Федеральное государственное бюджетное учреждение науки Институт вычислительной математики им. Г.И. Марчука Российской академии наук (ИВМ РАН)

На правах рукописи

Желтков Дмитрий Александрович

Методы аппроксимации и оптимизации на основе тензорных поездов и их приложения

Специальность 1.2.2 —
«Математическое моделирование, численные методы и комплексы программ»

Диссертация на соискание учёной степени
кандидата физико-математических наук

Научный руководитель:
академик РАН, доктор физико-математических наук, профессор
Тыртышников Евгений Евгеньевич

Москва — 2022

Оглавление

Стр.

Введение 4

Глава 1. Параллельная реализация TT-крестового алгоритма 13
1.1 Немного о тензорных разложениях и их свойствах 13
1.2 Матричный крестовый метод 17
1.3 Параллельный матричный крестовый метод 20
1.4 Метод TT-крестовой интерполяции 24
1.5 Параллельная версия TT-крестового метода 28
1.6 Численные эксперименты 28
1.6.1 Тестирование надёжности TT-крестового метода 28
1.6.2 Тестирование параллельной производительности TT-крестового метода 33
1.6.3 Интегрирование с помощью TT 35

Глава 2. Оптимизационные свойства крестового метода 37
2.1 Оценка величины максимального элемента 38
2.2 Оценка вероятности удачного старта для выборки столбцов 43
2.3 Оценка вероятности удачного старта для случайного столбца 47
2.4 Теоремы об оптимизационных свойствах крестового метода 48

Глава 3. Метод глобальной оптимизации на основе тензорных поездов 53
3.1 Математическая постановка задачи. 54
3.2 Последовательная версия алгоритма 55
3.3 Искусственное увеличение размерности 56
3.4 Параллельная версия алгоритма 58
3.5 Замечание об аппроксимационных свойствах алгоритма 59

Глава 4. Применение алгоритма глобальной оптимизации к задаче докинга 61
4.1 Постановка задачи 61
4.2 Критерии эффективности алгоритмов 62
4.3 Численные эксперименты 63

Стр.

Глава 5. Применение алгоритма глобальной оптимизации к задаче оценки параметров моделей ВИЧ-инфекции 68
5.1 Постановка задачи 68
5.2 Модели иммунного ответа на ВИЧ-инфекцию 69
5.3 Описание измерений 70
5.4 Численные эксперименты 70

Глава 6. Применение алгоритма глобальной оптимизации для построения антенных решеток в автомобильных радарах . . . 73
6.1 Постановка задачи 73
6.2 Численные эксперименты 76

Заключение 81

Список литературы 82

Приложение А. Описание моделей ВИЧ-инфекции 88
А.1 Модель 1 88
А.2 Модель 2 92
А.3 Модель 3 93
А.4 Модель 4 94

Введение

Актуальность темы. Современное развитие методов вычислительной математики во многом определяется необходимостью решать все более сложные задачи оптимизации, сложность которых характеризуется сразу несколькими факторами. Во-первых, эти задачи имеют большую размерность. Число переменных в них может достигать сотен и даже тысяч. Во-вторых, оптимизируемые функции оказываются невыпуклыми и обладают большим числом локальных экстремумов. Наконец не редко хранение данных, определяющих значение функционала, предполагает использование различных вычислительных узлов, а сами вычисления становятся распределенными.

Все эти требования, представляющие значительные и зачастую непреодолимые трудности для классических методов, привели к созданию новых подходов в задачах невыпуклой оптимизации. Предложенный нами и изучаемый в настоящей диссертационной работе *метод глобальной оптимизации на основе тензорного TT-представления* является именно таким новым подходом. Новый метод, как показывает практика, весьма эффективен для обширного класса задач, обладающих дополнительной структурой. А именно, конструкция метода предполагает, что рассматриваемые нами функции многих переменных допускают специальное представление в виде тензоров малого тензорного ранга. Поясним сказанное.

Рассмотрим задачу поиска максимума вещественной функции d переменных $A(x_1, x_2, \cdots, x_d)$, заданной в ограниченном кубе Π пространства $\mathbb{R}^d$

$$(x_1^*, x_2^*, \cdots, x_d^*) = \arg \max_{(x_1, x_2, \cdots, x_d) \in \Pi} \left(A(x_1, x_2, \cdots, x_d)\right). \tag{1}$$

По каждому из направлений x_i зададим равномерную сетку и рассмотрим значения функции $A(x_1, x_2, \cdots, x_d)$ в узлах сетки. Отождествим функцию с многоиндексным массивом $A_{i_1, i_2, \cdots, i_d}$ или, как принято говорить, с *тензором размерности* d. Тем самым задача (1) сводится к поиску максимального элемента среди элементов массива

$$(i_1^*, i_2^*, \cdots, i_d^*) = \arg \max_{(i_1, i_2, \cdots, i_d)} \left(A_{i_1, i_2, \cdots, i_d}\right). \tag{2}$$

Очевидным образом возможность построения эффективного метода для (2) напрямую зависит от наличия в функции дополнительных структур скрытых или

явных. Так конструкция классических методов оптимизации предполагает, что оптимизируемая функция допускает хорошее приближение с помощью моделей низких порядков, например, линейной для методов типа градиентного спуска или квадратичной для методов типа Ньютона. Однако классические методы обладают лишь локальной сходимостью, поскольку модели низких порядков качественно описывают сложные функции лишь в относительно небольших окрестностях или, как принято говорить, локально.

С другой стороны, для методов, предназначенных для поиска глобального оптимума, например, таких как метод ветвей и границ, требуется лишь наличие минимальной информации о функции. Как правило, это оценка на её константу Липшица. Тем не менее, для сложных функций получение такой оценки затруднительно. Более того, в случае функций, зависящих от большого числа переменных, алгоритмическая сложность метода ветвей и границ растёт экспоненциально с размерностью задачи (явление известное как проклятие размерности) и оказывается неприемлемо высокой, даже при наличии точной оценки на константу Липшица.

Как мы сказали выше, предложенный нами и изучаемый в настоящей диссертационной работе метод оптимизации на основе тензорного TT-представления предназначен для поиска глобального оптимума сложных функций и предполагает наличие в оптимизируемой функции малопараметрической структуры специального вида. А именно, мы будем считать, что тензор $A_{i_1,i_2,\cdots,i_d}$ приближается тензорами малого тензорного ранга.

Опираясь на теоретический анализ, полученный в недавних работах Таунсенда [1], можно с уверенностью предположить, что такой малоранговый тензорный тип структуры широко распространён в задачах, связанных с современными приложениями. Так например, из [1] следует, что абсолютное большинство используемых моделей, в том числе относящихся к исследованию больших данных, допускают *поэлементное* (чебышевское) приближение матрицами и тензорами малых рангов.

Начало широкому использованию в вычислительной математике малоранговых приближений было положено в работах С.А. Горейнова, Н.Л. Замарашкина и Е.Е. Тыртышникова [2; 3], где на основе принципа максимального объёма был получен положительный ответ на ключевой вопрос о возможности эффективного приближения матриц матрицами ранга не выше r с использованием лишь небольшого числа элементов приближаемой матрицы. Вслед за этим в [4] был

предложен *крестовый алгоритм интерполяционного типа*, дающий на практике удовлетворительное решение задачи приближения матрицами малого ранга. Идеи матричного крестового алгоритма составляют основу современных крестовых алгоритмов для тензоров [5; 6].

Остановимся на этих важных для представленной работы аспектах подробнее. Итак, пусть матрица $A \in \mathbb{R}^{m\times n}$ допускает приближение A_r матрицей ранга не выше r вида

$$\|A - A_r\|_F \leqslant \varepsilon. \tag{3}$$

Если точность приближения ε достаточна, то заменяя A на A_r, получаем малопараметрическое представление для A. Действительно, так как любая матрица ранга r является произведением двух матриц $A_r = UV^T$ размеров $m \times r$ и $n \times r$, то для хранения A_r достаточно не более $r\,(m+n)$ параметров. Кроме того, представление вида $A_r = UV^T$ позволяет эффективно выполнять различные алгебраические операции с A.

По теореме Экарта-Янга наилучшее приближение во фробениусовой норме матрицы A матрицей ранга не выше r можно найти, построив сингулярное разложение A и оставив в приближении только r старших триплетов. При этом точность наилучшего приближения ранга r определяется сингулярными числами начиная с $(r+1)$-го сингулярного числа $\sigma_{r+1}(A)$ матрицы A. Однако алгоритмы построения сингулярного разложения имеют сложность $\mathcal{O}\left(\min(m,n)mn\right)$ и используют все элементы A. Поэтому для матриц больших размеров и/или для матриц, элементы которых получаются с помощью алгоритмически сложной вычислительной процедуры, такой стандартный подход неприменим.

Напротив, крестовый метод на основе принципа максимального объёма [2—4] позволяет находить приближение $\tilde{A}$ ранга r, используя $\mathcal{O}\left((m+n)r\right)$ элементов матрицы и затрачивая $\mathcal{O}\left((m+n)r^2\right)$ арифметических операций. При этом точность получаемого приближения удовлетворяет соотношению

$$\|A - \tilde{A}\|_C \leqslant (r+1)\,\varepsilon,$$

где $\|\cdot\|_C$ – чебышевская (поэлементная) матричная норма. Отметим, что именно такая поэлементная оценка представляет наибольший интерес для методов оптимизации.

В случае данных, задаваемых многоиндексными массивами (тензорами), значение задачи эффективного построения малорангового приближения многократно возрастает. В отличие от матриц понятие ранга тензора определяется

неоднозначно и зависит от вида представления. Наиболее популярными являются представления (приближения) в каноническом формате, в формате Таккера и в формате тензорного поезда. Среди этих возможных способов малопараметрического представления тензоров нас будет интересовать только приближение в формате тензорного поезда (Tensor Train, TT) или, как мы будем говорить для краткости, TT-представление [7].

TT-представление обладает целым рядом критически важных свойств, которые делают его незаменимым инструментом при работе с многомерными массивами:

1. для хранения тензора в формате TT-представления требуется всего $\sum_{i=1}^{d} n_i r_{i-1} r_i$ элементов памяти (r_i – TT-ранги тензора, точное определение которых будет дано в главе 1);
2. основные арифметические операции над тензорами, среди которых сложение, поэлементное умножение, округление, вычисление нормы Фробениуса и др., в TT-представлении выполняются за $O(dnr^3)$ арифметических действий, где $r = \max\limits_{i \in [1,d-1]}(r_i), n = \max\limits_{i \in [1,d]}(n_i)$;
3. предложен крестовый метод [6], получающий приближение тензора в TT-формате с использованием $O(dnr^2)$ его элементов и алгоритмической сложностью $\mathcal{O}\left(dnr^3\right)$.

В определённом смысле TT-представление – единственный известный алгебраический способ преодолеть проклятие размерности при работе с большими многомерными данными.

Первоначально идея метода глобальной оптимизации на основе TT-представления состояла в следующем:

1. построить *высокоточное* малопараметрическое TT-представление тензора $A_{i_1,i_2,\cdots,i_d}$, элементы которого представляют значение функции в узлах сетки;
2. используя малопараметрическое TT-представление разработать алгоритм поиска наибольшего (наименьшего) элемента в $A_{i_1,i_2,\cdots,i_d}$

Однако в численных экспериментах было замечено следующее. Оказалось, что в процессе вычислений крестовые методы просматривают элементы, близкие к наибольшему по модулю элементу во всём тензоре. Более того, это утверждение остается справедливым и в том случае, когда крестовый метод работает с тензорными рангами, значения которых существенно ниже требуемых для

аппроксимации тензора с высокой точностью. Другими словами, позиции максимальных по модулю элементов тензора определяются в крестовом методе существенно раньше, чем достигается приближение высокой точности.

Именно этот, на первый взгляд, несколько неожиданный факт позволяет построить эффективный метод глобальной оптимизации на основе ТТ-представления. В главе 2 диссертационной работы приводится теоретическое обоснование указанного наблюдения для частного случая приближения матриц и тензоров крестовым методом с предписанным рангом 1. В качестве дополнительного следствия из построенного теоретического анализа была получена оценка на сложность крестового метода (на число внутренних итераций), которая хорошо совпадает с наблюдаемой на практике.

В диссертационной работе метод глобальной оптимизации на основе ТТ-представления применялся сразу к нескольким трудным практическим задачам: расчёту положения молекулы-ингибитора при связывании с белком (метод докинга) [8—12]; идентификации параметров моделей ВИЧ инфекций [13—15]; построению и оптимизации антенных устройств для автомобильных радаров.

Число переменных (размерность пространства переменных) в каждой из указанных задач превышала 10 и для ряда задач превышала 100. Оптимизируемые функции этих приложений являются невыпуклыми, имеют большое число локальных экстремумов и сложные алгоритмы вычисления значения функции. Применение детерминистических алгоритмов глобальной оптимизации (метода ветвей и границ и др.) в такой ситуации невозможно.

В современной вычислительной практике в этом случае используются так называемые *недетерминистические методы*, среди которых упомянем следующие: стохастические методы (например, метод Монте-Карло); эвристические и метаэвристические методы (например, метод имитации отжига или генетические алгоритмы). Недетерминистические методы используют для оптимизации функций простые модели стохастических, физических, биологических и других процессов. Дополнительная информация о структуре оптимизируемой функции в них, как правило, не используется. Методы не гарантируют нахождение глобального оптимума. Тем не менее, для многих практических задач недетерминистические алгоритмы позволяют относительно быстро находить значение функции, близкое к глобальному оптимуму. При обосновании недетерминистических методов характерны теоремы о сходимости двух типов:

1. теоремы о быстрой квазисходимости утверждают, что метод достаточно быстро сходится к состоянию, когда кандидаты в глобальные оптимумы начинают слабо меняться (утверждением такого типа является, например, теорема “схем” для генетических алгоритмов). Однако, данные утверждения не гарантируют какую-либо близость к глобальному оптимуму.
2. теоремы о глобальной сходимости – метод сходится к глобальному оптимуму (или к его окрестности), однако, данные утверждения не гарантируют быстрой сходимости к нему, выкладки, используемые при доказательстве глобальной сходимости обеспечивают, как правило, лишь скорость сходимости к глобальному оптимуму сопоставимую со случайным поиском.

Таким образом, недетерминистические алгоритмы являются естественными конкурентами предлагаемому в работе методу. Практическое сравнение, однако, показывает, что метод глобальной оптимизации на основе TT-представления оказывается применимым к задачам существенно большей размерности, требует меньше вычисленных значений функции, более удобен в реализации на современных вычислительных системах с учётом многочисленных нюансов, связанных с организацией вычислений и доступом к оперативной памяти.

Эффективность нового метода во многом зависит от реализации крестового алгоритма для приближения тензоров. Несмотря на то, что крестовые методы интерполяции матриц и тензоров имеют низкую теоретическую вычислительную сложность, на практике в случае задач больших размеров или задач с высокой сложностью вычисления тензорных элементов, время на построение аппроксимаций может оставаться весьма существенным. Для решения этой проблемы в диссертационной работе предложены параллельные реализации крестовых методов как для систем с общей памятью, так и для систем с распределённой памятью. Кроме того, для TT-крестового метода интерполяции тензоров предложена новая стратегия эвристического поиска рангов аппроксимации достаточных для достижения требуемой точности.

Все алгоритмы, включая крестовые методы для матриц и тензоров, реализованы в виде библиотеки на языке C++ с использованием технологий параллельного программирования MPI и OpenMP и доступны для свободного использования. Данная библиотека использовалась при реализации численных методов в кандидатских диссертациях С.А. Матвеева и Д.А. Стефонишина.

Целью данной работы является построение метода оптимизации на основе тензорных поездов, теоретическое обоснование метода, разработка параллельных алгоритмов для методов крестовой интерполяции и полученного метода глобальной оптимизации, а также применение метода глобальной оптимизации к прикладным задачам.

Для достижения поставленной цели необходимо было решить следующие **задачи**:

1. Исследовать возможности построения метода глобальной оптимизации на основе тензорных поездов.
2. Выполнить теоретический анализ разработанного метода.
3. Исследовать возможность адаптивного поиска рангов в TT-крестовом методе.
4. Разработать параллельные реализации методов крестовой интерполяции и метода глобальной оптимизации на основе тензорных поездов.
5. Применить метод к различным прикладным задачам.

Научная новизна:

1. Впервые разработан метод глобальной оптимизации на основе тензорных поездов.
2. Произведено частичное обоснование метода глобальной оптимизации на основе тензорных поездов.
3. Предложен оригинальный способ адаптивного поиска ранга для TT-крестового метода.
4. Впервые произведён докинг с подвижным активным центром белка.

Практическая значимость работы заключается в создании открытой библиотеки параллельных методов крестовой интерполяции и глобальной оптимизации на основе тензорных поездов.

Основные положения, выносимые на защиту:

1. Разработаны и реализованы параллельные крестовые методы.
2. Предложена стратегия адаптивного поиска рангов в TT-крестовом методе.
3. Предложен метод глобальной оптимизации на основе тензорных поездов.
4. Приведено частичное обоснование метода глобальной оптимизации на основе тензорных поездов.

5. Метод глобальной оптимизации применён к ряду практических задач:
   - К задаче докинга белок-лиганд.
   - К задаче идентификации параметров моделей ВИЧ-инфекции на клеточном уровне.
   - К задаче оптимизации конфигурации антенн автомобильных радаров.

Достоверность Достоверность полученных результатов подкреплена согласованностью выводов аналитического исследования и численного моделирования.

Апробация работы. Основные результаты работы докладывались на следующих мероприятиях:

1. Международная конференция “Современные проблемы прикладной математики и информатики (MPAMCS 2014)”, Дубна, Россия, 2014.
2. Всероссийская конференция “Тихоновские Чтения-2014”, Москва, Россия, 2014.
3. XXIII Российский национальный конгресс «Человек и лекарство», Москва, Россия, 2016.
4. Международный симпозиум “21st European Symposium on Quantitative Structure-Activity Relationship: Where Molecular Simulations Meet Drug Discovery”, Верона, Италия, 2016.
5. Всероссийская конференция “Ломоносовские чтения - 2017”, Москва, Россия, 2017.
6. Международная конференция “Суперкомпьютерные дни в России 2017”, Москва, Россия, 2017.
7. XXV Российский национальный конгресс «Человек и лекарство», Москва, Россия, 2018.
8. Всероссийская конференция “Ломоносовские чтения-2018”, Москва, Россия, 2018.
9. Международная Конференция “12th International Conference on Large-Scale Scientific Computations (LSSC’19)”, Созополь, Болгария, 2019.
10. Международный симпозиум “10th International Symposium on Computational Methods in Toxicology and Pharmacology Integrating Internet Resources CMTPI-2019”, Янина, Греция, 2019.
11. Всероссийская конференция “Тихоновские Чтения-2019”, Москва, Россия, 2019.

Личный вклад. Все результаты работы получены автором лично под научным руководством академика РАН, д.ф.-м.н., проф. Е.Е. Тыртышникова. В работах, написанных в соавторстве, вклад автора диссертации в полученные результаты в части аналитического исследования, математического моделирования, численных методов и разработки комплекса программ является определяющим.

Публикации. Основные результаты по теме диссертации изложены в 16 печатных изданиях, удовлетворяющих требованиям ВАК [8—23], из них 13 — в периодических научных журналах, индексируемых Web of Science или Scopus [9—15; 18—23], 5 — в тезисах докладов. Зарегистрирована 1 программа для ЭВМ [24].

Объем и структура работы. Диссертация состоит из введения, 6 глав, заключения и 1 приложения. Полный объём диссертации составляет 94 страницы, включая 13 рисунков и 21 таблицу. Список литературы содержит 53 наименования.

Глава 1. Параллельная реализация ТТ-крестового алгоритма

1.1 Немного о тензорных разложениях и их свойствах

Во многих современных прикладных задачах возникают многомерные массивы данных или, как принято говорить, тензоры. Не будет большим преувеличением сказать, что тензоризация данных является одним из главных трендов вычислительной математики. Число индексов в тензоре называется его размерностью. Для современных приложений размерность тензора может достигать нескольких тысяч, а формальное число элементов в таком тензоре с лёгкостью превосходит число атомов во вселенной. Понятно, что для вычислений с такими объектами требуются специальные способы их малопараметрического хранения и быстрой обработки. К настоящему времени наиболее востребованными являются три способа малопараметрических тензорных представлений: каноническое, в формате Таккера и тензорный поезд (ТТ-представление). Мы кратко рассмотрим каждое из этих тензорных представлений.

Начнём с наиболее известного. Каноническое разложение d-мерного тензора $A(i_1, \cdots, i_d) \in {}^{n_1 \times \cdots \times n_d}$ записывается в виде

$$A(i_1, \cdots, i_d) = \sum_{\alpha=1}^{R} u_1(i_1, \alpha) \dots u_d(i_d, \alpha). \tag{1.1}$$

Число R называется каноническим тензорным рангом. Предполагается, что R достаточно мало.

К достоинствам канонического представления (в случае малого ранга R) относятся:

1. компактное хранение, требующее лишь $\mathcal{O}(dnR)$ параметров. Здесь n определяется как $n = \max\limits_{1 \leqslant i \leqslant d} n_i$;
2. некоторые операции с тензорами имеют низкую сложность. Так, например, можно вычислить скалярное произведение двух тензоров за $\mathcal{O}\left(dnR^2\right)$ арифметических операций, применить к тензору d-мерное преобразование Фурье за $\mathcal{O}(dnR \log n)$ операций, умножить тензор вдоль одного из направлений на матрицу за $\mathcal{O}\left(dn^2R\right)$ операций;

3. при выполнении некоторых весьма необременительных условий каноническое разложение единственно с точностью до некоторых тривиальных преобразований. Этот результат известен как теорема Крускала [25], относится к важнейшим свойствам канонического представления и позволяет восстанавливать физически значимые параметры, чем активно пользуются при решении практических задач;
4. каноническое разложение тесно связано с задачей поиска асимптотически быстрых алгоритмов [26].

С другой стороны каноническое представление имеет недостатки, которые затрудняют алгоритмическую работу с ним и существенно ограничивают область его применения в вычислительной математике. Эти недостатки особенно заметны при работе с большими массивами данных. Перечислим лишь некоторые из них:

1. не существует быстрого метода определения тензорного ранга R. Более того, множество тензоров предписанного ранга R может быть незамкнутым;
2. не существует эффективной процедуры приближения заданной точности в каноническом формате;
3. не существует надёжного метода округления (дожатия) тензоров, представленных в каноническом формате.

Указанные недостатки канонического представления не имеют места в представлении Таккера [27]. По определению представлением Таккера тензора $A(i_1, \cdots, i_d)$ называется следующее выражение для его элементов

$$A(i_1, \cdots, i_d) = \sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_d=d}^{r_d} u_1(i_1, \alpha_i) \dots u_d(i_d, \alpha_d) C(\alpha_1, \dots, \alpha_d) \qquad (1.2)$$

Числа $r_1, \dots, r_d$ называются Таккеровскими рангами тензора, тензор $C \in \mathbb{C}^{r_1,\dots,r_d}$ называются ядром разложения Таккера.

При небольших размерностях тензора и рангах, представление Таккера имеет целый ряд существенных достоинств:

1. хранение только $\mathcal{O}) \left(dnr + r^d\right)$ параметров вместо n^d;
2. низкая сложность операций над тензорами;
3. ранги Таккера тензора связаны с рангами специальных матриц развёрток, получаемых переупорядочиванием элементов тензора;
4. существует квазиоптимальный алгоритм нахождения приближения тензора в тензором в формате Таккера:

5. существует квазиоптимальный алгоритм округления (дожатия) тензоров в формате Таккера;
6. существует крестовый метод [28] метод нахождения приближения тензора в тензором в формате Таккера, который использует лишь малое число его элементов;
7. ранги тензора не зависят от перестановки его индексов.

Однако этот формат обладает одним существенным недостатком. Число параметров и алгоритмическая сложность многих операций растут экспоненциально с размерностью тензора d. Поэтому на практике представление Таккера используют только для тензоров небольших размерностей, как правило, не превышающих 5.

В опредёленном смысле единственным универсальным алгебраическим способом работы с большими данными (единственным решением проблемы проклятия размерности для многомерных массивов данных) в вычислительной математике является разложение в тензорный поезд (Tensor Train, TT) [7]. Рассмотрим это малопараметрическое представление подробнее.

Любой тензор $A(i_1, \cdots, i_d) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ может быть представлен в виде

$$A(i_1, \cdots, i_d) = \sum_{\alpha_1=1,\ldots,\alpha_{d-1}=1}^{r_1,\ldots,r_{d-1}} G_1(i_1, \alpha_1) G_2(\alpha_1, i_2, \alpha_2) \ldots G_d(\alpha_{d-1}, i_d). \tag{1.3}$$

Такое представление тензора называется тензорным поездом (Tensor Train, TT). Числа $r_1, \ldots, r_{d-1}$ называются TT-рангами тензора. Для удобства часто вводят дополнительно фиктивные ранги $r_0 = r_d = 1$. Трёхмерные (первый и последний — с учётом фиктивных рангов r_0 и r_1) тензоры $G \in \mathbb{R}^{r_{k-1} \times n_k \times r_k}$ называются TT-ядрами.

Пусть $r = \max\limits_i r_i$, $n = \max\limits_i n_i$. Если TT-ранги тензора относительно небольшие, то TT-разложение обладает рядом критически важных полезных свойств:

1. для хранения тензора требуется всего $\mathcal{O}\left(dnr^2\right)$ элементов вместо $O(n^d)$;
2. вычисление элементов тензора выполняется за $\mathcal{O}\left(dr^2\right)$ арифметических операций;
3. большинство операций над тензорами в TT формате выполняются за $\mathcal{O}\left(dnr^3\right)$ арифметических операций или быстрее;
4. алгоритмическая сложность алгоритма округления в формате тензорного поезда $\mathcal{O}\left(dnr^3\right)$ операций;
5. наличие надёжного квазиоптимального метода TT-SVD получения аппроксимации тензора в TT-формате;

6. ТТ-ранги тензора связаны с рангами специальных матриц развёрток, получаемых перестановкой элементов тензора;
7. возможность построения ТТ-крестового метода интерполяционного типа с использованием лишь $\mathcal{O}\left(dnr^2\right)$ элементов тензора и алгоритмической сложностью порядка $\mathcal{O}\left(dnr^3\right)$.

К некоторому недостатку ТТ-разложения можно отнести то, что ТТ-ранги тензора зависят от перестановки его индексов (с другой стороны это свойство можно рассматривать и как дополнительную возможность при работе с данным форматом).

Когда размерность тензора и количество элементов в нем велико, то получение аппроксимации в ТТ-формате с помощью квазиоптимального алгоритма TT-SVD невозможно. Поэтому крайне важно наличие ТТ-крестового метода интерполяции тензора, позволяющего получить его аппроксимацию, не вычисляя всех элементов. Общая идея такого метода была предложена в [6]. На данный момент существует множество различных реализаций данного алгоритма. Реализации различаются подходами к выбору узлов интерполяции. Поскольку выбор узлов интерполяции определяет точность приближения, то в нашей работе этому вопросу уделяется большое внимание.

Для удобства описания методов введём специальные матрицы, связанные с тензором — матрицы развёрток. $A_k \in \mathbb{R}^{n_1 \dots n_k \times n_{k+1} \dots n_d}$ называется матрицей развёртки номер k тензора $A \in \mathbb{R}^{n_1 \times \dots \times n_d}$, если её элементы являются переупорядоченными элементами тензора

$$A(i_1 \dots i_k, i_{k+1} \dots i_d) = A(i_1, \dots, i_d). \tag{1.4}$$

Ранг k-ой матрицы развёртки совпадает с тензорным рангом r_k.

Отметим, что переупорядовачивание элементов может быть осуществлено разными способами. Используемый порядок не важен, важно лишь чтобы отображение индексов размерностей $2, 3, \dots, d$ в одномерный индекс было взаимооднозначным. Например, оно может быть аналогично используемому в языке Фортран для хранения многомерных массивов

$$i_1 \dots i_k = i_1 + n_1(i_2 - 1 + n_2(i_3 - 1 + \dots)) \tag{1.5}$$

$$i_{k+1} \dots i_d = i_{k+1} + n_{k+1}(i_{k+2} - 1 + n_{k+2}(i_{k+3} - 1 + \dots)) \tag{1.6}$$

Наконец заметим, что также существуют другие тензорные разложения. Среди наиболее известных — HT-разложение (Hierarchical Tucker, иерархическое

разложение Таккера). TT-представление является частным случаем HT. Однако в большинстве практических задач HT-разложение имеет сопоставимое с TT число параметров и сложность, но является значительно более трудным с точки зрения практической реализации алгоритмов.

1.2 Матричный крестовый метод

Матричный крестовый метод, опирающийся на принципе максимального объёма, лежит в основе крестового метода интерполяции для представления тензоров в формате тензорного поезда. Он позволяет строить приближение матрицы $A \in \mathbb{R}^{m\times n}$, используя $\mathcal{O}\left((m+n)r\right)$ её элементов, за $\mathcal{O}\left((m+n)r^2\right)$ операций, причём полученное приближение является квазиоптимальным.

Опишем идею матричного крестового метода [3; 4; 29]. Пусть ранг матрицы $A \in \mathbb{R}^{m\times n}$ равен r. Пусть также выбрано некоторое множество r строк и r столбцов матрицы A. Обозначим $R \in \mathbb{R}^{r\times n}$ подматрицу из соответствующих строк, а $C \in \mathbb{R}^{m\times r}$ подматрицу из соответствующих столбцов. Предположим, что на пересечении выбранных строк и столбцов стоит невырожденная матрица $\widehat{A} \in \mathbb{R}^{r\times r}$. Тогда матрица A может быть представлена следующим образом

$$A = C\widehat{A}^{-1}R. \tag{1.7}$$

В случае, если ранг матрицы A больше r и $\widehat{A}$ является подматрицей максимального объёма, то можно показать, что верна следующая оценка

$$\|A - C\widehat{A}^{-1}R\|_C \leqslant (r+1)\sigma_{r+1}(A). \tag{1.8}$$

Поиск матриц большого объёма может осуществляться с помощью алгоритма MaxVol (maximal volume). Опишем этот алгоритм. Для начала решим задачу поиска подматрицы максимального объёма размера $r \times r$ для матрицы $C \in \mathbb{R}^{m\times r}$. Пусть выбрано некоторое начальное множество из r строк матрицы C так, что подматрица $\widehat{A}_0$, состоящая из этих строк, является невырожденной. Тогда поиск подматрицы максимального объёма можно осуществлять по следующей схеме.

1. $k = 0$.
2. Вычислим матрицу $D_k = C\widehat{A}_k^{-1}$.

3. Найдем в матрице D_k максимальный по модулю элемент. Обозначим его m_k, а номера строки и столбца, в которых он содержится, обозначим как i_k и j_k соответственно.
4. Если $|m_k| \leqslant 1$, то алгоритм заканчивает свою работу. В противном случае построим матрицу $\widehat{A}_{k+1}$ следующим образом:
   - Все строки, кроме строки с номером j_k равны соответствующим строкам матрицы $\widehat{A}_k$.
   - Строка с номером j_k равна строке с номером i_k матрицы C.
5. $k = k + 1$ и переходим к шагу 2.

Покажем, что приведённый алгоритм на каждом следующем шаге переходит к матрице большего объёма. Действительно, в строках матрицы D_k с теми же номерами, что у строк, содержащих матрицу $\widehat{A}_k$ в матрице C, находится единичная матрица. Если заменить строку j_k единичной матрицы на строку с номером i_k матрицы D, то определитель матрицы станет равен m_k, а определитель соответствующей подматрицы матрицы $C - m_k \det \widehat{A}_k$. Таким образом, если $|m_k| > 1$, то объем подматрицы на каждом следующем шаге возрастает. Заметим, что поскольку матрица $\widehat{A}_k$ получается из матрицы $\widehat{A}_{k-1}$ одноранговым обновлением, пересчёт D_k можно осуществлять быстро.

Матричный крестовый метод состоит в последовательном применении алгоритма MaxVol для столбцов и строк матрицы A.

Отметим, что приведённый алгоритм требует знание ранга приближения r заранее. Однако он может быть легко изменён для построения адаптивного ранга приближения. Будем строить приближение матрицы итерационно и на каждом увеличивать ранг приближения на 1, пока не достигнем требуемой точности приближения.

Пусть выполнены k шагов матричного крестового метода и получено приближение $A_k = U_k V_k$. Тогда осуществим добавку ранга 1 для приближения матрицы $A - A_k$. Для этого найдём максимальный по модулю элемент в матрице $A - A_k$, приблизим её крестом ранга 1, а позиции максимального элемента добавим к текущему приближению. На практике часто ограничиваются нахождением достаточно большого по модулю элемента вместо максимального.

Опишем процесс построения матричного крестового приближения с адаптивным рангом детально.

1. $I = \{1, \ldots, m\}$ $J = \{1, \ldots, n\}$ — множество не выбранных ранее строк и столбцов матрицы.

2. Из номеров строк в I выберем случайный номер $\widetilde{i}$.
3. Вычислим строку матрицы $E = A - UV$ с номером $\widetilde{i}$.
4. Выполним поиск максимального по модулю элемента этой строки. Номер столбца, содержащий максимальный по модулю элемент, обозначим j.
5. Вычислим столбец матрицы E с номером j и обозначим его c_{k+1}.
6. Найдем максимальный по модулю элемент этого столбца и обозначим номер строки, содержащий максимальный по модулю элемент i.
7. Вычислим строку матрицы E с номером i и обозначим её r_{k+1}.
8. Выполним проверку точности аппроксимации. Если аппроксимация недостаточно точна, выполним обновление матриц:

$$u = \frac{c_{k+1}}{\sqrt{|A_{ij}|}} \tag{1.9}$$

$$v = \frac{r_{k+1}\sqrt{|A_{ij}|}}{A_{ij}} \tag{1.10}$$

$$U = \begin{bmatrix} U & u \end{bmatrix} \tag{1.11}$$

$$V = \begin{bmatrix} V \\ v \end{bmatrix} \tag{1.12}$$

   В противном случае удалим строку i и столбец j из множества не выбранных ранее $I = I \setminus \{i\}$, $J = J \setminus \{j\}$ и перейдем к шагу 2.

На практике в качестве критерия останова часто используют

$$\varepsilon \|UV\|_F < |A_i j| \sqrt{(m-r)(n-r)}, \tag{1.13}$$

где ε — параметр, отвечающий за точность приближения. Ниже приводится псевдокод построенного алгоритма 1.2.1.

Заметим, что вычислять $\|UV\|_F$ каждый раз заново для проверки точности приближения не требуется. Действительно,

$$\| \begin{bmatrix} U & u \end{bmatrix} \begin{bmatrix} V \\ v \end{bmatrix} \|_F^2 = \|UV\|_F^2 + (U^T u, V v^T) + (V v^T, U^T u) + \|u\|_2^2 \|v\|_2^2 \tag{1.14}$$

Поскольку $\|UV\|_F$ на предыдущем шаге известна, пересчёт значения этой величины может быть произведён за $\mathcal{O}\left((m+n)r\right)$ операций. Таким образом, сложность матричного крестового метода составляет $\mathcal{O}\left((m+n)r^2\right)$ арифметических операций. При этом используется $\mathcal{O}\left((m+n)r\right)$ элементов матрицы. Это может быть

особенно важно, когда нет возможности вычислять все элементы матрицы, например, нахождение элементов матриц требует большого количества вычислений или матрица имеет большие размеры.

Алгоритм 1.2.1 Крестовый метод(m,n,A,ε)

$I \leftarrow \{1, \ldots, m\}$

$J \leftarrow \{1, \ldots, n\}$

$r \leftarrow 0$

while $true$

do
- $r \leftarrow r + 1$
- $i \leftarrow \arg\max\limits_{i \in I} |A_{iJ(1)} - [UV]_{iJ(1)}|$
- $j \leftarrow \arg\max\limits_{j \in J} |A_{ij} - [UV]_{ij}|$
- **if** $\varepsilon \|UV\|_F >= |A_{ij}| \sqrt{(m-r)(n-r)}$
  - **then**
    - $r \leftarrow r - 1$
    - **break**
- $u \leftarrow \dfrac{A^j - [UV]^j}{\sqrt{|A_{ij}|}}$
- $v \leftarrow \dfrac{(A_i - [UV]_i)\sqrt{|A_{ij}|}}{A_{ij}}$
- $U \leftarrow \begin{bmatrix} U & u \end{bmatrix}$
- $V \leftarrow \begin{bmatrix} V \\ v \end{bmatrix}$
- $I \leftarrow I \setminus \{i\}$
- $J \leftarrow J \setminus \{j\}$

return (U,V,r)

1.3 Параллельный матричный крестовый метод

В случае больших размеров матрицы аппроксимация может занимать существенное время. Ускорить метод в этих случаях можно с помощью параллельных алгоритмов. Для простоты изложения будем считать, что размеры матрицы нацело делятся на количество процессов. Будем искать приближение вида

$$A \approx UV. \tag{1.15}$$

Пусть число процессов равно p. Разобьём матрицы U и V при построение матричного крестового метода на p блоков

$$U = \begin{bmatrix} U_1 \\ U_2 \\ \vdots \\ U_p \end{bmatrix}, V = \begin{bmatrix} V_1 & V_2 & \dots & V_p \end{bmatrix}. \tag{1.16}$$

и будем хранить блоки U_k и V_k на k-м процессе. Опишем возможный способ распараллеливания матричного крестового метода на разделенной памяти [17].

1. На каждом процессе заполним номера не выбранных ранее строк и столбцов I и J в соответствии с той частью матриц U и V, которая хранится на данном процессе.
2. Выберем в первом процессе случайный, ранее не выбранный столбец $\widetilde{j}$.
3. Разошлём всем процессам информацию о номере столбца $\widetilde{j}$, а также вычислим какой процесс хранит соответствующий столбец матрицы V. Разошлём также этот столбец всем процессам и обозначим его v.
4. Вычислим столбец $c^0 = A_j^{(k)} - U_k v$ каждым процессом и найдём в нем максимальный по модулю элемент. Здесь $A_j^{(k)}$ обозначена та часть столбца A_j, которая соответствует строкам, хранящимся процессом k.
5. Соберём позиции и величины максимальных по модулю элементов на первом процессе, выберем из них максимальный и отправим соответствующий номер строки i всем процессам.
6. Повторим аналогичную процедуру для выбора максимального по модулю элемента в строке i. В результате получим на первом процессе номер столбца j и значение максимального по модулю элемента $\max$.
7. Вычислим, на каком процессе хранится столбец j матрицы V, разошлём этот столбец всем процессам и обозначим его v.
8. Вычислим добавки к соответствующим блокам матриц U и V на каждом процессе.

$$u_{upd}^{(k)} = \frac{(A^{j,(k)} - U_k v)}{\sqrt{|\max|}} \tag{1.17}$$

$$v_{upd}^{(k)} = \frac{(A_i^{(k)} - u V_k)\sqrt{|\max|}}{\max} \tag{1.18}$$

$$U_k = \begin{bmatrix} U_k & u_{upd}^{(k)} \end{bmatrix} \tag{1.19}$$

$$V_k = \begin{bmatrix} V_k \\ v_{upd}^{(k)} \end{bmatrix} \tag{1.20}$$

9. Если достигнута требуемая точность

$$\|UV\|_F < |\max|\sqrt{(m-r)(n-r)}, \tag{1.21}$$

то завершаем работу алгоритма. В противном случае удаляем столбцы i и j из множеств ранее не выбранных столбцов I и J и переходим к шагу 2.

Заметим, что величину $\|UV\|_F$ также можно вычислять параллельно. Используя выражение

$$\| \begin{bmatrix} U & u \end{bmatrix} \begin{bmatrix} V \\ v \end{bmatrix} \|_F^2 = \|UV\|_F^2 + (U^T u, Vv^T) + (Vv^T, U^T u) + \|u\|_2^2 \|v\|_2^2 \tag{1.22}$$

легко видеть, что слагаемое $\|UV\|_F^2$ с предыдущего шага уже известно. Покажем, как вычислить значение $(U^T u, Vv^T)$. Остальные слагаемые могут быть вычислены аналогично. На каждом процессе вычислим

$$x_k = u_{upd}^{(k)} U_k \tag{1.23}$$

$$y_k = u_{upd}^{(k)} U_k \tag{1.24}$$

Перешлём вычисленные векторы на первый процесс и просуммируем

$$x = \sum_{k=1}^{p} x_k \tag{1.25}$$

$$y = \sum_{k=1}^{p} y_k \tag{1.26}$$

После чего легко видеть, что $x = U^T u$, $y = Vv^t$ и, вычисляя их скалярное произведение, получаем второе слагаемое суммы.

Приведём также псевдокод метода 1.3.1, где k — номер процесса (нумерация начинается с нуля), функция *Broadcast(i, a, b)* рассылает переменную $+a$ с процесса i в переменную b всех процессов, а функция *Gather(i, a, b)* собирает переменную a всех процессов в переменную b. Таким образом, b — массив из переменных a процесса i.

Заметим, что приведённый алгоритм осуществляет $\mathcal{O}(r)$ обменов сообщениями с другими процессами, причём длина каждого сообщения не больше r. Параллельная сложность получившегося метода составляет $\mathcal{O}\left(\frac{m+n}{p} r^2 + rp\right)$ операций.

Алгоритм 1.3.1 Параллельный крестовый метод (m,n,A,p,k)

$b_m \leftarrow \frac{m}{p}$

$b_n \leftarrow \frac{n}{p}$

if $k = 0$

then $J_* \leftarrow \{1, \ldots, n\}$

$I \leftarrow \{b_m k + 1, \ldots, b_m(k+1)\}$

$J \leftarrow \{b_n k + 1, \ldots, b_n(k+1)\}$

$r \leftarrow 0$

repeat

$r \leftarrow r + 1$

if $k = 0$

then $j \leftarrow J_*(1)$

$Broadcast(0, j, j)$

$Broadcast([\frac{j-1}{p}], V^j, v)$

$i \leftarrow \arg\max_{i \in I} |A_{ij} - [Uv]_i|$

$Gather(0, \{i, |A_{ij} - [Uv]_i|\}, max_i)$

if $k = 0$

then $\{i, column\_maximum\} \leftarrow \max_{t \in [1;p]}(max_i[t])$

$Broadcast(0, i, i)$

$Broadcast([\frac{i-1}{p}], U_i, u)$

$j \leftarrow \arg\max_{j \in J} |A_{ij} - [uV]_j|$

$Gather(0, \{j, |A_{ij} - [uV]_j|\}, max_j)$

if $k = 0$

then $\{j, maximum\} \leftarrow \max_{k \in [1;t]}(max_j[t])$

$Broadcast(0, \{j, maximum\}, \{j, maximum\})$

$Broadcast([\frac{j-1}{p}], V^j, v)$

$u_{upd} \leftarrow \frac{(A^j - Uv)[b_m k+1 : b_m(k+1)]}{\sqrt{|maximum|}}$

$v_{upd} \leftarrow \frac{(A_i - uV)[b_n k+1 : b_n(k+1)]\sqrt{|maximum|}}{maximum}$

$U \leftarrow \begin{bmatrix} U u_{upd} \end{bmatrix}$

$V \leftarrow \begin{bmatrix} V \\ v_{upd} \end{bmatrix}$

$I \leftarrow I \setminus \{i\}$

$J \leftarrow J \setminus \{j\}$

if $(k = 0)$

then $J_* \leftarrow J_* \setminus \{j\}$

until $tol\|UV\|_F < |maximum|\sqrt{(m-r)(n-r)}$

return (U,V,r)

1.4 Метод TT-крестовой интерполяции

Пусть $A = A(i_1,\dots,i_d) \in \mathbb{R}^{n_1\times n_2\times\dots\times n_d}$ — тензор. Число d называется размерностью тензора, а про индекс i_k говорят, что он соответствует k-му направлению или k-ой моде. i_k принимает значения от 1 до n_k, n_k называется размером k-ой моды. Общее количество элементов, которое содержит тензор, равно $N = \prod_{k=1}^{d} n_k$. Таким образом, количество памяти, необходимое для хранения тензора, экспоненциально зависит от d.

В данной работе изучаются методы построения приближения тензоров в формате тензорного поезда (Tensor Train, TT) ([7])

$$A(i_1,\dots,i_d) = \sum_{\alpha_1=1,\dots,\alpha_{d-1}=1}^{r_1,\dots,r_{d-1}} G_1(i_1,\alpha_1)G_2(\alpha_1,i_2,\alpha_2)\dots G_d(\alpha_{d-1},i_d), \tag{1.27}$$

где G_i называются ядрами или вагонами тензорного поезда. Для единообразия часто вводят фиктивные индексы $\alpha_0 = 1,\alpha_d = 1$, тогда все вагоны — трёхмерные тензоры.

Заметим, что для хранения тензора в этом формате достаточно $\mathcal{O}\left(dnr^2\right)$ памяти. Кроме того, существует быстрая тензорная арифметика в TT-формате, большинство операций которой выполняются за $\mathcal{O}\left(dnr^3\right)$ действий.

Опишем, как можно построить приближение тензора в TT-формате с помощью TT-крестового метода [6]. Рассмотрим следующую развёртку тензора

$$A_1(i_1,i_2\dots i_d) = A(i_1,\dots,i_d),\ \ A_1 \in \mathbb{R}^{n_1\times n_2\dots n_d}. \tag{1.28}$$

Применим матричный крестовый метод с рангом r_1 к матрице развёртки A_1. Пусть в результате было выбрано множество строк I_1 и множество столбцов J_1. Тогда

$$A_1 \approx A_1(:,J_1)A_1(I_1,J_1)^{-1}A_1(I_1,:). \tag{1.29}$$

Обозначим матрицу $A_1(:,J_1)A_1(I_1,J_1)^{-1}$ размеров $n_1\times r_1$ как G_1, а матрицу $A_1(I_1,:)$, размерность которой $r_1\times n_2\dots n_d$, обозначим как $\widehat{A}_1$. Ясно, что $\widehat{A}_1$ является подматрицей матрицы A_1. Тогда выполнено следующее поэлементное соотношение:

$$A_1(i_1,i_2\dots i_d) \approx \sum_{\alpha_1=1}^{r_1} G_1(i_1,\alpha_1)\widehat{A}_1(\alpha_1,i_2\dots i_d). \tag{1.30}$$

Пусть $\widehat{A}^{(1)}$ является подтензором тензора A, где по первой координате выбраны только индексы из множества I_1. Тогда ясно, что $\widehat{A}_1$ является матрицей развёртки тензора $\widehat{A}^{(1)}$. Так как элементы тензора A и матрицы A_1 по определению совпадают, то выполнено следующее соотношение:

$$A(i_1, i_2, \ldots, i_n) \approx \sum_{\alpha_1=1}^{r_1} G_1(i_1, \alpha_1)\widehat{A}^{(1)}(\alpha_1, i_2, \ldots, i_n). \tag{1.31}$$

Повторим процедуру для тензора $\widehat{A}^{(1)}$. Рассмотрим вторую матрицу развёртки тензора $\widehat{A}^{(1)}$ и обозначим её A_2:

$$A_2(i_1 i_2, i_3 \ldots i_d) = \widehat{A}^{(1)}(i_1, i_2, \ldots, i_d), \quad A_2 \in \mathbb{R}^{r_1 n_1 \times n_2 \ldots n_d}. \tag{1.32}$$

Применим матричный крестовый метод с рангом r_2 к матрице развёртки A_2. Пусть в результате было выбрано множество строк I_2 и столбцов J_2. Тогда

$$A_2 \approx A_2(:, J_2)A_2(I_2, J_2)^{-1}A_2(I_2, :). \tag{1.33}$$

Матрица $A_2(:, J_2)A_2(I_2, J_2)^{-1} \in \mathbb{R}^{r_1 n_2 \times r_2}$ является матрицей развёртки тензора размеров $r_1 \times n_2 \times r_2$, который мы обозначим G_2. А матрица $A_2(I_2, :)$ является матрицей развёртки тензора $\widehat{A}^{(2)}$ размерности $d - 1$ и имеющей размеры $r_2 \times n_3 \times \cdots \times n_d$. Отметим, что тензор $\widehat{A}^{(2)}$ является подтензором исходного тензора A. Таким образом, для тензора $\widehat{A}^{(1)}$ выполнено следующее соотношение:

$$\widehat{A}^{(1)}(r_1, i_2, \ldots, i_n) \approx \sum_{\alpha_2=1}^{r_2} G_2(r_1, i_2, \alpha_2)\widehat{A}^{(2)}(\alpha_2, i_3, \ldots, i_n). \tag{1.34}$$

Подставляя это в выражение (1.31) получаем

$$A(i_1, i_2, \ldots, i_n) \approx \sum_{\alpha_1=1, \alpha_2=1}^{r_1, r_2} G_1(i_1, \alpha_1)G_2(\alpha_1, i_2, \alpha_2)\widehat{A}^{(2)}(\alpha_2, i_3, \ldots, i_n). \tag{1.35}$$

Продолжая процесс, получим аппроксимацию тензора в формате тензорного поезда.

Оценим требуемое число операций приведённого метода. На первом шаге мы строим приближение с помощью матричного крестового метода для матрицы размера $n_1 \times n_2 \ldots n_d$, что требует $\mathcal{O}\left((n_1 + n_2 \ldots n_d)r_1^2\right)$ операций. Это число сопоставимо с числом элементов в тензоре и неприемлемо на практике. Однако предположим, что нам известно относительно небольшое количество номеров

столбцов матрицы A_1, в которых содержится матрица большого объёма. Тогда можно осуществить аппроксимацию лишь на множестве этих столбцов. При этом в процессе будут получены номера строк матрицы развёртки, содержащие большой объем.

Теперь рассмотрим тензор B следующего вида

$$B(i_1, \ldots, i_d) = A(i_d, \ldots, i_1). \tag{1.36}$$

Очевидно, что его матрицы развёрток B_k являются транспонированными матрицами к соответствующим матрицам развёрток тензора A, а именно

$$B_k = A_{d-k}^T. \tag{1.37}$$

Поэтому если для матриц развёрток тензора A известны номера строк, содержащие большой объем, то применяя изложенный выше алгоритм к тензору B, используя эти номера как номера столбцов соответствующих матриц B_k, получим номера столбцов матриц развёрток тензора A, содержащих большой объем.

Метод TT-крестовой интерполяции заключается в последовательном нахождении номеров строк и столбцов матриц развёрток, стартуя со случайных (или выбранных из физических соображений) номеров. При этом каждый раз строится аппроксимация тензора, и по разности между последовательными приближениями можно судить о сходимости метода. Сложность TT-крестового метода — $\mathcal{O}\left(dnr^3\right)$ арифметических операций, при этом метод использует лишь $\mathcal{O}\left(dnr^2\right)$ элементов тензора.

Заметим, что построенный алгоритм требует, чтобы заранее были известны ранги приближения. В данной работе предлагается метод адаптивного нахождения рангов в TT-крестовом методе. Для этого используется матричный крестовый метод с адаптивным выбором ранга, а для аппроксимируемых матриц столбцы при проходе слева направо и строки при проходе справа налево выбираются с некоторым избытком. Опишем процедуру построения TT-крестового метода с адаптивным выбором рангов детально.

Пусть для каждой матрицы развёртки известны наборы индексов элементов тензора $\widetilde{J}_k$, состоящие из “хороших” столбцов, в которых содержится матрица большого объёма. Изначально эти множества могут быть пустыми. Проход слева направо TT-крестового метода осуществляется последовательной работой с матрицами развёрток от A_1 до A_{d-1}:

1. Сформируем набор номеров строк I_k следующим образом. Для каждой точки $i = (i_1, \ldots, i_d)$ из $\widetilde{J}_{k-1}$ добавим n_k номеров строк к I_k : $(i_1, \ldots, i_{k-1}, 1) \ldots (i_1, \ldots, i_{k-1}, n_k)$. В этом множестве строк мы будем искать матрицу максимального объёма. Общее число строк в множестве I_k равно $r_{k-1}n_k$.
2. Формируем множество J_k следующим образом:
   а) Для каждой точки $i = (i_1, \ldots, i_d)$ из $\widetilde{J}_{k-1} \cup \widetilde{J}_k$ добавим к J_k столбец $(i_{k+1}, \ldots, i_d)$.
   б) Дополним J_k случайными столбцами так, чтобы общее число столбцов в J_k было не меньше числа строк в I_k.
3. Выполним матричную крестовую интерполяцию матрицы $A_k(I_k, J_k)$. Обозначим число узлов интерполяции как r_k. Строки и столбцы матрицы A_k, содержащие узлы интерполяции, обозначим как $\widehat{I}_k$ и $\widehat{J}_k$, соответственно.
4. Запишем в качестве ТТ-ядра G_k представленную в виде тензора размеров $r_{k-1} \times n_k \times r_k$ матрицу $A_k(I_k, \widehat{J}_k)A_k(\widehat{I}_k, \widehat{J}_k)^{-1}$.
5. В качестве матрицы G_d запишем матрицу $A_{d-1}(\widehat{I}_{d-1}, :)$.

В результате прохода слева направо получаем тензор в формате тензорного поезда и улучшенные наборы точек интерполяции $\widetilde{J}_k$. Проход справа налево осуществляется последовательной работой с матрицами развёрток от A_{d-1} до A_1 и аналогичен проходу слева направо для тензора B, индексы которого идут в обратном порядке (1.36). В результате, получаем тензор в ТТ-формате и улучшенные наборы точек интерполяции.

Заметим, что в ТТ-формате легко посчитать разность между тензорами и норму тензоров. Поэтому в ТТ-крестовом методе выполняются последовательно проходы справа налево и слева направо до тех пор, пока разность между двумя последовательными приближениями тензора не станет достаточно малой (меньше произведения ε и нормы последнего приближения). Ниже приводится псевдокод построенного алгоритма 1.5.1.

1.5 Параллельная версия TT-крестового метода

Можно построить и параллельную версию TT-крестового метода. Основное её отличие от последовательной будет состоять в том, что индексы столбцов и строк будут формироваться параллельно на соответствующих процессах, и будет использоваться параллельная версия матричного крестового метода. Ниже приводится псевдокод 1.5.2 параллельного TT-крестового интерполяционного метода.

1.6 Численные эксперименты

1.6.1 Тестирование надёжности TT-крестового метода

Был проведён ряд экспериментов для тестирования надёжности предложенной стратегии выбора TT-крестового метода.

В качестве первого примера взята функция $f(x) = \ln(\frac{1}{x})$. На отрезке $[\frac{1}{2^{63}}, 1]$ введена равномерная сетка с 2^{63} узлами. Получившийся вектор $f(x_i)$ был преобразован в 63-мерный тензор с размером 2 каждой из мод, и выполнялась аппроксимация с разными параметрами точности tol. В таблице 1 приведены нормы погрешности E в фробениусовой и чебышёвской нормах на случайной выборке из 100000 элементов, а также затраченное на аппроксимацию время, количество вызовов функции и TT-ранг полученного приближения.

Аналогичные операции были проделаны с функцией $f(x) = \frac{\sin(1000x)}{\sqrt{x}}$, взятой на отрезке $[0, 1000]$, результаты данного эксперимента приведены в таблице 2

Следующий пример - функция $f(x) = \frac{\sin(1000x)}{x}$ на отрезке $[0, 1000]$, результаты представлены в таблице 3.

Следующая функция - $f(x_1,x_2,x_3) = e^{-ikr}$. Каждая из переменных рассматривалась на отрезке $[0; \frac{1}{\sqrt{3}}]$, количество точек сетки по каждой переменной 2^{20}. После квантизации получился 60-мерный тензор, результаты тестирования для значений $k = 1, 10, 100$ приведены в таблицах 4, 5 и 6.

Алгоритм 1.5.1 TT-крестовый алгоритм(d, n, A, tol)

for $k \leftarrow 0$ **to** d

do $\{\widehat{I}_k \leftarrow \varnothing, \widetilde{I}_k \leftarrow \varnothing, \widehat{J}_k \leftarrow \varnothing, \widetilde{J}_k \leftarrow \varnothing$

$TT \leftarrow 0, r_0 \leftarrow 1, r_d \leftarrow 1, diff \leftarrow 1$

while $tol < diff$

do
- **for** $k \leftarrow 1$ **to** $d-1$
  - **do**
    - $\widehat{I}[k] \leftarrow \varnothing$
    - **for** $i \leftarrow 1$ **to** n_k **do** $\widehat{I}_k \leftarrow \widehat{I}_k \cup \{\widehat{I}_{k-1}, i\}$
    - **if** $k < d-1$
      - **then**
        - $\widehat{J}_k \leftarrow \widehat{J}_k \cup \widetilde{J}_k$
        - **while** $|\widehat{J}_k| < n_i * r_{i-1}$ **do** $\widehat{J}_k \leftarrow \widehat{J}_k \cup \{rand(n_{k+1}, n_{k+2}, \ldots, n_d)\}$
      - **else**
        - **for** $i \leftarrow 1$ **to** n_d **do** $\widehat{J_k} \leftarrow \widehat{J_k} \cup \{i\}$
    - $\{r_k, \widehat{I}_k, \widehat{J}_k, C, \widehat{A}, R\} \leftarrow cross\_approximation(A_k, \widehat{I}_k, \widehat{J}_k)$
    - $\widehat{G}_k \leftarrow reshape(\widehat{A}^{-1}R, [r_k, n_{k+1}, r_{k+1}])$
- $\widehat{G_d} \leftarrow reshape(R, [r_{d-1}, n_d, r_d])$
- **for** $k \leftarrow d-1$ **downto** 1
  - **do**
    - $\widetilde{J}[k] \leftarrow \varnothing$
    - **for** $i \leftarrow 1$ **to** n_{k+1} **do** $\widetilde{J}_k \leftarrow \widetilde{J}_k \cup \{i, \widetilde{J}_{k+1}\}$
    - **if** $k > 1$
      - **then**
        - $\widetilde{I}_k \leftarrow \widetilde{I}_k \cup \widehat{I}_k$
        - **while** $|\widetilde{I}_k| < n_{i+1} * r_{i+1}$ **do** $\widetilde{I}_k \leftarrow \widetilde{I}_k \cup \{rand(n_1, n_2, \ldots, n_k)\}$
      - **else**
        - **for** $i \leftarrow 1$ **to** n_1 **do** $\widetilde{I}_k \leftarrow \widetilde{I}_k \cup \{i\}$
    - $\{r_k, \widetilde{I}_k, \widetilde{J}_k, C, \widehat{A}, R\} \leftarrow cross\_approximation(A_k, \widetilde{I}_k, \widetilde{J}_k)$
    - $\widetilde{G}_k \leftarrow reshape(\widehat{A}^{-1}R, [r_k, n_{k+1}, r_{k+1}])$
- $\widetilde{G_1} \leftarrow reshape(C, [r_0, n_1, r_1])$
- $\widehat{TT} \leftarrow TT$
- $TT \leftarrow \{\widetilde{G}, r\}$
- $diff \leftarrow \frac{\|\widehat{TT} - TT\|_F}{\|TT\|_F}$

return (TT)

Алгоритм 1.5.2 Параллельный TT-крестовый алгоритм(d, n, A, tol)

parallel for $k \leftarrow 0$ **to** d

do $\{\widehat{I}_k \leftarrow \varnothing, \widetilde{I}_k \leftarrow \varnothing, \widehat{J}_k \leftarrow \varnothing, \widetilde{J}_k \leftarrow \varnothing$

$TT \leftarrow 0, r_0 \leftarrow 1, r_d \leftarrow 1, diff \leftarrow 1$

while $tol < diff$

do
- **for** $k \leftarrow 1$ **to** $d-1$
  - **do**
    - $\widehat{I}[k] \leftarrow \varnothing$
    - **parallel for** $i \leftarrow 1$ **to** n_k
      - **do** $\widehat{I}_k \leftarrow \widehat{I}_k \cup \{\widehat{I}_{k-1}, i\}$
    - **if** $k < d-1$
      - **then**
        - **if** $processor\_rank = 0$
          - **then**
            - $\widehat{J}_k \leftarrow \widehat{J}_k \cup \widetilde{J}_k$
            - **while** $|\widehat{J}_k| < n_i * r_{i-1}$
              - **do** $\widehat{J}_k \leftarrow \widehat{J}_k \cup \{rand(n_{k+1}, n_{k+2}, \ldots, n_d)\}$
        - $Broadcast(\widehat{J}_k, 0, ALL)$
      - **else**
        - **parallel for** $i \leftarrow 1$ **to** n_d
          - **do** $\widehat{J_k} \leftarrow \widehat{J_k} \cup \{i\}$
    - $\{r_k, \widehat{I}_k, \widehat{J}_k, C, \widehat{A}, R\} \leftarrow parallel\_cross\_approximation(A_k[\widehat{I}_k, \widehat{J}_k], \|\widehat{I}_k\|, \|\widehat{J}_k)\|)$
    - $\widehat{G}_k \leftarrow reshape(\widehat{A}^{-1}R, [r_k, n_{k+1}, r_{k+1}])$
- $\widehat{G_d} \leftarrow reshape(R, [r_{d-1}, n_d, r_d])$
- **for** $k \leftarrow d-1$ **downto** 1
  - **do**
    - $\widetilde{J}[k] \leftarrow \varnothing$
    - **parallel for** $i \leftarrow 1$ **to** n_{k+1}
      - **do** $\widetilde{J}_k \leftarrow \widetilde{J}_k \cup \{i, \widetilde{J}_{k+1}\}$
    - **if** $k > 1$
      - **then**
        - **if** $processor\_rank = 0$
          - **then**
            - $\widetilde{I}_k \leftarrow \widetilde{I}_k \cup \widehat{I}_k$
            - **while** $|\widetilde{I}_k| < n_{i+1} * r_{i+1}$
              - **do** $\widetilde{I}_k \leftarrow \widetilde{I}_k \cup \{rand(n_1, n_2, \ldots, n_k)\}$
        - $Broadcast(\widetilde{I}_k, 0, ALL)$
      - **else**
        - **parallel for** $i \leftarrow 1$ **to** n_1
          - **do** $\widetilde{I}_k \leftarrow \widetilde{I}_k \cup \{i\}$
    - $\{r_k, \widetilde{I}_k, \widetilde{J}_k, C, \widehat{A}, R\} \leftarrow parallel\_cross\_approximation(A_k[\widetilde{I}_k, \widetilde{J}_k], \|\widetilde{I}_k\|, \|\widetilde{J}_k\|)$
    - $\widetilde{G}_k \leftarrow reshape(\widehat{A}^{-1}R, [r_k, n_{k+1}, r_{k+1}])$
- $\widetilde{G_1} \leftarrow reshape(C, [r_0, n_1, r_1])$
- $\widehat{TT} \leftarrow TT, TT \leftarrow \{\widetilde{G}, r\}, diff \leftarrow \frac{\|\widehat{TT} - TT\|_F}{\|TT\|_F}$

return (TT)

Таблица 1 — Тестирование надёжности TT-крестового метода, $f = \ln(\frac{1}{x})$

tol	$\|E\|_F$	$\|E\|_C$	Время, с	Количество вызовов	TT-ранг
10^{-2}	$6.32 * 10^{-3}$	$4.36 * 10^{-3}$	0.023	12260	2.94
10^{-3}	$5.35 * 10^{-4}$	$4.59 * 10^{-4}$	0.029	19661	3.84
10^{-4}	$7.64 * 10^{-5}$	$6.62 * 10^{-5}$	0.036	30801	4.14
10^{-5}	$6.53 * 10^{-6}$	$4.51 * 10^{-6}$	0.032	32409	4.84
10^{-6}	$3.29 * 10^{-7}$	$4.95 * 10^{-7}$	0.036	42317	5.59
10^{-7}	$7.15 * 10^{-8}$	$1.03 * 10^{-8}$	0.049	49384	5.88

Таблица 2 — Тестирование надёжности TT-крестового метода, $f = \frac{\sin(1000x)}{\sqrt{x}}$

tol	$\|E\|_F$	$\|E\|_C$	Время, с	Количество вызовов	TT-ранг
10^{-2}	$6.46 * 10^{-2}$	$1.39 * 10^{-3}$	0.095	73668	3.00
10^{-3}	$4.53 * 10^{-3}$	$2.01 * 10^{-4}$	0.115	115072	3.85
10^{-4}	$1.71 * 10^{-4}$	$3.11 * 10^{-5}$	0.138	164729	4.67
10^{-5}	$2.41 * 10^{-5}$	$1.71 * 10^{-6}$	0.148	221714	5.44
10^{-6}	$2.12 * 10^{-6}$	$1.82 * 10^{-7}$	0.181	288449	6.18
10^{-7}	$1.54 * 10^{-7}$	$3.54 * 10^{-8}$	0.205	348201	6.88

Таблица 3 — Тестирование надёжности TT-крестового метода, $f = \frac{\sin(1000x)}{x}$

tol	$\|E\|_F$	$\|E\|_C$	Время, с	Количество вызовов	TT-ранг
10^{-2}	$1.61 * 10^{-2}$	$7.70 * 10^{-3}$	0.021	5799	1.95
10^{-3}	$5.59 * 10^{-3}$	$1.83 * 10^{-4}$	0.016	8863	2.56
10^{-4}	$3.31 * 10^{-4}$	$5.79 * 10^{-5}$	0.104	75726	3.08
10^{-5}	$1.23 * 10^{-5}$	$2.38 * 10^{-6}$	0.104	112636	3.82
10^{-6}	$6.45 * 10^{-6}$	$2.15 * 10^{-7}$	0.123	143821	4.40
10^{-7}	$3.58 * 10^{-7}$	$2.75 * 10^{-8}$	0.142	184572	5.00

Последний пример - потенциал Хенон-Хайлеса: $f(q_1, \ldots, q_s) = \frac{1}{2}\sum\limits_{k=1}^{s} q_k^2 + \lambda \sum\limits_{k=1}^{s-1}(q_k^2 q_{k+1} - \frac{1}{3}q_k^3)$. $q_i, i = 1, \ldots, s$ изменялись от 0 до 10. Для каждой перемен-

Таблица 4 — Тестирование надёжности реализации TT-крестового метода, $f = e^{-ir}$

tol	$\|E\|_F$	$\|E\|_C$	Время, с	Количество вызовов	TT-ранг
10^{-2}	$1.87 * 10^{-3}$	$1.45 * 10^{-2}$	0.021	6938	3.61
10^{-3}	$3.26 * 10^{-4}$	$5.33 * 10^{-3}$	0.035	13322	5.12
10^{-4}	$1.43 * 10^{-5}$	$8.88 * 10^{-4}$	0.045	54387	8.77
10^{-5}	$1.83 * 10^{-6}$	$4.71 * 10^{-5}$	0.096	144310	12.05
10^{-6}	$2.66 * 10^{-7}$	$1.76 * 10^{-6}$	0.198	276698	16.69
10^{-7}	$6.81 * 10^{-8}$	$7.24 * 10^{-7}$	0.338	625837	23.79

Таблица 5 — Тестирование надёжности TT-крестового метода, $f = e^{-10ir}$

tol	$\|E\|_F$	$\|E\|_C$	Время, с	Количество вызовов	TT-ранг
10^{-2}	$3.55 * 10^{-3}$	$2.12 * 10^{-2}$	0.035	37208	5.97
10^{-3}	$2.63 * 10^{-4}$	$1.38 * 10^{-3}$	0.049	54379	8.55
10^{-4}	$2.76 * 10^{-5}$	$2.11 * 10^{-4}$	0.150	269256	12.53
10^{-5}	$2.83 * 10^{-6}$	$1.00 * 10^{-5}$	0.172	336028	17.25
10^{-6}	$9.29 * 10^{-7}$	$2.64 * 10^{-6}$	0.240	487564	21.95
10^{-7}	$8.51 * 10^{-8}$	$2.29 * 10^{-7}$	0.693	1456133	32.42

Таблица 6 — Тестирование надёжности TT-крестового метода, $f = e^{-100ir}$

tol	$\|E\|_F$	$\|E\|_C$	Время, с	Количество вызовов	TT-ранг
10^{-2}	$2.68 * 10^{-3}$	$3.44 * 10^{-2}$	0.192	350161	17.18
10^{-3}	$1.27 * 10^{-4}$	$1.79 * 10^{-3}$	0.304	593592	24.41
10^{-4}	$4.52 * 10^{-5}$	$8.05 * 10^{-4}$	0.458	937344	28.76
10^{-5}	$4.01 * 10^{-6}$	$6.59 * 10^{-5}$	0.706	1403030	35.86
10^{-6}	$7.38 * 10^{-7}$	$9.75 * 10^{-6}$	1.63	3182595	43.34
10^{-7}	$9.62 * 10^{-8}$	$3.90 * 10^{-7}$	3.717	6847664	56.97

ной было взято по 2^{20} точек. Результаты тестирования для семимерного случая приведены в таблице 7, для двенадцатимерного случая – в таблице 8.

Таблица 7 — Тестирование надёжности ТТ-крестового метода, $f(q_1,\ldots,q_7) = \frac{1}{2}\sum\limits_{k=1}^{7} q_k^2 + \lambda \sum\limits_{k=1}^{6}(q_k^2 q_{k+1} - \frac{1}{3}q_k^3)$

tol	$\|E\|_F$	$\|E\|_C$	Время, с	Количество вызовов	ТТ-ранг
10^{-2}	$3.19 * 10^{-3}$	$3.47 * 10^{-3}$	0.044	26948	3.31
10^{-3}	$3.21 * 10^{-4}$	$3.56 * 10^{-4}$	0.056	31825	3.61
10^{-4}	$4.26 * 10^{-5}$	$5.01 * 10^{-5}$	0.050	22390	3.87
10^{-5}	$4.50 * 10^{-6}$	$4.94 * 10^{-6}$	0.075	60568	4.15
10^{-6}	$6.65 * 10^{-7}$	$1.05 * 10^{-6}$	0.036	27218	4.34
10^{-7}	$6.08 * 10^{-8}$	$9.24 * 10^{-8}$	0.389	357278	4.45

Таблица 8 — Тестирование надёжности ТТ-крестового метода, $f(q_1,\ldots,q_{12}) = \frac{1}{2}\sum\limits_{k=1}^{12} q_k^2 + \lambda \sum\limits_{k=1}^{11}(q_k^2 q_{k+1} - \frac{1}{3}q_k^3)$

tol	$\|E\|_F$	$\|E\|_C$	Время, с	Количество вызовов	ТТ-ранг
10^{-2}	$3.67 * 10^{-3}$	$4.67 * 10^{-3}$	0.103	48913	3.38
10^{-3}	$7.37 * 10^{-4}$	$9.50 * 10^{-4}$	0.064	34249	3.66
10^{-4}	$5.70 * 10^{-5}$	$6.50 * 10^{-5}$	0.075	40088	3.99
10^{-5}	$6.29 * 10^{-6}$	$8.87 * 10^{-6}$	0.096	45193	4.24
10^{-6}	$3.22 * 10^{-7}$	$5.93 * 10^{-7}$	0.086	50408	4.53
10^{-7}	$8.23 * 10^{-8}$	$1.3 * 10^{-7}$	0.868	648575	4.62

Также отметим, что приведённая реализация использовалась в применении к ядрам уравнения Смолуховского в работах [19—21], продемонстрировав в данных работах высокую надёжность работы алгоритма аппроксимации.

1.6.2 Тестирование параллельной производительности ТТ-крестового метода

Для тестирования параллельной производительности был рассмотрен ряд модельных функций. Так как использование параллельного метода предполагается для решения больших задач или аппроксимации тензоров с долговычисля-

емыми элементами, в ряде случаев выполнялась имитация долгого вычисления элемента функции.

В качестве первого примера использовалась функция $f(i_1, i_2, \ldots, i_{10}) = \frac{1}{i_1+i_2+\cdots+i_{10}+1}$. i_k - целочисленные индексы, изменяются от 0 до 10000. Время вычисления функции было искусственно увеличено в 100 раз. После этого функция приближалась параллельным TT-крестовым методом с параметром точности 10^{-4}. Результаты приведены в таблице 9.

Таблица 9 — Тестирование производительности TT-крестового метода, $f(i_1, i_2, \ldots, i_{10}) = \frac{1}{i_1+i_2+\cdots+i_{10}+1}$, время вычисления увеличено в 100 раз

Число процессоров	1	2	4	8	16	32
Время	144.85	75.30	38.02	19.16	9.64	4.91
Ускорение	1	1.92	3.81	7.56	15.03	29.50

Для той же функции, с увеличенным в 1000 раз временем вычисления результаты приведены в таблице 10.

Таблица 10 — Тестирование производительности TT-крестового метода, $f(i_1, i_2, \ldots, i_{10}) = \frac{1}{i_1+i_2+\cdots+i_{10}+1}$, время вычисления увеличено в 1000 раз

Число процессоров	1	2	4	8	16	32
Время	1390.12	698.55	350.16	175.96	88.26	44.24
Ускорение	1	1.99	3.97	7.90	15.75	31.42

Следующий пример $f(\bar{x}) = e^{x^T D x}$, где D - одномерный дискретный оператор Лапласа. $\bar{x}$ – 8-мерный вектор, $x_i, i = 1 \ldots 8$ дискретизированно 10000 точек на отрезке $[0,1]$. Время вычисления увеличено в 100 раз. Результаты приведены в таблице 11.

В качестве третьего примера возьмём функцию модели Вальтерра. Это комплекснозначная функция от 11 комплексный переменных. Комплексные переменные были разбиты на радиус и фазу, и использовалось 2^k значений каждого параметра.

Таким образом, получился 22-мерный тензор с размером каждой моды 2^k. Для $k = 8$ результаты приведены в таблице 12, для $k = 16$ – таблице 13.

Таблица 11 — Тестирование производительности TT-крестового метода, $f(\bar{x}) = e^{\bar{x}^T D \bar{x}}$, время вычисления увеличено в 100 раз

Число процессоров	1	2	4	8	16	32
Время	876.95	447.12	224.86	112.72	56.87	28.56
Ускорение	1	1.96	3.90	7.78	15.42	30.71

Таблица 12 — Тестирование производительности TT-крестового метода, функция модели Вальтерра, $k = 8$

Число процессоров	1	2	4	8	16	32
Время	137.93	70.73	35.55	17.87	8.99	4.83
Ускорение	1	1.95	3.88	7.72	15.35	30.53

Таблица 13 — Тестирование производительности TT-крестового метода, функцию модели Вальтерра, $k = 16$

Число процессоров	1	2	4	8	16	32
Время	6895.19	3448.15	1726.85	864.13	432.39	216.12
Ускорение	1	2.00	3.99	7.98	15.95	31.88

1.6.3 Интегрирование с помощью TT

В качестве ещё одного примера, демонстрирующего как надёжность алгоритма, так и высокую скорость работы, приведём результаты численного интегрирования, по аналогии с примером из [6].

Сравним вычисление интеграла $\int\limits_{[0,1]^d} sin(x_1 + x_2 + \cdots + x_d)$ с помощью метода квази-Монте-Карло и TT-крестового метода.

Для вычисления с помощью TT-крестового метода использовались 11 чебышёвских узлов по каждому направлению.

Для квази-Монте-Карло (КМК) генерировались 2^{30} точек в пространстве.

В таблице 14 приведены результаты сравнения TT-крестового метода с методом квази-Монте-Карло (КМК), а также приведены результаты TT-крестового метода, полученные в [6].

Таблица 14 — Сравнение метода квази-Монте-Карло, реализации TT-крестового метода, предложенной автором, и результатов приведённых в [6] при вычислении интеграла $\int\limits_{[0,1]^d} sin(x_1 + x_2 + \cdots + x_d)$

d	КМК		TT-крест		TT-крест[6]	
	Время	Ошибка	Время	Ошибка	Время	Ошибка
10	93.534	$1.88 * 10^{-10}$	0.00674	$3.52 * 10^{-16}$	0.14	$1.41 * 10^{-15}$
100	434.615	$2.86 * 10^{-3}$	0.0645	$1.87 * 10^{-14}$	0.77	$2.92 * 10^{-13}$
500	777.131	$1.12 * 10^{9}$	0.340	$2.01 * 10^{-13}$	4.64	$2.37 * 10^{-12}$
1000	1573.27	$1.50 * 10^{18}$	1.12	$8.90 * 10^{-13}$	11.60	$1.41 * 10^{-15}$
2000	*	*	4.03	$1.79 * 10^{-12}$	33.05	$8.91 * 10^{-12}$
4000	*	*	15.53	$6.78 * 10^{-12}$	105.49	$2.28 * 10^{-10}$

Глава 2. Оптимизационные свойства крестового метода

Как описано в предыдущей главе, ключевой идеей матричного крестового метода является использование неполных данных: алгоритм использует лишь небольшое число строк и столбцов входной матрицы для построения её аппроксимации.

Другим интересным свойством матричного крестового метода, особенно полезным в задачах оптимизации, является обнаружение им позиций больших по абсолютному значению элементов матрицы. Другими словами, в численных экспериментах было замечено, что несмотря на использование неполных данных (даже в том случае, когда данные зашумлены), матричный крестовый метод за небольшое число итераций просматривает столбцы и строки, на пересечении которых находится либо наибольший по абсолютному значению элемент исходной матрицы, либо элемент с абсолютным значением, очень близким к наибольшему. Соответственно, похожими свойствами обладает и TT-крестовый метод в случае приближения тензоров.

В этой главе будет дано теоретическое обоснование этому экспериментальному наблюдению для частного случая матриц и крестовой интерполяции ранга 1 [18]. В этом случае крестовый алгоритм ищет большой по абсолютному значению элемент a_{ij} матрицы и строит приближение вида $c_j a_{ij}^{-1} r_i$, где c_j и r_i – это, соответственно, столбец и строка, содержащие $a_{ij.}$

Поиск такого элемента осуществляется с помощью специальной процедуры. Вначале задаётся случайный индекс столбца j_0. Далее проводятся следующие итерации (k - номер итерации):

1. рассматриваются элементы входной матрицы, соответствующие столбцу j_{k-1}. Среди них находится наибольший по модулю элемент, и его строчный номер обозначается как i_k:

$$|A_{i_k,j_{k-1}}| \geqslant |A_{i,j_{k-1}}|, \forall i$$

2. рассматриваются элементы входной матрицы, соответствующие строке i_k. Среди них находится наибольший по модулю элемент, и его столбцовый номер обозначается как j_k:

$$|A_{i_k,j_k}| \geqslant |A_{i_k,j}|, \forall j$$

Итерации повторяются либо наперёд заданное число раз, либо до тех пор, пока не выполнится условие $i_k = i_{k-1}$ или $j_k = j_{k-1}$.

2.1 Оценка величины максимального элемента

Построим оценки на величины элементов на пересечениях строк i_k и столбцов j_k, появляющихся по ходу работы матричного крестового метода. Будем сравнивать абсолютные значения элементов, получаемых на итерациях, с наибольшим по модулю элементом всей входной матрицы. Будем предполагать, что рассматриваемая матрица является суммой матрицы ранга 1 и матрицы шума

$$A = \vec{u}\vec{v}^T + E, \vec{u} \in \mathbb{R}^m, \vec{v} \in \mathbb{R}^n; \|u\|_2 = \|v\|_2 = 1,$$

и что поэлементно абсолютная величина шума ограничена величиной $\delta := \|E\|_C$, а относительная - величиной $\varepsilon = \frac{\|E\|_C}{\|\vec{u}\vec{v}^T\|_C}$. Введем дополнительно обозначения

$$\vec{\mu}_u := \frac{|\vec{u}|}{\|\vec{u}\|_\infty}, \quad \vec{\mu}_v := \frac{|\vec{v}|}{\|\vec{v}\|_\infty},$$

где $|\vec{u}|, |\vec{v}|$ – векторы составленные из модулей компонентов векторов u и v, соответственно. Таким образом, $(\vec{\mu}_u)_i \leqslant 1$, $(\vec{\mu}_v)_j \leqslant 1$.

Докажем несколько лемм, характеризующих поведение величин $|(\vec{u}\vec{v}^T)_{i_k,j_k}|$ по ходу крестового алгоритма. Так как

$$|(\vec{u}\vec{v}^T)_{i_k,j_k}| = (\vec{\mu}_u)_{i_k}(\vec{\mu}_v)_{j_k}\|\vec{u}\vec{v}^T\|_C,$$

то близость величин $|(\vec{u}\vec{v}^T)_{i_k,j_k}|$ к максимальному по модулю элементу характеризуется близостью величин $(\vec{\mu}_u)_{i_k}, (\vec{\mu}_v)_{j_k}$ к единице.

Лемма 2.1.1 (Об увеличении элемента)**.** *Пусть* $\varepsilon < \frac{1}{8}$. *Обозначим* $m := \frac{1-\sqrt{1-8\varepsilon}}{2}, M := \frac{1+\sqrt{1-8\varepsilon}}{2}$. *Пусть на некоторой итерации крестового метода*

$$m < (\vec{\mu}_v)_{j_{k-1}} < M.$$

Тогда выполнено

$$(\vec{\mu}_u)_{i_k} > (\vec{\mu}_v)_{j_{k-1}}.$$

Доказательство. Для удобства записи, здесь и в дальнейшем тексте будем опускать стрелки в обозначениях векторов u, v, а также будем использовать сокращенную запись $\mu_{u,i_k} = (\vec{\mu}_u)_{i_k}$, $\mu_{v,j_k} = (\vec{\mu}_v)_{j_k}$. Рассмотрим произвольный элемент столбца j_{k-1} матрицы A

$$a_{i,j_{k-1}} = u_i v_{j_{k-1}} + e_{i,j_{k-1}},$$

где $e_{i,j_{k-1}}$ - соответствующий элемент матрицы шума E. Тогда для абсолютных величин имеем

$$\begin{gathered} |u_i||v_{j_{k-1}}| - \delta \leqslant |a_{i,j_{k-1}}| \leqslant |u_i||v_{j_{k-1}}| + \delta, \\ \mu_{v,j_{k-1}}|u_i|\|v\|_C - \delta \leqslant |a_{i,j_{k-1}}| \leqslant \mu_{v,j_{k-1}}|u_i|\|v\|_C + \delta. \end{gathered} \tag{2.1}$$

Теперь заметим, что справедливо равенство $\|u\|_C\|v\|_C = \|uv^T\|_C$ и предположим, что утверждение леммы не выполняется. То есть $(\vec{\mu}_u)_{i_k} \leqslant (\vec{\mu}_v)_{j_{k-1}}$.

Рассмотрим два строчных индекса

1. Рассмотрим такой строчный индекс $i = i_{max}$, на котором достигается $|u_i| = \|u\|_C$, тогда с помощью левой части 2.1 имеем

$$|a_{i_{max},j_{k-1}}| \geqslant \mu_{v,j_{k-1}}|u_{i_{max}}|\|v\|_C - \delta = \mu_{v,j_{k-1}}\|uv^T\|_C - \delta \tag{2.2}$$

2. Рассмотрим такой строчный индекс i_k, на котором достигается максимум модуля $|a_{i_k,j_{k-1}}|$ элементов столбца j_{k-1}. Воспользуемся правой частью неравенства 2.1

$$|a_{i_k j_{k-1}}| \leqslant \mu_{v,j_{k-1}}|u_{i_k}|\|v\|_C + \delta \leqslant \mu^2_{v,j_{k-1}}\|uv^T\|_C + \delta \tag{2.3}$$

По определению индекса i_k в методе крестовой аппроксимации имеем $|a_{i_k,j_{k-1}}| \leqslant |a_{i_{max},j_{k-1}}|$, откуда, комбинируя неравенства 2.2 и 2.3, имеем

$$\mu^2_{v,j_{k-1}}\|uv^T\|_C + \delta \geqslant \mu_{v,j_{k-1}}\|uv^T\|_C - \delta,$$

откуда, делением на $\|uv^T\|_C$ и переносом всех слагаемых в левую часть получаем

$$\mu^2_{v,j_{k-1}} - \mu_{v,j_{k-1}} + 2\varepsilon \geqslant 0.$$

Теперь, заметим, что корни квадратного трёхчлена $x^2 - x + 2\varepsilon = 0$ имеют вид m, M, и по условию $\mu_{v,j_{k-1}} \in (m, M)$, откуда приходим к противоречию. Таким образом, $(\vec{\mu}_u)_{i_k} \leqslant (\vec{\mu}_v)_{j_{k-1}}$. □

Замечание 1. Очевидно, что для второй части одной итерации матричного крестового метода (т.е. выбора индекса j_k по заданному i_k) можно сформулировать и доказать аналогичное утверждение.

Замечание 2. В условиях предыдущей леммы и при малых ε имеем $m = O(\varepsilon), M = O(1)$. Таким образом, условие леммы предполагает, что значение $|v_{j_{k-1}}|$ превосходит уровень шума, но не “слишком” близко к максимальному $|v_{j_{max}}|$.

Лемма 2.1.2 (О финальном элементе)**.** *Пусть для индекса j_0, поданного на вход матричного крестового метода, выполнено $\mu_{v,j_0} > m$, а после некоторого числа шагов k метода выполнено $i_k = i_{k-1}$ или $j_k = j_{k-1}$ (т.е., элемент a_{i_k,j_k} является максимальным по модулю и в строке i_k, и в столбце j_k матрицы A). Тогда выполнено*

$$\mu_{u,i_k} \geqslant M, \mu_{v,j_k} \geqslant M.$$

Доказательство. Пусть $\mu_{v,j_0} < M$. В этом случае по Лемме 2.1.1 имеем $\mu_{u,i_1} > \mu_{v,j_0} > m$. До тех пор, пока все $\mu_{u,i_s}, \mu_{v,j_s} < M, s = 1, \ldots, k$ выполнено $\mu_{v,j_k} > \mu_{u,i_k} > \mu v, j_{k-1} \cdots > \mu_{v,j_0}$. Так как элементы μ_u, μ_v соответствуют скалированным модулям элементов u, v, совпадение $i_k = i_{k-1}$ или $j_k = j_{k-1}$ возможно, только если для некоторого k выполнено $\mu_{v,j_k} \geqslant M$ или $\mu_{u,i_k} \geqslant M$. Без ограничения общности можно считать, что $\mu_{v,j_k} \geqslant M$.

Аналогично доказательству предыдущей леммы, рассмотрим элементы $a_{i_{max}j_k}$ и $a_{i_{k+1}j_k}$, соответствующие максимальным по модулю элементам матриц uv^T и A в столбце j_k:

$$\begin{aligned} |a_{i_{max}j_k}| &\leqslant \mu_{v,j_k}\|uv^T\|_C - \delta \\ |a_{i_{k+1}j_k}| &\leqslant \mu_{v,j_k}\mu_{u,i_{k+1}}\|uv^T\|_C + \delta \end{aligned}$$

Так как $|a_{i_{k+1}j_k}| \geqslant |a_{i_{max}j_k}|$, то:

$$\begin{aligned} \mu_{v,j_k}\mu_{u,i_{k+1}}\|uv^T\|_C + \delta &\geqslant \mu_{v,j_k}\|uv^T\|_C - \delta \\ \mu_{v,j_k}(1 - \mu_{u,i_{k+1}}) &\leqslant 2\varepsilon \end{aligned} \tag{2.4}$$

Так как $2\varepsilon = mM$, $M+m = 1$ (m,M - корни квадратного трёхчлена $x^2-x+2\varepsilon = 0$), и $\mu_{v,j_k} \geqslant M$, то имеем:

$$\begin{aligned} 1 - \mu_{u,i_{k+1}} &\leqslant m \\ \mu_{u,i_{k+1}} &\geqslant 1 - m = M, \end{aligned}$$

поэтому если $\mu_{v,j_k} \geqslant M$, то и $\mu_{u,i_{k+1}} \geqslant M$, а также и все последующие величины $\mu_{v,j_{k+1}}, \mu_{u,i_{k+2}}, \ldots$ не меньше, чем M. Откуда следует утверждение леммы. □

Лемма 2.1.3 (Об элементе после трёх шагов метода)**.** *Обозначим* $\tilde{m} := 4\varepsilon, \tilde{M} := 1 - 4\varepsilon$. *Пусть* $\mu_{v,j_0} \geqslant \tilde{m}$*, тогда выполнено:*

$$\mu_{v,j_1} \geqslant \tilde{M}, \mu_{u,i_1} \geqslant \tilde{M}$$

Замечание 3. $\tilde{m} > m$, $\tilde{M} < M$, где m,M заданы как в предыдущих леммах. При малых ε выполнено $\tilde{m} \approx m$, $\tilde{M} \approx M$.

Доказательство. Если $\mu_{v,j_0} \geqslant M$ или $\mu_{u,i_1} \geqslant M$ то, по Лемме 2.1.2, $\mu_{v,j_1} \geqslant M > \tilde{M}$, $\mu_{u,i_2} \geqslant M > \tilde{M}$.

Рассмотрим случай $\mu_{v,j_0} < M, \mu_{u,i_1} < M$. Воспользуемся неравенством 2.4 из Леммы 2.1.2:

$$\begin{aligned} \mu_{v,j_0}\mu_{u,i_1}\|uv^T\|_C + \delta &\geqslant \mu_{v,j_0}\|uv^T\|_C - \delta, \\ \mu_{u,i_1} &\geqslant 1 - \frac{2\varepsilon}{\mu_{v,j_0}} \end{aligned}$$

Аналогично, если $\mu_{u,i_1} < M$, то:

$$\mu_{v,j_1} \geqslant 1 - \frac{2\varepsilon}{\mu_{u,i_1}}$$

Поэтому, если $\mu_{v,j_0} \geqslant \tilde{m}$, то

$$\begin{aligned} \mu_{u,i_1} &\geqslant 1 - \frac{2\varepsilon}{\mu_{v,j_0}} \geqslant 1 - \frac{2\varepsilon}{\tilde{m}} = \frac{1}{2} \\ \mu_{v,j_1} &\geqslant 1 - \frac{2\varepsilon}{\mu_{u,i_1}} \geqslant 1 - 4\varepsilon = \tilde{M} \end{aligned}$$

Теперь, если $\mu_{v,j_1} \geqslant M$, то, как показано в Лемме 2.1.2, $\mu_{u,i_2} \geqslant M > \tilde{M}$, в ином случае по Лемме 2.1.1 $\mu_{u,i_2} > \mu_{v,j_1} \geqslant \tilde{M}$. □

Замечание 4. Таким образом, если $\mu_{v,j_0} \geqslant \tilde{m}$, то за три полушага матричного крестового метода, соответствующих поиску максимального по модулю элемента —

в двух столбцах и в одной строке (при старте поиска со столбца), можно найти позицию элемента невозмущённой матрицы, модуль которого отличается от максимального по модулю элемента невозмущённой матрицы не более чем в $\tilde{M}^2$ раз, где $\tilde{M} \to 1$ при $\varepsilon \to 0$. Это означает, что число внутренних итераций крестового метода для этого частного случая ограничено 3. Однако не сложно привести веские доводы в пользу того, что аналогичная ситуация будет и в случае поиска подматриц большого объёма произвольного порядка k.

Действительно, рассмотрим матрицу $A \in \mathbb{C}^{m\times n}$, которая представляется суммой матрицы ранга k и матрицы возмущения E. Если упростить рассмотрение, то крестовый алгоритм в этом случае действует следующим образом: (а) стартует с некоторого набора из k столбцов, в которых находит матрицу наибольшего объёма (с наибольшим значением модуля определителя) из всех подматриц порядка k, находящихся в этих столбцах; (б) в строках, связанных с полученной на предыдущем шаге подматрицы, снова находится подматрица наибольшего объёма и так далее. Алгоритм останавливается, когда найденная подматрица, обладает наибольшим объемом среди всех подматриц, находящихся в её столбцах и строках.

Свяжем с матрицей A присоединённую подматрицу, составленную из миноров порядка k. Хорошо известно [30], что для матрицы A ранга k присоединённая подматрица будет ранга 1. В нашем случае A – сумма матрицы ранга 1 и возмущения, поэтому присоединённая подматрица (по крайней мере в случае малого возмущения E) будет приближаться матрицей ранга 1. Более того, алгоритм поиска подматрицы порядка k в матрице A соответствует поиску наибольшего элемента в присоединённой матрице. Теперь остаётся только воспользоваться доказанными в вышеприведённых леммах утверждениями.

Таким образом, остаётся ответить на вопрос, насколько достижимо условие последней леммы вида $\mu_{v,j_0} \geqslant \tilde{m}$.

Исследуем этот вопрос в двух случаях: (а) когда стартовый столбец j_0 выбирается на основании анализа некой случайной выборки столбцов матрицы A; (б) когда метод стартует с полностью случайного j_0.

2.2 Оценка вероятности удачного старта для выборки столбцов

Рассмотрим следующий вариант выбора начального столбца j_0 матричного крестового метода: пусть до первой итерации метода вычисляются все элементы s столбцов входной матрицы A, где $s > 1, s \ll n$. Пусть тогда j_0 выбирается как столбец, содержащий наибольший по модулю элемент матрицы A среди всех s рассмотренных столбцов.

Покажем, что при таком старте с целой выборки столбцов, и при условия наличия в этой выборке достаточно большого элемента, матричный крестовый метод сохраняет свои свойства, описанные в предыдущем параграфе.

Лемма 2.2.1 (О старте с выборки столбцов)**.** *Пусть среди s столбцов матрицы A существует столбец j такой, что $\mu_{v,j} > m$. Пусть a_{i_1,j_0} – максимальный по модулю элемент матрицы входных данных среди всех элементов этих s столбцов (возможно, $j \neq j_0$). Тогда выполнено*

$$\mu_{u,i_1} > \min(\mu_{v,j}, M)$$

Доказательство. Аналогично доказательствам лемм предыдущего параграфа, применим левую и правую часть неравенства 2.1 к двум элементам входной матрицы A. Рассмотрим следующие элементы:

1. Элемент $a_{i_{max},j}$, где i_{max} соответствует наибольшей по модулю компоненте вектора u:

$$|a_{i_{max},j}| \geqslant \mu_{v,j}\|u\|_C\|v\|_C - \delta.$$

2. Элемент a_{i_1,j_0}, выбранный как наибольший по модулю среди входного множества столбцов:

$$|a_{i_1 j_0}| \leqslant \mu_{u,i_1}\mu_{v,j_0}\|u\|_C\|v\|_C + \delta$$

Теперь, используя полученные два неравенства и $|a_{i_1 j_0}| > |a_{i_{max},j}|$, имеем

$$\begin{aligned} \mu_{u,i_1}\mu_{v,j_0}\|u\|_C\|v\|_C + \delta &\geqslant \mu_{v,j}\|u\|_C\|v\|_C - \delta \\ \mu_{u,i_1}\mu_{v,j_0} &\geqslant \mu_{v,j} - 2\varepsilon \end{aligned}$$

Теперь, если $\mu_{v,j} \leqslant M$, то по лемме 2.1.1 $\mu_{v,j}^2 - \mu_{v,j} + 2\varepsilon \leqslant 0$, откуда $\mu_{v,j} - 2\varepsilon \geqslant \mu_{v,j}^2$, поэтому:

$$\mu_{u,i_1}\mu_{v,j_0} \geqslant \mu_{v,j}^2$$

Отсюда следует, что хотя бы одно число из $\mu_{u,i_1}, \mu_{v,j_0} \geqslant \mu_{v,j}$. Рассмотрим два случая.

1. Пусть $\mu_{v,j} \leqslant M$. Тогда, если $\mu_{v,j_0} \geqslant \mu_{v,j}$, то в случае $\mu_{v,j_0} < M$ по Лемме 2.1.1, а так как элемент a_{i_1,j_0} максимален по модулю в своём столбце, получаем $\mu_{u,i_1} > \mu_{v,j_0} \geqslant \mu_{v,j}$.
   В случае $\mu_{v,j_0} \geqslant M$ то, как показано в Лемме 2.1.2, $\mu_{u,i_1} \geqslant M \geqslant \mu_{v,j}$.
2. Пусть $\mu_{v,j} > M$; заметим, что $M - 2\varepsilon = M^2$, т.к., как было ранее введено, m, M - корни квадратного трехчлена $x^2 - x + 2\varepsilon = 0$. Откуда имеем

   $$\mu_{v,j} - 2\varepsilon > M^2$$
   $$\mu_{u,i_1}\mu_{v,j_0} > M^2.$$

   Значит, хотя бы одно число из $\mu_{u,i_1}, \mu_{v,j_0} > M$. Но если $\mu_{v,j_0} > m$ то, как показано в Лемме 2.1.2, $\mu_{u,i_1} \geqslant M$.

□

Замечание 5. Аналогичное утверждение можно сформулировать и доказать в случае, когда метод крестовой аппроксимации начинает с выбора строки i_0 как строки, содержащей наибольший по модулю элемент среди набора из s строк входной матрицы A.

Понятно, что легко привести матрицу ранга 1 (например, только один ненулевой элемент которой не равен нулю) такую, что среди случайного набора из ее s столбцов больших элементов не окажется. Однако, ниже покажем, что для “большинства” матриц вероятность события среди элементов из набора s случайных столбцов матрицы A существует элемент, для которого выполнено требуемое в леммах условие $\mu_{v,j_0} \geqslant \tilde{m}$ высока. Для этого будем рассматривать матрицы uv^T, задаваемые векторами u, v, случайно распределенными по единичной сфере. Докажем вспомогательную техническую лемму.

Лемма 2.2.2. *Пусть $x \in \mathbb{R}$ - случайная величина с распределением χ^2 с $n > 2$ степенями свободы. Тогда:*

$$\mathbb{P}(x > (1+\alpha)n) \leqslant \sqrt{\frac{n}{\pi\alpha^2}} e^{-\frac{n(\alpha - \ln(1+\alpha))}{2}}$$

Доказательство. Проинтегрируем функцию плотности распределения χ^2 и оценим гамма-функцию снизу с помощью формулы Стирлинга ($\Gamma(n) = \frac{n!}{n} \geqslant \sqrt{\frac{2\pi}{n}}\left(\frac{n}{e}\right)^n$):

$$\mathcal{P} = \mathbb{P}\left(x > (1+\alpha)n\right) = \int\limits_{(1+\alpha)n}^{\infty} \frac{x^{\frac{n}{2}-1}e^{-\frac{x}{2}}}{2^{\frac{n}{2}}\Gamma\left(\frac{n}{2}\right)}dx \leqslant \int\limits_{(1+\alpha)n}^{\infty} \frac{x^{\frac{n}{2}-1}e^{-\frac{x}{2}}\sqrt{n}}{2^{\frac{n}{2}+1}\sqrt{\pi}\left(\frac{n}{2e}\right)^{\frac{n}{2}}}dx \tag{2.5}$$

Произведём замену $y = x - n$:

$$\mathcal{P} \leqslant \sqrt{\frac{n}{\pi}}\int\limits_{\alpha n}^{\infty} \frac{(y+n)^{\frac{n}{2}-1}\,e^{-\frac{y+n}{2}}}{2^{\frac{n}{2}+1}\left(\frac{n}{2e}\right)^{\frac{n}{2}}}dx = \sqrt{\frac{n}{\pi}}\int\limits_{\alpha n}^{\infty} \frac{\left(1+\frac{y}{n}\right)^{\frac{n}{2}-1}e^{-\frac{y}{2}}}{2}dy \tag{2.6}$$

Воспользуемся заменой $z = y - n\ln\left(1+\frac{y}{n}\right), dy = \frac{n+y}{y}dz$:

$$\mathcal{P} \leqslant \sqrt{\frac{n}{\pi}}\int\limits_{n(\alpha-\ln(1+\alpha))}^{\infty} \frac{\left(1+\frac{y}{n}\right)^{-1}\frac{n+y}{y}e^{-\frac{z}{2}}}{2}dz = \sqrt{\frac{n}{\pi}}\int\limits_{n(\alpha-\ln(1+\alpha))}^{\infty} \frac{ne^{-\frac{z}{2}}}{2y}dz \tag{2.7}$$

Заметим, что $\frac{n}{y} \leqslant \frac{1}{\alpha}$:

$$\mathcal{P} \leqslant \sqrt{\frac{n}{\pi\alpha^2}}\int\limits_{n(\alpha-\ln(1+\alpha))}^{\infty} \frac{e^{-\frac{z}{2}}}{2}dz = \sqrt{\frac{n}{\pi\alpha^2}}e^{-\frac{n(\alpha-\ln(1+\alpha))}{2}} \tag{2.8}$$

□

Следствие 2.2.2.1. *Если выбрать* $\alpha = \frac{\pi}{2} - 1$, *получаем:*

$$\mathbb{P}\left(x > \frac{\pi}{2}n\right) \leqslant \sqrt{\frac{n}{\pi\left(\frac{\pi}{2}-1\right)^2}}e^{-\frac{n\left(\frac{\pi}{2}-\ln\frac{\pi}{2}-1\right)}{2}} < \sqrt{n}e^{-\frac{n}{18}}$$

Теперь, для случайного вектора с единичной сферы проведём оценку модуля наибольшего элемента из некоторой выборки элементов. Будем опираться на предыдущую лемму. Для этого будем использовать известный факт, что нормированный случайный вектор, каждый элемент которого первоначально задан в соответствии с нормальным распределением, равномерно распределён на единичной сфере.

Лемма 2.2.3. *Пусть случайный вектор v равномерно распределён на единичной сфере в $\mathbb{R}^n$. Тогда, с вероятностью не ниже $1 - \sqrt{\frac{n}{\pi\alpha^2}}e^{-\frac{n(\alpha-\ln(1+\alpha))}{2}} - \left(c\sqrt{\frac{2(1+\alpha)n}{\pi}}\right)^s$ среди s случайно выбранных в нём элементов найдётся хотя бы один, по модулю больший, чем c.*

Доказательство. Вектор v, равномерно распределенный по единичной сфере, может быть получен следующим образом: пусть каждая компонента вспомогательного вектора x выбрана случайно по стандартному нормальному распределению, и проведена последующая нормировка $v = \frac{x}{\|x\|}$. Тогда $|x_i|^2 \sim \chi^2(1)$ и $|v_i|^2 = \frac{|x_i|^2}{\sum\limits_{j=1}^{n}|x_j|^2}$.

Из Леммы 2.2.2 следует, что:

$$\mathbb{P}\left(\sum_{j=1}^{n}|x_j|^2 > (1+\alpha)n\right) \leqslant \sqrt{\frac{n}{\pi\alpha^2}}e^{-\frac{n(\alpha-\ln(1+\alpha))}{2}}.$$

В то же время $\mathbb{P}\left(|x_i|^2 < t^2\right) = \mathbb{P}\left(|x_i| < t\right) = 2\int\limits_0^t \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}}dx \leqslant \sqrt{\frac{2t^2}{\pi}}$. Тогда вероятность того, что при случайном выборе s элементов вектора x все они по модулю меньше t не превосходит $\left(\frac{2t^2}{\pi}\right)^{\frac{k}{2}}$, откуда

$$\mathbb{P}\left(|x_i|^2 < c^2(1+\alpha)n, i = \overline{1,\ldots,s}\right) \leqslant \left(c\sqrt{\frac{2(1+\alpha)n}{\pi}}\right)^s.$$

Таким образом, вероятность того, что все s выбранных компонент вектора v по модулю меньше c, не превосходит $\sqrt{\frac{n}{\pi\alpha^2}}e^{-\frac{n(\alpha-\ln(1+\alpha))}{2}} + \left(c\sqrt{\frac{2(1+\alpha)n}{\pi}}\right)^s$. □

Следствие 2.2.3.1. *В частном случае $\alpha = \frac{\pi}{2} - 1$ имеем более простую оценку: вероятность наличия среди s выбранных элементов элемента с модулем, не меньших c, не ниже*

$$1 - \sqrt{n}e^{-\frac{n}{18}} - (c\sqrt{n})^s.$$

Дополнительно рассмотрев частный случай $s = \beta\ln n$, получим оценку вида

$$1 - \sqrt{n}e^{-\frac{n}{18}} - n^{\beta\ln(c\sqrt{n})}.$$

2.3 Оценка вероятности удачного старта для случайного столбца

Теперь рассмотрим полностью случайный вариант выбора стартового столбца для алгоритма матричной крестовой аппроксимации. Оценим, в том же предположении о случайном распределении факторов u, v по единичной сфере, вероятность выбора такого столбца j_0, для которого выполнено условие $\mu_{v,j_0} \geqslant \tilde{m}$, требуемое в леммах о поиске больших элементов матрицы с помощью матричного крестового метода.

Сформулируем достаточное условие для выполнения условия $\mu_{v,j_0} \geqslant m$:

Лемма 2.3.1. *Пусть* $\mu_0 > 0$, $0 < \varepsilon_0 < 1$. *Пусть элемент* a_{ij} *матрицы входных данных* A, *удовлетворяет следующим условиям:*

$$|v_j| > \mu_0, \tag{2.9}$$

$$u_i v_j E_{ij} > 0, \tag{2.10}$$

$$\mu_0 |u_i| \geqslant \mu_0 \mu \|u\|_\infty + \varepsilon_0 \delta, \tag{2.11}$$

$$|E_{ij}| \geqslant (1 - \varepsilon_0)\delta, \tag{2.12}$$

тогда $\mu_{u,i} > m$.

Доказательство. Воспользуемся условием 2.10:

$$|a_{ij}| = |u_i v_j + E_{ij}| = |u_i v_j| + |E_{ij}|.$$

Из условия 2.12 получаем:

$$|a_{ij}| = |u_i v_j| + |E_{ij}| \geqslant |u_i||v_j| + (1 - \varepsilon_0)\delta = \mu_0 |u_i| \tfrac{|v_j|}{\mu_0} + (1 - \varepsilon_0)\delta$$

$$|a_{ij}| \geqslant \mu \|u\|_C |v_j| + \varepsilon_0 \delta \tfrac{|v_j|}{\mu_0} + (1 - \varepsilon_0)\delta$$

$$|a_{ij}| > \mu \|u\|_C |v_j| + \delta$$

□

Теперь оценим вероятность того, что эти достаточные условия выполнены в случае задания факторов u, v случайными и равномерно распределенными по единичной сфере. Для этого докажем следующие технические утверждения.

Лемма 2.3.2. *Пусть каждый элемент матрицы шума* E *распределён случайно и независимо по равномерному распределению на отрезке* $[-\delta; \delta]$. *Рассмотрим*

один столбец матрицы E. Тогда вероятность того, что в этом столбце ровно l элементов, не меньших по модулю $(1-\varepsilon_0)\delta$, равна

$$P_l = C_m^l \varepsilon_0^l (1-\varepsilon_0)^{m-l}$$

Доказательство.

$$\begin{aligned} P_l &= P(|\{i : |e_{ij}| \geqslant (1-\varepsilon_0)\delta\}| = l) = \\ &= C_m^l (P(|e_{ij}| \geqslant (1-\varepsilon_0)\delta))^l (P(|e_{ij}| \geqslant (1-\varepsilon_0)\delta))^{m-l} \\ P_l &= C_n^l \varepsilon_0^l (1-\varepsilon_0)^{n-l}. \end{aligned}$$

□

Лемма 2.3.3. *Пусть вектор u равномерно распределён на единичной сфере. Обозначим $\gamma := \left(\mu + \frac{\varepsilon_0\delta}{\mu_0}\right)\sqrt{\frac{2}{\pi}} \leqslant \frac{1}{2}$. Тогда, с вероятностью не ниже*

$$1 - \frac{(4\gamma(1-\gamma))^{\frac{n}{2}}}{2}$$

не меньше половины элементов вектора u удовлетворяют условию $\mu_0|u_i| \geqslant \mu_0\mu\|u\|_\infty + \varepsilon_0\delta$.

Доказательство. Как показано в доказательстве Леммы 2.2.3, $\mathbb{P}(|u_i| < t) \leqslant \sqrt{\frac{2t^2}{\pi}}$. Поэтому $\mathbb{P}\left(|u_i| < \sqrt{\frac{\pi}{2}}\gamma\right) \leqslant \gamma$. Пусть $s \leqslant \frac{n}{2}, s \in \mathbb{N}$. Тогда вероятность того, что ровно $n-s$ элементов вектора u не превосходят по модулю величины γ, не превышает

$$C_n^{n-s}\gamma^{n-s}(1-\gamma)^s = (\gamma(1-\gamma))^{\frac{n}{2}} C_n^{n-s}\left(\frac{\gamma}{1-\gamma}\right)^{\frac{n}{2}-s} \leqslant (\gamma(1-\gamma))^{\frac{n}{2}} C_n^{n-s}.$$

Подстановкой $s = \frac{n}{2}$ имеем требуемую оценку вида

$$\sum_{s=\left[\frac{n+1}{2}\right]}^{n} (\gamma(1-\gamma))^{\frac{n}{2}} C_n^{n-s} \leqslant (\gamma(1-\gamma))^{\frac{n}{2}} 2^{n-1} = \frac{(4\gamma(1-\gamma))^{\frac{n}{2}}}{2}$$

□

2.4 Теоремы об оптимизационных свойствах крестового метода

В этом параграфе сформулируем теоремы, объединяющие результаты предыдущих лемм для обоих способов выбора начального столбца - по элементу

с наибольшим модулем из случайной выборки небольшого числа столбцов, либо полностью случайным образом.

Теорема 2.4.1 (О максимизации модуля крестовым методом при старте с выборки столбцов)**.** *Пусть вектор v равномерно распределён на единичной сфере. Пусть стартовый индекс столбца j_0 матричного крестового метода выбирается как столбец, содержащий наибольший по модулю элемент среди всех элементов первых s столбцов матрицы входных данных. Тогда:*

1. *После трёх шагов матричного крестового метода для индексов найденного элемента $a_{i_1 j_1}$ выполнено, что $|u_{i_1} v_{j_1}| \geqslant \tilde{M}^2 \|uv^T\|_C$ с вероятностью не ниже:*
$$1 - \sqrt{\frac{n}{\pi\alpha^2}} e^{-\frac{n(\alpha - \ln(1+\alpha))}{2}} - \left(4\|v\|_C \varepsilon \sqrt{\frac{2(1+\alpha)n}{\pi}} \right)^s$$
2. *Если найденный элемент a_{i_1, j_1} является максимальным по модулю и в соответствующей ему строке, и в соответствующем ему столбце матрицы входных данных, то $|u_{i_1} v_{j_1}| \geqslant M^2 \|uv^T\|_C$ с вероятностью не ниже:*
$$1 - \sqrt{\frac{n}{\pi\alpha^2}} e^{-\frac{n(\alpha - \ln(1+\alpha))}{2}} - \left(\|v\|_C \frac{1 - \sqrt{1 - 8\varepsilon}}{2} \sqrt{\frac{2(1+\alpha)n}{\pi}} \right)^s$$

Доказательство. Из Леммы 2.2.3 следует, что среди первых s компонент вектора v содержится компонента v_{j_0} такая что $|v_{j_0}| \geqslant \tilde{m}\|v\|_C$ с вероятностью не ниже
$$1 - \sqrt{\frac{n}{\pi\alpha^2}} e^{-\frac{n(\alpha - \ln(1+\alpha))}{2}} - \left(4\|v\|_C \varepsilon \sqrt{\frac{2(1+\alpha)n}{\pi}} \right)^s$$
Таким образом, с той же вероятностью $\mu_{v,j_0} \geqslant \tilde{m}$. Поэтому, по Лемме 2.2.1 максимальный по модулю элемент среди элементов первых s столбцов матрицы входных данных содержится в строке i_1, такой что $\mu_{u,i_1} \geqslant \tilde{m}$. Следовательно, после трёх шагов метода по Лемме 2.1.3 найденный элемент $a_{i_1 j_1}$ будет таким, что $|u_{i_1} v_{j_1}| \geqslant \tilde{M}^2 \|uv^T\|_C$.

Второе утверждение доказывается аналогично, однако используется m вместо $\tilde{m}$ и Лемма 2.1.2 вместо Леммы 2.1.3. □

Следствие 2.4.1.1. *Если $\alpha = \frac{\sqrt{\pi}}{2} - 1, s = \beta \ln n$ и $\varepsilon \leqslant \frac{1}{8\|v\|_c \sqrt{n}}$, то утверждения теоремы выполнены с вероятностью не ниже*
$$1 - \sqrt{n} e^{-\frac{n}{18}} - n^{-\beta \ln 2} = 1 - O(n^{-\beta \ln 2})$$

Теорема 2.4.2 (О максимизации модуля крестовым методом при старте со случайного столбца)**.** *Пусть $m = n$ (матрица A квадратная) и два вектора u, v выбраны случайно распределены на единичной сфере. Пусть каждый элемент матрицы E - независимая случайная величина, равномерно распределённая на отрезке $[-\delta, \delta]$. Тогда с вероятностью не ниже*

$$1 - O\left(\left(\frac{n}{\ln n}\right)^{-k}\right) \tag{2.13}$$

выполнено:

1. *Если найденный после k шагов метода элемент a_{i_k,j_k} является максимальным по модулю и в соответствующей ему строке и в соответствующем ему столбце матрицы входных данных, его содержащих, то $|u_{i_k} v_{j_k}| \geqslant M^2 \|uv^T\|_C$.*
2. *После $k + 3$ шагов матричного крестового метода для найденных индексов $i_{k+3} j_{k+3}$ выполнено, что $|u_{i_{k+3}} v_{j_{k+3}}| \geqslant \tilde{M}^2 \|uv^T\|_C$.*

Доказательство. Отметим, что в силу независимости uv^T и E вероятность обнаружения большого элемента входной матрицы на случайной выборке из небольшого числа s ее столбцов такая же, как и вероятность обнаружения элемента того же модуля на случайной выборке s строк той же матрицы.

Согласно Лемме 2.3.3, с вероятностью не менее $1 - \frac{(4\gamma(1-\gamma))^{\frac{n}{2}}}{2}$ не меньше половины элементов вектора u удовлетворяют условию 2.11. Аналогичное утверждение верно и для вектора v. Таким образом, вероятность того, что оба события выполняются одновременно, не меньше

$$1 - \left(4\gamma\left(1-\gamma\right)\right)^{\frac{n}{2}}.$$

В этом случае при случайном выборе элемента столбца (или строки) матрицы вероятность выполнения условия 2.11 не ниже $\frac{1}{2}$.

Вероятность выполнения условия 2.10 для элемента матрицы (в силу независимости u, v, E) равна $\frac{1}{2}$. Таким образом, вероятность одновременного выполнения условий 2.10 и 2.11 не ниже $\frac{1}{4}$. Поэтому вероятность того, что хотя бы одно из этих условий нарушено не выше $\frac{3}{4}$.

Используя Лемму 2.3.2 получаем, что вероятность того, что ни один из элементов столбца (строки) не удовлетворяет хотя бы одному из трёх условий 2.10,

2.11 и 2.12 не выше

$$\begin{aligned}\sum_{l=0}^{n} C_n^l \varepsilon_0^l (1-\varepsilon_0)^{n-l} \frac{3^l}{4^l} &= \sum_{l=0}^{n} C_n^l (1-\varepsilon_0)^{n-l} \left(\frac{3}{4}\varepsilon_0\right)^l \\ &= \left(1-\varepsilon_0+\frac{3}{4}\varepsilon_0\right)^n \\ &= \left(1-\frac{1}{4}\varepsilon_0\right)^n.\end{aligned}$$

Наконец, рассмотрим вероятность того, что условие 2.9 не выполняется ни для одного столбца, рассмотренного на протяжении k итераций матричного крестового метода. Её можно ограничить величиной

$$P^k(|v_j| \leqslant \mu_0) \leqslant \left(\mu_0\sqrt{\frac{2}{\pi}}\right)^k$$

Объединяя полученные оценки, вероятность того, что хотя бы одной из условий 2.9, 2.10, 2.11, 2.12 не выполнено, не превосходит

$$\left(4\gamma\left(1-\gamma\right)\right)^{\frac{n}{2}} + \left(1-\frac{1}{4}\varepsilon_0\right)^n + \left(\mu_0\sqrt{\frac{2}{\pi}}\right)^k.$$

Положим $\mu_0 := \frac{8k\delta\ln n}{\sigma n(1-8\varepsilon)}$, $\varepsilon_0 = \frac{4k\ln n}{n}$, тогда:

$$\gamma = \sqrt{\frac{2}{\pi}}\left(\mu + \frac{\varepsilon_0\delta}{\sigma\mu_0}\right) \leqslant \frac{1}{2}\sqrt{\frac{2}{\pi}} = \sqrt{\frac{1}{2\pi}} < \frac{2}{5}$$

.

$$\left(1-\frac{1}{4}\varepsilon_0\right)^n = \left(1-\frac{k\ln n}{n}\right)^n \leqslant n^{-k}$$

Тогда вероятность, что среди всех просмотренных элементов существует элемент, удовлетворяющий всем условиям 2.9, 2.10, 2.11, 2.12 не ниже:

$$1 - a^{-n} - n^{-k} - \left(\frac{n}{b\ln n}\right)^{-k} = 1 - O\left(\left(\frac{n}{\ln n}\right)^{-k}\right).$$

где $a = \frac{25}{24}$, $b = \frac{8k\delta\sqrt{2}}{\sigma(1-8\varepsilon)\sqrt{\pi}}$.

По Лемме 2.3.1 элемент удовлетворяет условиям 2.9, 2.10, 2.11, 2.12, поэтому для содержащей его строки i выполнено $\mu_{u,i} > m$. Аналогичное утверждение справедливо и для столбцов. Для доказательства первого утверждения остаётся воспользоваться Леммой 2.1.2.

Для доказательства второго утверждения заметим, что замена m на $\tilde{m}$ не меняет хода рассуждений, поэтому с вероятностью не менее приведённой в формуле 2.13 среди просмотренных матричным крестовым методом k строк и столбцов существует элемент, удовлетворяющий условиям 2.9, 2.10, 2.12 и условию 2.11, в котором m заменено на $\tilde{m}$. Лемма 2.3.1 в этом случае будет утверждать, что для строки i, содержащей удовлетворяющий условиям элемент, выполнено $\mu_{u,i} > \tilde{m}$. Очевидно, что аналогичное верно и для столбцов. Согласно Лемме 2.1.3 после ещё 3 шагов для найденного элемента $a_{i_{k+3}j_{k+3}}$ будет выполнено $\mu_{u,i_{k+3}} > \tilde{M}$, $\mu_{v,j_{k+3}} > \tilde{M}$. □

Глава 3. Метод глобальной оптимизации на основе тензорных поездов

Во многих прикладных задачах возникает проблема поиска глобального оптимума. При этом в большинстве случаев детерминистические методы глобальной оптимизации требуют слишком большого времени (часто десятки или сотни лет на наиболее быстрых суперкомпьютерах) для получения результата, а результаты, получаемые с помощью “жадных” алгоритмов или алгоритмов, основанных на методах выпуклой релаксации, далеки от глобального оптимума. В таких случаях, как правило, применяют недетерминистические методы глобальной оптимизации, которые работают как “чёрный ящик” и не используют структуру оптимизируемого функционала. Это позволяет применять недерминистические методы даже тем, кто не является специалистами в оптимизации, что объясняет широкую популярность таких методов в различных прикладных областях.

В данной работе представлен метод глобальной оптимизации на основе принципа максимального объёма и крестового метода приближения многомерных массивов в формате тензорного поезда [8; 16; 22]. В главе 2 приведено доказательство того, что в матричном случае крестовый метод поиска однорангового приближения с большой вероятностью выбирает в качестве узлов интерполяции узлы, близкие к глобальному оптимуму, причём даже в том случае, когда исходная матрица плохо приближается или просто не приближается матрицей ранга 1 во фробениусовой и/или спектральной нормах.

Представленное в главе 2 доказательство может быть расширено на d-мерный случай. Однако вероятностные модели одноранговых матриц, которые используются при доказательстве матричного результата, малопригодны в случае массивов большого числа измерений, поскольку получаемые с их помощью оценки малоинтересны уже для случая размерности более 4. Нам представляется, что это в большей степени недостаток метода доказательства. На практике, как мы увидим, метод продолжает работать и для большого числа измерений.

Мы рассматриваем метод глобальной оптимизации на основе тензорных поездов как альтернативу существующим недетерминистическим алгоритмам. Идея метода глобальной оптимизации на основе тензорных поездов заключается в применении метода TT-крестовой интерполяции с ограничением максимального допустимого ранга аппроксимации. В качестве результата оптимизации исполь-

зуется наилучшая просмотренная в процессе интерполяции точка. Для ускорения сходимости метода также можно использовать локальную оптимизацию или жадный поиск среди точек близких к данной.

3.1 Математическая постановка задачи.

Приведём детальное описание решаемой задачи. Пусть требуется найти оптимум функционала $f(x)$ в некоторой области, являющейся d-мерным параллелепипедом. В этом случае по каждому направлению можно ввести сетку. Значения функционала на сетке образуют тензор $A(i_1, \cdots, i_d)) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$. Если сетка достаточно мелкая, а функционал непрерывен в окрестности оптимумов, то позиция и значения оптимального элемента тензора будут близки к позиции и значению оптимума функционала в исходной области (при оптимизации дискретного аргумента значения и позиции будут совпадать).

Однако, как и в случае TT-крестового метода, тензорная часть алгоритма глобальной оптимизации осуществляет поиск не любого оптимального элемента, а лишь максимального по модулю элемента. Поэтому важной частью метода глобальной оптимизации является выбор отображения исходной задачи в эквивалентную задачу максимизации модуля. Такое отображение должно удовлетворять следующим условиям:

1. является строго монотонным;
2. гладким (в противном случае тензорные ранги тензора могут сильно возрасти, только вследствие применения данного отображения);
3. не приводит к появлению большого числа нулей (это приводит к ухудшению работы метода);
4. близкие к оптимальному значения должны как можно сильнее отличаться друг от друга.

Например, для задачи минимизации функционала, про область значений которого ничего не известно заранее, неплохим выбором является функционал $\mathrm{arcctg}(f - f_*)$, где f_* – минимум, найденный на момент начала итерации метода.

3.2 Последовательная версия алгоритма

Последовательная версия метода оптимизации на основе тензорных поездов, как и метод TT-крестовой интерполяции тензоров, состоит в последовательном применении матричного крестового метода к специальным образом выбираемым подматрицам матриц развёртки тензора.

Матрица $A_k \in \mathbb{R}^{n_1 \dots n_k \times n_{k+1} \dots n_d}$ называется k-ой матрицей развёртки тензора $A \in \mathbb{R}^{n_1 \times \dots \times n_d}$, если её элементы являются переставленными элементами данного тензора:

$$A_k(i_1 \dots i_k, i_{k+1} \dots i_d) = A(i_1, \dots, i_d)$$

В этой записи $i_1 \dots i_k \in [1, n_1 \dots n_k]$ (и аналогично $i_{k+1} \dots i_d \in [1, n_{k+1} \dots n_d]$) означает любое взаимооднозначное отображение k-мерного индекса $(i_1, \dots, i_k) \in [1, n_1] \times \dots \times [1, n_k]$ в одномерный индекс, например,

$$i_1 \dots i_k = i_1 + n_1(i_2 - 1) + \dots + n_1 \dots n_{d-1}(i_d - 1)$$

Пусть исходно стоит задача глобальной оптимизации для функции $f(x)$ в d-мерном параллелепипеде. Введём отображение $g(y)$, преобразующее данную задачу в эквивалентную задачу поиска максимального по модулю элемента. Как уже обсуждалось ранее, данный функционал может зависеть от найденного на начало итерации метода оптимума, то есть, вообще говоря, меняться от итерации к итерации.

Также введём сетку на этом d-мерном параллелепипеде. Тензор значений функционала $g(f(x))$ на этой сетке обозначим через G (но не будем его вычислять).

Введём для каждой матрицы развёртки G_k множества индексов (если нет дополнительной информации о задаче – то исходно пустые множества) строк I_k и столбцов J_k, которые потенциально содержат большие элементы.

Тогда итерация метод глобальной оптимизации с рангом не более r_{max} выглядит следующим образом:

I. Последовательно для каждой матрицы развёртки $G_k, k = \overline{1 \dots d-1}$:

1. Сформируем множество индексов строк $\hat{I}_k$: в него войдут все индексы из I_k, а также для каждого индекса $i_1 \dots i_{k-1} \in I_{k-1}$ добавим к множеству индексов все индексы вида $i_1 \dots i_{k-1} i$, $i = \overline{1 \dots n_k}$. Кроме того, добавим в него индексы строк, содержащие наибольшие найденные элементы.

2. Сформируем множество индексов столбцов $\hat{J}_k$: в него войдут все индексы из J_k, а также для каждого индекса $j_k \dots j_d \in J_{k-1}$ добавим к этому множеству индекс $j_{k+1} \dots j_d$. Кроме того, добавим в него индексы строк, содержащие наибольшие найденные элементы.
3. Дополним при необходимости множество индексов $\hat{J}_k$ случайными так, чтобы его мощность равна минимуму из $2|\hat{I}_k|$ и $n_{k+1} \dots n_d$.
4. Выполним крестовую интерполяцию подматрицы $G_k(\hat{I}_k, \hat{J}_k)$ с рангом не выше r_{max}.
5. Обозначим как I_k и J_k множества индексов строк и столбцов матрицы развёртки G_k, содержащие полученные узлы интерполяции.

   Также существенно ускоряет сходимость метода добавление в множества I_k и J_k индексов строк и столбцов, соответствующих ближайшим точкам на сетке к точкам, полученным в результате локальной оптимизации узлов интерполяции, при наличии процедуры, осуществляющей быстрый локальный поиск оптимума функционала.

II. Выполним аналогичные операции для тензора $\tilde{G}$, который соответствует G с записанными в обратном порядке индексами:

$$\tilde{G}(i_d, \dots, i_1) = G(i_1, \dots, i_d),$$

где при формировании подматриц матриц развёрток будем использовать множества индексов строк $\tilde{I}_k = J_{d-k}$ и столбцов $\tilde{J}_k = I_{d-k}$.

Сложность итерации полученного метода $\mathcal{O}\left(dnr_{max}^2\right)$ вычислений элементов, $\mathcal{O}\left(dnr_{max}^3\right)$ арифметических операций и $\mathcal{O}\left(dr_{max}\right)$ локальных оптимизаций (если последние используются).

3.3 Искусственное увеличение размерности

Идея искусственного увеличения числа измерений хорошо известна, например, в программировании: одномерный массив (вектор) часто рассматривается как двумерный массив (матрица), а иногда и как массив с числом измерений $d > 3$ (в частности, при развёртке циклов). Введение виртуальных (искусственных) измерений для тензоров было предложено в [31]; в этой же работе впервые даны оценки величин, теперь называемых QTT-рангами тензора.

Логарифмичность числа операций и хранимых элементов при использовании ТТ-формата делает идею искусственного увеличения размерности особенно привлекательной [32; 33]. Для простоты изложения будем считать, что $n_1 = \cdots = n_d = n$. Пусть $n = 2^m$. Преобразуем заданный d-мерный тензор $A \in \mathbb{R}n \times \cdots \times n$ в dm-мерный тензор $B \in \mathbb{R}^{2\times\dot{\times}2}$. Это, вообще говоря, можно сделать многими способами,например так:

$$B(i_{11}, \ldots, i_{m1}, i_{12}, \ldots, i_{m2}, \ldots, i_{1d}, \ldots, i_{md}) = A(j_1, j_2, \ldots, j_d),$$

где

$$j_k = i_{1k} + 2i_{2k} + \cdots + 2^{m-1} i_{mk}.$$

Если тензор A в ТТ-формате занимает $\mathcal{O}\left(dnr^2\right)$ ячеек памяти и операции с ним требуют $\mathcal{O}\left(dnr^3\right)$ арифметических действий, то тензор B в том же формате занимает $\mathcal{O}\left(dmR^2\right) = \mathcal{O}\left(dR^2 \log_2 n\right)$ ячеек памяти и операции с ним требуют $O(dmR^3) = O(dR^3 \log_2 n)$ действий. Очевидно, что $R \geqslant r$, и, как правило, $R > r$, поэтому в ряде случаев такое преобразование может не привести к уменьшению объёма хранимых данных и/или ускорению выполнения операций с тензором. Однако во многих практических задачах значение R близко к r, а в таких случаях возможен существенный выигрыш как по памяти, так и по числу операций. В задаче глобальной оптимизации, очевидно, при одинаковом ограничении на максимальный ранг r_{max} тензор искусственно увеличенной размерности будет требовать меньшего числа операций, чем исходный тензор.

Для простоты рассмотрим задачу оптимизации в непрерывном случае. В данном случае, как правило, мы можем выбирать на d-мерном параллелепипеде размеры сетки произвольно, и поэтому возможно ввести сетку размера $2^{m_1} \times \ldots 2^{m_d}$. Как и в методе без искусственного увеличение размерности, воспользуемся отображением, переводящим задачу глобальной оптимизации в эквивалентную задачу поиска максимального по модулю элемента, и построим (но не вычислим) d-мерный тензор $A \in \mathbb{R}^{2^{m_1}\times\cdots\times 2^{m_d}}$ значений функционала на данной сетке. Рассмотрим D-мерный ($D = \sum_{i=1}^{d} m_d$) тензор $B \in \mathbb{R}^{2\times\cdots\times 2}$, который получается из тензора A путём добавления виртуальных размерностей.

Далее работаем с тензором B аналогично работе с тензором A в случае без использования виртуальных размерностей.

Отметим, что для случая виртуальных размерностей все аппроксимируемые методом матрицы относительно небольшие. Поэтому использование лишь

параллельной версии метода крестовой интерполяции матриц имеет весьма ограниченный ресурс параллельности. Для увеличения параллельного ресурса ниже предложена параллельная по размерности тензора версия алгоритма.

3.4 Параллельная версия алгоритма

Несмотря на то, что метод глобальной оптимизации на основе TT разложения обладает относительно низкой сложностью, особенно при использовании искусственного увеличения размерности, для больших задач или при затратном вычислении значении функционала в точке, оптимизация может занимать существенное время. Отметим, что, при этом, особенно при использовании идеи искусственного увеличении размерности, использование лишь параллельной версии матричного крестового метода может быть недостаточно и не приводить к существенному ускорению работы. Для лучших параллельных свойств необходимо независимо обрабатывать матрицы развёрток.

Пусть в тензоре A дан набор P позиций, соответствующих наилучшие найденные значения исходного функционала. Кроме того, пусть для каждой матрицы развёртки задан набор позиций P_k узлов интерполяции и близких по значению к оптимуму элементов. Тогда итерация параллельного метода оптимизации выглядит следующим образом:

I. Построим (но не вычислим!) тензор $B(i_1, \ldots, i_d) = g(A(i_1, \ldots, i_d))$, где g – выбранное для этой итерации отображение исходной задачи в задачу поиска максимального по модулю элемента.

II. Для каждой матрицы B_k развёртки независимо выполним:

1. Сгенерируем случайные позиции R_k в тензоре. Количество позиций возьмём равное мощности множества $P \cup P_{k-1} \cup P_k \cup P_{k+1}$.
2. Сформируем множества строк I_k и столбцов J_k: для каждой точки $i = (i_1, \ldots, i_d)$ из множества $P \cup P_{k-1} \cup P_k \cup P_{k+1} \cup R_k$ добавим:
   - К множеству строк I_k:
     строки $(i_1, \ldots, i_{k-1}, 1) \ldots (i_1, \ldots, i_{k-1}, n_k)$.
   - К множеству столбцов J_k:
     столбцы $(1, i_{k+2}, \ldots, i_d) \ldots (n_{k+1}, i_{k+2}, \ldots, i_d)$.

3. Выполним интерполяцию матрицы $B_k(I_k, J_k)$ с рангом не выше заданного r_{max} с помощью матричного крестового метода.
4. Запишем в P_k узлы интерполяции, кроме того, если возможно выполнить локальную оптимизацию узлов интерполяции, для каждого узла добавим ближайшую точку тензора к локально оптимизированному узлу интерполяции.

III. Выберем r_{max} наилучших по значению функционала точек и запишем в P.

Из описания алгоритма видно, что метод глобальной оптимизации на основе TT-разложения базируется на TT-крестовом методе с ограниченным максимальным рангом. Однако для большей параллельности, матрицы развёрток на каждой итерации метода рассматриваются не последовательно, как при проходе справа налево (или слева направо) в TT-крестовом методе, а независимо. При этом точки интерполяции различных матриц развёрток теряют некоторую вложенность, присущую TT-крестовому методу (например, при проходе слева направо для любой точки интерполяции k-ой матрицы развёртки существует такая точка интерполяции $k-1$ матрицы развёртки, что их первые $k-1$ индексов совпадает). Это приводит к тому, что построить аппроксимацию тензора таким же способом, как в TT-крестовом методе, невозможно.

Тем не менее, вычислив дополнительно небольшое число элементов тензора, возможно восстановить аппроксимацию, используя полученные в ходе работы метода глобальной оптимизации точки интерполяции. В настоящее время, однако, такая аппроксимация не строится.

3.5 Замечание об аппроксимационных свойствах алгоритма

Очевидно, что предложенный метод глобальной оптимизации на основе алгоритма крестовой интерполяции тензоров и сам крестовый метод имеют много общего. Отметим, что даже в случае параллельной версии метода, в которой нельзя построить аппроксимацию на основе лишь вычисляемых в процессе работы метода элементов, её можно построить добавив небольшое (не более dnr^2) дополнительных элементов тензора, используя полученные в результаты работы метода индексы элементов.

К недостаткам разработанного метода глобальной оптимизации можно отнести то, что он строит некоторую (возможно, весьма грубую) аппроксимацию тензора, которая явно не используется как источник дополнительной информации о позициях потенциально близких к глобальному оптимуму значений функционала.

Глава 4. Применение алгоритма глобальной оптимизации к задаче докинга

4.1 Постановка задачи

Расчёт энергии связывания белка и лиганда (органической молекулы — кандидата в ингибиторы белка-мишени) является одной из ключевых задач применения компьютерного молекулярного моделирования для разработки новых лекарственных средств [34]. Когда лиганд связывается со специфическим участком связывания на белке-мишени, он начинает производить те или иные биохимические, физиологические или фармакологические эффекты. Правильное позиционирование лиганда в белке-мишени даёт, как следствие, и оценку их энергии связывания.

Докинг — один из подходов в молекулярном моделировании [35]. Он предполагает, что на многомерной энергетической поверхности системы белок-лиганд находится то положение лиганда, при котором потенциальная энергия этой системы минимальна. Даже если считать белок-мишень жёстким, число степеней свобод системы велико. Действительно, положение и ориентация лиганда задаются 6 параметрами, а для описания внутренних вращательных степеней свободы, кручений, требуется зачастую не менее 10 параметров. Таким образом, для решения задачи докинга необходимо использовать эффективные методы глобальной оптимизации в многомерном пространстве, в частности разработанный в настоящей диссертации метод на основе TT-разложений.

Эффективность того или иного метода глобальной оптимизации применительно к задаче докинга имеет смысл рассматривать с двух точек зрения. Во-первых, это быстродействие, то есть количество вычислений функционала энергии в ходе работы алгоритма. Расчёт энергии системы — наиболее трудоёмкая часть в методах докинга, даже с учётом того, что используются приближённые модели. Во-вторых, это согласованность получаемых результатов с известными экспериментальными данными.

Положение лиганда в активном центре белка обычно определяется методами рентгеноструктурного анализа закристализованного комплекса белок-лиганд, и соответствующие данные (положение лиганда в кристалле называют его нативным положением) помещаются в базу Protein Data Bank [36]. Важно, однако,

иметь в виду, что нативное положение лиганда вообще говоря не обязано ни соответствовать положению лигандов в белках в реальных организмах, ни доставлять глобальный минимум функционалу энергии. С одной стороны это связано с тем, что для получения кристалла систему белок-лиганд помещают в условия, отличные от свойственных организму. С другой стороны — в реальности лиганд перескакивает между локальными минимумами энергии в результате теплового движения.

4.2 Критерии эффективности алгоритмов

В качестве критериев эффективности методов стоит рассматривать

1. малое количество вычислений функционала энергии в ходе работы алгоритма,
2. возможность нахождения сразу нескольких вариантов положения лиганда, среди которых есть близкие к нативному,
3. низкое значение энергии найденных положений лиганда.

Знание нескольких локальных минимумов с достаточно низкой потенциальной энергией также позволяет оценить энтропийную составляющую энергия связывания, а значит более точно её вычислить.

Зафиксируем некоторую пару белок-лиганд и обозначим через $\mathbf{p}$ вектор параметров (по числу степеней свобод), задающих положение лиганда. Каждому набору параметров $\mathbf{p}$ отвечает определённое положение атомов, образующих лиганд, то есть вектор $\mathbf{x} = \mathbf{x}(\mathbf{p})$. Пусть энергия системы вычисляется как $\mathcal{E}_p(\mathbf{p}) = \mathcal{E}_x(\mathbf{x}(\mathbf{p}))$.

В зависимости от того, как устроен алгоритм, его результатом будет набор из k положений лиганда, заданных либо в виде параметров $\{\hat{\mathbf{p}}_j\}_{j=1}^k$, либо в виде положений атомов $\{\hat{\mathbf{x}}_j\}_{j=1}^k$. Нативное положение лиганда в базе Protein Data Bank задаётся положением его атомов $\mathbf{x}_N$. Поскольку, вообще говоря, нативное положение не доставляет глобальный минимум функционалу $\mathcal{E}_x$, проведём локальную дооптимизацию энергии $\mathcal{E}_x$ в окрестности нативного положения $\mathbf{x}_N$ и получим положение атомов $\mathbf{x}_{ON}$, близкое к нативному.

Введём индекс i_{NN}, индекс ближайшего элемента к нативному, то есть являющийся решением задачи

$$i_{NN} = \arg\min_{1 \leqslant i \leqslant k} i \quad \text{s.t.} \quad \|\mathbf{x}_N - \hat{\mathbf{x}}_i\|_2 \leqslant 2\overset{\circ}{A},$$

если оно существует; иначе $i_{NN} = \infty$. Аналогично введём

$$i_{NON} = \arg\min_{1 \leqslant i \leqslant k} i \quad \text{s.t.} \quad \|\mathbf{x}_{ON} - \hat{\mathbf{x}}_i\|_2 \leqslant 2\overset{\circ}{A}.$$

Индексам i_N и i_{NN} можно дать следующую интерпретацию. Если они близки к 1, то найденный алгоритмом минимум энергии системы белок-лиганд находится вблизи нативного положения.

4.3 Численные эксперименты

В численных экспериментах энергия $\mathcal{E}_x$ вычислялась в соответствии с силовым полем MMFF94 в вакууме [37] без использования сетки потенциалов. При этом задача минимизации функционала энергии была заменена на эквивалентную задачу поиска максимального по модуля значения функционала

$$\Phi(\mathbf{x}) = \exp[100\mathrm{arcctg}(\mathcal{E}_x(\mathbf{x}) - \mathcal{E}^*)], \tag{4.1}$$

где через $\mathcal{E}^*$ обозначен минимум энергии на предыдущем шаге алгоритма.

Мы сравнивали следующие параллельные программы для докинга:

SOL-P — основана на представленном в диссертации методе глобальной оптимизации с использованием TT-разложений [8],

FLM — основана на поиске методом Монте–Карло [10].

Все вычисления проводились на суперкомпьютере Ломоносов.

Жёсткий белок. В первой серии экспериментов [9] сравнивались две программы: переборная FLM и программа SOL-P, основанная на TT-разложении. Обе программы запускались по 1000 раз для 30 выбранных комплексов белок-лиганд из Protein Data Bank. Для глобальной TT-оптимизации была выбрана равномерная сетка из $n = 2^8$ точек для каждой из степеней свобод; для TT-рангов были взять значения $r_{\max} = 16$, $r_{\max} = 32$ и $r_{\max} = 64$.

В ходе экспериментов считалось, что белок жесткий. В таком случае размерность задачи определялась числом торсионных кручений лиганда. Для выбранных комплексов оно варьировалось в диапазоне между 0 и 19, то есть число степеней свободы было от 6 до 25.

Результаты представлены в Таблице 15. Видно, что для большинства комплексов программа FLM находит более низкое значение функционала энергии $\mathcal{E}_x$, чем программа SOL-P. Полученные же значения индекса i_{NN} показывает, что SOL-P справляется с задачей поиска положений лиганда, близких к нативному. При этом программа SOL-P на порядки быстрее, чем FLM: если запуск FLM для одного комплекса белок-лиганд занял около 10^8 ядросекунд, то SOL-P с рангом 16 — около 10^6 ядросекунд, а SOL-P с рангом 64 — около 10^7 ядросекунд.

Важно отметить, что способность находить положения лиганда, близкие к нативному, почти не зависит от величины ранга TT-разложения. Это наблюдение согласуется с Теоремой 2.4.2, которая говорит о том, что положение максимального по модулю элемента в тензоре находится с помощью разработанного в диссертации алгоритма даже в том случае, когда TT-аппроксимация далека от точной.

Белок с подвижными атомами. Выше была рассмотрена модель с неподвижным жёстким белком. Однако в реальности при связывании белка с лигандом некоторые его атомы могут прийти в движение, и число подвижных атомов увеличивает количество степеней свободы системы, а значит и размерность энергетической поверхности. Например, если в комплексе белок-лиганд у лиганда 7 торсионных кручений, а у белка — 48 подвижных атомов, то у системы будет $6 + 7 + 48 \times 3 = 157$ степеней свободы. Столь многомерные задачи не удаётся решать такими методами, как FLM, в силу ограниченности вычислительных ресурсов. В то же время предложенный в диссертации метод глобальной оптимизации на основе TT-разложений остаётся эффективным даже при высоких размерностях [10; 11], поскольку сложность метода крестовой интерполяции TT-Cross зависит от числа размерностей тензора линейно.

Результаты работы программы SOL-P представлены в Таблице 16. Рассмотрены 3 комплекса белок-лиганд с вариативным числом подвижных атомов. Для первого комплекса SOL-P находит близкое к нативному положение даже при малоранговом приближении тензора. В случае со вторым комплексом мы наблюдаем, что при увеличении числа степеней свободы системы SOL-P удаётся

Таблица 15 — Сравнение работы программ докинга FLM и SOL-P на 30 комплексах белок-лиганд из Protein Data Bank с жёстким белком.

PDB ID	Разница между минимальными значениями $\mathcal{E}_x$, найденными разными программами докинга [ккал/моль]				Значение i_{NN}			
	FLM	SOL-P, $r = 16$	SOL-P, $r = 32$	SOL-P, $r = 64$	FLM	SOL-P, $r = 16$	SOL-P, $r = 32$	SOL-P, $r = 64$
1B9V	0	2.33	1.27	1.21	∞	308	242	872
1BR5	0	3.69	2.73	2.73	121	25	24	149
1C5Y	0	0	0	0	1	1	1	1
1DWC	0	36.9	26.1	5.73	629	∞	7	18
1EFY	0	3.55	0	0	88	27	∞	127
1F5L	0	0	0	1.47	1	1	1	1
1HPV	0	7.11	6.47	2.91	1	1	1	1
1I7Z	0	0	0	0	1	1	1	1
1J01	0	0	10.1	8.34	1	1	∞	∞
1K1J	0	0	0.06	0	1	1	1	1
1LQD	0	0	0	0	1	1	1	1
1LZG	0	19.5	9.95	13.6	∞	∞	∞	∞
1MQ6	0	7.01	0.97	2.70	3	1	1	1
1O3P	0	1.51	0.35	0	13	6	7	10
1PPC	0	0	0	0	1	1	1	1
1SQO	0	0	0	0	1	1	1	1
1TOM	0	2.20	2.20	2.20	∞	∞	284	471
1VJ9	0	0	0	0.17	1	1	1	1
1VJA	0	1.89	0	1.89	4	∞	5	5
2P94	0	0.92	3.20	1.90	2	1	∞	∞
2PAX	0	0	0	0	1	1	1	1
3CEN	0	12.9	2.43	2.43	1	∞	∞	∞
3KIV	0	0	0	0	1	1	1	1
3PAX	0	0	0	0	1	1	1	1
4FSW	0	0	0	0	7	3	4	6
4FT0	0.09	26.5	0	18.2	12	10	4	∞
4FT9	0	14.5	0	0	27	∞	∞	31
4FTA	0	15.9	15.9	6.30	∞	∞	35	73
4FV5	0	15.9	17.2	0	120	11	12	∞

Таблица 16 — Результаты работы программы докинга SOL-P на 3 комплексах белок-лиганд из Protein Data Bank с подвижными атомами.

PDB ID (число кручений)	Число подвижных атомов	Значение i_{NON}		
		$n = 2^{16}$, $r = 4$	$n = 2^{12}$, $r = 8$	$n = 2^{12}$, $r = 16$
1SQO (3)	0, 6, 15, 27, 35	1	1	1
3CEN (7)	0, 6 13 26 48	∞ 1 1 2	∞ ∞ 1 2	∞ 17 2 1
4FT9 (5)	0 6 13 25 42	16 17 17 15 ∞	24 21 18 15 ∞	29 21 19 15 ∞

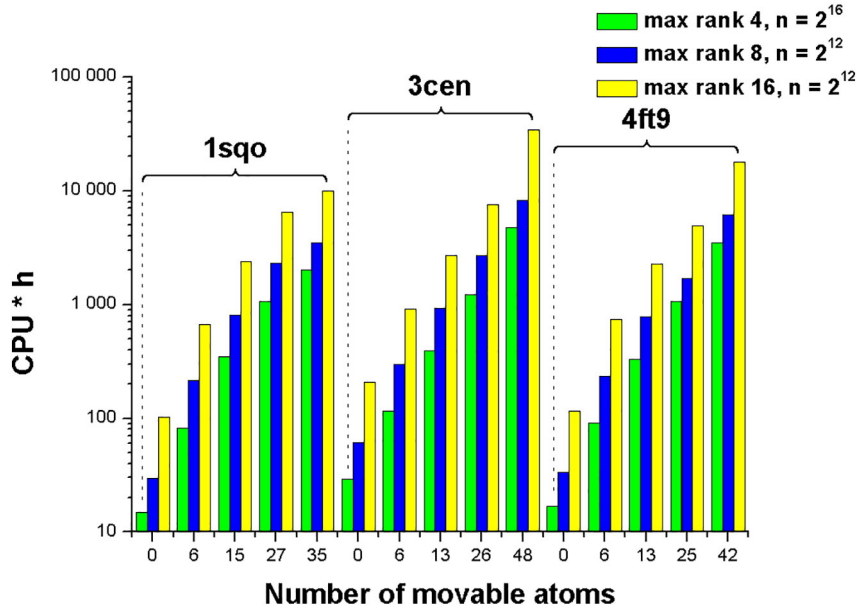


Рисунок 4.1 — Вычислительная сложность (в процессорочасах) в зависимости от числа подвижных атомов белка.

обнаружить минимумы энергии, расположенные рядом с оптимизированным нативным положением. При этом результаты вновь почти совпадают для разных TT-рангов. Наконец результаты, полученные для третьего комплекса, являются маркером того, что использованное силовое поле MMFF94 в вакууме без модификаций и поправок не может служить для описания связывания лиганда с соответствующим белком.

На рисунке 4.1 указаны вычислительные мощности, затраченные для расчёта таблицы 16, и виден характер роста числа ядрочасов в зависимости от числа подвижных атомов белка. Показательно, что в то время, как программе FLM потребовалось порядка 25000 ядрочасов для расчёта докинга одного комплекса с жёстким белком, программе SOL-P потребовалось от 10000 до 45000 ядрочасов для задачи с подвижными атомами белка (35 и 48 атомов соответственно). Расчёт докинга для комплексов белок-лиганд с настолько большим числом степеней свободы (157) был произведён впервые.

Глава 5. Применение алгоритма глобальной оптимизации к задаче оценки параметров моделей ВИЧ-инфекции

5.1 Постановка задачи

В задачах современной иммунологии широко применяется математическое моделирование. Использование математических моделей позволяет лучше понимать сложные взаимосвязи между компонентами систем, проводить численные эксперименты, формулировать гипотезы о причинах тех или иных явлений, выявлять возможные мишени для разработки препаратов, прогнозировать эффект применения различных режимов терапии. Наиболее общеупотребимыми являются модели, которые представляют собой системы нелинейных обыкновенных дифференциальных уравнений (ОДУ). При этом, как правило, приходится иметь дело не с отдельно взятой системой ОДУ, а с их параметрическим семейством. Даже в рамках описания на одном уровне одно и то же явление может быть представлено с помощью моделей разной степени детализации, а значит и с использованием различного числа параметров.

При разработке математической модели иммунологического процесса специалисты из предметной области сталкиваются с тем, что далеко не все параметры модели могут быть измерены экспериментально. Поэтому возникает актуальная для приложений в медицине задача идентификации модели, то есть оценки её параметров на основе данных клинических исследований. Поскольку же рассматриваемая система может обладать высокой размерностью, для идентификации модели требуются эффективные методы оптимизации.

В общем случае модель на основе ОДУ может быть записана как задача Коши

$$\begin{aligned} \frac{d}{dt}\mathbf{y} &= F(\mathbf{y}(t), \mathbf{p}), \quad 0 < t \leqslant T, \\ \mathbf{y}(0) &= \mathbf{y}_0 \in \mathbb{R}^n, \quad \mathbf{p} \in \mathbb{R}^m, \\ \mathbf{z}(t) &= h(\mathbf{y}(t)) \in \mathbb{R}^k, \end{aligned} \tag{5.1}$$

относительно n-мерной вектор-функции $\mathbf{y}(t)$ с m-мерным вектором параметров $\mathbf{p}$. Функция $h : \mathbb{R}^n \to \mathbb{R}^k$ описывает проводимые измерения. Под клиническими данными в таком случае понимаются L измерений временного ряда $\{t_l, \hat{\mathbf{z}}_l\}_{l=1}^{L}$, и

задача идентификации ставится как задача минимизации функционала

$$\Phi(\mathbf{p}) = \sum_{l=1}^{L} \varphi(\hat{\mathbf{z}}_l, \mathbf{z}(t_l; \mathbf{p})) \to \min_{\mathbf{p} \in \Omega \subseteq \mathbb{R}^m}, \tag{5.2}$$

где конкретный вид функции $\varphi : \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}_+$ зависит от распределения шума в измерениях (ищется оценка максимального правдоподобия на вектор параметров), а Ω задаёт область физических значений в пространстве параметров. Если в качестве Ω взять параллелепипед и ввести в нем равномерную сетку по каждому из направлений, придём к задаче поиска минимального элемента в многомерном неотрицательном тензоре, для решения которой может быть применён разработанный в настоящей диссертации метод на основе ТТ-разложений.

5.2 Модели иммунного ответа на ВИЧ-инфекцию

ВИЧ-инфекция — это хроническое заболевание, вызываемое вирусом иммунодефицита человека и характеризующееся длительным течением, которое завершается фазой СПИД [38]. Изучение всех аспектов ВИЧ-инфекции — даже в рамках моделирования на уровне популяций клеток — приводит к большому разнообразию математических моделей. Рассмотрим четыре из них (полные формулировки приведены в приложении).

Первая модель [13] основана на модели инфекционного заболевания Марчука–Петрова [39] и состоит из $n_1 = 19$ уравнений, которые описывают неинфицированные неспецифические клетки и антитела (всего 8 переменных), инфицированные неспецифические клетки (3 переменные), вирусные частицы (1 переменная), общее количество погибших клеток (1 переменная), неинфицированные ВИЧ-специфичные клетки (4 переменных), инфицированные ВИЧ-специфичные клетки (2 переменные). Описание данной модели включает 51 параметр, из которых $m_1 = 32$ требуется оценить.

Вторая рассматриваемая нами модель [14; 15] учитывает, что процесс активации и деления клеток при развитии иммунного ответа не является моментальным, и представляет собой аналогичную систему из $n_2 = 19$ ОДУ с запаздываниями. Величины запаздываний (5 штук) требует оценки, поэтому во второй модели всего 56 параметров, и $m_2 = 37$ должны быть оценены.

Третья модель [40] описывает динамику так называемых регуляторных T-лимфоцитов. Она состоит из $n_3 = 7$ уравнений и имеет $m_3 = 18$ параметров для оценивания. Четвёртая модель [15] — вариант третьей модели с одним запаздыванием, то есть с $n_4 = 7$ уравнениями и $m_4 = 19$ параметрами.

5.3 Описание измерений

Для первых двух моделей наблюдению подлежат $k = 3$ агрегированные величины, которые могут быть измерены в течение острой фазы ВИЧ-инфекции:

$$\begin{aligned} z_1(t) &= H_B(t) + H_E(t) + H_{B_{sp}}(t) + H_{E_{sp}}(t) \\ &\quad + H^*_B(t) + H^*_E(t) + H^*_{B_{sp}}(t) + H^*_{E_{sp}}(t), \\ z_2(t) &= E(t) + E_{sp}(t), \\ z_3(t) &= V(t). \end{aligned} \tag{5.3}$$

Для третьей и четвёртой модели таких величин две:

$$\begin{aligned} z_1(t) &= T(t), \\ z_2(t) &= V(t). \end{aligned} \tag{5.4}$$

Будем также предполагать, что погрешности измерений распределены логнормально. Тогда методу максимального правдоподобия будет отвечать функция

$$\varphi(\mathbf{z}, \hat{\mathbf{z}}) = \sum_{i=1}^{k} (\log z_i - \log \hat{z}_i)^2.$$

5.4 Численные эксперименты

Для проведения численных экспериментов были взяты конкретные числовые значения измерений из [41; 42], содержащие $L = 17$ измерений в разные моменты времени. При каждом вычислении функционала $\Phi(\mathbf{p})$ решается задача Коши (5.1) на отрезке $[0, t_L]$. Для численного решения системы ОДУ из первой модели применялся метод дифференцирования назад из библиотеки Sundials,

описанной в работе [43]; для решения системы ОДУ с запаздываниями из второй модели — аналогичный метод, реализованный в работе [44]. В обоих случаях интегрирование системы выполнялось с заданной точностью ε.

Минимизация функционала (5.2) проводилась с помощью 8 различных методов глобальной оптимизации. Среди них метод, разработанный в данной диссертации, и 7 методов из библиотеки NLopt [45]:

TT — метод на основе TT-разложения тензоров (разработан в диссертации),

CRS2 — метод контролируемого случайного поиска с локальной оптимизацией [46],

rMLSL — многоуровневый метод разбиения области на односвязные кластеры с псевдослучайными стартовыми точками [47],

qrMLSL — многоуровневый метод разбиения области на односвязные кластеры с квазислучайными стартовыми точками [47],

rMLSL + SBPLX — rMLSL с локальной оптимизацией методом Subplex [48],

qrMLSL + SBPLX — qrMLSL с локальной оптимизацией методом Subplex [48],

ISRES — улучшенная эволюционная стратегия стохастического ранжирования [49],

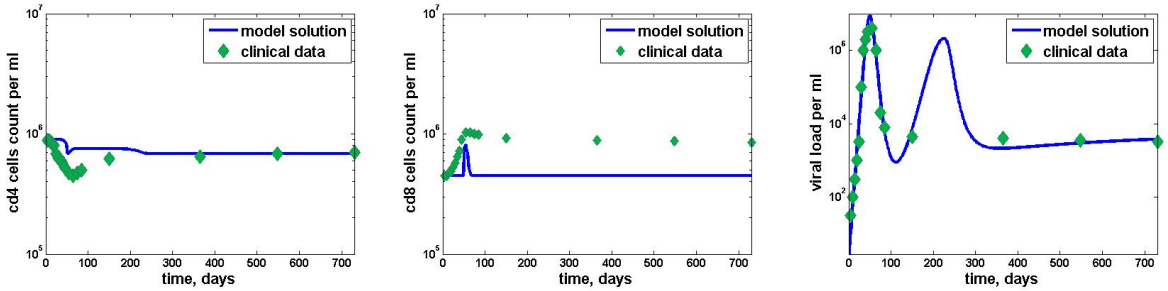
ESCH — эволюционный алгоритм [50].

Для всех методов было установлено ограничение на количество вычислений функционала $\Phi(\mathbf{p})$: 10^6 вычислений при интегрировании с точностью $\varepsilon = 10^{-6}$ или $\varepsilon = 10^{-4}$ и 10^7 вычислений при точности интегрирования $\varepsilon = 10^{-2}$. Также в ряде экспериментов проводилась локальная дооптимизация значений, найденных методами глобальной оптимизации. Для этого использовался симплекс-метод Нелдера–Мида [51].

Сравнительный анализ методов глобальной оптимизации представлен в Таблице 17. Он показывает, что предложенный в диссертации метод во всех 6 экспериментах показывает один из лучших результатов. На Рисунке 5.1 представлены графики измерений $\mathbf{z}(t)$, полученных для первой модели в результате оценивания её параметров по клиническим данным.

Метод	Мод. 1	Мод. 1	Мод. 2	Мод. 2	Мод. 3	Мод. 4
ε	10^{-4}	10^{-4}	10^{-4}	10^{-2}	10^{-6}	10^{-6}
Дооптимизация	Нет	Да	Да	Да	Да	Да
TT	2.28	2.28	2.56	2.15	1.75	1.86
CRS2	2.47	2.47	5.06	2.47	19.76	18.47
rMLSL	2.64	2.50	3.20	2.63	3.69	4.44
qrMLSL	4.65	2.51	2.89	2.72	2.14	4.49
rMLSL+SBPLX	2.13	2.13	2.82	2.70	1.88	2.06
qrMLSL+SBPLX	2.22	2.22	3.00	2.46	1.91	1.93
ISRES	2.34	2.34	2.72	2.16	14.93	1.59
ESCH	3.49	3.49	4.38	2.56	3.41	2.05

Таблица 17 — Значения функционала $\Phi(\mathbf{p})$, полученные с помощью методов глобальной оптимизации для задачи оценки параметров в модели ВИЧ-инфекции.



а) $z_1(t)$ б) $z_2(t)$ в) $z_3(t)$

Рисунок 5.1 — Клинические данные и измерения, полученные при решении ОДУ с оптимизированными параметрами для первой модели.

Глава 6. Применение алгоритма глобальной оптимизации для построения антенных решеток в автомобильных радарах

6.1 Постановка задачи

Задача распознавания объектов является основной математической задачей в области автомобильных радаров, эффективное и надежное решение которой необходимо для разработки беспилотных автомобилей. Физически автомобильный радар представляет собой набор антенн, способных испускать и/или принимать волны с определённой частотой; при этом геометрические расстояния между положениями этих антенн в пространстве сравнимы с величиной длины волны.

Непосредственно “распознавание” объектов производится следующим образом. В один и тот же момент времени весь набор отправляющих антенн одновременно генерирует волну одной и той же частоты, а затем каждая принимающая антенна регистрирует отражённую от объекта волну, отправленную каждой отправляющей антенной. Полученные данные дают информацию о временах задержки $t_{tx,rx}$ между временем отправления сигнала с отправляющей антенны с номером rx и временем получения сигнала на принимающей антенне с номером tx. На основании величин таких задержек, соответствующих различным комбинациям антенн, проводится оценка углов падения лучей от объекта на автомобильный радар, т.е, угловых координат объекта в системе координат, связанных с автомобильным радаром.

Рассмотрим одномерный случай и точечный объект. Тогда простейшая схема автомобильного радара имеет вид, описанный в [52].

На приведённой на рисунке 6.1 схеме красный цвет соответствует принимающим антеннам и соответствующим лучам из объекта до антенн, а зелёный - отправляющим антеннам и соответствующим лучам из антенн до объекта. Как правило, используется линейное приближение, предполагающее, что расстояние от автомобильного радара до объекта значительно больше расстояний между антеннами радара. В таком случае можно предполагать входящие и исходящие лучи параллельными, откуда следует, что величина $t_{tx,rx} - t_{1,1}$ пропорциональна $(\Delta x_{tx} + \Delta x_{rx})\cos(\alpha)$, где α - угол возвышения объекта.

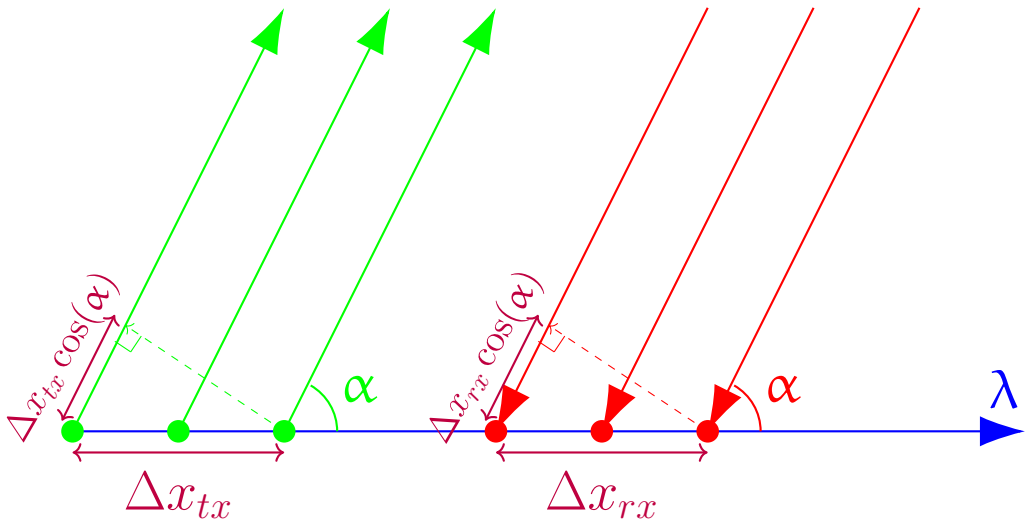


Рисунок 6.1 — Схема одномерного автомобильного радара

Из этих соображений формируется модель сигнала, соответствующего отправляющей антенне tx и принимающей антенне rx

$$s_{tx,rx} = ce^{i\frac{2\pi}{\lambda}(\Delta x_{tx}+\Delta x_{rx})\cos(\alpha)},$$

где c - комплекснозначная константа (соответствующая $s_{0,0}$). Такой вектор $\vec{s}$ называют вектором фазовых задержек. В общем трёхмерном случае, можно получить аналогичное соотношение [52]

$$\vec{k}_{tx,rx} := \frac{2\pi}{\lambda}(\vec{a}_{tx} + \vec{b}_{rx}), k \in \mathbb{R}^3;$$
$$s_{tx,rx} = ce^{i(\vec{k}_{tx,rx},\vec{d})}.$$

В этом соотношении $\vec{d}$ - трёхмерный вещественный вектор, заданный началом в точке геометрического положения автомобильного радара и концом в точке, соответствующей положению точечного объекта. При этом вид вектора $\vec{s}$ не зависит от масштабирования $\vec{d}$ (константу можно вынести в c), и можно предполагать $||d||_2 = 1$, $d = (\cos\varphi\cos\psi, \sin\varphi\cos\psi, \sin\psi)$, где φ, ψ - неизвестные углы, соответствующие полярным координатам вектора направления $\vec{d}$.

С помощью введённых определений вектора $\vec{s}$ фазовых задержек задача распознавания объектов ставится следующим образом. Предполагается, что известна зашумленная версия вектора $\tilde{\vec{s}} = \vec{s} + \vec{n}$. Требуется с наименьшей ошибкой найти соответствующие исходному $\vec{s}$ угловые параметры (α в одномерном случае, φ, ψ в двумерном случае).

Классическим методом решения такой задачи является алгоритм MUSIC [53], предполагающий введение сетки на пространстве угловых параметров, построение соответствующего каждому элементу сетки вектора $\vec{s}_{grid}$, и поиска максимума скалярного произведения $(\vec{s}_{grid}, \tilde{\vec{s}})$. Угловые параметры вектора $\vec{s}_{grid}$, наиболее сонаправленного поступившему на вход зашумленному $\tilde{\vec{s}}$, возвращаются как результат алгоритма.

Отсюда следует, что желаемым свойством векторов фазовых задержек является “непохожесть” векторов $\vec{s}(\varphi_1, \psi_1), \vec{s}(\varphi_2, \psi_2)$, соответствующих различным углам направления. Кроме того, на практике интересен случай большого числа объектов, т.е. случай, где измерения имеют вид

$$\tilde{\vec{s}} = \sum_k \vec{s}(\varphi_k, \psi_k) + \vec{n},$$

где $k > 1, k \ll n_{tx} \times n_{rx}$.

Из последнего выражения следует нежелательность не только близости пар векторов задержек, но и близости любых k таких векторов к линейно зависимым. Следовательно, можно поставить задачу оптимизации расположения антенн (т.е. выбора $\vec{a}_{tx}, \vec{b}_{rx} \in \mathbb{R}^3$). Из рассмотренных выше соображений предполагаемый функционал для оценки качества такого расположения должен измерять степень линейной независимости векторов фазовых задержек, соответствующих различным значениям параметров.

Формально, предполагается на пространстве параметров ввести двумерную сетку $\{\varphi_1, \varphi_2 \dots \varphi_M\} \times \{\psi_1, \psi_2 \dots \psi_N\}$ для достаточно больших M, N и записать соответствующие векторы фазовых задержек в виде матрицы

$$Z = \left[\vec{s}(\varphi_1, \psi_1) \;\; \vec{s}(\varphi_2, \psi_1) \;\; \dots \;\; \vec{s}(\varphi_N, \psi_1) \;\; \vec{s}(\varphi_2, \psi_1) \;\; \dots \;\; \vec{s}(\varphi_N, \psi_M)\right].$$

Тогда в качестве функционала оптимизации позиций антенн $\vec{a}_{tx}, \vec{b}_{rx}$ предполагается использование числа обусловленности матрицы Z:

$$f(\vec{a}_1 \dots \vec{a}_{tx}, \vec{b}_1 \dots \vec{b}_{rx}) = \frac{\sigma_{tx,rx}(Z)}{\sigma_1(Z)} \to \max.$$

Здесь предполагается, что $MN \gg n_{tx} n_{rx}$, таким образом, функционал оценивает линейную независимость строк матрицы Z.

В качестве альтернативного способа оценки качества расстановки антенн можно использовать функционал, который показывает, насколько отличаются в худшем случае от вектора фазовых задержек для некоторых фиксированных углов

(в рассматриваемом случае $\varphi = \psi = 0$) вектора фазовых задержек для всех улов вне некоторой окрестности зафиксированных углов:

$$g(\vec{a}_1 \dots \vec{a}_{tx}, \vec{b}_1 \dots \vec{b}_{rx}) = \max_{|\varphi| > \varphi_0 \text{или} |\psi| > \psi_0} |e^T \vec{s}(\varphi, \psi),$$

где e – вектор, все компоненты которого равны 1, то есть вектор фазовых задержек для $\varphi = \psi = 0$.

Подход, основанный на минимизации функционала g, позволяет контролировать баланс между точностью и однозначностью оценки углов, причём для каждого из углов в отдельности.

Для оптимизации данных функционалов был использован предложенный в данной работе метод глобальной оптимизации на основе TT-разложения.

6.2 Численные эксперименты

Оптимизация функционала f производилась для двух конфигураций антенн:

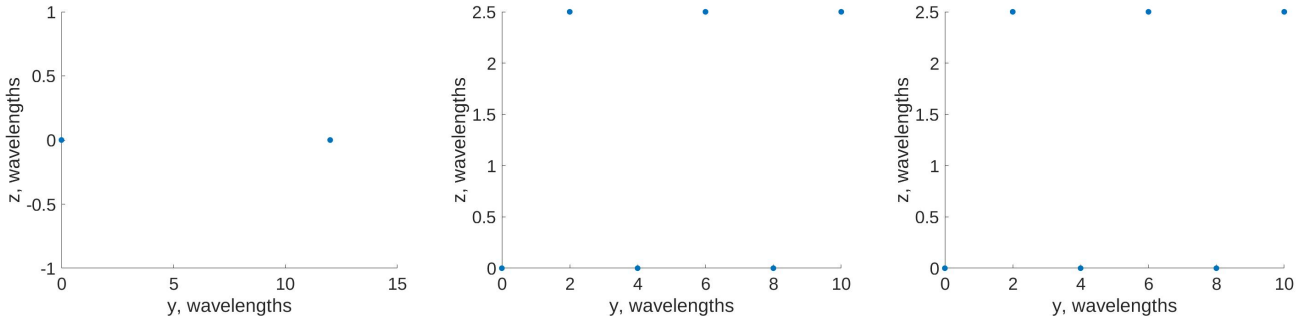
1. 3 передающие и 4 принимающие антенны (далее будем её называть 3×4).
2. 4 одинаковых группы по 3 передающих и 4 одинаковых группы по 4 принимающих антенны (далее будем её называть $4 \times (3 \times 4)$).

Результаты оптимизации приведены в таблице 18, расположение антенн и диаграммы направленности для референсного радара (Continental ARS-408, который также имеет 12 виртуальных антенн) приведены на рис. 6.2 и 6.3, расположение антенн и диаграммы направленности для оптимизированной конфигурации 3×4 приведены на рис. 6.4 и 6.5, расположение антенн и диаграммы направленности для оптимизированной конфигурации $4 \times (3 \times 4)$ приведены на рис. 6.6 и 6.7.

Заметим интересную деталь – в конфигурации $4 \times (3 \times 4)$ группы антенн (по 3 передающих и по 4 принимающих) оказываются смещёнными друг относительно друга на горизонтальные сдвиги, хотя исходно оптимизация допускала произвольные сдвиги. Также заметим, что, исходя из диаграммы направленности, оптимизированная конфигурация 3×4 несколько теряет точность по азимуту относительно конфигурации антенн радара Continental ARS-408, однако обладает

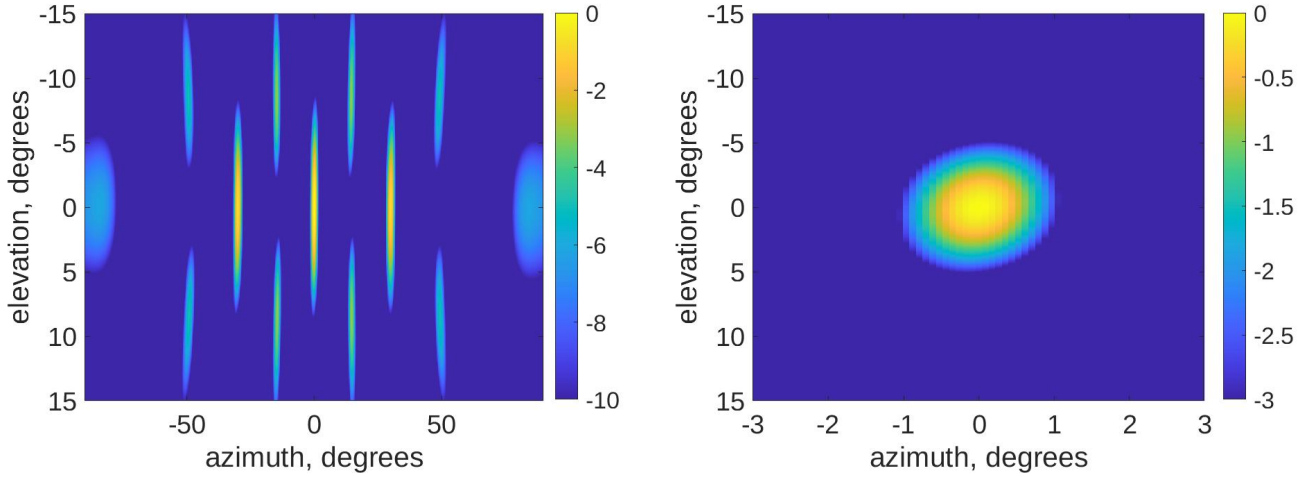
Таблица 18 — Значение функционала f для различных конфигураций

Конфигурация	Значение f
ARS-408	0.534
Оптимизированная по f 3×4	0.996
Оптимизированная по f $4 \times (3 \times 4)$	0.912



а) Передающие б) Принимающие в) Виртуальные

Рисунок 6.2 — Конфигурация антенн радара Continental ARS-408



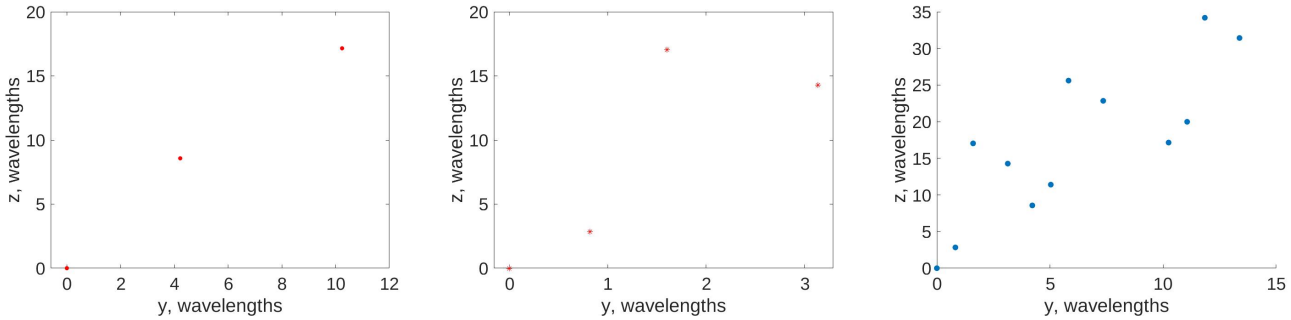
а) Диаграмма в целевом секторе б) Центральная часть

Рисунок 6.3 — Диаграмма направленности радара Continental ARS-408

более высокой точностью для возвышения и значительно лучшей однозначностью определения углов.

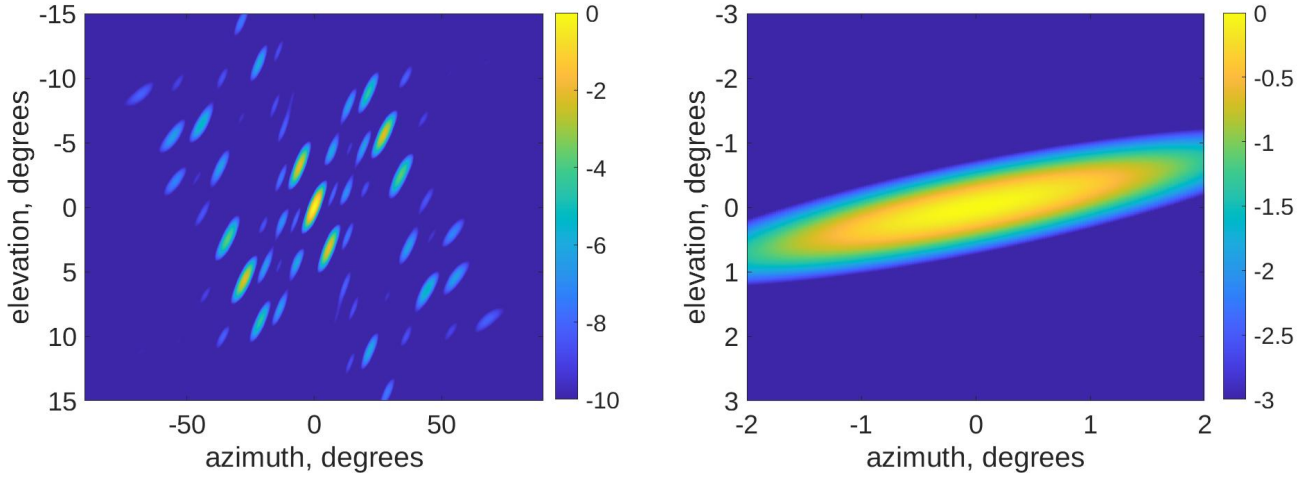
Оптимизация функционала g производилась для двух конфигураций антенн:

1. 3 передающие и 4 принимающие антенны (далее будем её называть 3×4).
2. 4 группы по 3 передающих и 4 принимающих антенны, в каждой из групп антенны должны помещаться в квадрат со стороной 5 длин волн (далее будем её называть 12×16).



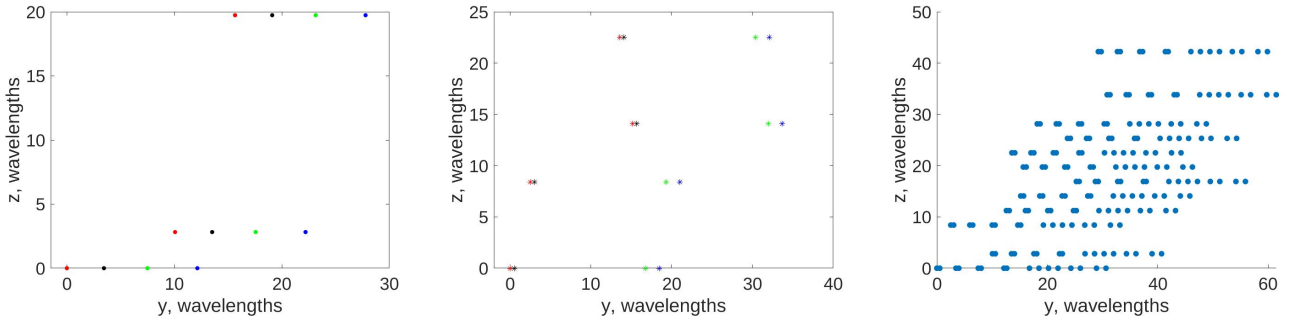
а) Передающие б) Принимающие в) Виртуальные

Рисунок 6.4 — Оптимизированная по f конфигурация антенн 3×4



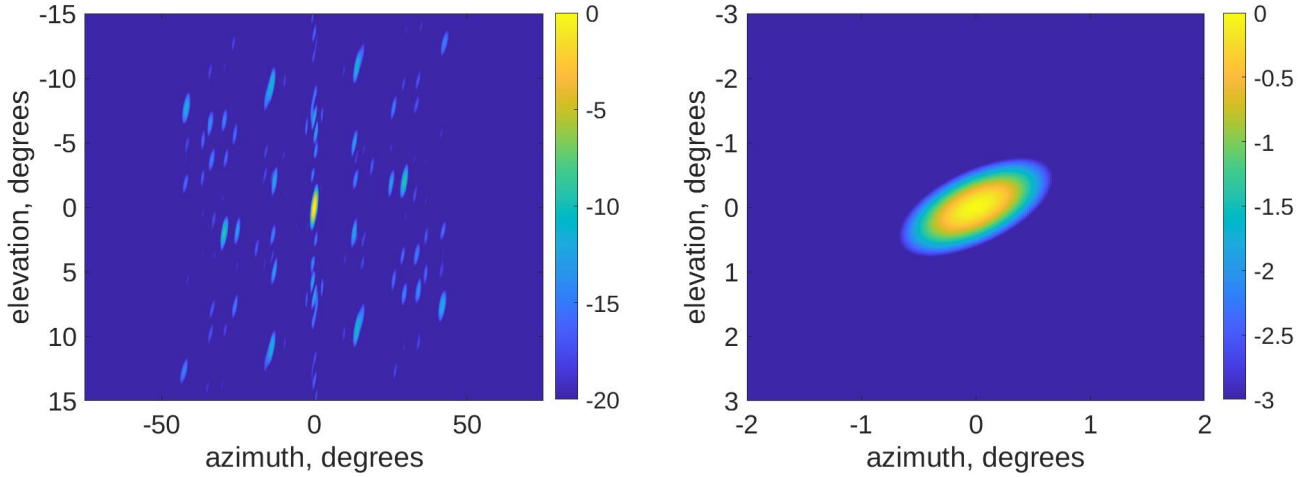
а) Диаграмма в целевом секторе б) Центральная часть

Рисунок 6.5 — Диаграмма направленности оптимизированной по f конфигурации антенн 3×4



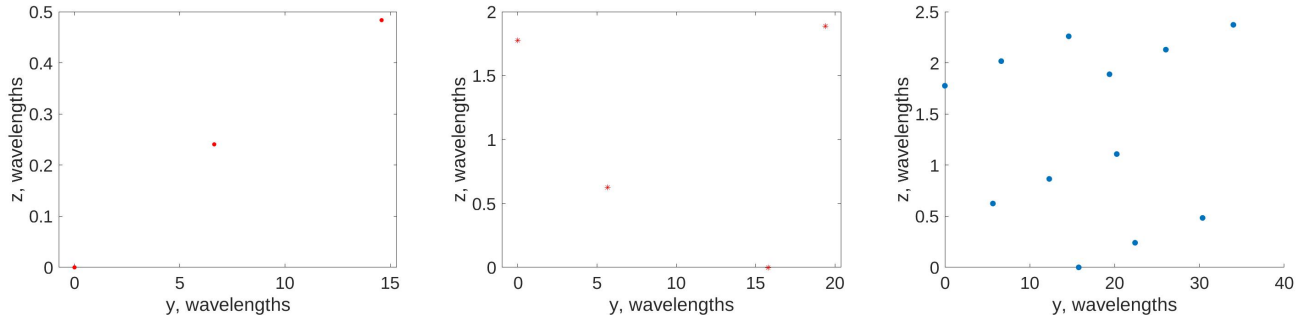
а) Передающие б) Принимающие в) Виртуальные

Рисунок 6.6 — Оптимизированная по f конфигурация антенн $4 \times (3 \times 4)$



а) Диаграмма в целевом секторе б) Центральная часть

Рисунок 6.7 — Диаграмма направленности оптимизированной по f конфигурации антенн $4 \times (3 \times 4)$

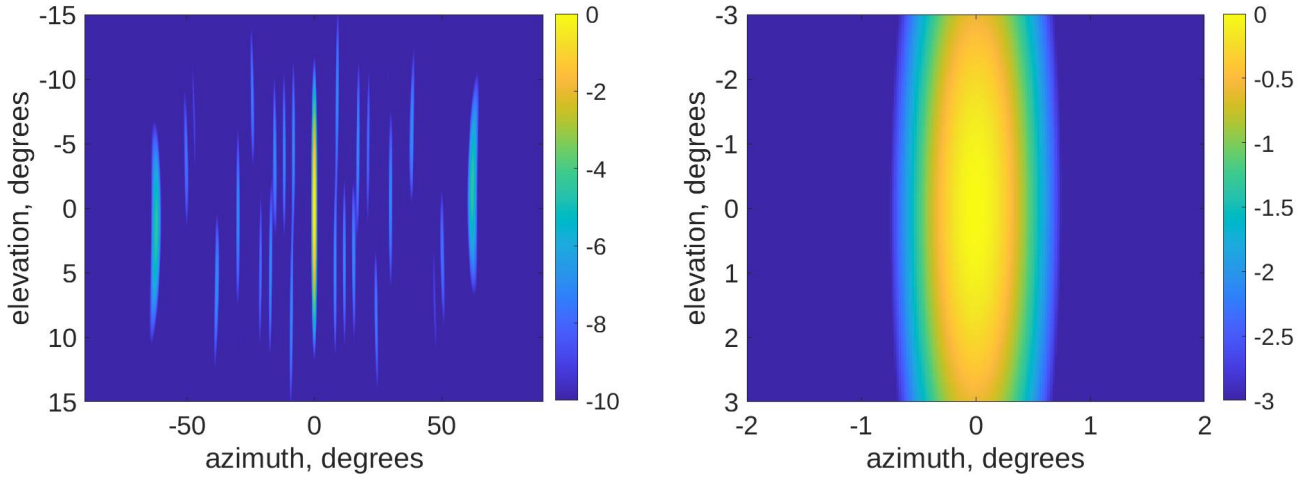


а) Передающие б) Принимающие в) Виртуальные

Рисунок 6.8 — Оптимизированная по g конфигурация антенн 3×4

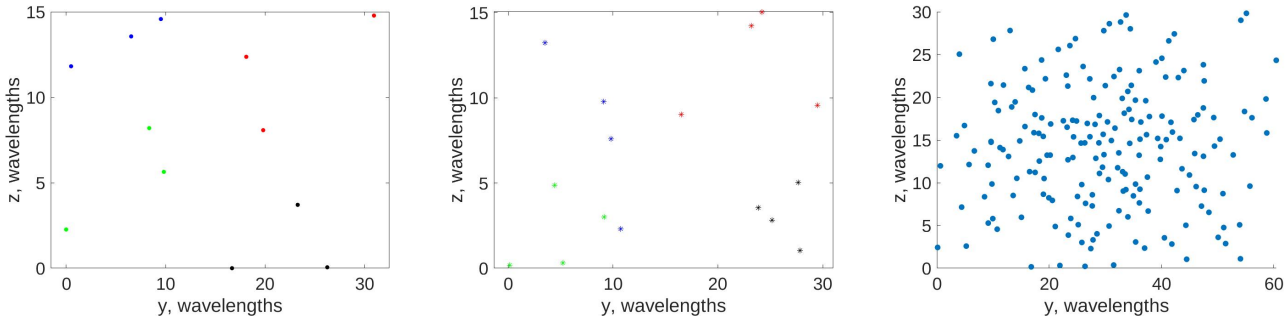
Расположение антенн и диаграммы направленности для оптимизированной по g конфигурации 3×4 приведены на рис. 6.8 и 6.9, расположение антенн и диаграммы направленности для оптимизированной по g конфигурации 12×16 приведены на рис. 6.10 и 6.11.

Отметим, что использование функционала g позволяет (за счёт выбора φ_0, ψ_0) регулировать точность каждого угла независимо. Так, в оптимизированной конфигурации 3×4 существенно снижена точность определения угла возвышения (который менее важен для автомобильных радаров), но повышена точность определения азимута и существенно улучшена однозначность определения углов.



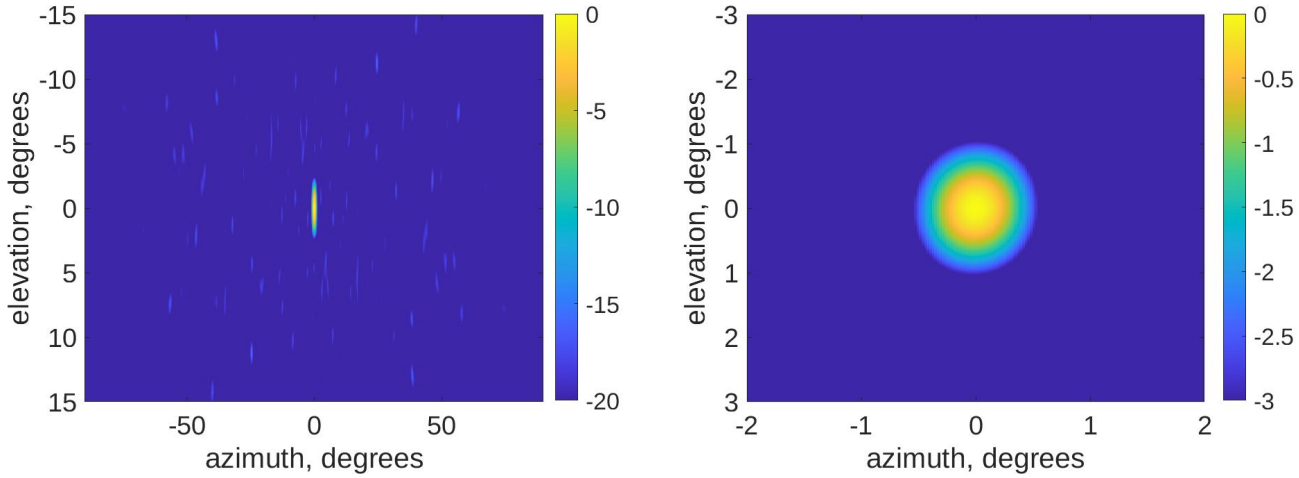
а) Диаграмма в целевом секторе б) Центральная часть

Рисунок 6.9 — Диаграмма направленности оптимизированной по g конфигурации антенн 3×4



а) Передающие б) Принимающие в) Виртуальные

Рисунок 6.10 — Оптимизированная по g конфигурация антенн 12×16



а) Диаграмма в целевом секторе б) Центральная часть

Рисунок 6.11 — Диаграмма направленности оптимизированной по g конфигурации антенн 12×16

Заключение

Основные результаты работы заключаются в следующем.

1. Разработан и реализован метод глобальной оптимизации на основе тензорных разложений.
2. Приведено частичное обоснование метода глобальной оптимизации на основе тензорных разложений.
3. TT-метод глобальной оптимизации применён к ряду прикладных задач, в которых он показал высокую эффективность.
4. Разработаны параллельные алгоритмы крестовой интерполяции.
5. Предложена стратегия адаптивного поиска ранга для TT-крестового метода.
6. Проведены численные эксперименты, подтверждающие высокую надёжность и эффективность реализованных методов.

Список литературы

1. *Udell, M.* Why are big data matrices approximately low rank? / M. Udell, A. Townsend // SIAM Journal on Mathematics of Data Science. — 2019. — Т. 1, № 1. — С. 144—160.
2. *Goreinov, S. A.* Pseudo–skeleton approximations of matrices / S. A. Goreinov, E. E. Tyrtyshnikov, N. L. Zamarashkin // Reports of Russian Academy of Sciences. — 1995. — Т. 342, № 2. — С. 151—152.
3. *Goreinov, S. A.* A theory of pseudo–skeleton approximations / S. A. Goreinov, E. E. Tyrtyshnikov, N. L. Zamarashkin // Linear Algebra Appl. — 1997. — Т. 261. — С. 1—21.
4. *Tyrtyshnikov, E. E.* Incomplete cross approximation in the mosaic–skeleton method / E. E. Tyrtyshnikov // Computing. — 2000. — Т. 64, № 4. — С. 367—380.
5. *Savostyanov, D. V.* Quasioptimality of maximum-volume cross interpolation of tensors / D. V. Savostyanov // Linear Algebra and its Applications. — 2014. — Т. 458. — С. 217—244.
6. *Oseledets, I. V.* TT-cross approximation for multidimensional arrays / I. V. Oseledets, E. E. Tyrtyshnikov // Linear Algebra Appl. — 2010. — Т. 432, № 1. — С. 70—88.
7. *Oseledets, I. V.* Breaking the curse of dimensionality, or how to use SVD in many dimensions / I. V. Oseledets, E. E. Tyrtyshnikov // SIAM J. Sci. Comput. — 2009. — Т. 31, № 5. — С. 3744—3759.
8. TTDock: метод докинга на основе тензорных поездов / Д. А. Желтков [и др.] // Вычислительные методы и программирование. — 2013. — Т. 14. — С. 279—291.
9. Evaluation of the docking algorithm based on tensor train global optimization / I. V. Oferkin [и др.] // Вестник Южно-Уральского государственного университета. Серия: Математическое моделирование и программирование. — 2015. — Т. 8, № 4.
10. Evaluation of the novel algorithm of flexible ligand docking with moveable target-protein atoms / A. V. Sulimov [и др.] // Computational and structural biotechnology journal. — 2017. — Т. 15. — С. 275—285.

11. Tensor train global optimization: Application to docking in the configuration space with a large number of dimensions / A. V. Sulimov [и др.] // Russian Supercomputing Days. — Springer. 2017. — С. 151—167.

12. Docking of oligopeptides / A. Sulimov [и др.] // Russian Chemical Bulletin. — 2019. — Т. 68, № 9. — С. 1780—1786.

13. Tensor based approach to the numerical treatment of the parameter estimation problems in mathematical immunology / V. V. Zheltkova [и др.] // Journal of Inverse and Ill-posed Problems. — 2018. — Т. 26, № 1. — С. 51—66.

14. *Zheltkova, V. V.* Modelling HIV infection: model identification and global sensitivity analysis / V. V. Zheltkova, D. A. Zheltkov, G. A. Bocharov // Matematicheskaya Biologiya i Bioinformatika. — 2019. — Т. 14, № 1. — С. 19—33.

15. Application of the global optimization methods for solving the parameter estimation problem in mathematical immunology / V. V. Zheltkova [и др.] // International Conference on Large-Scale Scientific Computing. — Springer. 2019. — С. 203—209.

16. *Желтков, Д. А.* Увеличение размерности в методе докинга на основе тензорных поездов / Д. А. Желтков, Е. Е. Тыртышников // Вычислительные методы и программирование. — 2013. — Т. 14. — С. 292—294.

17. *Желтков, Д. А.* Параллельная реализация матричного крестового метода / Д. А. Желтков, Е. Е. Тыртышников // Вычислительные методы и программирование. — 2015. — Т. 16, № 3. — С. 369—375.

18. *Zheltkov, D.* Global optimization based on TT-decomposition / D. Zheltkov, E. Tyrtyshnikov // Russian Journal of Numerical Analysis and Mathematical Modelling. — 2020. — Т. 35, № 4. — С. 247—261.

19. Fast and accurate finite-difference method solving multicomponent Smoluchowski coagulation equation with source and sink terms / A. P. Smirnov [и др.] // Procedia Computer Science. — 2016. — Т. 80. — С. 2141—2146.

20. Tensor train versus Monte Carlo for the multicomponent Smoluchowski coagulation equation / S. A. Matveev [и др.] // Journal of Computational Physics. — 2016. — Т. 316. — С. 164—179.

21. *Stefonishin, D. A.* Tensors in modelling multi-particle interactions / D. A. Stefonishin, S. A. Matveev, D. A. Zheltkov // International Conference on Large-Scale Scientific Computing. — Springer. 2019. — С. 173—180.

22. *Zheltkov, D. A.* Global Optimization Algorithms Using Tensor Trains / D. A. Zheltkov, A. Osinsky // International Conference on Large-Scale Scientific Computing. — Springer. 2019. — С. 197—202.

23. Supercomputer docking with a large number of degrees of freedom / A. Sulimov [и др.] // SAR and QSAR in Environmental Research. — 2019. — Т. 30, № 10. — С. 733—749.

24. *Свидетельство о государственной регистрации программы для ЭВМ*. TTDock / Д. А. Желтков [и др.] ; ИВМ РАН. — № 2013613257 ; заявл. 17.06.2013 ; опубл. 17.06.2013, 2013613257 (Рос. Федерация).

25. *Kruskal, J. B.* Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics / J. B. Kruskal // Linear algebra and its applications. — 1977. — Т. 18, № 2. — С. 95—138.

26. *Strassen, V.* The asymptotic spectrum of tensors and the exponent of matrix multiplication / V. Strassen // 27th Annual Symposium on Foundations of Computer Science (sfcs 1986). — IEEE. 1986. — С. 49—54.

27. *Tucker, L. R.* Some mathematical notes on three-mode factor analysis / L. R. Tucker // Psychometrika. — 1966. — Т. 31, № 3. — С. 279—311.

28. *Oseledets, I. V.* Tucker dimensionality reduction of three-dimensional arrays in linear time / I. V. Oseledets, D. Savostianov, E. E. Tyrtyshnikov // SIAM Journal on Matrix Analysis and Applications. — 2008. — Т. 30, № 3. — С. 939—956.

29. How to find a good submatrix : Research Report / S. A. Goreinov [и др.] ; ICM HKBU. — Kowloon Tong, Hong Kong, 2008. — № 08—10. — URL: www.math.hkbu.edu.hk/ICM/pdf/08-10.pdf.

30. *Horn, R. A.* Matrix analysis / R. A. Horn, C. R. Johnson. — Cambridge university press, 2012.

31. *Tyrtyshnikov, E. E.* Tensor approximations of matrices generated by asymptotically smooth functions / E. E. Tyrtyshnikov // Sbornik: Mathematics. — 2003. — Т. 194, № 6. — С. 941.

32. *Oseledets, I. V.* Approximation of 2^d\times2^d matrices using tensor decomposition / I. V. Oseledets // SIAM Journal on Matrix Analysis and Applications. — 2010. — Т. 31, № 4. — С. 2130—2145.

33. *Khoromskij, B. N.* QTT approximation of elliptic solution operators in higher dimensions / B. N. Khoromskij, I. V. Oseledets. — 2011.

34. *Садовничий, В.* Суперкомпьютерные технологии в медицине / В. Садовничий, В. Сулимов // Суперкомпьютерные технологии в науке, образовании и промышленности/Под ред. ВА Садовничего, ГИ Савина, Вл. В. Воеводина. М: Изд-во Моск. ун-та. — 2009. — С. 16—23.

35. *Gupta, M.* Docking techniques in pharmacology: How much promising? / M. Gupta, R. Sharma, A. Kumar // Computational biology and chemistry. — 2018. — Т. 76. — С. 210—217.

36. The protein data bank / H. M. Berman [и др.] // Nucleic acids research. — 2000. — Т. 28, № 1. — С. 235—242.

37. *Halgren, T. A.* Merck molecular force field. I. Basis, form, scope, parameterization, and performance of MMFF94 / T. A. Halgren // Journal of computational chemistry. — 1996. — Т. 17, № 5/6. — С. 490—519.

38. Human immunodeficiency virus infection: from biological observations to mechanistic mathematical modelling / G. Bocharov [и др.] // Mathematical Modelling of Natural Phenomena. — 2012. — Т. 7, № 5. — С. 78—104.

39. *Marchuk, G. I.* Mathematical modelling of immune response in infectious diseases. Т. 395 / G. I. Marchuk. — Springer Science & Business Media, 1997.

40. Modeling adaptive regulatory T-cell dynamics during early HIV infection / M. Simonov [и др.] // PloS one. — 2012. — Т. 7, № 4. — e33924.

41. Patterns of viral dynamics during primary human immunodeficiency virus type 1 infection / G. R. Kaufmann [и др.] // The Journal of infectious diseases. — 1998. — Т. 178, № 6. — С. 1812—1815.

42. *Munier, M.* Acutely dysregulated, chronically disabled by the enemy within: T-cell responses to HIV-1 infection / M. Munier, A. Kelleher // Immunology and cell biology. — 2007. — Т. 85, № 1. — С. 6—15.

43. SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers / A. C. Hindmarsh [и др.] // ACM Transactions on Mathematical Software (TOMS). — 2005. — Т. 31, № 3. — С. 363—396.

44. *Бочаров, Г.* Численное решение дифференциальных уравнений с запаздывающим аргументом на основе линейных многошаговых методов / Г. Бочаров, А. Романюха // Аппроксимация, сходимость и устойчивость. Препринт ОВМ АН СССР. — 1986. — Т. 116.

45. *Johnson, S. G.* The NLopt nonlinear-optimization package [Электронный ресурс] / S. G. Johnson. — Дата обращения: 11.10.2021. http://github.com/stevengj/nlopt.

46. *Kaelo, P.* Some variants of the controlled random search algorithm for global optimization / P. Kaelo, M. Ali // Journal of optimization theory and applications. — 2006. — Т. 130, № 2. — С. 253—264.

47. *Kan, A. R.* Stochastic global optimization methods part I: Clustering methods / A. R. Kan, G. T. Timmer // Mathematical programming. — 1987. — Т. 39, № 1. — С. 27—56.

48. *Rowan, T. H.* Functional stability analysis of numerical algorithms : дис. … канд. / Rowan Thomas Harvey. — The University of Texas at Austin, 1990.

49. *Runarsson, T. P.* Search biases in constrained evolutionary optimization / T. P. Runarsson, X. Yao // IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews). — 2005. — Т. 35, № 2. — С. 233—243.

50. *Silva Santos, C. H. da*. Designing novel photonic devices by bio-inspired computing / C. H. da Silva Santos, M. S. Goncalves, H. E. Hernandez-Figueroa // IEEE Photonics Technology Letters. — 2010. — Т. 22, № 15. — С. 1177—1179.

51. *Nelder, J. A.* A simplex method for function minimization / J. A. Nelder, R. Mead // The computer journal. — 1965. — Т. 7, № 4. — С. 308—313.

52. *Wang, S.* Multidimensional Radar Signal Processing Based on Sparse Fourier Transforms : дис. … канд. / Wang Shaogang. — Rutgers The State University of New Jersey, School of Graduate Studies, 2019.

53. *Schmidt, R.* Multiple emitter location and signal parameter estimation / R. Schmidt // IEEE transactions on antennas and propagation. — 1986. — Т. 34, № 3. — С. 276—280.

Приложение А

Описание моделей ВИЧ-инфекции

А.1 Модель 1

Первая из использованных в диссертации моделей ВИЧ-инфекции состоит из 19 ОДУ с 51 параметром. Первая группа уравнений описывает динамику неспецифичных клеток и антител:

$$
\begin{aligned}
\frac{\mathrm{d}D}{\mathrm{d}t} &= \alpha_D(\xi D^0 - D) - \sigma_D V D - \gamma_{DV} V D,\\
\frac{\mathrm{d}D_V}{\mathrm{d}t} &= \gamma_{DV} V D - \alpha_{D_V} D_V - \sigma_D V D_V,\\
\frac{\mathrm{d}H_B}{\mathrm{d}t} &= \alpha_{H_B}(\xi H_B^0 - H_B) - \sigma_{H_B} H_B V - \sigma_{H_B}^D H_B D_V^*,\\
\frac{\mathrm{d}H_E}{\mathrm{d}t} &= \alpha_{H_E}(\xi H_E^0 - H_E) - \sigma_{H_E} H_E V - \sigma_{H_E}^D H_E D_V^*,\\
\frac{\mathrm{d}B}{\mathrm{d}t} &= \alpha_B(B^0 - B),\\
\frac{\mathrm{d}E}{\mathrm{d}t} &= \alpha_E(E^0 - E),\\
\frac{\mathrm{d}P}{\mathrm{d}t} &= b_P^p \xi \rho_P (D_V + D_V^*)(H_{Bsp} + H_{Bsp}^*) B_{sp} + \alpha_P(P_0 - P),\\
\frac{\mathrm{d}F}{\mathrm{d}t} &= \rho_F P - \gamma_{VF} V F - \alpha_F F.
\end{aligned}
$$

Вторая группа — инфицированные неспецифичные клеток:

$$
\begin{aligned}
\frac{\mathrm{d}D_V^*}{\mathrm{d}t} &= \sigma_D V(D + D_V) - b_{D_V E} D_V^* E_{sp} - b_{D_V^*} D_V^*,\\
\frac{\mathrm{d}H_B^*}{\mathrm{d}t} &= \sigma_{H_B} H_B V + \sigma_{H_B}^D H_B D_V^* - b_{H_B E} H_B^* E_{sp} - b_{H_B^*} H_B^*,\\
\frac{\mathrm{d}H_E^*}{\mathrm{d}t} &= \sigma_{H_E} H_E V + \sigma_{H_E}^D H_E D_V^* - b_{H_E E} H_E^* E_{sp} - b_{H_E^*} H_E^*.
\end{aligned}
$$

Третья группа — вирусные частицы и общее количество погибших клеток:

$$\begin{aligned}
\frac{\mathrm{d}V}{\mathrm{d}t} &= \nu_{D_V} D_V^* + \nu_{H_E}(H_E^* + H_{Esp}^*) + \nu_{H_B}(H_B^* + H_{Bsp}^*) + N_{D_V} b_{D_V^*} D_V^* \\
&+ N_{H_E} b_{H_E^*}(H_E^* + H_{Esp}^*) + N_{H_B} b_{H_B^*}(H_B^* + H_{Bsp}^*) - \frac{kV(D_v + D_v^*)}{a(c+V)} \\
&- \gamma_{VH_B} V(H_B + H_{Bsp}) - \gamma_{VH_E} V(H_E + H_{Esp}) - \gamma_{VD} VD - \gamma_{VF} VF - \gamma_{VM} V, \\
\frac{\mathrm{d}m}{\mathrm{d}t} &= b_{D_V E} D_V^* E_{sp} + b_{D_V^*} D_V^* + b_{H_E^*}(H_E^* + H_{Esp}^*) + b_{H_E E}(H_E^* + H_{Esp}^*) E_{sp} \\
&+ b_{H_B^*}(H_B^* + H_{Bsp}^*) + b_{H_B E}(H_B^* + H_{Bsp}^*) E_{sp}.
\end{aligned}$$

Четвёртая группа — неинфицированные ВИЧ-специфичные клетки:

$$\begin{aligned}
\frac{\mathrm{d}H_{Bsp}}{\mathrm{d}t} &= \alpha_{H_B}(\xi\theta H_B^0 - H_{Bsp}) - \sigma_{H_B} H_{Bsp} V - \sigma_{H_B}^D H_{Bsp} D_V^* + 2b_{H_B}(D_V + D_V^*) H_{Bsp} \\
&- b_{H_B}^P (D_V + D_V^*) H_{Bsp} B_{sp}, \\
\frac{\mathrm{d}H_{Esp}}{\mathrm{d}t} &= \alpha_{H_E}(\xi\theta H_E^0 - H_{Esp}) - \sigma_{H_E} H_{Esp} V - \sigma_{H_E}^D H_{Esp} D_V^* + 2b_{H_E}(D_V + D_V^*) H_{Esp} \\
&- b_{H_E}^P (D_V + D_V^*) H_{Esp} E_{sp}, \\
\frac{\mathrm{d}B_{sp}}{\mathrm{d}t} &= \alpha_B(\theta B^0 - B_{sp}) + 2b_B^P (D_V + D_V^*)(H_{Bsp} + H_{Bsp}^*) B_{sp}, \\
\frac{\mathrm{d}E_{sp}}{\mathrm{d}t} &= \alpha_E(\theta E^0 - E_{sp}) + 2b_E^P (D_V + D_V^*)(H_{Esp} + H_{Esp}^*) E_{sp} - b_{ED_V} D_V^* E_{sp} \\
&- b_{EH_E} H_E^* E_{sp} - b_{EH_B} H_B^* E_{sp}.
\end{aligned}$$

Пятая группа — инфицированные ВИЧ-специфичные клетки:

$$\begin{aligned}
\frac{\mathrm{d}H_{Bsp}^*}{\mathrm{d}t} &= \sigma_{H_B} H_{Bsp} V + \sigma_{H_B}^D H_{Bsp} D_V^* + 2b_{H_B}(D_V + D_V^*) H_{Bsp}^* \\
&- b_{H_B}^P (D_V + D_V^*) H_{Bsp}^* B_{sp} - b_{H_B E} H_{Bsp}^* E_{sp} - b_{H_B^*} H_{Bsp}^*, \\
\frac{\mathrm{d}H_{Esp}^*}{\mathrm{d}t} &= \sigma_{H_E} H_{Esp} V + \sigma_{H_E}^D H_{Esp} D_V^* + 2b_{H_E}(D_V + D_V^*) H_{Esp}^* \\
&- b_{H_E}^P (D_V + D_V^*) H_{Esp}^* E_{sp} - b_{H_E E} H_{Esp}^* E_{sp} - b_{H_E^*} H_{Esp}^*.
\end{aligned}$$

Отрицательная обратная связь задаётся с помощью

$$\xi = \frac{(1 - \varepsilon m)}{\varepsilon m + H_E^0 + H_B^0 + D^0}.$$

Таблица 19 — Переменные модели

Обозначение	Переменная	Нач. значение
D	число антигенсодержащих клеток (APC), кл/мл	$5 \cdot 10^5$
D_V	число активированных APC , кл/мл	0
H_E	число CD4 Th1, кл/мл	$4.5 \cdot 10^5$
H_B	число CD4 Th2, кл/мл	$4.5 \cdot 10^5$
B	число B-лимфоцитов, кл/мл	$2.7 \cdot 10^5$
P	число клеток плазмы, кл/мл	10
F	число антител, частиц/мл	0
E	чисто цитотоксических T-лимфоцитов, кл/мл	$4.5 \cdot 10^5$
D_V^*	число инф. активированных APC, кл/мл	0
H_E^*	число инф. CD4 Th1, кл/мл	0
H_B^*	число инф. CD4 Th2, кл/мл	0
V	число вирусных частиц, частиц/мл	100
m	число погибших кл-миш. (в рез. инф.), кл/мл	0
H_{Esp}	число ВИЧ-спец. CD4 Th1, кл/мл	5
H_{Bsp}	число ВИЧ-спец. CD4 Th2, кл/мл	5
B_{sp}	число ВИЧ-спец. B-клеток, кл/мл	3
E_{sp}	число ВИЧ-спец. цитотокс. T-кл, кл/мл	5
H_{Esp}^*	число инф. ВИЧ-специфичных CD4 Th1, кл/мл	0
H_{Bsp}^*	число инф. ВИЧ-специфичных CD4 Th2, кл/мл	0

Таблица 20 — Параметры модели, знаком ”+”отмечены оцениваемые параметры.

Обозначение	Биологическое значение	Ед. измерения	Диапазон
H_E^0	Скорость гомеостаза H_E	кл/мл	$2.43 \cdot 10^5 - 7.3 \cdot 10^5$
H_B^0	Скорость гомеостаза H_B	кл/мл	$2.43 \cdot 10^5 - 7.3 \cdot 10^5$
E^0	Скорость гомеостаза E	кл/мл	$2.39 \cdot 10^5 - 1.1 \cdot 10^6$
B^0	Скорость гомеостаза B	кл/мл	$1.14 \cdot 10^5 - 6.71 \cdot 10^5$
P^0	Скорость гомеостаза P	кл/мл	-
D^0	Скорость гомеостаза D	кл/мл	$5 \cdot 10^5$
$\alpha_{H_E}(+)$	Смертность H_E	1/д	0.8-2
$\alpha_{H_B}(+)$	Смертность H_B	1/д	0.8-2
$\alpha_E(+)$	Смертность E	1/д	0.33-2
$\alpha_B(+)$	Смертность E	1/д	0.05-0.1
$\alpha_P(+)$	Смертность P	1/д	0.33-0.5
$\alpha_F(+)$	Смертность антител	1/д	0.043
$\alpha_D(+)$	Смертность D	1/д	0.01-0.04
$\alpha_{D_V}(+)$	Смертность D_V	1/д	0.3-1.2
$\rho_F(+)$	Скорость F произв. на клетку плазмы	частиц/(кл · д)	0.33-2
θ	Доля ВИЧ-специфичных клеток		10^{-5}
$\nu_{H_E}(+)$	Число вир. частиц, произв. H_E^*	частиц/д	$10 - 10^3$
$\nu_{H_B}(+)$	Число вир. частиц, произв. H_B^*	частиц/д	$10 - 10^3$
$\nu_{D_V}(+)$	Число вир. частиц, произв. D_V^*	частиц/д	$10 - 10^3$
N_{H_E}	Число V, произв. H_E^* в рез. инфекции	частиц/д	$10 - 10^3$
N_{H_B}	Число V, произв. H_B^* в рез. инфекции	частиц/д	$10 - 10^3$
N_{D_V}	Число V, произв. D_V^* в рез. инфекции	частиц/д	$10 - 10^3$
$\sigma_{H_B}(+)$	Частота инф. H_E от св. вируса	1/(частиц· д)	$10^{-10} - 10^{-6}$
$\sigma_{H_E}(+)$	Частота инф. H_B от св. вируса	1/(частиц· д)	$10^{-10} - 10^{-6}$
$\sigma_D(+)$	Частота инф. D_V от св. вируса	1/(частиц· д)	$9 \cdot 10^{-8}$
$\sigma_{H_E}^D(+)$	Частота инф. H_E от D_V^*	1/(кл·д)	$10^{-5} - 10^{-3}$
$\sigma_{H_B}^D(+)$	Частота инф. H_B от св. вируса	1/(кл·д)	$10^{-5} - 10^{-3}$
$b_{H_B^*}(+)$	Смертность H_B^*	1/д	$1 - 2$
$b_{H_E^*}(+)$	Смертность H_E^*	1/д	$1 - 2$
$b_{D_V^*}(+)$	Смертность D_V^*	1/д	$1 - 2$
γ_{VH_E}	нагрузка V в инф. H_E	1/(кл·д)	$8.3 \cdot 10^{-17} - 1.6 \cdot 10^{-16}$
γ_{VH_B}	нагрузка V в инф. H_B	1/(кл·д)	$8.3 \cdot 10^{-17} - 1.6 \cdot 10^{-16}$
γ_{VD}	нагрузка V в инф. D	1/(кл·д)	$8.3 \cdot 10^{-17} - 1.6 \cdot 10^{-16}$
γ_{VM}	Смертность V	1/д	3
$b_{H_B}(+)$	Скорость стимуляции H_B	1/(кл·д)	$6.6 \cdot 10^{-10} - 10^{-7}$
$b_{H_E}(+)$	Скорость стимуляции H_E	1/(кл·д)	$6.6 \cdot 10^{-10} - 10^{-7}$
$b_P^p(+)$	Скорость стимуляции P	1/(кл^2· д)	$1.37 \cdot 10^{-16} - 2.75 \cdot 10^{-15}$
$b_B^p(+)$	Скорость стимуляции B	1/(кл^2· д)	$1.37 \cdot 10^{-16} - 2.75 \cdot 10^{-15}$
$b_E^p(+)$	Скорость стимуляции E	1/(кл^2· д)	$1.37 \cdot 10^{-16} - 1.93 \cdot 10^{-14}$
$b_{H_BE}(+)$	Скорость нейтр. H_B^*	1/(кл·д)	$10^{-7} - 10^{-5}$
$b_{H_EE}(+)$	Скорость нейтр. H_E^*	1/(кл·д)	$10^{-7} - 10^{-5}$
$b_{D_VE}(+)$	Скорость нейтр. D_V^*	1/(кл·д)	$10^{-9} - 10^{-7}$
$\gamma_{D_v}(+)$	Скорость активации D	1/(частиц· д)	
b_{EH_B}	смертность E вызв. нейтр. H_B^*	1/(кл·д)	$10^{-8} - 10^{-6}$
b_{EH_E}	смертность E вызв. нейтр. H_E^*	1/(кл·д)	$10^{-8} - 10^{-6}$
b_{ED_V}	смертность E вызв. нейтр. D_V^*	1/(кл·д)	$10^{-8} - 10^{-6}$
$b_{H_B}^p(+)$	Скорость стимуляции H_B	1/(кл^2· д)	$1.37 \cdot 10^{-16} - 2.75 \cdot 10^{-15}$
$b_{H_E}^p(+)$	Скорость подавл. H_E	1/(кл^2· д)	$1.37 \cdot 10^{-16} - 2.75 \cdot 10^{-15}$
k_m	Макс. число V, погл. D_V	частиц/д	60
c	Константа Михаэлиса-Мертен	частиц/мл	10^6
$\varepsilon(+)$	Константа, опис. отр. обр. связь		

A.2 Модель 2

Вторая модель отличается от первой в пяти уравнениях:

$$
\begin{aligned}
\frac{\mathrm{d}P}{\mathrm{d}t} &= b_P^p \xi \rho_P (D_V(t-\tau_P) + D_V^*(t-\tau_P))(H_{Bsp}(t-\tau_P) \\
&\quad + H_{Bsp}^*(t-\tau_P))B_{sp}(t-\tau_P) + \alpha_P(P_0 - P), \\
\frac{\mathrm{d}H_{Bsp}}{\mathrm{d}t} &= \alpha_{H_B}(\xi\theta H_B^0 - H_{Bsp}) - \sigma_{H_B} H_{Bsp} V - \sigma_{H_B}^D H_{Bsp} D_V^* \\
&\quad + b_{H_B}(2(D_V(t-\tau_{HB}) + D_V^*(t-\tau_{HB}))H_{Bsp}(t-\tau_{HB}) - (D_V + D_V^*)H_{Bsp}) \\
&\quad - b_{H_B}^P (D_V + D_V^*) H_{Bsp} B_{sp}, \\
\frac{\mathrm{d}H_{Esp}}{\mathrm{d}t} &= \alpha_{H_E}(\xi\theta H_E^0 - H_{Esp}) - \sigma_{H_E} H_{Esp} V - \sigma_{H_E}^D H_{Esp} D_V^* \\
&\quad + b_{H_E}(2(D_V(t-\tau_{HE}) + D_V^*(t-\tau_{HE}))H_{Esp}(t-\tau_{HE}) - (D_V + D_V^*)H_{Esp}) \\
&\quad - b_{H_E}^P (D_V + D_V^*) H_{Esp} E_{sp}, \\
\frac{\mathrm{d}B_{sp}}{\mathrm{d}t} &= \alpha_B(\theta B^0 - B_{sp}) \\
&\quad + b_B^P(2(D_V(t-\tau_B) + D_V^*(t-\tau_B))(H_{Bsp}(t-\tau_B) + H_{Bsp}^*(t-\tau_B))B_{sp}(t-\tau_B)), \\
\frac{\mathrm{d}E_{sp}}{\mathrm{d}t} &= \alpha_E(\theta E^0 - E_{sp}) \\
&\quad + 2b_E^P(D_V(t-\tau_E) + D_V^*(t-\tau_E))(H_{Esp}(t-\tau_E) + H_{Esp}^*(t-\tau_E))E_{sp}(t-\tau_E) \\
&\quad - b_{ED_V} D_V^* E_{sp} - b_{EH_E} H_E^* E_{sp} - b_{EH_B} H_B^* E_{sp}.
\end{aligned}
$$

Соответственно добавляются пять параметров, требующих оценивания: величины запаздываний $\tau_P, \tau_{HB}, \tau_{HE}, \tau_B, \tau_E$.

A.3 Модель 3

Третья модель состоит из 7 ОДУ и описывается 18 параметрами.

$$\begin{aligned}
&\frac{\mathrm{d}T}{\mathrm{d}t} = \lambda - d_T T - kVT - \alpha_1 RT,\\
&\frac{\mathrm{d}T^*}{\mathrm{d}t} = kVT - \delta T^* - mE_m T^*,\\
&\frac{\mathrm{d}E_i}{\mathrm{d}t} = p_{E_i} - d_{E_i} * E_i - k_E V E_i - \alpha_2 R E_i\\
&\frac{\mathrm{d}E_m}{\mathrm{d}t} = k_E V E_i - \delta_E E_m - \alpha_3 R_\alpha E_m\\
&\frac{\mathrm{d}V}{\mathrm{d}t} = N\delta T^* - cV,\\
&\frac{\mathrm{d}R}{\mathrm{d}t} = p_R - d_r R - \gamma RV,\\
&\frac{\mathrm{d}R_\alpha}{\mathrm{d}t} = \gamma RV - \delta_R R_\alpha,
\end{aligned}$$

Таблица 21 — Параметры модели.

Параметр	Биологическое значение	ед. изм.
λ	Скорость производства клеток-мишеней	кл/(мл*д.)
d_T	Скорость гибели клеток-мишеней	1/д.
k	Скорость заражения клеток-мишеней	мл/(част. * д.)
δ	Скорость гибели инфицированных клеток-мишеней	1/д.
N	Скорость производства вирусных частиц	част./кл
c	Скорость гибели вирусных частиц	1/д.
m	Скор. уничтожения зараженных клеток CD8 Т-лимф.	мл/(кл. * д.)
δ_E	Скорость гибели CD8 Т-лимфоцитов	1/день
λ_E	Скорость производства CD8 Т-лимфоцитов	кл/(мл*д.)
α_1	Скор. подавления CD4 Т-лимф. натур. рег. Т-лимф.	мл/(кл. * д.)
p_{E_i}	Скор. производства незрелых CD8 Т-лимф.	кл/(мл*д.)
d_{E_i}	Скор. гибели незрелых CD8 Т-лимфоцитов	1/д.
k_E	Скор. перехода $E_i -> E_m$	мл/(част. * д.)
α_2	Скор. подавления E_i натуральными рег. Т-лимф.	мл/(кл. * д.)
α_3	Скор. подавления E_m адаптивными рег. Т-лимф.	мл/(кл. * д.)
p_R	Скор. производства натуральных рег.Т-лимф.	кл/(мл*д.)
d_R	Скор. гибели натуральных рег. Т-лимфоцитов	1/д.
γ_R	Скор. перехода $R -> R_\alpha$	мл/(част. * д.)
δ_R	Скор. гибели адаптивных рег. Т-лимфоцитов	1/д.

A.4 Модель 4

Четвертая модель отличается от третьей в одном уравнении:

$$\frac{\mathrm{d}E_m}{\mathrm{d}t} = k_E V(t-\tau) E_i(t-\tau) - \delta_E E_m - \alpha_3 R_\alpha E_m,$$

где τ — дополнительный параметр для оценивания.