = 45$ дБГц, полоса следящей системы $\Delta f = 0.5$ Гц.

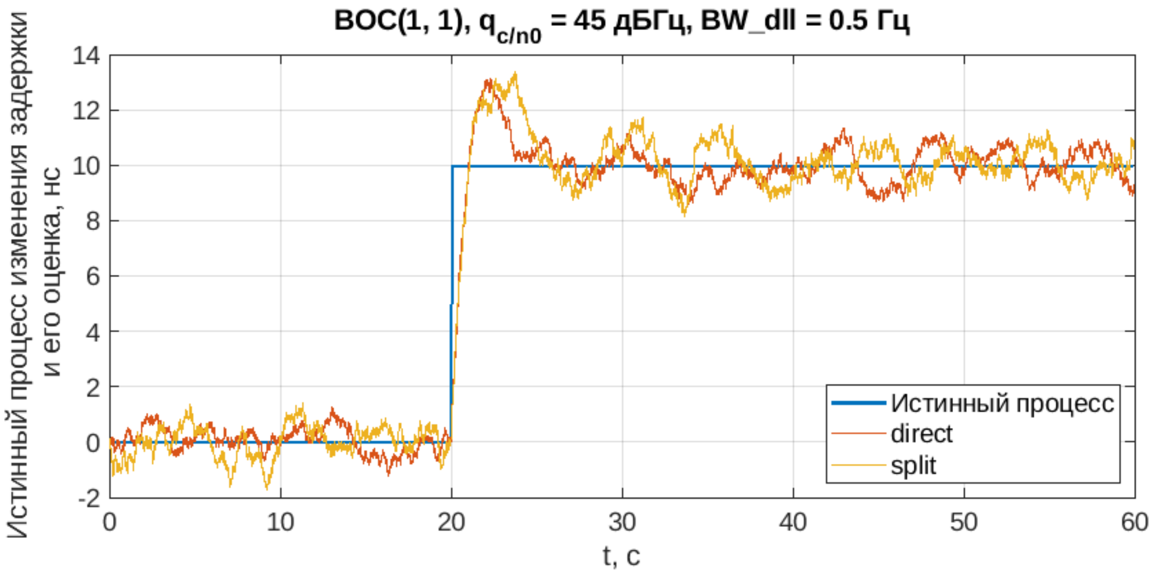

Рисунок 5.10 — Истинный процесс изменения задержки и его оценка для двух случаев: split и direct. Сигнал BOC(1,1)

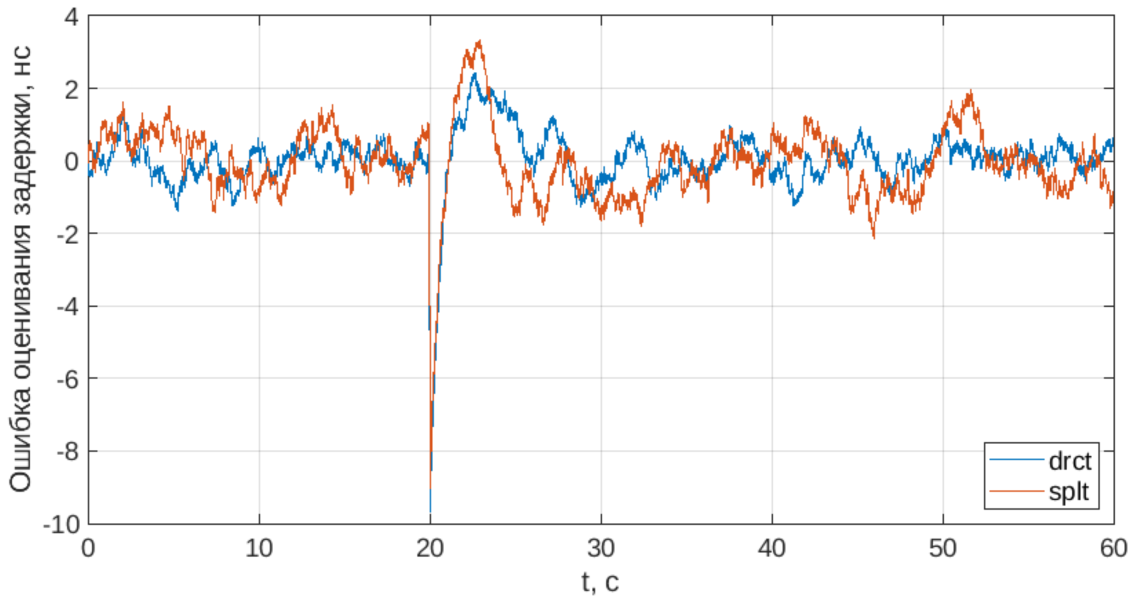

Рисунок 5.11 — Ошибка оценивания задержки для двух случаев: split и direct. Сигнал BOC(1,1)

Как и ожидалось, следящая система отрабатывает скачкообразное изменение задержки как для случая direct, так и для split. Ошибка слежения в установившемся режиме имеет околонулевое значение. Длительность переходного процесса около 5 секунд, что составляет 2.5 обратных значений полосы следящей системы.

5.2.3 Результаты моделирования

С помощью модели получены зависимости СКО ошибок слежения за задержкой сигнала от отношения сигнал/шум (см. рис. 5.12 - 5.14). Моделирование проводилось при фиксированной полосе ССЗ $\Delta f_{\text{ссз}} = 0.5$ Гц, усреднение проводилось по 500 реализациям на каждое значение отношения сигнал/шум. Зависимости получены для трех типов модуляции сигнала: BOC(1,1), BOC(5,2.5), BOC(15,10). Величина расстройки между компонентами early и promt отличается для каждого типа модуляции и равна для модуляции BOC(1,1) $\Delta^C = 30$ м, для BOC(5,2.5) $\Delta^C = 9$ м, для BOC(15,10) $\Delta^C = 3$ м.

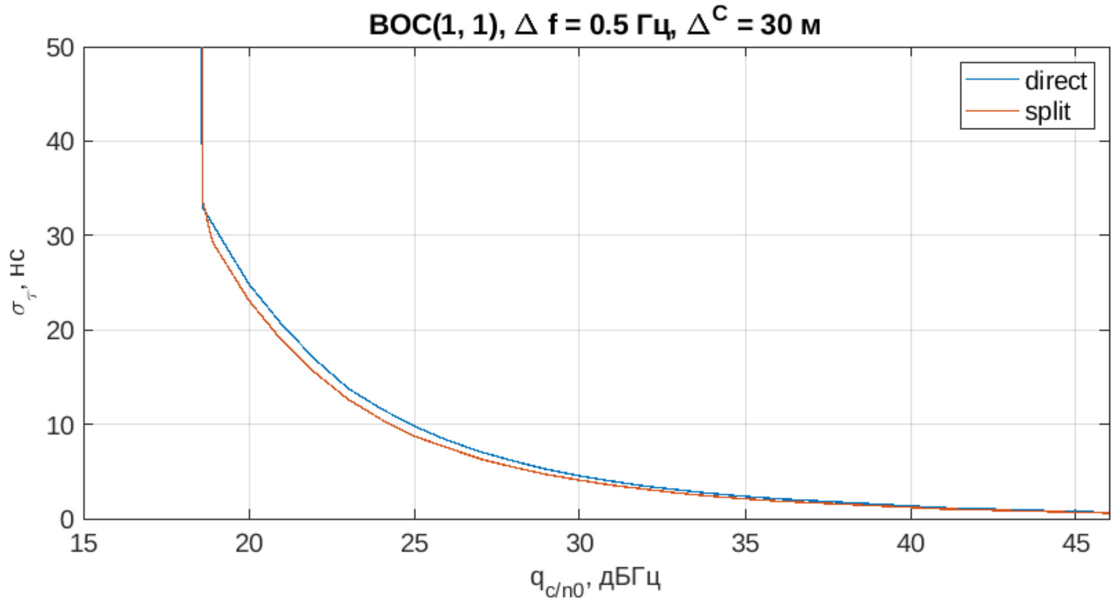

Рисунок 5.12 — Зависимость точности слежения за задержкой сигнала от отношения сигнал/шум для двух случаев: split и direct. Сигнал BOC(1,1)

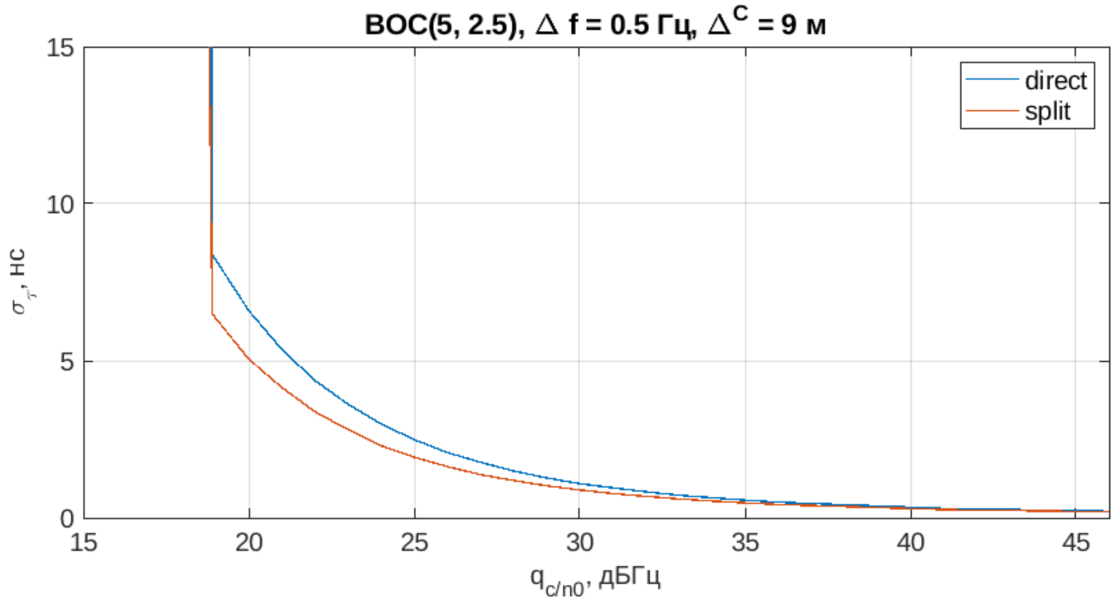

Рисунок 5.13 — Зависимость точности слежения за задержкой сигнала от отношения сигнал/шум для двух случаев: split и direct. Сигнал BOC(5,2.5)

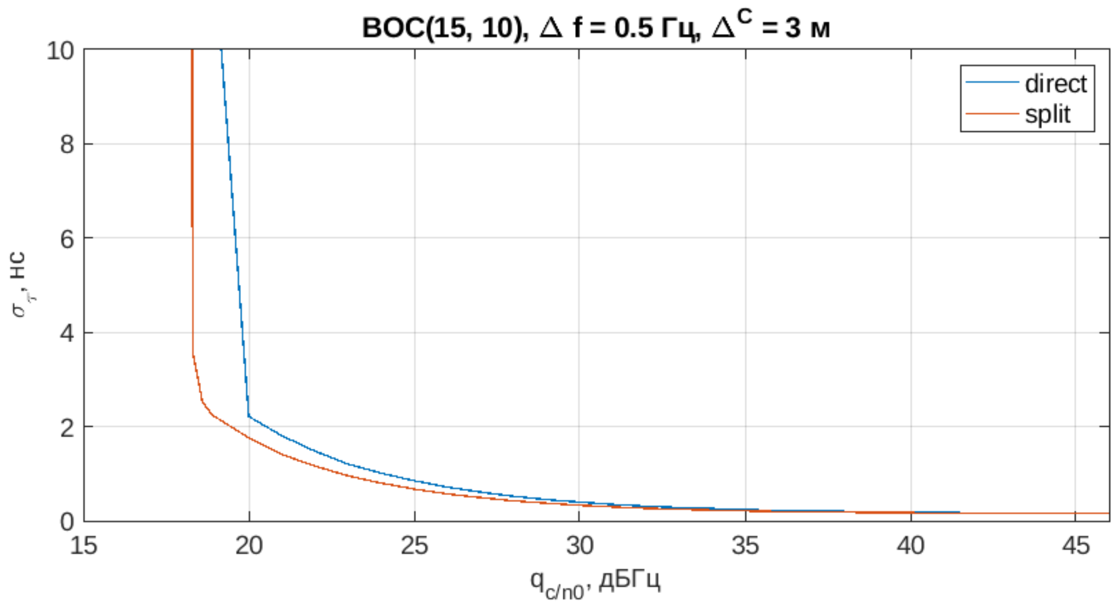

Рисунок 5.14 — Зависимость точности слежения за задержкой сигнала от отношения сигнал/шум для двух случаев: split и direct. Сигнал BOC(15,10)

По графикам рис. 5.12 - 5.14 видно, что точность слежения за задержкой сопоставима для двух алгоритмов дискриминатора: при использовании сплит корреляционных сумм и при использовании корреляционных сумм в прямой форме.

По графикам можно определить пороговое значение отношения сигнал/шум при котором происходит срыв слежения, что характеризуется резким увеличением СКО ошибки слежения. В качестве критерия срыва слежения использовалось следующее практическое правило [14]:

$$3\sigma_\tau > \frac{D}{2}, \tag{5.9}$$

где D – расстройка между компонентами early и late.

Учитывая используемые в данной работе обозначения формула (5.9) для критерия срыва слежения принимает вид:

$$\sigma_\tau > \frac{\Delta^C}{3} \tag{5.10}$$

где Δ^C – расстройка между компонентами early и promt.

Для сравнения моделируемых алгоритмов получим две важных характеристики слежения: точность и чувствительность. Под чувствительностью в данном случае будем понимать пороговое значение отношения сигнал/шум, при котором СКО ошибки слежения за задержкой еще не превышает порог, заданный (5.10). Под точностью будем понимать СКЗ ошибки слежения за задержкой сигнала.

Для удобства сравнения построим зависимости точности и чувствительности от Δ^C – расстройки между компонентами early и promt для алгоритмов при использовании сплит-компонент (split) и при использовании корреляционных сумм в прямой форме (direct). Полученные зависимости приведены на рисунках 5.15 - 5.17) для трех типов модуляции BOC.

Моделирование проводилось при фиксированной полосе ССЗ $\Delta f_{\text{ссз}} = 0.5$ Гц, при отношении сигнал/шум $q_{c/n0} = 45$ дБГц, усреднение проводилось по разным методикам для зависимости чувствительности и точности от расстройки.

Для построения усредненного графика чувствительности от расстройки на первом этапе выполнялось построение зависимости СКО ошибки слежения от отношения сигнал/шум, усредненной по 200 реализациям. На втором этапе по полученной усредненной зависимости с помощью формулы (5.10) определялось пороговое значение отношения сигнал/шум при котором происходит срыв слежения. Определение порогового значения таким образом проводилось 30 раз. Итоговая зависимость порогового значения сигнал/шум (чувствительность) от расстройки между компонентами early и promt откладывались на график.

Усреднение зависимости точности слежения от расстройки проводилось по 10000 реализациям.

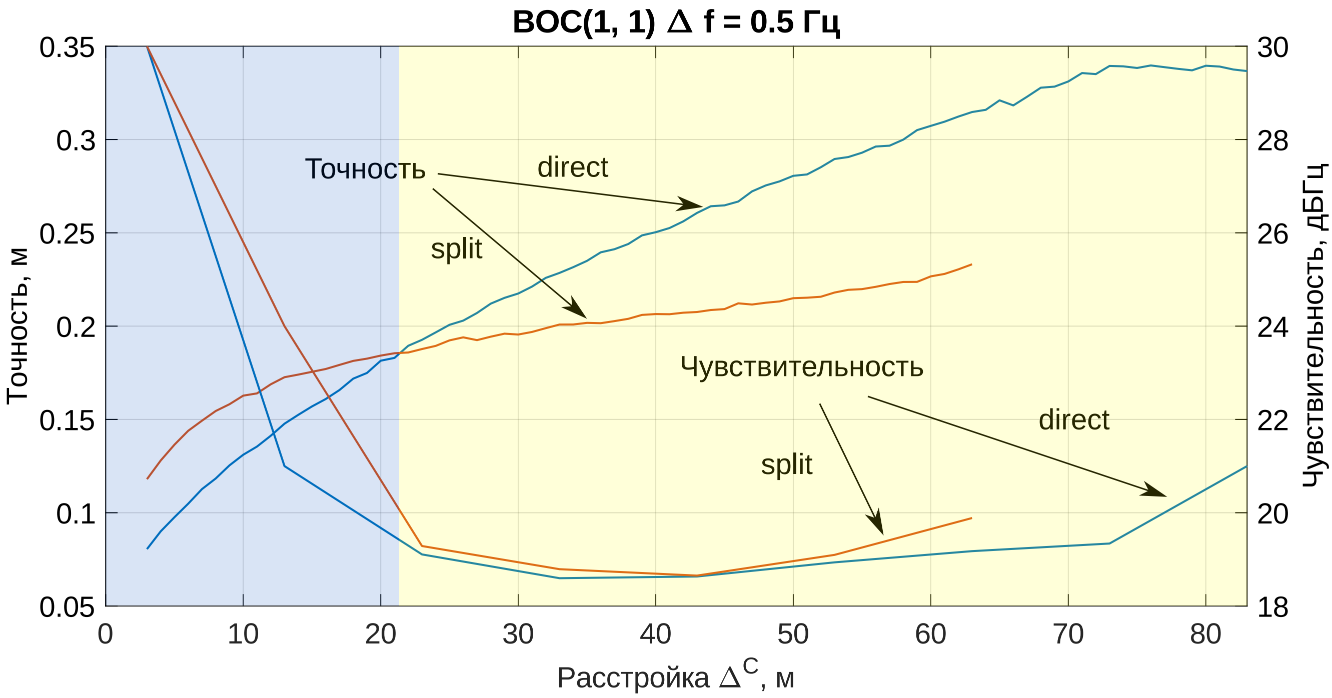

Рисунок 5.15 — Зависимости точности и чувствительности от расстройки для двух случаев: split и direct. Сигнал BOC(1,1)

Рассмотрим полученный график зависимости точности и чувствительности от расстройки (см. рис. 5.15) подробнее. Здесь можно выделить две области по расстройке Δ^C между компонентами early и promt:

— синяя – здесь direct алгоритм выигрывает и по точности, и по чувствительности,

— желтая – здесь split алгоритм выигрывает по точности, а direct алгоритм выигрывает по чувствительности.

По графику видно, что чувствительность алгоритмов дискриминатора split и direct сопоставима. Теперь рассмотрим график точности, видно, что при малых расстройках split алгоритм дискриминатора немного уступает в точности direct алгоритму, но начиная с расстройки ≈ 21 м split алгоритм превосходит direct алгоритм в точности. В реальных условиях приема, когда сигналы пропущены через фильтры в РЧБ использовать малые расстройки нерационально. Так как при прохождении через фильтр фронты сигнала, ввиду ограниченной полосы фильтра, будут сглаживаться, а значит и корреляционная функция сигнала уже не будет иметь острый пик, а будет также сглажена.

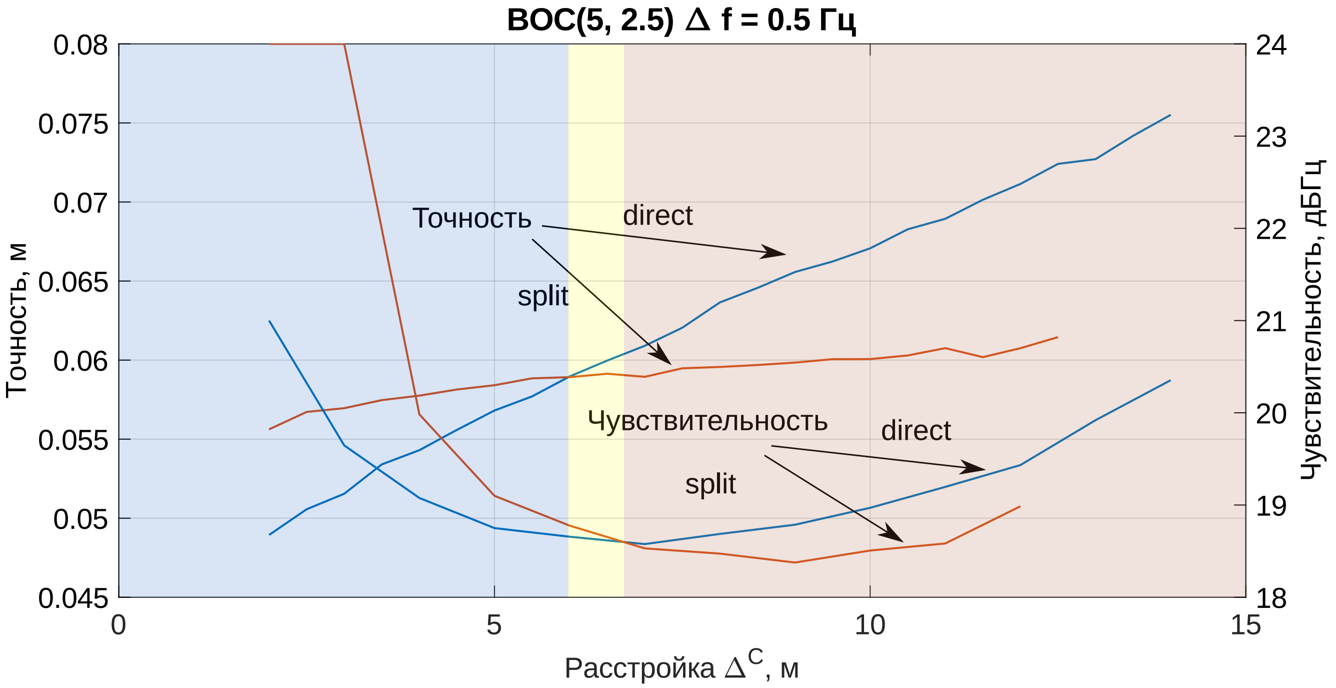

Рисунок 5.16 — Зависимости точности и чувствительности от расстройки для двух случаев: split и direct. Сигнал BOC(5,2.5)

Рассмотрим рисунок 5.16 для случая сигнала с модуляцией BOC(5,2.5). Здесь можно выделить три области по расстройке Δ^C между компонентами early и promt:

— синяя – здесь direct алгоритм выигрывает и по точности, и по чувствительности,

— желтая – здесь split алгоритм выигрывает по точности, а direct алгоритм выигрывает по чувствительности,

— красная – здесь split алгоритм выигрывает и по точности, и по чувствительности.

Учитывая, что использование малых расстроек при реальных условиях приема не оправдано, более рационально использование расстроек из красной и желтой областей.

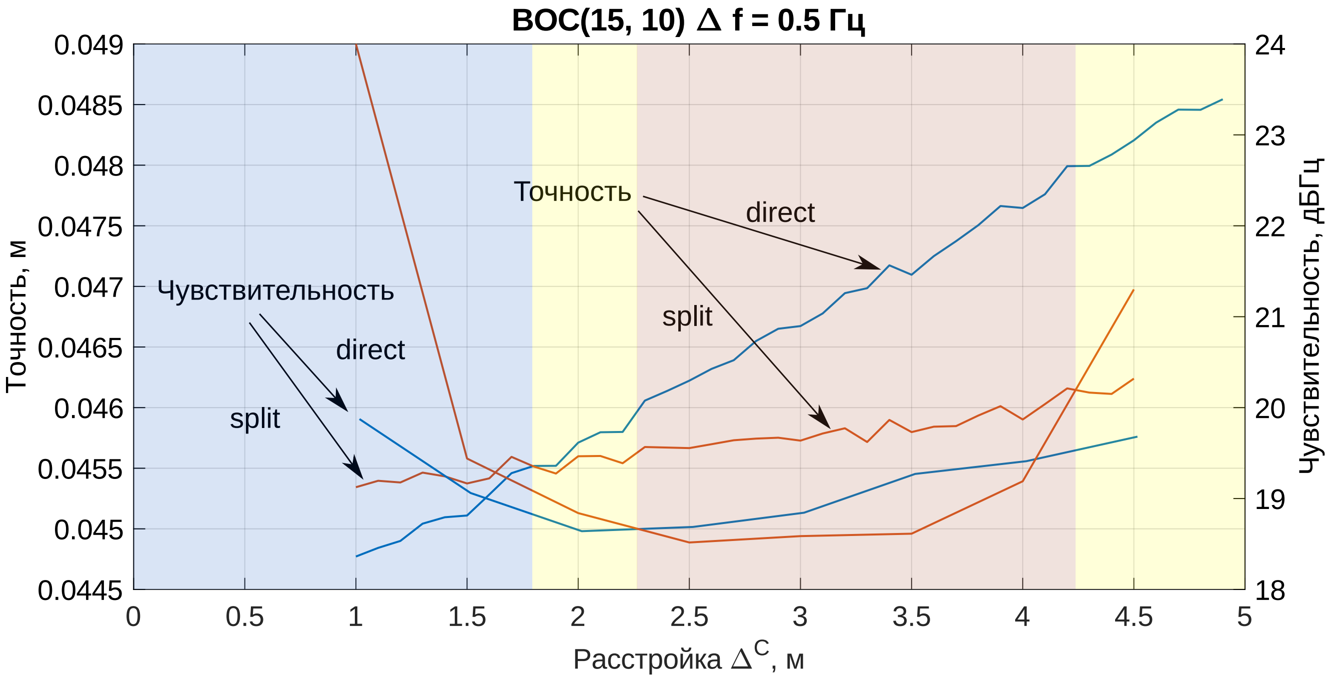

Рисунок 5.17 — Зависимости точности и чувствительности от расстройки для двух случаев: split и direct. Сигнал BOC(15,10)

Аналогично для сигнала с модуляцией BOC(15,10) можно выделить три области по расстройке Δ^C между компонентами early и promt:

— синяя – здесь direct алгоритм выигрывает и по точности, и по чувствительности,

— желтая – здесь split алгоритм выигрывает по точности, а direct алгоритм выигрывает по чувствительности,

— красная – здесь split алгоритм выигрывает и по точности, и по чувствительности.

Здесь также предпочтительнее использование расстроек из красной и желтой областей.

5.3 Характеристики захвата

В разделе 4.3 был рассмотрен алгоритм раздельного втягивания с использование сплит корреляторов, который может применяться для разрешения неоднозначности при захвате на слежение BOC сигналов.

Для получения характеристик захвата было проведено моделирование системы слежения за задержкой сигнала. Модель составлена на статистическом эквиваленте дискриминатора. Листинг программы на языке Matlab приведен в Приложении Г.

Для сравнения характеристик захвата моделировались два алгоритма:

1) Direct – моделируется ССЗ и алгоритм дискриминатора с использованием корреляционных сумм в прямой форме.

2) Split – моделируется ССЗ и алгоритм дискриминатора с использованием сплит корреляционных сумм. Применяется режим раздельного управления при втягивании системы слежения.

При моделировании split алгоритма в начале времени моделирования осуществляется управление только задержкой огибающей дальномерного кода τ^C, при фиксированной задержке цифровой поднесущей τ^B. Причем начальное значение задержки поднесущей принимается равным начальному значению задержки огибающей дальномерного когда. Время моделирования составляет 20 секунд, причем режим раздельного втягивания работает первые 10 секунд, а далее включается управление задержкой поднесущей.

Результатом работы каждого алгоритма является оценка задержки. Полученная оценка является случайной величиной.

В качестве метрики вводятся три вероятности:

1) Захват (захват за главный пик) – вероятность попадания оценки в главный пик АКФ,

2) Ложный захват (захват за боковой пик) – вероятность попадания оценки в интервалы, соответствующие боковым пикам АКФ,

3) Срыв (отсутствие захвата) – вероятность попадания оценки за пределы корреляционного пика.

В разных экспериментах при наборе статистики алгоритмы стартуют со случайной задержкой внутри заданного окна, то есть со случайной начальной ошибкой. В качестве оценки задержки, при расчете вероятности, берется последняя задержка на интервале моделирования.

Для наглядности на дискриминационной характеристики показаны три зоны оценки задержки: захват, ложный захват и срыв слежения для трех типов сигнала (см. рис 5.18 - 5.20).

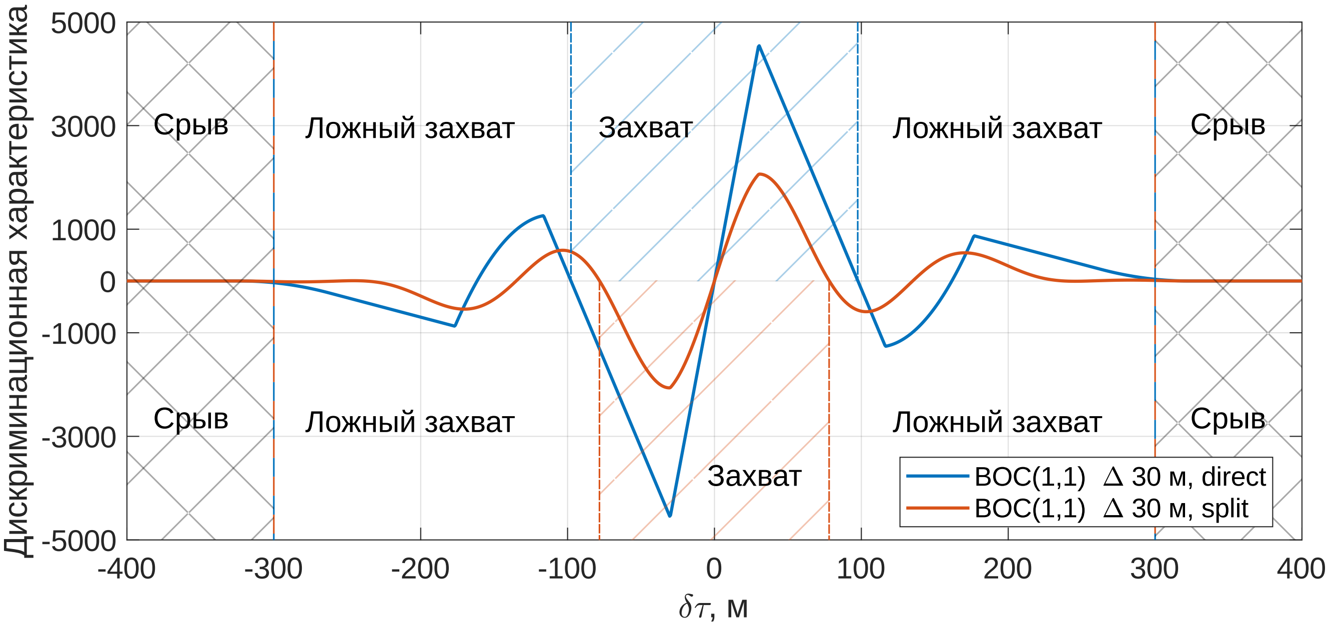

Рисунок 5.18 — Сигнал BOC(1,1)

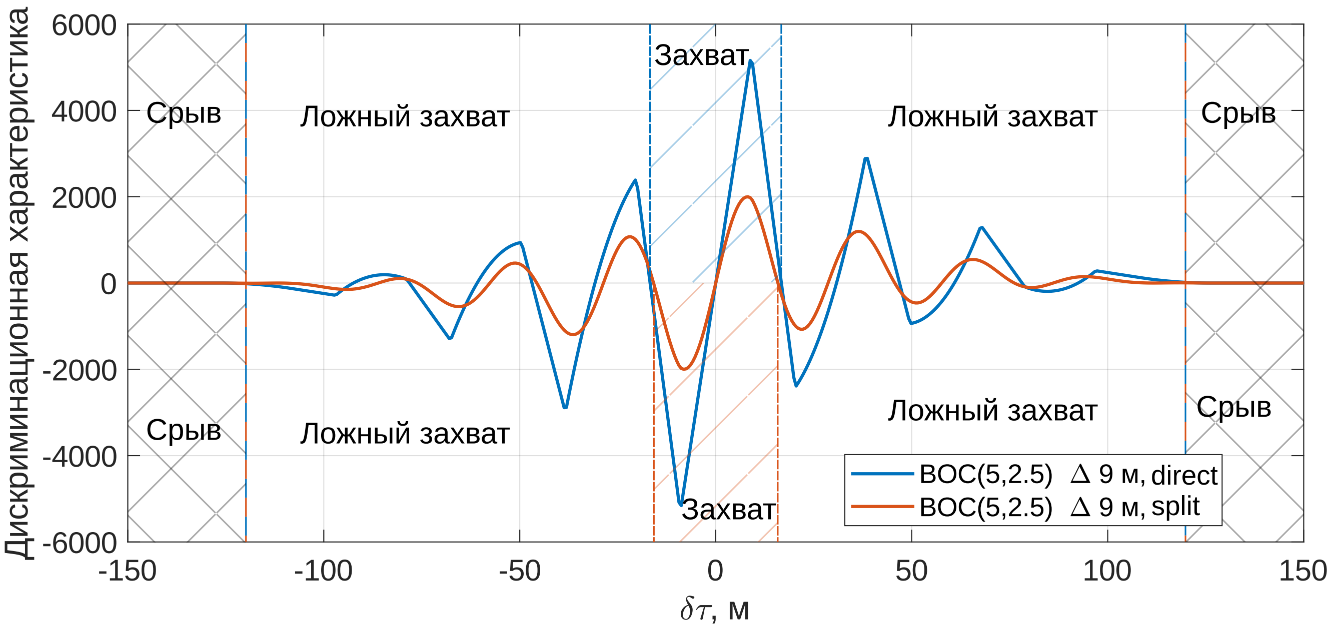

Рисунок 5.19 — Сигнал BOC(5,2.5)

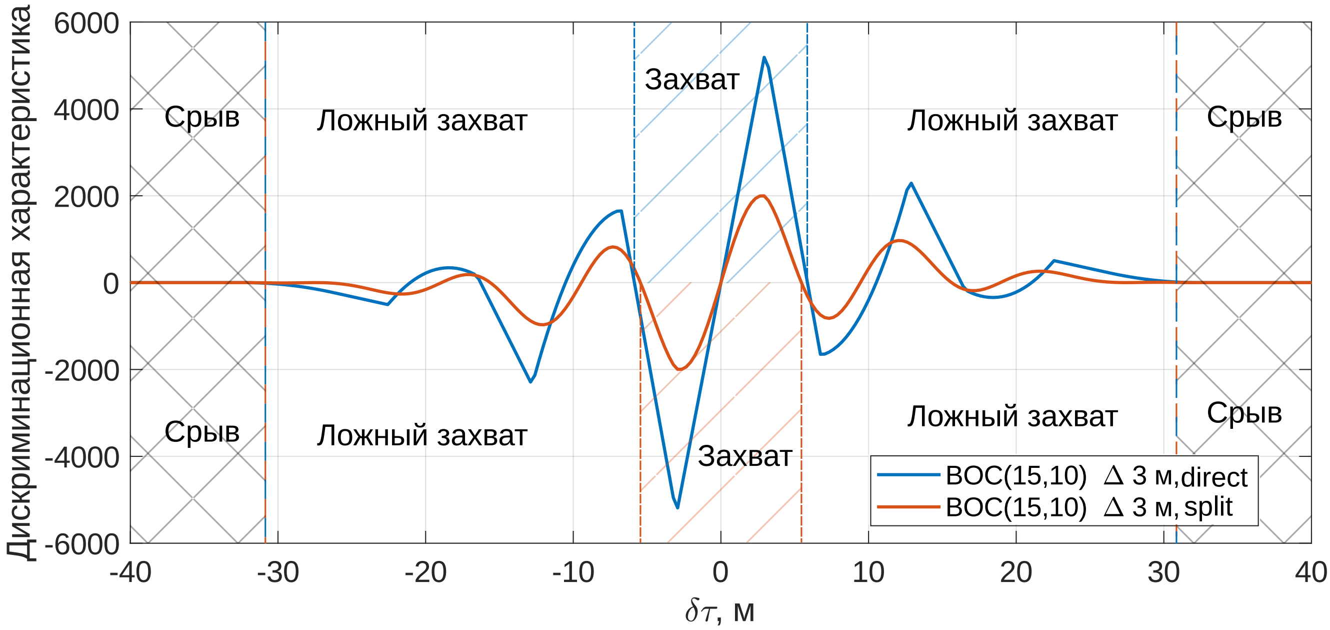

Рисунок 5.20 — Сигнал BOC(15,10)

5.3.1 Верификация имитационной модели

Рассмотрим случай обработки сигнала с модуляцией BOC(1,1). По графику 5.18 выберем три значения начальной ошибки слежения:

1) $\delta\tau = 50$ м, точка находится в зоне "Захват" ,

2) $\delta\tau = 170$ м, точка находится в зоне "Ложный захват" ,

3) $\delta\tau = 350$ м, точка находится в зоне "Срыв".

Моделируются три реализации процесса втягивания при использовании алгоритма дискриминатора direct с тремя значениями начальной ошибки слежения. Моделирование проводилось при отношении сигнал/шум $q_{c/n0} = 35$ дБГц и полосе следящей системы $\Delta f = 3$ Гц.

Ожидаемый результат:

В случае начальной ошибки слежения из зоны захвата $\delta\tau = 50$ м ожидается, что после переходного процесса, ошибка слежения за задержкой будет иметь установившееся значение в области нуля. В случае начальной ошибки слежения из зоны ложного захвата $\delta\tau = 170$ м ожидается, что после переходного процесса, ошибка слежения за задержкой будет иметь значение примерно равное расстоянию до второго нуля дискриминационной характеристики, то есть $\delta\tau \approx 160$ м. В случае начальной ошибки слежения из зоны срыва слежения $\delta\tau = 350$ м ожидается возрастание ошибки слежения за задержкой. Это приводит к выходу ошибки слежения за апертуру дискриминационной характеристики, а следовательно к срыву слежения =(.

Результаты работы модели для трех начальный значений ошибки слежения за задержкой сигнла приведены на рисунках 5.21 - 5.23. На каждом графике также отмечены сплошными синими линиями – апертура дискриминационной характеристики, а пунктирными линиями – границы зоны захвата.

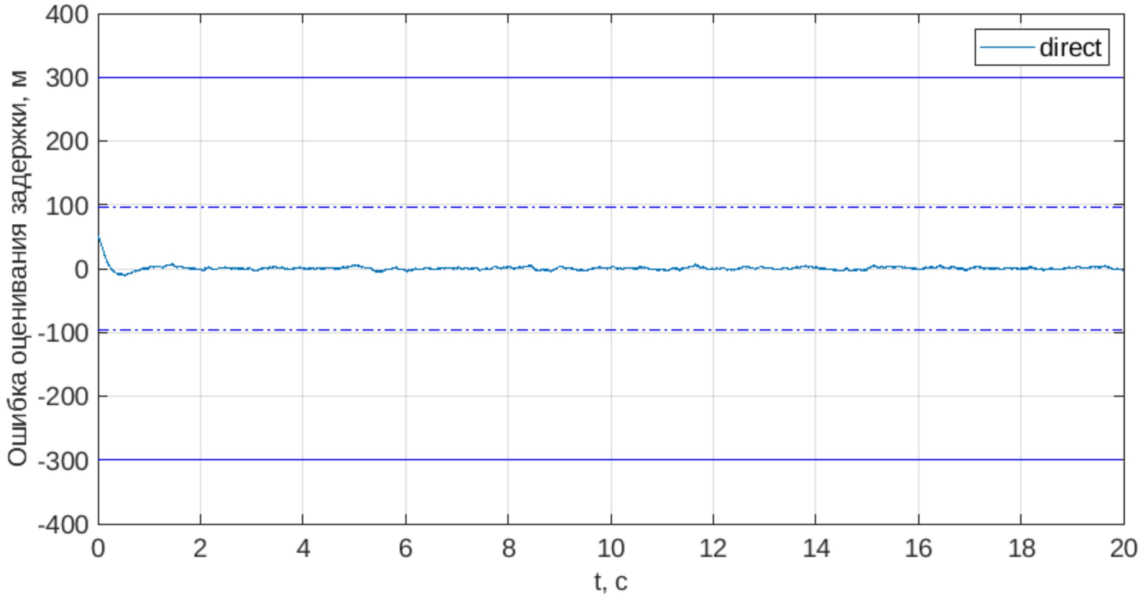

Рисунок 5.21 — Реализация захвата. Сигнал BOC(1,1)

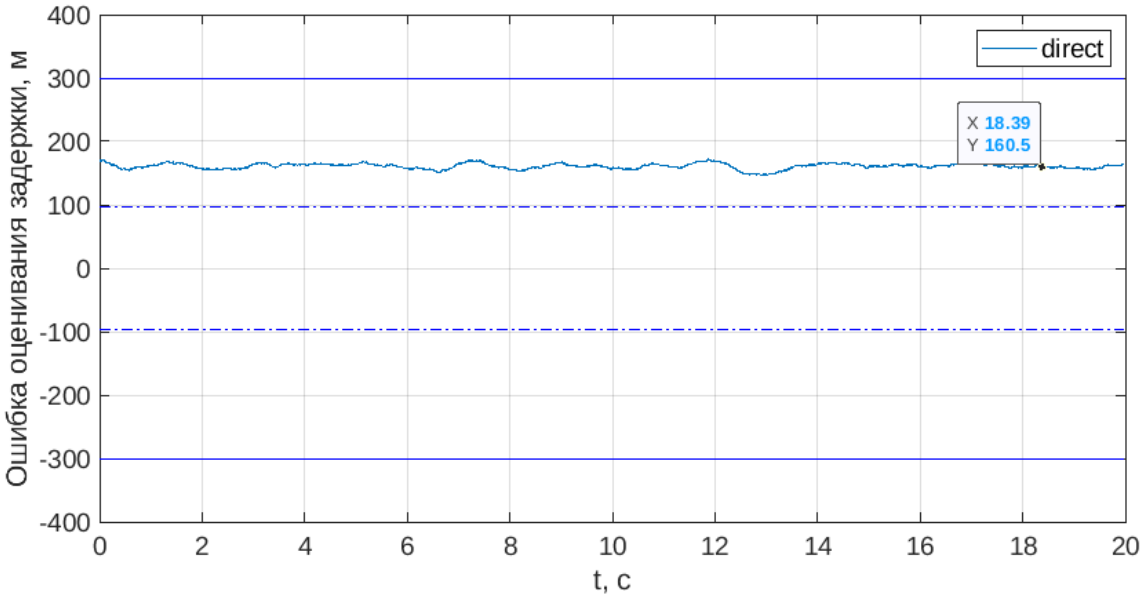

Рисунок 5.22 — Реализация ложного захвата. Сигнал BOC(1,1)

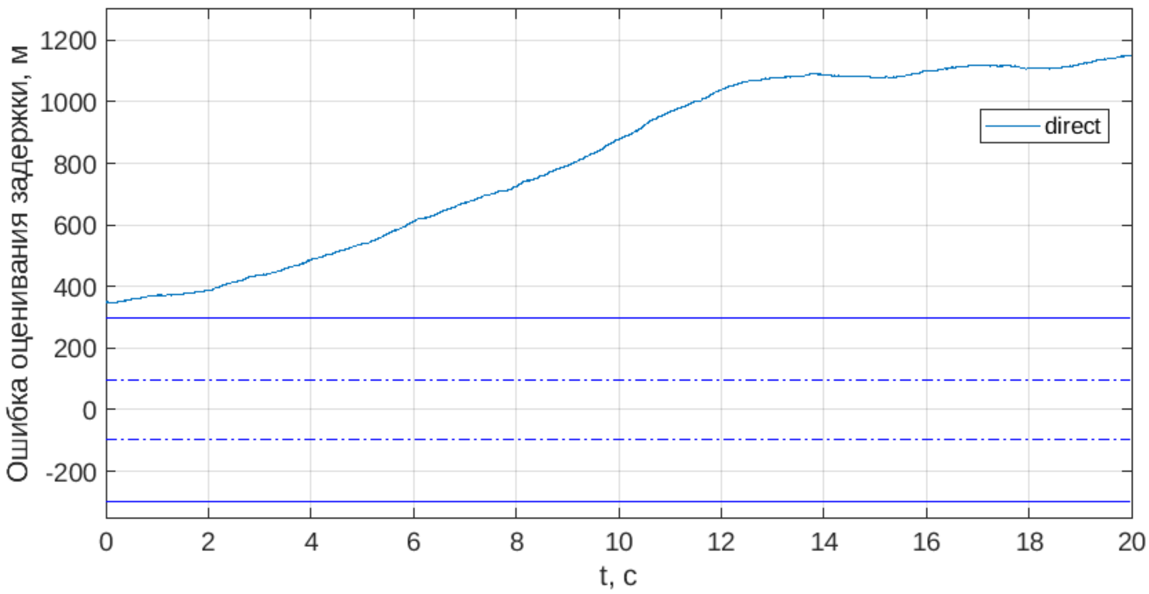

Рисунок 5.23 — Реализация срыва слежения. Сигнал BOC(1,1)

Как и ожидалось, для случая начального значения ошибки слежения из зоны "Захват" , ошибка слежения после переходного процесса устанавливается около нуля, для случая начального значения из зоны "Ложный захват" , ошибка слежения после переходного процесса устанавливается около 160 м, что соответствует расположению ложного нуля дискриминационной характеристики, а для случая начального значения ошибки слежения из зоны "Срыв" наблюдается возрастание ошибки слежения, выходящее за апертуру дискриминационной характеристики.

5.3.2 Результаты моделирования

Рассмотрим полученные с помощью модели реализации работы алгоритмов втягивания.

Начальная расстройка по задержке была выбрана равной 250 метров. Рассмотрим работу двух алгоритмов втягивания direct и split.

На рисунке 5.24 приведена реализация работы direct алгоритма, также зеленой точкой отмечено начальное значение ошибки слежения, а красной точкой – конечная ошибка слежения. Как видно из рисунка, мы наблюдаем реализацию ложного захвата.

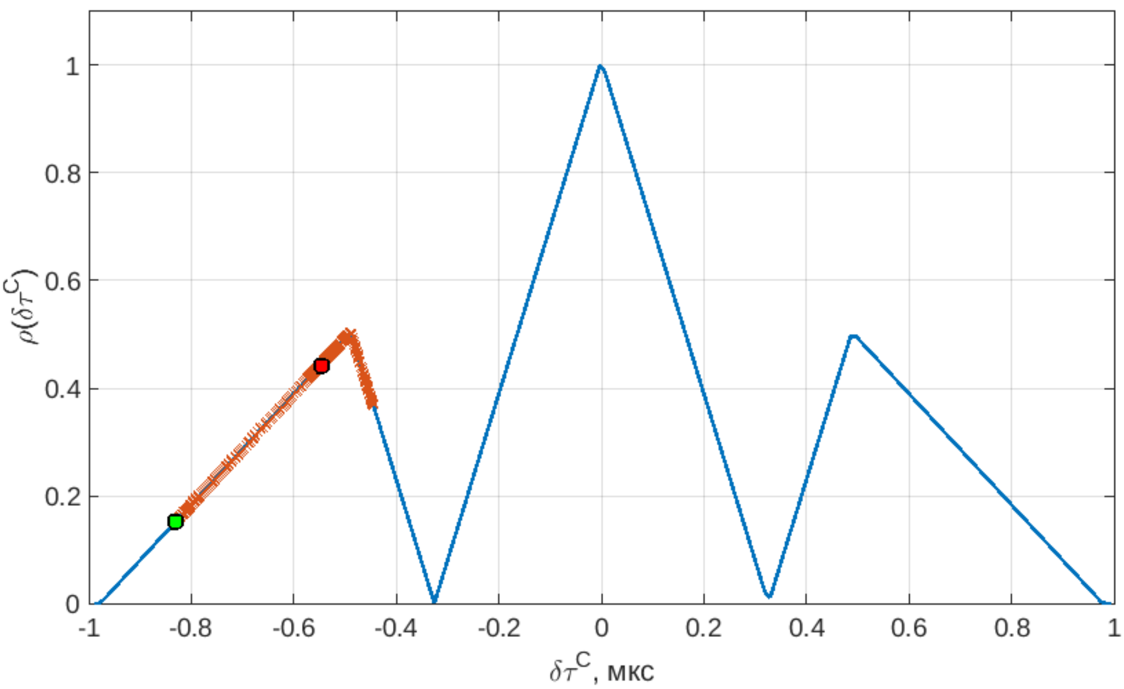

Рисунок 5.24 — Реализация ложного захвата при использовании direct метода

На рисунке 5.25 приведена реализация реализация работы split алгоритма. На рисунке оранжевым цветом показан процесс втягивания системы слежения за задержкой. Первые 10 секунд моделирования работает режим раздельного втягивания (т.е $\delta\tau^B = const$). В этом режиме ошибка по коду сводится практически к нулю. Затем режим раздельного втягивания выключается и включается управление задержкой поднесущей, то есть $\delta\tau^B = \delta\tau^C$. На рисунке 5.25 моменту переключения режимов соответствует переход на срез поверхности $\delta\tau^B = \delta\tau^C$ (срез поверхности обозначен зеленым цветом). Затем происходит процесс втягивания с управлением $\delta\tau^B$.

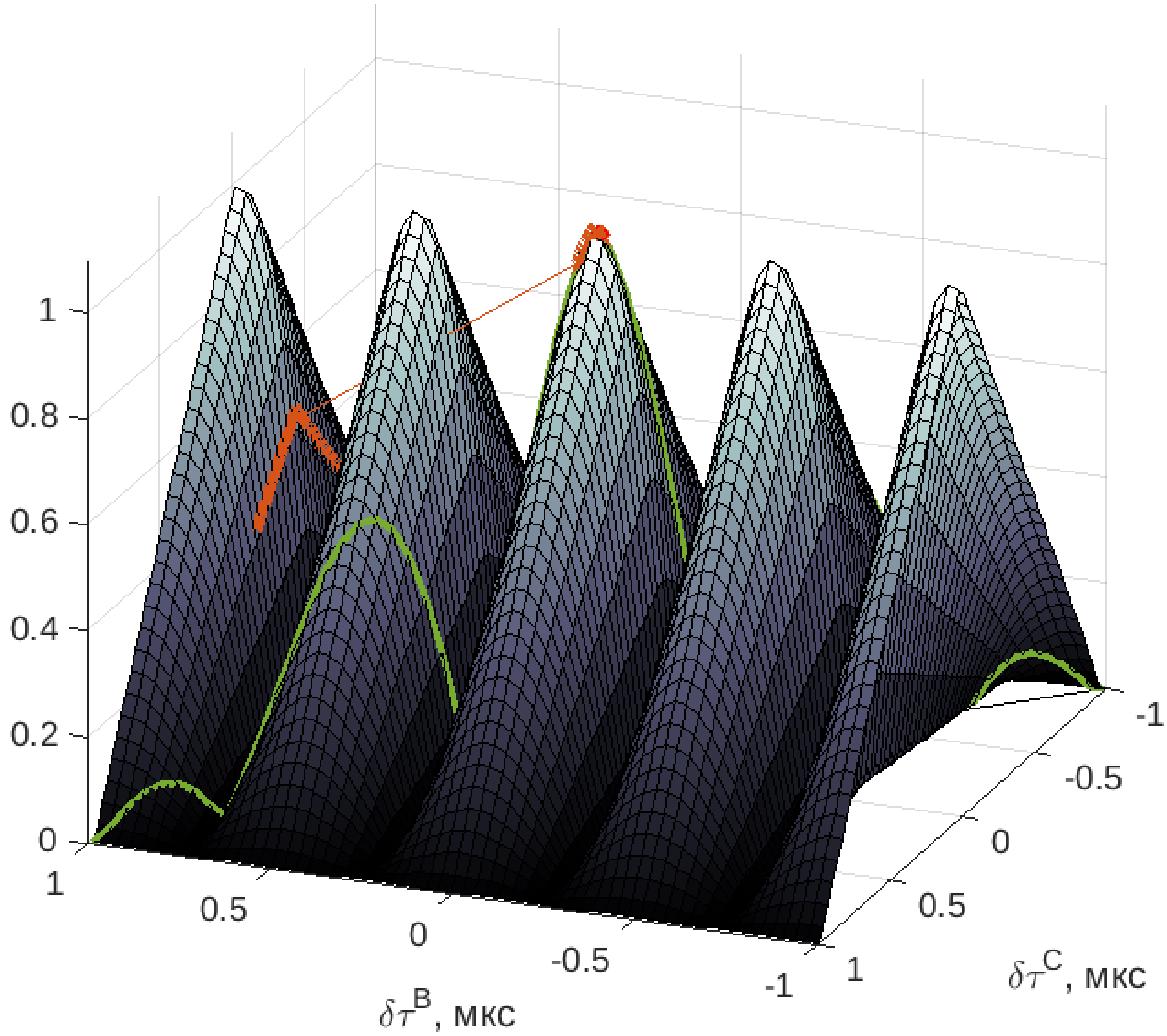

Рисунок 5.25 — Реализация захвата при использовании split алгоритма

По полученным реализациям (см. рис. 5.24 - 5.25.) видно, что даже при условии, что начальная ошибка слежения попала в ложный пик АКФ, split алгоритм позволяет избежать захвата за ложный пик, в отличии от direct алгоритма.

С помощью модели получены зависимости вероятностей захвата, ложного захвата и срыва от отношения сигнал/шум для трех типов модуляции. Моделирование проводилось при фиксированной полосе $\Delta f = 3$ Гц, усреднение проводилось по 2000 реализациям на каждое отношение сигнал/шум, время моделирования составляло 20 секунд, из них 10 секунд моделировался режим раздельного втягивания и 10 секунд обычное втягивание. Полученные зависимости приведены на рисунках 5.26 - 5.28.

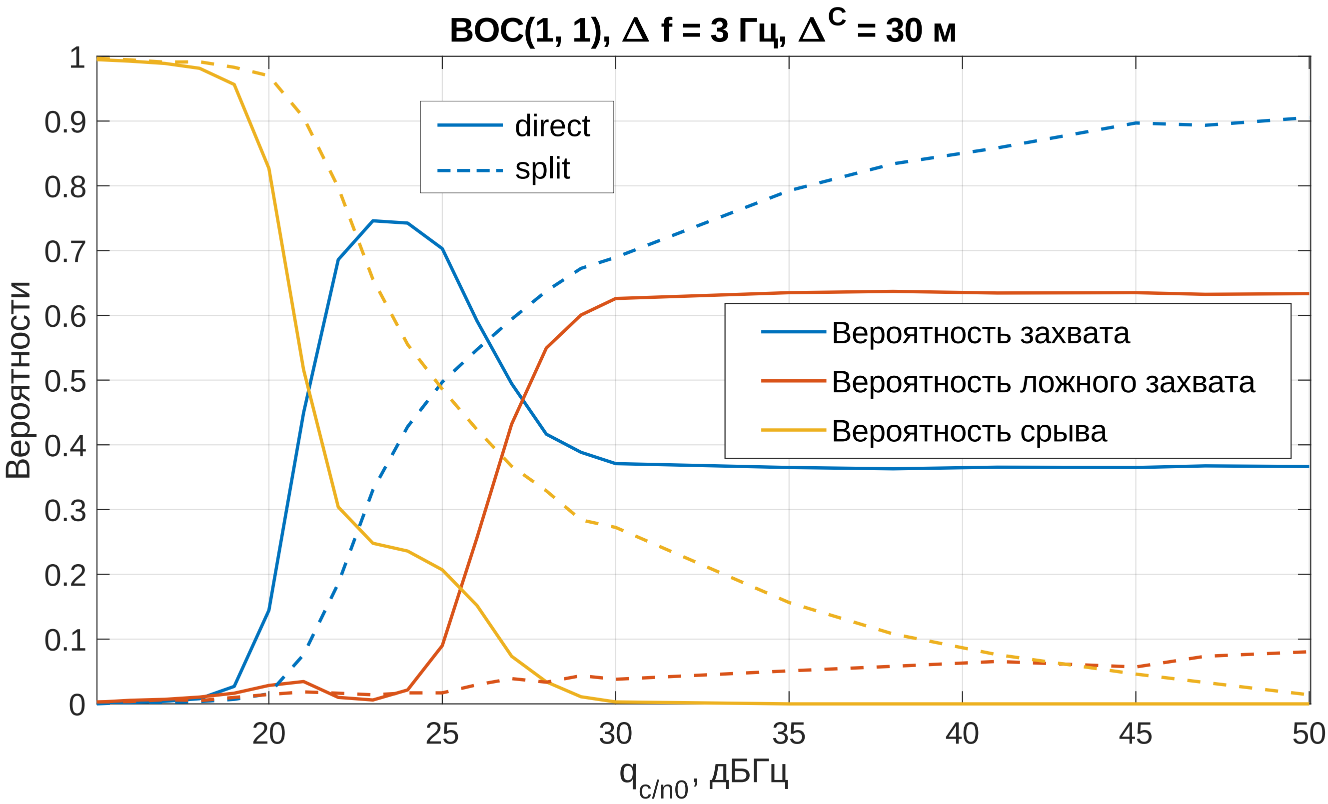

Рисунок 5.26 — Характеристики захвата для сигнала BOC(1,1)

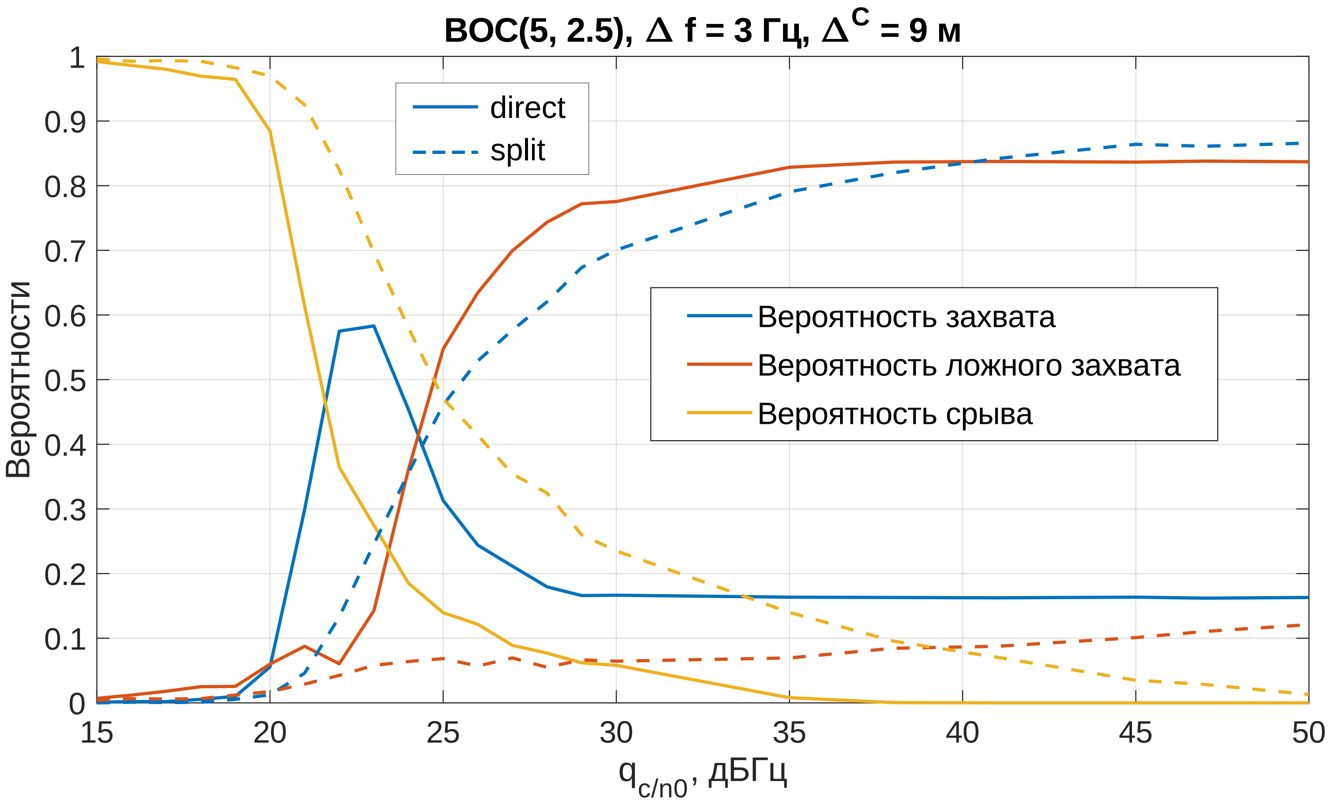

Рисунок 5.27 — Характеристики захвата для сигнала BOC(5,2.5)

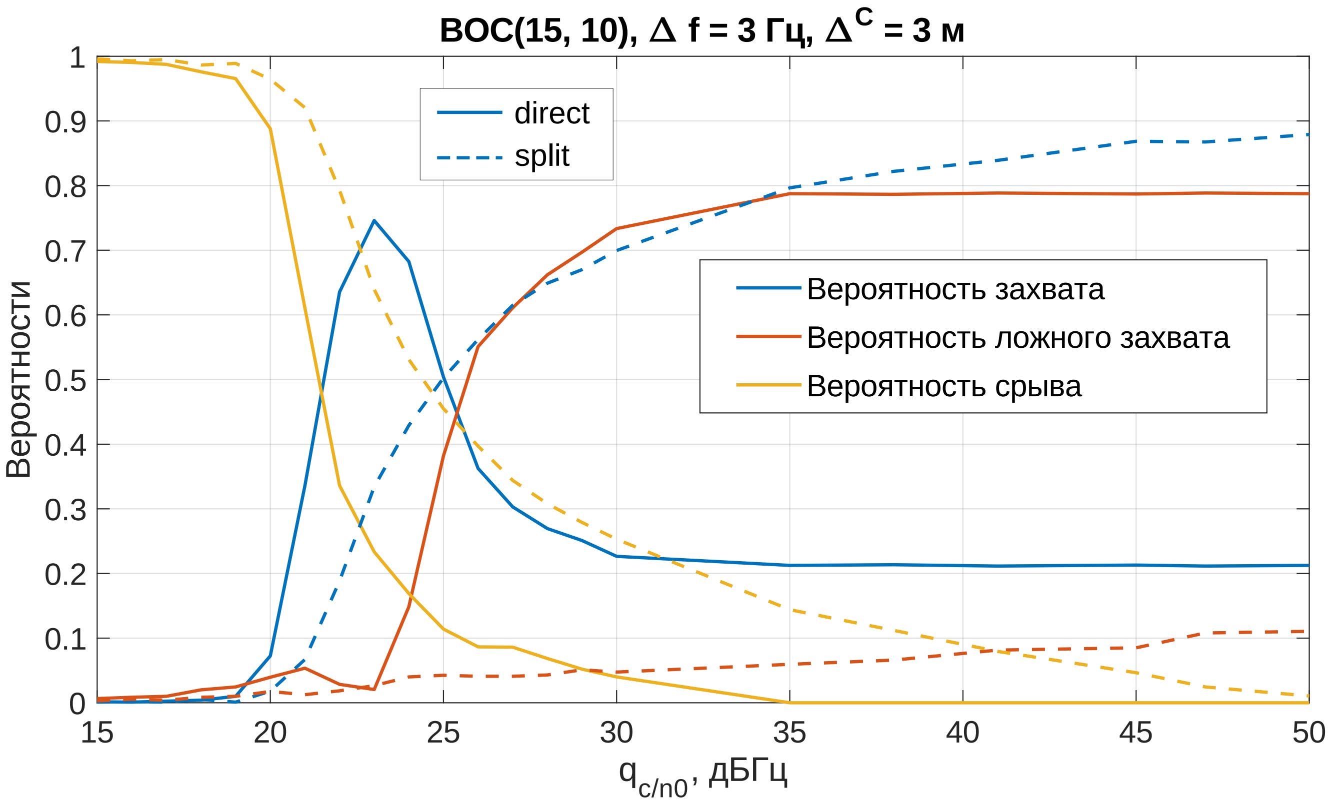

Рисунок 5.28 — Характеристики захвата для сигнала BOC(15,10)

Самым плохим результатом работы системы слежения является состояние ложного захвата, так как система считает, что ошибка сведена к нулю и не предпринимает никаких действий к ее уменьшению. На самом деле присутствует ошибка слежения, например, для случая обработки сигнала BOC(1,1) ошибка составляет около 160 метров. По графикам 5.26 - 5.28 видно, что алгоритм втягивания split имеет очень низкую вероятность ложного захвата по сравнению с прямым алгоритмом direct, что является явным преимуществом. Также split алгоритм имеет бОльшее значение вероятности захвата чем direct алгоритм начиная с отношения сигнал/шум более 25 дБГц.

Резкое возрастание вероятности захвата для direct алгоритма при уменьшении отношения сигнал/шум связано с пропорциональном уменьшении вероятности ложного захвата. Это связано с тем, что при уменьшении отношения сигнал/шум возрастают шумы в системе, поэтому зацепиться за ложный ноль дискриминационной характеристики сложнее, так как процесс сильно зашумлен и проскакивает стабильные участки ДХ в районе ложных нулей. Поэтому процесс втягивания заканчивается либо захватом за главный пик КФ, либо срывом слежения.

ЗАКЛЮЧЕНИЕ

В работе проведен синтез и анализ системы слежения за задержкой BOC сигналов при использовании двух корреляторов, изначально предназначенных для приема только BPSK сигналов. Такой подход не требует модификации аппаратной части навигационного приемника.

Работоспособность алгоритма и статистические характеристики проверены имитационным моделированием.

Получен алгоритм дискриминатора задержки (split) при использовании сплит-корреляционных сумм, найдены его статистические характеристики. Работоспособность алгоритма и статистические характеристики проверены имитационным моделированием.

Показана возможность раздельного управления параметрами задержки огибающей для дальномерного кода $\tau_k \to \tau_k^C$, $\Delta \to \Delta^C$ и поднесущей $\tau_k \to \tau_k^B$, $\Delta \to \Delta^B$ в опорном сигнале, что используется для разрешения неоднозначности задержки огибающей BOC сигнала, вызванной многопиковым характером его корреляционной функции. Предложен алгоритм втягивания системы слежения за задержкой сигнала, снижающий вероятность ложного захвата.

Возможность отдельной настройки каналов коррелятора позволяет обрабатывать сигналы с AltBOC модуляцией. Также это позволяет обрабатывать BOC и AltBOC сигналы, пропуская разные лепестки спектра через разные радиочастотные тракты, что снижает требования к последним. Описан случай такой обработки с помощью популярной микросхемы NT1065 “Nomada”.

Имитационное моделирование синтезированного алгоритма проводилось в сравнении с алгоритмом дискриминатора, использующим корреляционные суммы в прямой форме (direct).

С помощью моделирования получены экспериментальные характеристики слежения за задержкой сигнала. В области высоких отношений сигнал/шум (от 30 дБГц) точность (СКО ошибки оценивания задержки сигнала) алгоритмов практически совпадает. Показано, что при определенных значениях расстройки между early и promt компонентами коррелятора split алгоритм имеет лучшие (по сравнению с direct алгоритмом) значения точности и чувствительности слежения.

При помощи имитационной модели выявлено, что применение алгоритма втягивания системы слежения за задержкой сигнала с использованием сплит-корреляционных сумм позволяет свести практически к нулю вероятность ложного захвата.

Литература

1. Wendel Jan, Schubert Frank, Hager S. A Robust Technique for Unambiguous BOC Tracking // Navigation. — 2014. — 09. — Vol. 61.

2. Simsky Andrew, Sleewaegen Jean-Marie. Experimental and Professional Galileo Receivers. — 2015. — 09. — P. 273–288. — ISBN: 978-94-007-1829-6.

3. ГЛОНАСС. Принципы построения и функционирования. — изд. 4-е, перераб. и доп. / А. И. Перов, В. Н. Харисов, Р. В. Бакитько и др. — М.: Радиотехника, 2010.

4. Шатилов А. Ю. Характеристики радиосигналов глобальных спутниковых радионавигационных систем ГЛОНАСС, GPS, GALILEO, BEIDOU и функциональных дополнений SBAS. — Изд-во МЭИ, 2015.

5. Analysis of Non Ambiguous BOC Signal Acquisition Performance / Vincent Calmettes, Vincent Heiries, Daniel Roviras, Lionel Ries. — 2004. — 01.

6. П.В. Штро, Т.В. Краснов, В.Ф. Гарифулин. Обзор алгоритмов поиска перспективных сигналов ГЛОНАСС с кодовым разделением // Успехи современной радиоэлектроники. — 2016. — С. 157–160.

7. Musso Maristella, Cattoni Andrea, Regazzoni Carlo. A New Fine Tracking Algorithm for Binary Offset Carrier Modulated Signals. — 2006. — 09.

8. Fine Paul, Wilson Warren. Tracking Algorithm for GPS Offset Carrier Signals. — 1999. — 01. — P. 671 – 676.

9. Unambiguous Tracking Technique Based on Combined Correlation Functions for Sine BOC Signals / Tian Li, Zuping Tang, Jiaolong Wei et al. // Journal of Navigation. — 2019. — Vol. 72, no. 1. — P. 140–154.

10. Fante Ronald. Unambiguous Tracker for GPS Binary-Offset- Carrier Signals. — 2003. — 01.

11. Yao Zheng. Unambiguous Processing Techniques of Binary Offset Carrier Modulated Signals. — 2012. — 02. — ISBN: 978-953-307-843-4.

12. Шатилов А. Ю. Модель уходов частоты опорного генератора приемника СРНС. — https://www.srns.ru/images/1/15/20120310_Oscillator_model.pdf.

13. Перов А. И., В.Н. Замолодчиков, В.М. Чиликин. Радиоавтоматика. — М.: Радиотехника, 2014.

14. E.D. Kaplan, C.J. Hegarty. Understanding GPS: Principles and Applications. — second edition. — Artech House, 2005. — ISBN: 1580538940.

Приложение А

Листинг программы проверки дискриминационных характеристик дискриминатора задержки

main.m

```
close all; clear; clc
tauChip = 1e-3/1023;
lightC = physconst('LightSpeed');
DCB = 1/99.375e6;
default = {'m', 15,  'n',  10, 'split', true, 'noise', false,...
    'deltaC', tauChip/5, 'deltaB', tauChip/5,...
    'analit', false, 'newcolor', true, 'solid', true, 'norm', true};
DD = ...
    {
     {'m', 1,  'n',  1, 'split', false,   'noise', true,...
     'deltaC', 10*DCB, 'deltaB', 10*DCB,...
     'analit', false, 'newcolor', true, 'solid', false},...
     {'m', 1,  'n',  1, 'split', true,   'noise', true,...
     'deltaC',  10*DCB, 'deltaB', 10*DCB,...
     'analit', false, 'newcolor', true, 'solid', false},...
     {'m', 1,  'n',  1, 'split', false,   'noise', false,...
     'deltaC', 10*DCB, 'deltaB', 10*DCB,...
     'analit', true, 'newcolor', true, 'solid', true},...
     {'m', 1,  'n',  1, 'split', true,   'noise', false,...
     'deltaC',  10*DCB, 'deltaB', 10*DCB,...
     'analit', true, 'newcolor', true, 'solid', true}%,...
%      {'m', 5,  'n',  2.5, 'split', false,   'noise', true,...
%      'deltaC',  3*DCB 'deltaB', 3*DCB,...
%      'analit', false, 'newcolor', true, 'solid', false},...
%      {'m', 5,  'n',  2.5, 'split', true,   'noise', true,...
%      'deltaC',  3*DCB, 'deltaB', 3*DCB,...
%      'analit', false, 'newcolor', true, 'solid', false}, ...
%      {'m', 5,  'n',  2.5, 'split', false,   'noise', false,...
%      'deltaC',  3*DCB 'deltaB', 3*DCB,...
%      'analit', true, 'newcolor', true, 'solid', true},...
%      {'m', 5,  'n',  2.5, 'split', true,   'noise', false,...
%      'deltaC',  3*DCB, 'deltaB', 3*DCB,...
%      'analit', true, 'newcolor', true, 'solid', true}%, ...
%      {'m', 15,  'n',  10, 'split', false,   'noise', true,...
```

```
%      'deltaC', 1*DCB, 'deltaB', 1*DCB,...
%      'analit', false, 'newcolor', true, 'solid', false},...
%      {'m', 15,  'n',  10, 'split', true,   'noise', true,...
%      'deltaC',  1*DCB, 'deltaB', 1*DCB,...
%      'analit', false, 'newcolor', true, 'solid', false},...
%      {'m', 15,  'n',  10, 'split', false,   'noise', false,...
%      'deltaC', 1*DCB, 'deltaB', 1*DCB,...
%      'analit', true, 'newcolor', true, 'solid', true},...
%      {'m', 15,  'n',  10, 'split', true,   'noise', false,...
%      'deltaC',  1*DCB, 'deltaB', 1*DCB,...
%      'analit', true, 'newcolor', true, 'solid', true},...
    };
for i = 1:length(DD)
    S = struct(DD{1,i}{:});
    f = fieldnames(S);
    D(i) = struct(default{:});
    for j = 1:size(f,1)
        D(i).(f{j}) = S.(f{j});
    end
end

for i = 1:length(D)
    D(i).tauChip = tauChip/D(i).n;
%     D(i).deltaC =  D(i).deltaC/D(i).n;
%     D(i).deltaB =  D(i).deltaB/D(i).n;
end

sep_pull_in = 0; % режим раздельного втягивания
Np = 1000;
Tc = 5e-3;
qcno_dB = 35;
stdn_IQ = 13;
qcno = 10^(qcno_dB/10);
A_IQ = stdn_IQ * sqrt(2 * qcno * Tc);
LightC = 3e8;
tauIst = tauChip/5;
tauExtr= tauIst-1*tauChip:1*tauChip/1000:tauIst+1*tauChip;
NtauExtr = length(tauExtr);

deltaPhi = 0;% 1*rand(1,1)*2*pi;
```

```
deltaW = 0;% 1*10*2*pi;

Ud = zeros(length(D),NtauExtr);
Udteor = zeros(length(D),NtauExtr);

p_promt = nan(length(D),NtauExtr);
p_early = nan(length(D),NtauExtr);
p_late  = nan(length(D),NtauExtr);
deltaTauC = nan(length(D),NtauExtr);
deltaTauB = nan(length(D),NtauExtr);
sqrt_P = nan(length(D),NtauExtr);
sqrt_E = nan(length(D),NtauExtr);
SdTeor = nan(1,length(D));
tauZero = nan(1,length(D));

for i = 1:length(D)
    fprintf('boc(%d,%d), split %d, analit %d\n', D(i).m, D(i).n,
      D(i).split, D(i).analit);
    omega_B = 2*pi*D(i).m*1.023e6;
    if D(i).split == 1
        eval('Ud_split')
    else
        eval('Ud_nelp')
    end
end

set(0,'DefaultAxesFontSize', 14)
newcolors = [0.00,0.45,0.74    % blue
    0.85,0.33,0.10    % red
    0.93,0.69,0.13    % yellow
    0.49,0.18,0.56    % purple
    0.47,0.67,0.19    % green
    0.30,0.75,0.93    % light blue
    0,    0,    0]; % black
linestyle = {'--', '-.', ':', '-'}';
figure(1)
    % axis for dtau/tauchip
    b=axes('Position',[.1 .1 .8 1e-12], 'FontSize', 14);
    set(b,'Units','normalized');
    a = axes('Position',[.1 .2 .8 .7], 'FontSize', 14);
```

```
    set(a,'Units','normalized');

for i = 1:length(D)

    if D(i).solid == 0
        set(gca, 'LineStyleOrder', linestyle(1));
    end

    plot(a, deltaTauC(i,:)*lightC,
        sqrt_P(i,:)./max(sqrt_P(i,:)),'LineWidth',1); hold on;
    xlabel(a,'\delta\tau, м', 'FontSize', 14)%/\tau_{c}
    xlabel(b,'\delta\tau/\tau_{c}', 'FontSize', 14)%/\tau_{c}
    ylabel('\rho(\delta\tau)', 'FontSize', 14)

    if D(i).m == 0
        str_acf{i} = ['BPSK(',num2str(D(i).n), ')  split =  ',...
            num2str(D(i).split)];
    else
        str_acf{i} = ['BOC(',num2str(D(i).m), ',', num2str(D(i).n),...
            ')  split =  ',num2str(D(i).split)];
    end
    legend(str_acf)
    grid on
    set(gca,'LineStyleOrder','remove')
    set(gca,'ColorOrder','remove')

end

if D(i).m == 15
    kk = 0.1;
else
    kk =1;
end
set(a,'xlim',[(-D(i).tauChip - kk*50/lightC)*lightC (D(i).tauChip +
    kk*50/lightC)*lightC]);
set(b,'xlim',[(-D(i).tauChip - kk*50/lightC)/D(i).tauChip
    (D(i).tauChip + kk*50/lightC)/D(i).tauChip]); hold on
    plot(([-D(i).deltaC -D(i).deltaC])*lightC, [0 1], '-k',...
        ([D(i).deltaC D(i).deltaC])*lightC, [0 1], '-k')
```

```
figure(2)
for i = 1:length(D)

    if D(i).solid == 0
        set(gca, 'LineStyleOrder', linestyle(1));
    end

    if D(i).analit == 1
        if D(i).solid == 1
            plot(deltaTauC(i,:)*lightC, Udteor(i,:),'LineWidth',1);
               hold on;
        else
            plot(deltaTauC(i,:)*lightC, Udteor(i,:),
               '-.','LineWidth',1); hold on;
        end
    else
        plot(deltaTauC(i,:)*lightC, Ud(i,:),'LineWidth',1); hold on;
    end

    if D(i).split == 0 && D(i).m == 0
        str_dh{i} = ['BPSK(',num2str(D(i).n), ')  \Delta ',...
            num2str(D(i).deltaC*lightC),...
            ', analit = ' num2str(D(i).analit)];

    elseif  D(i).split == 0 && D(i).m ~= 0 %&& D(i).analit == 0
        str_dh{i} = ['BOC(',num2str(D(i).m), ',', num2str(D(i).n),...
            ')  \Delta ',num2str(D(i).deltaC*lightC),...
            ', analit = ' num2str(D(i).analit)];
    else
        str_dh{i} = ['BOC(',num2str(D(i).m), ',', num2str(D(i).n),...
            ')  \Delta ',num2str(D(i).deltaC*lightC),...
            '/', num2str(D(i).deltaB*lightC),...
            ', analit = ' num2str(D(i).analit)];
    end

    set(gca,'LineStyleOrder','remove')
    set(gca,'ColorOrder','remove')
    end

xlabel('\delta\tau m', 'FontSize', 16)
```

```
ylabel('U_{dll}', 'FontSize', 16)
ylim([min(min(Ud))-10 max(max(Ud))+10])
grid on
str = str_dh;
legend(str)
n = 1;
for i = 1:length(D)

    if D(i).analit == 1 || D(i).analit == 0

        plot(deltaTauC(i,:)*lightC, SdTeor(i).*deltaTauC(i,:), '--',
            'Color', newcolors(i,:),'LineWidth',1); hold on;
        if D(i).m == 0
            str_Sd{n} = ['S_{d} ', 'BPSK(',num2str(D(i).n),...
                ') split = ',num2str(D(i).split)];
        else
            str_Sd{n} = ['S_{d} ', 'BOC(',num2str(D(i).m),...
                ',', num2str(D(i).n), ') split =
                    ',num2str(D(i).split)];
        end
    else
        continue
    end
    n = n + 1;
end
if D(i).analit == 1
    str = [str str_Sd];
end
legend(str)

figure(3)
for i = 1:length(D)

    if D(i).newcolor == 0
        set(gca, 'ColorOrder', newcolors(length(newcolors),:));
    else
        set(gca, 'ColorOrder', newcolors(i,:));
    end
    if D(i).solid == 0
        set(gca, 'LineStyleOrder', linestyle(1));
```

```
    end

    if D(i).analit == 1 % && D(i).split == 1
        if D(i).solid == 1
            plot(deltaTauC(i,:)*lightC,
               Udteor(i,:)./SdTeor(i),'LineWidth',1); hold on;
        else
            plot(deltaTauC(i,:)*lightC, Udteor(i,:)./SdTeor(i),
               '-.','LineWidth',1); hold on;
        end
    else
        plot(deltaTauC(i,:)*lightC,
           Ud(i,:)./SdTeor(i),'LineWidth',1); hold on;
    end

    if D(i).split == 0 && D(i).m == 0
        str2_norm{i} = ['BPSK(',num2str(D(i).n), ')  \Delta ',...
            num2str(D(i).deltaC*lightC) ,...
            ', analit = ' num2str(D(i).analit)];

    elseif  D(i).split == 0 && D(i).m ~= 0 %&& D(i).analit == 0
        str2_norm{i} = ['BOC(',num2str(D(i).m), ',',
           num2str(D(i).n) ,...
            ')  \Delta ',num2str(D(i).deltaC*lightC) ,...
            ', analit = ' num2str(D(i).analit)];
    else
        str2_norm{i} = ['BOC(',num2str(D(i).m), ',',
           num2str(D(i).n) ,...
            ')  \Delta ',num2str(D(i).deltaC*lightC) ,...
            '/', num2str(D(i).deltaB*lightC) ,...
            ', analit = ' num2str(D(i).analit)];
    end

    set(gca,'LineStyleOrder','remove')
    set(gca,'ColorOrder','remove')
end
m = 1;
for i = 1:length(str2_norm)
    if isempty(str2_norm{i}) ~= 1
        str_norm{m} = str2_norm{i};
```

```
            m = m + 1;
        end
    end

xlabel('\delta\tau m', 'FontSize', 14)
ylabel('U_{dll}/ S_{d}', 'FontSize', 14)
% xlim([-1.5 1.5])
ylim([min(min(Udteor/ max(SdTeor)))-1e-8 ...
    max(max(Udteor/ max(SdTeor)))+1e-8])
grid on
legend(str_norm)
title('Нормированная дискриминационная характеристика')

logfile = ['ACF_BOC' num2str(D(1).m) '_' num2str(D(1).n) '.mat'];
save(logfile, 'deltaTauC', 'sqrt_P');
```

Ud_nelp.m

```
tauZero(i) = findZeroACF(D(i).m, D(i).n, D(i).tauChip);

SdTeor(i) = 8*qcno*Tc*stdn_IQ^2*sinc(deltaW*Tc/2 /pi)^2*...
    (1/tauZero(i) - (D(i).deltaC/tauZero(i)^2));

for k = 1:NtauExtr

    deltaTauC(i,k) = tauIst - tauExtr(k);
    deltaTauB(i,k) = deltaTauC(i,k);

    p_promt(i,k) = ro(deltaTauC(i,k), D(i).m, D(i).n, D(i).tauChip);
    p_late(i,k)  = ro(deltaTauC(i,k) - D(i).deltaC, D(i).m, D(i).n,
        D(i).tauChip);
    p_early(i,k) = ro(deltaTauC(i,k) + D(i).deltaC, D(i).m, D(i).n,
        D(i).tauChip);

    Dp = stdn_IQ^2;
    Dpe = ro(D(i).deltaC, D(i).m, D(i).n, D(i).tauChip)*stdn_IQ^2;
    Del = ro(D(i).deltaC*2, D(i).m, D(i).n, D(i).tauChip)*stdn_IQ^2;

    L=chol([Dp  Dpe Dpe;
        Dpe Dp  Del;
```

```
            Dpe  Del  Dp]) ';

        for  j = 1:Np
            nI  = L * randn(3,1);
            nQ  = L * randn(3,1);
            mIp  = A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2  /pi)*cos(deltaW*Tc/2
               + deltaPhi);
            mIe  = A_IQ*p_early(i,k)*sinc(deltaW*Tc/2  /pi)*cos(deltaW*Tc/2
               + deltaPhi);
            mIl  = A_IQ*p_late(i,k)  *sinc(deltaW*Tc/2  /pi)*cos(deltaW*Tc/2
               + deltaPhi);

            mQp  = -A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
               /pi)*sin(deltaW*Tc/2 + deltaPhi);
            mQe  = -A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
               /pi)*sin(deltaW*Tc/2 + deltaPhi);
            mQl  = -A_IQ*p_late(i,k)  *sinc(deltaW*Tc/2
               /pi)*sin(deltaW*Tc/2 + deltaPhi);

            Ip  = mIp + D(i).noise*nI(1,1);
            Ie  = mIe + D(i).noise*nI(2,1);
            Il  = mIl + D(i).noise*nI(3,1);
            Qp  = mQp + D(i).noise*nQ(1,1);
            Qe  = mQe + D(i).noise*nQ(2,1);
            Ql  = mQl + D(i).noise*nQ(3,1);

            Ud(i,k) = Ud(i,k) - (Ie^2+Qe^2) + (Il^2+Ql^2);

            sqrt_P(i,k) = sqrt(Ip^2 + Qp^2);
            sqrt_E(i,k) = sqrt(Ie^2 + Qe^2);
        end
        Ud(i,k) = Ud(i,k)/Np;

        Udteor(i,k) = 2*qcno* stdn_IQ^2 *Tc*(sinc(deltaW*Tc/2
           /pi)^2)*(-p_early(i,k)^2 + p_late(i,k)^2);
        if ~mod(k,100)
            fprintf('Progress: %.2f %%\n', k*100/NtauExtr)
        end
    end
```

Ud_split.m

```
if sep_pull_in
D(i).deltaB = 0;% режим раздельного втягивания
end

SdTeor(i) = ((2/pi)*A_IQ)^2 *(sinc(deltaW*Tc/2 /pi)^2)*... %sqrt(2)*
    (2/D(i).tauChip^2)*(D(i).tauChip-D(i).deltaB)*...
    (1+cos(2*omega_B*D(i).deltaB) + (D(i).tauChip-D(i).deltaC)*...
    omega_B*sin(2*omega_B*D(i).deltaB));%-0.1e10

for k = 1:NtauExtr

    deltaTauC(i,k) = tauIst - tauExtr(k);

    if sep_pull_in
        deltaTauB(i,k) = 1.9511e-06; % режим раздельного втягивания
    else
        deltaTauB(i,k) = deltaTauC(i,k);%1.9511e-06;%deltaTauC(i,k);
    end

    tauRep = tauIst - deltaTauB(i,k);%tauRep = tauIst -
        deltaTauC(i,k);

    p_promt(i,k) = ro(deltaTauC(i,k), 0, D(i).n, D(i).tauChip);
    p_late(i,k)  = ro(deltaTauC(i,k) - D(i).deltaC, 0, D(i).n,
        D(i).tauChip);
    p_early(i,k) = ro(deltaTauC(i,k) + D(i).deltaC, 0, D(i).n,
        D(i).tauChip);

    Dp = stdn_IQ^2;
    Dpe = ro(D(i).deltaC, 0, D(i).n, D(i).tauChip)*stdn_IQ^2;
    Del = ro(D(i).deltaC*2, 0, D(i).n, D(i).tauChip)*stdn_IQ^2;

    L=chol([Dp  Dpe Dpe;
        Dpe Dp  Del;
        Dpe Del Dp])';

    for j = 1:Np
        nI1 = L * randn(3,1);
        nQ1 = L * randn(3,1);
        nI2 = L * randn(3,1);
```

```
nQ2 = L * randn(3,1);

mIp1 = -(2/pi)*A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
    /pi)*sin(deltaW*Tc/2+deltaPhi + omega_B*deltaTauB(i,k));
mIp2 =  (2/pi)*A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
    /pi)*sin(deltaW*Tc/2+deltaPhi - omega_B*deltaTauB(i,k));
mQp1 = -(2/pi)*A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
    /pi)*cos(deltaW*Tc/2+deltaPhi + omega_B*deltaTauB(i,k));
mQp2 =  (2/pi)*A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
    /pi)*cos(deltaW*Tc/2+deltaPhi - omega_B*deltaTauB(i,k));

mIe1 = -(2/pi)*A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
    /pi)*sin(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
mIe2 =  (2/pi)*A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
    /pi)*sin(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

mIl1 = -(2/pi)*A_IQ*p_late(i,k) *sinc(deltaW*Tc/2
    /pi)*sin(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
mIl2 =  (2/pi)*A_IQ*p_late(i,k) *sinc(deltaW*Tc/2
    /pi)*sin(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

mQe1 = -(2/pi)*A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
    /pi)*cos(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
mQe2 =  (2/pi)*A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
    /pi)*cos(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

mQl1 = -(2/pi)*A_IQ*p_late(i,k) *sinc(deltaW*Tc/2
    /pi)*cos(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
mQl2 =  (2/pi)*A_IQ*p_late(i,k) *sinc(deltaW*Tc/2
    /pi)*cos(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

Ip1 = mIp1 + D(i).noise*nI1(1,1);
Ip2 = mIp2 + D(i).noise*nI2(1,1);
Ie1 = mIe1 + D(i).noise*nI1(2,1);
Ie2 = mIe2 + D(i).noise*nI2(2,1);
Il1 = mIl1 + D(i).noise*nI1(3,1);
Il2 = mIl2 + D(i).noise*nI2(3,1);

Qp1 = mQp1 + D(i).noise*nQ1(1,1);
Qp2 = mQp2 + D(i).noise*nQ2(1,1);
```

```
        Qe1 = mQe1 + D(i).noise*nQ1(2,1);
        Qe2 = mQe2 + D(i).noise*nQ2(2,1);
        Ql1 = mQl1 + D(i).noise*nQ1(3,1);
        Ql2 = mQl2 + D(i).noise*nQ2(3,1);

        shL = omega_B*(tauRep + D(i).deltaB);
        shE = omega_B*(tauRep - D(i).deltaB);

        Ie = (-sin(shE)*(Ie1+Ie2) - cos(shE)*(Qe1-Qe2))/2;
        Il = (-sin(shL)*(Il1+Il2) - cos(shL)*(Ql1-Ql2))/2;

        Qe = (cos(shE)*(Ie1-Ie2) - sin(shE)*(Qe1+Qe2))/2;
        Ql = (cos(shL)*(Il1-Il2) - sin(shL)*(Ql1+Ql2))/2;

        Ip = (Qp2 - Qp1)/2; % * sqrt(2);
        Qp = (-Ip2 + Ip1)/2; % * sqrt(2);

        Ud(i,k) = Ud(i,k) - ((Ie)^2 + (Qe)^2) + ((Il)^2 + (Ql)^2);

        sqrt_P(i,k) = sqrt(Ip^2 + Qp^2);
        sqrt_E(i,k) = sqrt(Ie^2 + Qe^2);

    end
    Ud(i,k) = Ud(i,k)/Np;
    Udteor(i,k) = ((2/pi)*A_IQ)^2 *(sinc(deltaW*Tc/2
      /pi)^2)*...%(2*((2*stdn_IQ * sqrt(2 * qcno * Tc))/pi)^2)
        (-(p_early(i,k))^2*(cos(omega_B*(deltaTauB(i,k) +
          D(i).deltaB)))^2 +...
        +(p_late(i,k) )^2*(cos(omega_B*(deltaTauB(i,k) -
          D(i).deltaB)))^2);
    if ~mod(k,100)
        fprintf('Progress: %.2f %%\n', k*100/NtauExtr)
    end
end
```

ro.m

```
function r=ro(x, m, n, tauChip)
    if m == 0
        r = (abs(x) < tauChip).*(1 - abs(x)/tauChip);
    else
```

```
tc = tauChip;
tau = x;
p = m / n ; v = ceil (2* p * abs ( tau ) / tc ) ;
r_boc = ( ( -1) .^( v +1) .* (1./ p .* ( - v .^2 + 2* v .* p + v - p
    ) - abs ( tau ) / tc .* (4* p - 2* v +1) ) ) .* ( abs ( tau ) < tc
    ) ;
r = r_boc;
    end
end
```

findZeroACF.m

```
function tauZero = findZeroACF(m, n, tauChip)
fs = 1/tauChip;

if m == 0
    tauZero = tauChip;
elseif m == n || m == 2*n
    Ts = 1/(2*(m/n)*fs);
    k = tauChip/Ts;
    j = -2*k + 2:2*k - 2;
    tau_b = (j*Ts)/2;
    for i = 1:length(j)
        if mod(j(i), 2) == 0
            % even
            A(i) = (-1).^(j(i)/2) * (k - abs(j(i)/2))/k;
        else
            % odd
            A(i) = ((-1).^((abs(j(i)) - 1)/2))/(2*k);
        end
    end
    indA_min = A == min(A);
    zero_x = -max(A)* (tau_b(indA_min)/(min(A) - max(A)));
    tauZero = zero_x(end);
else
    xx = 0:(tauChip/1000):tauChip;
    for i = 1:length(xx)
        if (ro(xx(i), m, n, tauChip) < 0)
            break;
        end
        tauZero = xx(i);
```

```
        end
    end
    end
```

Приложение Б

Листинг программы проверки флуктуационных характеристик дискриминатора задержки

fluct.m

```
close all; clear; clc

tauChip = 1e-3/1023; % Длительность чипа

lightC = physconst('LightSpeed');

DCB = 1/99.375e6;

default = {'m', 15,  'n',  10, 'split', true, 'noise', false,...
    'deltaC', tauChip/5, 'deltaB', tauChip/5,...
    'analit', false, 'newcolor', true, 'solid', true, 'norm', true};
DD = ...
    {
    {'m', 1,  'n',  1, 'split', false,   'noise', true,...
    'deltaC', 10*DCB, 'deltaB', 10*DCB,...
    'analit', false, 'newcolor', false, 'solid', false},...
    {'m', 1,  'n',  1, 'split', true,   'noise', true,...
    'deltaC',  10*DCB, 'deltaB', 10*DCB,...
    'analit', false, 'newcolor', false, 'solid', false}%,...
%     {'m', 5,  'n',  2.5, 'split', false,   'noise', true,...
%     'deltaC',  3*DCB 'deltaB', 3*DCB,...
%     'analit', false, 'newcolor', false, 'solid', false},...
%     {'m', 5,  'n',  2.5, 'split', true,   'noise', true,...
%     'deltaC',  3*DCB, 'deltaB', 3*DCB,...
%     'analit', false, 'newcolor', false, 'solid', false}%, ...
%     {'m', 15,  'n',  10, 'split', false,   'noise', true,...
%     'deltaC', 1*DCB, 'deltaB', 1*DCB,...
%     'analit', false, 'newcolor', false, 'solid', false},...
%     {'m', 15,  'n',  10, 'split', true,   'noise', true,...
%     'deltaC',  1*DCB, 'deltaB', 1*DCB}
    };

for i = 1:length(DD)
    S = struct(DD{1,i}{:});
```

```
        f = fieldnames(S);
        D(i) = struct(default{:});
        for j = 1:size(f,1)
            D(i).(f{j}) = S.(f{j});
        end
    end

    for i = 1:length(D)
        D(i).tauChip = tauChip/D(i).n;
    %       D(i).deltaC =  D(i).deltaC/D(i).n;
    %       D(i).deltaB =  D(i).deltaB/D(i).n;
    end

    Np = 5000;
    Tc = 5e-3; % Период интегрирования в корреляторе
    LightC = 3e8;

    deltaPhi = 0;%1*rand(1,1)*2*pi;
    deltaW = 0;%1*10*2*pi;
    tauIst = D(i).tauChip*rand(1,1);

    fs = 1.023e6;

    qcno_dB = 15:1:45;
    qcno = 10.^(qcno_dB/10);
    stdn_IQ = 13; % СКО шума квадратурных сумм
    A_IQ = stdn_IQ * sqrt(2 * qcno * Tc);

    deltatauC = nan(1,length(D));
    deltatauB = nan(1,length(D));

    Du_sim  = zeros(length(D), length(qcno));
    Du_teor = nan(length(D), length(qcno));

    lege = {};
    for i = 1:length(D)
        fprintf('Du for boc(%d,%d), split %d, analit %d\n',...
            D(i).m, D(i).n, D(i).split, D(i).analit);

        omega_B = 2*pi*D(i).m*fs;
```

```
    if D(i).split == 1
        eval('Du_split')
        if D(i).m ==0
            spec = ['BPSK(' num2str(D(i).n) '), \Delta^{C}='
               sprintf('%.f', D(i).deltaC*lightC) ' м, split'];
        else
            spec = ['BOC(' num2str(D(i).m) ',' num2str(D(i).n) '),
               \Delta^{C}=' sprintf('%.f', D(i).deltaC*lightC) ' м,
               split'];
        end
    else
        eval('Du_nelp')
        if D(i).m ==0
            spec = ['BPSK(' num2str(D(i).n) '), \Delta^{C}='
               sprintf('%.f', D(i).deltaC*lightC) ' м, direct'];
        else
            spec = ['BOC(' num2str(D(i).m) ',' num2str(D(i).n) '),
               \Delta^{C}=' sprintf('%.f', D(i).deltaC*lightC) ' м,
               direct'];
        end
    end

    lege{length(lege)+1} = [spec ': модел.'];
    lege{length(lege)+1} = [spec ': аналит.'];
%     lege{length(lege)+1} = [spec ': srns'];
end

set(0,'DefaultAxesFontSize', 14)

figure(1)
for i = 1:length(D)
    li = plot(qcno_dB, sqrt(Du_sim(i,:)),'LineWidth',1);  hold on;
    plot(qcno_dB, sqrt(Du_teor(i,:)), '-.', 'Color',
       li.Color,'LineWidth', 1)
%     plot(qcno_dB, sqrt(Du_teor_srns(i,:)), '--')
end
legend(lege);
xlabel('q_{c/n0}, dBHz', 'FontSize', 14)
ylabel('RMS ud noise', 'FontSize', 14);
```

```
grid on


figure(2)
for i = 1:length(D)

    li = plot(qcno_dB,
      LightC*sqrt(Du_sim(i,:)./SdTeor(i,:),'LineWidth', 1); hold
      on;
    plot(qcno_dB,  LightC*sqrt(Du_teor(i,:))./SdTeor(i,:), '-.',
      'Color', li.Color,'LineWidth',1); hold on;
%     plot(qcno_dB,  LightC*sqrt(Du_teor_srns(i,:))./SdTeor(i,:),
  '--'); hold on;
%     plot(qcno_dB,  LightC*sqrt(Du_norm(i,:)), '--'); hold on; % дис
  персия эквивалентных наблюдений

end
legend(lege);
xlabel('q_{c/n0}, дБГц', 'FontSize', 14)
ylabel('СКО \delta\tau, м', 'FontSize', 14);
grid on
```

Du_nelp.m

```
Dp = stdn_IQ^2;
Dpe = ro(D(i).deltaC, D(i).m, D(i).n, D(i).tauChip)*stdn_IQ^2;
Del = ro(D(i).deltaC*2, D(i).m, D(i).n, D(i).tauChip)*stdn_IQ^2;

L=chol([Dp  Dpe Dpe;
    Dpe Dp  Del;
    Dpe Del Dp])';

for q = 1:length(qcno)
    deltatauC(i) = 0;
    deltatauB(i) = deltatauC(i);

    p_promt = ro(deltatauC(i), D(i).m, D(i).n, D(i).tauChip);
    p_late  = ro(deltatauC(i) - D(i).deltaC, D(i).m, D(i).n,
      D(i).tauChip);
    p_early = ro(deltatauC(i) + D(i).deltaC, D(i).m, D(i).n,
      D(i).tauChip);
```

```
for j = 1:Np
    nI = L * randn(3,1); %
    nQ = L * randn(3,1);

    mIp = A_IQ(q)*p_promt*sinc(deltaW*Tc/2 /pi)*cos(deltaW*Tc/2 +
        deltaPhi);
    mIe = A_IQ(q)*p_early*sinc(deltaW*Tc/2 /pi)*cos(deltaW*Tc/2 +
        deltaPhi);
    mIl = A_IQ(q)*p_late *sinc(deltaW*Tc/2 /pi)*cos(deltaW*Tc/2 +
        deltaPhi);

    mQp = -A_IQ(q)*p_promt*sinc(deltaW*Tc/2 /pi)*sin(deltaW*Tc/2
        + deltaPhi);
    mQe = -A_IQ(q)*p_early*sinc(deltaW*Tc/2 /pi)*sin(deltaW*Tc/2
        + deltaPhi);
    mQl = -A_IQ(q)*p_late *sinc(deltaW*Tc/2 /pi)*sin(deltaW*Tc/2
        + deltaPhi);

    Ip = mIp + D(i).noise*nI(1,1);
    Ie = mIe + D(i).noise*nI(2,1);
    Il = mIl + D(i).noise*nI(3,1);
    Qp = mQp + D(i).noise*nQ(1,1);
    Qe = mQe + D(i).noise*nQ(2,1);
    Ql = mQl + D(i).noise*nQ(3,1);

    udtau = -(Ie^2+Qe^2) + (Il^2+Ql^2);
    Du_sim(i,q) = Du_sim(i,q) + udtau^2;

end
Du_sim(i,q) = Du_sim(i,q) / Np;

r1 = ro(D(i).deltaC, D(i).m, D(i).n, D(i).tauChip);
r2 = ro(2*D(i).deltaC, D(i).m, D(i).n, D(i).tauChip);
Du_teor(i,q) = (1 - r2) * 16 * qcno(q) * Tc * stdn_IQ^4 * (r1^2 +
    ((1 + r2) / (2 * qcno(q) * Tc)));

tauZero(i) = findZeroACF(D(i).m, D(i).n, D(i).tauChip);
```

```
    Du_norm(i,q) = ((1 - r2) * (r1^2 + ((1 + r2) / (2 * qcno(q) *
        Tc))))/...
        (4*qcno(q)*Tc*(1/tauZero(i) -
            1*(D(i).deltaC/tauZero(i)^2))^2);
end

SdTeor(i,:) = 8*qcno*Tc*stdn_IQ^2*sinc(deltaW*Tc/2 /pi)^2*...
                (1/tauZero(i) - 1*(D(i).deltaC/tauZero(i)^2));

% Du_norm(i,:) = Du_teor(i,:)./SdTeor(i,:).^2;
```

Du_split.m

```
Dp = stdn_IQ^2;
Dpe = ro(D(i).deltaC, 0, D(i).n, D(i).tauChip)*stdn_IQ^2;
Del = ro(D(i).deltaC*2, 0, D(i).n, D(i).tauChip)*stdn_IQ^2;

L=chol([Dp  Dpe Dpe;
    Dpe Dp  Del;
    Dpe Del Dp])';

for q = 1:length(qcno)
    deltatauC(i) = 0;
    deltatauB(i) = deltatauC(i);%1.9511e-06%deltatauC(i)
    tauRep = tauIst - deltatauC(i);

    p_promt = ro(deltatauC(i), 0, D(i).n, D(i).tauChip);
    p_late  = ro(deltatauC(i) - D(i).deltaC, 0, D(i).n, D(i).tauChip);
    p_early = ro(deltatauC(i) + D(i).deltaC, 0, D(i).n, D(i).tauChip);

    for j = 1:Np
        nI1 = L * randn(3,1);
        nQ1 = L * randn(3,1);
        nI2 = L * randn(3,1);
        nQ2 = L * randn(3,1);

        mIe1 = -(2/pi)*A_IQ(q)*p_early*sinc(deltaW*Tc/2
            /pi)*sin(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
        mIe2 =  (2/pi)*A_IQ(q)*p_early*sinc(deltaW*Tc/2
            /pi)*sin(deltaW*Tc/2+deltaPhi - omega_B*tauIst);
```

```
    mIl1 = -(2/pi)*A_IQ(q)*p_late *sinc(deltaW*Tc/2
        /pi)*sin(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
    mIl2 =  (2/pi)*A_IQ(q)*p_late *sinc(deltaW*Tc/2
        /pi)*sin(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

    mQe1 = -(2/pi)*A_IQ(q)*p_early*sinc(deltaW*Tc/2
        /pi)*cos(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
    mQe2 =  (2/pi)*A_IQ(q)*p_early*sinc(deltaW*Tc/2
        /pi)*cos(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

    mQl1 = -(2/pi)*A_IQ(q)*p_late *sinc(deltaW*Tc/2
        /pi)*cos(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
    mQl2 =  (2/pi)*A_IQ(q)*p_late *sinc(deltaW*Tc/2
        /pi)*cos(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

    Ie1 = mIe1 + D(i).noise*nI1(2,1);
    Ie2 = mIe2 + D(i).noise*nI2(2,1);
    Il1 = mIl1 + D(i).noise*nI1(3,1);
    Il2 = mIl2 + D(i).noise*nI2(3,1);

Qe1 = mQe1 + D(i).noise*nQ1(2,1);
    Qe2 = mQe2 + D(i).noise*nQ2(2,1);
    Ql1 = mQl1 + D(i).noise*nQ1(3,1);
    Ql2 = mQl2 + D(i).noise*nQ2(3,1);

    shE = omega_B*(tauRep - D(i).deltaB);
    shL = omega_B*(tauRep + D(i).deltaB);

    Ie = (-sin(shE)*(Ie1+Ie2) - cos(shE)*(Qe1-Qe2))/2;
    Il = (-sin(shL)*(Il1+Il2) - cos(shL)*(Ql1-Ql2))/2;

    Qe = (cos(shE)*(Ie1-Ie2) - sin(shE)*(Qe1+Qe2))/2;
    Ql = (cos(shL)*(Il1-Il2) - sin(shL)*(Ql1+Ql2))/2;

    udtau = -(Ie^2+Qe^2) + (Il^2+Ql^2);
    Du_sim(i,q) = Du_sim(i,q) + udtau^2;

end
Du_sim(i,q) = Du_sim(i,q) / Np;
```

```
        r1 = ro(D(i).deltaC,    0, D(i).n, D(i).tauChip) * cos(omega_B *
          D(i).deltaB);
        r2 = ro(2*D(i).deltaC, 0, D(i).n, D(i).tauChip) * cos(omega_B *
          2*D(i).deltaB);
        Du_teor(i,q) = (1 - r2) * 4 * qcno(q) * Tc * stdn_IQ^4 *
          (2*(2/pi)^2 * r1^2 + (1 + r2) / (2 * qcno(q) * Tc));

        Du_norm(i,q) = ((1 - r2) * (2*(2/pi)^2 * r1^2 + (1 + r2) / (2 *
          qcno(q) * Tc)))/...
           ((4 * (2/pi)^4 * qcno(q) * Tc  *
              ((tauChip-D(i).deltaB)/tauChip^2)^2*...
           (1 + cos(2 * omega_B * D(i).deltaB) + omega_B * (tauChip -
              D(i).deltaB) * sin(2 * omega_B * D(i).deltaB))^2));

    end

    tauZero(i) = findZeroACF(D(i).m, D(i).n, D(i).tauChip);

    SdTeor(i,:) = ((2/pi)*A_IQ).^2 *(sinc(deltaW*Tc/2 /pi)^2)*...
        (2/D(i).tauChip^2)*(D(i).tauChip-D(i).deltaB)*...
        (1+cos(2*omega_B*D(i).deltaB) + (D(i).tauChip-D(i).deltaC)*...
        omega_B*sin(2*omega_B*D(i).deltaB));
```

Приложение В

Листинг программы экспериментального определения характеристик слежения за задержкой сигнала

dll_split_stat.m

```
close all; clear; clc

tauChip = 1e-3/1023;
lightC = physconst('LightSpeed');

DCB = 1/99.375e6;

default = {'m', 15,  'n',  10, 'split', true, 'noise', false,...
    'deltaC', tauChip/5, 'deltaB', tauChip/5,...
    'analit', false, 'newcolor', true, 'solid', true, 'norm', true};
DD = ...
    {
     {'m', 1,  'n',  1, 'split', false,   'noise', true,...
     'deltaC', 10*DCB, 'deltaB', 10*DCB,...
     'analit', false, 'newcolor', true, 'solid', false},...
     {'m', 1,  'n',  1, 'split', true,   'noise', true,...
     'deltaC',  10*DCB, 'deltaB', 10*DCB,...
     'analit', false, 'newcolor', true, 'solid', false},...
     {'m', 1,  'n',  1, 'split', false,   'noise', false,...
     'deltaC', 10*DCB, 'deltaB', 10*DCB,...
     'analit', true, 'newcolor', true, 'solid', true},...
     {'m', 1,  'n',  1, 'split', true,   'noise', false,...
     'deltaC',  10*DCB, 'deltaB', 10*DCB,...
     'analit', true, 'newcolor', true, 'solid', true}%,...
%      {'m', 5,  'n',  2.5, 'split', false,   'noise', true,...
%      'deltaC',  3*DCB 'deltaB', 3*DCB,...
%      'analit', false, 'newcolor', true, 'solid', false},...
%      {'m', 5,  'n',  2.5, 'split', true,   'noise', true,...
%      'deltaC',  3*DCB, 'deltaB', 3*DCB,...
%      'analit', false, 'newcolor', true, 'solid', false}, ...
%      {'m', 5,  'n',  2.5, 'split', false,   'noise', false,...
%      'deltaC',  3*DCB 'deltaB', 3*DCB,...
%      'analit', true, 'newcolor', true, 'solid', true},...
```

```
%     {'m', 5,  'n',  2.5, 'split', true,   'noise', false,...
%     'deltaC',  3*DCB, 'deltaB', 3*DCB,...
%     'analit', true, 'newcolor', true, 'solid', true}%, ...
%     {'m', 15,  'n',  10, 'split', false,   'noise', true,...
%     'deltaC', 1*DCB, 'deltaB', 1*DCB,...
%     'analit', false, 'newcolor', true, 'solid', false},...
%     {'m', 15,  'n',  10, 'split', true,   'noise', true,...
%     'deltaC',  1*DCB, 'deltaB', 1*DCB,...
%     'analit', false, 'newcolor', true, 'solid', false},...
%     {'m', 15,  'n',  10, 'split', false,   'noise', false,...
%     'deltaC', 1*DCB, 'deltaB', 1*DCB,...
%     'analit', true, 'newcolor', true, 'solid', true},...
%     {'m', 15,  'n',  10, 'split', true,   'noise', false,...
%     'deltaC',  1*DCB, 'deltaB', 1*DCB,...
%     'analit', true, 'newcolor', true, 'solid', true},...
    };

for i = 1:length(DD)
    S = struct(DD{1,i}{:});
    f = fieldnames(S);
    D(i) = struct(default{:});
    for j = 1:size(f,1)
        D(i).(f{j}) = S.(f{j});
    end
end

for i = 1:length(D)
    D(i).tauChip = tauChip/D(i).n;
%     D(i).deltaC =  D(i).deltaC/D(i).n;
%     D(i).deltaB =  D(i).deltaB/D(i).n;
end

sep_pull_in = 0; % режим раздельного втягивания

Np = 1000;
Tc = 5e-3;
qcno_dB = 35;
stdn_IQ = 13;
qcno = 10^(qcno_dB/10);
A_IQ = stdn_IQ * sqrt(2 * qcno * Tc);
```

```
LightC = 3e8;
tauIst = tauChip/5;
tauExtr= tauIst-1*tauChip:1*tauChip/1000:tauIst+1*tauChip;
NtauExtr = length(tauExtr);

deltaPhi = 0;% 1*rand(1,1)*2*pi;
deltaW = 0;% 1*10*2*pi;

Ud = zeros(length(D),NtauExtr);
Udteor = zeros(length(D),NtauExtr);

p_promt = nan(length(D),NtauExtr);
p_early = nan(length(D),NtauExtr);
p_late  = nan(length(D),NtauExtr);
deltaTauC = nan(length(D),NtauExtr);
deltaTauB = nan(length(D),NtauExtr);
sqrt_P = nan(length(D),NtauExtr);
sqrt_E = nan(length(D),NtauExtr);
SdTeor = nan(1,length(D));
tauZero = nan(1,length(D));

for i = 1:length(D)
    fprintf('boc(%d,%d), split %d, analit %d\n', D(i).m, D(i).n,
      D(i).split, D(i).analit);

    omega_B = 2*pi*D(i).m*1.023e6;

    if D(i).split == 1
        eval('Ud_split')
    else
        eval('Ud_nelp')
    end
end

set(0,'DefaultAxesFontSize', 14)

newcolors = [0.00,0.45,0.74    % blue
    0.85,0.33,0.10    % red
    0.93,0.69,0.13    % yellow
    0.49,0.18,0.56    % purple
```

```
    0.47,0.67,0.19    % green
    0.30,0.75,0.93    % light blue
    0,    0,    0]; % black
linestyle = {'--', '-.', ':', '-'}';

figure(1)
   % axis for dtau/tauchip
   b=axes('Position',[.1 .1 .8 1e-12], 'FontSize', 14);
   set(b,'Units','normalized');
   a = axes('Position',[.1 .2 .8 .7], 'FontSize', 14);
   set(a,'Units','normalized');

for i = 1:length(D)

    if D(i).solid == 0
        set(gca, 'LineStyleOrder', linestyle(1));
    end

    plot(a, deltaTauC(i,:)*lightC,
      sqrt_P(i,:)./max(sqrt_P(i,:)),'LineWidth',1); hold on;
    xlabel(a,'\delta\tau, м', 'FontSize', 14)%/\tau_{c}
    xlabel(b,'\delta\tau/\tau_{c}', 'FontSize', 14)%/\tau_{c}
    ylabel('\rho(\delta\tau)', 'FontSize', 14)

    if D(i).m == 0
        str_acf{i} = ['BPSK(',num2str(D(i).n), ')  split =  ',...
            num2str(D(i).split)];
    else
        str_acf{i} = ['BOC(',num2str(D(i).m), ',', num2str(D(i).n),...
            ')  split =  ',num2str(D(i).split)];
    end

    legend(str_acf)
    grid on
    set(gca,'LineStyleOrder','remove')
    set(gca,'ColorOrder','remove')
end

if D(i).m == 15
    kk = 0.1;
```

```
    else
        kk =1;
    end
    set(a,'xlim',[(-D(i).tauChip - kk*50/lightC)*lightC (D(i).tauChip +
       kk*50/lightC)*lightC]);
    set(b,'xlim',[(-D(i).tauChip - kk*50/lightC)/D(i).tauChip
       (D(i).tauChip + kk*50/lightC)/D(i).tauChip]); hold on
        plot(([-D(i).deltaC -D(i).deltaC])*lightC, [0 1], '-k',...
            ([D(i).deltaC D(i).deltaC])*lightC, [0 1], '-k')

    figure(2)
    for i = 1:length(D)

        if D(i).solid == 0
            set(gca, 'LineStyleOrder', linestyle(1));
        end

        if D(i).analit == 1
            if D(i).solid == 1
                plot(deltaTauC(i,:)*lightC, Udteor(i,:),'LineWidth',1);
                    hold on;
            else
                plot(deltaTauC(i,:)*lightC, Udteor(i,:),
                    '-.','LineWidth',1); hold on;
            end
        else
            plot(deltaTauC(i,:)*lightC, Ud(i,:),'LineWidth',1); hold on;
        end

        if D(i).split == 0 && D(i).m == 0
            str_dh{i} = ['BPSK(',num2str(D(i).n), ') \Delta ',...
                num2str(D(i).deltaC*lightC) ,...
                ', analit = ' num2str(D(i).analit)];

        elseif D(i).split == 0 && D(i).m ~= 0 %&& D(i).analit == 0
            str_dh{i} = ['BOC(',num2str(D(i).m), ',', num2str(D(i).n) ,...
                ') \Delta ',num2str(D(i).deltaC*lightC) ,...
                ', analit = ' num2str(D(i).analit)];
        else
            str_dh{i} = ['BOC(',num2str(D(i).m), ',', num2str(D(i).n) ,...
```

```
            ')  \Delta ',num2str(D(i).deltaC*lightC),...
            '/', num2str(D(i).deltaB*lightC),...
            ', analit = ' num2str(D(i).analit)];
    end

    set(gca,'LineStyleOrder','remove')
    set(gca,'ColorOrder','remove')
    end
xlabel('\delta\tau m', 'FontSize', 16)
ylabel('U_{dll}', 'FontSize', 16)
ylim([min(min(Ud))-10 max(max(Ud))+10])
grid on
str = str_dh;
legend(str)
n = 1;
for i = 1:length(D)

    if D(i).analit == 1 || D(i).analit == 0

        plot(deltaTauC(i,:)*lightC, SdTeor(i).*deltaTauC(i,:), '--',
           'Color', newcolors(i,:),'LineWidth',1); hold on;

        if D(i).m == 0
            str_Sd{n} = ['S_{d} ', 'BPSK(',num2str(D(i).n),...
                ') split = ',num2str(D(i).split)];
        else
            str_Sd{n} = ['S_{d} ', 'BOC(',num2str(D(i).m),...
                ',', num2str(D(i).n), ') split =
                  ',num2str(D(i).split)];
        end
    else
        continue
    end
    n = n + 1;
end
if D(i).analit == 1
    str = [str str_Sd];
end
legend(str)
```

```
figure(3)
for i = 1:length(D)

    if D(i).newcolor == 0
        set(gca, 'ColorOrder', newcolors(length(newcolors),:));
    else
        set(gca, 'ColorOrder', newcolors(i,:));
    end
    if D(i).solid == 0
        set(gca, 'LineStyleOrder', linestyle(1));
    end

    if D(i).analit == 1 % && D(i).split == 1
        if D(i).solid == 1
            plot(deltaTauC(i,:)*lightC,
               Udteor(i,:)./SdTeor(i),'LineWidth',1); hold on;
        else
            plot(deltaTauC(i,:)*lightC, Udteor(i,:)./SdTeor(i),
               '-.','LineWidth',1); hold on;
        end
    else
        plot(deltaTauC(i,:)*lightC,
           Ud(i,:)./SdTeor(i),'LineWidth',1); hold on;
    end

    if D(i).split == 0 && D(i).m == 0
        str2_norm{i} = ['BPSK(',num2str(D(i).n), ')  \Delta ',...
            num2str(D(i).deltaC*lightC),...
            ', analit = ' num2str(D(i).analit)];

    elseif  D(i).split == 0 && D(i).m ~= 0 %&& D(i).analit == 0
        str2_norm{i} = ['BOC(',num2str(D(i).m), ',',
           num2str(D(i).n),...
            ')  \Delta ',num2str(D(i).deltaC*lightC),...
            ', analit = ' num2str(D(i).analit)];
    else
        str2_norm{i} = ['BOC(',num2str(D(i).m), ',',
           num2str(D(i).n),...
            ')  \Delta ',num2str(D(i).deltaC*lightC),...
            '/', num2str(D(i).deltaB*lightC),...
```

```
                ', analit = ' num2str(D(i).analit)];
        end

        set(gca,'LineStyleOrder','remove')
        set(gca,'ColorOrder','remove')
    end
    m = 1;
    for i = 1:length(str2_norm)
        if isempty(str2_norm{i}) ~= 1
            str_norm{m} = str2_norm{i};
            m = m + 1;
        end
    end

    xlabel('\delta\tau m', 'FontSize', 14)
    ylabel('U_{dll}/ S_{d}', 'FontSize', 14)
    % xlim([-1.5 1.5])
    ylim([min(min(Udteor/ max(SdTeor)))-1e-8 ...
        max(max(Udteor/ max(SdTeor)))+1e-8])
    grid on
    legend(str_norm)
    title('Нормированная дискриминационная характеристика')

    logfile = ['ACF_BOC' num2str(D(1).m) '_' num2str(D(1).n) '.mat'];
    save(logfile, 'deltaTauC', 'sqrt_P');
```

Ud_nelp.m

```
tauZero(i) = findZeroACF(D(i).m, D(i).n, D(i).tauChip);

SdTeor(i) = 8*qcno*Tc*stdn_IQ^2*sinc(deltaW*Tc/2 /pi)^2*...
    (1/tauZero(i) - (D(i).deltaC/tauZero(i)^2));

for k = 1:NtauExtr

    deltaTauC(i,k) = tauIst - tauExtr(k);
    deltaTauB(i,k) = deltaTauC(i,k);

    p_promt(i,k) = ro(deltaTauC(i,k), D(i).m, D(i).n, D(i).tauChip);
```

```
p_late(i,k)   = ro(deltaTauC(i,k) - D(i).deltaC, D(i).m, D(i).n,
    D(i).tauChip);
p_early(i,k) = ro(deltaTauC(i,k) + D(i).deltaC, D(i).m, D(i).n,
    D(i).tauChip);

Dp = stdn_IQ^2;
Dpe = ro(D(i).deltaC, D(i).m, D(i).n, D(i).tauChip)*stdn_IQ^2;
Del = ro(D(i).deltaC*2, D(i).m, D(i).n, D(i).tauChip)*stdn_IQ^2;

L=chol([Dp  Dpe Dpe;
      Dpe Dp  Del;
      Dpe Del Dp])';

for j = 1:Np
      nI = L * randn(3,1);
      nQ = L * randn(3,1);
      mIp = A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2 /pi)*cos(deltaW*Tc/2
          + deltaPhi);
      mIe = A_IQ*p_early(i,k)*sinc(deltaW*Tc/2 /pi)*cos(deltaW*Tc/2
          + deltaPhi);
      mIl = A_IQ*p_late(i,k) *sinc(deltaW*Tc/2 /pi)*cos(deltaW*Tc/2
          + deltaPhi);

      mQp = -A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
          /pi)*sin(deltaW*Tc/2 + deltaPhi);
      mQe = -A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
          /pi)*sin(deltaW*Tc/2 + deltaPhi);
      mQl = -A_IQ*p_late(i,k) *sinc(deltaW*Tc/2
          /pi)*sin(deltaW*Tc/2 + deltaPhi);

      Ip = mIp + D(i).noise*nI(1,1);
      Ie = mIe + D(i).noise*nI(2,1);
      Il = mIl + D(i).noise*nI(3,1);
      Qp = mQp + D(i).noise*nQ(1,1);
      Qe = mQe + D(i).noise*nQ(2,1);
      Ql = mQl + D(i).noise*nQ(3,1);

      Ud(i,k) = Ud(i,k) - (Ie^2+Qe^2) + (Il^2+Ql^2);
```

```
        sqrt_P(i,k) = sqrt(Ip^2 + Qp^2);
        sqrt_E(i,k) = sqrt(Ie^2 + Qe^2);

    end
    Ud(i,k) = Ud(i,k)/Np;


    Udteor(i,k) = 2*qcno* stdn_IQ^2 *Tc*(sinc(deltaW*Tc/2
       /pi)^2)*(-p_early(i,k)^2 + p_late(i,k)^2);
    if ~mod(k,100)
        fprintf('Progress: %.2f %%\n', k*100/NtauExtr)
    end
end
```

Ud_split.m

```
if sep_pull_in
D(i).deltaB = 0;% режим раздельного втягивания
end

SdTeor(i) = ((2/pi)*A_IQ)^2 *(sinc(deltaW*Tc/2 /pi)^2)*... %sqrt(2)*
    (2/D(i).tauChip^2)*(D(i).tauChip-D(i).deltaB)*...
    (1+cos(2*omega_B*D(i).deltaB) + (D(i).tauChip-D(i).deltaC)*...
    omega_B*sin(2*omega_B*D(i).deltaB));%-0.1e10

for k = 1:NtauExtr

    deltaTauC(i,k) = tauIst - tauExtr(k);

    if sep_pull_in
        deltaTauB(i,k) = 1.9511e-06; % режим раздельного втягивания
    else
        deltaTauB(i,k) = deltaTauC(i,k);%1.9511e-06;%deltaTauC(i,k);
    end

    tauRep = tauIst -  deltaTauB(i,k);%tauRep = tauIst -
       deltaTauC(i,k);

    p_promt(i,k) = ro(deltaTauC(i,k), 0, D(i).n, D(i).tauChip);
    p_late(i,k)  = ro(deltaTauC(i,k) - D(i).deltaC, 0, D(i).n,
       D(i).tauChip);
```

```
p_early(i,k) = ro(deltaTauC(i,k) + D(i).deltaC, 0, D(i).n,
    D(i).tauChip);

Dp = stdn_IQ^2;
Dpe = ro(D(i).deltaC, 0, D(i).n, D(i).tauChip)*stdn_IQ^2;
Del = ro(D(i).deltaC*2, 0, D(i).n, D(i).tauChip)*stdn_IQ^2;

L=chol([Dp  Dpe Dpe;
      Dpe Dp  Del;
      Dpe Del Dp])';

for j = 1:Np
      nI1 = L * randn(3,1);
      nQ1 = L * randn(3,1);
      nI2 = L * randn(3,1);
      nQ2 = L * randn(3,1);

      mIp1 = -(2/pi)*A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
          /pi)*sin(deltaW*Tc/2+deltaPhi + omega_B*deltaTauB(i,k));
      mIp2 =  (2/pi)*A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
          /pi)*sin(deltaW*Tc/2+deltaPhi - omega_B*deltaTauB(i,k));
      mQp1 = -(2/pi)*A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
          /pi)*cos(deltaW*Tc/2+deltaPhi + omega_B*deltaTauB(i,k));
      mQp2 =  (2/pi)*A_IQ*p_promt(i,k)*sinc(deltaW*Tc/2
          /pi)*cos(deltaW*Tc/2+deltaPhi - omega_B*deltaTauB(i,k));

      mIe1 = -(2/pi)*A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
          /pi)*sin(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
      mIe2 =  (2/pi)*A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
          /pi)*sin(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

      mIl1 = -(2/pi)*A_IQ*p_late(i,k) *sinc(deltaW*Tc/2
          /pi)*sin(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
      mIl2 =  (2/pi)*A_IQ*p_late(i,k) *sinc(deltaW*Tc/2
          /pi)*sin(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

      mQe1 = -(2/pi)*A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
          /pi)*cos(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
      mQe2 =  (2/pi)*A_IQ*p_early(i,k)*sinc(deltaW*Tc/2
          /pi)*cos(deltaW*Tc/2+deltaPhi - omega_B*tauIst);
```

```
mQl1 = -(2/pi)*A_IQ*p_late(i,k) *sinc(deltaW*Tc/2
    /pi)*cos(deltaW*Tc/2+deltaPhi + omega_B*tauIst);
mQl2 =  (2/pi)*A_IQ*p_late(i,k) *sinc(deltaW*Tc/2
    /pi)*cos(deltaW*Tc/2+deltaPhi - omega_B*tauIst);

Ip1 = mIp1 + D(i).noise*nI1(1,1);
Ip2 = mIp2 + D(i).noise*nI2(1,1);
Ie1 = mIe1 + D(i).noise*nI1(2,1);
Ie2 = mIe2 + D(i).noise*nI2(2,1);
Il1 = mIl1 + D(i).noise*nI1(3,1);
Il2 = mIl2 + D(i).noise*nI2(3,1);

Qp1 = mQp1 + D(i).noise*nQ1(1,1);
Qp2 = mQp2 + D(i).noise*nQ2(1,1);
Qe1 = mQe1 + D(i).noise*nQ1(2,1);
Qe2 = mQe2 + D(i).noise*nQ2(2,1);
Ql1 = mQl1 + D(i).noise*nQ1(3,1);
Ql2 = mQl2 + D(i).noise*nQ2(3,1);

shL = omega_B*(tauRep + D(i).deltaB);
shE = omega_B*(tauRep - D(i).deltaB);

Ie = (-sin(shE)*(Ie1+Ie2) - cos(shE)*(Qe1-Qe2))/2;
Il = (-sin(shL)*(Il1+Il2) - cos(shL)*(Ql1-Ql2))/2;

Qe = (cos(shE)*(Ie1-Ie2) - sin(shE)*(Qe1+Qe2))/2;
Ql = (cos(shL)*(Il1-Il2) - sin(shL)*(Ql1+Ql2))/2;

Ip = (Qp2 - Qp1)/2; % * sqrt(2);
Qp = (-Ip2 + Ip1)/2; % * sqrt(2);

Ud(i,k) = Ud(i,k) - ((Ie)^2 + (Qe)^2) + ((Il)^2 + (Ql)^2);

sqrt_P(i,k) = sqrt(Ip^2 + Qp^2);
sqrt_E(i,k) = sqrt(Ie^2 + Qe^2);
end
Ud(i,k) = Ud(i,k)/Np;
Udteor(i,k) = ((2/pi)*A_IQ)^2 *(sinc(deltaW*Tc/2
  /pi)^2)*...%(2*((2*stdn_IQ * sqrt(2 * qcno * Tc))/pi)^2)
```

```
            (-(p_early(i,k))^2*(cos(omega_B*(deltaTauB(i,k) +
               D(i).deltaB)))^2 +...
            +(p_late(i,k) )^2*(cos(omega_B*(deltaTauB(i,k) -
               D(i).deltaB)))^2);
        if ~mod(k,100)
            fprintf('Progress: %.2f %%\n', k*100/NtauExtr)
        end
end
```

ro.m

```
function r=ro(x, m, n, tauChip)
    if m == 0
        r = (abs(x) < tauChip).*(1 - abs(x)/tauChip);
    else
tc = tauChip;
tau = x;
p = m / n ; v = ceil (2* p * abs ( tau ) / tc ) ;
r_boc = ( ( -1) .^( v +1) .* (1./ p .* ( - v .^2 + 2* v .* p + v - p
   ) - abs ( tau ) / tc .* (4* p - 2* v +1) ) ) .* ( abs ( tau ) < tc
   );
r = r_boc;
    end
end
```

findZeroACF.m

```
function tauZero = findZeroACF(m, n, tauChip)
fs = 1/tauChip;

if m == 0
    tauZero = tauChip;
elseif m == n || m == 2*n
    Ts = 1/(2*(m/n)*fs);
    k = tauChip/Ts;
    j = -2*k + 2:2*k - 2;
    tau_b = (j*Ts)/2;

    for i = 1:length(j)
        if mod(j(i), 2) == 0
            % even
            A(i) = (-1).^(j(i)/2) * (k - abs(j(i)/2))/k;
```

```
        else
            % odd
            A(i) = ((-1).^((abs(j(i)) - 1)/2))/(2*k);
        end
    end
    indA_min = A == min(A);
    zero_x = -max(A)* (tau_b(indA_min)/(min(A) - max(A)));
    tauZero = zero_x(end);
else
    xx = 0:(tauChip/1000):tauChip;
    for i = 1:length(xx)
        if (ro(xx(i), m, n, tauChip) < 0)
            break;
        end
        tauZero = xx(i);
    end
end
end
```

Приложение Г

Листинг программы экспериментального определения характеристик захвата

pullin.m

```
clc; clear all; close all;
Light_C       = physconst('LightSpeed');
omega_0       = 2*pi*(1176.45e6 + 1207.14e6)/2;

m             = 1; % if m = 0 -> BPSK
n             = 1;

% m             = 5; % if m = 0 -> BPSK
% n             = 2.5;

% m             = 15; % if m = 0 -> BPSK
% n             = 10;

tauChip       = 1e-3/1023/n; % Длительность чипа
tauZero       = findZeroACF(m, n, tauChip);

initial_err   = -1*tauChip + 1*(tauChip+tauChip).*rand(1,1);
fs            = 1.023e6;
omega_B       = 2*pi*m*fs;
deltaW  = 0;
deltaPhi  = 0;
Tmod          = 20; % Время моделирования
T             = 5e-3; % Период работы фильтров (накопления в корреляторах
    )
N             = fix(Tmod/T);
Td            = 0.2e-6;
L             = T/Td;

DCB = 1/99.375e6; %клок клоникуса в с

D = 30; % BOC(1,1)  [m]3:1:90
% D = 9; % BOC(5,2.5)
% D = 3; % BOC(15,10)
hW = waitbar(0, 'q_{c/n0} progress', 'Name', 'Total model progress');
```

```
for d = 1:length(D)
    Delta = D(d)/Light_C;
    errThr = Delta/3; % Пороговое значение ошибки слежения, дальше считаем, что срыв. Каплан стр. 194.
    aiding_pll = 0;
    tf          = 0:T:Tmod-T;
    F = [1 T   0;
         0 1   T;
         0 0   1];
    G = [0 0 1]';
    F_dll = [1 T;
             0 1];

    BW_dll    = 3;

        K_dll = nan(2,1); % Вектор-столбец коэффициентов фильтра
        K_dll(2) = 2*16/9*BW_dll^2; % Коэффициенты непрерывной системы в установившемся режиме
        K_dll(1) = sqrt(2*K_dll(2));
        K_dll     = K_dll*T; % Переход к коэффициентам дискретной системы


    % % Расчет параметров формирующего шума. GLONASS, page 162
    alpha = 0.1; %Ширина спектра ускорения, с^-1
    std_a = 1; %СКЗ ускорения
    Sksi = 2*(33*std_a)^2 * alpha; %Спектральная плотность формирующего шума
    stdIst = 1*sqrt(Sksi * T); %СКО формирующего шума
    Dksi = stdIst^2; % Дисперсия формирующего шума

    % расчет формирующего шума для модели дрейфа частоты ОГ
    % (из статьи с срнс)
    N_ksi_nu = 1.1e-19;
    N_og = N_ksi_nu*(omega_0)^2;
    D_og = N_og/2/T;
    std_og = sqrt(D_og);

    N_exp = length(initial_err);
```

```
qcno_dB = [15:3:30 35 45 50];
qcno_dB = 35;

P_zakhvat = zeros(1, length(qcno_dB));
P_logj_zero = zeros(1, length(qcno_dB));
P_sryv = zeros(1, length(qcno_dB));

P_zakhvat_split = zeros(1, length(qcno_dB));
P_logj_zero_split = zeros(1, length(qcno_dB));
P_sryv_split = zeros(1, length(qcno_dB));

for q = 1:length(qcno_dB)

    qcno        = 10^(qcno_dB(q)/10);
    A_IQ        = (A*L)/2;
    stdn_IQ = 13; % СКО шума квадратурных сумм
    A_IQ = stdn_IQ * sqrt(2 * qcno * T);
    stdn = stdn_IQ*sqrt(2/L);

    for n_exp = 1:N_exp
        DeltaC      = Delta;
        DeltaB      = 0; % включаем режим раздельного втягивания
        deltaTauB   = initial_err(n_exp);%tauChip/10;%1.9511e-06;

        x_ist       =  nan(3,length(tf));
        x_ist(:,1)  = [0 0 0]';
        %   direct
        x_est_pll               = nan(3,length(tf));
        x_est_pll(:,1)          = [0 0 0]';
        %     % split
        x_est_pll_split         = nan(3,length(tf));
        x_est_pll_split(:,1)    = [0 0 0]';

        tau_est_dll      = nan(2,length(tf));
        tau_est_dll(:,1) = [initial_err(n_exp) 0].';

        tau_est_split_dll      = nan(2,length(tf)); %split
        tau_est_split_dll(:,1) = [initial_err(n_exp) 0].'; %split
```

```
phi_ist = nan(1,length(tf));
phi_ist(1) = x_ist(1,1);
tau_ist = nan(1,length(tf));
tau_ist(1) = -x_ist(1,1)/omega_0;

n_ksi = randn(1, length(tf))*stdIst;
D_pll = stdn^2 * sqrt(L/2);

n_og = randn(1, length(tf))*std_og;

% direct
r1 = ro(DeltaC, m, n, tauChip);
r2 = ro(2*DeltaC, m, n, tauChip);
D_dll = (1 - r2) * 16 * qcno * T * stdn_IQ^4 *...
    (r1^2 + ((1 + r2) / (2 * qcno * T)));
n_dll = randn(1, length(tf))*sqrt(D_dll);

% split
r1_split = ro(DeltaC,   0, n, tauChip) * cos(omega_B
  *DeltaB);
r2_split = ro(2*DeltaC, 0, n, tauChip) * cos(omega_B *
  2*DeltaB);
D_split_dll = (1 - r2_split) * 4 * qcno * T * stdn_IQ^4
  ...
   * (2*(2/pi)^2 * r1_split^2 + (1 + r2_split) / (2 *
      qcno * T));
n_split_dll = randn(1, length(tf))*sqrt(D_split_dll);

sep_pull_in_on  = 1; % режим раздельного втягивания включе
  н
flag1 = 0;
restart_flag = 0;
for i = 2:N
    if flag1 == 0
        if sep_pull_in_on == 0
            DeltaB = DeltaC;
            % split
            r1_split = ro(DeltaC,   0, n, tauChip) *
                cos(omega_B *DeltaB);
```

```
            r2_split = ro(2*DeltaC, 0, n, tauChip) *
               cos(omega_B * 2*DeltaB);
            D_split_dll = (1 - r2_split) * 4 * qcno * T *
              stdn_IQ^4 ...
                * (2*(2/pi)^2 * r1_split^2 + (1 +
                  r2_split) / (2 * qcno * T));
            n_split_dll = randn(1,
               length(tf))*sqrt(D_split_dll);
            flag1 = 1;
        end
    end

    x_ist(:,i)  = 1*x_ist(:,1) + 0*F*x_ist(:,i-1) + [0;
       T; 0]*n_og(i-1)*0 + 0*G*n_ksi(i-1);

    phi_ist(i)  = x_ist(1,i);
    omega_ist   = x_ist(2,i);

    tau_ist(i)  = -phi_ist(i)/omega_0;

    tau_extr_dll       = F_dll*tau_est_dll(:,i-1);
    tau_extr_split_dll = F_dll*tau_est_split_dll(:,i-1);
          % split

    deltaTau        = tau_ist(i) - tau_extr_dll(1);
    deltaTau_split = tau_ist(i) - tau_extr_split_dll(1);
       %split
    deltaTau_debug(i) = deltaTau;

    % direct
    p_late          = ro(deltaTau - DeltaC, m, n, tauChip);
    p_early         = ro(deltaTau + DeltaC, m, n, tauChip);

    % split
    p_late_split    = ro(deltaTau_split - DeltaC, 0, n,
       tauChip);
    p_early_split = ro(deltaTau_split + DeltaC, 0, n,
       tauChip);
```

```
delta_omega = 0;
ud_dll = 2*qcno* stdn_IQ^2 *T*(-p_early^2 + p_late^2);
Sd_dll = 8*qcno*T*stdn_IQ^2*(1/tauZero -
   (DeltaC/tauZero^2));
Ud_dll = ud_dll + n_dll(i);
delta_omega_split = 0;

if sep_pull_in_on
    deltaTauB = initial_err(n_exp); % режим раздельног
       о втягивания
else
    deltaTauB = deltaTau_split;
end

ud_split_dll = ((2/pi)*A_IQ)^2 *...%(2*((2*stdn_IQ *
   sqrt(2 * qcno * Tc))/pi)^2)
   *(sinc(delta_omega_split*T/2 /pi)^2)
    (-(p_early_split)^2*(cos(omega_B*(deltaTauB +
       DeltaB)))^2 +...
    +(p_late_split)^2*(cos(omega_B*(deltaTauB -
       DeltaB)))^2);
Sd_split_dll = ((2/pi)*A_IQ)^2 *... %sqrt(2)*
   *(sinc(delta_omega_split*T/2 /pi)^2)
    (2/tauChip^2)*(tauChip-DeltaB)*...
    (1+cos(2*omega_B*DeltaB) + (tauChip-DeltaC)*...
    omega_B*sin(2*omega_B*DeltaB));
Ud_split_dll = 0*Sd_split_dll*deltaTau_split +
   1*ud_split_dll + 1*n_split_dll(i);

tau_est_dll(:,i) = tau_extr_dll + K_dll*Ud_dll/Sd_dll;
tau_est_split_dll(:,i) = tau_extr_split_dll +
   K_dll*Ud_split_dll/Sd_split_dll; %split
curr_errTau_split = (tau_est_split_dll(1,i) -
   tau_ist(i));

if m == 1
    Thr_off = 30;
    Trh_1 = 97;
    Trh_2 = 300;
    Trh_1_split = 77;
```

```
        Trh_2_split = 300;
    elseif m == 5
        Thr_off = 9;
        Trh_1 = 16;
        Trh_2 = 120;
        Trh_1_split = 15;
        Trh_2_split = 120;
    elseif m == 15
        Thr_off = 3;
        Trh_1 = 5.8;
        Trh_2 = 31;
        Trh_1_split = 5.4;
        Trh_2_split = 31;
    end

    if sep_pull_in_on
        if i == (find(tf == Tmod/2))
            time_on(n_exp) = tf(i);
            sep_pull_in_on = 0;
            restart_flag = 0;
        end
    end

    deltaTauB_curr(i) = deltaTauB;
    deltaTauC_curr(i) = deltaTau_split;
    deltaTau_curr(i) = deltaTau;

    p_promt = ro(deltaTau_split, 0, n, tauChip);
    mIp1 = -(2/pi)*A_IQ*p_promt*sinc(deltaW*T/2
        /pi)*sin(deltaW*T/2+deltaPhi + omega_B*deltaTauB);
    mIp2 =  (2/pi)*A_IQ*p_promt*sinc(deltaW*T/2
        /pi)*sin(deltaW*T/2+deltaPhi - omega_B*deltaTauB);
    mQp1 = -(2/pi)*A_IQ*p_promt*sinc(deltaW*T/2
        /pi)*cos(deltaW*T/2+deltaPhi + omega_B*deltaTauB);
    mQp2 =  (2/pi)*A_IQ*p_promt*sinc(deltaW*T/2
        /pi)*cos(deltaW*T/2+deltaPhi - omega_B*deltaTauB);

    Ip1 = mIp1 + 0;
    Ip2 = mIp2 + 0;
```

```
        Qp1 = mQp1 + 0;
        Qp2 = mQp2 + 0;

        Ip = (Qp2 - Qp1)/2;
        Qp = (-Ip2 + Ip1)/2;
        sqrt_P(i) = sqrt(Ip^2 + Qp^2);

        %% direct
        p_promt = ro(deltaTau, m, n, tauChip);

        mIp = A_IQ*p_promt*sinc(deltaW*T/2
           /pi)*cos(deltaW*T/2 + deltaPhi);
        mQp = -A_IQ*p_promt*sinc(deltaW*T/2
           /pi)*sin(deltaW*T/2 + deltaPhi);

        Ip_direct = mIp + 0; Qp_direct = mQp + 0;

        sqrt_P_direct(i) = sqrt(Ip_direct^2 + Qp_direct^2);
    end

    errTau = (tau_est_dll(1,:) - tau_ist);
    errTau_split = (tau_est_split_dll(1,:) - tau_ist); % split

    if (abs(errTau(end)) < Trh_1/Light_C)
        P_zakhvat(q) = P_zakhvat(q) + 1;
    elseif (abs(errTau(end)) > Trh_1/Light_C) &&
       (abs(errTau(end)) < Trh_2/Light_C)
        P_logj_zero(q) = P_logj_zero(q) + 1;
    else
        P_sryv(q) = P_sryv(q) + 1;
    end

    if (abs(errTau_split(end)) < Trh_1_split/Light_C)
        P_zakhvat_split(q) = P_zakhvat_split(q) + 1;
    elseif (abs(errTau_split(end)) > Trh_1_split/Light_C) &&
       (abs(errTau_split(end)) < Trh_2_split/Light_C)
        P_logj_zero_split(q) = P_logj_zero_split(q) + 1;
    else
        P_sryv_split(q) = P_sryv_split(q) + 1;
    end
```

```
            figure(n_exp)
            set(0,'DefaultAxesFontSize', 14)
%             plot(tf, errTau*1e6, tf, errTau_split*1e6);hold on
%             plot(tf, ones(1,length(tf))*Trh_1/Light_C*1e6, '-.b',
   tf, ones(1,length(tf))*(-Trh_1/Light_C*1e6), '-.b');hold on
%             plot(tf, ones(1,length(tf))*Trh_2/Light_C*1e6, '-b',
   tf, ones(1,length(tf))*(-Trh_2/Light_C*1e6), '-b');hold on
%             plot(tf, ones(1,length(tf))*Trh_1_split/Light_C*1e6,
   '-.r', tf, ones(1,length(tf))*(-Trh_1_split/Light_C*1e6),
   '-.r');hold on
%             plot(tf, ones(1,length(tf))*Trh_2_split/Light_C*1e6,
   '--r', tf, ones(1,length(tf))*(-Trh_2_split/Light_C*1e6),
   '--r');hold on
%             ylim([-400/Light_C*1e6 400/Light_C*1e6])
%             xlabel('t, c', 'FontSize', 14)
%             ylabel('Ошибка оценивания задержки, мкс', 'FontSize',
   14)

            plot(tf, errTau*Light_C, tf, errTau_split*Light_C);hold on
            plot(tf, ones(1,length(tf))*Trh_1, '-.b', tf,
               ones(1,length(tf))*(-Trh_1), '-.b');hold on
            plot(tf, ones(1,length(tf))*Trh_2, '-b', tf,
               ones(1,length(tf))*(-Trh_2), '-b');hold on
            plot(tf, ones(1,length(tf))*Trh_1_split, '-.r', tf,
               ones(1,length(tf))*(-Trh_1_split), '-.r');hold on
            plot(tf, ones(1,length(tf))*Trh_2_split, '--r', tf,
               ones(1,length(tf))*(-Trh_2_split), '--r');hold on
            ylim([-400 400])
            grid on
%             if sep_pull_in_on == 0
%                 title('Режим раздельного втягивания');
%             end
            legend({'direct', 'split'}, 'FontSize', 14)

            xlabel('t, c', 'FontSize', 14)
            ylabel('Ошибка оценивания задержки, м', 'FontSize', 14)


            fprintf('Progress po Nexp: %.2f %% \n', n_exp*100/N_exp)
```

```
        end

        fprintf('***Progress po qcn0: %.2f %% \n',
           q*100/length(qcno_dB))
       waitbar(q/length(qcno_dB), hW);
    end

    fprintf('Progress po Delta: %.2f %% \n', d*100/length(D))
end
close(hW);

set(0,'DefaultAxesFontSize', 14)
% figure(1)
% plot(qcno_dB, [P_zakhvat./N_exp; P_logj_zero./N_exp;
   P_sryv./N_exp],'LineWidth',1);hold on;
% plot(qcno_dB, P_zakhvat_split./N_exp,'--', qcno_dB,
   P_logj_zero_split./N_exp,'--', qcno_dB,
   P_sryv_split./N_exp,'--','LineWidth',1);hold on;
% grid on
% xlabel('q_{c/n0}, дБГц', 'FontSize', 14)
% ylabel('Вероятности', 'FontSize', 14)
% legend({'Вероятность захвата', 'Вероятность ложного захвата',
   'Вероятность срыва'}, 'FontSize', 14)
%% title(['BW_{dll} = ' num2str(BW_dll) ' Гц'])
% title(['BOC(' num2str(m) ', ' num2str(n) '),' ' \Delta f = '
   num2str(BW_dll) ...
%       ' Гц, \Delta^{C} = ' num2str(Delta*Light_C) ' м'])
% figure(2)
% plot(tf, deltaTauB_curr, tf, deltaTauC_curr)
%

figure(3)

shag = 30;%18 35
deltaTauC_acf = -1/1e6:tauChip/shag:1/1e6;
deltaTauB_acf = deltaTauC_acf;

for i = 1:length(deltaTauC_acf)

    p_promt(i) = ro(deltaTauC_acf(i), 0, n, tauChip);
```

```
    for j = 1:length(deltaTauB_acf)

        mIp1 = -(2/pi)*A_IQ*p_promt(i)*sinc(deltaW*T/2
            /pi)*sin(deltaW*T/2+deltaPhi + omega_B*deltaTauB_acf(j));
        mIp2 =  (2/pi)*A_IQ*p_promt(i)*sinc(deltaW*T/2
            /pi)*sin(deltaW*T/2+deltaPhi - omega_B*deltaTauB_acf(j));
        mQp1 = -(2/pi)*A_IQ*p_promt(i)*sinc(deltaW*T/2
            /pi)*cos(deltaW*T/2+deltaPhi + omega_B*deltaTauB_acf(j));
        mQp2 =  (2/pi)*A_IQ*p_promt(i)*sinc(deltaW*T/2
            /pi)*cos(deltaW*T/2+deltaPhi - omega_B*deltaTauB_acf(j));

        Ip1 = mIp1 + 0;
        Ip2 = mIp2 + 0;

        Qp1 = mQp1 + 0;
        Qp2 = mQp2 + 0;

        Ip = (Qp2 - Qp1)/2;
        Qp = (-Ip2 + Ip1)/2;
        sqrt_P_acf(i,j) = sqrt(Ip^2 + Qp^2);
    end
end
[deltaTauB_acf , deltaTauC_acf] = meshgrid (deltaTauB_acf,
    deltaTauC_acf);

sqrt_P_triu = triu(sqrt_P_acf);
sqrt_P_triu(sqrt_P_triu==0)=NaN;

s_full = surf(deltaTauB_acf*1e6, deltaTauC_acf*1e6,
    sqrt_P_acf./max(max(sqrt_P_acf))); hold on
% s_full = surf(deltaTauB_acf*1e6, deltaTauC_acf*1e6,
    sqrt_P_triu./max(max(sqrt_P_acf))); hold on

xlabel('\delta\tau^{B}, мкс')
ylabel('\delta\tau^{C}, мкс')
colormap bone
zlim([0 1.1])
```

```
plot3(deltaTauB_curr(:,2:end)*1e6, deltaTauC_curr(:,2:end)*1e6,
    sqrt_P(:,2:end)/max(max(sqrt_P_acf)), '-x','Color',[0.8500 0.3250
    0.0980]); hold on % путь
plot3(deltaTauB_curr(end)*1e6, deltaTauC_curr(end)*1e6,
    sqrt_P(end)/max(max(sqrt_P_acf)), 'or','LineWidth',2); hold on % ко
    нечная точка
plot3(deltaTauB_curr(2)*1e6, deltaTauC_curr(2)*1e6,
    sqrt_P(2)/max(max(sqrt_P_acf)), 'og'); hold on % начальная точка
plot3(diag(deltaTauB_acf)*1e6, diag(deltaTauC_acf)*1e6,
    diag(sqrt_P_acf)/max(max(sqrt_P_acf)), 'Color',[0.4660 0.6740
    0.1880],'LineWidth', 3); hold on % диагональ
% plot3(ones(1, length(deltaTauC_acf(:,1))),deltaTauC_acf(:,1)*1e6 ,
    sqrt_P_acf(:,1)/max(max(sqrt_P_acf)), 'Color','red','LineWidth',
    3); hold on % АКФ кода
% plot3(deltaTauB_acf(1,:)*1e6, -ones(1, length(deltaTauC_acf(:,1))),
    ...
% sqrt_P_acf(find(sqrt_P_acf(:,1) ==
    max(sqrt_P_acf(:,1))),:)/max(max(sqrt_P_acf)), 'Color',[0 0.4470
    0.7410],'LineWidth', 1.5); hold on % АКФ поднесущей
plot3([1 -1], [1 -1], [0 0], 'k','LineWidth', 0.5); hold on % диагонал
    ь дельвaус = дельтатауб
% plot3([0 0], [-deltaTauC_acf(end,end)*1e6
    -deltaTauC_acf(end,end)*1e6], [0 155]/max(max(sqrt_P_acf)),
    'k','LineWidth', 0.5); hold on
% plot3([-deltaTauB_acf(1,1)*1e6 -deltaTauB_acf(1,1)*1e6], [0 0], [0
    155]/max(max(sqrt_P_acf)), 'k','LineWidth', 0.5); hold on
scale = 40;
deltaTauB_curr_downsample = downsample(deltaTauB_curr,scale);
deltaTauC_curr_downsample = downsample(deltaTauC_curr,scale);
sqrt_P_downsample = downsample(sqrt_P,scale);

% figure(4)
% plot(deltaTauB_acf(1,:)*1e6,sqrt_P_acf(find(sqrt_P_acf(:,1) ==
    max(sqrt_P_acf(:,1))),:)/max(max(sqrt_P_acf)), 'Color',[0 0.4470
    0.7410],'LineWidth', 1.5); hold on
% plot(deltaTauB_curr(2)*1e6, sqrt_P(2)/max(max(sqrt_P_acf)), 'og');
    hold on

%% direct pull in
figure(5)
```

```
shag = 100;%18 35
deltaTau_acf = -1/1e6:tauChip/shag:1/1e6;
for i = 1:length(deltaTau_acf)
    p_promt(i) = ro(deltaTau_acf(i), m, n, tauChip);
        mIp = A_IQ*p_promt(i)*sinc(deltaW*T/2 /pi)*cos(deltaW*T/2 +
           deltaPhi);
        mQp = -A_IQ*p_promt(i)*sinc(deltaW*T/2 /pi)*sin(deltaW*T/2 +
           deltaPhi);
        Ip = mIp + 0; Qp = mQp + 0;
        sqrt_P_acf_direct(i) = sqrt(Ip^2 + Qp^2);
end

plot(deltaTau_acf(2:end)*1e6,
   sqrt_P_acf_direct(2:end)/max(sqrt_P_acf_direct(2:end)),'LineWidth',2);
   hold on
xlabel('\delta\tau^{C}, мкс')
ylim([0 1.1])

plot(deltaTau_curr(2:end)*1e6,
   sqrt_P_direct(2:end)/max(sqrt_P_acf_direct(2:end)),
   '-x','Color',[0.8500 0.3250 0.0980]); hold on
plot(deltaTau_curr(end)*1e6,
   sqrt_P_direct(end)/max(sqrt_P_acf_direct(2:end)),
   'xr','LineWidth',5); hold on
plot(deltaTau_curr(2)*1e6,
   sqrt_P_direct(2)/max(sqrt_P_acf_direct(2:end)), 'og'); hold on
grid on;

 % Срезы многомерной АКФ
 figure(6);
 sqrt_P_triu = triu(sqrt_P_acf);
 sqrt_P_triu(sqrt_P_triu==0)=NaN;

 s = surf(deltaTauB_acf*1e6, deltaTauC_acf*1e6,
    sqrt_P_triu/max(max(sqrt_P_acf))); hold on
 colormap Bone
 xlabel('\delta\tau^{B}, мкс')
 ylabel('\delta\tau^{C}, мкс')
```

```
plot3(diag(deltaTauB_acf)*1e6, diag(deltaTauC_acf)*1e6,
    diag(sqrt_P_acf)/max(max(sqrt_P_acf)), 'Color',[0.4660 0.6740
    0.1880],'LineWidth', 3); hold on
plot3(ones(1, length(deltaTauC_acf(:,1))),deltaTauC_acf(:,1)*1e6 ,
    sqrt_P_acf(:,1)/max(max(sqrt_P_acf)), 'Color','red','LineWidth',
    3); hold on
plot3(deltaTauB_acf(1,:)*1e6, -ones(1, length(deltaTauC_acf(:,1))) ,
    sqrt_P_acf(find(sqrt_P_acf(:,1) ==
    max(sqrt_P_acf(:,1))),:)/max(max(sqrt_P_acf)), 'Color',[0 0.4470
    0.7410],'LineWidth', 1.5); hold on
plot3([1 -1], [1 -1], [0 0], 'k','LineWidth', 0.5); hold on

%% Сохранить в svg
print('-painters','-dsvg','myVectorFile')

%% GIF'ka
% fig = figure(10);
%% surf(surfACF.deltaTauC*1e6, surfACF.deltaTauB*1e6,
   surfACF.sqrt_P); hold on
% surf(deltaTauC_acf*1e6, deltaTauB_acf*1e6, sqrt_P_acf); hold on
% xlabel('\delta\tau^{B}, мкс')
% ylabel('\delta\tau^{C}, мкс')
% xlim([-tauChip*1e6, tauChip*1e6]);
%     ylim([-tauChip*1e6, tauChip*1e6]);
%     zlim([0, 150]);
%% создание первого пустого кадра
%% set(fig,'Position',[350,200,700,300]);
% frame = getframe(fig);
% [im,map] = rgb2ind(frame.cdata,256);
% imwrite(im,map,'animation1.gif','DelayTime',0,'Loopcount',0);
% цикл анимации
% for i=2:length(deltaTauB_curr_downsample)
%     plot3(deltaTauB_curr_downsample(i)*1e6,
   deltaTauC_curr_downsample(i)*1e6,
   sqrt_P_downsample(i)+1,'-xm','LineWidth',2);
%     hold on;
%     plot3([deltaTauB_curr_downsample(i)*1e6
   deltaTauB_curr_downsample(i+1)*1e6],...
%         [deltaTauC_curr_downsample(i)*1e6
   deltaTauC_curr_downsample(i+1)*1e6], ...
```

```
%           [sqrt_P_downsample(i)+1
   sqrt_P_downsample(i+1)+1],'-m','LineWidth',2);
%       hold on;
%
%       frame = getframe(fig);
%       [im,map] = rgb2ind(frame.cdata,256);
%
   imwrite(im,map,'animation1.gif','DelayTime',0.01,'WriteMode','Append');
% end

% 2D ACF
 figure(6)
 subplot(3,1,1)
 plot(deltaTauC_acf(:,1)*1e6 , sqrt_P_acf(:,1)/max(sqrt_P_acf(:,1)),
    'Color','red','LineWidth', 1.5); hold on
 xlabel('\delta\tau^{C}, мкс')
 ylabel('|\rho(\delta\tau^{C})|')
 grid on
 subplot(3,1,2)
 plot(deltaTauB_acf(1,:)*1e6, sqrt_P_acf(find(sqrt_P_acf(:,1) ==
    max(sqrt_P_acf(:,1))),:) ...
     /max(sqrt_P_acf(find(sqrt_P_acf(:,1) ==
        max(sqrt_P_acf(:,1))),:)), 'Color',[0 0.4470
        0.7410],'LineWidth', 1.5); hold on
 xlabel('\delta\tau^{B}, мкс')
 ylabel('|\rho(\delta\tau^{B})|')
 grid on
 subplot(3,1,3)
 plot(diag(deltaTauB_acf)*1e6, diag(sqrt_P_acf), 'Color',[0.4660
    0.6740 0.1880],'LineWidth', 1.5); hold on
 xlabel('\delta\tau^{B} = \delta\tau^{C}, мкс')
 ylabel('|\rho(\delta\tau^{B} = \delta\tau^{C})|')
 grid on
```