МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий

Кафедра «Прикладная математика и информатика»

01.04.02 ПРИКЛАДНАЯ МАТЕМАТИКА И ИНФОРМАТИКА

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему «Моделирование и визуализация динамики планарных квазичастиц в терминах 3D струн»

Студент К. Э. Бегун

Научный руководитель С. В. Талалов

Руководитель программы доктор ф.-м.н., доцент, С.В. Талалов _____

«_____»_____2017 г.

Допустить к защите

Заведующий кафедрой к.т.н., доцент, А.В. Очеповский _____

«_____»_____2017 г.

Тольятти 2017

Оглавление

Введение 3
Глава 1 Теоретико-методологический обзор 7
1.1 Планарные квазичастицы 7
1.2 Кубит как элементарная единица квантовой информации 15
1.3 Геометрический подход в теории струн 20
Глава 2 Математическая модель трехмерной струны 27
2.1 Задача Штурма-Лиувилля 27
2.2 N-солитонные решения спектральной задачи второго порядка 33
2.3 Разбор простого случая N=3 44
Глава 3. Практическое вычисление моделируемого объекта 47
3.1 Листинг программы диссертации 47
3.2 Результаты 76
Заключение 82
Список использованной литературы 83

Введение

На сегодняшний день ведется исследование проблемы создания полноценного квантового компьютера. Необходимость в последнем возникает при использовании физического метода для огромного числа частиц при моделировании сложных систем.

Например, пространство состояний системы из различных микроорганизмов имеет экспоненциальный рост при увеличении количества частиц, что сделает невозможным моделирование их поведения на классических компьютерах уже для тысячи таких объектов.

Чем квантовый компьютер отличается от классического? Квантовый компьютер основан на таких квантовомеханических эффектах, как, например, квантовая запутанность и квантовый параллелизм. Это позволяет преодолеть известное ограничение классических компьютеров, которое выходит за рамки настоящей диссертации.

Функционирование такого компьютера связано с необычными свойствами мира элементарных частиц. Кванты, даже будучи в громадном отдалении друг от друга, могут оказаться в запутанном состоянии, в котором эти объекты все-таки зависят друг от друга. О квантовом параллелизме вычислений можно кратко рассказать на основе следующего сравнения: в ЭВМ изменение состояния одного бита никак не затронет остальные биты, а в квантовом компьютере любое воздействие (в том числе и наблюдение) на одну его частицу повлечет изменение состояния всех других. Поэтому последний выдаст большую производительность по сравнению с первым. Для выполнения программы, такой как криптографической, квантовый компьютер задействует систему из сотни атомов, общая производительность которых превысит миллиард операций в миллисекунды.

В качестве научной проблемы, к которой имеет отношение настоящая диссертация, выступает возможность построения квантового компьютера через топологические косы, реализованные посредством энионов.

Итак, характеристиками энионов будут обладать так называемые планарные квазичастицы – возбуждения динамических систем с большим (бесконечным, в предельном случае) числом степеней свободы. Поэтому актуальной является проблема математического моделирования планарных квазичастиц с заданными свойствами. В качестве модели может выступать представление струны в виде совокупности точек возврата.

Целью работы является исследование специальных случаев динамики, а так же поведения точек возврата планарной струны, отвечающих N-солитонным решениям спектральной задачи Штурма – Лиувилля, возникающей при исследовании некоторой (известной ранее) системы нелинейных дифференциальных уравнений в частных производных. Последняя возникает при исследовании динамики планарных струн. Для достижения данной цели в работе были поставлены следующие задачи:

1. Исследовать динамику точек возврата планарной струны.
2. Вывести формулы для построения мировых линий точек возврата, соответствующим N-солитонным решениям.
3. Провести серию вычислительных экспериментов по моделированию планарных квазичастиц точками возврата с помощью разработанного ПО, используя различные данные рассеяния спектральной задачи.

Научная новизна работы заключается в следующем:

- будут выведены расчетные формулы для визуализации мировых линий точек возврата, в терминах спектральной задачи второго порядка;
- будет проведена серия компьютерных экспериментов, в которых будет исследована динамика поведения точек возврата планарной струны, в том числе – топология кос мировых линий данных точек. На основе данных экспериментов

предполагается сделать вывод о возможности дальнейшего использования рассмотренных объектов для моделирования энионов.

Актуальность выполненной работы заключается в сложности наблюдения за реальными физическими энионами, поэтому приобретает важность построение их математических моделей.

Метод исследования заключается в следующем: сначала будет дано математическое обоснование данного исследования, затем произойдет эксперимент посредством компьютерного моделирования, следом соберутся статистические данные по данному вопросу.

Объектом исследования будет являться планарная струна с точками возврата.

Предметом исследования станет динамика точек возврата при различных комбинациях данных рассеяния спектральной задачи второго порядка.

Данная магистерская диссертационная работа состоит из аннотации, введения, 3 главы, выводов по работе, списка источников и приложения.

Во введении обосновывается актуальность выбранной темы, формулируются цель и задачи работы.

В первой главе будет выполнен обзор литературных источников, посвященных геометрическому подходу в теории струн и методу обратной задачи в теории нелинейных эволюционных уравнений, а также предоставлен анализ современного состояния исследований в данной области. Результатом первой главы будет являться теоретико-методологическое обоснование подхода к решению научно-практической проблемы представления мировых линий планарных квазичастиц как топологические косы.

Вторая глава будет посвящена изучению возможностей применения метода обратной задачи к модели планарных квазичастиц. Будет рассчитана динамика точек возврата 3D-струн для последующего процесса визуализации. Произойдет вывод расчетных формул для пространственных эволюционирующих кривых,

которые отвечают N-солитонным решениям спектральной задачи второго порядка уравнения Штурма-Лиувилля.

В третьей главе будет произведена серия компьютерных экспериментов над моделью планарных квазичастиц (по выведенным расчетным формулам) посредством разработанных программ с последующим накоплением статистики по начальным данным и данным рассеяния. На основе полученной информации, будет проведен анализ динамики точек возврата планарной струны.

В заключении будут сделаны выводы по результатам анализа исследуемого, моделируемого явления.

На защиту выносятся следующие положения:

1. Построена модель локализованных планарных объектов (точек возврата), которые можно интерпретировать как планарные квазичастицы.

2. Проведена серия компьютерных экспериментов, показывающие, что мировые линии данных объектов образуют косы различной топологии.

3. В ходе компьютерных экспериментов было продемонстрировано изменение топологии кос при изменении дискретного набора данных рассеяния спектральной задачи второго порядка.

Глава 1 Теоретико-методологический обзор

1.1 Планарные квазичастицы

В первом параграфе первой главы сначала дадим определение планарных квазичастиц, затем упомянем каждый проведенный эксперимент, призванный доказать их существование с нужными свойствами.

Планарные квазичастицы – это коллективные возбуждения в плоской микросистеме, свойства которых можно описать как свойства частиц. В качестве планарных квазичастиц, которые предполагается использовать при построении топологического квантового компьютера, можно представить так называемые энионы [13].

Итак, в топологическом квантовом компьютере сплетаются между собой мировые линии квазичастиц. В квантовой механике, в отличие от классической, свойства частицы описываются волновой функцией, которая определяет вероятность нахождения частицы в различных точках пространства (в зависимости от величины амплитуды), вероятность получения того или иного значения при измерении ее скорости и т.д. [1].

Группа электронов описывается объединенной волновой функцией, и когда, например, в парной системе два электрона меняются местами, результирующая волновая функция меняет знак («вал» становится «ямой» и наоборот), но амплитуда колебаний остается неизменной. То есть, все измеряемые характеристики электронов остаются прежними, если рассматривать частицы по отдельности, но после перестановки они будут по-другому взаимодействовать с другими электронными группами [14].

В рамках квантовой механики, в трехмерном пространстве частицы будут представлены бозонами и фермионами, в зависимости от знака фазы. Но в двухмерном пространстве, в уравнении квантовой механики в качестве множителя может выступать фаза, которую можно рассматривать как точку комплексной плоскости, лежащую на единичной окружности [15].

Частицы, которые при перестановке могут приобрести любую такую фазу, называются энионами, а, соответственно, другие частицы будут иметь определенную фазу [16].

Упомянем несколько явлений квантовой физики.

Квантовая запутанность – квантовомеханическое явление, при котором квантовое состояние двух или большего количества объектов должно описываться во взаимосвязи друг с другом, даже если отдельные объекты разнесены в пространстве.

Квантовая телепортация – передача состояния от одного кванта к другому и последующее разрушение исходного состояния в первом.

Квантовая суперпозиция – дает суперпозицию взаимоисключающих состояний.

Квантовый параллелизм – воздействие на одну частицу квантового компьютера повлечет изменение состояния всех других.

Декогеренция – это процесс нарушения связи, вызываемый взаимодействием квантовомеханической системы с окружающей средой посредством необратимого, с точки зрения термодинамики, процесса.

Теперь дадим определение спина квазичастицы.

Спин – собственный момент импульса элементарных частиц, имеющий квантовую природу и не связанный с перемещением частицы как целого. Спином называют также собственный момент импульса атомного ядра или атома; при этом спин определяется как векторная сумма, вычисленная по правилам сложения моментов в квантовой механике, спинов элементарных частиц, образующих систему, и орбитальных моментов этих частиц, обусловленных их движением внутри системы [35].

Спин измеряется в единицах $\hbar$ (приведенной постоянной Планка-Дирака) и равен $\hbar_{\mathrm{j}}$, где j – характерное для каждого вида частиц целое или полуцелое

положительное число. Поэтому принято говорить о целом или полуцелом спине частицы [36].

В 2005 г. Владимир Голдман, Фернандо Камино и Вэй Чжоу из нью-йоркского университета Стоуни-Брук экспериментально подтвердили, что квазичастицы в дробном квантовом состоянии Холла являются энионами [17]. Сделано это было на экспериментальном устройстве (рис. 1.1). Температура прибора, размещенного в сильном магнитном поле, была понижена до 0,01 К. В свободном пространстве между четверкой электродов образовался двухмерный электронный газ с различными типами квазичастиц в желтой и внутренней белой областях. Свойства пограничного тока указывают, что квазичастицы на желтом островке проявляют свойства энионов [2].

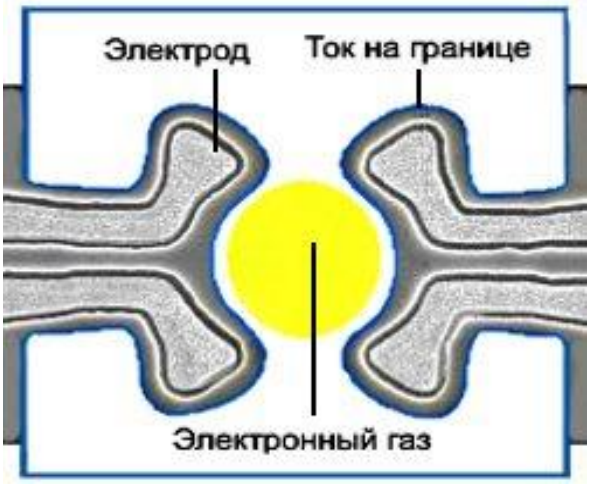

Рис. 1.1 – Схема детектора энионов

Для создания топологического квантового компьютера необходимо решить еще такую проблему: энионы должны быть неабелевыми, иначе говоря, в данном случае важен порядок, в котором обмениваются частицы. Неабелевы свойства возникают потому, что для таких энионов множитель волновой функции представляет собой матрицу чисел, а результат умножения двух матриц зависит от последовательности их перемножения [2].

При перестановке частиц в двух измерениях фаза, приобретаемая волновой функцией, зависит от того, перемещаются они по часовой стрелке или против нее. С точки зрения топологии, эти два направления различаются, потому что невозможно при помощи плавной деформации изменить движение по часовой стрелке на движение против часовой стрелки, потому что, в этом случае, будет иметь место пересечение траекторий и столкновение частиц. Таким образом, все возможные переплетения мировых линий (траекторий в пространстве-времени) множества энионов образуются в результате их перестановки либо по часовой стрелке, либо против нее [3].

Имея в распоряжении неабелевы энионы, появляется возможность создать физическое представление группы кос – математической структуры, помогающей описать все способы, которыми можно переплести данный набор нитей. Любую косу можно построить из ряда элементарных операций, в которых две смежные нити движутся или по часовой стрелке, или против нее. Каждая возможная последовательность манипуляций с энионами соответствует косе и наоборот. Кроме того, каждой из этих кос соответствует сложная матрица, получающаяся в результате объединения всех матриц перестановки энионов [18].

Последовательность действий такова: во-первых, создаются пары энионов, выстраивающиеся в ряд, чтобы изобразить кубиты, участвующие в вычислениях. Следующий этап: в заданной последовательности осуществляются перестановки смежных энионов, соответствующих операциям, выполняемым над кубитами. Далее, пары смежных энионов объединяются и измеряются для получения результатов вычисления. Результат на выходе зависит от топологии косы, которая задается проведенными манипуляциями. Небольшие возмущения энионов не изменяют топологию, таким образом, процесс вычисления оказывается защищенным от обычных помех [3].

Главное препятствие на пути создания квантового компьютера, помимо технологических проблем, представляет декогерентизация, нарушающая

состояние квантовой системы и затрудняющая процесс вычисления. Работа физиков-теоретиков идет по трем направлениям. Одни разрабатывают методы коррекции ошибок. Для квантового компьютера это сделать затруднительно из-за явления, сформулированного в теореме о запрете клонирования неизвестного состояния квантовой системы. В настоящее время возможно только исправление некоторых типов ошибок. В другом направлении ищут возможности подавления процессов декогерентизации. И в третьем направлении изобретают квантовые системы, устойчивые к этим процессам [15].

Рассмотрим, как косы связаны с квантовыми вычислениями. Состояние классического компьютера можно охарактеризовать совокупностью всех его битов – последовательностью нулей и единиц в регистрах. Точно так же состояние квантового компьютера определяется состоянием всех его кубитов. В топологическом квантовом компьютере кубиты можно представить группами энионов. В квантовом компьютере процесс перехода от начального состояния к конечному описывается матрицей, которая умножается на объединенную волновую функцию всех кубитов. Здесь можно увидеть сходство с тем, что происходит в топологическом квантовом компьютере: в данном случае это матрица, относящаяся к конкретной косе, соответствующей последовательности манипуляций с энионами. Таким образом, операции с энионами приводят к квантовому вычислению [4].

Проблемой обнаружения кос, выполняющих определенные вычисления, занимались Николас Боунстил вместе с коллегами из Университета штата Флорида и лаборатории Lucent Technologies фирмы Bell Labs. Группа продемонстрировала построение логической операции «управляемое НЕ» (CNOT), который работает с точностью до $2 \cdot 10^{-3}$, с помощью косы из шести энионов. Эта операция означает, что конечное состояние входного кубита зависит от состояния второго кубита (управляющего). Для упрощения расчетов при проектировании логического элемента было решено энионы одной тройки оставить неподвижными

и перемещать вокруг них две ближайшие частицы из другой тройки. Действуя на кубиты с помощью сети из элементов CNOT и операции умножения отдельных кубитов на комплексную фазу, можно построить любое вычисление. Это в очередной раз подтверждает, что топологические квантовые компьютеры могут справиться с любым квантовым вычислением [19].

Фридман, Китаев и Ванг доказали, что работу топологического квантового компьютера можно эффективно моделировать с произвольной точностью на обычном квантовом компьютере, а значит, все, что может вычислить топологический квантовый компьютер, доступно и обычному квантовому [4]. Несомненно, все достаточно развитые вычислительные системы, использующие квантовые средства, характеризуются в точности одинаковыми вычислительными возможностями. Аналогичный тезис для классических вычислений был предложен Алонсо Черчем и Аланом Тьюрингом в 30-х гг. XX в. [38].

Квантовые методы выполнения вычислительных операций, а также передачи и обработки информации, уже начинают воплощаться в реально функционирующих экспериментальных устройствах, что стимулирует усилия по реализации квантовых компьютеров – этого нового направления в вычислительной технике [39].

Количество публикаций по квантовой теории информации и квантовым вычислениям приобрело в последнее время лавинообразный характер, появились и экспериментальные работы.

Итак, важнейшей и, одновременно, самой парадоксальной основой квантовых вычислений является сцепленное состояние нескольких частиц: если несколько частиц составляют единую квантовую систему, то они вполне могут разлететься на неограниченное расстояние, не теряя своего квантового единства. А это означает, что любое воздействие на одну из них автоматически меняет состояние другой точно так же, как если бы она была совсем рядом [2].

Эйнштейн, Подольский и Розен в 1935 году в своей совместной статье рассматривали мысленный эксперимент с указанным парадоксом, дающий основание для серьезной критики основ квантовой механики. В конечном счете, эта критика стимулировала глубокий анализ и подтолкнула дальнейшее развитие квантовой теории. Следует заметить, что сцепленные состояния сегодня не только экспериментально подтвержденный, но и промышленно используемый феномен. Хотя пока это «промышленное использование» ограничивается только системами квантовой криптографии [3].

Но такая сложность (а скорее, непривычность) дает громадное преимущество квантовой вычислительной техники по сравнению с обычной: уровень сложности некоторых вычислительных задач очень сильно снижается, если их «переложить на плечи» квантового компьютера. Например, разложение числа на множители (задача, сложность решения которой обеспечивает сейчас устойчивость множества шифров к взлому) и нахождение произведения двух сомножителей (в обычном умножении) для квантового компьютера являются задачами сравнимой сложности. При этом первую затруднительно реализовать на «классическом» компьютере в приемлемые сроки для достаточно большого числа (к примеру, чтобы разложить на сомножители число из 256 цифр, потребуется несколько лет), а вторая является посильной для простейшего калькулятора [9].

Ещё одна надежда, возлагаемая на квантовые компьютеры, связана с невозможностью точно рассчитать реальные физические системы.

Совокупность всех возможных операций на входе данного компьютера, формирующих исходные состояния, а также осуществляющих унитарные локальные преобразования, соответствующие алгоритму вычисления, способы подавления потери сцепления квантовых состояний и исправления случайных ошибок, играют здесь ту же роль, что и программное обеспечение в классическом компьютере [4].

После того, как упомянут объект, на основе которого будет построена модель для проводимого эксперимента в рамках данной работы, теперь перейдем к описанию единицы информации в квантовых компьютерах.

1.2 Кубит как элементарная единица квантовой информации

Расскажем о единице измерения квантовой информации.

Бит – это наименьшая возможная единица информации в классической вычислительной технике. Аналогом бита в квантовых компьютерах является кубит, или квантовый бит (q-бит, от quantum bit) – единица квантовой информации, или наименьший элемент для хранения информации в квантовом компьютере [8].

Бит может иметь одно из двух состояний: 0 или 1. Поэтому бит можно представить абстрактно – стрелкой, направленной вверх или вниз.

Как и бит, кубит допускает два собственных состояния, обозначаемых $|0\rangle$ (северный полюс) и $|1\rangle$ (южный полюс), но при этом может находиться не в одном из этих основных состояний, а и в так называемой квантовой суперпозиции, то есть в состоянии $A*|0\rangle+B*|1\rangle$, где A и B – комплексные числа, удовлетворяющие условию $|A|^2+|B|^2=1$. Эти состояния можно представить стрелкой, указывающей на точку сферы (радиус-вектором). Наличие континуума состояний между 0 и 1 – причина многих необычных свойств квантовой информации [5].

Подобно биту, кубит – абстрактный объект, никак не связанный с принципами квантовой механики. Свойства кубита не зависят от его физической реализации [9]. Сначала в качестве кубитов было предложено использовать фотоны, в дальнейшем – другие физические реализации, например, спины ядер и электронов. У всех этих реализаций есть свои достоинства и недостатки.

Велико заблуждение о том, что кубит хранит неограниченное количество информации, так как координаты расшифровываются в бесконечный числовой ряд. Но информация должна быть извлечена из кубита путем измерения. Нельзя напрямую узнать состояние кубита, так как он при этом тут же перейдет в другое свое собственное состояние. Существует ограничение: требуется, чтобы результат, выводимый после измерения кубита, был системой с двумя состояниями.

Вероятности перехода из одного состояния в другое равны соответственно $|A|^2$ и $|B|^2$, то есть имеется возможность косвенно судить о первоначальном состоянии кубита [20].

Чтобы задать квантовое состояние, мы должны указать координаты соответствующей точки на сфере. Их можно закодировать как определенную последовательность битов. Результат определения состояния кубита при помощи обычного прямого измерения даст либо 0, либо 1. Вероятность полученного результата зависит от широты исходного состояния.

Данное умозаключение было высказано известным советским специалистом в области квантовой информатики Александром Холево в 1973 г. в качестве доказательства гипотезы, выдвинутой американским ученым-физиком Дж. Гордоном в 1964 г. В гипотезе говорится, что в кубите есть скрытая информация, которую можно изменить, но нельзя узнать. Но все же лучше рассматривать ее в качестве особой квантовой единицы информации, а не как бесконечную последовательность нулей и единиц [8].

Кубиты могут быть сцеплены друг с другом, то есть, на них может быть наложена незаметная связь, выражающаяся в том, что при всяком воздействии на один из нескольких кубитов, остальные согласованно изменятся. То есть, совокупность связанных между собой кубитов можно интерпретировать как квантовый регистр. Как и отдельный кубит, квантовый регистр гораздо более информативен. Он может находиться не только во всевозможных комбинациях составляющих его битов, но и реализовывать всевозможные тонкие зависимости между ними [21].

Несмотря на то, что нельзя непосредственно наблюдать состояние кубитов или квантовых регистров во всей полноте, между собой они могут взаимодействовать. На основе этого можно создать компьютер, способный к параллельным вычислениям на уровне своего физического устройства и остаётся выяснить, как прочесть конечный результат вычислений [9].

Сцепленные кванты будут взаимосвязаны между собой, и расстояние ни в коем случае не ослабит их сцепление. Если какой-то объект сцеплен с другими, то узнать о его состоянии можно вместе со сведениями о его партнерах [8].

В 1991 г. Артур Экерт изобрел способ применения сцепленности для неперехватываемой передачи криптографических ключей. Также, в 1992 г. Чарльз Беннет и Стивен Виснер показали, что посредством сцепленности можно пересылать классическую информацию.

В 1993 г. группа исследователей связала телепортацию квантового состояния со сцепленностью. Наконец, появились другие предположения по практическому использованию явления сцепления [22].

Как и отдельные кубиты, реализация свойств сцепленности не зависит от её физического представления [9]. Например, для квантовой криптографии можно использовать сцепленные пары и фотонов, и атомных ядер, и даже пара "фотон-ядро".

Во второй половине 90-х ученые обнаружили, что различные формы сцепленности не отличимы друг от друга – так, сцепленность одних элементарных частиц возможно передать на другие.

Основываясь на качественных соотношениях, ученые пытаются определить количественную меру сцепленности [9]. Единства в выборе способа её оценки нет. На сегодня наиболее успешная схема опирается на понятие о стандартной единице сцепленности, которая напоминает единицу массы или энергии. Величина сцепленности сравнивается в двух различных состояниях и определяется тем, сколько копий эталонной единицы сцепленности нужно для уравновешивания [23].

Для создания работоспособного топологического квантового компьютера необходимо продумать механизм установки исходного значения кубитов, а также процедуру считывания получаемых результатов.

На этапе инициализации создаются пары квазичастиц.

На этапе считывания производится измерение состояний энионов. Если они далеко отстоят друг от друга, то провести измерение невозможно: энионы должны поступать парами.

Грубо говоря, требуется выяснить, полностью ли аннигилируют пары (как истинные античастицы), или после их взаимодействия остаются заряды и потоки, показывающие, как состояния были изменены при переплетении по сравнению с точным отношением античастиц, с которого начался процесс.

Для классической вычислительной техники давно реализован целый ряд мер для исправления ошибок и защиты информации от разрушительных помех. Например – код с повторами. В этой схеме ноль изображается как строка из трех битов: 000, а единица – как строка из трех битов: 111. Если флуктуации относительно малы, то они могут исказить один из битов в тройке, изменив, например, 000 на 010; изменение же двух битов в триплете происходит значительно реже. Если встречается триплет 010, то правильное значение, скорее всего, равно 000, т.е. был передан ноль. Более сложные обобщения этой идеи дают очень хорошие результаты в коррекции ошибок для защиты информации [11].

К сожалению, нельзя сказать, что топологический компьютер полностью нечувствителен к ошибкам. Хрупкие квантовые состояния необходимо максимально изолировать от внешней среды, так как они легко разрушаются посторонними воздействиями или тепловыми шумами. Поэтому методы повышения помехоустойчивости имеют огромное значение [10].

Главный источник шумов – тепловые колебания в материале подложки, при которых может возникнуть дополнительная, паразитная, пара энионов. Она вплетается в косу, искажает процесс вычисления, а в конце снова аннигилирует. К счастью, процесс тепловой генерации подавляется низкой температурой, при которой работает топологический компьютер. Кроме того, вероятность такого вмешательства убывает по экспоненте по мере увеличения расстояния между энионами. Таким образом, можно достигнуть любой необходимой степени

точности, создавая достаточно большой компьютер и удерживая работающие энионы в ходе плетения кос вполне обособленными [10].

Теперь можно кратко рассказать о геометрическом подходе в теории струн.

1.3 Геометрический подход в теории струн

Будем давать модель релятивистской струны через обобщение релятивистской динамики точки на одномерно-протяженные объекты. Для однозначного определения действий струны потребуем релятивистскую инвариантность. Действие релятивистской струны выбирается пропорциональным площади мировой поверхности, покрываемой струной в процессе своего движения в пространстве Минковского [6].

Теория струн – направление теоретической физики, изучающее динамику взаимодействия не точечных частиц, а одномерных протяжённых объектов – квантовых струн. Теория струн основана на гипотезе о том, что все элементарные частицы и их фундаментальные взаимодействия возникают в результате колебаний и взаимодействий ультрамикроскопических квантовых струн на масштабах порядка планковской длины 10^{-35} м. Характер колебаний струны задаёт свойства материи, такие как электрический заряд и масса [14].

Как хорошо известно, релятивистская динамика точки с массой m и координатами r(t) задается функцией действия

$$S_m = -mc^2 \int_{t_1}^{t_2} \sqrt{1 - \frac{v^2}{c^2}}\, dt. \qquad (1.1)$$

Пусть теперь мы имеем одномерно-протяженный объект, который для простоты будем называть релятивистской струной. Его положение в пространстве в момент времени t задается трехмерной вектор-функцией r(t, σ), $\sigma_1 < \sigma < \sigma_2$. Параметр σ "нумерует" точки струны. Далее, предположим, что вдоль струны однородно распределена масса с плотностью на единицу длины ρ_0. Теперь предположим, что внутренние силы между соседними точками струны таковы, что они не дают вклад в действие (например, среди внутренних сил нет обычных сил упругости). Тогда действие релятивистской струны получаем, интегрируя (1.1) вдоль всей струны [24]:

$$S_{cmp} = -\rho_0 c^2 \int_{t_1}^{t_2} dt \int_0^l dl \sqrt{1 - \frac{v_\perp^2}{c^2}}. \quad (1.2)$$

Здесь $v_\perp(t,o)$ – нормальная составляющая вектора скорости струны в точке σ,

$$v(t,\sigma) = \frac{\partial r(t,\sigma)}{\partial t} - k\left(\frac{\partial r}{\partial t}\frac{\partial r}{\partial \sigma}\right), \quad (1.3)$$

где k – касательный вектор к струне в точке σ:

$$k = (\partial \dot{r} / \partial \sigma)/(\partial r / \partial \sigma)^2; \quad (1.4)$$

dl — дифференциальный элемент длины струны,

$$dl = d\sigma \sqrt{(\partial r / \partial \sigma)^2}. \quad (1.5)$$

Подставляя (1.5) в (1.2), получаем

$$S_{cmp} = -\rho_0 c^2 \int_{t_1}^{t_2} dt \int_{\sigma_1}^{\sigma_2} d\sigma \sqrt{\left(\frac{\partial r}{\partial \sigma}\right)^2 \left(1 - \frac{v^2}{c^2}\right)}. \quad (1.6)$$

В действие струны введена только нормальная составляющая скорости точек струны, т. е. внутренняя динамика струны не рассматривается [25].

Недостатком формул (1.6) и (1.1), является отсутствие явной релятивистской ковариантности. Сначала устраним это на примере случая точечной частицы. Время t в (1.1) будем считать функцией некоторого параметра τ, тогда функциями от него будут координаты точки $r(t) = r(t(\tau)) = X(\tau)$. Лоренцевский вектор $x^\mu(\tau) = \{ct(\tau), x(\tau)\}$ задает траекторию материальной точки в пространстве Минковского. После такой замены формула (1.1) принимает следующий вид [49]:

$$S_m = -mc \int_{\tau_1}^{\tau_2} \sqrt{\dot{x}^2}\, d\tau, \quad (1.7)$$

где $\dot{x}^\mu = \frac{dx^\mu(\tau)}{d\tau}$; $\mu = 0,1,2,3$; $\dot{x}^2 = c^2 t^2 - \dot{X}^2$.

Действие точечной частицы в форме (1.7) имеет наглядную геометрическую интерпретацию: оно пропорционально длине мировой траектории частицы [6].

Сделаем аналогичную замену в действии струны (1.2). Будем считать время t в этой формуле функцией некоторого нового параметра τ и старого параметра σ. В результате координаты струны становятся функциями этих параметров: $r(t,\sigma) = r(t(\tau,\sigma),\sigma) = X(\tau,\sigma)$. Можно ввести в рассмотрение лоренцевский вектор, задающий четырехмерные координаты струны в пространстве-времени $x^{\mu}(\tau,\sigma) = \{ct(\tau,\sigma), x(\tau,\sigma)\}$. Сделаем замену переменных интегрирования [26] в формуле (1.2):

$$\left.\begin{aligned} & t = t(\tau,\sigma);\ \sigma \to \sigma; \\ & dtd\sigma = \frac{\partial(t,\sigma)}{\partial(\tau,\sigma)} d\tau\, d\sigma = \dot{t} d\tau\, d\sigma; \\ & \frac{\partial r}{\partial t} = \frac{\dot{X}}{\dot{t}}, \frac{\partial r}{\partial \sigma} = X' - \dot{X}\frac{t'}{\dot{t}}. \end{aligned}\right\} \qquad (1.8)$$

Здесь используются следующие сокращенные обозначения частных производных по параметрам τ и σ:

$$\dot{f}(\tau,\sigma) = \frac{\partial f(\tau,\sigma)}{\partial \tau}\ ;\ f'(\tau,\sigma) = \frac{\partial f(\tau,\sigma)}{\partial \sigma} \qquad (1.9)$$

После подстановки (1.3), (1.4) и (1.8) в (1.2) действие струны принимает релятивистский инвариантный вид:

$$S_{cmp} = -\rho_0 c \int_{\sigma_1}^{\sigma_2} d\sigma \int_{\tau_1(\sigma)}^{\tau_2(\sigma)} d\tau \sqrt{(\dot{x}x')^2 - \dot{x}^2 x'^2}, \qquad (1.10)$$

Функции $\tau_i(\sigma),\ \ i = 1,2$ задают область интегрирования в новых переменных τ и σ [6].

Формула (1.10) имеет простой геометрический смысл – с точностью до множителя $-\rho_0 c$ она представляет собой функционал площади мировой поверхности струны, заданной параметрически: $x^{\mu}(\tau,\sigma)$ [6].

Релятивистскую струну возможно задать тремя равнозначными способами [13]:

1. Координатно: пространственная кривая общего положения, с каждой точкой которой связан квантовый гармонический осциллятор. С точки зрения динамики при движении заметает двумерную поверхность общего вида [13].

2. Алгебро-геометрически: алгебраическая кривая общего вида, с допустимыми на ней математическими структурами [13].

3. Геометрически: точка без параметров общего положения в пространстве всех физических конфигураций струн, то есть не зависящих от системы координат (пространство петель) [13].

Существуют струны, у которых есть концы, их называют открытыми, и у которых концов нет, их называют замкнутыми. В случае если струна зависит только от бозонных переменных, то она является бозонной. Если зависит только от фермионных переменных – то фермионной. Если и от бозонных, и от фермионных, при условии суперсимметрии, то суперсимметричной или суперструной. Если требование суперсимметрии частично невыполнимо, то гетеротической [40].

С точки зрения изменения формы струны (топологии) допустимы 5 следующих элементарных локальных актов, согласующихся с физическими принципами [14]:

1. Разрыв в точке открытой струны на пару новых [14].

2. Сход замкнутой струны во внутреннюю точку касания и последующее расщепление на пару новых [14].

3. Разрыв в точке замкнутой струны в открытую [14].

4. Обмен сегментами в точке касания 2 открытых струн [14].

5. Утрата сегмента открытой струны в качестве замкнутой струны через внутреннюю точку касания [14].

Теперь опишем первую и вторую квадратичные формы.

Первая квадратичная форма g_{ij} задает внутреннюю геометрию поверхности (т.е. свойства, сохраняющиеся при изометрических преобразованиях) [6]:

$$dl^2 = dx^\mu dx^\mu c_\mu = x_i^\mu x_j^\mu c_\mu du^i du^j = \sum_{i,j=1}^{2} g_{ij}(u) du^i du^j,$$
$$g_{ij} = x_i^\mu x_j^\mu c_\mu = e_i e_j, \tag{1.11}$$

где x^μ – радиус-вектор, u_1, u_2 – параметры поверхности $x^\mu = x^\mu(u_1, u_2)$, $\mu = \overline{0, n-1}$,

n – размерность пространства (в нашем случае n = 3), множители $c_\mu = \pm 1$ учитывают сигнатуру метрики объемлющего пространства, e_i, e_j – базисные векторы [6, 7].

Вторая квадратичная форма b_{ij} определяет нормальное ускорение точки на поверхности [7] и задается формулой

$$\left\langle \ddot{x}^\mu, \vec{n}_e \right\rangle dt^2 = b_{ij} dx^i dx^j = b_{11} du_1^2 + 2b_{12} du_1 du_2 + b_{22} du_2^2, \tag{1.12}$$

где $\vec{n}_e$ – единичный вектор нормали к поверхности [7].

Первая и вторые квадратичные формы задают движение по поверхности базиса, образованного набором из m касательных векторов $x_i^\mu, i = 1,\ldots,m$ и m - n нормалями e_a^μ, $a = m + 1,\ldots, n$ из ортонормированного базиса. Движение этого базиса описывается уравнениями (1.12) и аналогичными уравнениями на нормали e_a^μ, $a = m + 1,\ldots, n$, которые в терминах g_{ij} и b_{ij} имеют следующий вид [6]:

$$\frac{\partial e_a^\mu}{\partial u^i} = -b_{a|ij} g^{ik} x_k^\mu + \sum_\beta c_\beta \nu_{\beta a|i} e_\beta^\mu. \tag{1.13}$$

Здесь дополнительно к квадратичным дифференциальным формам g_{ij} и $b_{a|ij}$ введены так называемые векторы кручения $\nu_{\beta\alpha|i} = -\nu_{\alpha\beta|i}$; $\alpha, \beta = m+1,\ldots,n, i = 1,\ldots,m$.

В модели релятивистской струны фигурируют двумерные поверхности, вложенные в трехмерное псевдоевклидово пространство с сигнатурой метрики (+ – –). Поэтому в рассматриваемом случае $c_0 = -c_1 = -c_2 = 1$ [12]. Таким образом,

квадратичные формы используются для полного геометрического описания струны.

Теперь перейдем к описанию псевдоевклидова пространства.

Псевдоевклидово пространство $R^n_{p,q}, p+q=n$ определяется как пространство с координатами $x_1,...,x_n$, в которых «длина» вектора $r=(r^1,...,r^n)$ имеет вид [7]:

$$|r| = \sqrt{\langle r, r \rangle} = \sqrt{\sum_{i=1}^{p} (r^i)^2 - \sum_{j=1}^{q} (r^{p+j})^2}. \qquad (1.14)$$

При n = 4, p = 1 получаем пространство-время специальной теории относительности, т. е. пространство Минковского $R^4_{1,3} = R^4_1$, с координатами x_0, x_1, x_2, x_3; обычно полагают $x_0 = ct$, а x_1, x_2, x_3 – обычные координаты точки в трехмерном пространстве [7].

Величина $|r|^2$ является знаконеопределенной [7]. Векторы r_0, для которых $|r| = 0$, образуют в пространстве R^4_1 конус, называемый изотропным, или световым (рис. 1). Векторы r_+, лежащие внутри этого конуса, имеют положительный квадрат длины $|r|^2 > 0$, и называются времениподобные (рис. 1). Векторы r_-, лежащие вне конуса, имеют отрицательный квадрат длины $|r|^2 < 0$, и называются пространственноподобные (рис. 1.2).

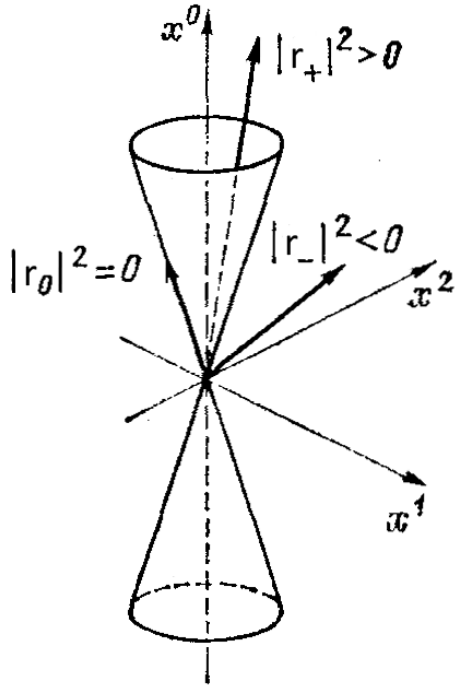

Рис. 1.2 – Изотропный конус в пространстве $R_1^3: (x^0)^2-(x^1)^2-(x^2)^2=0$

Сейчас приступим к методу обратной задачи.

Глава 2 Математическая модель трехмерной струны

2.1 Задача Штурма-Лиувилля

Прямой задачей Штурма-Лиувилля называют задачу исследования спектра соответствующего оператора. Будем рассматривать задачу Штурма-Лиувилля для операторов, определенных на функциях $t(x) \in W_2^2$ пространства Соболева [30].

Дифференциальный оператор $l_w t(x) = -t''(x) + w(x)\,\mathrm{t}(x)$ на множестве $D(l_w)$ функций $t(x) \in W_2^2$ с условием

$$\int_{-\infty}^{+\infty} \left|\mathrm{w}(x)\right|(1+\left|x\right|)dx < \infty \qquad (2.1)$$

где $w(x) \in L_2$ называется оператором Штурма-Лиувилля на всей числовом пространстве Лебега. Функция $w(x)$ в (2.1) является потенциалом оператора l_w [29].

Значение параметра λ, при котором уравнение оператора (2.2)

$$l_w\,\mathrm{t}(x) = \lambda t(x) \qquad (2.2)$$

имеет нетривиальное решение, называется собственным значением оператора l_w, а соответствующее решение – собственной функцией. Множество собственных значений оператора составляет его спектр [30].

Приведем в (2.3) классическую задачу Штурма-Лиувилля при разделенных граничных условиях

$$-t'' + w(x)t = \lambda t, \quad x \in (-\infty, +\infty), \qquad (2.3)$$

Для исследования свойств спектра введем понятие характеристической функции. Обозначим через $\varphi(x,\lambda), \psi(x,\lambda)$ решения уравнения (2.3).

Введем

$$\Delta(\lambda) = \psi(x,\lambda)\varphi'(x,\lambda) - \psi'(x,\lambda)\varphi(x,\lambda). \qquad (2.4)$$

Согласно формуле Остроградского-Лиувилля вронскиан не зависит от x. Функция $\Delta(\lambda)$ – характеристическая функция оператора l_w.

Функция $\Delta(\lambda)$ является целой по λ и имеет не более счетного множества нулей $\{\lambda_n\}$. Из (2.4) следует, что функции $\varphi(x,\lambda_n),\psi(x,\lambda_n)$ являются решениями уравнения $l_w t(x) = \lambda_n t(x)$, т. е. собственными функциями оператора l_w. Следовательно, нули $\{\lambda_n\}$ характеристической функции (2.4) совпадут с собственными значениями оператора l_w. Собственные функции $\varphi(x,\lambda_n),\psi(x,\lambda_n)$ пропорциональны друг другу, т. е. найдется такая последовательность чисел $\{\beta_n\}$, что

$$\psi(x,\lambda_n) = \beta_n \varphi(x,\lambda_n), \quad \beta_n \neq 0. \tag{2.5}$$

Для каждого собственного значения можно подобрать собственную функцию с точностью до постоянного множителя [27].

Числа

$$\alpha_n = \int_0^\pi \varphi^2(x,\lambda_n)dx \tag{2.6}$$

называются весовыми числами, а пара $\{\lambda_n,\alpha_n\}$ называется спектральными данными оператора l_w.

Весовые числа α_n связаны с числами β_n, входящими в формулы (2.5), соотношениями

$$\beta_n \alpha_n = -\dot{\Delta}(\lambda_n), \tag{2.7}$$

где $\dot{\Delta}(\lambda) = \dfrac{d\Delta(\lambda)}{d\lambda}$. Из (2.7) видно, что все нули функции $\Delta(\lambda)$ будут простыми, т. е. $\Delta(\lambda_n) \neq 0$.

Собственные значения $\{\lambda_n\}_{n\geq 0}$ оператора Штурма-Лиувилля l_w на конечном интервале вещественны и имеют на плюс бесконечности предельную точку. При больших n верны асимптотические формулы для λ_n, а также собственных функций $\varphi(x,\lambda_n)$ [28]:

$$\sqrt{\lambda_n} = n + \frac{\omega}{\pi n} + o\left(\frac{1}{n}\right), \tag{2.8}$$

$$\varphi(x,\lambda_n)=\cos\sqrt{\lambda_n}x+O\left(\frac{1}{n}\right), \tag{2.9}$$

где $\omega=\frac{1}{2}\int_0^{\pi}w(y)dy.$

В случае если известен спектр $\{\lambda_n\}_{n\geq 0}$ можно однозначно определить характеристическую функцию $\Delta(\lambda)$ по формуле

$$\Delta(\lambda)=\pi(\lambda_0-\lambda)\prod_{n=1}^{\infty}\frac{\lambda_n-\lambda}{n^2}. \tag{2.10}$$

Заметим, что аналогичные результаты можно получить для операторов Штурма-Лиувилля с другими видами нераспадающихся краевых условий [31].

Рассмотрим, например, оператор $l_w^1y(x)=-t''(x)+w(x)t(x)$ с областью определения $x\in(-\infty,+\infty)$. Все собственные значения $\{\mu_n\}_{n\geq 0}$ оператора l_w^1 являются простыми и совпадают с нулями характеристической функции $d(\lambda)=\varphi(\lambda)$, причем

$$d(\lambda)=\prod_{n=0}^{\infty}\frac{\mu_n-\lambda}{(n+1/2)^2}. \tag{2.11}$$

Для спектральных данных $\{\mu_n,\alpha_n^1\}_{n\geq 0}$, $\alpha_n^1=\int_{-\infty}^{+\infty}\varphi^2(x,\mu_n)dx$ оператора l_w^1 справедливы асимптотические формулы

$$\sqrt{\mu_n}=n+\frac{1}{2}+\frac{\omega^1}{\pi n}+\frac{\beta_n}{n},\quad \{\beta_n\}\in l_2, \tag{2.12}$$

$$\alpha_n^1=\frac{\pi}{2}+\frac{\beta_n^1}{n},\quad \{\beta_n^1\}\in l_2, \tag{2.13}$$

где $\omega^1=\frac{1}{2}\int_{-\infty}^{+\infty}w(y)dy.$

Спектры операторов l_w и l_w^1 связаны соотношениями

$$\lambda_n<\mu_n<\lambda_{n+1},\quad n\geq 0, \tag{2.14}$$

т. е. собственные значения двух операторов l_w и l_w^1 перемежаются.

Система собственных функций $\{\varphi(x,\lambda_n)\}_{n\geq 0}$ оператора l_w полна в L_2.

Пусть $f(x), x \in (-\infty, +\infty),$ – абсолютно непрерывная функция. Тогда

$$f(x) = \sum_{n=0}^{\infty} f_n \varphi(x, \lambda_n), \quad f_n = \frac{1}{\alpha_n} \int_{-\infty}^{\infty} f(y) \varphi(y, \lambda_n) dy, \tag{2.15}$$

причем ряд сходится равномерно на всей числовой прямой.

Для $f(x) \in L_2$ ряд (2.15) сходится в L_2, причем имеет место *равенство Парсеваля*

$$\int_{-\infty}^{\infty} |f(x)|^2 dx = \sum_{n=0}^{\infty} \alpha_n |f_n|^2. \tag{2.16}$$

Обратные задачи Штурма-Лиувилля состоят в восстановлении оператора l_w (т. е. его потенциала $w(x) \in L_2$) по его спектральным характеристикам.

Рассмотрим уравнение $l_w t = \lambda t$ и задачу восстановления оператора l_w по спектральным данным $\{\lambda_n, \alpha_n\}_{n \geq 0}$.

Наряду с l_w рассмотрим оператор $\bar{l}_w$ того же вида, но с потенциалом $\bar{w}(x)$. Если некоторый символ, скажем γ, обозначает объект, относящийся к оператору l_w, то символ $\bar{\gamma}$ будет обозначать аналогичный объект, относящийся к $\bar{l}_w$, а $\tilde{\gamma} = \gamma - \bar{\gamma}$ [32].

Теорема. Если $\lambda_n = \bar{\lambda}_n, \alpha_n = \bar{\alpha}_n$ при всех $n \geq 0$, то $l_w = \bar{l}_w$, т. е. $w(x) = \bar{w}(x)$ почти всюду на $w(x) \in L_2$. Таким образом, задание спектральных данных $\{\lambda_n, \alpha_n\}_{n \geq 0}$ однозначно определяет потенциал [30].

Доказательство (по В. А. Марченко). Представим собственные функции $\varphi(x, \lambda)$ и $\bar{\varphi}(x, \lambda)$ операторов l_w и $\bar{l}_w$, соответственно, в виде

$$\varphi(x, \lambda) = \cos \sqrt{\lambda} x + \int_{-\infty}^{x} G(x, y) \cos \sqrt{\lambda} y \, dy,$$

$$\bar{\varphi}(x, \lambda) = \cos \sqrt{\lambda} x + \int_{-\infty}^{x} \bar{G}(x, y) \cos \sqrt{\lambda} y \, dy.$$

Другими словами,

$$\varphi(x, \lambda) = (E + G) \cos \sqrt{\lambda} x, \quad \bar{\varphi}(x, \lambda) = (E + \bar{G}) \cos \sqrt{\lambda} x,$$

где

$$(E+G)f(x) = f(x) + \int_{-\infty}^{x} G(x,y)f(y)dy,$$

$$(E+\bar{G})f(x) = f(x) + \int_{-\infty}^{x} \bar{G}(x,y)f(y)dy.$$

Разрешая соотношение $\bar{\varphi}(x,\lambda) = (E+\bar{G})\cos\sqrt{\lambda}x$ относительно $\cos\sqrt{\lambda}x$, находим

$$\cos\sqrt{\lambda}x = \bar{\varphi}(x,\lambda) + \int_{-\infty}^{x} \bar{H}(x,y)\bar{\varphi}(y,\lambda)dy,$$

где $\bar{H}(x,y)$ – непрерывная функция, которая является ядром обратного оператора $(E+\bar{H}) = (E+\bar{G})^{-1}$, $\bar{H}f(x) = \int_{-\infty}^{x} \bar{H}(x,y)f(y)dy$. Следовательно,

$$\varphi(x,\lambda) = \bar{\varphi}(x,\lambda) + \int_{-\infty}^{x} Q(x,y)\bar{\varphi}(y,\lambda)dy, \qquad (2.17)$$

где *Q(x,t)* – вещественная непрерывная функция.

Возьмем произвольную функцию $f(x) \in L_2$. Из (2.22) вытекает равенство

$$\int_{-\infty}^{\infty} f(x)\varphi(x,\lambda)dx = \int_{-\infty}^{\infty} g(x)\bar{\varphi}(x,\lambda)dx,$$

где

$$g(x) = f(x) + \int_{x}^{\infty} Q(y,x)f(y)dy.$$

Следовательно, при всех $n \geq 0$ справедливы соотношения

$$f_n = \bar{g}_n$$

$$f_n = \int_{-\infty}^{\infty} f(x)\varphi(x,\lambda_n)dx, \quad \bar{g}_n = \int_{-\infty}^{\infty} g(x)\bar{\varphi}(x,\lambda_n)dx.$$

Используя равенство Парсеваля (2.20), вычисляем

$$\int_{-\infty}^{\infty} |f(x)|^2 dx = \sum_{n=0}^{\infty} \frac{|f_n|^2}{\alpha_n} = \sum_{n=0}^{\infty} \frac{|\bar{g}_n|^2}{\alpha_n} = \sum_{n=0}^{\infty} \frac{|\bar{g}_n|^2}{\tilde{\alpha}_n} = \int_{-\infty}^{\infty} |g(x)|^2 dx,$$

т. е.

$$\|f\|_{L_2} = \|g\|_{L_2}. \qquad (2.18)$$

Рассмотрим оператор

$$Bf(x) = f(x) + \int_{x}^{\infty} Q(y,x)f(y)dy \qquad (2.19)$$

и заметим, что $Bf = g$. В силу (2.6) $\|Bf\|_{L_2} = \|f\|_{L_2}$ при всех $f(x) \in L_2(0,\pi)$. Следовательно, $B^* = B^{-1}$, что возможно лишь при $Q(x,t) \equiv 0$. Таким образом, $\varphi(x,\lambda) = \bar{\varphi}(x,\lambda)$, т. е. $w(x) = \bar{w}(x)$ почти всюду на L_2, ч.т.д.

В симметричном случае, когда $w(x) = w(\pi - x)$, для определения потенциала $w(x)$ достаточно задать только спектр $\{\lambda_n\}_{n\geq 0}$.

Можно переходить к описанию N-солитонных решений обратной задачи рассеяния.

2.2 N-солитонные решения спектральной задачи второго порядка

Будем вести поиск N-солитонных решений в рамках спектральной задачи второго порядка (2.20):

$$t'' + w(x)t = 0, \quad x \in (-\infty, \infty), \tag{2.20}$$

где φ, ψ – решения (2.20), а $t(x) \in W_2^2$.

Поставим потенциал w(x) в класс ограниченных функций в своей области определения. Кроме дискретного спектра, собственные функции которого квадрируемы, появляется дважды вырожденный непрерывный спектр, у которого не выйдет нормирование собственными функциями [27]. Дискретный спектр искомого уравнения при нулевом непрерывном спектре даст N солитонов. В задаче Штурма-Лиувилля (2.4) λ было приравнено к нулю. Для избавления от потенциально бесконечного количества собственных значений (по законам квантовой механики) [27], на потенциал будет наложено дополнительное ограничение:

$$\int_{-\infty}^{+\infty} |\mathrm{w}(x)|(1+|x|)dx < \infty \tag{2.21}$$

Сначала введем коэффициенты прохождения и отражения. Для этого зафиксируем в двумерном пространстве 2 базиса, состоящих из следующих функций Йоста [43]:

$$\begin{matrix} \psi_1(x,k) = e^{-ikx} + o(1) \\ \psi_2(x,k) = e^{ikx} + o(1) \end{matrix} \text{ при } x \to +\infty, \tag{2.22}$$

$$\begin{matrix} \varphi_1(x,k) = e^{-ikx} + o(1) \\ \varphi_2(x,k) = e^{ikx} + o(1) \end{matrix} \text{ при } x \to -\infty. \tag{2.23}$$

Первый из них (ψ_1, ψ_2) (2.22) определяется асимптотическим условием на $+\infty$ по x, второй (φ_1, φ_2) (2.23) – на $-\infty$ по x. Ввиду вещественности потенциала в (2.20) имеем [44]

$$\varphi_1(x,k)=\overline{\varphi}_2(x,k),\ \ \psi_1(x,k)=\overline{\psi}_2(x,k),$$
$$\varphi_1(x,k)=\overline{\varphi}_2(x,-k),\ \ \psi_1(x,k)=\psi_2(x,-k). \tag{2.24}$$

Векторы любого из базисов представляют собой линейную комбинацию векторов другого [27]:

$$\varphi_i(x,k)=\sum_{l=1,2}P_{il}(k)\psi_l(x,k),\quad i=1,2. \tag{2.25}$$

Определенную таким образом (2.25) матрицу *P(k)* будем называть матрицей перехода. Ввиду (2.24) P имеет вид (2.26) [27]

$$P(k)=\begin{pmatrix} a(k) & b(k) \\ \overline{b}(k) & \overline{a}(k) \end{pmatrix}. \tag{2.26}$$

В дальнейшем индекс 1 у функций φ_1 и ψ_1 будем опускаться, и таким образом:

$$\varphi(x,k)=a(k)\psi(x,k)+b(k)\overline{\psi}(x,k). \tag{2.27}$$

Определитель Вронского $W(\varphi_1,\varphi_2)=\varphi_1\frac{d\varphi_2}{dx}-\varphi_2\frac{d\varphi_1}{dx}$ любой пары решений φ_1,φ_2 уравнения (2.20) не станет зависеть от x [27]. Очевидно, $W(\varphi,\overline{\varphi})=W(\psi,\overline{\psi})=2ik$. Это соотношение вместе с формулой (2.27) будет означать, что

$$|a(k)|^2-|b(k)|^2=1, \tag{2.28}$$

т. е. матрица перехода унимодулярна: det P(k) = 1 [27].

Легко видеть, что величины $a^{-1}(k)$ и $b(k)a^{-1}(k)$ представляют собой соответственно коэффициенты прохождения и отражения для волны, падающей на потенциал w(x) справа. В самом деле, асимптотика собственной функции $\varphi(x,k)a^{-1}(k)$ имеет при $x\to+\infty$ вид [45]

$$\frac{\varphi(x,k)}{a(k)}=e^{-ikx}+\frac{b(k)}{a(k)}e^{ikx}+o(1), \tag{2.29}$$

т. е. (2.29) представляет собой суперпозицию падающей (e^{-ikx}) и отраженной ($ba^{-1}e^{ikx}$) волн [27]. На другом конце прямой x

$$\frac{\varphi(x,k)}{a(k)} = \frac{e^{-ikx}}{a(k)} + o(1), \tag{2.30}$$

т. е. имеем в (2.30) прошедшую волну. Иными словами, t(k) = a^{-1}(k) представляет собой амплитуду рассеяния вперед, а r(k) = b(k)a^{-1}(k) – назад. Из (2.28) следует при этом, что рассеяние унитарно (2.31):

$$|t(k)|^2 + |r(k)|^2 = 1. \tag{2.31}$$

Матрица перехода P(k) доставляет нам исчерпывающую информацию о непрерывном спектре оператора Штурма-Лиувилля [29]. Фактически же вся информация о P(k) содержится в коэффициенте отражения $r(k) = \frac{b(k)}{a(k)}$ (который достаточно задать на полуоси k > 0, поскольку $r(-k) = \bar{r}(k)$). Действительно, из (2.28) находим

$$|a(k)|^2 = (1 - |r(k)|^2)^{-1}, \tag{2.32}$$

т. е. модуль r(k) однозначно определяет |a(k)| (2.32) [27]. Зная же нули аналитической в верхней полуплоскости функции a(k), мы можем однозначно определить аргумент последней по ее модулю. Для этого рассмотрим функцию a_1(k):

$$a_1(k) = a(k)\prod_{n=1}^{N} \frac{k + i\kappa_n}{k - i\kappa_n}. \tag{2.33}$$

Эта функция (2.33) по-прежнему аналитична при Im k > 0, но уже не имеет там нулей. Следовательно, $\ln a_1(k)$ аналитичен в верхней полуплоскости и обращается в нуль при $|k| \to \infty$. Причем на вещественной оси выполнится $|a_1(k)| = |a(k)|$ [27]. Дисперсионное соотношение для функции $\ln a_1(k) = \ln|a_1(k)| + i \arg a_1(k)$ дает $\arg a_1(k)$:

$$\arg a_1(k) = -\frac{1}{\pi} \int_{-\infty}^{\infty} \frac{\ln|a(k')}{k' - k}\, dk'. \tag{2.34}$$

Таким образом (2.34),

$$\arg a(k) = \frac{1}{i}\sum_{n=1}^{N} \ln \frac{k-i\kappa_n}{k+i\kappa_n} - \frac{1}{\pi}\int_{-\infty}^{\infty} \frac{\ln|a(k')}{k'-k} dk'. \qquad (2.35)$$

Итак, a(k) восстанавливается по модулю коэффициента отражения (2.35). Функция же b(k) есть просто $r(k)a^{-1}(k)$ [27].

Пусть заданы два натуральных числа N_+, N_-. N_+, N_- войдут в состав количества (2.28) линий сингулярности [34] (точнее, в данном случае, линий, составленных из точек возврата мирового листа планарной струны):

$$N = N_+ + N_- + 1. \qquad (2.36)$$

Обозначим собственные значения, соответствующие дискретному спектру [27], через $\kappa_{n\pm}$, а весовые числа [29] – как $b_{n\pm}$, причем

$$\kappa_{n\pm} > 0, \quad b_{1\pm} > 0, \quad b_{n\pm} = |b_{n\pm}|(-1)^{n\pm-1}, \quad n\pm = \overline{1, N_\pm}. \qquad (2.37)$$

Опишем теперь характеристики дискретного спектра (2.37), естественным образом примыкающие к обратной задаче рассеяния с данными $\{\kappa_n, b_n\}$ [41]. На бесконечности собственные функции для (2.20) имеют асимптотики $\Gamma_{n\pm} \to e^{\mp\kappa_{n\pm}x}; x \to \pm\infty; n\pm = \overline{1, N_\pm}$. Фиксируем функцию $\Gamma_{n\pm}(x)$ ее асимптотикой на $+\infty$ по x:

$$\Gamma_{n\pm}(x) = b_{n\pm}e^{-\kappa_{n\pm}x} + o(e^{-\kappa_{n\pm}x}). \qquad (2.38)$$

При этом при $x \to -\infty$ функции $\Gamma_{n\pm}(x) = e^{\kappa_{n\pm}x} + o(e^{\kappa_{n\pm}x})$ станут вещественны, так как величины $b_{n\pm}$ вещественны. Занумеруем собственные значения $\kappa_{n\pm}$ в порядке убывания [42]: $\kappa_{1\pm} > \kappa_{2\pm} > ... > \kappa_{N_\pm} > 0$.

При этом $\kappa_{1\pm}$ представляет собой энергию основного состояния, $\Gamma_{1\pm}(x)$ – волновую функцию этого состояния без нулей, а $\Gamma_{n\pm}(x)$ – ровно (n – 1) раз проходит через нуль [27]. Поэтому

$$b_n = |b_n|(-1)^{n-1}. \qquad (2.39)$$

Таким образом, после указанного пояснения, весовые числа $b_{n\pm}$ составляют дополнительные по отношению к собственным значениям $\kappa_{n\pm}$ характеристики дискретного спектра [27].

Найдем наборы функций $\Gamma_{\pm}$. Для этого для каждого знака решим СЛАУ [33]

$$\Gamma_n(x) = \frac{b_n e^{-2\kappa_n x}}{a'(i\kappa_n)}\left\{1 + i\sum_{m=1}^{N}\frac{\Gamma_m(x)}{\kappa_m + \kappa_n}\right\}, \tag{2.40}$$

где $a_{\pm}(k) = \prod_{n=1}^{N_{\pm}} \frac{k - i\kappa_{n\pm}}{k + i\kappa_{n\pm}}$ – обратное коэффициенту прохождения число, причем $a_{\pm}(i\kappa_{n\pm}) = 0$, однако $a'_{\pm}(i\kappa_{n\pm}) \neq 0$ при $\kappa_{1\pm} > \kappa_{2\pm} > ... > \kappa_{N_{\pm}} > 0$, поэтому функция $a'_{\pm}(i\kappa_{n\pm})$ не приведет к делению на ноль.

Теперь распишем подробнее, как были получены выражения (2.40) и $a_{\pm}(k)$, а также собственные функции $\mathrm{X}^{\pm}(x,k)$. Для этого продифференцируем уравнение $L\varphi(x,k) = k^2\varphi(x,k)$ по k в точке $k = i\kappa_{n\pm}$ [27]:

$$(\mathrm{L} + \kappa_{n\pm}^2)\varphi'(x, i\kappa_{n\pm}) = 2i\kappa_{n\pm}\varphi(x, i\kappa_{n\pm}). \tag{2.41}$$

Умножим полученное соотношение (2.41) на $\varphi(x, i\kappa_{n\pm})$ и следом проинтегрируем результирующее равенство по всем x [46]. Дважды перебрасывая дифференцирование в интеграле (2.42)

$$\int_{-\infty}^{+\infty} \varphi(x, i\kappa_{n\pm})(\mathrm{L} + \kappa_{n\pm}^2)\varphi'(x, i\kappa_{n\pm})\,\mathrm{dx} \tag{2.42}$$

а также учитывая, что главный член в асимптотике $\varphi(x, i\kappa_{n\pm})$ на $+\infty$ по x имеет вид $a'_{\pm}(i\kappa_{n\pm})e^{\kappa_{n\pm}x}$ (поскольку $a_{\pm}(i\kappa_{n\pm}) = 0$), получим

$$\int_{-\infty}^{+\infty} \varphi^2(x, i\kappa_{n\pm})\,\mathrm{dx} = ia'(i\kappa_{n\pm})\,\mathrm{b}_{n\pm}, \tag{2.43}$$

т. е. $a'_{\pm}(i\kappa_{n\pm}) \neq 0$. Из этой формулы следует также, что величина $ia'(i\kappa_{n\pm})$ вещественна и имеет тот же знак, что и $b_{n\pm}$ [27].

Пусть

$$\frac{X^{+}(x,k)}{a(k)} = X^{-}(x,k), \tag{2.44}$$

причем

$$X^{+}(x,k) = 1 + \int_{-\infty}^{x} \frac{e^{2ik(x-y)}-1}{2ik} w(y) X^{+}(y,k)\,dy, \tag{2.45}$$

$$X^{-}(x,k) = 1 - \int_{x}^{+\infty} \frac{e^{2ik(x-y)}-1}{2ik} w(y) X^{-}(y,k)\,dy. \tag{2.46}$$

Левая часть (2.44) аналитична в верхней полуплоскости *k*, за исключением точек мнимой оси $k = i\kappa_{n\pm}$, в которых собственная функция $X^{+}(x,k)$ (2.45) имеет простые полюсы. Функция $X^{-}(x,k)$ (2.46) аналитична в нижней полуплоскости *k* [27]. Рассмотрим

$$\Phi_{\pm}(x,k) = \begin{cases} \dfrac{X^{+}(x,k)}{a(k)}, & \operatorname{Im} k > 0 \\ X^{-}(x,k), & \operatorname{Im} k < 0 \end{cases}, \tag{2.47}$$

$\Phi_{\pm}(x,k)$ имеет простые полюсы в верхней полуплоскости и скачок на вещественной оси (2.47), равный $r(k)\bar{X}^{-}(x,k)e^{2ikx}$. Во всех остальных точках комплексной плоскости функция $\Phi_{\pm}(x,k)$ аналитична и на бесконечности имеет предельным значением единицу. Очевидно, что функция $\Phi_{\pm}(x,k)$ может быть представлена в виде [47]

$$\Phi_{\pm}(x,k) = 1 + \sum_{n=1}^{N_{\pm}} \frac{\Gamma_{n\pm}(x)}{k - i\kappa_{n\pm}} + \frac{1}{2\pi i} \int_{-\infty}^{+\infty} \frac{r(q)\bar{X}^{\pm}(x,q)e^{2iqx}}{q-k}\,dq \quad (\operatorname{Im} k \neq 0), \tag{2.48}$$

где $\Gamma_{n\pm}(x)$ нужны для вычета функции $\Phi_{\pm}(x,k)$ в точке $k = i\kappa_{n\pm}$.

Величину $\Gamma_n(x)$ можно связать [27] со значением $\Phi_{\pm}(x,k)$ в точке нижней полуплоскости $k = -i\kappa_{n\pm}$ (т. е. с $\bar{X}^{-}(x,-i\kappa_{n\pm})$). В самом деле (2.49),

$$\Gamma_n = \frac{\varphi(x,i\kappa_n)e^{-\kappa_n x}}{a'(i\kappa_n)} = \frac{b_n\psi(x,-i\kappa_n)e^{-\kappa_n x}}{a'(i\kappa_n)} = \frac{b_n}{a'(i\kappa_n)}X^{-}(x,-i\kappa_n)e^{-2\kappa_n x}. \qquad (2.49)$$

Определяя $X^{-}(x,-i\kappa_{n\pm})$ с помощью (2.48), находим (2.50)

$$\Gamma_n(x) = \frac{b_n e^{-2\kappa_n x}}{a'(i\kappa_n)}\left\{1 + i\sum_{m=1}^{N}\frac{\Gamma_m(x)}{\kappa_m+\kappa_n} + \frac{1}{2\pi i}\int_{-\infty}^{+\infty}\frac{r(q)\bar{X}^{-}(x,q)e^{2iqx}}{q+i\kappa_{n\pm}}dq\right\}. \qquad (2.50)$$

Устремим теперь *k* в формуле (2.48) к вещественной оси снизу (2.51):

$$X^{\pm}(x,k) = 1 + \sum_{m=1}^{N_{\pm}}\frac{\Gamma_{m\pm}(x)}{k-i\kappa_{m\pm}} + \frac{1}{2\pi i}\int_{-\infty}^{+\infty}\frac{r(q)\bar{X}^{\pm}(x,q)e^{2iqx}}{q-k}dq. \qquad (2.51)$$

Замечательной особенностью солитонов является отсутствие при их столкновении неупругих эффектов. Т. к. требуется найти солитонное решение, то коэффициент отражения $r(k) = 0$, и $\mathrm{X}^{\pm}$ примет вид [33]:

$$\mathrm{X}^{\pm}(x,k) = 1 + \sum_{m=1}^{N_{\pm}}\frac{\Gamma_{m\pm}(x)}{k-i\kappa_{m\pm}}. \qquad (2.52)$$

Уравнения (2.40), (2.52) однозначно разрешимы. В качестве собственных значений $\kappa_{n\pm}$ можно выбирать любые положительные числа $\kappa_{1\pm} > \kappa_{2\pm} > \ldots > \kappa_{N_{\pm}} > 0$, и к формуле в b_n можно подобрать положительные $|b_n|$ [27].

Определим матрицу решений (2.20) *T(x)* [28, 32] через собственные функции $\mathrm{X}^{\pm}$:

$$T(x)=\begin{pmatrix} t_{11+}(x)=-i\dfrac{\partial}{\partial k}\mathrm{X}^{+}(x,k)\Big|_{k=0} & t_{12+}(x)=\mathrm{X}^{+}(x,0) \\ t_{21-}(x)=i\dfrac{\partial}{\partial k}\mathrm{X}^{-}(x,k)\Big|_{k=0} & t_{22-}(x)=\mathrm{X}^{-}(x,0) \end{pmatrix}. \tag{2.53}$$

Эта матрица содержит линейно-независимые решения дифференциального уравнения (2.20). Заметим, что первый столбец в (2.53) будут представлять собой формулы электромагнитных импульсов, представленные в физическом устройстве как пара электродов [48].

Решение (2.53) должно удовлетворять следующим вронскианам (2.54), (2.55):

$$\mathrm{W}(t_{12+}(x),t_{11+}(x))=-1, \tag{2.54}$$

$$\mathrm{W}(t_{22-}(x),t_{21-}(x))=1. \tag{2.55}$$

После того, как мы выберем элементы матриц *T(x)* [32] для обоих знаков в виде (2.53), запишем выражение, определяющее коэффициенты первой квадратичной формы струны:

$$F(t,x)=t_{11+}(x+t)\,t_{22-}(x-t)-t_{12+}(x+t)\,t_{21-}(x-t). \tag{2.56}$$

Нули функция (2.56) – т.е. особенности первой квадратичной формы – и определяют эволюционирующие точки возврата. Аргументы *(x + t)* и *(x - t)* здесь выступают в качестве конусных переменных.

При каждом значении *t* из набора ищем нули $x_j(t)$ функции *F* [29]:

$$F(t,x_j(t))=0, \tag{2.57}$$

где $j=\overline{1,\textit{количество нулей функции}}$.

Представим краткую последовательность действий в решении обратной спектральной задачи рассеяния второго порядка в виде следующей схемы (2.58) [27]:

$$w(x)\xrightarrow{I}\Gamma_{n\pm}(x)\xrightarrow{II}T(x)\xrightarrow{III}F(t,x(t))=0. \tag{2.58}$$

Первый этап в схеме (2.58) состоит в вычислении данных рассеяния $\Gamma_{n\pm}(x)$ по начальному условию $\{\kappa_{n\pm}, b_{n\pm}\}$ с помощью решения уравнений на собственные функции оператора Штурма-Лиувилля с потенциалом w(x) [27]. Второй этап заключается в решении задачи Коши *T(x)* в терминах данных рассеяния. Наконец, на последнем этапе схемы (2.58) необходимо указать потенциал $F(t,x)$ в операторе Штурма-Лиувилля, имеющий *T(x)* в качестве данных рассеяния и условие $F(t,x(t))=0$, т. е. решить обратную задачу рассеяния [27].

Следует напомнить, что через решение спектральной задачи второго порядка уравнения Штурма-Лиувилля нужно найти точки возврата, то есть каспы (2.57), на мировом листе планарной струны (рис. 2.1), которая была представлена в (2.56). Энионы и будут этими искомыми точками возврата. Каспы следует объединить в мировую линию, которая будет рассматриваться во времени-пространстве Минковского $E_{1,2}$. Пример планарной струны будет представлен на рис. 2.2:

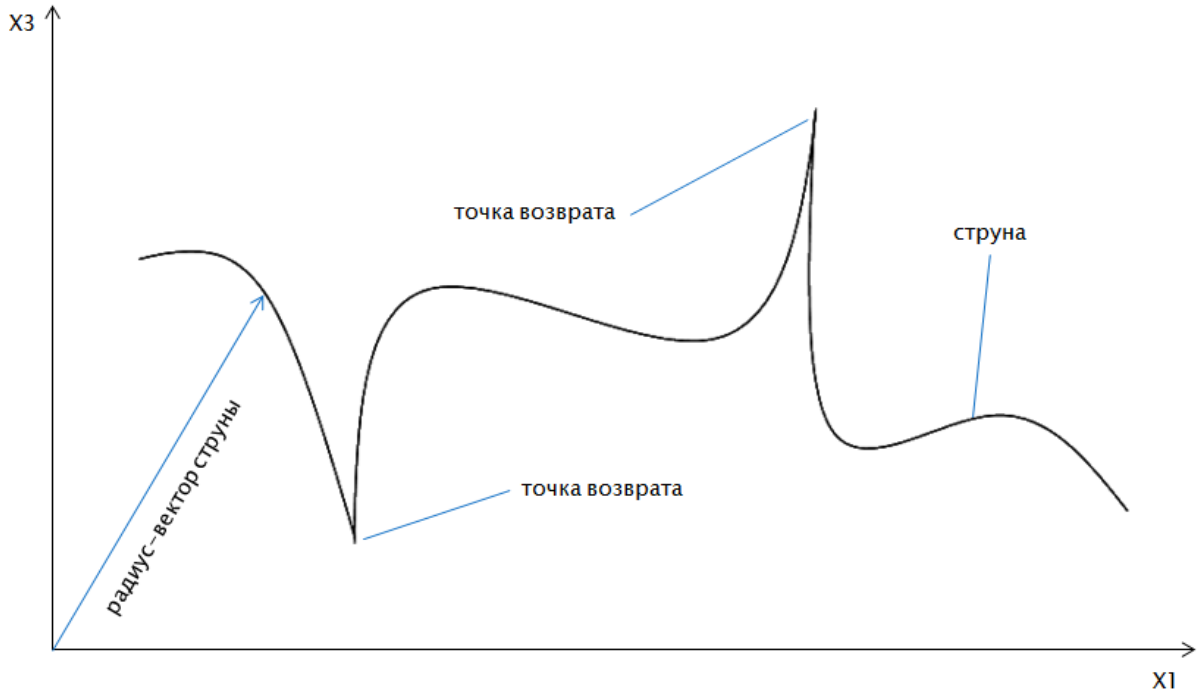

Рис. 2.2 – Планарная струна с представленными точками возврата и радиус-вектором

Теперь строим графики для представления полученного набора значений для каждого $x_j(t)$. $F(t, x(t))$, по сути, является срезом по времени мирового листа (рис. 2.1, 2.2), а каждый график – двумерной проекцией мировой линии точки возврата $x_j(t)$, представляющий собой искомый энион [29].

В пространстве $E_{1,2}$ построим радиус-вектор струны [28] (по умолчанию будем принимать $p = 1, \overline{X_0} = \overline{0}$):

$$\overline{X}(t,x) = \overline{X_0} + p\int_0^{t+x} \overline{e_+}(\eta)d\eta - p\int_0^{t-x} \overline{e_-}(\eta)d\eta\,, \qquad (2.59)$$

где

$$\overline{e_\pm} = \tfrac{1}{2}(t_{i1\pm}^2 + t_{i2\pm}^2)\overline{b_0} - (t_{i1\pm}\cdot t_{i2\pm})\overline{b_1} - \tfrac{1}{2}(t_{i1\pm}^2 - t_{i2\pm}^2)\overline{b_3} \quad (i = \tfrac{3\mp 1}{2}). \qquad (2.60)$$

Остановимся на (2.60). $\overline{e_\pm}$ – светоподобные векторы, которые можно достроить до ортонормированного базиса в пространстве $E_{1,3}$ [28].

В [28] подробно написано, как можно прийти к формуле (2.61):

$$\bar{e}_{\pm} = \frac{1}{2}(|t_{i1\pm}|^2 + |t_{i2\pm}|^2)\bar{b}_0 - \mathrm{Re}(t_{i1\pm}\bar{t}_{i2\pm})\bar{b}_1 - \\ - \mathrm{Im}(t_{i1\pm}\bar{t}_{i2\pm})\bar{b}_2 - \frac{1}{2}(|t_{i1\pm}|^2 - |t_{i2\pm}|^2)\bar{b}_3 \quad , \quad i = \frac{3 \mp 1}{2} \; . \tag{2.61}$$

Струны будут рассматриваться в пространстве Минковского $E_{1,2}$. Следовательно, координата по направлению $\bar{b}_2$ будет равна нулю, поэтому её можно отбросить из расчетов. Это достигается рассмотрением только вещественных функций t(x).

В (2.59) подставляем полученные точки возврата $x_j(t)$. Теперь в пространстве $E_{1,2}$ построим график, представляющий мировые линии энионов по формуле (2.62) [28]:

$$\overline{X}^{(j)}(t) = \overline{X}(t, x_j(t)), \tag{2.62}$$

где $j = \overline{1, \textit{количество нулей функции}}$.

Приведем в пример случай данных рассеяния $N_+ = N_- = 1$.

2.3 Разбор простого случая N=3

В качестве элементарного примера разберем случай $N_+ = N_- = 1$.

Пусть определены данные рассеяния $\{\kappa_\pm, b_\pm\}$ (исходные данные имеют только положительные значения).

Найдем $a'_\pm(i\kappa_\pm)$:

$$a_+(k) = \frac{k - i\kappa_+}{k + i\kappa_+}, a_-(k) = \frac{k - i\kappa_-}{k + i\kappa_-}. \quad (2.63)$$

Подставив $i\kappa_\pm$ в (2.63), получаем $a'_\pm(i\kappa_{n\pm}) = \frac{1}{2i\kappa_\pm}$.

Найдем собственные функции $\Gamma_\pm(x)$ ($a'_\pm(i\kappa_\pm)$ уже учтено):

$$\Gamma_\pm(x) = 2i\kappa_\pm b_\pm e^{-2\kappa_\pm x}\left(1 + \frac{\Gamma_\pm(x)}{2\kappa_\pm}\right). \quad (2.64)$$

Решение (2.64) – $\Gamma_\pm(x) = \frac{2i\kappa_\pm b_\pm}{b_\pm + e^{2\kappa_\pm x}}$.

Теперь вычислим $\mathrm{X}^\pm(x,k) = 1 + \frac{\Gamma_\pm(x)}{k - i\kappa_\pm}$:

$$\mathrm{X}^\pm(x,k) = 1 + \frac{1}{k - i\kappa_\pm} \cdot \frac{2i\kappa_\pm b_\pm}{b_\pm + e^{2\kappa_\pm x}}. \quad (2.65)$$

Для облегчения расчетов после (2.65), укажем промежуточные переменные:

$$\mathrm{X}^\pm(x,0) = \frac{e^{2\kappa_\pm x} - b_\pm}{e^{2\kappa_\pm x} + b_\pm}, \quad (2.66)$$

$$i\mathrm{X}_k^\pm(x,0) = \frac{-2b_\pm}{\kappa_\pm(b_\pm + e^{2\kappa_\pm x})}. \quad (2.67)$$

Теперь возьмемся за матрицу T(x):

$$T(x)=\begin{pmatrix} t_{11+}(x)=-i\mathrm{X}_k^+(x,0)-\mathrm{x}\,\mathrm{X}^+(x,0) & t_{12+}(x)=\mathrm{X}^+(x,0) \\ t_{21-}(x)=i\mathrm{X}_k^-(x,0)+\mathrm{x}\,\mathrm{X}^-(x,0) & t_{22-}(x)=\mathrm{X}^-(x,0) \end{pmatrix}. \qquad (2.68)$$

Подставив (2.66), (2.67) в (2.68), получим:

$$T(x)=\begin{pmatrix} t_{11+}(x)=\dfrac{2b_+}{\kappa_+(b_++e^{2\kappa_+x})}-x\dfrac{e^{2\kappa_+x}-b_+}{e^{2\kappa_+x}+b_+} & t_{12+}(x)=\dfrac{e^{2\kappa_+x}-b_+}{e^{2\kappa_+x}+b_+} \\ t_{21-}(x)=-\dfrac{2b_-}{\kappa_-(b_-+e^{2\kappa_+x})}+x\dfrac{e^{2\kappa_-x}-b_-}{e^{2\kappa_-x}+b_-} & t_{22-}(x)=\dfrac{e^{2\kappa_-x}-b_-}{e^{2\kappa_-x}+b_-} \end{pmatrix}. \qquad (2.69)$$

Удостоверимся, что (2.69) удовлетворяет соответствующим вронскианам $\mathrm{W}(t_{12+}(x),t_{11+}(x))=-1$ и $\mathrm{W}(t_{22-}(x),t_{21-}(x))=1$.

Формула для $F(t,x)$ выглядит следующим образом:

$$F(t,x)=-\frac{2b_+}{\kappa_+}(e^{2\kappa_-(x-t)}-b_-)+\frac{2b_-}{\kappa_-}(e^{2\kappa_+(x+t)}-b_+)+2t(e^{2\kappa_+(x+t)}-b_+)(e^{2\kappa_-(x-t)}-b_-). \qquad (2.70)$$

Для (2.70) были найдены $x_1(t)$, $x_2(t)$, $x_3(t)$ (изображены на рис. 2.3), которые были вставлены в (2.59) (рис. 2.4).

Приступим к созданной программе и полученным с помощью неё результатам вычислений математической модели.

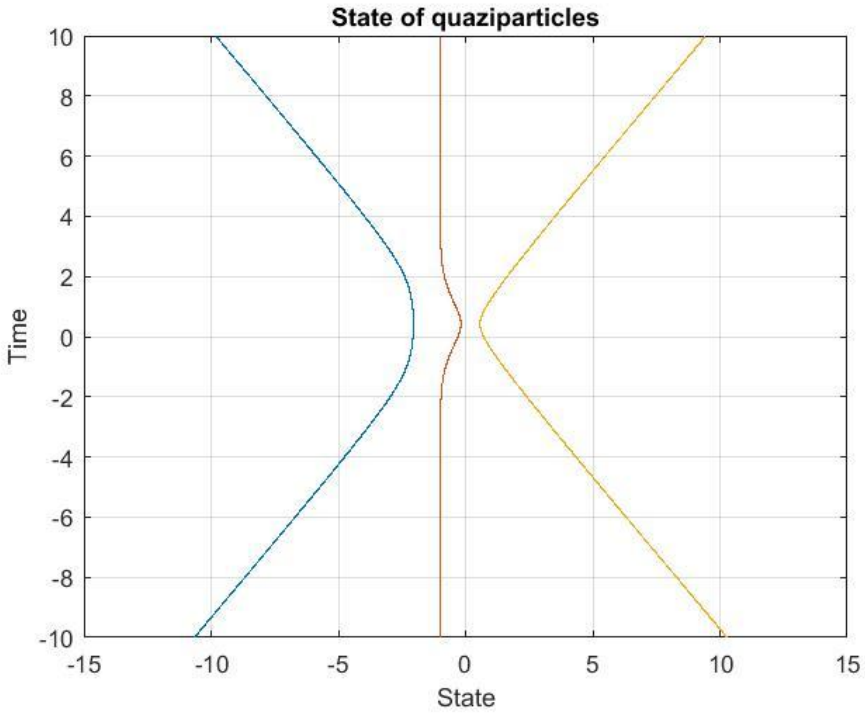

Рис. 2.3 – График точек возврата функции F при N_+=1, N_-=1

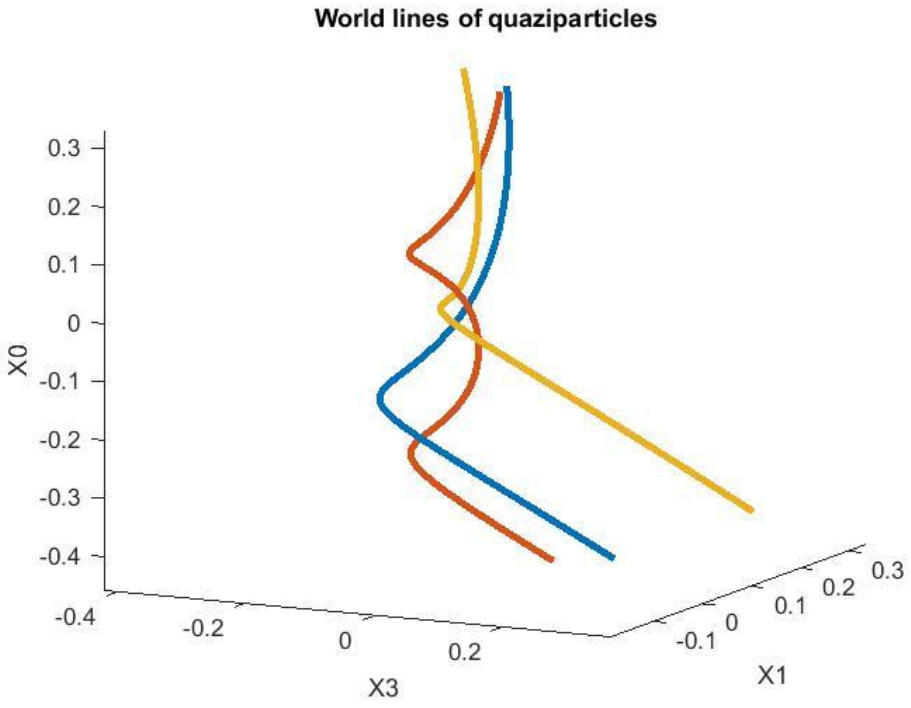

Рис. 2.4 – График мировых линий энионов при N_+=1, N_-=1

Глава 3. Практическое вычисление моделируемого объекта

3.1 Листинг программы диссертации

В качестве высокоуровневого языка для разработки программного продукта (рис. 3.1) был использован MATLAB R2016a. Основной особенностью языка MATLAB являются его широкие возможности по работе с матрицами [50].

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% model1_launch.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Step 1,2

switch 1
    case 1
        F_zero = @find_F_zero;
    case 2
        F_zero = @FFZ_w_o_line_interp;
    case 3
        F_zero = @FFZ_bot_to_top;
end

switch 1
    case 1
        det_T = @determinate_T;
    case 2
        det_T = @determinate_T_alt;
end

[filename,dirname] = uigetfile('*.*', 'Choose file input...');
[const_plus, const_minus] = data_load( cat(2, dirname, filename) );
[x_interval, t_interval] = form_intervals(2000, -6.0, 6.0, -10.0,
10.0);
```

```
% Step 3-10
[x_i, x0, x1, x3, vX] = model1(const_plus, const_minus, x_interval,
...
    t_interval, F_zero, det_T);

draw_x_i(t_interval, x_i);
draw_iX(x0, x1, x3);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% model1.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [x_i, x0, x1, x3, vX] = model1(const_plus, const_minus, ...
    x_interval, t_interval, F_zero, det_T)
% MODEL1 Start relativity string modelling
%   [X_I, X0, X1, X3, VX] = MODEL1(CONST_PLUS, CONST_MINUS, ...
%    X_INTERVAL, T_INTERVAL, F_ZERO)



% Step 3
da_plus = create_func_da(const_plus);
da_minus = create_func_da(const_minus);

% Step 4
gamma_plus = create_func_gamma(const_plus, da_plus, x_interval);
gamma_minus = create_func_gamma(const_minus, da_minus, x_interval);

% Step 5
khi_plus = @(x, k)khi(const_plus, gamma_plus, x, k);
khi_minus = @(x, k)khi(const_minus, gamma_minus, x, k);
```

```
% Step 6
pieces = 1500;
poolobj = gcp();
tol = 1e-4;
plus_zero = find_khi_zero(khi_plus, x_interval, pieces, tol);
minus_zero = find_khi_zero(khi_minus, x_interval, pieces, tol);
if(isempty(plus_zero) || isempty(minus_zero))
    disp('Zeros of khi haven`t been found.');
    return;
end

d_plus = @(x, k)dKhi(const_plus, gamma_plus, x, k);
d_minus = @(x, k)dKhi(const_minus, gamma_minus, x, k);
t_matr = create_functional_t(x_interval, khi_plus, khi_minus, d_plus,
d_minus);

% Step 7
[f_plus, f_minus] = det_T(t_matr);
F_func = @(t, x)(f_minus(t, x) - f_plus(t, x));

% Step 8,9
stationary_points = sort(uniquetol([(x_interval(1) + t_interval(1))
...
  plus_zero minus_zero (x_interval(end) + t_interval(end))], tol)) +
1e-10;

wave_count = length(stationary_points) - 1;
asymptote_points = cell(1, wave_count + 1);
for idx = 1:length(stationary_points)
    if(sum(ismembertol(stationary_points(idx), plus_zero, tol)) > 0)
        asymptote_points{idx} = @(t)(stationary_points(idx) - t);
    elseif(sum(ismembertol(stationary_points(idx), minus_zero, tol))
> 0)
```

```
        asymptote_points{idx} = @(t)(stationary_points(idx) + t);
    else
        asymptote_points{idx} = @(t)(stationary_points(idx) .* ...
            (1.0 + sqrt( 1.0 + abs(t) ) ) );
    end
end

x_i = zeros(wave_count, length(t_interval));
exitflag = zeros(wave_count, length(t_interval));
zero_vect = zeros(1,3);
vX = create_func_vX(t_matr, zero_vect, 1.0);
x0 = zeros(size(x_i));
x1 = zeros(size(x_i));
x3 = zeros(size(x_i));

for idx = wave_count:-1:1
    param = struct;
    param.idx = idx;
    param.asymptote = asymptote_points;
    param.center = ceil(length(t_interval)./2);
    param.tol = tol;
    fObj(idx) = parfeval(poolobj, F_zero, 2, ...
       t_interval, F_func, param );
end
for idx = wave_count:-1:1
   [completedIdx,value,extval] = fetchNext(fObj);
   x_i(completedIdx,:) = value;
   exitflag(completedIdx,:) = extval;
   fObj2(completedIdx)   =   parfeval(poolobj,   vX,   3,   t_interval,
x_i(completedIdx,:));
end

% Step 10
```

```
for idx = 1:wave_count
   [completedIdx,v0,v1,v3] = fetchNext(fObj2);
   x0(completedIdx,:) = v0;
   x1(completedIdx,:) = v1;
   x3(completedIdx,:) = v3;
end

clear fObj fObj2

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% khi.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function khi_val = khi(const, gamma, x, k)
% KHI Wave function.
%    KHI = KHI(CONSTANTS, GAMMA, X, K)
%    Calculate wave limited function khi.
%
%    Constants are:
%    N+, N- -- quantities of number sets. Must be positive integers.
%
%    Happa+, Happa- -- eigenvalues. Must be positive.
%
%    b+, b- -- factor before reversed exponent.
%    Elements with odd index must be positive, even -- negative.
%
%    See also CREATE_FUNC_GAMMA, class CONST.
khi_val = 1.0;
for n = 1:const.N
  khi_val = khi_val + gamma{n}(x) ./ (k - 1i .* const.happa(n));
end
```

```
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% getNearest.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [a,b] = getNearest(asymptote, init, ti)
% GETNEAREST Find nearest to point
%   [A,B] = GETNEAREST(ASYMPTOTE, INIT, TI)
% Find nearest bounds from asymptotes to the point.
    for i=length(asymptote):-1:1
        asympt_temp(i) = asymptote{i}(ti);
    end

    a = asympt_temp(asympt_temp < init) - init;
    if(~isempty(a))
        a = min(abs(a)) .* sign(a) + init;
        a = a(1);
    else
        a = min(asympt_temp);
    end

    b = asympt_temp(asympt_temp > init) - init;
    if(~isempty(b))
        b = min(abs(b)) .* sign(b) + init;
        b = b(1);
    else
        b = max(asympt_temp);
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% form_intervals.m
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function                                [x_m,t_m]                                =
form_intervals(step_size,x_start,x_end,t_start,t_end)
%FORM_INTERVALS Get square cell table of time and distance
%   [X_M,T_M] = FORM_INTERVALS(STEP_SIZE,X_START,X_END,T_START,T_END)
%   Get time and distance and divide them to equal intervals each.
assert(step_size > 0.0, 'Nonpositive step.')
assert(isnumeric([x_start, x_end, t_start, t_end]), ...
    'Non-numeric input parameters: %s \n %s \n %s \n %s', ...
    x_start, x_end, t_start, t_end)
assert(isfinite(x_start + x_end + t_start + t_end), ...
    'Invalid number input parameters: %s \n %s \n %s \n %s', ...
    x_start, x_end, t_start, t_end)
x_step = abs(x_end - x_start) / step_size;
t_step = abs(t_end - t_start) / step_size;

x_m = x_start:x_step:x_end;
t_m = t_start:t_step:t_end;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% find_khi_zero.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function res = find_khi_zero(khi, x_interval, pieces, tol)
%FIND_KHI_ZERO Get zeros of khi function.
%   RES = FIND_KHI_ZERO(KHI, X_INTERVAL, CONST)
%   See also KHI, class CONST.
k_interval = linspace(x_interval(1), x_interval(end), pieces);
opts = optimoptions('fsolve', 'Display', 'none');
temp_arr = nan(1,length(k_interval));
parfor a = 1:length(temp_arr)
```

```
    [temp,~,extfl] = fsolve(@(x)khi(x, 0.0), k_interval(a), opts);
    if(extfl >= 0)
        temp_arr(a) = temp;
    end
end
temp_arr(isnan(temp_arr)) = [];
temp_arr = uniquetol(temp_arr, tol);
temp_arr = temp_arr(temp_arr >= x_interval(1));
res = temp_arr(temp_arr <= x_interval(end));

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% find_F_zero.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [x_i, exitflag] = find_F_zero(t_interval, F_func, param)
% FIND_F_ZERO Finds x(t) zeros.
%    (x_i(t)) = FIND_F_ZERO(T_INTERVAL, F, PARAM)
%   Finds x(t) zeros from expression F(t,x_i(t)) = 0.
%   _i lies in [1, quantity_of_x_zeros].
%   Can correct some inaccuracies in result and
%   can try another point if failed.
%
%    See also CREATE_FUNCTIONAL_T, CREATE_FUNC_F.

cnt = param.center;

init = (param.asymptote{param.idx}(t_interval(cnt)) + ...
     param.asymptote{param.idx + 1}(t_interval(cnt)) ) ./ 2.0;

[a_init,     b_init]     =     getNearest(param.asymptote,     init,
t_interval(cnt));
```

```
if param.tol < 1.0
    softtol = sqrt(param.tol);
else
    softtol = power(param.tol, 1.5);
end

s = length(t_interval);
opts = optimset('Display', 'none', 'TolFun', param.tol, ...
    'TolX', param.tol, 'DiffMaxChange', param.tol);
x_i = zeros(1,s);
exitflag = zeros(1,s);
noise = init .* softtol .* length(param.asymptote);

[x_i(cnt), ~, ~, exitflag(cnt)] = lsqnonlin( ...
    @(x)F_func(t_interval(cnt), x), init, a_init, b_init, opts);
[x_i(cnt + 1), ~, ~, exitflag(cnt + 1)] = ...
    lsqnonlin(@(x)F_func(t_interval(cnt + 1), x), ...
    x_i(cnt) + t_interval(cnt + 1) .* noise, a_init, b_init, opts);
[x_i(cnt - 1), ~, ~, exitflag(cnt - 1)] = ...
    lsqnonlin(@(x)F_func(t_interval(cnt - 1), x), ...
    x_i(cnt) + t_interval(cnt - 1) .* noise, a_init, b_init, opts);
threshhold = s - cnt;

for i=2:threshhold

% Bottom half
k = cnt - i;
[a, b] = getNearest(param.asymptote, x_i(k + 1), t_interval(k));
[x_i(k), ~, ~, exitflag(k)] = lsqnonlin(@(x)F_func(t_interval(k), x),
...
    x_i(k + 1) + t_interval(k) .* noise, a, b, opts);
```

```
if(abs(x_i(k) - x_i(k + 1)) > softtol)
[a, b] = getNearest(param.asymptote, x_i(k + 2), t_interval(k + 1));
x_i(k) = lsqnonlin(@(x)F_func(t_interval(k), x), ...
    x_i(k + 1) + t_interval(k) .* noise, a, b, opts);
exitflag(k) = 4;

    if(abs(x_i(k) - x_i(k + 1)) > softtol)
    x_i(k) = lsqnonlin(@(x)F_func(t_interval(k), x), ...
     x_i(k + 1) - t_interval(k) .* noise, a - softtol, b + softtol,
opts);
    exitflag(k) = 5;

        if(abs(x_i(k) - x_i(k + 1)) > softtol)
            x_i(k) = x_i(k + 1) + (x_i(k + 2) - x_i(k + 3));
            exitflag(k) = 6;
        end

    end

end

% Top half
k = cnt + i;
[a, b] = getNearest(param.asymptote, x_i(k - 1), t_interval(k));
[x_i(k), ~, ~, exitflag(k)] = lsqnonlin(@(x)F_func(t_interval(k), x),
...
    x_i(k - 1) + t_interval(k) .* noise, a, b, opts);

if(abs(x_i(k) - x_i(k - 1)) > softtol)
[a, b] = getNearest(param.asymptote, x_i(k - 2), t_interval(k - 1));
x_i(k) = lsqnonlin(@(x)F_func(t_interval(k), x), ...
    x_i(k - 1) + t_interval(k) .* noise, a, b, opts);
exitflag(k) = 4;
```

```
    if(abs(x_i(k) - x_i(k - 1)) > softtol)
    x_i(k) = lsqnonlin(@(x)F_func(t_interval(k), x), ...
     x_i(k - 1) - t_interval(k) .* noise, a - softtol, b + softtol,
opts);
    exitflag(k) = 5;

        if(abs(x_i(k) - x_i(k - 1)) > softtol)
            x_i(k) = x_i(k - 1) + (x_i(k - 2) - x_i(k - 3));
            exitflag(k) = 6;
        end

    end

end

end % for i=1:threshhold

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FFZ_w_o_line_interp.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [x_i, exitflag] = FFZ_w_o_line_interp(t_interval, F_func,
param)
% FFZ_W_O_LINE_INTERP Finds x(t) zeros.
%    [X_I, EXITFLAG] = FFZ_W_O_LINE_INTERP(T_INTERVAL, F_FUNC, PARAM)
%   Finds x(t) zeros from expression F(t,x_i(t)) = 0.
%   _i lies in [1, quantity_of_x_zeros].
%   Can correct some inaccuracies in result.
%
%    See also CREATE_FUNCTIONAL_T, CREATE_FUNC_F.
```

```
cnt = param.center;

init = (param.asymptote{param.idx}(t_interval(cnt)) + ...
     param.asymptote{param.idx + 1}(t_interval(cnt)) ) ./ 2.0;

[a_init,     b_init]     =     getNearest(param.asymptote,     init,
t_interval(cnt));

if param.tol < 1.0
    softtol = sqrt(param.tol);
else
    softtol = power(param.tol, 1.5);
end

s = length(t_interval);
opts = optimset('Display', 'none', 'TolFun', param.tol, ...
    'TolX', param.tol, 'DiffMaxChange', param.tol);
x_i = zeros(1,s);
exitflag = zeros(1,s);
noise = init .* softtol .* length(param.asymptote);

[x_i(cnt), ~, ~, exitflag(cnt)] = lsqnonlin( ...
    @(x)F_func(t_interval(cnt), x), init, a_init, b_init, opts);
[x_i(cnt + 1), ~, ~, exitflag(cnt + 1)] = ...
    lsqnonlin(@(x)F_func(t_interval(cnt + 1), x), ...
    x_i(cnt) + t_interval(cnt + 1) .* noise, a_init, b_init, opts);
[x_i(cnt - 1), ~, ~, exitflag(cnt - 1)] = ...
    lsqnonlin(@(x)F_func(t_interval(cnt - 1), x), ...
    x_i(cnt) + t_interval(cnt - 1) .* noise, a_init, b_init, opts);
threshhold = s - cnt;

for i=2:threshhold
```

```
% Bottom half
k = cnt - i;
[a, b] = getNearest(param.asymptote, x_i(k + 1), t_interval(k));
[x_i(k), ~, ~, exitflag(k)] = lsqnonlin(@(x)F_func(t_interval(k), x),
...
    x_i(k + 1) + t_interval(k) .* noise, a, b, opts);

if(abs(x_i(k) - x_i(k + 1)) > softtol)
[a, b] = getNearest(param.asymptote, x_i(k + 2), t_interval(k + 1));
x_i(k) = lsqnonlin(@(x)F_func(t_interval(k), x), ...
    x_i(k + 1) + t_interval(k) .* noise, a, b, opts);
exitflag(k) = 4;

    if(abs(x_i(k) - x_i(k + 1)) > softtol)
    x_i(k) = lsqnonlin(@(x)F_func(t_interval(k), x), ...
     x_i(k + 1) - t_interval(k) .* noise, a - softtol, b + softtol,
opts);
    exitflag(k) = 5;

    end

end

% Top half
k = cnt + i;
[a, b] = getNearest(param.asymptote, x_i(k - 1), t_interval(k));
[x_i(k), ~, ~, exitflag(k)] = lsqnonlin(@(x)F_func(t_interval(k), x),
...
    x_i(k - 1) + t_interval(k) .* noise, a, b, opts);

if(abs(x_i(k) - x_i(k - 1)) > softtol)
[a, b] = getNearest(param.asymptote, x_i(k - 2), t_interval(k - 1));
```

```
x_i(k) = lsqnonlin(@(x)F_func(t_interval(k), x), ...
    x_i(k - 1) + t_interval(k) .* noise, a, b, opts);
exitflag(k) = 4;

    if(abs(x_i(k) - x_i(k - 1)) > softtol)
    x_i(k) = lsqnonlin(@(x)F_func(t_interval(k), x), ...
     x_i(k - 1) - t_interval(k) .* noise, a - softtol, b + softtol,
opts);
    exitflag(k) = 5;

    end

end

end % for i=1:threshhold

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FFZ_bot_to_top.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [x_i, exitflag] = FFZ_bot_to_top(t_interval, F_func, param)
% FFZ_BOT_TO_TOP Finds x(t) zeros.
%    [X_I, EXITFLAG] = FFZ_BOT_TO_TOP(T_INTERVAL, F_FUNC, PARAM)
%   Finds x(t) zeros from expression F(t,x_i(t)) = 0.
%   _i lies in [1, quantity_of_x_zeros].
%
%    See also CREATE_FUNCTIONAL_T, CREATE_FUNC_F.

init = (param.asymptote{param.idx}(t_interval(1)) + ...
     param.asymptote{param.idx + 1}(t_interval(1)) ) ./ 2.0;
```

```
[a, b] = getNearest(param.asymptote, init, t_interval(1));

if param.tol < 1.0
    softtol = sqrt(param.tol);
else
    softtol = power(param.tol, 1.5);
end

s = length(t_interval);
opts = optimset('Display', 'none', 'TolFun', param.tol, ...
    'TolX', param.tol, 'DiffMaxChange', param.tol);
x_i = zeros(1,s);
exitflag = zeros(1,s);
noise = init .* softtol .* length(param.asymptote);

[x_i(1), ~, ~, exitflag(1)] = lsqnonlin( ...
    @(x)F_func(t_interval(1), x), init, a, b, opts);

for k=2:s

[a, b] = getNearest(param.asymptote, x_i(k - 1), t_interval(k));
[x_i(k), ~, ~, exitflag(k)] = lsqnonlin(@(x)F_func(t_interval(k), x),
...
    x_i(k - 1) + t_interval(k) .* noise, a, b, opts);

end % for i=1:threshhold

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% draw_x_i.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
function draw_x_i(t_interval, x_i)
% DRAW_X_I Draw x_i(t) on the 2D-plane.
% DRAW_X_I(T_INTERVAL, X_I)
%
% See also CREATE_FUNCTIONAL_T, FIND_F_ZERO.

figure
plot(x_i, t_interval)
xlabel('State')
ylabel('Time')
grid on
title('State of quaziparticles')
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% draw_iX.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function draw_iX(x0,x1,x3)
% DRAW_IX(X0,X1,X3) Draw vX_i(t,x) on the 3D-plane.
%
% See also CREATE_FUNCTIONAL_T, FIND_F_ZERO, DRAW_X_I.
len = length(x1(:,1));
limits = [-len len];
figure
hold on
for r = 1:len
    x0_mod = x0(r,:);
    x1_mod = x1(r,:);
    x3_mod = x3(r,:);
    borders0 = (x0_mod >= limits(1)) & (x0_mod < limits(2));
    borders1 = (x1_mod >= limits(1)) & (x1_mod < limits(2));
    borders3 = (x3_mod >= limits(1)) & (x3_mod < limits(2));
```

```
    borders = borders0 & borders1 & borders3;
    x0_mod = x0_mod(borders);
    x1_mod = x1_mod(borders);
    x3_mod = x3_mod(borders);
    plot3(x1_mod, x3_mod, x0_mod, 'LineWidth', 3.0)
end
xlabel('X1')
ylabel('X3')
zlabel('X0')
title('World lines of quaziparticles')
hold off
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% dKhi.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function dKhi_val = dKhi(const, gamma, x, k)
% DKHI k-derivative of wave function.
%    DKHI = DKHI(CONSTANTS, GAMMA, X, K)
%    Calculate derivative of wave limited function khi by k.
%
%    Constants are:
%    N+, N- -- quantities of number sets. Must be positive integers.
%
%    Happa+, Happa- -- eigenvalues. Must be positive.
%
%    b+, b- -- factor before reversed exponent.
%    Elements with odd index must be positive, even -- negative.
%
%    See also CREATE_FUNC_GAMMA, KHI.
dKhi_val = 0.0;
for n = 1:const.N
```

```
  dKhi_val   =   dKhi_val   -   gamma{n}(x)   ./   power(k   -   1i   .*
const.happa(n),2.0);
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% determinate_T.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [f_plus, f_minus] = determinate_T(func_t)
% DETERMINATE_T Get determinate of T(t, x).
%    det(T) = DETERMINATE_T(FUNC_T, T, X)
%
%                | t11(x + t) t12(x + t) |
% det(T(t, x)) = |                       |
%                | t21(x - t) t22(x - t) |
%
%    See also CREATE_FUNCTIONAL_T.

f_plus = @(t, x)(func_t{1, 1}(x + t) ./ func_t{1, 2}(x + t));
f_minus = @(t, x)(func_t{2, 1}(x - t) ./ func_t{2, 2}(x - t));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% determinate_T_alt.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [f_plus, f_minus] = determinate_T_alt(func_t)
% DETERMINATE_T_ALT Get determinate of T(t, x).
%    det(T) = DETERMINATE_T_ALT(FUNC_T, T, X)
%
%                | t11(x + t) t12(x + t) |
% det(T(t, x)) = |                       |
```

```
%                | t21(x - t) t22(x - t) |
%
%    See also CREATE_FUNCTIONAL_T.

f_plus = @(t, x)(func_t{1, 1}(x + t) .* func_t{2, 2}(x - t));
f_minus = @(t, x)(func_t{2, 1}(x - t) .* func_t{1, 2}(x + t));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% data_load.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [const_plus, const_minus] = data_load(filename)
% DATA_LOAD    Load constants to model.
%    [CONST_PLUS, CONST_MINUS] = DATA_LOAD(FILENAME)
%   Load constants from file to model.
%
%    Constants are:
%    N+, N- -- quantities of number sets. Must be positive integers.
%
%    Happa+, Happa- -- eigenvalues. Must be positive.
%
%    b+, b- -- factor before reversed exponent.
%    Elements with odd index must be positive, even -- negative.
%
%    See also DLMREAD, class CONST.

N_temp = dlmread(filename,';',[0 0 0 1]);    % Load quantities
% Check for positiveness of quantities
assert((N_temp(1, 1) >= 1) && (N_temp(1, 2) >= 1), ...
    'N must be positive.')

% Load other constants
```

```
raw_plus = dlmread(filename,';',[1 0 2 (N_temp(1, 1) - 1)]);
raw_minus = dlmread(filename,';',[3 0 4 (N_temp(1, 2) - 1)]);

% Construct objects from loaded data
const_plus = consts(N_temp(1, 1), raw_plus(1,:), raw_plus(2,:));
const_minus = consts(N_temp(1,2),raw_minus(1,:),raw_minus(2,:));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% create_functional_t.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function  t_funcs  =  create_functional_t(x_int,  khi_plus,  khi_minus,
...
    d_plus, d_minus)
% function t_funcs = create_functional_t(c_integral, conf, ...
%     khi_plus, khi_minus, pie, plus_zero, minus_zero)
% CREATE_FUNCTIONAL_T Creates t(x) functions.
%    T(x) = CREATE_FUNCTIONAL_T(INTEGRAL_CONSTANT, KHI+, KHI-)
%    Creates four t(x) functions, packed in 2x2 cell array.
%
%    See also CREATE_FUNC_KHI.

t_funcs = cell(2);
t_funcs{1,2} = @(x)t2_(khi_plus, x);
t_funcs{2,2} = @(x)t2_(khi_minus, x);

% Get to real numbers.
t_funcs{1,1} = @(x)-(d_plus(x, 0.0) .* 1i + x .* khi_plus(x, 0.0));
t_funcs{2,1} = @(x)(d_minus(x, 0.0) .* 1i + x .* khi_minus(x, 0.0));

coef_plus = sqrt( abs( sum( t_funcs{1,2}(x_int(1:end-1)) .* ...
    diff(t_funcs{1,1}(x_int)) - ...
    diff(t_funcs{1,2}(x_int)) .* t_funcs{1,1}(x_int(1:end-1)) )));
```

```
coef_minus = sqrt( abs( sum( t_funcs{2,2}(x_int(1:end-1)) .* ...
    diff(t_funcs{2,1}(x_int)) - ...
    diff(t_funcs{2,2}(x_int)) .* t_funcs{2,1}(x_int(1:end-1)) )));

t_funcs{1,2} = @(x)(t2_(khi_plus, x) ./ coef_plus);
t_funcs{2,2} = @(x)(t2_(khi_minus, x) ./ coef_minus);

t_funcs{1,1}                              =                              @(x)(-
(d_plus(x,0.0).*1i+x.*khi_plus(x,0.0))./coef_plus);
t_funcs{2,1}                                                                 =
@(x)((d_minus(x,0.0).*1i+x.*khi_minus(x,0.0))./coef_minus);

end

function t2 = t2_(khi, x)
    t2 = khi(x, 0.0);
end

function t1 = t1_(c, khi, zero, pie, tol, cf, x)
    t1 = c;
    ch = abs(x) > tol;
if(any(ch))
    if(isscalar(x))
        if(x >= 0.0)
            lin = [-cf zero(zero >= 0.0 & zero <= x) (x+cf)];
        else
            lin = [(x-cf) zero(zero >= x & zero < 0.0) cf];
        end
        for li = 1:length(lin)-1
            pies = linspace(lin(li) + cf, lin(li+1) - cf, pie);
            t1 = t1 + trapz(pies, power(khi(pies, 0.0), -2.0));
        end
    else
```

```
        prelin = x(ch);
        if(~any(prelin < 0.0))
          lin  =  [(prelin(1)-cf)  zero(zero  >=  0.0  &  zero  <=
prelin(end)) ...
                (prelin(end)+cf)];
        elseif(~any(prelin > 0.0))
          lin = [(prelin(1)-cf) zero(zero >= prelin(1) & zero < 0.0)
...
                (prelin(end)+cf)];
        else
          lin = [prelin(1) zero prelin(end)];
          lin = sort(lin);
        end
        for li = 1:length(lin)-1
            pies = prelin( (prelin >= (lin(li) + cf)) & ...
                (prelin <= (lin(li+1) - cf)) );
            t1 = t1 + trapz(pies, power(khi(pies, 0.0), -2.0), 2);
        end
    end

end
    t1 = t1 .* khi(x, 0.0);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% create_func_vX.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function vX = create_func_vX(funcs_t, const_vect, p)
%CREATE_FUNC_VX Creates vX(t,x).
%    VX(T,X) = CREATE_FUNC_VX(FUNCS_T, CONST_VECT, P)
%   Creates vX(t,x) function for use in the graphic.
%
```

```
%    See also CREATE_FUNCTIONAL_T.

function [ax0, ax1, ax3] = vX_aux(t, x)

size_t = length(t);
tol = 1e-12;

if(size_t > 1)
res = zeros(3,size_t);

for k = 1:size(res,2)
b0 = const_vect(1) + p * integral(@(nu)b_0(funcs_t, nu, 1), tol, ...
    x(k)+t(k)) - p * integral(@(nu)b_0(funcs_t, nu, 2), tol, x(k) -
t(k));
b1 = const_vect(2) + p * integral(@(nu)b_1(funcs_t, nu, 1), tol, ...
    x(k)+t(k)) - p * integral(@(nu)b_1(funcs_t, nu, 2), tol, x(k) -
t(k));
b3 = const_vect(3) + p * integral(@(nu)b_3(funcs_t, nu, 1), tol, ...
    x(k)+t(k)) - p * integral(@(nu)b_3(funcs_t, nu, 2), tol, x(k) -
t(k));

res(:,k) = [b0 b1 b3];
end

else % if(size_t > 1)
res = zeros(3,length(x));

for k = 1:size(res,2)
b0 = const_vect(1) + p * integral(@(nu)b_0(funcs_t, nu, 1), tol, ...
    x(k) + t) - p * integral(@(nu)b_0(funcs_t, nu, 2), tol, x(k) -
t);
b1 = const_vect(2) + p * integral(@(nu)b_1(funcs_t, nu, 1), tol, ...
```

```
    x(k) + t) - p * integral(@(nu)b_1(funcs_t, nu, 2), tol, x(k) -
t);
b3 = const_vect(3) + p * integral(@(nu)b_3(funcs_t, nu, 1), tol, ...
    x(k) + t) - p * integral(@(nu)b_3(funcs_t, nu, 2), tol, x(k) -
t);

res(:,k) = [b0 b1 b3];
end

end % if(size_t > 1)

ax0 = res(1,:);
ax1 = res(2,:);
ax3 = res(3,:);
end

vX = @vX_aux;

end


function b0 = b_0(funcs_t,nu,i)
    b0      =      0.5      *      (power(funcs_t{i,2}(nu),2.0)      +
power(funcs_t{i,1}(nu),2.0));
end

function b1 = b_1(funcs_t,nu,i)
    b1 = -(funcs_t{i,1}(nu) .* funcs_t{i,2}(nu));
end

function b3 = b_3(funcs_t,nu,i)
    b3      =      0.5      *      (power(funcs_t{i,2}(nu),2.0)      -
power(funcs_t{i,1}(nu),2.0));
```

```
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% create_func_gamma.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function gamma = create_func_gamma(const, da_func, x_interval)
% CREATE_FUNC_GAMMA Generates bundle of functions.
%   [GAMMA] = CREATE_FUNC_GAMMA(CONSTANTS,DA) Generates bundle of
%   functions through the solution of linear algebraic system.
%
%    Constants are:
%    N+, N- -- quantities of number sets. Must be positive integers.
%
%    Happa+, Happa- -- eigenvalues. Must be positive.
%
%    b+, b- -- factor before reversed exponent.
%    Elements with odd index must be positive, even -- negative.
%
%    See also CREATE_FUNC_DA, SPLINE, class CONST.

% Solve Cx = B algebraic equation
function B_val = B(n, x_m)
 B_val = exp(-2.0 * const.happa(n) * x_m);

 for m = n
  B_val(m,:) = ...
     B_val(m,:) .* const.b(m) ./ da_func(1i * const.happa(m));
 end

end

% Get values from solution function...
```

```
B_temp = B(1:const.N, x_interval);
C = cell(1,length(x_interval));
for k = 1:length(C)
    C{k} = eye(const.N, const.N);
    for i_ = 1:const.N
        %b = 1i .* B(i, x_interval(k));
        for j_ = 1:const.N
            C{k}(i_,j_) = C{k}(i_,j_) - 1i .* ...
                B_temp(i_,k) ./ (const.happa(i_) + const.happa(j_));
        end
    end
end

gamma_aux = zeros(size(B_temp));
for k = 1:size(C,2)
    gamma_aux(:,k) = C{k} \ B_temp(:,k);
end

% ...and present them as "continuous function"
gamma = cell(1, const.N);
spl = cell(1, const.N);
for k = 1:const.N
   spl{k} = griddedInterpolant(x_interval, gamma_aux(k,:), 'spline');
   gamma{k} = @(x)spl{k}(x);
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% create_func_da.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function da = create_func_da(const)
% CREATE_FUNC_DA Creates da(k).
```

```
%    DA(k) = CREATE_FUNC_DA(CONSTANTS)
%   Creates derivative of function a(k) for further use.
%    Constants are:
%    N+, N- -- quantities of number sets. Must be positive integers.
%    Happa+, Happa- -- eigenvalues. Must be positive.
%    b+, b- -- factor before reversed exponent.
%    Elements with odd index must be positive, even -- negative.
%    See also CREATE_FUNC_A, class CONST.
da = @(k)da_aux(const, k);
end

function da_res = da_aux(C, k)
    da_res = 0.0;
    for n = 1:C.N
        da_val = 1.0;
        for i_ = 1:(n-1)
            da_val = da_val .* (k - 1i .* C.happa(i_)) ./ ...
                (k + 1i .* C.happa(i_));
        end
        da_val = da_val .* 2i .* C.happa(n) ./ ...
            (power(k, 2.0) + 2i .* k .* C.happa(n) - ...
            power(C.happa(n), 2.0));
        for k_ = (n+1):C.N
            da_val = da_val .* (k - 1i .* C.happa(k_)) ./ ...
                (k + 1i .* C.happa(k_));
        end
        da_res = da_res + da_val;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% consts.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
classdef consts
    % CONST Class for representing constants as one object
    %      Objects of this class represent constants for later
calculations.

    properties (SetAccess = immutable)
        N uint8     % Quantity of numbers in set. Must be positive
integer.
        happa       % Array of eigenvalues. Must be positive.

        % b - Array of factors before reversed exponent, where
        % elements with odd index must be positive, even -- negative.
        b
    end

    methods
        function obj = consts(n, Happa, B, varargin)
            % Constructor for CONST object.
            % [CONST] = CONST(N, HAPPA, B) or [CONST] = CONST()
            %  Before assignment tries run default constructor
statements,
            % then checks for argument correctness.
            % All arguments must be numeric values.

            if nargin == 0
                obj.N = 1;
                obj.happa = 0.5;
                obj.b = 0.25;
                return
            end
            assert(isvector([Happa, B]), ...
                'Input values (exclude quantity) must be vectors.')
```

```
            assert(isnumeric([n, Happa, B]), ...
                'Input values must be numeric.')
            assert(n >= 1, 'N must be positive.')
            assert(all(Happa > 0.0), 'All Happa must be positive.')
            assert(all(B(1:2:n) > 0.0) && ...
                ((n < 2) || all(B(2:2:n) < 0.0)), ...
                'Incorrect b initialisation.')
            obj.N = uint8(n);
            obj.happa = Happa;
            obj.b = B;
            if(~iscolumn(obj.happa))
                obj.happa = obj.happa.';
            end
            if(~iscolumn(obj.b))
                obj.b = obj.b.';
            end
            for v_s = 1:length(varargin)
                switch varargin{v_s}
                    case 'InverseB'
                        obj.b = -obj.b;
                end
            end
        end
    end
end
```

Рис. 3.1 – Листинг программы

Перейдем к результату, взятого из программного обеспечения (рис. 3.1).

3.2 Результаты

Сейчас будет рассматриваться проведенный эксперимент, чтобы сделать выводы о целесообразности дальнейших вычислений.

Для этого, сначала определим входные данные для задачи Штурма-Лиувилля. Пусть заданы два натуральных числа N_+, N_-, и пусть определены собственные значения $\kappa_{i\pm}$, соответствующие дискретному спектру, а также числитель коэффициента отражения $b_{i\pm}$, где $i = \overline{1, N_\pm}$. Моделирование будет проводиться на временном промежутке $t \in [-10;10]$ и на заданной области определения $x \in [-5;5]$.

К моделированию допущены следующие входные данные (табл. 3.1-3.4):

Табл. 3.1 – Входные данные при N_+=2, N_-=1

	N	κ_1	κ_2	b_1	b_2
+	2	1,5	1,0	1,6	-0,8
–	1	1,0	-	0,3	-

Табл. 3.2 – Входные данные при N_+=2, N_-=3

	N	κ_1	κ_2	κ_3	b_1	b_2	b_3
+	2	2,1	1,4	-	0,46	-0,14	-
–	3	1,7	1,15	0,9	0,4	-0,6	1,2

Табл. 3.3 – Входные данные при N_+=3, N_-=3

	N	κ_1	κ_2	κ_3	b_1	b_2	b_3
+	3	1,8	1,5	0,6	0,5	-0,3	0,7
–	3	1,5	1,2	0,9	0,3	-1,6	2,0

Табл. 3.4 – Входные данные при N_+=4, N_-=4

	N	κ_1	κ_2	κ_3	κ_4	b_1	b_2	b_3	b_4
+	4	2,0	1,7	1,4	1,1	0,5	-0,3	0,7	-0,6
–	4	1,5	1,2	0,9	0,6	0,3	-1,6	2	-0,1

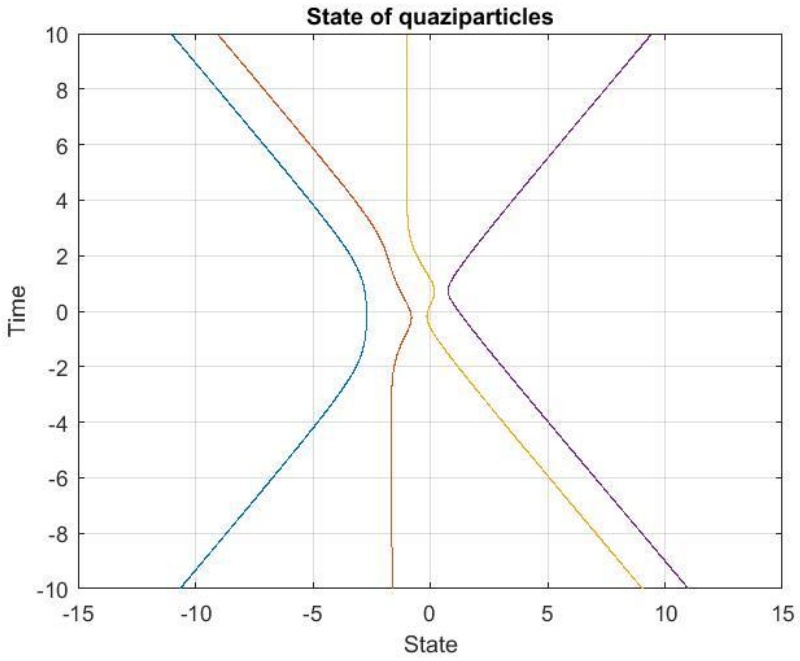

Рис. 3.2 – График нулей функции F при N_+=2, N_-=1

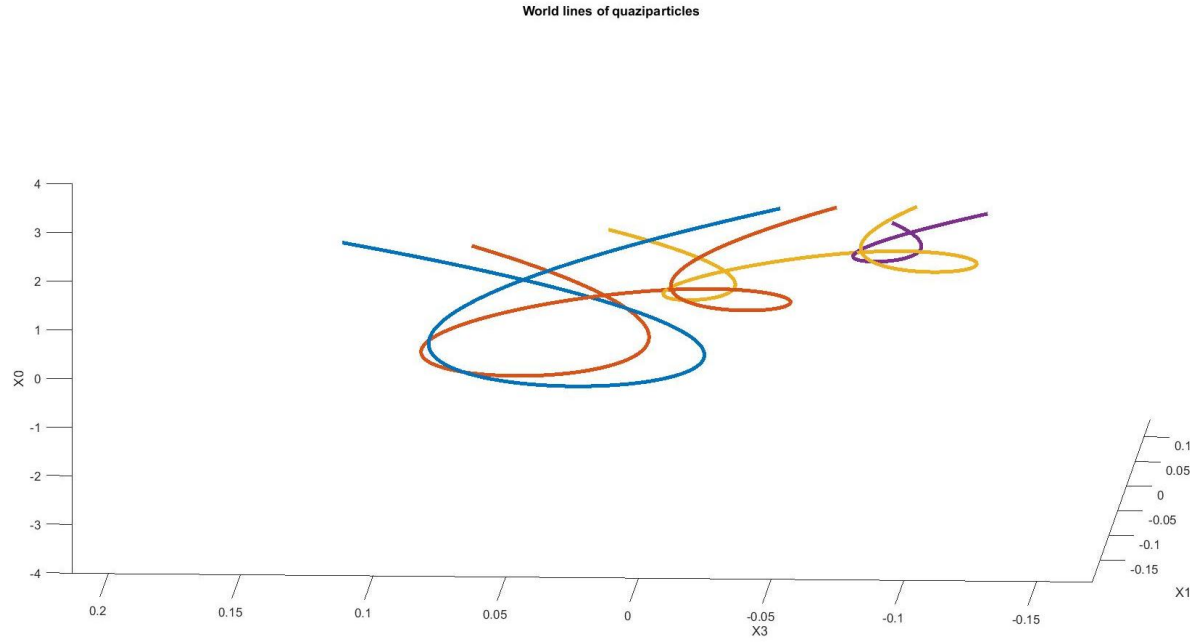

Рис. 3.3 – График мировых линий энионов при N_+=2, N_-=1

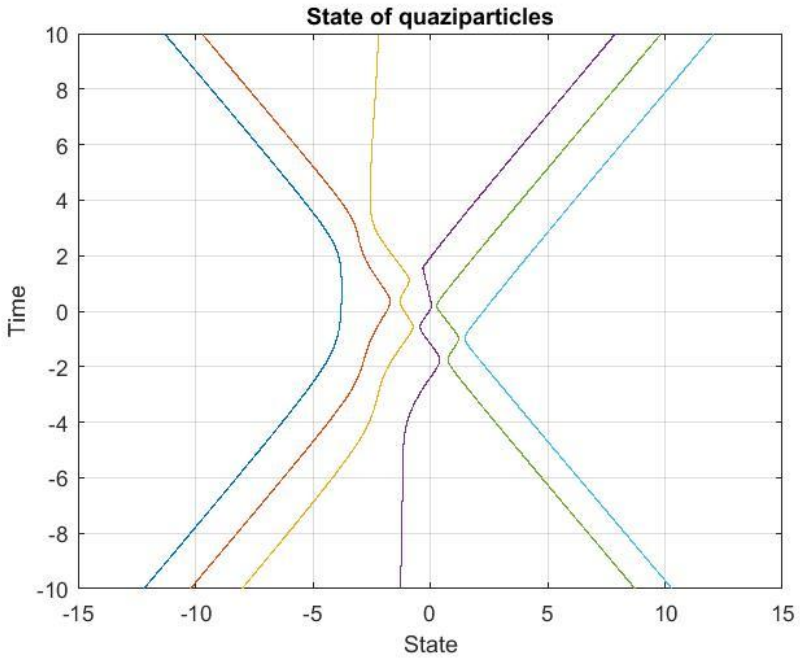

Рис. 3.4 – График нулей функции F при N_+=2, N_-=3

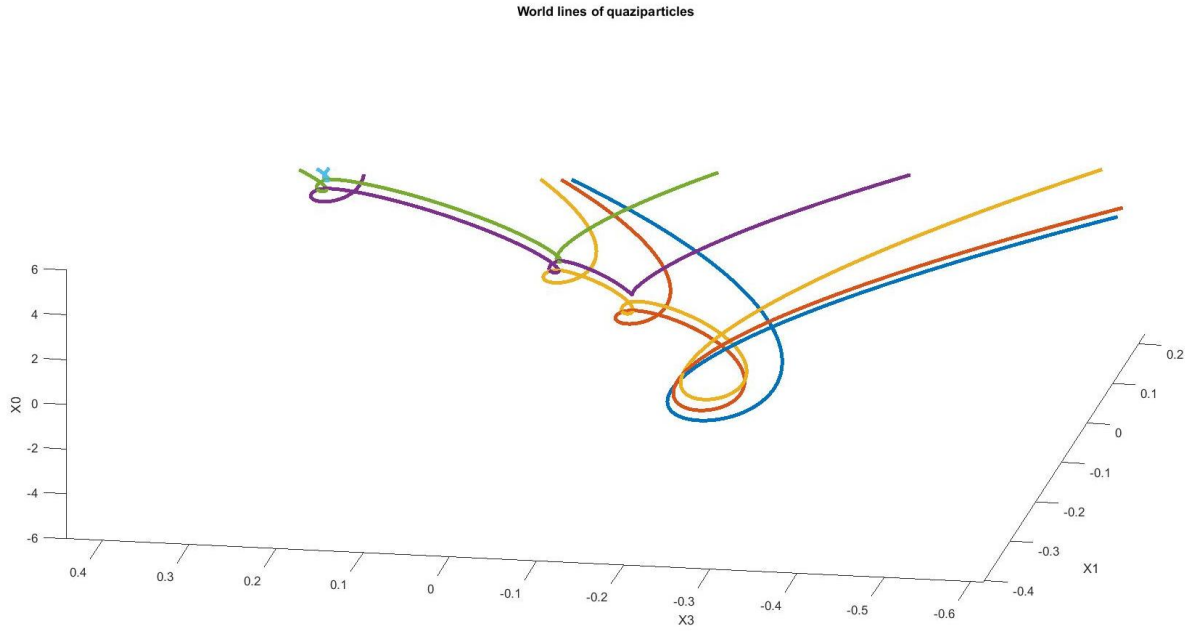

Рис. 3.5 – График мировых линий энионов при N_+=2, N_-=3

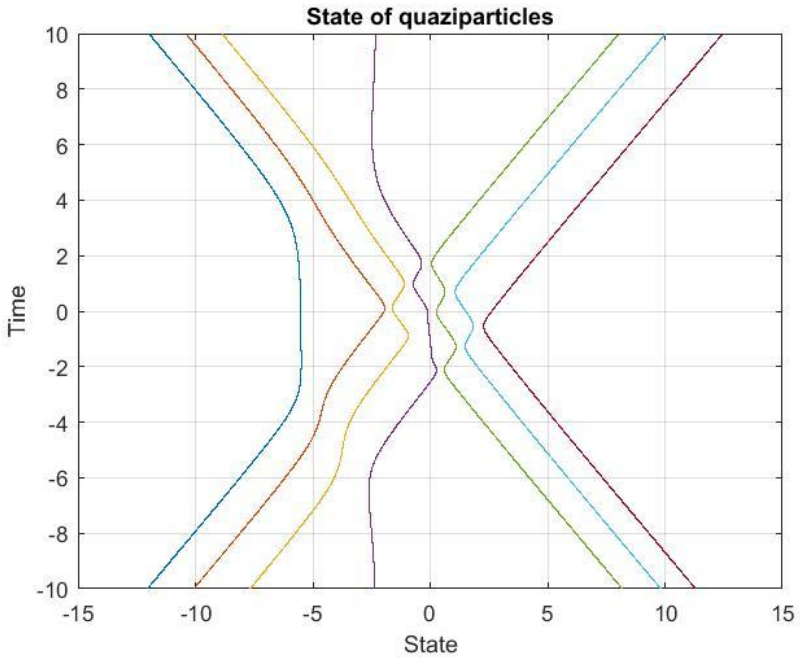

Рис. 3.6 – График нулей функции F при N_+=3, N_-=3

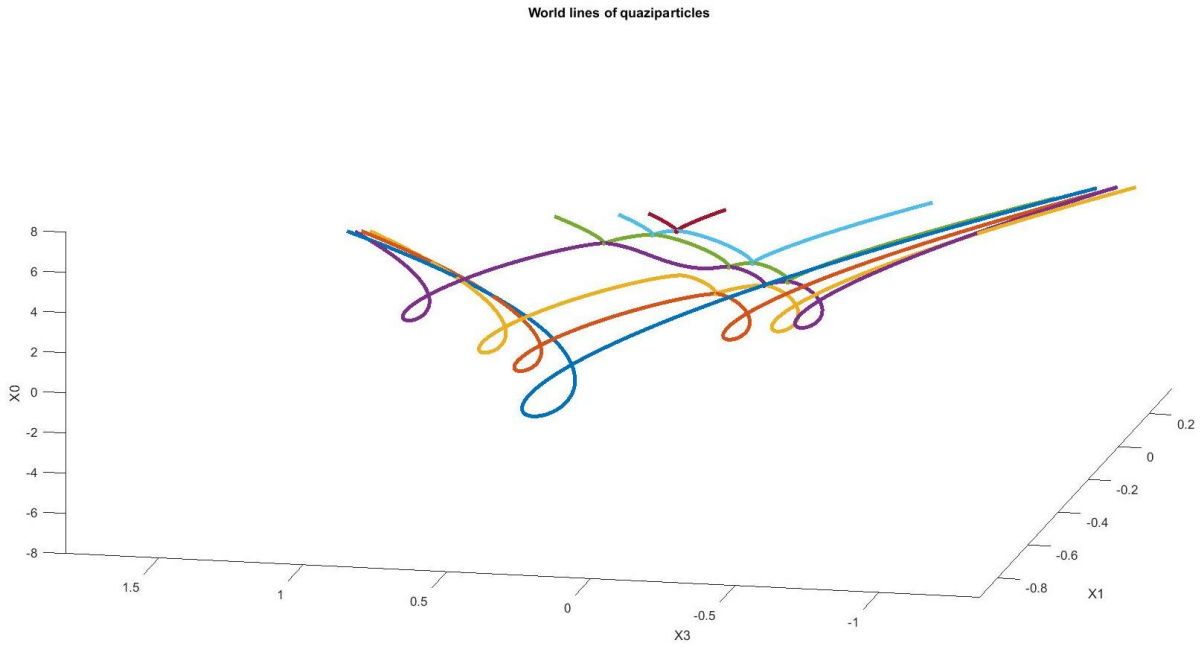

Рис. 3.7 – График мировых линий энионов при N_+=3, N_-=3

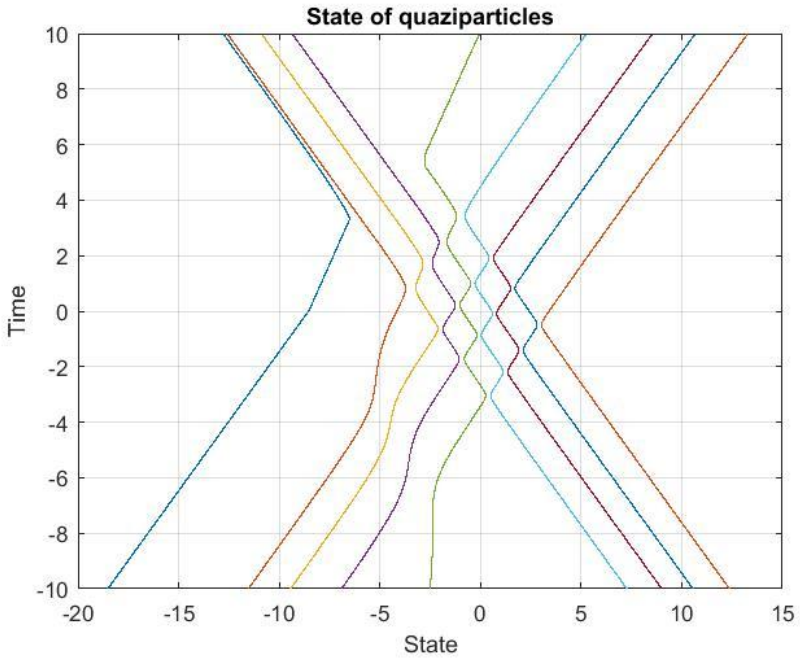

Рис. 3.8 – График нулей функции F при N_+=4, N_-=4

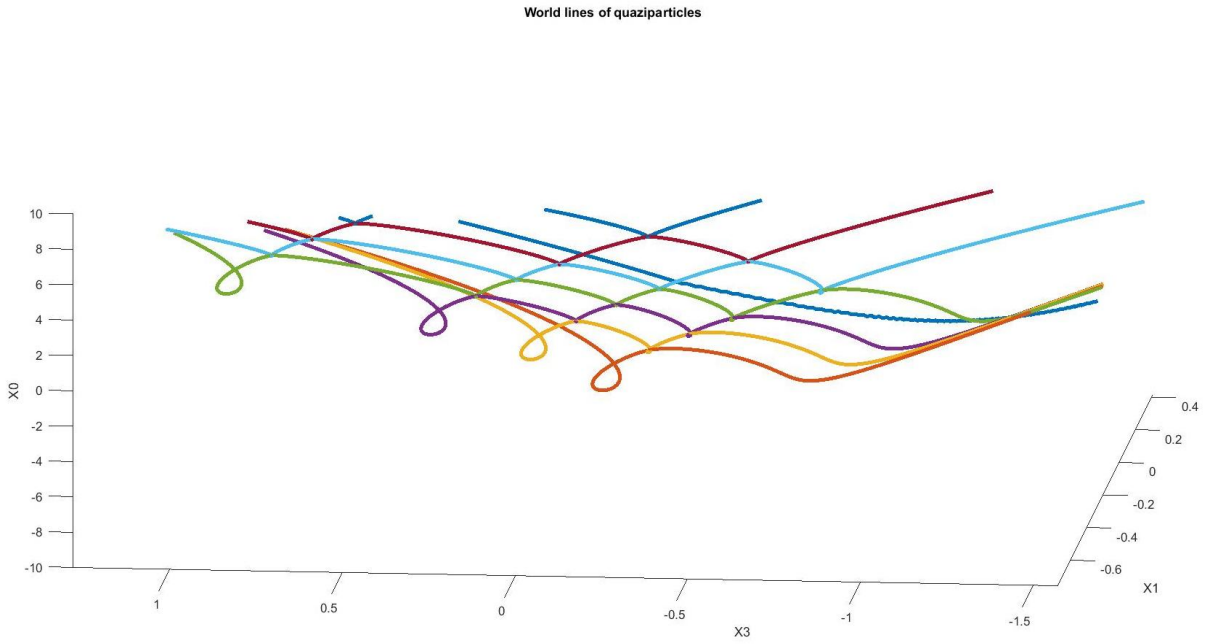

Рис. 3.9 – График мировых линий энионов при N_+=4, N_-=4

На рисунках 3.2, 3.4, 3.6, 3.8 можно увидеть линии, составленные из точек возврата мирового листа, соответствующего начальным данным рассеяния на таблицах 3.1-3.4. Заметно, в зависимости от коэффициентов преломления, как сближение частиц друг к другу, так и их отдаление. Максимальная концентрация точек возврата наблюдается в области около нуля ($t \in [-2;2] \cap x \in [-5;5]$). Вне этой области, линии станут расходиться в бесконечность, не пересекаясь и даже не приближаясь друг к другу. На рисунке 3.6 (центральная фиолетовая линия), и на рисунке 3.8 (крайняя левая линия) можно сделать вывод о возникшей ошибке в проведенных программой вычислениях.

На рисунках 3.3, 3.5, 3.7, 3.9 представлены мировые линии энионов в трехмерном пространстве Минковского $E_{1,2}$. Ось OX0 будет временной осью (отрицательные значения – прошлое, положительное – будущее). Можно заметить, как мировые линии энионов переплетаются между собой, образуя искомые косы.

Подведем итоги.

Заключение

Использованы несколько наборов данных рассеяния уравнения Штурма-Лиувилля на примере быстроубывающего случая.

По результатам проведенного анализа, были выведены математические формулы для программы моделирования квазичастиц точками возврата планарных струн. Собрана статистика по результатам моделирования на различных наборах входных данных, имитирующих поведение мировых линий энионов. Была исследована динамика движения точек возврата. Собрана статистика по результатам моделирования на различных наборах входных данных, имитирующих поведение мировых линий энионов.

В ходе проведенных компьютерных экспериментов, установлено переплетение мировых линий друг с другом, что свидетельствует об образовании кос различной топологии. Это означает, что нужно продолжать исследование моделирования энионов в целях приложений построенных моделей к исследованию топологического квантового компьютера.

Программное обеспечение надо усовершенствовать: расширить возможный количество собственных значений до сорока, увеличить охватываемый диапазон моделирования, исправить некоторые недочеты в запрограммированном вычислении.

Список использованной литературы

1. Sarma S. D. Topologically Protected Qubits from a Possible Non-Abelian Fractional Quantum Hall State // Physical Review Letters. – 29 апреля 2005. – Часть 94. – С. 166802-1–168802-4.

2. Day C. Devices Based on the Fractional Quantum Hall Effect May Fulfill the Promise of Quantum Computing // Physics Today. – Октябрь 2005. – Часть 58. – С. 21–24.

3. Lindley D. Anyon There?, 2 ноября 2005. // Physical Review Focus, часть 16, статья № 14: http://focus.aps.org/story/v16/st14.

4. Preskill J. Topological Quantum Computation // Lecture Notes for Physics 219: Quantum Computation: www.theory.caltech.edu/˜preskill/ph219/topological.pdf.

5. Anderson M. R. The Mathematical Theory of Cosmic Strings. Cosmic Strings in the Wire Approximation // High Energy Phys. Cosmol. Gravit., IOP Publ., Bristol. – 2003.

6. Барбашов Б. М. Модель релятивисткой струны в физике адронов / Б. М. Барбашов, В. В. Нестеренко. – Москва : Энергоатомиздат, 1987.

7. Дубровин Б. А. Современная геометрия: методы и приложения / Б. А. Дубровин, С. П. Новиков, А. Т. Фоменко. – Москва : Наука, 1986.

8. Валиев К. А. Квантовые компьютеры: надежды и реальность. – Ижевск: НИЦ «Регулярная и хаотическая динамика», 2001.

9. Ladd. T. D. Quantum Computing // Nature. – 2010. – Ч. 464. – С. 45-53.

10. Андреев И. В. Хромодинамика и жесткие процессы при высоких энергиях. Современные проблемы физики. – М.: Наука, 1981.

11. Индурайн Ф. Квантовая хромодинамика. – М.: Мир, 1986.

12. Барбашов Б. М. Динамика релятивистской струны // Физика элементарных частиц и атомного ядра. – 1978. – Т. 9, вып. 5. – С. 709-758.

13. Барбашов Б. М. Некоторые вопросы классической динамики релятивистской струны. – Дубна, 1976.

14. Барбашов Б. М. Классическая динамика релятивистской струны. – Дубна, 1974.

15. Барбашов Б. М. Геометрический подход к динамике релятивистской струны // Теорет, мат. физ. – 1979. – Т. 39, № 1. – С. 27-34.

16. Dickey L. Soliton equations and Hamiltonian systems // Advanced Ser. Math. Phys. – Singapore : World Sci., 2003. – № 26, 2-е изд.

17. Виленкин Н. Я. Специальные функции и теория представлений групп. – М. : Наука, 1965.

18. Фущич В. И. Симметрия уравнений квантовой механики. – М. : Наука, 1990.

19. Сэффмен Ф. Дж. Динамика вихрей. – М. : Научный мир, 2000.

20. Алексеенко С. В. Введение в теорию концентрированных вихрей. – Новосибирск : Ин-т теплофизики СО РАН, 2003.

21. Изергин А. Г. Квантовый метод обратной задачи. – Дубна : ЭЧАЯ, 1982. – т. 13, № 3. – с. 501-54.1.

22. Корепин В. Е. Квантование солитонов // Физика элементарных частиц. – Л., 1977. – с. 130-146.

23. Ablowitz M. J. Solitons and the Inverse Scattering Transform. – Philadelphia : SIAM, 1981.

24. Лэкс П. Интегралы нелинейных эволюционных уравнений и уединенные волны // сб. Математика. – 1969. – т. 13, № 15. – с. 128-150.

25. Захаров В. Е. О взаимодействии солитонов в устойчивой среде. – ЖЭТФ, 1973, т. 64, № 5, с. 1627-1639.

26. Марченко В. А. Восстановление потенциальной энергии по фазам рассеянных волн. – ДАН СССР, 1955, т. 104, № 5, с. 695-698.

27. Захаров В. Е. Теория солитонов: метод обратной задачи – Москва : Наука, 1979.

28. Талалов С. В. Об N-солитонных струнах в четырехмерном пространстве-времени // Теоретическая и математическая физика. – 2007. – Т. 152, № 3. – С. 430-439.

29. Талалов С. В. Решения типа струн, вихрей и энионов для иерархии нелинейного уравнения Шредингера // Теоретическая и математическая физика. – 2013. – Т. 176, № 3. – С. 372-384.

30. Марченко В. А. Операторы Штурма-Лиувилля и их приложения. – Киев : Издательство «Наукова думка», 1977.

31. Кабанихин С. И. Обратные и некорректные задачи. – Изд.2-е – Новосибирск : Сибирское научное издательство, 2009.

32. Тахтаджян Л. А. Гамильтонов подход в теории солитонов – Москва : ФГУП «Академиздатцентр «Наука»», 1986.

33. Ощепков, А. Ю. Теория солитонов. Математическое описание и физические приложения. – Пермь : Федеральное агентство по образованию ГОУ ВПО “Пермский государственный университет”, 2007.

34. Джорджадзе Г. П. Сингулярные решения уравнения $\Box\varphi+\frac{m^2}{2}e^{\varphi}=0$ и динамика особенностей // Теоретическая и математическая физика. – 1979. – Т. 40, № 2. – С. 221-234.

35. Захаров В. Е. Теория упругих сред с микроструктурой. – Москва : Наука, 1975.

36. Стокер Дж. Волны на воде. – Москва : ИЛ, 1959.

37. Lax P. D. Commun. Pure Appl. Math. – 1968. – Ч. 21. – С. 467.

38. Новиков С. П. Функц. анализ. – 1974. – Т. 8, вып. 3. – С. 54.

39. Calogero F. Nonlinear evolution equations solvable by the spectral transform // Collection of papers. – Лондон : Питман, 1978.

40. Захаров В. Е. Функц. анализ. – 1971. – Т. 5, вып. 4. – С. 18.

41. Марченко В. А. Матем. сборник. – 1974. – Т. 95, № 3. – С. 331.

42. Новиков С. П. Мультизначные функции и функционалы. Аналог теории Морзе // ДАН СССР. – 1981. – Т. 260, № 1. – С. 31.

43. Талалов С. В. Замечание о геометрическом описании релятивистской струны // ТМФ. – 2000. – Т. 123, № 1. – С. 38.

44. Талалов С. В. Об одной модели эниона // ТМФ. – 2010. – Т. 165, № 2. – С. 329-340.

45. Клименко С. В. Исследование особенностей на мировых листах открытых релятивистских струн // ТМФ. – 1998. – Т. 114, № 3. – С. 380-398.

46. Рубаков В. А. Большие и бесконечные дополнительные размерности // УФН. – 2001. – Т. 171, №9. – С. 913.

47. Моффатт К. Вихревая динамика: наследие Гельмгольца и Кельвина // Нелинейная динам. – 2006. – Т. 2, № 4. – С. 401-410.

48. Протогенов А. П. Анионная сверхпроводимость в сильно коррелированных спиновых системах // УФН. – 1992. – Т. 162, № 7. – С. 1-80.

49. Клейн Ф. Неевклидова геометрия. – Москва : НКТП СССР, 1936.

50. Дьяконов В. П. MATLAB R2016. – Москва : ДМК, 2016.