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

|  |  |
| --- | --- |
| **Институт** | **ИВТ** |
| **Кафедра** | **УИТ** |

**ЗАДАНИЕ**

**НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ (бакалаврскую работу)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Направление** | | | 27.03.04 Управление в технических системах | | | |
| (код и наименование) | | | | | | |
| **Направленность (профиль)** | | | | | | Управление и информатика в |
| технических системах | | | | | | |
|  | | | | | | |
| **Форма обучения** | | | | | **очная** | |
|  | (очная/очно-заочная/заочная) | | | | | |
| |  |  | | --- | --- | | **Тема:** | «Исследование влияния коррелированности значений решающей | | функции в алгоритмах обнаружения разладки стохастических сигналов, | | | основанных на теории случайных блужданий» | | | | | | | | |
| **Студент** | | **А-01-19 Викторов Г.О.** | | | | |
| группа подпись фамилия и инициалы | | | | | | |
| **Научный руководитель** | | | | **к.т.н. доцент Рябов С.Н.** | | |
| 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** | | | | **д.т.н. доцент Бобряков А.В.** | | |
| уч. степень звание подпись фамилия и инициалы | | | | | | |
|  | | | | | | |
| **Место выполнения работы** | | | | | | НИУ «МЭИ» Кафедра Управления и |
| интеллектуальных технологий | | | | | | |

**СОДЕРЖАНИЕ РАЗДЕЛОВ ЗАДАНИЯ И ИСХОДНЫЕ ДАННЫЕ**

|  |
| --- |
| 1.Изучение литературы по проблематике обнаружения разладки |
| случайных процессов, известным методам обнаружения и их классификации. |
| Составление краткого обзора. |
| 2.Алгоритм, основанный на теории случайных блужданий : назначение, |
| структура, принцип действия и основные характеристики. |
| Программная реализация и апробация. |
| 3.Изучение и подбор методов задания корреляции для гауссовского процесса. |
| 4.Имитационное моделирование как инструмент исследования алгоритма. |
| Разработка программных средств имитационного моделирования . |
| Планирование имитационного эксперимента для |
| исследования варианта разладки коррелированного гауссовского процесса. |
| 5.Реализация имитационного эксперимента и обработка его результатов. |
| 6.Сопоставление исследуемого алгоритма с другими известными |
| алгоритмами обнаружения разладки по показателю эффективности. |
| 7.Анализ и обсуждение полученных результатов. |
| 8.Написание и оформление работы. |

**ПЕРЕЧЕНЬ ГРАФИЧЕСКОГО МАТЕРИАЛА**

|  |  |
| --- | --- |
|  |  |
| **Количество слайдов в презентации** |  |
|  |  |

**РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА**

|  |
| --- |
| **1. Феллер У.** Введение в теорию вероятностей и ее приложения. Том 1. |
| (An Introduction to Probability Theory and its Applications. Volume I), М.: Изд- |
| во «Мир», 1970. 499 с. |
| **2.** **Филаретов Г.Ф., Червова А.А., Бучаала З.** Последовательный |
| непараметрический алгоритм обнаружения разладки временнóго ряда // |
| Датчики и системы, 2020, №1, с.9 – 16. |
| **3.** **Филаретов Г.Ф., Червова А.А., Бучаала З.** Непараметрический |
| метод обнаружения разладки временнóго ряда c использованием механизма |
| случайных блужданий».// Известия ЮФУ. Технические науки. 2020, №4, с.107-117. |
| **4.** **Филаретов Г.Ф., Бучаала З.** Разработка алгоритмов обнаружения |
| разладки временных рядов на основе непараметрических критериев проверки |
| гипотез // Вестник МЭИ. 2021, №3. – с. 67 – 77. |

АННОТАЦИЯ

В работе рассмотрены вопросы, связанные с решением задачи обнаружения спонтанного изменения (разладки) вероятностных свойств временного ряда в реальном масштабе времени.Предметом рассмотрения является последовательный алгоритм обнаружения, основанный на методе случайных блужданий. Исследована эффективность работы алгоритма для гауссовского некоррелированного и коррелированного временных рядов.

Работа состоит из введения, шести разделов, заключения, списка литературы из 4 наименований, одного приложения, содержит 47 страниц текста, 18 рисунков, 14 таблиц.

# Содержание

[Содержание 4](#_Toc137485389)

[Введение 5](#_Toc137485390)

[1. АЛГОРИТМЫ ОБНАРУЖЕНИЯ СПОНТАННОЙ РАЗЛАДКИ НАБЛЮДАЕМЫХ ПРОЦЕССОВ 6](#_Toc137485391)

[2. АЛГОРИТМ, ОСНОВАННЫЙ НА ТЕОРИИ СЛУЧАЙНЫХ БЛУЖДАНИЙ:НАЗНАЧЕНИЕ, СТРУКТУРА, ПРИНЦИП ДЕЙСТВИЯ И ОСНОВНЫЕ ХАРАКТЕРИСТИКИ. 15](#_Toc137485392)

[3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ И АПРОБАЦИЯ 18](#_Toc137485393)

[3.1 Структура программы 18](#_Toc137485394)

[3.2 Первоначальная апробация программы 20](#_Toc137485395)

[3.3 Определение длины серий, по заданному среднему времени между ложными тревогами 25](#_Toc137485396)

[3.4 Выводы 27](#_Toc137485397)

[4. ИЗУЧЕНИЕ И ПОДБОР МЕТОДОВ ЗАДАНИЯ КОРРЕЛЯЦИИ ДЛЯ ГАУССОВСКОГО ПРОЦЕССА. 28](#_Toc137485398)

[5. ИССЛЕДОВАНИЕ РАБОТЫ МЕТОДА СЛУЧАЙНЫХ БЛУЖДАНИЙ НА БИНАРНОМ КОРРЕЛИРОВАННОМ СИГНАЛЕ В ОТСУТСТВИИ РАЗЛАДКИ 32](#_Toc137485399)

[5.1 Сравнение работы метода для сигнала без корреляции и для коррелированного сигнала 32](#_Toc137485400)

[5.2 Исследование работы алгоритма при коррелированном бинарном сигнале 34](#_Toc137485401)

[5.3 Выводы 35](#_Toc137485402)

[6. ИССЛЕДОВАНИЕ РАБОТЫ МЕТОДА СЛУЧАЙНЫХ БЛУЖДАНИЙ НА “ГАУССОВСКОМ” КОРРЕЛИРОВАННОМ СИГНАЛЕ В В ПРИСУТСТВИИ РАЗЛАДКИ 35](#_Toc137485403)

[6.1 Сравнение работы метода для сигнала без корреляции и для коррелированного сигнала: 36](#_Toc137485404)

[6.2 Определение длины серий, по заданному среднему времени между ложными тревогами 41](#_Toc137485405)

[6.3 Выводы 43](#_Toc137485406)

[ЗАКЛЮЧЕНИЕ 45](#_Toc137485407)

[СПИСОК ЛИТЕРАТУРЫ 46](#_Toc137485408)

[*Приложение* 47](#_Toc137485409)

[Листинг программы 47](#_Toc137485410)

# Введение

Исследуется задача, которая заключается в максимально быстром и точном определении момента, при котором происходит изменение свойств изучаемых объектов и характера протекающих в них процессов. Эта задача, которую часто называют задачей о разладке, имеет целью выявление и фиксацию существенных изменений в данных и системах.

Обнаружение разладки в стохастических сигналах является актуальной и важной проблемой во многих областях науки и техники. Стохастические сигналы широко применяются в таких областях, как финансовая аналитика, электроника, медицина, экология и многие другие. Разладка в сигналах может указывать на изменения в системе или среде, и ее обнаружение имеет большое значение для принятия решений и предотвращения потенциальных проблем.

Разладка стохастических процессов представляет собой неожиданное изменение характеристик наблюдаемого случайного процесса, которое происходит в неизвестный и непредсказуемый момент времени. Задача обнаружения разладки дискретного стохастического процесса *X(t)* в реальном времени может быть сформулирована следующим образом: предполагается, что имеется последовательность наблюдений *x(t1) = x1*, *x(t2) = x2* и так далее. Возможны два состояния объекта: "норма" с определенными вероятностными характеристиками процесса *X(t)* и "разладка", когда вероятностные свойства *X(t)* отличаются от "нормальных". Переход из состояния "норма" в состояние "разладка" происходит в заранее неизвестный момент времени *t0*. Задача состоит в обнаружении этого момента *t0* на основе последовательности значений *x1*, *x2*…, при этом желательно минимизировать задержку *τ*зап между моментом разладки и его обнаружением.

Для оперативного обнаружения разладки необходимо использовать методы и алгоритмы, способные работать в режиме реального времени и обрабатывать последовательные значения по мере их поступления.

# АЛГОРИТМЫ ОБНАРУЖЕНИЯ СПОНТАННОЙ РАЗЛАДКИ НАБЛЮДАЕМЫХ ПРОЦЕССОВ

Методы определения разладки могут быть разделены на две категории: статистические методы и методы машинного обучения. Статистические методы основаны на анализе свойств данных, распределений и структуры сигнала, а также на применении статистических тестов и моделей. Методы машинного обучения, в свою очередь, используют обучающие данные для построения моделей, которые могут обнаруживать и анализировать разладку в данных.

Выбор метода определения разладки зависит от специфики задачи, доступных данных, требований к скорости обработки и точности обнаружения. Кроме того, эффективность методов может варьироваться в зависимости от типа разладки (внезапная или постепенная), структуры сигнала и уровня шума. Разработка эффективных алгоритмов определения разладки является активной областью исследования, и новые подходы и методы постоянно разрабатываются для улучшения точности и скорости обнаружения разладки.

Виды статистических методов разладки включают параметрические и непараметрические подходы:

1. Параметрические методы разладки: Эти методы предполагают известную модель данных и параметры, которые описывают статистические свойства процесса. Они основаны на статистических тестах, которые сравнивают ожидаемое поведение данных с наблюдаемыми значениями, чтобы обнаружить отклонения и разладки. Примерами параметрических методов являются контрольные карты Шухарта, экспоненциально взвешенное скользящее среднее (*EWMA*), а также методы, основанные на *ARIMA*-моделях и регрессии.

2. Непараметрические методы разладки: Эти методы не требуют предположения о распределении данных или модели процесса. Они основаны на непараметрических тестах и алгоритмах, которые исследуют структуру данных и обнаруживают аномалии, не полагаясь на конкретную модель. Примерами непараметрических методов являются методы, основанные на рангах (например, критерий Уилкоксона-Манна-Уитни), методы бутстрэпа и методы, использующие алгоритмы кластеризации и машинное обучение.

Выбор между параметрическими и непараметрическими методами зависит от характера данных, доступной информации о модели и статистическом свойстве процесса. Каждый подход имеет свои преимущества и ограничения, и выбор определенного метода требует анализа и понимания особенностей конкретной задачи обнаружения разладки.

Любой последовательный алгоритм обнаружения разладки должен иметь определенные оптимальные свойства. С одной стороны, желательно минимизировать среднее время обнаружения разладки (минимальную задержку в обнаружении разладки *τзап*), а с другой стороны, максимизировать среднее время между ложными тревогами ().

На практике, для оценки производительности контрольного алгоритма, часто используются две основные характеристики:

1. Среднее время между ложными тревогами (): Это среднее время между случаями, когда алгоритм срабатывает и сообщает о разладке, хотя разладки фактически нет. Меньшее значение указывает на более надежный алгоритм, который редко генерирует ложные тревоги.

2. Среднее время задержки обнаружения (): Это среднее время, которое проходит от момента разладки до момента обнаружения разладки алгоритмом. Меньшее значение означает, что алгоритм оперативно реагирует на разладку и быстро обнаруживает изменения в данных.

При настройке контрольного алгоритма необходимо учитывать информацию о характеристиках наблюдаемого процесса до и после разладки, а также желаемые свойства самого алгоритма. Настройка включает выбор номинального значения разладки и порогового значения. Для настройки и оценки производительности алгоритма необходима справочная информация, которая может быть представлена в графическом или табличном виде и содержать:

- Зависимость порогового значения от среднего времени между ложными тревогами для различных значений номинальной разладки.

- Зависимость времени задержки обнаружения от порогового значения или среднего времени между ложными тревогами для оценки скорости работы алгоритма при наличии номинальной разладки.

Использование такой справочной информации позволяет эффективно настроить контрольный алгоритм и оценить его производительность в соответствии с требованиями задачи.

Существуют два основных подхода к решению задачи обнаружения разладки: апостериорный и последовательный подходы.

В апостериорном подходе данные регистрируются и сохраняются, а затем, в процессе последующей обработки, выполняется оценка неизвестного момента t0. Этот подход интерпретируется как обычная задача оценивания, и для соответствующих алгоритмов применяются стандартные требования к получению оценок с наибольшей точностью, например, с минимальным смещением и дисперсией.

Однако больший интерес представляет последовательный подход, при котором обнаружение разладки происходит в реальном времени, одновременно с получением данных о значениях случайного процесса. В этом случае требуется обнаружить разладку как можно быстрее после ее возникновения при текущей обработке данных. Такие подходы называются последовательными алгоритмами обнаружения разладки.

Последовательные алгоритмы обнаружения разладки имеют практическую ценность, поскольку позволяют оперативно реагировать на изменения в данных и обнаруживать разладку практически в режиме реального времени. Эти алгоритмы основаны на анализе последовательности данных, с использованием статистических тестов, моделей или других методов, чтобы определить момент разладки и принять соответствующие меры.

Рассмотрим классификацию последовательных алгоритмов для обнаружения разладки в стохастических временных рядах. В данной классификации выделяются четыре основные группы: байесовские алгоритмы, эвристические алгоритмы, алгоритмы на основе подхода Неймана-Пирсона и алгоритмы, использующие идеи последовательного анализа.

Байесовские алгоритмы направлены на минимизацию среднего риска, который складывается из потерь от наличия нераспознанной разладки и ложных срабатываний. Они используют байесовский подход и учитывают априорную информацию о вероятности разладки и статистических свойствах процесса.

Эвристические алгоритмы основаны на интуитивных представлениях о наилучшей организации обнаружения разладки и удобны для практической реализации. Они не всегда имеют строгое математическое обоснование, но в некоторых случаях дают хорошие результаты. Однако в других случаях эти алгоритмы могут давать неверные результаты.

Алгоритмы на основе подхода Неймана-Пирсона используют классические критерии проверки гипотез. Они многократно проверяют гипотезы о виде функции распределения вероятностей или значениях параметров (например, среднего или дисперсии). Наиболее известным примером таких алгоритмов являются контрольные карты Шухарта, которые применяются для отслеживания изменчивости параметров и обеспечения стабильности технологических процессов.

Алгоритмы, основанные на идеях последовательного анализа, используют статистику отношения правдоподобия, но с особенностью - в них реализуется операция отражения от нижней решающей границы. Это позволяет эффективно обнаруживать разладку при анализе последовательности данных.

При применении последовательных алгоритмов на практике наиболее часто используют алгоритмы на основе подхода Неймана-Пирсона.

Алгоритмы, основанные на подходе Неймана-Пирсона, включают:

1. Алгоритм кумулятивных сумм (*CUSUM*-алгоритм): Он использует накопление сумм разностей между текущими значениями и их ожидаемыми значениями для обнаружения разладки.

2. Алгоритм скользящего среднего (*Moving Average*) или *MA*-алгоритм: Он вычисляет среднее значение последовательности данных и сравнивает его с пороговым значением для определения разладки.

3. Алгоритм экспоненциального сглаживания (*EWMA*-алгоритм): Он применяется для обнаружения разладки во временных рядах путем сглаживания последовательности и сравнения с пороговым значением.

4. Контрольные карты Шухарта: Это наиболее известный пример. Они используются для отслеживания изменчивости параметров и обеспечения стабильности технологических процессов.

Все эти алгоритмы основаны на статистических методах и используют критерии проверки гипотез, предложенные Нейманом и Пирсоном, для обнаружения разладки в данных.

Все последовательные методы обнаружения разладки основаны на вычислении решающих функций *gi = gi (xi , xi-1 , xi-2,…)*на основе наблюдаемых дискретных значений *xi*. Эти функции вычисляются рекуррентно, используя значения статистики на предыдущем шаге и текущее измеренное значение xi. Затем вычисленное значение *gi* сравнивается с пороговым уровнем *H*, который разделяет область возможных значений статистики на две подобласти: *Ω0* и *Ω1*. Подобласть *Ω0*соответствует отсутствию разладки, а подобласть *Ω1*- критической области. Если *gi* принадлежит к *Ω0*, процесс контроля продолжается, и считается, что разладки нет. Если *gi* принадлежит к *Ω1*, подается сигнал о наличии разладки (сигнал тревоги). По желанию экспериментатора контрольная процедура может быть запущена заново.

Чаще всего предполагается, что разладка проявляется в изменении параметров стохастического процесса, таких как математическое ожидание, дисперсия, корреляционно-спектральные свойства и т.д., а не в изменении функции распределения вероятностей *.*

Более подробно рассмотрим каждый из алгоритмов.

Алгоритм кумулятивных сумм (*CUSUM*-алгоритм)

Алгоритм кумулятивных сумм (*CUSUM*) - это последовательный метод обнаружения разладки, который используется для мониторинга изменений в стохастических процессах. Он основан на идее накопления суммарных отклонений от некоторого базового уровня.

Принцип работы *CUSUM* алгоритма заключается в следующем:

1. Определяется статистика *C\_i* на каждом шаге *i*, которая представляет накопленную сумму отклонений от некоторой базовой значения или среднего.

*Ci = max(0, Ci-1 + xi - μ)*

Где *xi* - текущее измерение ряда, *μ* - базовое значение или среднее.

2. Решающая функция *gi* вычисляется как разница между статистикой *Ci* и пороговым значением *H*.

*gi = Ci - H*

Если *gi* превышает нулевое значение, то это указывает на наличие разладки.

3. Если *gi* положительное, то считается, что разладка произошла, и сигнализируется об этом. Затем статистика *Ci* обнуляется и процесс начинается заново.

*CUSUM* алгоритм позволяет обнаруживать разладки с высокой чувствительностью, так как накапливает суммарные отклонения от базового значения. Он также способен обнаруживать как резкие, так и постепенные изменения в ряде данных.

*CUSUM* алгоритм широко используется в различных областях, включая производство, экономику, финансы, медицину и т.д., где необходимо раннее обнаружение и контроль изменений в процессах и системах.

Алгоритм скользящего среднего (*Moving Average*) или *MA*-алгоритм

Алгоритм скользящего среднего (*Moving Average*) может использоваться для обнаружения разладки во временных рядах. Он основан на следующем принципе: если точка данных во временном ряду значительно отличается от своего окружения, это может указывать на наличие разладки.

Для определения разладки с помощью алгоритма скользящего среднего используется следующий подход:

1. Вычисление скользящего среднего для каждой точки временного ряда. Размер окна выбирается в соответствии с характеристиками данных и желаемой чувствительностью алгоритма. Обычно используются окна размером от нескольких точек до нескольких временных интервалов.

2. Вычисление отклонения каждой точки данных от соответствующего скользящего среднего значения. Это можно сделать путем вычитания значения скользящего среднего из соответствующей точки временного ряда.

3. Оценка разладки. Если отклонение точки данных превышает определенный пороговый уровень, считается, что произошла разладка. Пороговый уровень выбирается на основе анализа исторических данных и требуемой чувствительности алгоритма.

4. Повторение алгоритма для последующих точек данных для обнаружения дополнительных разладок во временном ряду.

Алгоритм экспоненциального сглаживания (*Exponential Smoothing*) также может быть использован для обнаружения разладки во временных рядах. Он позволяет сгладить шумы и колебания в данных и выявить отклонения, которые могут свидетельствовать о разладке.

Для определения разладки с использованием алгоритма экспоненциального сглаживания следуйте следующим шагам:

1. Инициализация начального значения сглаженного ряда (*g0)* и параметра сглаживания (*α*). Значение параметра *α* должно быть выбрано в диапазоне от 0 до 1, где более близкое к 1 значение означает большую чувствительность к последним значениям ряда.

2. Для каждой последующей точки данных *Xt* во временном ряду:

- Вычисление текущего сглаженного значения *St* с использованием формулы экспоненциального сглаживания:

*gi = α \* Xi + (1 - α) \* gi-1*

- Вычисление ошибки разладки (*Et*) путем нахождения разницы между текущим значением ряда и соответствующим сглаженным значением:

*Ei = Xi - gi*

3. Оценка разладки. Если ошибка разладки (*Ei*) превышает определенный пороговый уровень, то считается, что произошла разладка. Пороговый уровень выбирается на основе анализа исторических данных и требуемой чувствительности алгоритма.

4. Повторение алгоритма для последующих точек данных для обнаружения дополнительных разладок во временном ряду.

Контрольные карты Шухарта (*Shewhart Control Charts*) - это графический метод мониторинга и обнаружения разладки в процессах. Они были разработаны Уолтером Шухартом и используются для контроля статистических показателей процесса, таких как среднее значение, стандартное отклонение или диапазон.

Основная идея контрольных карт Шухарта заключается в следующем:

1. Измерение показателя процесса. Для каждой выборки из процесса измеряется соответствующий показатель, например, среднее значение, стандартное отклонение или диапазон.

2. Вычисление статистических показателей. Для каждой выборки вычисляются статистические показатели, такие как среднее значение и стандартное отклонение. Эти значения используются в дальнейшем для построения контрольных карт.

3. Построение контрольных карт. Для каждого статистического показателя строится контрольная карта. Контрольные карты имеют ось времени по горизонтали и ось значений показателя по вертикали. На графике отображается среднее значение показателя и контрольные пределы, которые определяются на основе статистических свойств процесса.

4. Обнаружение разладки. При мониторинге процесса, если точки показателя выходят за контрольные пределы или показывают другие необычные шаблоны или тренды, считается, что произошла разладка или нарушение процесса.

Контрольные карты Шухарта являются эффективным инструментом для наблюдения за процессами и своевременного обнаружения разладки. Они позволяют операторам и аналитикам проводить мониторинг качества процесса и принимать меры по устранению отклонений, что способствует повышению производительности и снижению доли брака.

# АЛГОРИТМ, ОСНОВАННЫЙ НА ТЕОРИИ СЛУЧАЙНЫХ БЛУЖДАНИЙ:НАЗНАЧЕНИЕ, СТРУКТУРА, ПРИНЦИП ДЕЙСТВИЯ И ОСНОВНЫЕ ХАРАКТЕРИСТИКИ.

Впервые идея использования механизма одномерных случайных блужданий для обнаружения разладки была представлена в работе, опубликованной в 1975 году. В этой работе была рассмотрена классическая схема одномерных случайных блужданий, где на каждом шаге в дискретный момент времени n текущая координата точки *zn*, представляющей процесс блуждания, с вероятностью p получает приращение +1 или с вероятностью *q=1–p* приращение 0.

,

где *Me(x)* представляет собой значение медианы наблюдаемого временного ряда в состоянии "норма". Это соотношение описывает случай, когда *p = q =* 0.5, но при наличии разладки симметрия нарушается, и вероятность *p* становится больше 0.5, что соответствует сдвигу медианы в положительную сторону.

Затем рассматривается одно из возможных направлений развития методов обнаружения разладки, основанных на случайных блужданиях, а именно использование теории серий "успехов". Контрольная процедура предлагаемого метода сводится к вводу значения контролируемого процесса *xn* на каждом такте работы, его преобразованию в значение приращения *zn* в соответствии с вышеуказанным соотношением, подсчету текущей длины серии kn следующих значений *zn = +1: kn = k­n-1 +1*, если *z­n = 0,* то *kn = 0*, также фиксируется номер такта *n\** на котором будет выполнено условие *kn\*= kn= k*, где *k* представляет собой заранее заданное пользователем значение, выбранное с учетом желаемого среднего времени между ложными тревогами *.*

Исходными данными, вводимыми пользователем, являются номер такта, на котором будет введена разладка *N*, общее количество значений временного ряда *L*, медиана временного ряда в состоянии “разладка” *mx*, длина серий *k* и количество запусков *m*. Также программа позволяет выбрать режим работы с некоррелированным или коррелированным сигналом.

В разработанной программе предусматривается *m* запусков, при каждом запуске генерируется временной ряд длиной *L*, с разладкой, которая начинается на такте *N*, далее сгенерированный ряд прогоняется через разработанный алгоритм по определению разладки, считается среднее время между ложными тревогами и время запаздывания . Затем данный процесс повторяется, пока не будет выполнено *m* запусков. Далее проводится усреднение среднего времени между ложными тревогами и времени запаздывания, полученные результаты выводятся пользователю.

Общая последовательность операций каждого запуска выглядит следующим образом:

1. Генерируется *N* значений *x1, x2,…,xN* , распределённых заданным образом с медианой *Me(X)* = 0.5 и *L-N* значений *xN+1, xN+2,…,xL*  с медианой *Me(X)*= *mx*.
2. Сгенерированный ряд последовательно прогоняется *i(0,…..,L)* тактов через алгоритм по обнаружению разладки.

2.1 На каждом *i* – ом такте Значение *xi* преобразуется в значение *zi* по следующему правилу:

* 1. Количество подряд идущих «плюсов» - *kn*, сравнивается с заданным пользоватетелем *k*.
* Если условие истинно, то *i*-ыйзапуск программы завершается, фиксируется номер такта. Если *i <= N*, то это значение запоминается как , если *i > N*, то это значение запоминается как и происходит переход к следующему такту
* Если условие ложно, происходит переход к следующему такту

1. При окончании цикла, рассчитыватся и запоминается и , затем программа запускается заного и так до тех пор, пока не будет достигнуто *m* запусков.
2. Вывод пользователю усредненого по *m* значениям среднего времени между ложными тревогами и среднего времени запаздывания.

# ПРОГРАММНАЯ РЕАЛИЗАЦИЯ И АПРОБАЦИЯ

## Структура программы

Рассмотрим функцию по генерации временного ряда *get\_signal()*. В неё передаются 3 входных параметра: *N*- номер такта номинальной разладки, *L*- общая длина ряда, *mx*- медиана ряда с разладкой и глобальный параметр *corr*, который определяет тип генерируемого сигнала(коррелированный/некоррелированный).

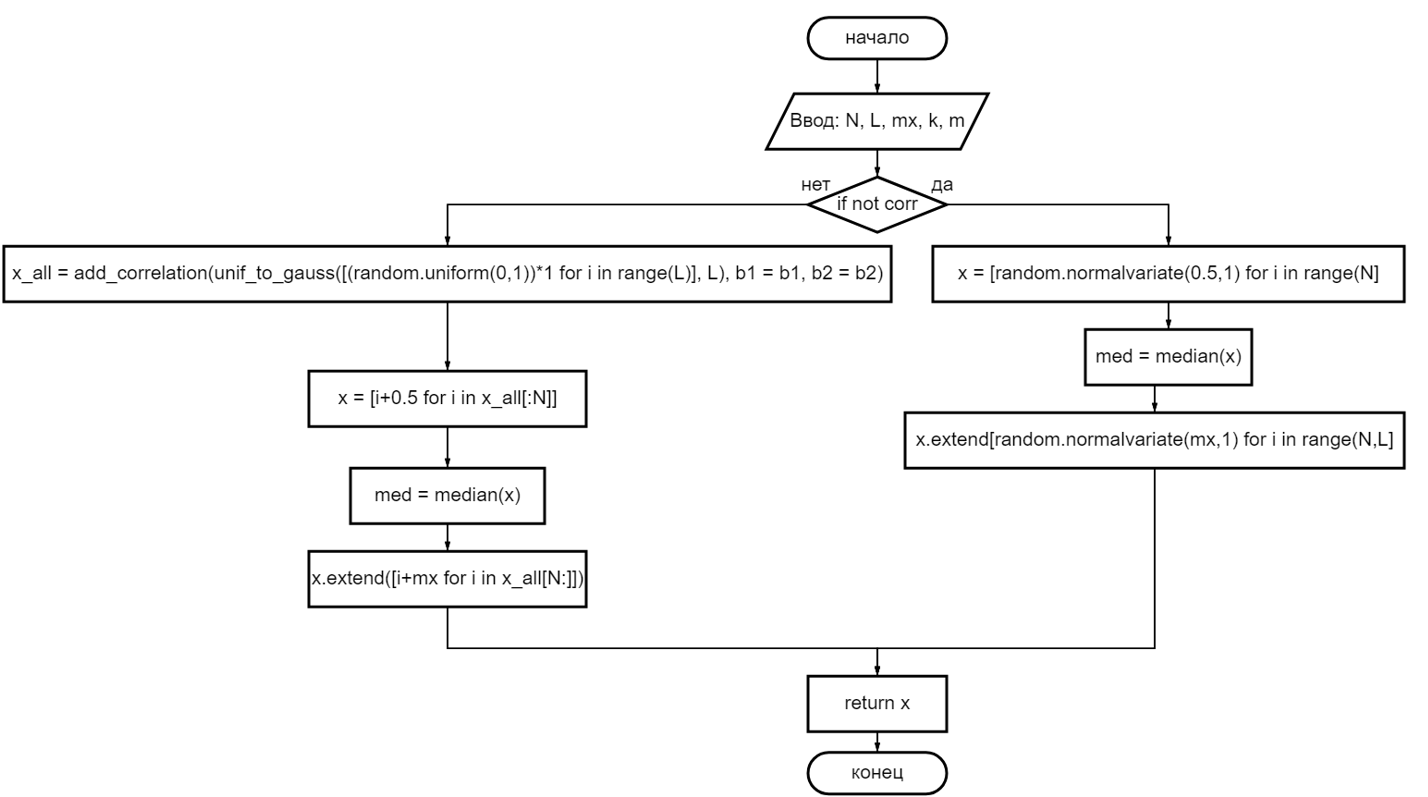
Блок-схема программы, имитирующей работу алгоритма, изображена на рисунке 1: 

Рисунок 1 - Функция генерации временного ряда

Рассмотрим функцию по определению разладки сигнала(*opr\_razl*()). В неё передаются 4 входных параметра: *x*-временной ряд длиной *L*, *med*-значение медианы временного ряда до разладки, *k*-длина серии и *N*-номер такта номинальной разладки.

Блок-схема программы, имитирующей работу алгоритма, изображена на рисунке 2:

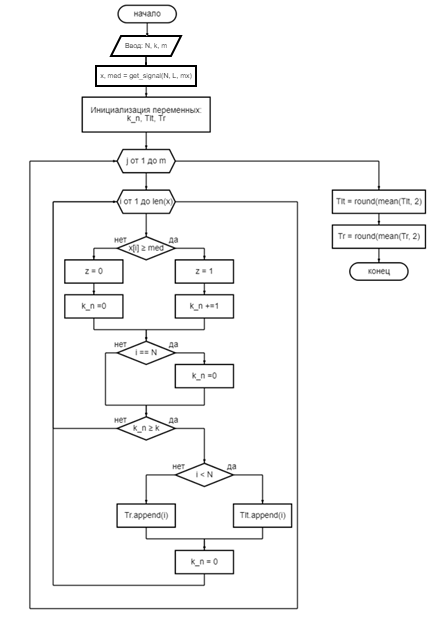


Рисунок 2 - Структура функции по определению разладки

В функции *opr\_razl()* инициализируются следующие переменные*:k\_n,Tlt,Tr*, где *k\_n* – количество идущих подряд значений, больших значения медианы процесса до разладки. Tlt-моменты времени ложных тревог и Tr-моменты времени реальных тревог. Реализация метода происходит в цикле, где начиная с 0 элемента ряда, происходит сравнение его значения с медианой. Если значение ряда окажется больше медианы, то к переменной *kn* прибавляется единица *kn = kn+1,* а если меньше, то *kn*обнуляется. Если *kn* достигло заданного значения *k*, то проверяется, не зафиксировалось ли обнаружение разладки после наступления *N*, если это так, то значение текущего шага добавляется в список *Tr*, если это условие не выполнено, то значение шага добавляется в список *Tlt*. После успешного выполнения, функция возвращает усреднённые значения моментов времени ложных и реальных тревог. Характеристики программы имитационного моделирования следующие:

Тип ЭВМ - ПК с процессорами *Pentium* или аналогичными им.

ОС - *MS Windows XP*/7/8/10/11.

Язык программирования – *Python*.

Объем программы - 18 КБ

Использованные библиотеки – *random, matplotlib, statistics, pandas, numpy, math*

## Первоначальная апробация программы

Проведём первоначальную апробацию программы. Убедимся в правильности работы разработанного алгоритма, для этого получим зависимость среднего времени между ложными тревогами от длины серии. Данная зависимость представлена в таблице 1.

Таблица 1- *Зависимость среднего времени между ложными тревогами от длины серии*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *k* | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  | 30 | 62 | 127 | 258 | 514 | 1020 | 2041 | 3213 | 3508 |

Сравним полученные результаты с уже известными эталонными результами, которые представлены в таблице 2.[3]

Таблица 2 - *Зависимость среднего времени между ложными тревогами от длины серии*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *k* | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  | 30 | 62 | 126 | 254 | 510 | 1022 | 2046 | 4094 | 8190 |

Можно увидеть, что среднее время ложных тревог практически совпадает для всех *k*, значительные расхождения начинаются с значения *k*, равного 11.

Выведем таблицу зависимости , от *k* и найдём по ней оптимальное значение *k*. Оценим эффективность алгоритма по следующей формуле *E* = /. В дальнейшем , будут измеряться в тактах.

Исходные данные:

Медиана ряда с разладкой - 0.8

Количество повторений для одного k - 1000

Номер такта номинальной разладки - 5000

Длина сигнала – 10000

Результаты представлены в таблице 3.

Таблица 3 *- Зависимость среднего времени между ложными тревогами, среднего времени запаздывания и эффективности от длины серии*

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Длина серии | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Среднее время между ложными тревогами | 62 | 125.9 | 255 | 489.8 | 950 | 1229 | - | - | - | - | - |
| Среднее время запаздывания | 10 | 14 | 19 | 24 | 32 | 40 | 54 | 67 | 84 | 109 | 130 |
| Эффективность | 6.2 | 9 | 13.4 | 20.4 | 29.7 | 30.7 | - | - | - | - | - |

По результатам, представленным в таблице можно сделать вывод о том что с увеличением длины серии, увеличивается среднее время между ложными тревогами, при длине серий 11 и выше, они вообще пропадают, также увеличивается и среднее время запаздывания.

Здесь 10-оптимальное значение длины серий, так как отношение в этом случае будет наибольшим.

График процесса для исходных данных и *k*=10 изображен на рисунке 3.

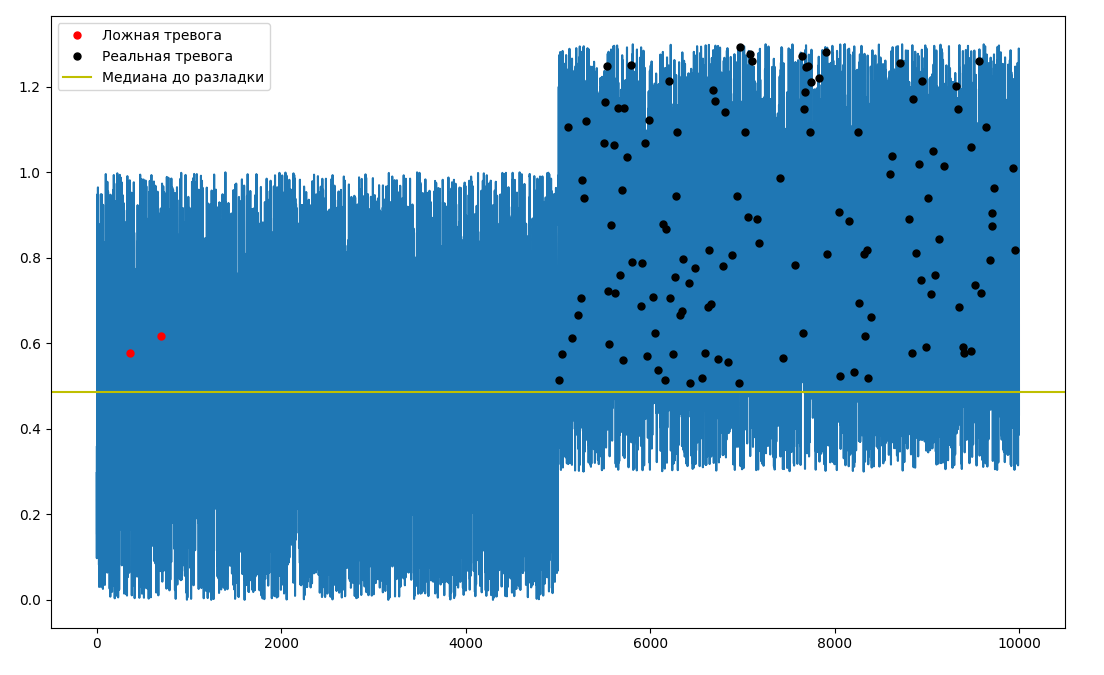


Рисунок 3 - График процесса с данными о тревогах

Здесь красными точками обозначены ложные тревоги, в данном случае их всего две, что довольно хорошо. Черными точками обозначены моменты реальных тревог, видно, что первая реальная тревога наступает почти сразу после разладки, также алгоритм продолжает довольно часто сигнализировать пользователя о том, что произошла разладка.

Изменим медиану процесса с разладкой с 0.8 на 1 и сравним результаты моделирования.

Исходные данные:

Медиана ряда с разладкой - 1.0

Количество повторений для одного k - 1000

Номер такта номинальной разладки - 5000

Длина сигнала – 10000

Результаты моделирования представлены в таблице 4.

Таблица 4 *- Зависимость среднего времени между ложными тревогами, среднего времени запаздывания и эффективности от длины серии*

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Длина серии | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Среднее время между ложными тревогами | 62.2 | 125.9 | 256 | 505.7 | 924 | 1254 | - | - | - | - | - |
| Среднее время запаздывания | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Эффективность | 12,4 | 21 | 36,6 | 63,2 | 102,7 | 125,4 | - | - | - | - | - |

По результатам, представленным в таблице видно, что в сравнении с предыдущими результатами, среднее время между ложными тревогами почти не поменялось, а среднее время запаздывания, наоборот, изменилось очень сильно и стало равным длине серии.

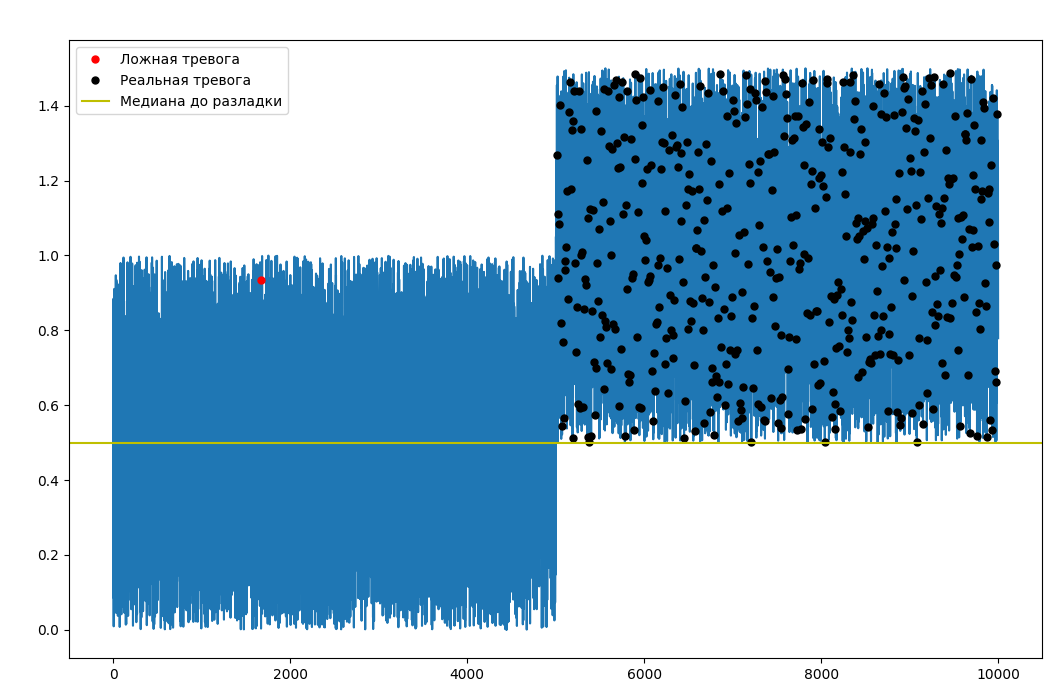
График процесса для исходных данных и *k*=10 изображен на рисунке 4.

Рисунок 4 - График процесса с данными о тревогах

Наблюдается всего одна ложная тревога. Видно, что сигналов о реальной тревоге в несколько раз больше, чем в прошлом случае.

Для сопоставления с другими методами, величина разладки характеризуется показателем *d*. Для случая разладки по математическому ожиданию, стандартный набор включает следующие значения *dm*: 0,25;0,5;1,0; 1,5;2,0;2,5, здесь  
, где *m1 и m0* – значения математических ожиданий временного ряда до и после разладки соответственно.

Определим зависимость среднего времени запаздывания  от длины серии *k,* при различных значениях разладки. Данная зависимость приведена в таблице 5.

Таблица 5 - *Зависимость среднего времени запаздывания  от длины серии для различных значений dm*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *k* | *dm*= 0.25 | *dm* = 0.5 | *dm* = 1.0 | *dm* = 1.5 | *dm* = 2.0 | *dm* = 2.5 |
| 4 | 15 | 16,75 | 11,71 | 6,09 | 4,54 | 4,09 |
| 5 | 33 | 26,25 | 19,53 | 7,95 | 5,98 | 5,32 |
| 6 | 50 | 49,43 | 28 | 11,62 | 8,09 | 6,42 |
| 7 | 76 | 101,27 | 35,39 | 14,87 | 9,84 | 7,56 |
| 8 | 144 | 152,3 | 69,08 | 19,46 | 10,6 | 8,57 |
| 9 | 265 | 258,35 | 85,76 | 24,27 | 12,69 | 10,06 |
| 10 | 376 | 402,79 | 114,6 | 27,42 | 14,24 | 11,44 |
| 11 | 680 | 699,12 | 191,99 | 33,69 | 15,9 | 12,44 |
| 12 | 1320 | 966,45 | 304,81 | 43,5 | 20,75 | 13,96 |

По полученным результатам можно увидеть, что с увеличением *dm* для всех *k*, среднее время запаздывания уменьшается, алгоритм начинает обнаруживать разладку быстрее. При *dm* = 2.5, среднее время запаздывания приближается к соответствующему значению *k.*

Определим зависимость эффективности от длины серии *k,* при различных значениях разладки. Данная зависимость приведена в таблице 6.

Таблица 6 - *Зависимость эффективности E от длины серии для различных значений dm*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *k* | *dm*= 0.25 | *dm* = 0.5 | *dm* = 1.0 | *dm* = 1.5 | *dm* = 2.0 | *dm* = 2.5 |
| 4 | 1,91 | 3,02 | 4,91 | 6,26 | 6,82 | 7,42 |
| 5 | 2,32 | 3,93 | 7,83 | 9,28 | 11,13 | 12,29 |
| 6 | 2,61 | 4,59 | 11,05 | 15,76 | 19,45 | 20,64 |
| 7 | 3,16 | 5,34 | 15,98 | 26,38 | 33,30 | 35,78 |
| 8 | 3,34 | 7,78 | 31,8 | 43,91 | 59,46 | 57,84 |
| 9 | 4,21 | 10,53 | 44,96 | 74,39 | 105,26 | 110,55 |
| 10 | 4,80 | 14,48 | 76,93 | 141,83 | 184,07 | 208,27 |
| 11 | 5,18 | 22,88 | 119,43 | 218,57 | 344,05 | 344,34 |
| 12 | 7,60 | 30,66 | 222,1 | 411,22 | 611,66 | 727,42 |

По полученным результатам можно увидеть, что с увеличением *dm* для всех *k*, значение эффективности увеличивается, кроме того, наибольшая эффективность достигается при *k = 12*. Зависимость показателя эффективности от длины серии для разных *dm* представлена на рисунке 5.

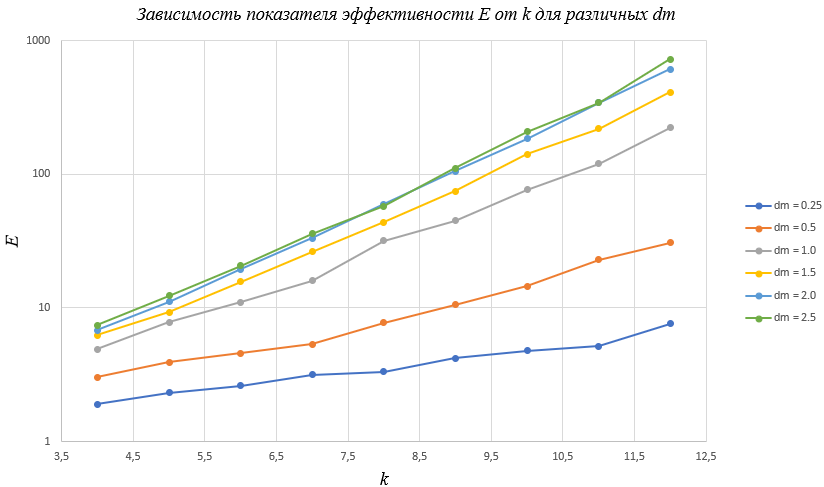


Рисунок 5 - Зависимость показателя эффективности *E* от *k* для различных *dm*

По рисунку видно, что эффективность возрастает с увеличением *k*, также она зависит от показателя *dm*, при его увеличении, эффективность возрастает, но затем её рост замедляется.

## Определение длины серий, по заданному среднему времени между ложными тревогами

Выполним другую задачу: по заданному , определим значение *k*, при котором достигается наиболее близкое к заданному, так как *k* изменяется только в целом диапазоне чисел.

Описание алгоритма представлено ниже:

Исходными данными, вводимыми пользователем, являются номер такта, на котором будет введена разладка *N*, общее количество значений временного ряда *L*, медиана временного ряда в состоянии “разладка” *mx*, желаемое время между ложными тревогами и количество запусков *m*. Также программа позволяет выбрать режим работы с некоррелированным или коррелированным сигналом.

Во-первых происходит вызов функции по определению разладки(*opr\_razl*) с параметром *k=1*, данная функция возвращает список из двух значений:среднего времени между ложными тревогами и среднего времени запаздывания. Вводится переменная *delta*, равная модулю разности между введённым пользователем и средним временем между ложными тревогами, которое определил алгоритм.

Затем происходит запуск бесконечного цикла *while(1)*, который выполнятся пока не будет выполнено условие *delta\_next > delta*. В цикле *k* увеличивается на 1, затем происходит вызов функции по определению разладки и расчёт *delta\_next*, после этого выполняется проверка условия на выход из цикла, если условие не выполнено, то *delta* приравнивается к *delta\_next* и происходит переход к следующей итерации.

По окончанию цикла пользователю выводится определённое *k*. Блок-схема данной программы представлена на рисунке 6.

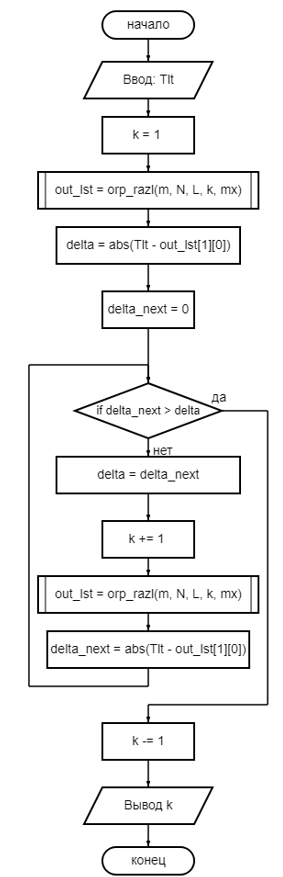


Рисунок 6- Блок-схема программы

Исходные данные:

Заданное среднее время между ложными тревогами - 500

Медиана ряда с разладкой - 0.8

Количество повторений для одного *k* - 1000

Номер такта номинальной разладки - 5000

Длина сигнала - 10000

Результаты моделирования:

Среднее время между ложными тревогами – 494

Среднее время запаздывания - 118

Определённое значение *k* – 8

Результаты моделирования при *k=8* представлены на рисунке 7.

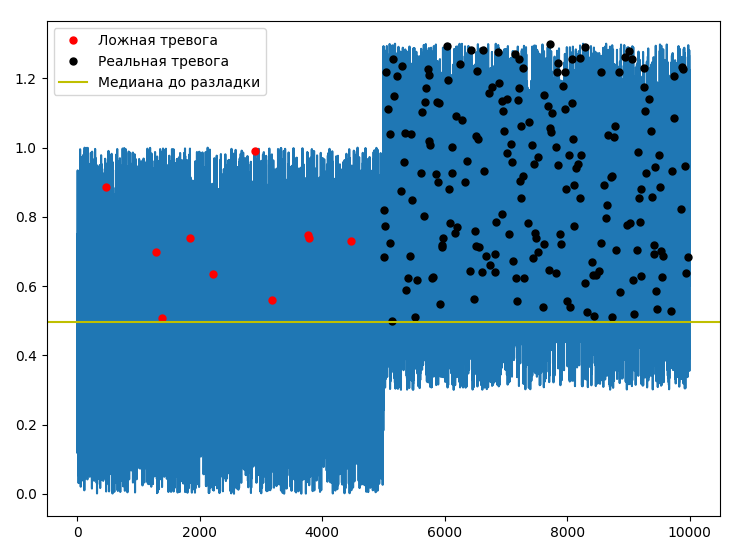


Рисунок 7 - График процесса с данными о тревогах

На вход программы подано необходимое среднее время между ложными тревогами: . Алгоритм определил значение длины серии *k*=8, при котором достигается наиболее близкое к заданному, и вывел график с результатами моделирования. Из результатов видно, что на практике среднее время между ложными тревогами получилось примерно равным 494, что довольно-таки хорошо, так как *k* может изменяться только в целом диапазоне чисел, поэтому сложно добиться высокой точности попадания в заданное.

## Выводы

В ходе данного исследования по проблематике обнаружения разладки временных рядов, реализована программа имитационного моделирования, основанная на теории случайных блужданий.

В ходе апробации получены зависимости и , а также *E* от длины серии *k* при различных значениях разладки. По полученным данным выбрано оптимальное значение *k*. Выполнена задача определения оптимального значения длины серии *k* для заданного значения среднего времени между ложными тревогами.

# ИЗУЧЕНИЕ И ПОДБОР МЕТОДОВ ЗАДАНИЯ КОРРЕЛЯЦИИ ДЛЯ ГАУССОВСКОГО ПРОЦЕССА.

Гауссовский процесс - это вероятностный процесс, в котором любая конечная коллекция случайных переменных имеет многомерное нормальное распределение. Он широко применяется в моделировании случайных функций и аппроксимации данных.

Корреляционная функция (или функция ковариации) играет ключевую роль в описании зависимости между значениями гауссовского процесса в разных точках. Она определяет степень связи или сходства между случайными переменными в разных местах. Выбор правильной корреляционной функции влияет на свойства и поведение гауссовского процесса.

Подбор методов задания корреляции для гауссовского процесса зависит от конкретной задачи и требований моделирования. Некоторые из наиболее распространенных методов включают следующие корреляционные функции:

1. Экспоненциальная корреляционная функция: Одна из наиболее простых и распространенных форм корреляционной функции. Она определяется параметром масштаба и описывает экспоненциальное затухание корреляции с увеличением расстояния между точками.

2. Гауссовская корреляционная функция: Используется в случаях, когда зависимость между точками может быть описана гауссовым распределением. Она имеет параметры масштаба и ширины и обладает гладкими свойствами.

3. Материал-корреляционная функция: Представляет собой корреляцию, которая уменьшается с расстоянием, но с медленной скоростью, что позволяет моделировать более длинные зависимости.

4. Квадратичная корреляционная функция: Используется для моделирования нерегулярных и неоднородных процессов. Она обладает большей гибкостью и позволяет учитывать нелинейные зависимости.

Выбор конкретного метода задания корреляции зависит от природы данных, предметной области и требований моделирования. Часто требуется проводить эксперименты и анализировать поведение гауссовского процесса с разными корреляционными функциями, чтобы найти наиболее подходящую для конкретной задачи.

Для задания корреляции гауссовского процесса выбрана модель авторегрессии второго порядка АР(2). Модель авторегрессии второго порядка представляет собой стохастическую модель, в которой каждое значение процесса зависит от двух предыдущих значений. Она является одной из моделей временных рядов, которая учитывает автокорреляцию и зависимость между значениями процесса на разных временных отрезках.

В модели авторегрессии второго порядка, текущее значение процесса *y(t)* выражается через два предыдущих значения *y(t-1)* и *y(t-2)* с определенными коэффициентами авторегрессии:

*y(t) = ϕ₁ \* y(t-1) + ϕ₂ \* y(t-2) + ε(t),*

где *ϕ₁* и *ϕ₂* - коэффициенты авторегрессии, определяющие веса предыдущих значений,

*ε(t)* - случайная ошибка или шум с нулевым средним и конечной дисперсией.

Модель авторегрессии второго порядка позволяет учесть зависимость между текущим значением процесса и двумя предыдущими значениями. Коэффициенты авторегрессии *ϕ₁* и *ϕ₂* определяют влияние предыдущих значений на текущее значение процесса. Они могут быть положительными или отрицательными, что указывает на положительную или отрицательную корреляцию между значениями процесса.

Основными характеристиками модели авторегрессии второго порядка являются способность учесть зависимость между предыдущими значениями, возможность предсказания будущих значений и анализ изменений в поведении процесса. Однако следует отметить, что выбор модели всегда зависит от конкретной задачи, данных и предполагаемой структуры временного ряда.

Алгоритм для получения коррелированного временного ряда с использованием АР(2) модели:

1. На вход поступает временной ряд с нормальным распределением и два параметра *b1* и *b2*.
2. Ряд последовательно прогоняется *i(0,…..,L)* тактов через алгоритм по вводу корреляции.
   1. На первой итерации задается значение *e0*= *x0*
   2. Новое значение ряда вычисляется согласно данной формуле
3. Проводится стандатизация временного ряда, то есть изменение масштаба распределения значений так, чтобы среднее значение наблюдаемых значений было 0, а стандартное отклонение - 1.
4. На выходе получаем коррелированный временной ряд

Блок-схема получения коррелированных значений на основе модели АР(1) изображена на рисунке 8:

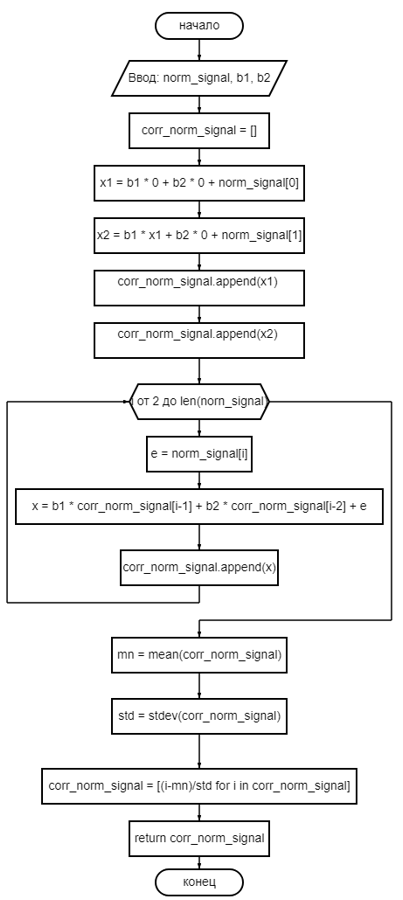


Рисунок 8 - Блок-схема получения коррелированных значений на основе модели АР(1)

# ИССЛЕДОВАНИЕ РАБОТЫ МЕТОДА СЛУЧАЙНЫХ БЛУЖДАНИЙ НА БИНАРНОМ КОРРЕЛИРОВАННОМ СИГНАЛЕ В ОТСУТСТВИИ РАЗЛАДКИ

Разработана программа для получения коррелированного бинарного сигнала.Настраиваемыми параметрами являются *b1* и *b2*.

По корреляционным функциям определяются характеристики:

* *τмк* – определяется как абсцисса при вхождении автокорреляционной функции в интервал [-0.05, +0.05].
* *τк* – определяется как сумма значений автокорреляционной функции, позволяет оценить степень коррелированности между значениями временного ряда.

## Сравнение работы метода для сигнала без корреляции и для коррелированного сигнала

Все исследования проводятся при длине серий *k*=5, длине сигнала 10000. Разладка не вводилась.

**Исследование сигнала без ввода корреляции**

График временного ряда с отметками о ложных тревогах представлен на рисунке 9:

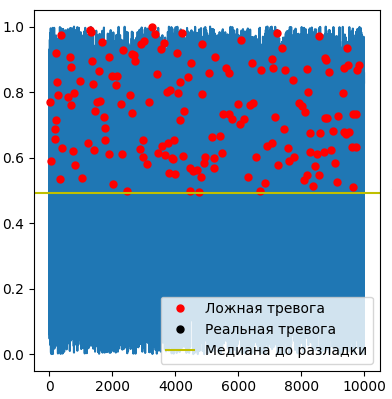


Рисунок 9 - График процесса с данными о тревогах

Результаты моделирования:

Множество ложных тревог

Среднее время между ложными тревогами – 65

**Исследование коррелированного сигнала**

**Параметры возьмем следующие:**

*b1* = 0.45, *b2* = 0.45

Корреляционная функция сигнала представлена на рисунке 10:

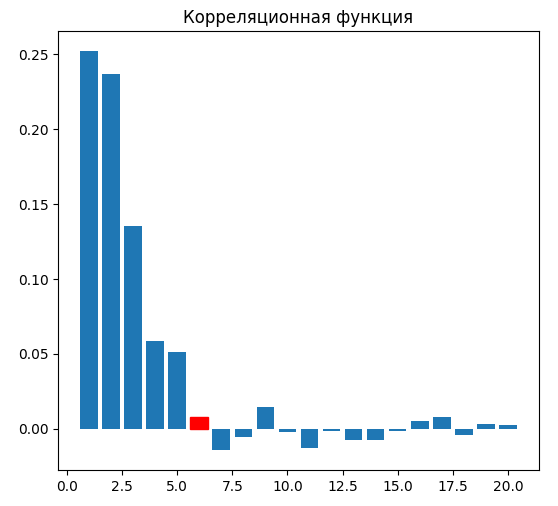


Рисунок 10 - Корреляционная функция

На графике корреляционной функции красным цветом отмечено начало соблюдения вхождения в интервал [-0.05, +0.05].

Результаты моделирования:

Множество ложных тревог

Среднее время между ложными тревогами - 183

*τмк* = 7

*τк* = 1.55

**Возьмем другие параметры:**

*b1* = 0.35, *b2* = 0.35

Корреляционная функция сигнала представлена на рисунке 11:

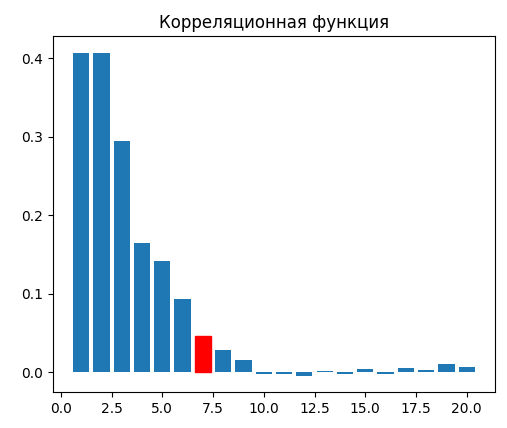


Рисунок 11 - Корреляционная функция

Результаты моделирования:

Множество ложных тревог

Среднее время между ложными тревогами - 103

*τмк* = 6

*τк* = 0.75

## Исследование работы алгоритма при коррелированном бинарном сигнале

Исследуем зависимость τмк и τк от параметров *b1* и *b2*, результаты преставлены в таблице 7:

Таблица 7 - *Зависимость τмк и τк от параметров b1 и b2.*

|  |  |  |
| --- | --- | --- |
| Параметры модели авторегрессии | *τмк* | *τк* |
| *b1* = 0.45, *b2* = 0.45 | 7 | 1.55 |
| *b1* = 0.35, *b2* = 0.35 | 6 | 0.75 |
| *b1*= 0.25, *b2* = 0.25 | 3 | 0.3 |
| *b1* = 0.2, *b2* = 0.2 | 3 | 0.2 |

Посмотрим, как меняется среднее время между ложными тревогами при различных значениях коэффициентов в зависимости от числа серий, результаты представлены в таблице 8.

Длина сигнала 5000, число усреднений 1000

Таблица 8 - *Зависимость сренего времени между ложными тревогами от параметров b1 и b2 и длины серии.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Параметры модели авторегрессии | Длина серии | | | | | |
| 5 | 6 | 7 | 8 | 9 | 10 |
| *b1* = 0.45, *b2* = 0.45 | 190,38 | 412,34 | 846,73 | 1589,19 | 2811,26 | - |
| *b1* = 0.35, *b2* = 0.35 | 92,14 | 191,14 | 380,97 | 686,47 | 1137,92 | - |
| *b1*= 0.25, *b2* = 0.25 | 72,46 | 147,66 | 295,88 | 565,52 | 1052,62 | 1243,11 |
| *b1* = 0.2, *b2* = 0.2 | 68,84 | 138,2 | 275,29 | 575,53 | 1115,31 | 1336,51 |

Как видно из таблицы, эффективность работы алгоритма сильно зависит от параметров *b*1 и *b*2 и, соответственно, от величины *τк*, при уменьшении *τк* ложные тревоги начинаю появляться чаще, следовательно время между ложными тревогами падает.

## Выводы

В ходе исследования работы метода случайных блужданий на бинарном коррелированном сигнале в отсутствии разладки, выяснено, что эффективность работы метода случайных блужданий сильно зависит от параметров *b1* и *b2*. Так как данные параметры определяют веса предыдущих значений, от них будет зависеть вид гистограммы анализируемого процесса. Следовательно будет менятся эффективность работы алгоритма.

# ИССЛЕДОВАНИЕ РАБОТЫ МЕТОДА СЛУЧАЙНЫХ БЛУЖДАНИЙ НА “ГАУССОВСКОМ” КОРРЕЛИРОВАННОМ СИГНАЛЕ В В ПРИСУТСТВИИ РАЗЛАДКИ

Разработана программа для получения коррелированного гауссовского сигнала.Настраиваемыми параметрами являются *b1* и *b2*.

По корреляционным функциям определяются характеристики:

* *τмк* – определяется как абсцисса при вхождении автокорреляционной функции в интервал [-0.05, +0.05].
* *τк* – определяется как сумма значений автокорреляционной функции, позволяет оценить степень коррелированности между значениями временного ряда.

## Сравнение работы метода для сигнала без корреляции и для коррелированного сигнала:

Все исследования проводятся при числе серий *k*=10, длине сигнала 10000. Разладка начинается с 5000 такта.

Медиана до разладки – 0.5 , Медиана после разладки – 0.7.

***Исследования для сигнала без корреляции***

График временного ряда с отметками о ложных и реальных тревогах представлен на рисунке 12:

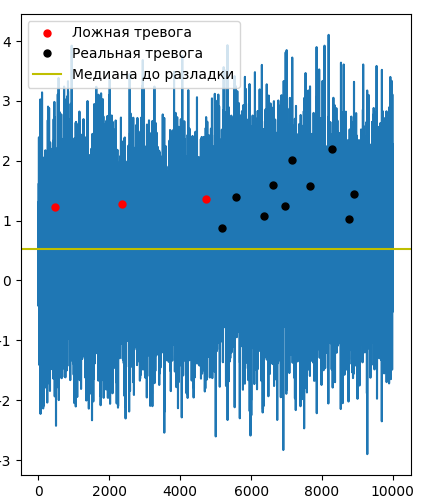


Рисунок 12 - График процесса с данными о тревогах

Результаты моделирования:

Моменты времени ложных тревог - [455, 2362, 4740]

Среднее время между ложными тревогами – 2143

Среднее время запаздывания - 187

Наблюдаем всего три ложные тревоги и небольшое количество реальных тревог. Время запаздывания довольно мало, что хорошо.

***Исследование для сигнала с корреляцией***

**Параметры возьмем следующие:**

*b1* = 0.35, *b2* = 0.35

Корреляционная функция сигнала представлена на рисунке 13:

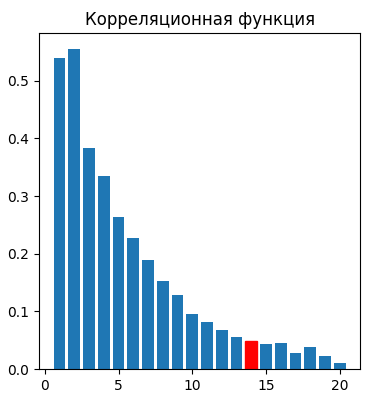


Рисунок 13 - Корреляционная функция

На графике корреляционной функции красным цветом отмечено начало соблюдения вхождения в интервал [-0.05, +0.05].

Результаты моделирования:

Множество ложных тревог

Среднее время между ложными тревогами - 84

Среднее время запаздывания - 67

*τмк* = 14

*τк* = 3.12

**Возьмем другие параметры:**

*b1* = 0.2, *b2* = 0.2

Корреляционная функция сигнала представлена на рисунке 14:

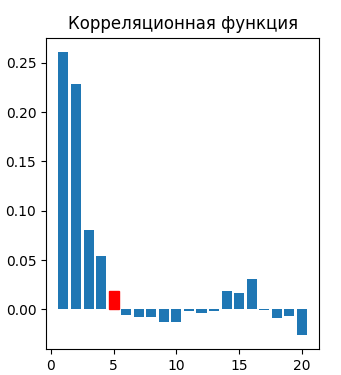


Рисунок 14 - Корреляционная функция

Результаты моделирования:

Множество ложных тревог

Среднее время между ложными тревогами - 306

Среднее время запаздывания - 138

*τмк* = 5

*τк* = 0.64

Исследуем зависимость τмк и τк от параметров *b1* и *b2*, результаты преставлены в таблице 9:

Таблица 9 - *Зависимость τмк и τк от параметров b1 и b2.*

|  |  |  |
| --- | --- | --- |
| Параметры модели авторегрессии | *τмк* | *τк* |
| *b1* = 0.45, *b2* = 0.45 | 16 | 4.18 |
| *b1* = 0.35, *b2* = 0.35 | 14 | 3.2 |
| *b1*= 0.25, *b2* = 0.25 | 6 | 1.07 |
| *b1* = 0.2, *b2* = 0.2 | 5 | 0.64 |

Посмотрим, как меняется среднее время запаздывания при различных значениях коэффициентов в зависимости от числа серий, результаты представлены в таблице 10.

Длина сигнала 10000, число усреднений 1000

Таблица 10 - *Зависимость сренего времени запаздывания от параметров b1 и b2 и длины серии.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Параметры модели авторегрессии | Длина серии | | | | | |
| 5 | 6 | 7 | 8 | 9 | 10 |
| *b1* = 0.45, *b2* = 0.45 | 16,67 | 21,29 | 25 | 32,15 | 38,31 | 52,8 |
| *b1* = 0.35, *b2* = 0.35 | 20,63 | 22,74 | 32,2 | 36,48 | 48,24 | 67,28 |
| *b1*= 0.25, *b2* = 0.25 | 24,73 | 32,31 | 39,2 | 50,37 | 84,31 | 103,81 |
| *b1* = 0.2, *b2* = 0.2 | 24,87 | 33,46 | 50,83 | 64,6 | 101,3 | 154,87 |

Как видно из таблицы, эффективность работы алгоритма сильно зависит от параметров *b1* и *b2* и, соответственно, от величины *τк*, при уменьшении *τк* время запаздывания запаздывания увеличивается, следовательно количество тревог уменьшается, что не есть хорошо.

Определим зависимость эффективности от длины серии *k,* при различных значениях разладки и различных параметрах *b1*, *b2*.

Зависимость при параметрах *b1* = 0.40, *b2* = 0.40,приведена в таблице 11.

Таблица 11 - *Зависимость эффективности E от длины серии для различных значений dm*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *k* | *dm*= 0.25 | *dm* = 0.5 | *dm* = 1.0 | *dm* = 1.5 | *dm* = 2.0 | *dm* = 2.5 |
| 4 | 0,93 | 1,33 | 1,9 | 2,68 | 2,92 | 3,05 |
| 5 | 0,89 | 1,36 | 2,11 | 2,91 | 3,38 | 3,37 |
| 6 | 1,04 | 1,34 | 2,07 | 2,97 | 3,47 | 3,75 |
| 7 | 0,96 | 1,71 | 2,51 | 3,53 | 3,66 | 4,13 |
| 8 | 1,14 | 1,73 | 2,52 | 3,65 | 4 | 4,5 |
| 9 | 1,33 | 1,6 | 3,18 | 3,76 | 4,89 | 5,01 |
| 10 | 1,11 | 1,85 | 2,88 | 4,44 | 4,75 | 5,54 |
| 11 | 1,21 | 1,89 | 3,24 | 3,83 | 5,61 | 6,19 |
| 12 | 1,35 | 1,9 | 3,45 | 5 | 6,18 | 6,66 |

Зависимость при параметрах *b1* = 0.35, *b2* = 0.35,приведена в таблице 12.

Таблица 12 - *Зависимость эффективности E от длины серии для различных значений dm*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *k* | *dm*= 0.25 | *dm* = 0.5 | *dm* = 1.0 | *dm* = 1.5 | *dm* = 2.0 | *dm* = 2.5 |
| 4 | 1,13 | 1,31 | 2,16 | 2,95 | 3,44 | 3,54 |
| 5 | 1,35 | 1,55 | 2,58 | 3,21 | 3,79 | 3,98 |
| 6 | 1,35 | 1,58 | 2,93 | 3,8 | 4,13 | 4,73 |
| 7 | 1,02 | 1,68 | 2,95 | 4,63 | 4,98 | 5,38 |
| 8 | 1,61 | 2,07 | 3,53 | 4,96 | 5,91 | 6,12 |
| 9 | 1,47 | 2,29 | 3,79 | 5,27 | 6,67 | 6,93 |
| 10 | 1,69 | 2,36 | 4,54 | 5,34 | 7,81 | 8,16 |
| 11 | 1,5 | 2,49 | 4,95 | 7,06 | 8,89 | 9,52 |
| 12 | 1,68 | 2,87 | 5,27 | 8,38 | 10,15 | 10,8 |

Зависимость при параметрах *b1* = 0.25, *b2* = 0.25,приведена в таблице 13.

Таблица 13 - *Зависимость эффективности E от длины серии для различных значений dm*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *k* | *dm*= 0.25 | *dm* = 0.5 | *dm* = 1.0 | *dm* = 1.5 | *dm* = 2.0 | *dm* = 2.5 |
| 4 | 1,35 | 1,84 | 3,05 | 3,8 | 4,22 | 4,41 |
| 5 | 1,52 | 2,34 | 3,63 | 4,74 | 5,33 | 5,5 |
| 6 | 1,59 | 2,52 | 4,18 | 5,47 | 6,67 | 6,96 |
| 7 | 1,82 | 2,49 | 4,83 | 7,21 | 8,24 | 8,85 |
| 8 | 1,67 | 3,03 | 5,59 | 8,72 | 10,39 | 11,18 |
| 9 | 1,79 | 3,41 | 6,88 | 10,88 | 13,59 | 14,61 |
| 10 | 2,23 | 4,51 | 8,29 | 13,17 | 17,2 | 18,53 |
| 11 | 2,23 | 4,06 | 10,9 | 17,39 | 21,94 | 24,22 |
| 12 | 2,3 | 5,64 | 11,63 | 23,55 | 30,69 | 34,38 |

Зависимость при параметрах *b1* = 0.2, *b2* = 0.2,приведена в таблице 14.

Таблица 14 - *Зависимость эффективности E от длины серии для различных значений dm*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *k* | *dm*= 0.25 | *dm* = 0.5 | *dm* = 1.0 | *dm* = 1.5 | *dm* = 2.0 | *dm* = 2.5 |
| 4 | 1,48 | 1,91 | 3,46 | 4,1 | 4,56 | 4,8 |
| 5 | 1,53 | 2,31 | 4,39 | 5,62 | 6,07 | 6,25 |
| 6 | 1,81 | 2,84 | 4,6 | 6,86 | 8,03 | 8,71 |
| 7 | 1,95 | 2,99 | 6,42 | 9,04 | 11,14 | 11,57 |
| 8 | 2,13 | 3,57 | 7,87 | 12,03 | 14,69 | 15,62 |
| 9 | 2,29 | 4,99 | 10,07 | 17,26 | 20,1 | 21,07 |
| 10 | 3,15 | 5,07 | 13,68 | 22,19 | 26,64 | 29,42 |
| 11 | 2,66 | 5,26 | 17,88 | 31,35 | 39,97 | 39,56 |
| 12 | 3,01 | 7,2 | 22,06 | 36,78 | 54,74 | 60,62 |

Для последнего случая построим зависимость показателя эффективности от длины серии для разных *dm*, данная зависимость представлена на рисунке 15.

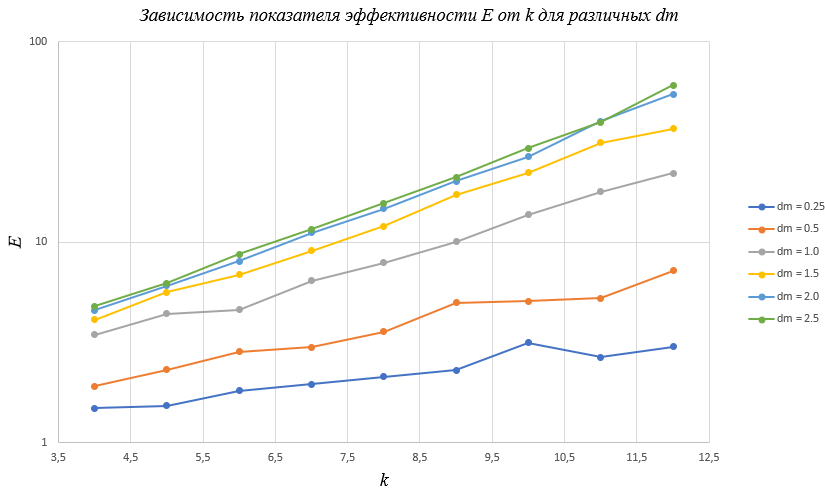


Рисунок 15 - Зависимость показателя эффективности *E* от *k* для различных *dm*

По рисунку видно, что эффективность возрастает с увеличением *k*, также она зависит от показателя *dm*, при его увеличении, эффективность возрастает, но затем её рост замедляется.

Также построим зависимость показателя эффективности от параметров *b1 и b2* для различных *dm,* все зависимости построены при *k*=10. данная зависимость представлена на рисунке 16.

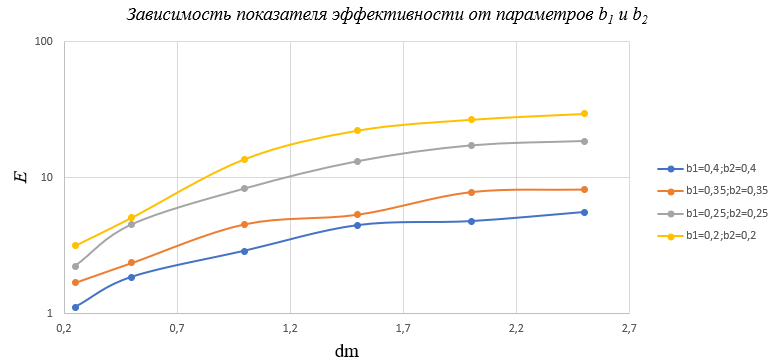


Рисунок 16 Зависимость показателя эффективности *E* от параметров *b1 и b2* для различных *dm*

По рисунку видно, что эффективность возрастает с увеличением *dm*, также она зависит от значений параметров *b1 и b2*, при уменьшенни параметров, эффективность возрастает.

## Определение длины серий, по заданному среднему времени между ложными тревогами

Теперь выполним другую задачу: по заданному , определим наиболее значение *k*, при котором достигается наиболее близкое к заданному.

Исходные данные:

Заданное среднее время между ложными тревогами - 500

Медиана ряда с разладкой - 0.7

Количество повторений для одного *k* - 1000

Номер такта номинальной разладки - 5000

Длина сигнала – 10000

***Результаты моделирования для некоррелированного сигнала:***

Определённое значение *k* - 8

Среднее время между ложными тревогами - 480

Время запаздывания – 43

График временного ряда с отметками о ложных и реальных тревогах представлен на рисунке 17:

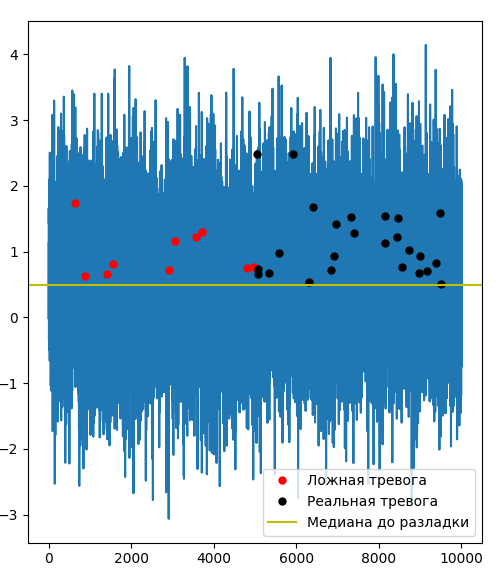


Рисунок 17 - График процесса с данными о тревогах

На вход программы подано необходимое среднее время между ложными тревогами: . Алгоритм определил значение длины серии *k*=8, при котором достигается наиболее близкое к заданному, и вывел график с результатами моделирования. Из результатов видно, что на практике среднее время между ложными тревогами получилось примерно равным 480, что довольно-таки хорошо, так как *k* может изменяться только в целом диапазоне чисел, поэтому сложно добиться высокой точности попадания в заданное.

***Результаты моделирования для коррелированного сигнала:***

Возьмём следующие значения параметров

*b1* = 0.25, *b2* = 0.25

Определённое значение *k* – 13

Среднее время между ложными тревогами - 558

Время запаздывания - 256

*τмк* = 6

*τк* = 1.04

График временного ряда с отметками о ложных и реальных тревогах представлен на рисунке 18:

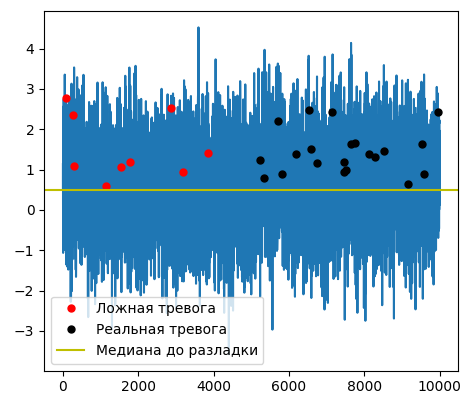


Рисунок 18 - График процесса с данными о тревогах

На вход программы подано необходимое среднее время между ложными тревогами: . Алгоритм определил значение длины серии *k*=13, при котором достигается наиболее близкое к заданному, и вывел график с результатами моделирования. Из результатов видно, что на практике среднее время между ложными тревогами получилось примерно равным 558, что довольно-таки хорошо, так как *k* может изменяться только в целом диапазоне чисел, поэтому сложно добиться высокой точности попадания в заданное.

## Выводы

В ходе исследования работы метода случайных блужданий на “гауссовском” коррелированном сигнале в присутствии разладки, выяснено, что эффективность работы метода случайных блужданий также сильно зависит от параметров *b*1 и *b*2. При уменьшении *τк* эффективность возрастает.

В сравнении с некоррелированным сигналом, алгоритм показывает себя хуже.

В программе по определению *k* по заданному , при коррелированном временном ряде, алгоритм показывает себя хуже, время запаздывания в несколько раз больше, нежели при моделировании с некоррелированным сигналом.

# ЗАКЛЮЧЕНИЕ

В ходе данной работы рассмотрена литература по проблематике обнаружения «разладки» временных рядов; составлен краткий обзор известных методов обнаружения разладки.

Кроме того, поставлена задача исследования алгоритма, основанного на теории случайных блужданий, предназначенного для выявления разладки гауссовского случайного процесса по математическому ожиданию (с помощью метода имитационного моделирования).

Разработана программа имитационного моделирования алгоритма, с помощью которой получены его основные характеристики, необходимые для синтеза соответствующей контролирующей процедуры.

По итогам данной работы, полученные результаты позволяют найти оптимальное значение параметров для того, чтобы максимизировать эффективность алгоритма.

Исследовалось влияние корреляции на эффективность работы метода, если при анализе некоррелировнного гауссовского временного ряда на эффективность влияют только длина серии *k* и величина разладки, то при коррелированном сигнале на эффективность в значительной степени влияет степень корреляции *τ*к, но в любом случае эффективность работы метода при коррелированном сигнале значительно меньше эффективности метода при некоррелированном сигнале.

Таким образом, задание ВКР выполнено в полном объеме.

# СПИСОК ЛИТЕРАТУРЫ

|  |
| --- |
| **1. Феллер У.** Введение в теорию вероятностей и ее приложения. Том 1. |
| (An Introduction to Probability Theory and its Applications. Volume I), М.: Изд- |
| во «Мир», 1970. 499 с. |
| **2.** **Филаретов Г.Ф., Червова А.А., Бучаала З.** Последовательный |
| непараметрический алгоритм обнаружения разладки временнóго ряда // |
| Датчики и системы, 2020, №1, с.9 – 16. |
| **3.** **Филаретов Г.Ф., Червова А.А., Бучаала З.** Непараметрический |
| метод обнаружения разладки временнóго ряда c использованием механизма |
| случайных блужданий».// Известия ЮФУ. Технические науки. 2020, №4, с.107-117. |
| **4.** **Филаретов Г.Ф., Бучаала З.** Разработка алгоритмов обнаружения |
| разладки временных рядов на основе непараметрических критериев проверки |
| гипотез // Вестник МЭИ. 2021, №3. – с. 67 – 77. |

# *Приложение*

## Листинг программы

import random

import matplotlib.pyplot as plt

from statistics import mean, median, stdev

from gen\_of\_corr import get\_corr\_row

import pandas as pd

import numpy as np

import math

def add\_correlation(norm\_signal, b1, b2):

    """

    Введение корреляции в гауссовский сигнал

    """

    corr\_norm\_signal = []

    x1 = b1 \* 0 + b2 \* 0 + norm\_signal[0]

    x2 = b1 \* x1 + b2 \* 0 + norm\_signal[1]

    corr\_norm\_signal.append(x1)

    corr\_norm\_signal.append(x2)

    for i in range(2, len(norm\_signal)):

        e = norm\_signal[i]

        x = b1 \* corr\_norm\_signal[i-1] + b2 \* corr\_norm\_signal[i-2] + e

        corr\_norm\_signal.append(x)

    # corr\_norm\_signal = corr\_norm\_signal[2:]

    # стандратизация

    mn = mean(corr\_norm\_signal)

    std = stdev(corr\_norm\_signal)

    corr\_norm\_signal = ((np.array(corr\_norm\_signal) - mn) / std).tolist()

    return corr\_norm\_signal

def mean\_of\_razl(m, N, L, k, mx = 0, out = 0):

    """

    Усреднение по m значениям

    """

    lst = [[], [], [], []]

    res\_list = [[], [], [], []]

    for i in range(m):

        out\_lst = razl(N, L, k, mx = mx, out = 0)

        res\_list[0] = out\_lst[0]

        res\_list[1].append(out\_lst[1])

        res\_list[2].append(out\_lst[2])

        res\_list[3].append(out\_lst[3])

    if res\_list[1].count("-") <= (len(res\_list[1])-res\_list[1].count("-")):

        buf = []

        for i in range(len(res\_list[1])):

            if str(type(res\_list[1][i])) != "<class 'str'>":

                buf.append(res\_list[1][i])

        res\_list[1] = buf

        res\_list[1] = mean(res\_list[1])

    else:

        res\_list[1] = "-"

    res\_list[2] = round(mean(res\_list[2]),2)

    res\_list[3] = round(mean(res\_list[3]),2)

    lst[0].append(res\_list[0])

    if (str(type(res\_list[1])) == "<class 'float'>"):

        lst[1].append(round(res\_list[1], 2))

    else:

        lst[1].append(res\_list[1])

    lst[2].append(res\_list[2])

    lst[3].append(res\_list[3])

    res\_list.clear()

    print("Результаты моделирования:")

    print(f'Среднее время между ложными тревогами - {lst[1]}')

    print(f'Среднее время запаздывания - {lst[2]}')

    return lst

def out\_table(N, L, mx = 0, m = 0):

    k\_l, k\_r = map(int, input("Введите пределы изменения длины серий через запятую (левый,правый) - ").split(','))

    shag = int(input("Введите шаг изменения длины серий (целое число) - "))

    lst = [[], [], [], []]

    k\_lst =[i for i in range(k\_l, k\_r+1, shag)]

    df = pd.DataFrame({'k':k\_lst})

    df2 = pd.DataFrame({'k':k\_lst})

    # for i in k\_lst:

    #     out\_lst = mean\_of\_razl(m, N, L, i, mx = mx, out = 0)

    #     lst[0].append(out\_lst[0][0])

    #     lst[1].append(out\_lst[1][0])

    #     lst[2].append(out\_lst[2][0])

    #     lst[3].append(out\_lst[3][0])

    for j in [0.25, 0.5, 1, 1.5, 2, 2.5]:

        mx = 0.5+j

        lst = [[], [], [], []]

        for i in k\_lst:

            out\_lst = mean\_of\_razl(m, N, L, i, mx = mx, out = 0)

            lst[0].append(out\_lst[0][0])

            lst[1].append(out\_lst[1][0])

            lst[2].append(out\_lst[2][0])

            lst[3].append(out\_lst[3][0])

        df[f'{j}'] = lst[2]

        df2[f'{j}'] = [round(lst[1][i]/lst[2][i], 2)if lst[1][i]!='-' else '-' for i in range(len(lst[1]))]

    print("Исходные данные:")

    print(f"Медиана ряда с разладкой - {mx}")

    print(f"Количество повторений для одного k - {m}")

    print(f"Номер такта номинальной разладки - {N}")

    print(f"Длина сигнала - {L}")

    # df = pd.DataFrame(lst, columns=list(range(1, (len(lst[0])+1))), index=[

    #                   "Длина серии", "Среднее время между ложными тревогами", "Среднее время запаздывания", "Номер такта обнаружения разладки"])

    what = ""

    while what not in ["Y", "y", "N", "n"]:

        what = input(

            "Вывести таблицу в excel / Вывести таблицу в консоль 'y/n' ")

    if what == 'y':

        df.to\_excel("table.xlsx", sheet\_name="sheet\_1", header=False)

        df2.to\_excel("eff.xlsx", sheet\_name="sheet\_1", header=False)

    elif what == 'n':

        print(df)

        print(df2)

def razl(N, L, k, mx = 0, out=1):

    '''

    Определение разладки сигнала

    '''

    def opr\_razl(x, med, k, N):

        '''

        Реализация алгоритма случайных блужданий

        '''

        global Tlt

        k\_p = 0  # Число положительных серий

        Tlt = []  # Моменты времени ложных тревог

        Tr = []  # Моменты реальных тревог

        for i in range(len(x)):

            if not corr:

                if x[i] >= med:

                    k\_p += 1

                elif x[i] < med:

                    k\_p = 0

            # elif corr and bin\_1:

            #     if x[i] == 1:

            #         k\_p += 1

            #     elif x[i] == 0:

            #         k\_p = 0

            else:

                if x[i] >= med:

                    k\_p += 1

                elif x[i] < med:

                    k\_p = 0

            if i == N:

                k\_p = 0

            if k\_p >= k:

                if i < N:

                    Tlt.append(i)

                elif i >= N:

                    Tr.append(i)

                k\_p = 0

        return Tr

    def get\_signal(N, L, mx):

        """

        Генерация временного ряда

        """

        if not corr:

            x = []

            x = [random.normalvariate(0.5,1) for i in range(N)]

            med = median(x)

            x.extend([random.normalvariate(mx,1) for i in range(N, L)])

        else:

            if bin\_1:

                x = get\_corr\_row(N+1, lamida1 = b1, lamida2 = b2)

                med = median(x)

                x.extend(get\_corr\_row(L-N+1, lamida1 = b1, lamida2 = b2))

            else:

                x\_all = add\_correlation(unif\_to\_gauss([(random.uniform(0,1))\*1 for i in range(L)], L), b1 = b1, b2 = b2)

                x = [i+0.5 for i in x\_all[:N]]

                med = median(x)

                x.extend([i+mx for i in x\_all[N:]])

        return [x,med]

    x, med = get\_signal(N, L, mx)

    Tr = opr\_razl(x, med, k, N)

    razn = []  # Список разностей между ложными тревогами

    # Определение среднего времени между ложными тревогами

    mean\_Tlt = 0  # Среднее время между ложными тревогами

    if len(Tlt) != 0:

        if len(Tlt) == 2:

            mean\_Tlt = Tlt[1]-Tlt[0]

        elif len(Tlt) == 1:

            mean\_Tlt = "-"

        else:

            for i in range(len(Tlt)-1):

                razn.append(Tlt[i+1]-Tlt[i])

            mean\_Tlt = mean(razn)

    else:

        if len(Tlt) == 0:

            mean\_Tlt = "-"

    if out == 1:

        # Вывод результатов

        print("Результаты моделирования:")

        if len(Tlt) != 0:

            print(f'Моменты времени ложных тревог - {Tlt}')

        else:

            print("Ложные тревоги отсутствуют")

        print(f'Среднее время между ложными тревогами - {mean\_Tlt}')

        if len(Tr)!=0:

            print(f'Время запаздывания - {Tr[0]-N}')

        else:

            print("Реальные тревоги отсутствуют")

        x\_dot = []      # Значения ряда в моменты реальных тревог

        for i in Tlt:

            x\_dot.append(x[i])

        x\_dot\_real = []  # Значения ряда в моменты реальных тревог

        for i in Tr:

            x\_dot\_real.append(x[i])

        if not bin\_1:

            if corr:

                plot\_auto\_corr(np.array(x[:N]), 20)

            fig, axs = plt.subplots(1,3)

            axs[1].hist(x[:N], bins = 10)

            axs[1].set\_title('Гистограмма исходного сигнала', fontsize=15)

            axs[1].set\_ylabel('Число наблюдений', fontsize=10)

            axs[2].hist(x[N:L], bins = 10)

            axs[2].set\_title('Гистограмма сигнала с разладкой', fontsize=15)

            axs[2].set\_ylabel('Число наблюдений', fontsize=10)

        else:

            if corr:

                plot\_auto\_corr(np.array(x),20)

            fig, axs = plt.subplots(1,3)

            axs[1].hist(x[:N], bins = 20)

            axs[1].set\_title('Гистограмма исходного сигнала', fontsize=15)

            axs[1].set\_ylabel('Число наблюдений', fontsize=10)

            axs[2].hist(x[N:L], bins = 20)

            axs[2].set\_title('Гистограмма сигнала с разладкой', fontsize=15)

            axs[2].set\_ylabel('Число наблюдений', fontsize=10)

        axs[0].plot(range(len(x)), x)

        axs[0].plot(Tlt, x\_dot, '.', label="Ложная тревога",

                 color="r", markersize=10)

        axs[0].plot(Tr, x\_dot\_real, ".", label="Реальная тревога",

                    markersize=10, color="0")

        axs[0].axhline(med, color='y', label='Медиана до разладки')

        axs[0].legend()

        plt.show()

    else:

        if not bin\_1:

            out\_lst = [k, mean\_Tlt, Tr[0]-N, Tr[0]]

        else:

            out\_lst = [k, mean\_Tlt, 0, 0]

        return out\_lst

def opt\_k\_for\_Tlt(N, L, Tlt, mx = 0, m = 1):

    """

    Нахождение k, удовлетворяющего заданному Tlt

    """

    k = 1

    out\_lst = mean\_of\_razl(m, N, L, k, mx = mx)

    delta = abs(Tlt-out\_lst[1][0])

    while (1):

        k += 1

        out\_lst = mean\_of\_razl(m, N, L, k, mx = mx)

        if str(type(out\_lst[1][0])) != "<class 'str'>":

            delta\_next = abs(Tlt-out\_lst[1][0])

            if delta\_next > delta:

                k -= 1

                break

        else:

            break

        delta = delta\_next

    print("Исходные данные")

    print(f"Заданное среднее время между ложными тревогами - {Tlt}")

    print(f"Медиана ряда с разладкой - {mx}")

    print(f"Количество повторений для одного k - {m}")

    print(f"Номер такта номинальной разладки - {N}")

    print(f"Длина сигнала - {L}")

    print(f"Оптимальное значение k - {k}")

    st = ""

    while st not in ["y", "Y", "n", "N"]:

        st = input("Вывести результаты исследования для данного k? 'y/n'")

    if st in ["Y", "y"]:

        mean\_of\_razl(m, N, L, k, mx = mx)

def main():

    global corr

    global bin\_1

    bin\_1 = False

    st = ""

    while st not in ["y", "Y", "n", "N"]:

        st = input("Исследовать коррелированный сигнал? 'y/n' ")

    m = int(input("Введите по скольким значениям проводить усреднение - "))

    if st in ['Y','y']:

        corr = True

    else:

        corr = False

    if not corr:

        st = 0

        while st not in [1, 2, 3]:

            st = int(input("""

            1-Определение разладки

            2-Построить таблицу зависимостей

            3-Определение оптимального k для заданного Tlt'

            """))

        N = int(input("Введите с какого отсчёта пойдёт разладка - "))

        L = int(input("Введите общее число отсчётов - "))

        mx = float(input(

        "Введите медиану процесса начиная с момента заданной разладки(исходная=0.5) - "))

        if st == 1:

            k = int(input("Введите длину серий успехов - "))

            razl(N, L, k, mx = mx)

            lst = mean\_of\_razl(m, N, L, k, mx = mx)

            print("Результаты моделирования:")

            print(f'Среднее время между ложными тревогами - {lst[1]}')

            if len(lst[2])!=0:

                print(f'Среднее время запаздывания - {lst[2]}')

            else:

                print("Реальные тревоги отсутствуют")

        elif st == 2:

            out\_table(N, L, mx = mx, m = m)

        elif st == 3:

            Tlt = float(input("Введите необходимое значение Tlt - "))

            opt\_k\_for\_Tlt(N, L, Tlt, mx = mx, m = m)

    else:

        st = 0

        while st not in [1, 2, 3]:

            st = int(input("""

            1-Определение разладки

            2-Построить таблицу зависимостей

            3-Определение оптимального k для заданного Tlt'

            """))

        global b1, b2

        b1, b2 = map(float, input("Введите коэффициенты для рекуррентного соотношения через запятую (b1,b2) - ").split(','))

        if not bin\_1:

            N = int(input("Введите с какого отсчёта пойдёт разладка - "))

            mx = float(input(

            "Введите медиану процесса начиная с момента заданной разладки(исходная=0.5) - "))

        L = int(input("Введите общее число отсчётов - "))

        if bin\_1:

            N = L

            mx = 0

        if st == 1:

            k = int(input("Введите длину серий успехов - "))

            print(f'B1={b1}, B2={b2}')

            razl(N, L, k, mx = mx)

            mean\_of\_razl(m, N, L, k, mx = mx)

            lst = mean\_of\_razl(m, N, L, k, mx = mx)

            print("Результаты моделирования:")

            print(f'Среднее время между ложными тревогами - {lst[1]}')

            if len(lst[2])!=0:

                print(f'Среднее время запаздывания - {lst[2]}')

            else:

                print("Реальные тревоги отсутствуют")

        elif st == 2:

            print(f'B1={b1}, B2={b2}')

            # razl(N, L, 1000, mx = mx)

            out\_table(N, L, mx = mx, m = m)

        elif st == 3:

            Tlt = float(input("Введите необходимое значение Tlt - "))

            opt\_k\_for\_Tlt(N, L, Tlt, mx = mx, m = m)

if \_\_name\_\_ == "\_\_main\_\_":

    main()