Содержание

1 Аннотация

В работе рассматриваются численные методы решения кинетического уравнения Больцмана с интегралом Шахова. В них 3-мерные массивы (тензоры), возникающие в методе дискретных скоростей, подвергаются сжатию путём применения различных тензорных разложений, что приводит к значительному выигрышу в компьютерной памяти и асимптотике числа вычислений.

Рассматриваются проблемы, связанные с заменой полных тензоров базового алгоритма на различные представления. Также приводятся сравнения методов на тестовой задаче обтекания плоского цилиндра сверхзвуковым потоком.

2 Введение

2.1 Актуальность темы

Работа посвящена методам численного решения уравнения Больцмана с приближённым интегралом столкновений с использованием малопараметрических представлений многомерных массивов, также известных как тензорные разложения.

Есть важные прикладные области теории разреженных газов, такие как течения в электромеханических устройствах, микронасосах, соплах, моделирование полета летательных и спускаемых аппаратов в верхних слоях атмосферы и т.п., которые описываются кинетическим уравнением Больцмана и его различными упрощениями. Для исследования различных процессов необходимо численно решать такие уравнения.

Уравнение Больцмана описывает статистическое распределение частиц газа. При его численном решении возникают большие объёмы данных и вычислений, поэтому есть необходимость искать способы упростить вычисления за счет различных приближений с сохранением удовлетворительной точности.

2.2 Цель работы

Цель данной работы – исследовать возможность упрощения вычислений при решении уравнения Больцмана с модельным интегралом столкновений за счет применения формата Tensor-Train (тензорный поезд) и формата Таккера. Эти форматы позволяют получать малоранговые аппроксимации тензоров большой размерности с заранее заданной точностью, что может существенно сократить объём данных, давая выигрыш как по памяти, так и по количеству вычислений.

2.3 Методы исследования

Для численного решения уравнения Больцмана с модельным интегралом столкновений используется хорошо известный метод дискретных скоростей. Дискретизация внутри метода дискретных скоростей сделана с помощью конечно-объёмного метода первого порядка точности. Для оценки сложности различных операций для тензоров в различных форматов используются теоретические оценки сложности базовых алгоритмов линейной алгебры для вычисления матричных факторизаций.

Алгоритм метода написан на языке Python с применением библиотек numpy, ttpy, tucker3d и объектно-ориентированного программирования, что позволяет использовать

различные тензорные форматы не меняя общий код метода дискретных скоростей. Для визуализации результатов расчетов использовались библиотека matplotlib для Python и программный пакет Tecplot 360.

2.4 Научная новизна

Предложен новый вариант метода дискретных скоростей для численного решения уравнения Больцмана с интегралом Шахова с применением тензорных разложений. Впервые исследована возможность применения такого метода для численного решения задач внешней аэродинамики. Получены теоретические оценки на тензорные ранги нескольких вспомогательных тензоров, возникающих в методе дискретных скоростей.

2.5 Теоретическая значимость

Показаны требования к тензорным форматам, потенциальные проблемы, которые могут возникнуть при адаптации алгоритма, описаны случаи, когда использование модифицированных методов будет наиболее целесообразно. Описаны преимущества и недостатки предложенных методов.

2.6 Практическая значимость

Программный комплекс, разработанный в рамках данной работы, можно применять для реальных расчётов течений разреженного газа, при этом будет сильно сокращено количество компьютерной памяти, требуемой для расчёта.

2.7 Степень достоверности и апробация работы

Достоверность результатов была экспериментально подтверждена сравнениями с другими работами и статьями, а также собственными тестовыми расчётами. Результаты были опубликованы в нескольких научных изданиях, в том числе рецензируемых ВАК. Были сделаны доклады на научных конференциях:

1. «Применение тензорных разложений для численного решения уравнения Больцмана с модельным интегралом столкновений», А.В.Чикиткин, Е.К.Корнев, 62-я Всероссийская научная конференция МФТИ

2. «A Tensorized Version of LU-SGS Solver for Discrete Velocity Method for Boltzmann Kinetic Equation with Model Collision Integral», X All-Russian conference «Actual Problems of Applied Mathematics and Mechanics», Abrau-Durso, 2020

3. «Метод LU-SGS для решения уравнения Больцмана с модельным интегралом столкновений с использованием тензорных разложений», А.В.Чикиткин, Е.К.Корнев, 63-я Всероссийская научная конференция МФТИ

4. «Численное решение уравнения Больцмана с модельным интегралом столкновений с помощью тензорных разложений», А.В.Чикиткин, Е.К.Корнев, Международная конференция «Математические идеи П.Л. Чебышёва и их приложения к современным проблемам естествознания»

2.8 Публикации по теме

Основные результаты по теме работы изложены в статье [6], а также сборнике докладов конференции [18]. Была опубликована статья по смежной теме [5].

3 Обзор литературы

Долгое время проблема численного решения уравнения Больцмана поднималась довольно редко по причине высокой вычислительной сложности. Даже современные компьютерные технологии в большинстве практически важных случаев не позволяют найти решение с удовлетворительной точностью. Однако развитие линейной алгебры и численных методов в последнее время вернули интерес к проблеме и публикаций с каждым годом становится больше. Основные сложности заключаются в очень подробном описании системы при помощи шестимерной функции распределения и тяжёлом с вычислительной точки зрения интеграле столкновений. Можно выделить два больших направления в развитии численных методов решения уравнения Больцмана и его приближений:

1. сокращение размеров скоростной сетки
2. снижение трудоёмкости расчёта точного интеграла столкновений
3. снижение размерности задачи.

К первому направлению можно отнести работы, связанные с использованием неструктурированных сеток [25], адаптивных сеток [1, 17, 27], разреженных сеток [15]. Практичность этого направления исходит из того, что нам не нужны точные значения функции распределения во всех точках очень подробной сетки, а нужно лишь, чтобы функция достаточно точно численно интегрировалась и позволяла восстановить макропараметры. Поэтому, для того чтобы избежать избыточности, имеет смысл вводить неравномерные сетки. Для построения оптимальных сеток можно пользоваться априорной информацией о конкретной задаче и строить неструктурированную сетку «вручную», сгущая сетку вблизи наиболее часто встречающихся скоростей: например, в задаче внешнего обтекания, вблизи скорости в набегающем потоке и нулевой скорости.

Второе направление частично связано с первым, потому что, как минимум, интегрирование в правой части производится в том же пространстве скоростей. Во многих работах применяются упрощённые интегралы столкновений, как, например, уравнение БГК (Бхатнагара - Гросса - Крука) [11, 3] или модель Шахова (S-модель) [30], которые получаются из разложения в ряд интеграла обратных столкновений по полиномам Эрмита. Использование S-модели даёт удовлетворительную точность при решении многих прикладных задач, при этом сложность вычисления интеграла столкновений сравнивается со сложностью вычисления макропараметров. В [22] показано, что результаты

расчета по S-модели близки к расчетам методом прямого статистического моделирования и дают правильные значения тепловых потоков к поверхности тела. Это очень важно при решение прикладных задач внешней аэродинамики.

К третьему типу относятся работы, в которых высокая численная сложность преодолевается за счёт понижения размерности данных, принимающих участие в вычислениях. Функция распределения в большинстве точек близка к нулю и поэтому информация должна быть хорошо подвержена сжатию. Это может достигаться с помощью разложений в ряды Фурье [11], усечённых сингулярных разложений [7] и тензорных разложений [3].

При использовании фиксированной структурированной скоростной сетки при численном решении возникают 3-мерные массивы (тензоры), состоящие из проекции значений функции распределения на скоростную сетку. Существует много подходов для аппроксимации таких тензоров тензорами, которые задаются малым числом параметров [14]. Поэтому имеет смысл применять при численном решении уравнения Больцмана различные методы, разработанные для работы с многомерными тензорами.

Можно выделить следующие наиболее популярные тензорные разложения:

- каноническое разложение [8]
- разложение Таккера [26]
- TT-разложение [20]
- Иерархическое разложение [13]

Каноническое разложение, в целом, неприменимо на практике для размерностей больше второй, так как проблема нахождения разложения в общем случае NP-полна и не существует надёжных способов его нахождения.

Разложение Таккера лучше всего себя проявляет для невысоких размерностей (третьей-четвёртой), так как проклятие размерности не уходит полностью, а лишь немного смягчается уменьшением основания степени роста числа параметров.

TT-разложение (tensor-train) [20] является «улучшенной» версией предыдущего формата, которое изначально рассматривалось как иерархическое разложение Таккера. Оно намного лучше побеждает проклятие размерности, однако для данной задачи является избыточным и менее эффективным, чем разложение Таккера.

Существует довольно много работ, посвященных использованию тензорных разложений в численных методах для кинетических уравнений. В статье [16] описывается

метод, использующий разложение многомерного тензора в сумму Кронекеровых произведений малых тензоров, однако в статье получены только теоретические оценки ошибки, не приведены примеры расчетов прикладных задач. В [9] тензорные разложения успешно применялись для решения уравнения Власова с БГК-интегралом столкновений. В статье [3] каноническое разложение функции распределения в уравнении Больцмана с БГК-интегралом находится с помощью метода наименьших квадратов, но рассматривается простая пространственная геометрия. Методы понижения размерности для сложных геометрий рассматривались в [7]. Также тензорные разложения используются в [29] в похожей с вычислительной точки зрения задаче для уравнения Смолуховского, описывающего процессы агрегации частиц разного размера.

Данная работа посвящена разработке аналога метода дискретных скоростей для численного решения уравнения Больцмана с модельным интегралом столкновений, в котором используется представление тензора значений функции распределения в различных тензорных форматах. Новизна такого подхода заключается в том, что за счет приближенного представления тензоров со значениями функции распределения на скоростной сетке предлагается ускорить не только вычисление интеграла столкновений, но и решение всей задачи в целом на произвольной пространственной сетке. Применение обычного метода дискретных скоростей с единой равномерной скоростной сеткой приводит к очень большому объему данных, и даже на современных многопроцессорных архитектурах возникают ограничения по памяти. В работе показывается, что предложенный метод позволяет сократить размер требуемой памяти в 50-100 раз даже при использовании относительно небольших скоростных сеток.

Практическая значимость данной работы состоит в том, что разработанный подход с минимальными затратами может быть применен ко многим прикладным задачам. Это позволит решать некоторые задачи значительно проще, т.к. исчезнет необходимость в больших вычислительных мощностях. Эффективное решение задачи будет обеспечено хорошо работающим математическим аппаратом.

4 Математическая модель

Состояние разреженного газа в некотором объёме задаётся распределением частиц по пространству и скоростям: $f = f(t, \boldsymbol{x}, \boldsymbol{\xi})$. Основной математической моделью для течений разреженного газа является уравнение Больцмана:

$$\frac{\partial f}{\partial t} + \sum_{\alpha=1}^{3} \xi_\alpha \frac{\partial f}{\partial x_\alpha} = \int (f' f_1' - f f_1) g d\sigma d\boldsymbol{v_1} \equiv I(f, f) \tag{1}$$

Здесь f, f_1 – функции распределения частиц со скоростями $\boldsymbol{v}, \boldsymbol{v_1}$ до столкновения, f', f_1' – функции распределения частиц со скоростями $\boldsymbol{v'}, \boldsymbol{v_1'}$ после столкновения, $d\sigma = bdbd\epsilon$ – эффективное сечение рассеяния, $g = |\boldsymbol{v} - \boldsymbol{v_1}|$ - относительная скорость молекул. $I(f, f)$ называют больцмановским интегралом столкновений.

Макроскопические переменные - числовая плотность n, средняя скорость $\boldsymbol{u}$, температура T, поток тепла $\boldsymbol{q}$ и т. д., вычисляются через интегралы по пространству скоростей от функции распределения, умноженной на степени компонент скорости, т.е. как моменты распределения:

$$\begin{aligned}
&n = \int f d\boldsymbol{\xi}, \quad \boldsymbol{u} = \frac{1}{n} \int \boldsymbol{\xi} f d\boldsymbol{\xi}, \\
&T = \frac{1}{3nR_g} \left(\int \xi^2 f \, d\boldsymbol{\xi} - nu^2 \right), \quad \rho = mn, \; p = \rho R_g T, \\
&\boldsymbol{v} = \boldsymbol{\xi} - \boldsymbol{u}, \quad \boldsymbol{q} = \frac{1}{2} m \int \boldsymbol{v} v^2 f \, d\boldsymbol{\xi}, \\
&u^2 = \sum_{\alpha=1}^{3} u_\alpha u_\alpha, \; v^2 = \sum_{\alpha=1}^{3} v_\alpha v_\alpha, \; d\boldsymbol{\xi} = d\xi_x d\xi_y d\xi_z.
\end{aligned} \tag{2}$$

Здесь R_g – газовая постоянная, m – масса одной молекулы.

Основные сложности с численным уравнением Больцмана связаны с его высокой размерностью и многомерным интегралом столкновений. Существует несколько упрощений интеграла столкновений, которые снижают затраты на его вычисление. В данной работе рассматривается приближение, называемое моделью Шахова или S-моделью [30], хотя разработанный подход может быть применен и к другим модельным интегралам. Идея вывода приближенного интеграла заключается в том, что предполагается, что интеграл столкновений зависит только от функции распределения при данных скоростях и от вектора макропараметров $\boldsymbol{a}$:

$$\frac{\partial f}{\partial t} + \xi_\alpha \frac{\partial f}{\partial x_\alpha} = J(f, \boldsymbol{\xi}, \boldsymbol{a}(\boldsymbol{x}, t)) \tag{3}$$

Конкретный вид функции J находится из условия, что какое-то количество первых

моментов, вычисленных по точному и приближенному интегралу, совпадают:

$$\int \phi(\boldsymbol{\xi}) I(f,f) d\boldsymbol{\xi} = \int \phi(\boldsymbol{\xi}) J(f,\boldsymbol{a}) d\boldsymbol{\xi}, \phi(\boldsymbol{\xi}) = 1, \boldsymbol{\xi}, \xi^2, \boldsymbol{\xi}\xi^2 \dots \tag{4}$$

В случае восьми моментов получается следующий модельный интеграл

$$\begin{aligned} &J(f)(t,\boldsymbol{x},\boldsymbol{\xi}) = \nu(t,\boldsymbol{x})(f^S(t,\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{a}) - f(t,\boldsymbol{x},\boldsymbol{\xi})),\ \nu = \frac{p}{\mu(T)} \\ &f^S(t,\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{a}) = f^M\left[1 + \frac{4}{5}(1-Pr)\frac{n}{m(2R_gT)^2}\left(\sum_{\alpha=1}^{3} q_\alpha v_\alpha\right)\left(\frac{v^2}{2R_gT} - \frac{5}{2}\right)\right] \\ &f^M = f^M(t,\boldsymbol{x},\boldsymbol{\xi};\boldsymbol{a}) = \frac{n}{(2\pi R_gT)^{3/2}} \exp\left(-\frac{v^2}{2R_gT}\right), \boldsymbol{v} = \boldsymbol{\xi} - \boldsymbol{u}. \end{aligned} \tag{5}$$

Здесь $\mu = \mu(t)$ – динамическая вязкость, $Pr = 2/3$ – число Прандтля для одноатомного газа, f^M – максвелловская (равновесная) функция распределения.

Для поверхностей тел или стенок используется граничное условие диффузного отражения. Для этого вычисляются макропараметры для построения функции распределения отражённых молекул. При заданной температуре поверхности T_w:

$$\begin{aligned} n_w &= -\frac{N_-}{N_+}, \\ N_- &= \int\limits_{\xi_{li}<0} f d\boldsymbol{\xi}, \\ N_+ &= \int\limits_{\xi_{li}\geq 0} f^M(1,\boldsymbol{0},T_w) d\boldsymbol{\xi} \end{aligned} \tag{6}$$

Здесь $f^M(n,\boldsymbol{u},T)$ – функция распределения Максвелла для скоростной сетки с заданными плотностью, скоростью и температурой соответственно.

Тогда считаем, что на внешней стороне грани значение функции распределения $f_w = f^M(n_w,\boldsymbol{0},T_w)$. В результате обеспечивается точное выполнение условия непротекания:

$$\int\limits_{\xi_{li}\geq 0} f_w d\boldsymbol{\xi} + \int\limits_{\xi_{li}<0} f d\boldsymbol{\xi} = 0$$

Если задача имеет плоскость симметрии, то на ней можно ввести граничное условие, тем самым сильно сократив объём вычислений. Если на внутренней грани (находящейся со стороны области) ф. р. имеет вид $f(t,\boldsymbol{x},\boldsymbol{v})$, то граничное условие будет $f(t,\boldsymbol{x},\boldsymbol{\xi} - 2v_n\boldsymbol{n})$. В случае с массивами условие можно реализовать перестановкой значений в обратном порядке по соответствующему индексу.

Набегающий поток реализуется простым заданием нужной ф. р. с внешней стороны граничной грани.

5 Метод дискретных скоростей (базовый метод)

Одним из наиболее распространённых методов численного решения уравнения Больцмана является метод дискретных скоростей (МДС).

МДС заключается в том, что вместо интегрирования по всему пространству скоростей производится численное интегрирование по разбиению некоторой ограниченной области, вид которой определяется видом задачи и требованиями к точности решения.

В данной работе мы используем равномерную трёхмерную сетку вида $n_v \times n_v \times n_v$ с узлами $\xi_{i_1 i_2 i_3} = (\xi_{1,i_1}, \xi_{2,i_2}, \xi_{3,i_3})$ в центрах ячеек. Сетка задаётся двумя параметрами n_v и $\xi_{min} = -\xi_{max}$ и представляет из себя прямое произведение одномерных равномерных сеток по каждой из координат:

$$\xi_{\alpha,i_\alpha} = \xi_{min} + (i_\alpha - 1)\Delta\xi,\ i_\alpha = 1, \ldots, n_v,\ \alpha = 1, 2, 3$$

Основное требование к сетке - вычисление возникающих в задаче интегралов с удовлетворительной точностью. При этом могут возникнуть два типа проблем 1:

- Шаг сетки слишком крупный и пики ф. р. не разрешаются.
- Ненулевые значения функции распределения попали за границу сетки.

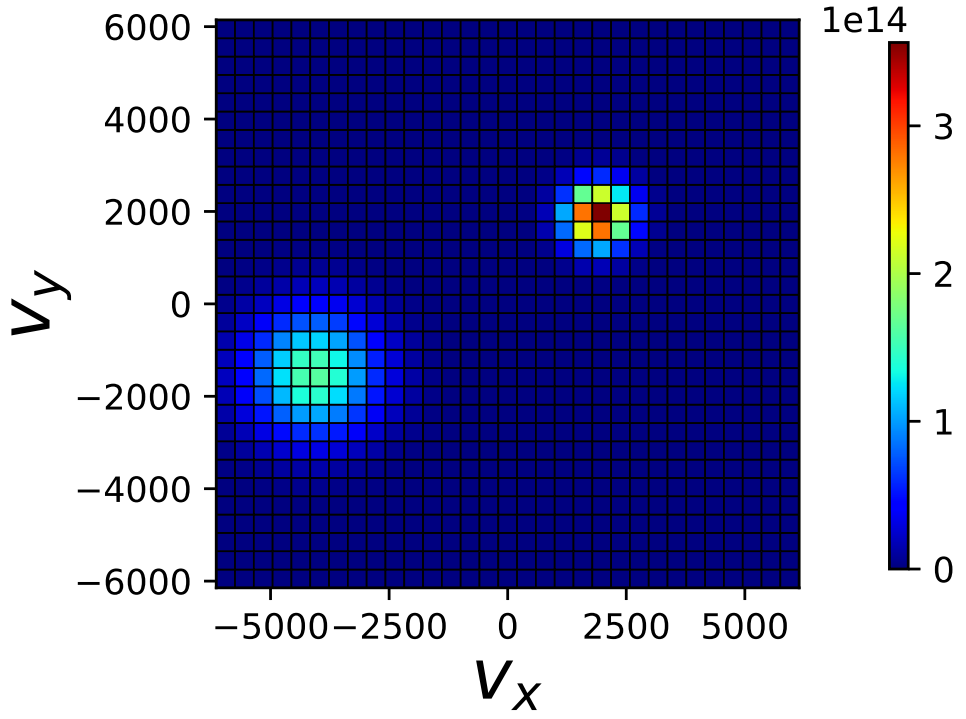

Рис. 1: Срез проекции суммы двух Максвеллианов на равномерную скоростную сетку.

Размеры и шаг скоростной сетки подбираются путём интегрирования моментов функции Максвелла с параметрами, характерными для задачи. Сетка считается подходящей, если интегралы считаются достаточно точно. При этом нет гарантии, что после некоторого количества временных шагов полученные в ходе эволюции решения функции распределения не перестанут хорошо интегрироваться. Таким образом, подбор оптимальной сетки уже представляется очень важной задачей на этапе планирования, особенно учитывая, что объём, занимаемый сеткой в памяти, зависит как 3-я степень от n_v.

Кроме того, можно использовать следующие оценочные формулы из [28]:

$$|\xi_\alpha| \le \max_{i=1}^{n_c}(\beta\sqrt{T} + |u_\alpha|),\ \Delta\xi \ge (0.5\ldots1)\min_{i=1}^{n_c} T,\ \alpha = 1,2,3$$

Здесь n_c – число ячеек пространственной сетки, максимумы и минимумы берутся по пространству, $\beta = 3\ldots4$.

Таким образом, функция распределения $f(t, \boldsymbol{x}, \xi_1, \xi_2, \xi_3)$ оказывается представлена в виде трехмерного массива (тензора) $\hat{f}(t, \boldsymbol{x})$ размера $n_v \times n_v \times n_v$, где $\hat{f}(t, \boldsymbol{x})(i_1, i_2, i_3)$ соответствует числу частиц, обладающих скоростями, лежащими в скоростной ячейке. Далее все тензоры будут помечаться символом $\hat{\cdot}$

Происходит замена точных интегралов квадратурными формулами. В данном случае используется многомерная формула прямоугольников с центральной точкой:

$$\int B(t, \boldsymbol{x}, \boldsymbol{\xi})\boldsymbol{f}d\boldsymbol{v} \to \Delta\xi^3 \sum_{i_1,i_2,i_3} B(t, \boldsymbol{x}, \xi_{1,i_1}, \xi_{2,i_2}, \xi_{3,i_3})\hat{f}(i_1, i_2, i_3) \tag{7}$$

где B произвольная функция (обычно это степени скорости).

Записав уравнение (3) для каждого узла скоростной сетки, получаем систему уравнений:

$$\frac{\partial \hat{f}(t,x)}{\partial t} + \frac{\partial}{\partial x_\alpha}\left(\hat{v}_\alpha \circ \hat{f}\right) = \hat{J} = \nu(\hat{f}^{(S)} - \hat{f}) \tag{8}$$

$\hat{v}_\alpha$ – тензор, заполненный α-компонентами скоростной сетки.

Далее область геометрического пространства разбивается на L ячеек гексаэдральной формы, образованных шестью четырёхугольными гранями каждая. Индексом i будем обозначать номер ячейки пространства.

Для получения дискретных уравнений конечно-объёмного метода уравнение (8) инегрируется по объёму ячейки V_i, затем, в соответствии с формулой Гаусса — Остроградского, интеграл по объёму преобразуется в сумму потоков через грани, спроециро-

ванных на нормали:

$$\frac{\partial \hat{\overline{f}}_i}{\partial t} = \hat{R}_i = -\frac{1}{|V_i|}\sum_{l=1}^{6}\hat{\Phi}_{li} + \hat{J}_i \tag{9}$$

Здесь

$$\hat{\overline{f}}_i = \frac{1}{|V_i|}\int_{V_i}\hat{f}(t_n, \boldsymbol{x})d\boldsymbol{x}$$

– интегральное среднее функции распределения в ячейке с индексом i на шаге по времени n, $t_n = \tau n$, τ – шаг по времени. $\hat{\Phi}_{li}$ – численный поток через l-ю грань i-й ячейки пространства.

У гладких функций интегральные средние отличаются от значений в центрах ячеек на $O(h^2)$, где h – линейный размер ячейки, поэтому для схем 2-го порядка они взаимозаменяемы. Далее черточка над интегральными средними будет опускаться. Интегральное среднее от источникового члена также заменяется значением от аргумента $\hat{\overline{f}}_i$.

Вычисление потоков на границах между ячейками удобно разбить на 2 этапа, которые обычно используются для решения более сложных нелинейных гиперболических систем:

1. Реконструкция: по значениям интегральных средних на каждой грани вычисляются значения слева и справа от грани (по отношению к направлению нормали): $\hat{f}^L_{li}$, $\hat{f}^R_{li}$.

2. Решение задачи о распаде разрыва (задачи Римана): по двум значениям $\hat{f}^L_{li}$, $\hat{f}^R_{li}$, которые в общем случае не совпадают, вычисляется значение потока на грани $\hat{\Phi}_{li}$, соответствующее решению задачи Римана.

Для каждой грани повернём матрицы скоростей так, чтобы распад разрыва можно было рассматривать как одномерный. Пусть

$$\hat{\xi}_{li} = n_1\hat{\xi}_1 + n_2\hat{\xi}_2 + n_3\hat{\xi}_3,$$

где $\boldsymbol{n} = (n_1, n_2, n_3)$ – вектор нормали к грани.

Тогда численный поток $\hat{\Phi}_{li}$ вычисляется по противопоточной формуле:

$$\hat{\Phi}_{li} = \frac{1}{2}|S_{li}|\big(\hat{\xi}_{li} \circ (\hat{f}_L + \hat{f}_R) - |\hat{\xi}_{li}| \circ (\hat{f}_R - \hat{f}_L)\big), \tag{10}$$

Можно также применить упрощенный поток типа Русанова, в котором вместо модуля скорости во втором слагаемом используется некоторая оценка скорости:

$$\hat{\Phi}_{li} = \frac{1}{2}|S_{li}|\big(\hat{\xi}_{li} \circ (\hat{f}_L + \hat{f}_R) - \hat{\xi}_{max} \circ (\hat{f}_R - \hat{f}_L)\big) \tag{11}$$

В качестве оценки можно взять, например, $\hat{\xi}_{max}(i_1, i_2, i_3) = \sqrt{\xi^2_{1,i_1} + \xi^2_{2,i_2} + \xi^2_{3,i_3}}$, $|S_{li}|$ – площадь грани.

5.1 Явный метод

Для интегрирования по времени системы (9) можно использовать явный метод Эйлера, то есть брать правую часть с предыдущего шага по времени:

$$\frac{\Delta \hat{f}_i}{\Delta t} = \hat{R}^n_i, \quad \Delta \hat{f}_i = \hat{f}^{n+1}_i - \hat{f}^n_i, \; \hat{R}^n_i = -\frac{1}{|V_i|} \sum_l \hat{\Phi}_{li} + \hat{J}_i(\hat{f}^n_i), \; i = 1, \ldots, N \tag{12}$$

Для описания алгоритма базового метода введем следующие обозначения:

1. n_c – число всех ячеек в пространственной сетке.
2. n_f – число всех граней.
3. $\hat{\xi}_1, \hat{\xi}_2, \hat{\xi}_3$ – тензоры со значениями x, y, z компонент скорости в каждом узле скоростной сетки.
4. $\hat{f}_L[j], \hat{f}_R[j]$ – значение ф.р. в ячейке слева и в ячейке справа от грани j по отношению к направлению нормали к этой грани (нормаль направлена вправо)
5. $\hat{F}[j]$ – численный поток по нормали к грани j
6. $S[j]$ – площадь грани
7. $V[i]$ – объём i-й ячейки
8. $\hat{\xi}_n[j] = \hat{\xi}_1 n_1[j] + \hat{\xi}_2 n_2[j] + \hat{\xi}_3 n_3[j]$, где n_1, n_2, n_3 – компоненты единичной нормали к грани j. Т.е. $\hat{\xi}_n[j]$ состоит из проекций вектора скорости на нормаль в каждом узле скоростной сетки
9. $sign[i, j] = +1$, если нормаль к грани j является внешней по отношению к ячейке i и -1 в противном случае

В принятых обозначениях алгоритм одного шага по времени имеет вид 1.

Ниже приведен псевдо-код функции для вычисления модельного интеграла столкновений 2. Функция sum вычисляет сумму всех элементов тензора. Символом $\hat{\mathbb{1}}$ обозначен тензор, состоящий из всех единиц.

Algorithm 1 Алгоритм явного шага по времени

1: ... ▷ Постановка граничных условий и реконструкция
2: **for** $j = 1, n_f$ **do** ▷ Вычисление потоков на гранях
3: $\quad \hat{F}[j] = \frac{1}{2}\hat{\xi}_n[j] \circ \left(\hat{f}_L[j] + \hat{f}_R[j]\right) - \frac{1}{2}|\hat{\xi}_n[j]| \circ \left(\hat{f}_R[j] - \hat{f}_L[j]\right)$
4: **end for**
5: **for** $i = 1, n_c$ **do** ▷ Вычисление правой части R
6: $\quad \hat{R}[i] = \textbf{computeJ}(\hat{f}[i])$ ▷ Вычисляем модельный интеграл столкновений
7: $\quad$ **for** $j \in \{$грани ячейки $i\}$ **do** ▷ Цикл по всем граням ячейки
8: $\quad\quad \hat{R}[i] = \hat{R}[i] - sign[i,j]\frac{S[j]}{V[i]}\hat{F}[j]$ ▷ Добавляем поток с нужным знаком
9: $\quad$ **end for**
10: **end for**
11: **for** $i = 1, n_c$ **do** ▷ Вычисление значений на следующем шаге по времени
12: $\quad \hat{f}[i] = \hat{f}[i] + \Delta t \hat{R}[i]$
13: **end for**

Algorithm 2 Процедура вычисления интеграла столкновения

1: **procedure** COMPUTEJ($\hat{f}, \hat{v}_1, \hat{v}_2, \hat{v}_3$)
2: $\quad n = \Delta\xi^3 \,\mathrm{sum}(f)$ ▷ Числовая плотность
3: $\quad u_1 = \Delta\xi^3 \,\mathrm{sum}(\hat{v}_1 \circ \hat{f})/n$
4: $\quad$ ...
5: $\quad \widehat{\xi^2} = \hat{\xi}_1 \circ \hat{\xi}_1 + \hat{\xi}_2 \circ \hat{\xi}_2 + \hat{\xi}_3 \circ \hat{\xi}_3$
6: $\quad u^2 = u_x^2 + u_y^2 + u_z^2$
7: $\quad T = \frac{1}{3nR_g}\left(\Delta\xi^3 \,\mathrm{sum}(\widehat{\xi^2} \circ \hat{f}) - n\,u^2\right)$
8: $\quad \rho = mn$ ▷ массовая плотность, m – масса молекулы
9: $\quad p = \rho R_g T$
10: $\quad \hat{c}_1 = \frac{\hat{\xi}_1 - u_1\hat{\mathbb{1}}}{\sqrt{2R_gT}}$
11: $\quad$ ...
12: $\quad \widehat{c^2} = \hat{c}_1 \circ \hat{c}_1 + \hat{c}_2 \circ \hat{c}_2 + \hat{c}_3 \circ \hat{c}_3$
13: $\quad S_1 = \Delta\xi^3 \,\mathrm{sum}(\hat{c}_1 \circ \widehat{c^2} \circ f)$
14: $\quad$ ...
15: $\quad \hat{f}_M = \mathrm{maxwell}(n, T, u_1, u_2, u_3, \hat{\xi}_1, \hat{\xi}_2, \hat{\xi}_3)$
16: $\quad \hat{f}_S = \hat{f}_M \circ \left(1 + \frac{4}{5}(1 - Pr)\,(S_1\hat{c}_1 + S_2\hat{c}_2 + S_3\hat{c}_3) \circ \left(\widehat{c^2} - \frac{5}{2}\hat{\mathbb{1}}\right)\right)$
17: $\quad J = \frac{p}{\mu(T)}\,(f_S - f)$
18: $\quad$ **return** J
19: **end procedure**

Важно отметить, в приведенном алгоритме используются три операции с тензорами: покомпонентное сложение, покомпонентное умножение, и вычисление суммы всех элементов тензора.

5.1.1 Неявный метод

Для решения системы неявным методом Эйлера линеаризуем правую часть с учётом предыдущего шага по времени.

$$\frac{\Delta\hat{f}_i}{\Delta t} = \hat{R}_i^{n+1} = \hat{R}_i^n + \left(\frac{\partial R}{\partial f}\right)^n \Delta\hat{f}_i, \quad \Delta\hat{f}_i = \hat{f}_i^{n+1} - \hat{f}_i^n,\ \hat{R}_i^n = -\frac{1}{|V_i|}\sum_l \hat{\Phi}_{li} + \hat{J}_i(\hat{f}_i^n),\ i = 1, \ldots, n_c \tag{13}$$

Здесь R и f – векторы тензоров: $R = [\hat{R}_1, \ldots, \hat{R}_{n_c}]$, $f = [\hat{f}_1, \ldots, \hat{f}_{n_c}]$.

Соответственно, $\frac{\partial R}{\partial f}$ это матрица Якоби с элементами в виде тензоров. Для приближенной линеаризации модельного интеграла столкновений мы опускаем сложную зависимость ν от $\hat{f}_i$ и берём ν с предыдущего шага. Поток в ячейке i зависит только от $\hat{f}_i$ и значений $\hat{f}_{i_l}$ в соседних ячейках i_l. Линеаризация описывается следующими формулами:

$$\hat{J}_i^{n+1} \approx \hat{J}_i^n - \nu_i^n \Delta\hat{f}_i, \quad \hat{\Phi}_{li}^{n+1} \approx \hat{\Phi}_{li}^n + \frac{\partial\hat{\Phi}_{li}^n}{\partial\hat{f}_i^n} \circ \Delta\hat{f}_i + \frac{\partial\hat{\Phi}_{li}^n}{\partial\hat{f}_{i_l}^n} \circ \Delta\hat{f}_{i_l} \tag{14}$$

Подставляя в (13), получаем i-е уравнение:

$$\left((\frac{1}{\Delta t} + \nu_i^n)\hat{\mathbb{1}} + \frac{1}{2|V_i|}\sum_l \hat{\xi}_{il} \circ (\hat{\mathbb{1}} + sign(\hat{\xi}_{il}))|A_{il}|\right) \circ \Delta\hat{f}_i + \\ + \frac{1}{2|V_i|}\sum_l \hat{\xi}_{il} \circ (\hat{\mathbb{1}} - sign(\hat{\xi}_{il}))|A_{il}| \circ \Delta\hat{f}_{i_l} = R_i^n \tag{15}$$

где $\hat{\mathbb{1}}$ это тензор, все элементы которого равны единице.

Мы получили СЛАУ относительно вектора тензоров Δf, а матрица системы является квадратной матрицей из тензоров. Заметим, что эта матрица имеет сильное блочно-диагональное преобладание, что следует из вида диагональных членов и малости Δt.

Для приближенного решения этой системы на каждом шаге по времени мы будем использовать одну итерацию метода LU-SGS (Lower-Upper Symmetric Gauss-Seidel). Этот метод был предложен для решения линейных систем, возникающих в неявных конечно-объемных методах для уравнений Эйлера и Навье-Стокса [4]. Впоследствии он

применялся для решения линейных системы в методе дискретных скоростей в работах [21, 25].

Опишем метод LU-SGS. Для системы уравнений $M\Delta f = R^n$, в которой матрица M обладает диагональным преобладанием, метод основан на следующей приближенной факторизации:

$$M\Delta\hat{f} = (L + D + U)\Delta\hat{f} \approx (L + D)D^{-1}(D + U)\Delta f = R^n$$
$$\begin{cases} (D + L)\Delta f^* = -\hat{R}^n \\ (D + U)\Delta f^n = D\Delta f^*, \end{cases} \tag{16}$$

где L, U, D строго нижнетреугольная, верхнетреугольная и диагональные матрицы: $M = L + D + U$. Учитывая, что $LD^{-1}U$ мала, мы получаем приближенную факторизацию и решаем последовательно две треугольные системы методом подстановки. В случае нашей задачи элементами матриц являются 3-х мерные тензоры.

Необходимость в неявном методе возникает из-за сильного ограничения явного метода на шаг по времени, из-за чего решение может очень медленно сходиться к стационарному. Если опустить правую часть уравнения, а слева оставить только уравнение переноса, то можно понять, откуда возникают эти ограничения:

1. В неструктурированной пространственной сетке могут быть ячейки сильно различающихся размеров. Обычно сетки сгущаются ближе к некоторой поверхности, при этом число Куранта для задачи переноса будет вычисляться по размерам наименьшей ячейки.

2. Максимальная скорость переноса равна $\sqrt{3}\xi_{max}$, что также может вызвать избыточное ограничение на шаг по времени.

Алгоритм одного шага по времени в неявном методе приведён ниже 3.

Algorithm 3 Алгоритм неявного шага по времени

1: ... ▷ Постановка граничных условий и реконструкция

2: **for** $j = 1, n_f$ **do** ▷ Вычисление потоков на гранях

3: $\hat{F}[j] = \frac{1}{2}\hat{\xi}_n[j] \circ \left(\hat{f}_L[j] + \hat{f}_R[j]\right) - \frac{1}{2}|\hat{\xi}_n[j]| \circ \left(\hat{f}_R[j] - \hat{f}_L[j]\right)$

4: **end for**

5: **for** $i = 1, n_c$ **do** ▷ Вычисление правой части R

6: $\hat{R}[i] = \textbf{computeJ}(\hat{f}[i])$ ▷ Вычисление модельного интеграла столкновений

7: **for** $j \in \{$грани ячейки $i\}$ **do** ▷ Цикл по всем граням ячейки

8: $\hat{R}[i] = \hat{R}[i] + sign[i, j]\frac{A[j]}{V[i]}\hat{F}[j]$ ▷ Добавляем поток с нужным знаком

9: **end for**

10: **end for**

11: **for** $i = n_c, 1$ **do** ▷ Обратный проход LU-SGS

12: $\Delta\hat{f}[i] = \hat{R}[i]$

13: **for** $j \in \{$грани ячейки $i\}$ **do**

14: $\hat{\xi}_n^-[j] = \frac{\hat{\xi}_n[j] - sign(\hat{\xi}_n[j])\hat{\xi}_n[j]}{2}$

15: **if** $(i_j \geq 0)$ **and** $(i_j > i)$ **then**

16: $\Delta\hat{f}[i] = \Delta\hat{f}[i] + \frac{A[j]}{V[i]}\hat{\xi}_n^-[j]\Delta\hat{f}[i_j]$

17: **end if**

18: **end for**

19: $\Delta\hat{f}[i] = \Delta\hat{f}[i] / \left((\frac{1}{\Delta t} + \nu_i)\hat{\mathbb{1}} + \frac{1}{2}\frac{1}{|V_i|}\sum_j A[j]sign(\hat{\xi}_n[j])\hat{\xi}_n[j]\right)$ ▷ Деление на диагональный коэффициент

20: **end for**

21: ... ▷ Прямой проход LU-SGS

22: **for** $i = 1, n_c$ **do** ▷ Обновить значения в ячейках

23: $\hat{f}[i] = \hat{f}[i] + \Delta\hat{f}[i]$

24: **end for**

6 Тензорные разложения

В этом разделе описаны основные свойства тензорных разложений tensor-train (тензорный поезд) и разложения Таккера, которые были использованы при разработке модифицированного МДС.

Основные идеи, лежащие в основе всех тензорных разложений, можно пояснить на примере тензора размерности 2 (т.е. матрицы). В этом случае самым простым разложением является сингулярное разложение матрицы

$$A = \sum_{\alpha=1}^{r} \sigma_\alpha \boldsymbol{u}_\alpha \boldsymbol{v}_\alpha^*, \; A(i,j) = \sum_{\alpha=1}^{r} \sigma_\alpha \boldsymbol{u}(i)_\alpha \boldsymbol{v}(j)_\alpha^*$$

где r - ранг A. Его отличительной особенностью является то, что по теореме Эккарта — Янга наилучшим приближением матрицей с фиксированным рангом $k < r$ в смысле минимума Фробениусовой нормы ошибки является матрица

$$A_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^*$$

[10]. Таким образом происходит некоторое «сжатие» данных, так как часть сингулярных векторов отбрасывается, и матрица, как функция двух индексов, представляется в виде суммы небольшого числа слагаемых в виде произведения двух функций одного индекса. Таким образом осуществляется разделение переменных.

Сингулярное разложение является частным случаем канонического разложения. Каноническое разложение представляет d-мерный тензор A в виде

$$A(i_1, i_2, ..., i_d) = \sum_{\alpha=1}^{r} u_1(\alpha, i_1) u_2(\alpha, i_2) ... u_d(\alpha, i_d)$$

Минимальное число r для которого существует такое представление называется тензорным рангом. В общем случае для размерностей выше второй тензорный ранг найти не представляется возможным. Существует тензор размера $9 \times 9 \times 9$, для которого до сих пор неизвестен его тензорный ранг. Известно, что он лежит между 20 и 24.

6.1 Tensor-Train

Это разложение имеет вид:

$$A(i_1, \ldots, i_d) = \sum_{\alpha_1, \ldots, \alpha_{d-1}} G_1(i_1, \alpha_1) G_2(\alpha_1, i_2, \alpha_2) \ldots G_d(\alpha_{d-1}, i_d), \alpha_k = 1, \ldots, r_k. \tag{17}$$

G_k называются ядрами разложения. Два ядра – первое и последнее – являются матрицами, а все остальные – 3-х мерными тензорами. Числа r_k называются TT-рангами.

Можно записать это разложение в более кратком виде, как произведение матриц:

$$A(i_1, \ldots, i_d) = G_1(i_1)G_2(i_2)\ldots G_d(i_d) \tag{18}$$

Здесь $G_1(i_1)$ – вектор строка, $G_d(i_d)$ – вектор столбец, остальные $G_k(i_k)$ – матрицы.

Для простоты представим, что $n_\alpha = n$ и $r_\alpha = r$. Тогда количество параметров, необходимых для представления тензора, растёт как $O(dnr^2)$.

Для применения TT-разложения в методе дискретных скоростей важны следующие свойства:

1. Есть алгоритм для вычисления приближения B с минимальными рангами в TT-формате к тензору A с заданной точностью

   $$\|A - B\|_F \leq \epsilon \|A\|_F$$

   $\|\cdot\|_F$ – норма Фробениуса (корень квадратный из суммы квадратов всех элементов)

2. Если тензоры A и B одинакового размера заданы в TT формате, т.е.

   $$A(i_1, \ldots, i_d) = A_1(i_1)\ldots A_d(i_d),\ B(i_1, \ldots, i_d) = B_1(i_1)\ldots B_d(i_d)$$

   т.е. поэлементная сумма двух тензоров $C = A + B$ представима в TT-формате с ядрами:

   $$C_k(i_k) = \begin{bmatrix} A_k(i_k) & 0 \\ 0 & B_k(i_k) \end{bmatrix},\ k = 2, \ldots, d-1$$

   $$C_1(i_1) = \begin{bmatrix} A_1(i_1)\ B_1(i_1) \end{bmatrix},\ C_d(i_d) = \begin{bmatrix} A_d(i_d) \\ B_d(i_d) \end{bmatrix}$$

   T.е. поэлементное сложение не требует никаких вычислений, при этом ранги суммы равны сумме рангов слагаемых, т.е. ранги складываются.

3. Поэлементное (адамарово) произведение $C = A \circ B$ двух тензоров представимо в TT-формате с ядрами вида

   $$C_k(i_k) = A_k(i_k) \otimes B_k(i_k)$$

   где $\otimes$ – кронекерово произведение матриц.

   Это означает, что такая операция требует $O(dnr^4)$ операций, и ранги перемножаются.

4. Есть алгоритм для вычисления округления тензора в TT-формате, т.е. для тензора A в TT-формате с рангами r_k можно найти тензор B с меньшими рангами, такой что

$$\|A - B\|_F \leq \epsilon \|A\|_F$$

Алгоритм состоит из последовательности сингулярных разложений и QR разложений вспомогательных матриц разверток, и имеет сложность $O(dnr^3)$

5. Поэлементное деление на тензор ранга один:

$$C(i_1, \ldots, i_d) = \frac{A(i_1, \ldots, i_d)}{B(i_1, \ldots, i_d)} = \frac{\overbrace{G_1^A(i_1) \ldots G_d^A(i_d)}^{\text{матрицы}}}{\underbrace{G_1^B(i_1) \ldots G_d^B(i_d))}_{\text{скаляры}}} = \frac{G_1^A(i_1)}{G_1^B(i_1)} \cdots \frac{G_d^A(i_d)}{G_d^B(i_d)}$$

где G_α^B – одномерные векторы.

6. Вычисление взвешенной суммы элементов тензора:

$$S = \sum_{i_1, i_2, i_3} A(i_1, i_2, i_3) w_1(i_1) w_2(i_2) w_3(i_3).$$

Алгоритмы для вышеперечисленных операций описаны в [20].

Вид TT-разложения (17) показывает, что для размерности $d = 3$ формат несколько избыточен, т.к. для представления используются тензоры той же размерности 3. Поэтому для модификации МДС был опробован и другой формат, а именно формат Таккера.

6.2 Разложение Таккера

Данное разложение имеет вид:

$$A(i_1, \ldots, i_d) = \sum_{\alpha_1, \ldots, \alpha_d = 1} G(\alpha_1, \ldots, \alpha_d) U_1(i_1, \alpha_1) \ldots U_d(i_d, \alpha_d), \alpha_k = 1, \ldots, r_k. \quad (19)$$

Здесь G это d-мерный тензор, который называется ядром разложения, а U_i это матрицы, которые называют факторами разложения. В данной работе мы будем рассматривать только случай $d = 3$. Также, для краткой записи оценок сложности вычислений и затрат памяти, будем считать, что $n_\alpha = n$ и $r_\alpha = r$.

Все те же операции, реализованные для TT-формата, есть и для формата Таккера:

1. Алгоритм для вычисления приближения B с минимальными рангами в формате Таккера к тензору A с заданной относительной ошибкой по норме Фробениуса:

$$\|A - B\|_F \leq \epsilon \|A\|_F$$

Требует $O(n^4)$ операций.

2. Вычисление поэлементной суммы двух тензоров $A+B = C$. Здесь ядро и факторы тензора C выражаются через ядра и факторы слагаемых таким образом:

$$G^C(k_1, k_2, k_3) = \begin{cases} G^A(k_1, k_2, k_3), & \text{при } 1 \leq k_\alpha \leq r_\alpha^A \\ G^B(k_1 - r_1^A, k_2 - r_2^A, k_3 - r_3^A), & \text{при } r_\alpha^A < k_\alpha \leq r_\alpha^A + r_\alpha^B \\ 0, & \text{иначе} \end{cases}$$

$$k_\alpha = 1, \ldots, r_\alpha^A + r_\alpha^B$$

$$U_\alpha^C = \begin{bmatrix} U_\alpha^A & U_\alpha^B \end{bmatrix}.$$

Также как и в случае с TT-форматом, вычисления не требуются, ранги складываются.

3. Поэлементное (адамарово) произведение $C = A \circ B$. Ядро и факторы вычисляются по следующим формулам:

$$G^C(k_1, k_2, k_3) = G^A(k_1^A, k_2^A, k_3^A) G^B(k_1^B, k_2^B, k_3^B),$$

$$k_\alpha^A = \lceil k_\alpha / r_\alpha^B \rceil, \quad k_\alpha^B = mod(k_\alpha, r_\alpha^B) + 1, \; k_\alpha = 1, \ldots, r_\alpha^A r_\alpha^B$$

$$U_\alpha^C(i_\alpha, :) = U_\alpha^A(i_\alpha, :) \otimes U_\alpha^B(i_\alpha, :).$$

Здесь $mod(a, b)$ это остаток от деления a на b, $\otimes$ – Кронекерово произведение и $U_\alpha^A(i_\alpha, :)$ – строка матрицы (по аналогии с numpy slicing).

Операция требует $O(r^6 + nr^2)$ элементарных операций; ранги перемножаются.

4. Округление с минимальной относительной ошибкой ϵ или до заданного ранга $r' < r$. Реализовано через усечённые сингулярные разложения факторов и требует $O(nr^2 + nrr' + r^4)$ операций.

5. Поэлементное деление на одноранговый тензор. Аналогично:

$$G^C(k_1, k_2, k_3) = \frac{G^A(k_1, k_2, k_3)}{G^B(1, 1, 1)},$$

$$U_\alpha^C = \frac{U_\alpha^A}{U_\alpha^A(:, 1)}$$

6. Взвешенная сумма элементов:

$$S = \sum_{i_1,i_2,i_3} A(i_1,i_2,i_3)w_1(i_1)w_2(i_2)w_3(i_3).$$

Вычисление занимает $O(nr^3)$ операций.

Операции описаны в [26].

Все упомянутые свойства позволяют переписать алгоритм метода дискретных скоростей в виде последовательности операций с тензорами, представленными в некотором формате: поэлементные операции заменяются на их аналоги, и после каждой операции добавляется операция округления, чтобы предотвратить рост рангов. Целесообразность применения тензорных разложений можно обосновать следующими соображениями:

1. Во многих ячейках расчетной области функция распределения близка к локально-максвелловской, а у этой функции переменные разделяются (далее приведен упрощенный вид функции):

$$f_M(\xi_1,\xi_2,\xi_3) = \exp\left(-|\boldsymbol{\xi}-\boldsymbol{u}|^2\right) = e^{-(\xi_1-u_1)^2}e^{-(\xi_2-u_2)^2}e^{-(\xi_3-u_3)^2} \tag{20}$$

т.е. тензор из значений локально-максвелловской функции на скоростной сетке представим в любом тензорном формате с единичными рангами. Также локально-максвелловская функция – гладкая (C^∞). Есть теоретические оценки, показывающие что тензоры из значений гладких функций на равномерных сетках имееют малые ранги [2].

2. В начальный момент времени функции распределения представляют из себя некоторую сумму Максвеллианов, которая в дальнейшем «размазывается» по скоростному пространству. Отсюда можно сделать вывод, что, так как физические процессы не должны претерпевать изменений при увеличении скоростной сетки, то ранги тензоров r не будут расти с увеличением n. Так как объем памяти, занимаемый тензором, растёт линейно с n, то наиболее выгодно использовать разложения вместе с большими сетками.

7 Численный метод дискретных скоростей с использованием тензорного формата

7.1 Адаптация алгоритма

В алгоритм базового метода были внесены такие изменения:

1. Для вычисления тензора значений максвелловской функции на сетке, вычислялись только одномерные распределения, после чего конструировались тензоры с соответствующими ядрами.

2. После операций поэлементного сложения и умножения были вставлены вызовы процедуры округления с заданной точностью ϵ, чтобы предотвратить рост рангов.

3. Процедура вычисления суммы всех элементов была заменена на процедуру вычисления суммы тензоров

4. Т.к. тензоры $|\hat{\xi}_n|$, состоящие из модулей проекции скорости на нормаль к грани, приближаются с высокой точностью только тензорами с большими рангами (из-за негладкости), то в формуле для вычисления потоков вместо них были использованы тензоры, полученные путём принудительного округления их до максимального тензорного ранга 6 (10). Несмотря на то, что модуль, в принципе, плохо аппроксимируется, результат округления имеет довольно низкую относительную ошибку (3% в норме Фробениуса). Сравнение точного и округлённого тензоров показано на рисунке 2.

5. В алгоритме неявного метода присутствует операция поэлементного деления. Так как данная операция не реализована точно для тензоров произвольных рангов ни в одном из форматов, то диагональный тензор был заменён на некоторую одноранговую оценку сверху, основанную на неравенстве:

$$\left|\hat{\xi}_n[j]\right| \leq \sqrt{\xi_1^2 + \xi_2^2 + \xi_3^2}$$

Выражение справа представляет собой гладкую функцию относительно ξ_1, ξ_2, ξ_3, поэтому её проекция на равномерную сетку даст одноранговый тензор. Далее домножением на скаляр тензор подгоняется как можно ближе к $\hat{D}_j$ (3).

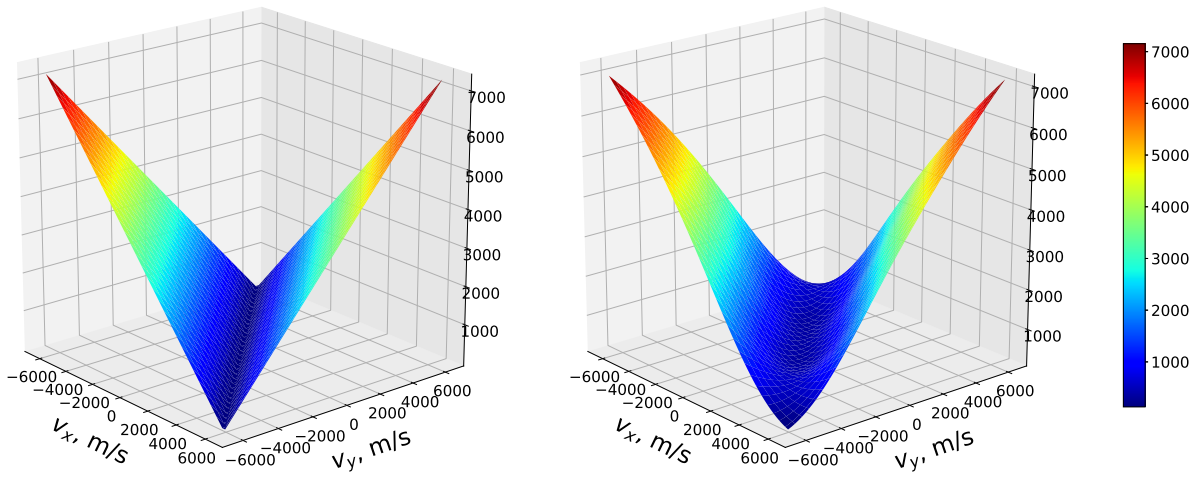

Рис. 2: Срезы модуля нормальной скорости, точного и оценки в формате Таккера максимального ранга 6

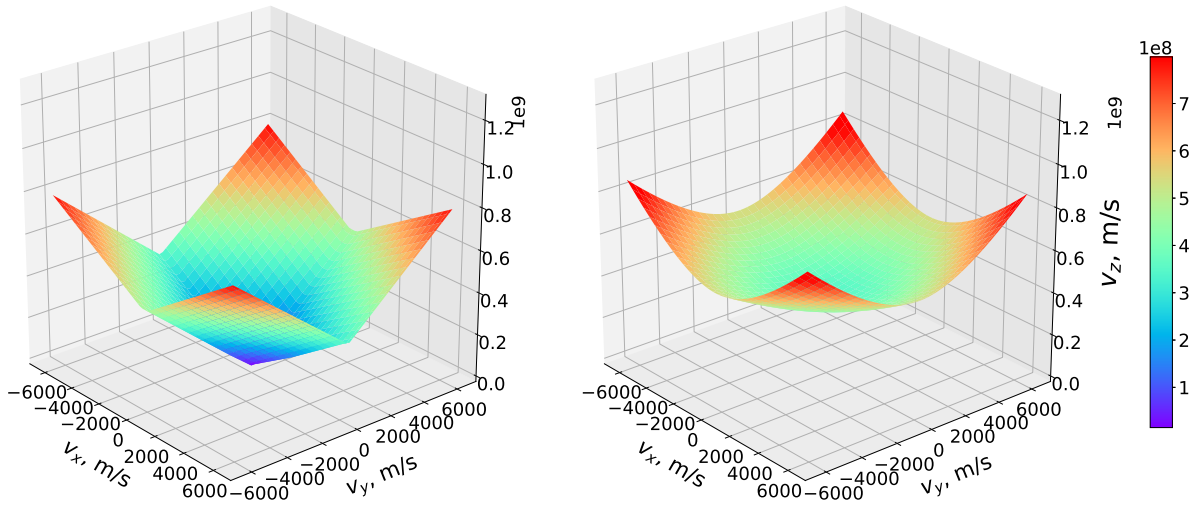

Рис. 3: Срезы диагональных тензоров, точного и одноранговой оценки в формате Таккера

7.2 Программная реализация

Для каждого из трёх форматов реализован свой решатель, при этом для тензоров в тензорных форматах перегружены операторы сложения, вычитания и умножения, поэтому алгоритм практически не претерпевает изменений. Код написан на языке Python 3.6 с использованием библиотек numpy для работы с массивами,
ttpy https://github.com/oseledets/ttpy для TT-формата и

tucker3d https://github.com/rakhuba/tucker3d для формата Таккера. Программа состоит из четырёх основных python-модулей:

1. *read_starcd.py* – модуль для работы с пространственной сеткой. В нём реализован класс *Mesh*. Он позоляет считать сетку из файла в формате StarCD, а также записать полученные результаты в формате файлов Tecplot для визуализации.

2. *solver.py* – определяет класс *Solution* для решения с полными тензорами и метод класса *make_time_steps*, который делает *nt* шагов решателя.

3. *solver_tt.py* – определяет версию метода с TT-тензорами.

4. *solver_tucker.py* – определяет версию метода с тензорами в формата Таккера.

Для того, чтобы запустить расчет, необходимо создать объект класса *Problem* и передать его на вход конструктору класса *Solution* вместе с объктами классов *Mesh* и *VelocityGrid*, который содержит тензоры для скоростной сетки.

Листинг 1: класс *Problem*

```
class Problem:
    def __init__(self, bc_type_list = None,
                 bc_data = None, f_init = None):
        # list of boundary conditions' types
        # according to order in starcd '.bnd' file
        # list of strings
        self.bc_type_list = bc_type_list
        # data for b.c.
        # list of lists
        self.bc_data = bc_data
        # Function to set initial condition
        self.f_init = f_init
```

Листинг 2: Начальные и граничные условия для задачи обтекания цилиндра

```
f_init = lambda x, y, z, v: tensor(
        solver.f_maxwell_t(
            v, n_in, u_in, 0., 0., T_in, gp.Rg))

f_bound = tensor(
        solver.f_maxwell_t(
            v, n_in, u_in, 0., 0., T_in, gp.Rg))

fmax = tensor(
        solver.f_maxwell_t(
                v, 1., 0., 0., 0., T_w, gp.Rg))

problem = solver.Problem(
        bc_type_list =
        ['sym-z', 'in', 'out', 'wall', 'sym-y'],
        bc_data = [[],
                   [f_bound],
                   [f_bound],
                   [fmax],
                   []], f_init = f_init)
```

Листинг 3: *Solution* class

```
class Solution:
    def __init__(self, gas_params, problem,
                                        mesh, v, config):
        # initialize all required tensors and initial
        # values of the solution
        ...
        if (config.init_type == 'default'):
            for i in range(mesh.nc):
                x = mesh.cell_center_coo[i, 0]
                y = mesh.cell_center_coo[i, 1]
                z = mesh.cell_center_coo[i, 2]
                self.f[i] = problem.f_init(x, y, z, v)
        elif (config.init_type == 'restart'):
            # restart from distribution function
            self.f = self.load_restart()

    def make_time_steps(self, config, nt):
        # perform nt time steps
        ...
```

8 Тестовые расчёты

Рассматривается задача обтекания цилиндра разреженным газом. Постановка задачи взята из [19]. Сравнение решения уравнения Больцмана с приближённым интегралом столкновений и точного уравнения Больцмана с решениями методом Монте-Карло приведено в [23, 12, 24]. Неструктурированная пространственная сетка показана на рисунке 4. Поток газа направлен по направлению оси x. На стенке цилиндра используется граничное условие стенка (6).

На внешней границе задаётся условие набегающий поток, на остальных – граничное условие симметрия.

Эта стационарная задача решается методом установления по времени, т.е. один шаг по времени можно интерпретировать как итерацию метода для решения стационарной задачи.

Набегающий поток описывается следующими параметрами: $v_0 = 2630$ м/с, $n_0 = 2{\cdot}10^{23}$ м$^{-3}$ $T_0 = 200\,K$. Температура на стенке $T_w = 5T_0$, радиус цилиндра $r = 1.35 \cdot 10^{-5}$ м. Число Кнудсена $Kn \approx 0.56$, число Маха $M = 10$. Для вязкости использовалась формула:

$$\mu(T) = \mu_0 \left(\frac{T}{T_0}\right)^{0.734}, \ \mu_0 = 1.61 \cdot 10^{-5}\ \text{Па} \cdot \text{с}, T_0 = 200\,\text{К}. \tag{21}$$

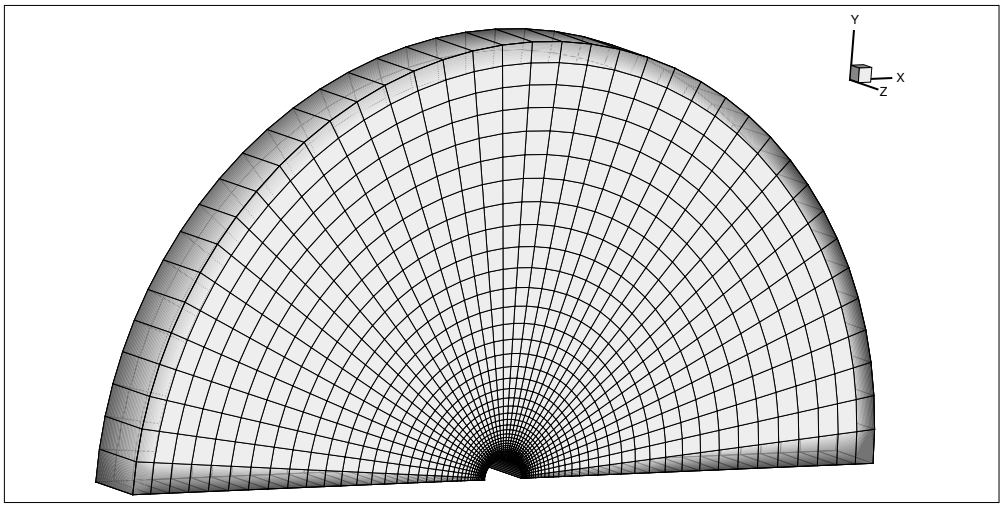

Рис. 4: Пространственная сетка для задачи обтекания цилиндра.

Равномерная скоростная сетка в границах $[-\xi_{max}, \xi_{max}]^3$ с $\xi_{max} \approx 6400$ м/с содержит $n_v = 64$ узлов по каждому направлению. Пространственная сетка содержит 1600 ячеек. Такие малые размеры сеток были обусловлены тем, что код явного метода нужно было запустить на процессоре с относительно небольшим количеством оперативной

памяти. Число Куранта вычислялось по формуле $CFL = \frac{\sqrt{3}\xi_{max}\tau}{\max\limits_{i=1}^{n_c}(diam_i)}$, где $diam_i$ – диаметр i-й пространственной ячейки. Для явного метода $CFL = 0.5$, для неявного $CFL = 50$.

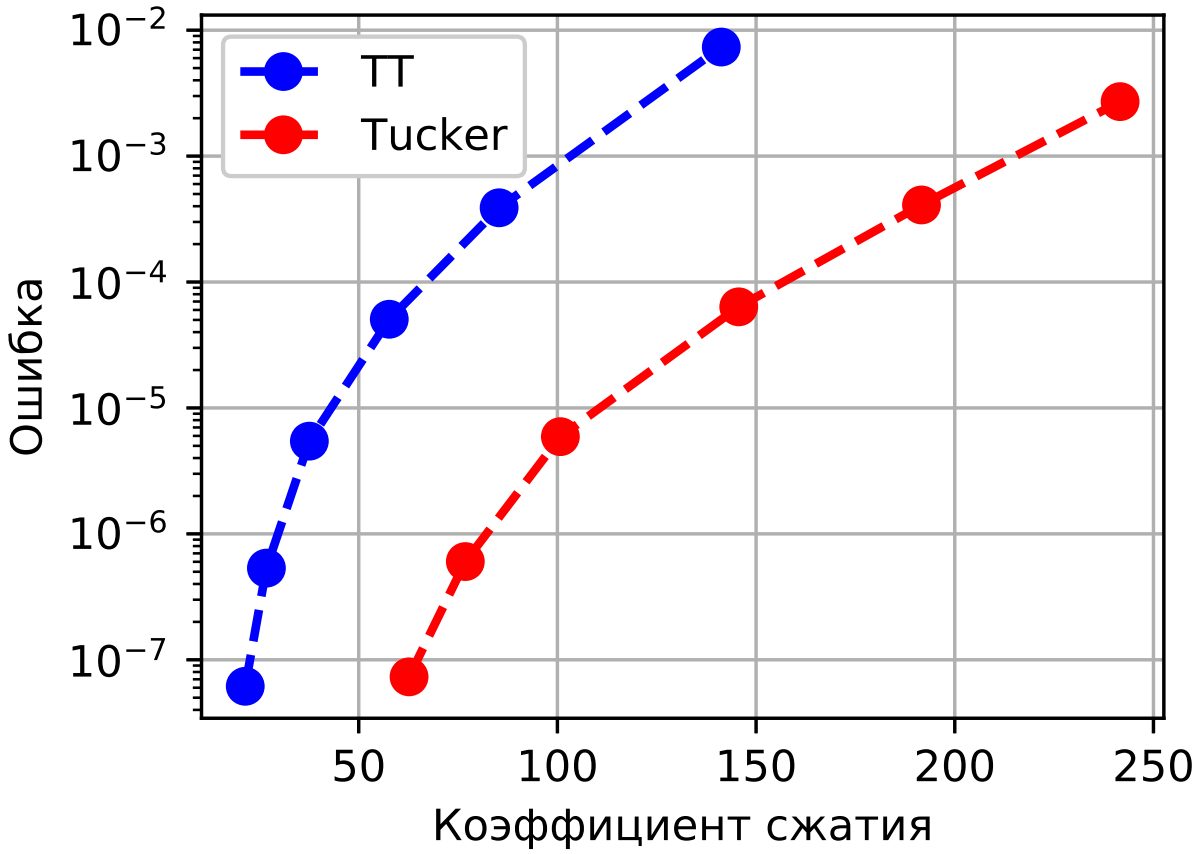

Рис. 5: Зависимость ошибки от сжатия при представлении функции распределения из расчета обтекания цилиндра

Заранее оценим, какой тензорный формат лучше подходит для метода. На графике 5 представлено сравнение формата Tensor-Train и формата Таккера. Для сравнения был взят некоторый трёхмерный тензор, к которому применялось сжатие (округление) с различными значениями параметра ϵ. На оси x отмечены коэффициенты сжатия, а на оси y – ошибки округления в норме Фробениуса. Коэффициент сжатия выражает отношение числа параметров, используемых в представлении сжатого тензора, к числу параметров в полном тензоре. Видно, что формат Таккера показывает лучшую эффективность в сжатии трёхмерных тензоров, как и предполагалось в разделе 6.

В таблице 1 представлены сравнения сошедшихся решений для различных параметров расчёта с базовым решением. Показаны отношения компьютерной памяти полного решения к сжатым и отношение корня из суммы квадратов разниц температур с базовым решением по ячейкам к корню из суммы квадратов температур базового решения. Видно, что формат Таккера предоставляет лучшее сжатие, а также лучше

вычисляет макропараметры для более точных порогов округления. Можно заметить, что точность вычисления макропараметров довольно медленно падает при экспоненциальном понижении точности округления. Предполагается, что это связано с тем, что тензоры $|\hat{\xi}_n|$, применяющиеся в вычислении потоков на гранях ячеек, были округлены до максимальных рангов 6, что привело к некоторой неустранимой ошибке на стенке цилиндра. Это также хорошо видно на рисунках 8 и 9.

Формат	ϵ	Сжатие	$\|\Delta T\|_F \,/\, \|T\|_F$
TT	10^{-3}	152.65	$5.5 \cdot 10^{-2}$
TT	10^{-4}	73.59	$3.3 \cdot 10^{-2}$
TT	10^{-5}	40.11	$2.9 \cdot 10^{-2}$
Tucker	10^{-3}	260.12	$5.9 \cdot 10^{-2}$
Tucker	10^{-4}	139.95	$2.3 \cdot 10^{-2}$
Tucker	10^{-5}	85.00	$2.0 \cdot 10^{-2}$

Таблица 1: Зависимость коэффициета сжатия и нормы суммы относительных разниц температур по ячейкам от относительной точности округления для разных тензорных форматов.

На рисунках 6 и 7 показаны сравнения результатов, полученных в базовом методе, и в методах с использованием тензорных разложений. 6 изображает сечения функции распределения в одной из ячеек, а 7 распределения температуры в плоскости $z = const$. Качественно результаты близки, но видно, что в расчете с тензорными разложениями немного изменились формы фронта головной ударной волны и значения пиков функции распределения.

На рисунках 8, 9 показаны одномерные сечения температур вдоль нормалей к цилиндру под углами 45 и 0 градусов соответственно (угол отсчитывался от линии торможения). Видно, что с уменьшением относительной ошибки округления значения температуры приближаются к базовому решению, но вплотную к стенке имеется ошибка, которая не уменьшается. Это может быть связано с тем, что в потоках используются неточная оценка модуля.

Рисунок 10 отражает отношения размера сжатых функций распределения к полным в каждой из ячеек для обоих тензорных форматов. Видно, что ранги тем больше, чем ближе расположение ячейки к цилиндру. Это легко объясняется тем, что область выбиралась с расчётом, что в отдалении от цилиндра будет сохраняться свободный по-

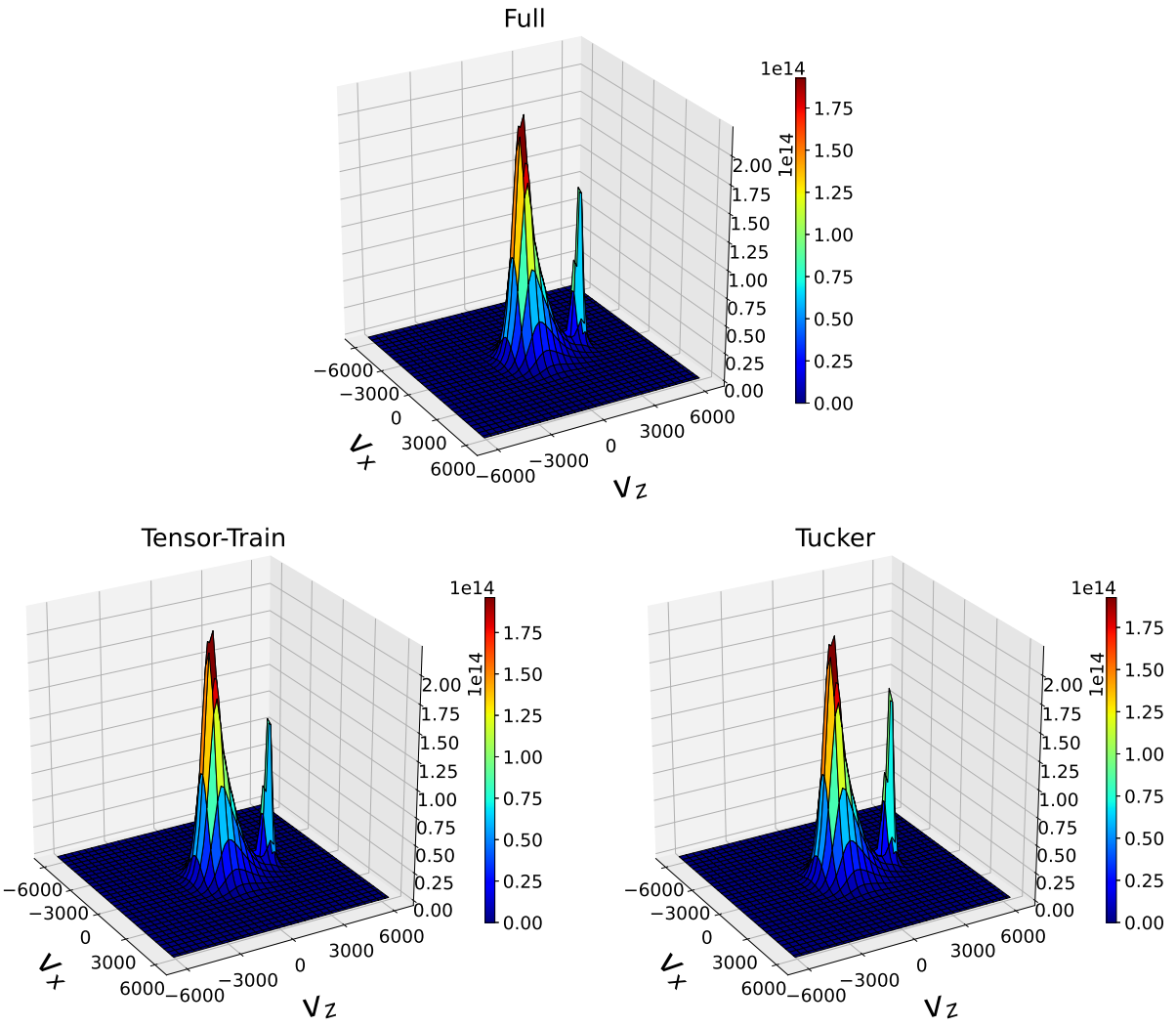

Рис. 6: Сечение функции распределения для различных тензорных разложений. Сверху – расчет базовым методом без разложений, снизу слева – расчёт с тензорами в TT-формате, снизу справа – в формате Таккера (сжатые тензоры были развёрнуты в полные).

ток, который хорошо приближается малоранговыми тензорами. Так как основные ранги сосредоточены в районе цилиндра и ударной волны, то вычисления также будут связаны с этими областями, что позволяет сделать вывод, что увеличение расчётной сетки в сторону увеличения области не должно привести к сильному усложнению вычислений.

Рисунок 11 отображает падения норм невязок различных методов при различных порогах округления. В качестве начального условия в каждую ячейку пространства помещался Максвеллиан с макропараметрами точного решения в этой ячейке. Видно, что формат Таккера сходится к стационарному решению быстрее. Также видно, как путём подбора порога округления можно подбирать баланс между точностью решения и скоростью сходимости.

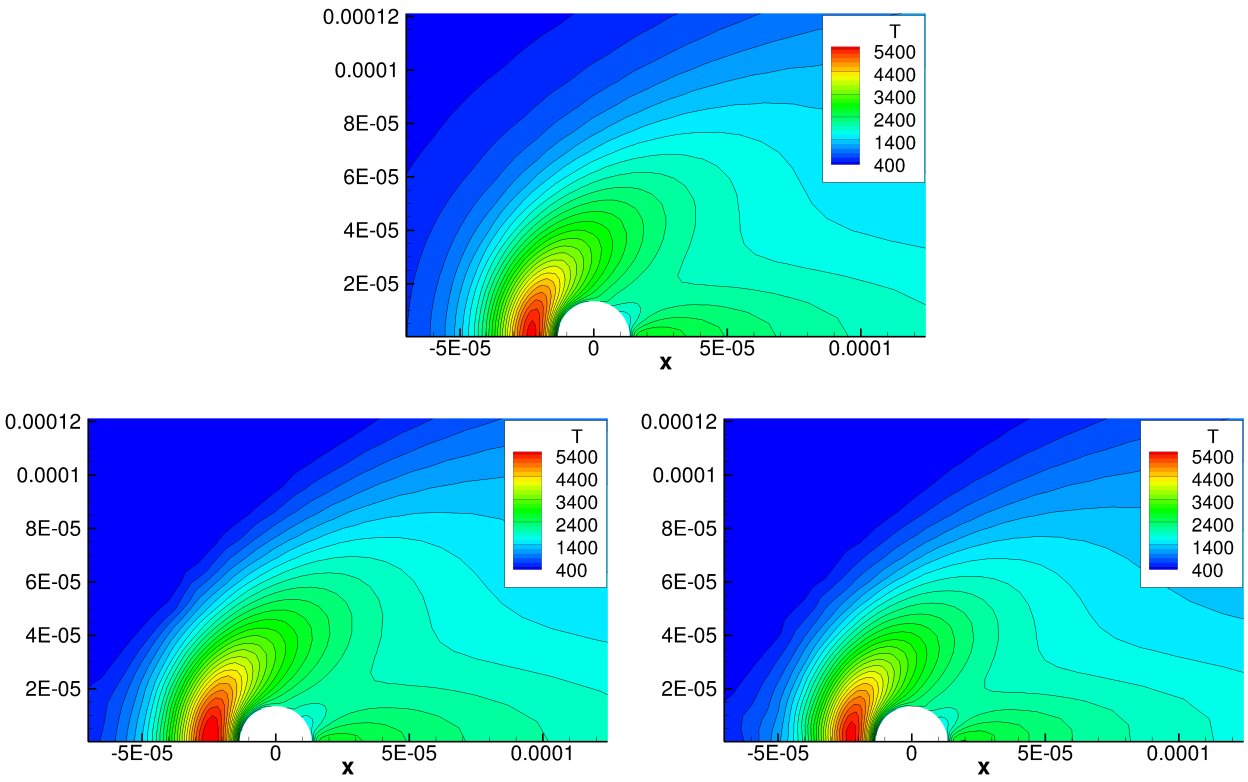

Рис. 7: Распределение температуры в плоскости $z = const$. Сверху – расчет базовым методом без разложений, снизу слева – расчёт с тензорами в TT-формате, снизу справа – в формате Таккера.

Отметим, что, несмотря на большой выигрыш по количеству памяти, фактическое время выполнения программ до сходимости было довольно большим и превышало время базового метода в 2-5 раз. При этом время выполнения одного шага по времени первые несколько шагов было небольшим (начальное условие состояло из одноранговых тензоров), но со временем ранги накапливались, что приводило к замедлению. Это можно связать с использованием языка Python и тем, что операции с тензорами не были так хорошо оптимизированы, как обычные операции с массивами. Поэтому для более точной оценки времени расчета требуется более эффективная программная реализация алгоритма.

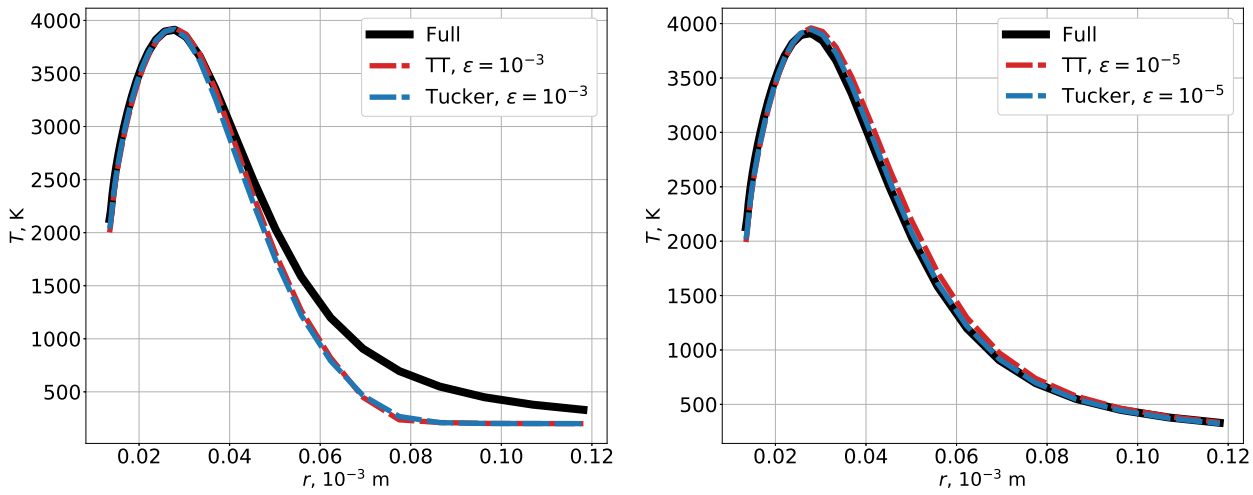

Рис. 8: Сечения температур вдоль нормали к поверхности цилиндра под углом 45 градусов при различных порогах округления ϵ.

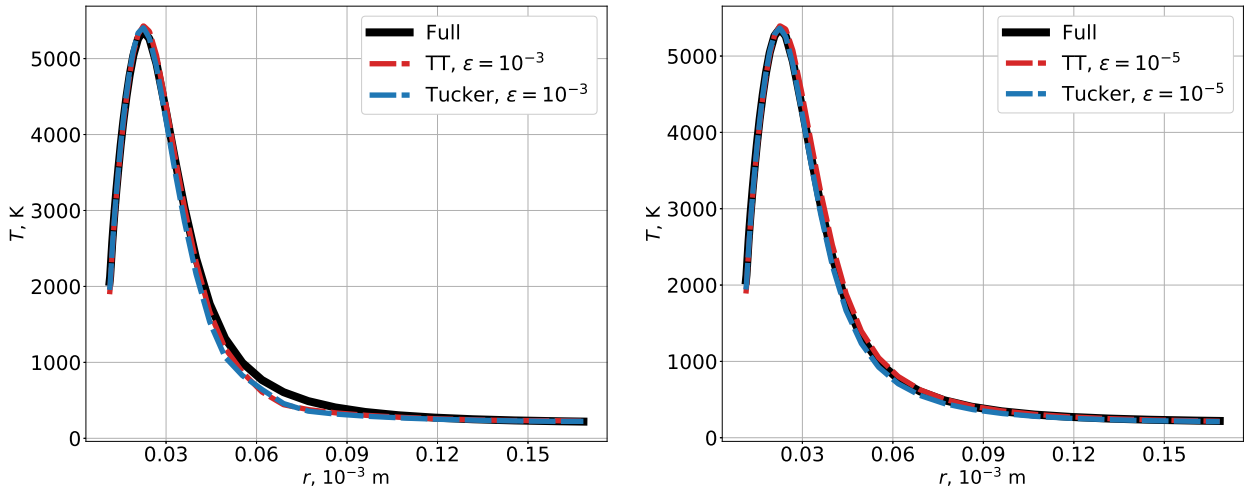

Рис. 9: Сечения температур вдоль нормали к поверхности цилиндра под углом 0 градусов при различных порогах округления ϵ.

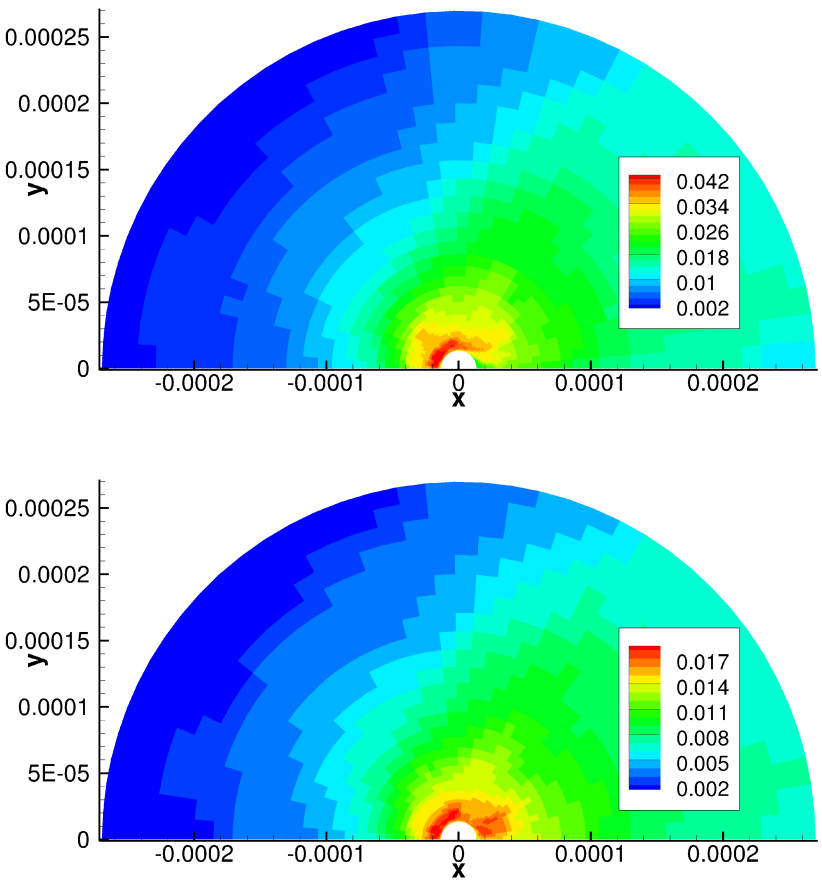

Рис. 10: Распределения отношений размеров сжатых и полных тензоров для TT-разложения (сверху) и разложения Таккера (снизу). Относительная точность округления в обоих случаях $\epsilon = 10^{-5}$.

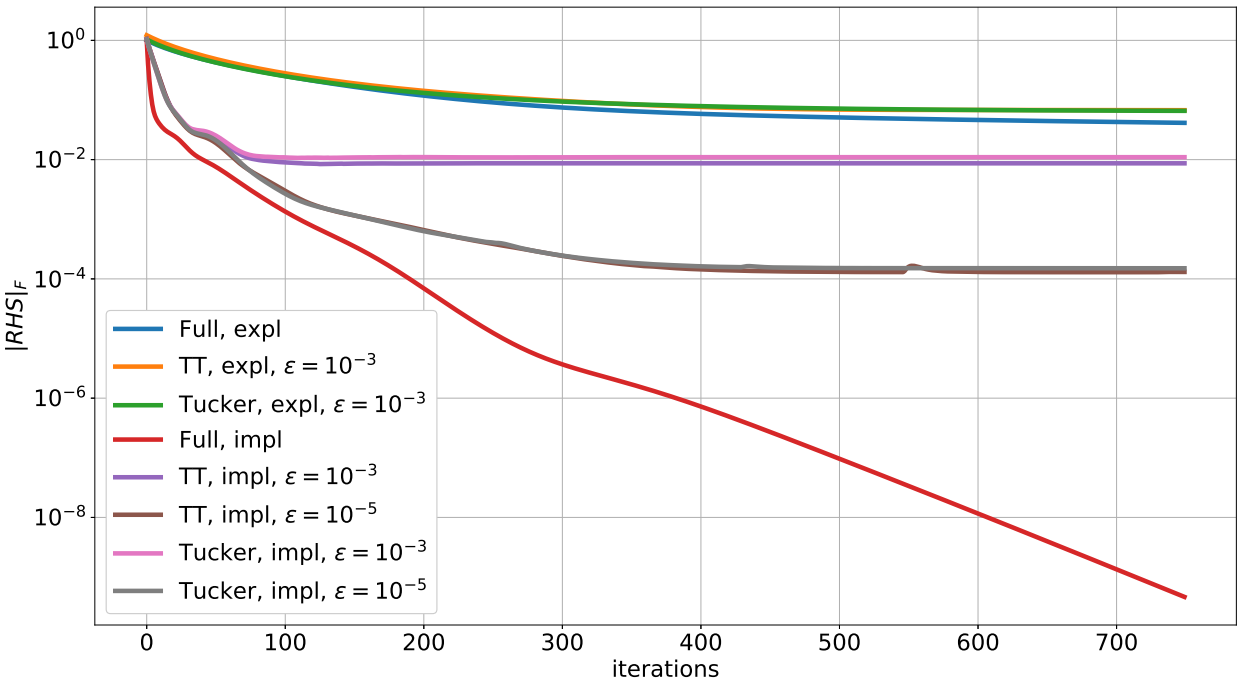

Рис. 11: Скорости сходимости неявных методов при различных порогах округления.

9 Заключение

Результаты сделанного исследования показали, что алгоритм метода дискретных скоростей может быть легко адаптирован для применения различных тензорных разложений. Для этого необходимо заменить процедуры вычисления поэлементной суммы, произведения полных тензоров, а также вычисления суммы элементов тензора на аналогичные процедуры в конкретном тензорном формате, и добавить округление, чтобы ранги не накапливались.

При адаптации алгоритма могут возникнуть трудности с аппроксимацией некоторых негладких тензоров, например модулей, возникающих при вычислении потоков на гранях и диагональных тензоров в LU-SGS (неявном методе). В обоих случаях применяются некоторые аппроксимации точных тензоров, что замедляет скорость сходимости, но не влияет на конечный результат.

Также из-за того, что операции поэлементного умножения и округления требуют довольно много вычислений, разработанный алгоритм будет давать выигрыш по числу элементарных операций только если возникающие ранги много меньше числа узлов скоростной сетки по одному измерению. Однако мы предполагаем, что увеличение скоростной сетки не будет приводить к сильному повышению рангов, так как сложность функции распределения зависит в первую очередь от физических процессов, а не от размера скоростной сетки. Это в теории позволит использовать сколь угодно большие скоростные сетки.

Также плюсом метода является то, что в областях малого градиента макропараметров будет сосредоточено относительно мало вычислений. Это также позволит не задумываться об избыточности в том числе и пространственных сеток.

Таким образом, основным достоинством метода является вычислительная адаптивность.

На основании результатов работы можно предложить следующие направления дальнейших исследований:

1. Провести вычислительные эксперименты со скоростными сетками большего размера для гиперзвуковых задач внешнего обтекания. Оценить выигрыш по памяти и по времени счета по сравнению с базовым методом.

2. Применить кросс-аппроксимацию для получения нового алгоритма для одного шага по времени.

3. Добавить возможность использования нелинейных ограничителей потоков в алгоритме метода с тензорными форматами. Это позволит получить схему 2-го порядка аппроксимации и повысить точность аппроксимации по пространству.

4. Переписать код на более компилируемом языке программирования, так как Python не очень хорошо подходит для эффективной параллельной реализации разработанного метода.

5. Разработать параллельную версию алгоритма для систем с общей памятью на основе разбиения пространственной сетки на блоки.

Список литературы

[1] C. Baranger, J. Claudel, N. Hérouard, and L. Mieussens. Locally refined discrete velocity grids for stationary rarefied flow simulations. *J. Comput. Phys.*, 257(PA):572–593, January 2014.

[2] Gregory Beylkin and Martin J. Mohlenkamp. Numerical operator calculus in higher dimensions. *Proceedings of the National Academy of Sciences*, 99(16):10246–10251, 2002.

[3] Arnout M.P. Boelens, Daniele Venturi, and Daniel M. Tartakovsky. Tensor methods for the boltzmann-bgk equation. *Journal of Computational Physics*, 421:109744, 2020.

[4] A. Chikitkin, M. Petrov, V. Titarev, and S. Utyuzhnikov. Parallel versions of implicit lu-sgs method. *Lobachevskii Journal of Mathematics*, 39(4):503–512, 2018.

[5] Aleksandr V. Chikitkin and Egor K. Kornev. Different approaches to numerical solution of the boltzmann equation with model collision integral using tensor decompositions. In Margarita N. Favorskaya, Alena V. Favorskaya, Igor B. Petrov, and Lakhmi C. Jain, editors, *Smart Modelling for Engineering Systems*, pages 105–116, Singapore, 2021. Springer Singapore.

[6] A.V. Chikitkin, E.K. Kornev, and V.A. Titarev. Numerical solution of the boltzmann equation with s-model collision integral using tensor decompositions. *Computer Physics Communications*, 264:107954, 2021.

[7] Youngsoo Choi, Peter Brown, William Arrighi, Robert Anderson, and Kevin Huynh. Space–time reduced order model for large-scale linear dynamical systems with application to boltzmann transport problems. *Journal of Computational Physics*, 424:109845, 2021.

[8] L. De Lathauwer, B. De Moor, and J. Vandewalle. On the best rank-1 and rank-(r1, r2, . . . , rn) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000.

[9] S.V. Dolgov, A.P. Smirnov, and E.E. Tyrtyshnikov. Low-rank approximation in the numerical modeling of the farley-buneman instability in ionospheric plasma. *Journal of Computational Physics*, 263:268–282, 2014.

[10] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, Sep 1936.

[11] Lukas Einkemmer, Jingwei Hu, and Lexing Ying. An efficient dynamical low-rank algorithm for the boltzmann-bgk equation close to the compressible viscous flow regime. 01 2021.

[12] A.A. Frolova and V.A. Titarev. Recent progress on supercomputer modelling of high-speed rarefied gas flows using kinetic equations. *Supercomputing frontiers and innovations*, 5(3):117–121, 2018.

[13] L. Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM Journal on Matrix Analysis and Applications*, 31(4):2029–2054, 2009.

[14] Lars Grasedyck, Daniel Kressner, and Christine Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitteilungen*, 36(1):53–78, 2013.

[15] W. Guo and Y. Cheng. A sparse grid discontinuous galerkin method for high-dimensional transport equations and its application to kinetic simulations. *SIAM Journal on Scientific Computing*, 38(6):A3381–A3409, 2016.

[16] B Khoromskij. Structured data-sparse approximation to high order tensors arising from the deterministic boltzmann equation. *Math. Comput.*, 76:1291–1315, 07 2007.

[17] V.I. Kolobov and R.R. Arslanbekov. Towards adaptive kinetic-fluid simulations of weakly ionized plasmas. *Journal of Computational Physics*, 231(3):839 – 869, 2012. Special Issue: Computational Plasma Physics.

[18] Egor Kornev and Aleksandr Chikitkin. A tensorized version of lu-sgs solver for discrete velocity method for boltzmann kinetic equation with model collision integral. *AIP Conference Proceedings*, 2312(1):050009, 2020.

[19] A.J. Lofthouse, L.C. Scalabrin, and I.D. Boyd. Velocity slip and temperature jump in hypersonic aerothermodynamics. *Journal of Thermophysics and Heat Transfer*, 22(1):38–49, 2008.

[20] I.V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[21] V.A. Titarev. Efficient deterministic modelling of three-dimensional rarefied gas flows. *Commun. Comput. Phys.*, 12(1):161–192, 2012.

[22] V.A. Titarev. Application of model kinetic equations to hypersonic rarefied gas flows. *Computers and Fluids*, 169:62 – 70, 2018.

[23] VA Titarev. Application of model kinetic equations to hypersonic rarefied gas flows. *Computers & Fluids*, 169:62–70, 2018.

[24] V.A. Titarev and A.A. Frolova. Application of model kinetic equations to computing of super- and hypersonic flows of molecular gas. *Fluid Dynamics*, 53(4):536–551, 2018.

[25] V.A. Titarev, S.V. Utyuzhnikov, and A.V. Chikitkin. Openmp + mpi parallel implementation of a numerical method for solving a kinetic equation. *Computational Mathematics and Mathematical Physics*, 56(11):1919–1928, 2016.

[26] Ledyard R Tucker. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change*, 15:122–137, 1963.

[27] Yajun Zhu and Kun Xu. The first decade of unified gas kinetic scheme. 02 2021.

[28] Титарев Владимир Александрович. Численное моделирование пространственных течений разреженного газа с использованием суперЭВМ. *Федеральный исследовательский центр "Информатика и управление" Российской Академии Наук*.

[29] Д. А. Стефонишин, С. А. Матвеев, А. П. Смирнов, and Е. Е. Тыртышников. Тензорные разложения для решения уравнений математических моделей агрегации, допускающих многочастичные столкновения. *Вычислительные методы и программирование*, 19(4):390–404, 2018.

[30] А. А. Фролова. Численное сравнение решений кинетических модельных уравнений. *Математика и математическое моделирование*, 6:61 – 77, 2015.