Федеральное государственное бюджетное учреждение науки
Институт прикладной математики им. М. В. Келдыша РАН

На правах рукописи
УДК 519.688

ГОРОБЕЦ АНДРЕЙ ВЛАДИМИРОВИЧ

ПАРАЛЛЕЛЬНЫЕ ТЕХНОЛОГИИ МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ ТУРБУЛЕНТНЫХ ТЕЧЕНИЙ НА СОВРЕМЕННЫХ СУПЕРКОМПЬЮТЕРАХ

Специальность 05.13.18 — математическое моделирование, численные методы и комплексы программ

Диссертация на соискание учёной степени
доктора физико-математических наук

Москва — 2015

Содержание

Введение

Актуальность работы

Научный прогресс в области математической физики тесно связан с возможностью эффективного использования современных вычислительных систем. Суперкомпьютерное моделирование широко применяется при решении актуальных инженерно-технических задач в различных высокотехнологичных отраслях промышленности, таких как авиастроение и авиационное двигателестроение, автомобилестроение, возобновляемая и атомная энергетика. Моделирование турбулентных течений, актуальное в отраслях, имеющих дело с аэродинамикой, аэроакустикой, гидродинамикой, тепломассопереносом, является одним из наиболее сложных и ресурсоёмких типов вычислительного эксперимента.

Непрерывный рост производительности суперкомпьютеров открывает все более широкие возможности перед вычислительным экспериментом. Уже существуют системы, обладающие пиковой производительностью в несколько десятков PFLOPS (1 PFLOPS = 10^{15} вычислительных операций в секунду). Но если в недалеком прошлом рост производительности поддерживался в основном за счёт увеличения числа процессоров в системе, то теперь рост сопряжён существенным усложнением архитектур, программных моделей и их разнообразием. Это делает создание расчётных кодов, в полной мере использующих возможности современных вычислительных, систем сложной и актуальной научной проблемой.

В настоящее время эволюция, с одной стороны, продолжает идти в сторону увеличения числа процессорных ядер, которое в крупнейших системах уже перевалило за миллион. Это обусловлено не только увеличением числа узлов в суперкомпьютерах, но и ростом числа ядер в процессорах, объединённых общей памятью узла. Последнее потребовало существенной перестройки распараллеливания и перехода на более сложную параллельную модель, сочетающую распределённую и общую память. Кроме того, рост пиковой производительности процессорного ядра теперь обеспечивается не за счёт повышения тактовой частоты, а в основном за счёт расширения векторных регистров, что, с учётом не столь значительного роста пропускной способности памяти, все более усложняет эффективное использование процессоров.

С другой стороны, рост производительности достигается за счёт использования массивно-параллельных ускорителей. К таким ускорителям относятся, в частности, графические процессоры GPU (Graphics Processing Unit) и ускорители Intel Xeon Phi существенно-многоядерной архитектуры MIC (Many Integrated Core). Среди десяти самых мощных

суперкомпьютеров мира уже можно видеть несколько систем такой гибридной архитектуры (Tianhe-2, Tianhe-1, Cray Titan, Intel Stampede). Гетерогенные вычисления, в частности, с использованием графических процессоров, сами по себе являются новой областью, появившейся всего несколько лет назад. Средства программирования и технологии вычислений находятся в настоящее время в активном развитии, постоянно дополняется функциональность и возможности низкоуровневых интерфейсов программирования CUDA, OpenCL, и высокоуровневых директивных средств OpenACC и OpenMP 4.0. Несмотря на все возрастающую популярность гетерогенных вычислений, развитие газодинамических алгоритмов, ориентированных на их использование, все ещё далеко от стадии зрелости, активно ведутся исследования в этой области. Таким образом, разработка параллельных алгоритмов, которые могут одновременно удовлетворять всё возрастающим требованиям по степени параллелизма и масштабируемости, и быть адаптированными к ещё более сложной параллельной модели, является крайне актуальной научной проблемой. Также важной и актуальной представляется разработка параллельных комплексов программ, позволяющих проводить расчёты широкого круга задач газовой динамики, аэроакустики, тепломассопереноса с использованием десятков тысяч процессорных ядер и различных типов ускорителей, включая архитектуры Intel, AMD и NVIDIA. Только такой подход на основе эффективных алгоритмов и программных комплексов, отвечающих современным тенденциям в эволюции суперкомпьютеров, может позволить достичь прогресса в математическом моделировании турбулентных течений.

Актуальность крупномасштабных расчётов фундаментальных задач обусловлена, в частности, необходимостью расширения набора эталонных численных решений для валидации моделей турбулентности, активно разрабатываемых во всем мире. Результаты рекордных расчётов позволяют получить новые данные о физике турбулентного течения и продвинуться в исследованиях этого сложного и до сих пор малоизученного явления.

Цели и задачи диссертационной работы

Связанные между собой общие цели диссертационной работы (рис. 1) заключаются в следующем.

- Создание нового подхода к математическому моделированию турбулентных течений, удовлетворяющего современным тенденциям в эволюции вычислительной техники. Подход должен охватывать все стадии моделирования, включая технологии разработки алгоритма в рамках многоуровневой параллельной модели, создания гетерогенной программной реализации для гибридных суперкомпьютеров, выполнения больших нестационарных расчётов.

- Создание новых параллельных алгоритмов повышенной точности и их реализующих программных комплексов для моделирования сжимаемых и несжимаемых турбулентных течений. Ключевым требованием является возможность эффективно задействовать десятки

тысяч процессоров, а также гибридные суперкомпьютеры с массивно-параллельными ускорителями.

– Численное исследование сложных физических процессов, связанных с турбулентными течениями, создаваемыми ими акустическими полями и явлениями тепломассопереноса. Пополнение набора эталонных численных решений для разработки и валидации моделей турбулентности.

Рисунок 1: Структура целей диссертационной работы

Для достижения поставленных целей необходимо решить следующие задачи.

– Разработать технологию разработки параллельных алгоритмов с многоуровневым распараллеливанием, эффективных на широком спектре вычислительных систем от рабочих станций до крупных суперкомпьютеров.

– Разработать технологию программной реализации параллельных алгоритмов для численного моделирования турбулентных течений на системах с массивно-параллельными ускорителями различной архитектуры.

– Разработать технологию для эффективного выполнения крупномасштабных нестационарных расчётов турбулентных течений на суперкомпьютерах.

– Создать по разработанной технологии параллельный алгоритм повышенной точности на неструктурированных сетках для моделирования сжимаемых турбулентных течений и явлений аэроакустики.

– Создать программный комплекс для расчётов сжимаемых турбулентных течений с использованием десятков тысяч процессоров и гибридных систем с массивно-параллельными ускорителями.

– Разработать новый параллельный метод решения уравнения Пуассона, эффективный для моделирования несжимаемых турбулентных течений на системах с десятками тысяч процессоров и на гибридных суперкомпьютерах.

– Создать на основе разработанного метода программный комплекс для крупномасштабных расчётов несжимаемых турбулентных течений.

– Выполнить крупномасштабные, в том числе рекордные, расчёты сжимаемых и несжимаемых турбулентных течений, получить репрезентативный набор данных для валидации моделей турбулентности.

Научная новизна

– Автором разработана новая универсальная параллельная технология математического моделирования турбулентных течений на современных суперкомпьютерах. В состав технологии входит, в частности, новый параллельный алгоритм построения схемы обменов данными для схем повышенного порядка точности на неструктурированных сетках, новая методика эффективной разработки гетерогенной реализации.

– Предложен новый алгоритм с многоуровневым распараллеливанием для расчётов сжимаемых течений на основе экономичных EBR схем повышенного порядка на неструктурированных сетках (Abalakin I.V., Bakhvalov P.A., Kozubskaya T.K. Edge-based reconstruction schemes for prediction of near field flow region in complex aeroacoustic problems // Int. J. Aeroacoust. 2014. V.13. p. 207-234). Параллельный алгоритм рассчитан на системы с сотнями тысяч ядер и адаптирован к использованию гибридных суперкомпьютеров с ускорителями Intel Xeon Phi.

– Представлен новый комплекс программ NOISEtte для математического моделирования сжимаемых турбулентных течений методами повышенного порядка точности на неструктурированных сетках. В нём впервые создана параллельная реализация экономичных численных схем семейства EBR.

– Предложен оригинальный параллельный метод решения уравнения Пуассона на гибридных системах с массивно-параллельными ускорителями. Новый метод за счёт сочетания прямых и итерационных методов решения СЛАУ наиболее эффективен для моделирования несжимаемых турбулентных течений в задачах с одним периодическим направлением. Метод также может применяться для расчётов трёхмерных задач без периодического направления, благодаря новому расширению на основе многосеточного

подхода. Для задач с пространственной симметрией предложен новый способ расширения границ масштабирования.

– На основе этого метода создан новый параллельный алгоритм и реализующий его программный комплекс для математического моделирования несжимаемых турбулентных течений на традиционных суперкомпьютерах с числом процессоров порядка сотни тысяч и на гибридных суперкомпьютерах с различными типами ускорителей (GPU NVIDIA, GPU AMD, Intel Xeon Phi)

– Выполнено численное исследование механизмов генерации шума от турбулентного следа и взаимодействия турбулентных структур с твёрдым телом на задаче об обтекании тандема квадратных цилиндров. Получены новые данные о применимости гибридных RANS-LES подходов на неструктурированных сетках к математическому моделированию аэродинамического шума и о качестве численного решения в сравнении с экспериментальными данными.

– Выполнена серия DNS расчётов, в том числе рекордных, моделированию турбулентных течений при естественной конвекции. Получены новые эталонные решения и данные о физике течения для различных чисел Рэлея. Использовались схемы повышенного порядка точности и сетки с числом узлов до 600 миллионов.

– Выполнена серия DNS расчётов, в том числе рекордных, по задачам вынужденной конвекции. Использовались схемы повышенного порядка точности и сетки с числом узлов до 320 миллионов. Получены новые научные результаты для течения вокруг бесконечного квадратного цилиндра и для падающей струи в канале, формирующие базис для валидации и исследований в области разработки перспективных моделей турбулентности.

Теоретическая и практическая значимость работы

Теоретическая ценность работы заключается в исследовании различных моделей параллельных вычислений, включая модели с распределённой и общей памятью, с одиночным и множественными потоками команд, применительно к различным типам конечно-объёмных (и конечно-разностных) алгоритмов повышенной точности. Исследованы различные подходы к программной реализации и повышению эффективности вычислений. Численно исследованы различные типы фундаментальных задач по моделированию турбулентных течений, решение которых вносит в клад исследование таких сложных физических явлений, как турбулентность, генерация шума, тепломассоперенос.

Практическую ценность представляют параллельные алгоритмы и комплексы программ для крупномасштабных суперкомпьютерных расчётов широкого круга задач газовой динамики. Реализованные программные средства позволяют задействовать десятки тысяч процессоров и массивно-параллельные ускорители различных архитектур. Полученные результаты прямого

численного моделирования различных задач представляют практическую ценность для разработки моделей и подходов к моделированию турбулентности, как для сжимаемых, так и для несжимаемых течений.

Методы исследования

В данной работе вычислительный эксперимент является методом исследования турбулентного течения. Для численного исследования используются конечно-объёмные и конечно-разностные численные методы повышенного порядка аппроксимации. Для моделирования турбулентности используются нестационарные вихреразрешающие подходы LES (Large eddie simulation - моделирование крупных вихрей), семейство гибридных подходов DES (Detached eddie simulation - моделирование отсоединённых вихрей), и метод прямого численного моделирования DNS (Direct numerical simulation - прямое численное моделирование). Параллельные алгоритмы и программные комплексы, реализующие численные методы, основываются на объектно-ориентированном подходе и многоуровневой параллельной модели, сочетающей различные типы параллелизма.

Основные положения, выносимые на защиту

- Многоуровневый параллельный алгоритм расчёта сжимаемых турбулентных течений с учётом явлений аэроакустики.
- Параллельный программный комплекс для крупномасштабных суперкомпьютерных расчётов сжимаемых турбулентных течений.
- Численный метод решения уравнения Пуассона и его реализующий параллельный алгоритм с многоуровневым распараллеливанием для моделирования несжимаемых течений на гибридных системах с массивно-параллельными ускорителями.
- Параллельный программный комплекс для крупномасштабных нестационарных расчётов несжимаемых турбулентных течений.
- Результаты вычислительных экспериментов по аэродинамическому шуму от турбулентного следа и от взаимодействия турбулентности с твёрдым телом.
- Результаты расчётов ряда фундаментальных задач по моделированию несжимаемых турбулентных течений и тепломассопереноса при естественной конвекции.
- Результаты расчётов ряда фундаментальных задач по моделированию несжимаемых турбулентных течений при вынужденной конвекции.

Достоверность результатов

Разработанные параллельные комплексы программ надежно и тщательно верифицированы на широком круге задач. Выполнено сравнение с аналитическим решением на модельных задачах, сравнение с экспериментальными данными, сравнение с численными результатами других авторов. Корректность реализации и порядки сходимости дискретных операторов подтверждены, в том числе, на основе широко известного метода MMS (Method of Manufactured Solutions). Эффективность и производительность параллельных вычислений подтверждается серией тестов, выполненных на многопроцессорных системах различных архитектур с использованием до 24000 процессоров.

Апробация работы

Результаты, входящие в данную диссертационную работу, были представлены в 70-ти докладах на конференциях (более 50 международных). В том числе в докладах:

1. А.В. Горобец, С.А. Суков, П.Б. Богданов, Ф.Х. Триас, Конечно-объёмные алгоритмы для моделирования турбулентных течений на гибридных суперкомпьютерах различной архитектуры, XV международная конференция "Супервычисления и математическое моделирование 13-17 октября 2014, г. Саров.

2. Gorobets, F.X. Trias and A. Oliva, Fighting against massively parallel accelerators of various architectures for the efficiency of finite-volume parallel CFD codes, 26th Parallel CFD, 2014, 20-22 of May, Trondheim, Norway.

3. Sergey Soukov, Andrey Gorobets and Pavel Bogdanov, OpenCL Implementation of Basic Operations for a High-order Finite-volume Polynomial Scheme on Unstructured Hybrid Meshes, Parallel CFD, 2013, May 20-24, Changsha, China.

4. Andrey Gorobets, Francesc Xavier Trias Miquel and Assensi Oliva, An OpenCL-based Parallel CFD Code for Simulations on Hybrid Systems with Massively-parallel Accelerators, Parallel CFD, 2013, May 20-24, Changsha, China.

5. И.А. Абалакин, П.А. Бахвалов, А.В. Горобец, А.П. Дубень, Т.К. Козубская, Комплекс программ NOISETTE для моделирования задач аэродинамики и аэроакустики, XXIV Научно-техническая конференция по аэродинамике, п. Володарского, 28 февраля - 1 марта 2013 г.

6. А. В. Горобец, С. А. Суков, На пути к освоению гетерогенных супервычислений в газовой динамике, Первый Национальный Суперкомпьютерный Форум (НСКФ-2012), Россия, Переславль-Залесский, ИПС имени А.К. Айламазяна РАН, 29-30 ноября 2012 года.

7. Andrey Gorobets, F. Xavier Trias, Assensi Oliva, A parallel OpenCL-based solver for large-scale DNS of incompressible flows on heterogenous systems, Parallel CFD 2012, Atlanta, USA, May 2012.

8. Gorobets, F. X. Trias, R. Borrell, M. Soria and A. Oliva, Hybrid MPI+OpenMP parallelization of an FFT-based 3D Poisson solver that can reach 100000 CPU cores, Parallel CFD 2011, Barcelona, Spain, 16-20 May 2011.

9. V. Gorobets, S. A. Soukov, P. B. Bogdanov, A. O. Zheleznyakov and B. N. Chetverushkin, Extension with OpenCL of the two-level MPI+OpenMP parallelization for large-scale CFD simulations on heterogeneous systems, Parallel CFD 2011, Barcelona, Spain, 16-20 May 2011.

10. А.В.Горобец, С.А.Суков, А.О.Железняков, П.Б.Богданов, Б.Н.Четверушкин, Применение GPU в рамках гибридного двухуровневого распараллеливания MPI+OpenMP на гетерогенных вычислительных системах, Параллельные вычислительные технологии (ПаВТ) 2011, 28 марта - 1 апреля, Москва.

Реализация и внедрение результатов работы

Работа выполнялась в рамках научных планов ИПМ им. М. В. Келдыша РАН. Работа поддерживалась грантами Российского фонда фундаментальных исследований, проектами Министерства образования и науки РФ, проектами совета по грантам при президенте РФ. Методики, методы, алгоритмы, программные средства, результаты расчётов, представляемые в работе к защите, использовались в проектах, совместных научных исследованиях и договорных работах со следующими организациями: ЦАГИ, ЦНИИМаш, ОАО «Авиадвигатель», ОАО «Камов», ОАО «ОКБ Сухого», МФТИ, РФЯЦ-ВНИИЭФ, НИИСИ РАН, ИБРАЭ РАН, Санкт-Петербургский государственный политехнический университет, Технический университет Каталонии (Испания), Университет Гронингена (Нидерланды), Termo Fluids S.L. (Испания), а также в проектах РНФ, Минобрнауки и в проекте 7-й рамочной программы Евросоюза VALIANT.

Основные публикации

По теме диссертации в печати опубликовано 28 работ в журналах, рекомендованных ВАК для опубликования научных результатов докторских диссертаций, включая 21 статью в международных журналах, входящих в реферативную базу Scopus.

Объём и структура работы

Диссертация состоит из введения, 6 глав, заключения и списка литературы. Объём составляет 226 машинописных страниц, текст содержит 131 рисунок и 11 таблиц. Список литературы содержит 200 наименований.

Первая глава посвящена технологии математического моделирования турбулентного течения посредством крупномасштабных вычислительных экспериментов с использованием конечно-объёмных и конечно-разностных численных методов повышенной точности. Технология представляет собой свод согласованных между собой подходов для эффективной программной реализации алгоритмов вычислительной газовой динамики, рассчитанной на современные суперкомпьютеры, в том числе гибридные.

Во второй главе представлен новый параллельный алгоритм повышенной точности с многоуровневым распараллеливанием для крупномасштабных расчётов задач аэродинамики и аэроакустики на неструктурированных сетках.

В третьей главе представлен разработанный программный комплекс NOISEtte, реализующий новый параллельный алгоритм для моделирования сжимаемых турбулентных течений.

Четвертая глава посвящена разработке параллельного решателя уравнения Пуассона для моделирования несжимаемых турбулентных течений на десятках тысяч процессоров и на гибридных системах с массивно-параллельными ускорителями различной архитектуры. Разработанный автором решатель обгоняет современные масштабируемые реализации многосеточных методов на классе задач с одним периодическим направлением.

В пятой главе представлен разработанный программный комплекс STG-CFD&HTдля крупномасштабных DNS и LES расчётов на гибридных суперкомпьютерах.

В шестой главе представлены крупномасштабные расчёты ряда фундаментальных задач по моделированию турбулентных течений, выполненные по разработанной технологии программными комплексами, представленными в данной работе.

В заключении резюмируются результаты диссертационной работы.

Благодарности

Автор благодарит научного консультанта д.ф.-м.н. Т. К. Козубскую за помощь, полезные советы и поддержку. Автор выражает признательность коллегам по научной работе за многолетнее сотрудничество и совместные исследования, помощь в подготовке материала, полезные замечания и комментарии. Среди них И. В. Абалакин, Б. А. Бахвалов, А. П. Дубень, С. А. Суков (ИПМ им. М. В. Келдыша РАН); F.-X. Trias, R. Borrell (CTTC UPC); П. Б. Богданов (НИИСИ РАН); Автор благодарен академику РАН, д.ф.-м.н. Б. Н. Четверушкину за постоянную поддержку в научной работе. В работе использовались суперкомпьютеры Ломоносов – НИВЦ МГУ, МВС-100К и МВС-10П – МСЦ РАН, К-100 – ИПМ им. М. В. Келдыша РАН, MareNostrum – BSC, JFF – CTTC UPC. Автор выражает благодарность этим организациям за предоставленные ресурсы.

Глава 1

Параллельная технология численного моделирования задач газовой динамики алгоритмами повышенной точности

Вводные замечания

В настоящее время в развитии вычислительных систем сложилась неоднозначная ситуация. Производительность суперкомпьютеров за последние десять лет увеличилась в тысячу раз (Tianhe-2 33.9 PFLOPS против Earth-Simulator 35.8 GFLOPS). Но при этом выросла и сложность использования. Если раньше производительность кластерных систем росла за счёт увеличения числа процессоров и повышения их тактовой частоты, то теперь тенденции более многосторонние. Само по себе увеличение числа процессоров в системе усложняет её эффективное использование: требуются все более масштабируемые алгоритмы, изощренные схемы обменов данными, упрощение алгоритмов в пользу параллельной эффективности, но зачастую в ущерб общей вычислительной стоимости. Число ядер в крупнейших системах составляет сотни тысяч, а максимальное уже перевалило за миллион (Sequoia – IBM, USA). Естественно, лишь очень узкий класс алгоритмов может задействовать такие ресурсы, а задачи, которые такие алгоритмы могут решать, часто не оправдывают использование таких вычислительных мощностей.

С появлением многоядерных процессоров потребовался переход на более сложную двухуровневую параллельную модель. Стало обычным размещение по два 8-ми – 16-ти ядерных процессора на одном вычислительном модуле. Хуже того, уже представлена архитектура CPU Intel Xeon Phi с 72 ядрами. Иметь на одном узле суперкомпьютера десятки процессов, обменивающихся с другими узлами, крайне неэффективно. Поэтому распараллеливанию с MPI (Message Passing Interface) пришло на смену MPI+OpenMP распараллеливание (OpenMP – Open Multiprocessing), сочетающее модели с распределённой и общей памятью. Также сложностей добавило то, что число ядер на узле в разы превысило число каналов памяти.

Поскольку возможности повышения тактовой частоты процессоров давно исчерпались, пиковая производительность процессорных ядер растет за счёт совершенствования логики работы процессора с одной стороны, и за счёт расширения векторных регистров, с другой стороны. Ядро CPU с AVX (Advanced Vector Extensions) может выполнять за один такт 8 операций с плавающей точкой двойной точности. Естественно, расширение вектора усложняет эффективное использование процессора. К требованию бесконечной масштабируемости, бесконечной эффективности доступа к памяти, добавилось требование бесконечно эффективной векторизации.

Дополняет картину широкое применение массивно-параллельных ускорителей, таких как графические процессоры GPU (Graphics Processing Unit) производства AMD и NVIDIA или ускорители Intel Xeon Phi архитектуры MIC (Many integrated core). Разрабатываются новые перспективные архитектуры, как, например, прототип Mont-Blanc на основе ARM процессоров и ускорителей [1], который сочетает в себе ради экономии энергии сложность гетерогенных систем со слабостью упрощённых RISC (Reduced instruction set computing) процессоров.

Зоопарк архитектур и средств разработки делают положение разработчика параллельных программ почти безнадежным, как и “алчность” производителей ускорителей, вводящих в заблуждение пользователей и разработчиков сомнительными примерами ускорения прикладных задач и навязывающих свои средства разработки, применимые только на их устройствах. Естественно, когда пиковая производительность становится самоцелью, в пределе будет система с бесконечной производительностью, на которой, наконец, нельзя будет посчитать абсолютно ничего.

Данная работа направлена на то, чтобы противостоять этим негативным тенденциям и разрабатывать численные алгоритмы, эффективно применимые на различных архитектурах. Для традиционных архитектур с CPU общие подходы к разработке и реализации параллельных алгоритмов широко представлены в различной учебной литературе, например, в [2, 3]. В [4] подробно представлены принципы построения MPI-параллельной программной платформы для численных моделей на сетках общего вида, описана организация универсальной структуры данных. Применительно к рёберно-ориентированным конечно-объёмным алгоритмам, различные способы организации вычислений на неструктурированных сетках в рамках модели с общей памятью представлены, например, в [5]. В то же время гетерогенные вычисления на системах с ускорителями являются новым, активно развивающимся направлением. Краткий обзор средств разработки и примеров их применения есть, например в [6].

В данной работе представлена технология параллельных вычислений, специфическая для моделирования задач механики сплошной среды конечно-объёмными и конечно-разностными методами повышенной точности. Аппроксимация повышенной точности в рамках этих подходов подразумевает использование расширенных пространственных шаблонов, включающих, вообще говоря, переменное число ячеек. Использование неструктурированных сеток усугубляет проблему сложностью структуры данных, нерегулярным доступом к памяти. При этом низкая

удельная вычислительная стоимость на единицу данных, характерная таким алгоритмам, ещё более усложняет задачу эффективных параллельных вычислений.

Рассматриваемая технология представляет собой свод согласованных между собой подходов для эффективной программной реализации алгоритмов вычислительной газовой динамики для гетерогенных суперкомпьютеров. Используется многоуровневая параллельная модель, сочетающая различные типы параллелизма: с общей и распределённой памятью, с множественными и одиночными потоками команд на множественные потоки данных. Технология основана на разделении алгоритма на базовые операции и адаптации этих операций к многоуровневой параллельной модели, к потоковой обработке, к различным вычислительным архитектурам. Она рассчитана на широкий класс численных методов для моделирования задач аэродинамики, аэроакустики, тепломассопереноса.

Отдельное внимание уделено методике выполнения крупномасштабных расчётов задач газовой динамики. Как известно, нестационарные расчёты задач механики сплошной среды, а особенно турбулентных течений, требуют больших затрат вычислительных ресурсов. Прямое численное моделирование, DNS (Direct Numerical Simulation), и моделирование методом крупных вихрей, LES (Large Eddie Simulation), требуют высокого разрешения по пространству и времени, а также применения схем повышенного порядка точности для корректного воспроизведения турбулентного течения. Накопление статистики течения (средних полей, интегральных характеристик, спектральных характеристик) требует длительного периода интегрирования по времени. В связи с этим, численный эксперимент часто под силу только суперкомпьютерам. Вычислительная стоимость подобных расчётов может составлять сотни тысяч и миллионы процессорных часов. Естественно, за вычислительной стоимостью скрывается стоимость реальная, которая может составлять миллионы рублей на один расчёт. Поэтому проблема повышения эффективности численного эксперимента является достаточно острой и актуальной.

Прогресс в области математического моделирования и расширение возможностей численного эксперимента связаны с двумя очевидными направлениями развития, делающими расчёты более доступными. Во-первых, это рост производительности и совершенствование архитектур суперкомпьютеров, направленное на повышение энергоэффективности и удешевление полезной производительности. Во-вторых, это развитие численных методов, моделей и алгоритмов, приводящее к снижению вычислительной стоимости расчётов: моделирование турбулентности, снижающее требования к пространственному разрешению [7, 8]; улучшение параллельных решателей СЛАУ, в частности, методов на основе подпространств Крылова [9] и многосеточных методов [10, 11]; экономичные схемы повышенного порядка пространственной аппроксимации, например, с квазиодномерной реконструкцией [12, 13]; удешевление на интегрировании по времени, например, за счёт схем с локальным или дробным шагом.

Ещё одним направлением снижения вычислительной стоимости и повышения эффективности, которому порой уделяется не так много внимания, является совершенствование самой методики выполнения расчёта на современных суперкомпьютерах. В этой главе

представлен соответствующий набор методических рекомендаций. Представленная в главе методика охватывает различные аспекты выполнения численного эксперимента и опирается на опыт выполнения крупных расчётов как сжимаемых, так и несжимаемых течений.

Данная глава основывается на материале, представленном в статьях [6, 14–17].

1.1 Параллельная модель и средства разработки

1.1.1 Выбор средств разработки

Для соответствия современным архитектурам суперкомпьютеров предлагается использовать гибридную многоуровневую параллельную модель, сочетающую модели с общей и распределённой памятью, параллелизм MIMD (Multiple Instruction, Multiple Data) и SIMD (Single Instruction, Multiple Data). Уровни упорядочены следующим образом: 1) распределённая память, MIMD; 2) общая память, MIMD; 3) потоковая обработка (Stream processing), SIMD; 4) векторизация, SIMD. Структура уровней параллельной модели и соответствующие средства разработки показаны на рис. 1.1.

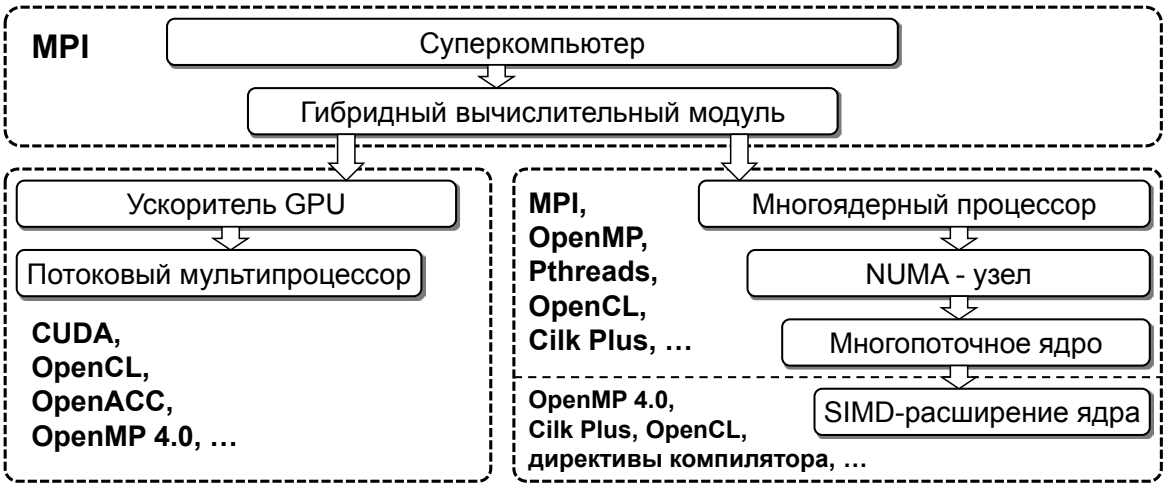


Рисунок 1.1: Многоуровневая параллельная модель и средства разработки для расчётов на гибридном суперкомпьютере

Параллелизм типа «потоковая обработка», упрощённая парадигма параллельного программирования, под которой понимается выполнение независимых однотипных вычислительных заданий, в полной мере соответствует архитектуре ускорителей, в частности графических процессоров. Под векторизацией понимается использование векторных регистров, требующее группировку аргументов в памяти, выравнивание по ширине вектора, и т.д. В потоковой обработке аргументы могут поступать из произвольных позиций в памяти, хотя также могут иметься определённые требования к доступу к памяти, в частности слияние доступа (coalescing). Стоит отметить, что алгоритм, записанный в рамках потоковой обработки, также хорошо подходит и для MPI и для OpenMP распараллеливания на CPU. Таким образом,

основной целью при разработке параллельного алгоритма является адаптация к потоковой обработке.

В качестве средства разработки на первом уровне, объединяющем узлы суперкомпьютера в рамках модели с распределённой памятью, наиболее широко используется интерфейс прикладного программирования MPI. Данный стандарт имеет множество реализаций, в том числе открытых, и имеется практически на всех кластерных системах. Выход нового стандарта MPI-3.0 [18], в котором, в частности, добавлены неблокирующие групповые операции, показывает, что стандарт активно развивается. Вышесказанное делает выбор MPI достаточно очевидным и обоснованным.

На втором уровне для распараллеливания по CPU ядрам многопроцессорного узла в рамках модели с общей памятью выбор более разнообразен. Основные интерфейсы прикладного программирования (API – Application programming interface) для многоядерных процессоров включают в себя OpenMP (Open multiprocessing) [19] и POSIX Threads. Первый реализован в большинстве современных компиляторов и представляется заметно более простым в использовании. Последний является более низкоуровневым стандартом, который предоставляет дополнительные возможности для управления параллельными потоками. Существует и множество других API для распараллеливания с общей памятью, как, например, Intel Cilk Plus [20] расширение языка C++.

Далее, использование SIMD расширения CPU ядра, то есть векторных регистров (технологии SSE, AVX, KNI и т.д.), требует адаптации алгоритма к SIMD параллельной модели. Векторизация программы может быть выполнена как с помощью встроенных автоматических средств компилятора и специальных директив (см. [21] для компилятора Intel), так и с помощью специализированных API. Первый подход, хоть и представляется простым, но, с одной стороны, достаточно ограничен, а с другой стороны, не является переносимым и ограничен конкретным компилятором. В рамках второго подхода может использоваться OpenMP 4.0 [19], новая версия стандарта, имеющая поддержку векторизации, или, например,Intel Cilk Plus, также поддерживающий векторизацию. Кроме того, открытый стандарт OpenCL (Open computing language), о котором речь пойдет далее, поддерживает векторные типы данных и может использоваться на CPU.

Для распараллеливания на многоядерных процессорах предлагается использовать OpenMP – открытый стандарт, поддерживаемый основными компиляторами языка C++ и Fortran. Из соображений переносимости и простоты использования выбор между OpenMP и Posix Threads был сделан в пользу первого. Кроме того, новые возможности для векторизации вычислений и перспективы программирования гетерогенных вычислений на массивно-параллельных ускорителях в рамках директивного подхода, предусмотренные новой версией стандарта, делают выбор OpenMP достаточно обоснованным.

Среди средств разработки для ускорителей можно выделить директивные и низкоуровневые подходы. Активно развивающимися директивными API являются, например OpenACC [22] и тот же OpenMP 4.0, во многом аналогичный первому. Директивные подходы проще в использовании,

но как правило более ограничены и не позволяют получить максимум производительности. Примеры CFD алгоритмов, для которых было получено хорошее ускорение с использованием директивных API представлены, например, в [23, 24].

Наиболее распространёнными низкоуровневыми API являются проприетарный API CUDA [25] для GPU производства NVIDIA и открытый вычислительный стандарт OpenCL [26], поддерживаемый всеми основными производителями оборудования. Опыт использования обоих стандартов показывает, что CUDA и OpenCL достаточно схожи между собой, а вычислительные ядра (kernel) часто выглядят практически идентично. Эти средства представляются более сложными, чем директивные подходы, но потенциально позволяют достичь более высокой производительности. В дополнение, существуют специализированные инфарструктуры для гетерогенных вычислений, упрощающие использование ресурсов гибридного узла, как, например, StarPU [27], CGCM [28], Distri-GPU [29].

В качестве средства разработки для ускорителей был выбран открытый вычислительный стандарт OpenCL, поддерживаемый основными производителями ускорителей, в том числе GPU NVIDIA, AMD, процессоры и ускорители Intel, процессоры и ускорители архитектуры ARM. Для сложных алгоритмов, предназначенных для крупномасштабных расчётов задач механики сплошной среды на суперкомпьютерах, было бы сомнительно полагаться на CUDA и ограничивать применимость программной реализации только ускорителями GPU NVIDIA. Для упрощения гетерогенной реализации была выбрана инфраструктура типа «планировщик задач», которая подразумевают декомпозицию исходной задачи на составляющие операции, представление задачи в виде графа, описывающего исполнение вычислительных и коммуникационных заданий и связи между ними. В качестве такого планировщика был выбран [30], являющийся отечественной разработкой с открытым исходным кодом. Планировщик упрощает работу по обслуживанию вычислений на гибридном вычислительном узле и автоматически обеспечивает обмен данными (по MPI и между хостом и ускорителем) одновременно с вычисления на ускорителе, так называемый «overlap» режим. Планировщик имеет три очереди заданий: EXEC – исполнение вычислительных ядер на устройстве, LD – загрузка данных на устройство, ST – выгрузка данных с устройства. Независимые по входным данным команды из разных очередей могут выполняться одновременно, за счёт чего автоматически обеспечивается одновременное выполнение обмена данными и вычислений. Общая схема работы инфраструктуры планировщика показана на рис. 1.2.

1.1.2 Декомпозиция расчётной области

Верхний уровень параллельной модели основан на геометрическом параллелизме. Расчётная область представляет собой сетку пространственной дискретизации, по сути, набор расчётных ячеек (контрольных объёмов) и связей между ними. Для численной схемы с определением сеточных функций в узлах сеточные данные определяются наборами трёх типов: данные по узлам (ячейкам), по граням ячеек, по элементам сетки (тетраэдры, пирамиды, призмы, гексаэдры

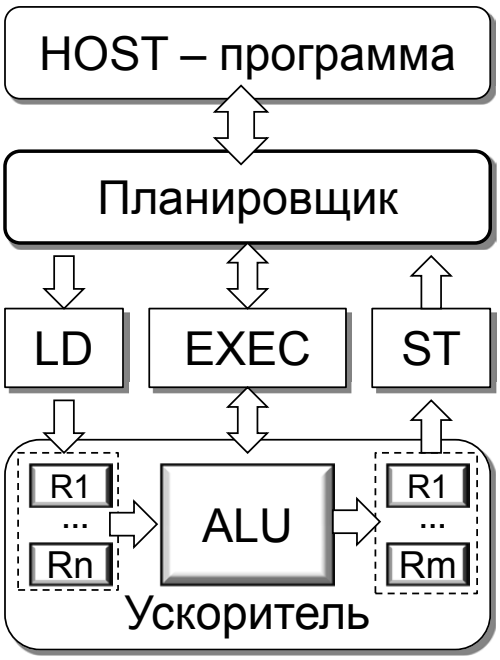


Рисунок 1.2: Упрощённая схема работы планировщика [30]

и т.д.). Поскольку ячейки строятся вокруг узлов, набор ячеек эквивалентен набору узлов сетки. Аналогично, набор граней ячеек эквивалентен набору сеточных ребер. Топология связей ячеек представляется графом сетки, в котором вершины и ребра соответствуют сеточным узлам и ребрам. Для схемы с определением сеточных функций в центрах элементов основных наборов будет два: данные по ячейкам – сеточным элементам и данные по граням ячеек. Топологию связей ячеек описывает дуальных граф сетки, в котором вершины – ячейки, ребра – грани между ними. Структура данных в любом случае дополняется набором граничных (внешних) граней, то есть граней, относящихся только к одной ячейке.

Расчётная область посредством рациональной декомпозиции графа сетки разделяется между параллельными процессами на подобласти 1-го уровня, для каждой ячейки сетки определяется номер процесса-владельца. Назовем *собственными* те ячейки, которые принадлежат данному процессу и составляют его подобласть (рис. 1.3).

Две ячейки являются *соседними*, если они связаны шаблоном численной схемы, то есть шаблон, центрированный в одной ячейке или её грани, включает вторую ячейку. Набор *чужих* ячеек, не принадлежащих подобласти, но соседних с ячейками данной подобласти, будем называть *гало*. Чужие ячейки, непосредственно граничащие с собственными ячейками – это гало 1-го уровня. Гало элементы 2-го уровня – чужие ячейки, соседние с гало ячейками 1-го уровня, и т.д. Для схемы повышенного порядка может требоваться несколько таких уровней соседства. Назовем *расширенной подобластью* объединение множества собственных и гало ячеек. *Интерфейсные ячейки* – собственные ячейки, имеющие соседей из гало. *Внутренние ячейки* – это собственные ячейки, имеющие связи только с собственными ячейками. Все типы ячеек показаны на рис. 1.3. Аналогичным образом определяются наборы сеточных элементов и

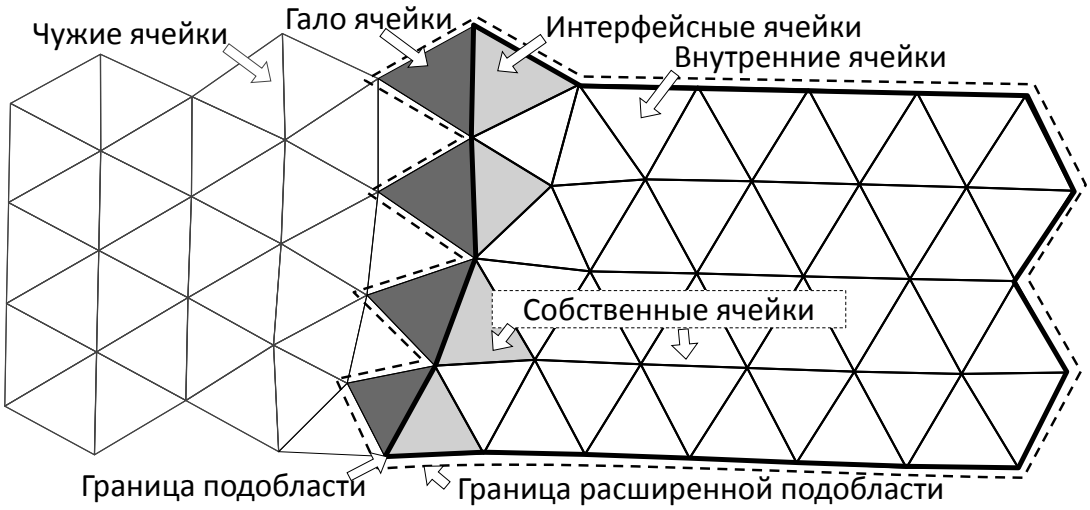


Рисунок 1.3: Декомпозиция расчётной области

граней. *Интерфейсные грани* разделяют собственную и чужую ячейки, внутренние – разделяют собственные ячейки.

1.2 Представление алгоритма в виде базовых операций

1.2.1 Тестовая оболочка для операций

Для того, чтобы противостоять сложности программной модели и сложности алгоритма для гетерогенных вычислений, необходимо создать комфортные условия для работы с программной реализацией. Предлагается разделить алгоритм на составляющие его операции и снабдить каждую операцию унифицированной инфраструктурой, тестовой оболочкой, для быстрого многократного запуска операции отдельно от основного кода и для проверки корректности вычислений. Каждая базовая операция реализуется в трёх вариантах:

1. вариант для CPU в исходной структуре данных;
2. переходный вариант, работающий на CPU, но в структуре данных, адаптированной к ускорителям и потоковой обработке;
3. вариант для ускорителя, реализация на OpenCL.

Для операции чётко определяется набор входных и выходных данных, из основного расчётного алгоритма входные данные на репрезентативном наборе задач выгружаются в файлы, для чего реализуются функции записи и чтения входных данных. Далее с базовой операцией можно работать отдельно от основного кода, избегая затрат времени на инициализацию расчёта реальной задачи и больших затрат памяти, также для повышения точности профилирования можно вызывать операцию многократно без аварийной остановки из-за некорректных данных. Для каждой операции создаётся функция проверки корректности путём сравнения результатов.

Основная CPU реализация трансформируется в переходную, работающую в ориентированной на ускоритель структуре данных, к ней создаётся соответствующий преобразователь входных данных. Потоковая обработка на ускорителе имитируется внешним циклом, перебирающим номера нитей рабочей группы. Выходные данные переходной реализации сравниваются с исходной реализацией, чтобы гарантировать корректность вычислений. Таким образом, существенная часть отладки остаётся на CPU, и процесс разработки значительно упрощается. Затем на основе переходной реализации с существенно меньшими трудозатратами создаётся версия для ускорителя. Тестовая оболочка операции дополняется, соответственно, копированием входных и выходных данных между хостом и ускорителем. Корректность вычислений гарантируется сравнением выходных данных с исходной реализацией. Далее, работая с операциями по отдельности в составе тестовой оболочки, реализующей проверку корректности, быстрый запуск с разными наборами входных данных, точное профилирование, выполняется оптимизация вычислений. Процесс поэтапного переноса вычислений для базовых операций показан на рис. 1.4. Такой подход значительно упрощает процесс разработки, повышает надежность кода, ускоряет отладку и поиск ошибок.

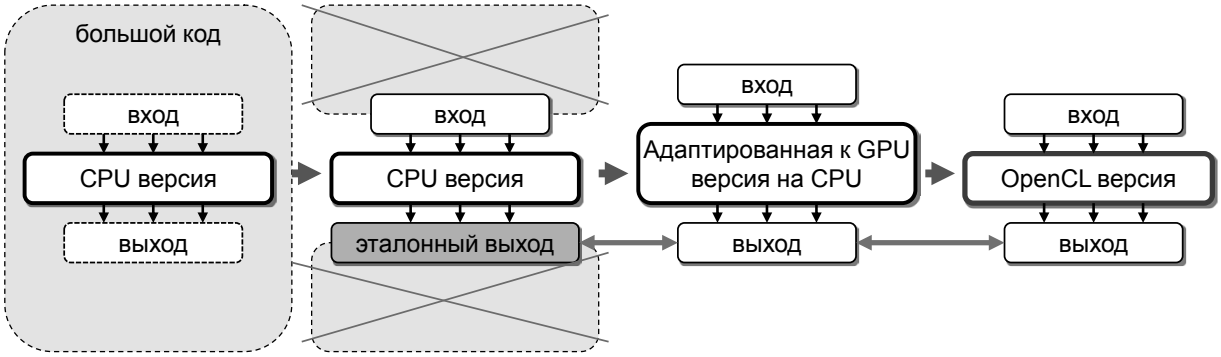


Рисунок 1.4: Процесс переноса базовых операций на ускоритель

1.2.2 Примеры разложения на базовые операции

Расчётный алгоритм можно представить в виде графа исполнения, разложив вычисления на шаге по времени на вычислительные и коммуникационные операции. Базовые вычислительные операции обрабатывают набор данных определённого типа – ячейки, грани, сеточные элементы и т.д.

Рассмотрим, например, конечно-объёмной алгоритм повышенного порядка точности [17] на основе полиномиальной схемы с определением переменных в центрах элементов. В случае явной схемы в простейшем виде действия на шаге интегрирования по времени состоят из расчёта конвективных и диссипативных потоков через грани контрольных объёмов, расчёта граничных условий, и шага интегрирования Рунге-Кутты. Сосредоточим внимание на стадии вычисления конвективных потоков. Эта стадия сама по себе не является базовой операцией,

поскольку состоит из нескольких операций над различными наборами данных. Данная стадия расщепляется на несколько операций, которые уже можно считать базовыми:

1. расчёт коэффициентов полиномов, определяемых в каждой ячейке, – в цикле по ячейкам;
2. реконструкции переменных в центрах граней – в цикле по граням;
3. суммирование потоков с граней в центры ячеек – в цикле по ячейкам.

Вообще говоря, конечно-объёмные схемы повышенной точности предполагают вычисление переменных в нескольких гауссовых точках на гранях, но это не меняет сути организации вычислений, представленные далее подходы применимы и для таких схем.

Аналогично, для алгоритма на основе схемы с квазиодномерной реконструкцией [31], который будет подробно представлен в следующей главе, расчёт конвективных потоков также распадается на несколько базовых операций:

1. вычисление реконструируемых переменных – цикл по узлам;
2. расчёт узловых (определённых в сеточных узлах) градиентов – цикл по сеточным элементам;
3. расчёт потоков через грани – цикл по граням;
4. суммирование потоков с граней в узлы – в цикле по узлам.

В [32] представлен пример компоновки достаточно сложного алгоритма для моделирования несжимаемых течений и теплопереноса, который включает решатель СЛАУ, модели турбулентности различных типов (LES и регуляризация), всего из 6-ти базовых вычислительно емких операций, реализованных для гетерогенных вычислений:

1. нелинейный оператор (конвекция);
2. линейный оператор (диффузия);
3. линейная комбинация векторов;
4. БПФ для набора векторов;
5. скалярное произведение;
6. матрично-векторное произведение.

Этот алгоритм подробно представлен в 4-й и 5-й главе.

Наиболее показательный пример – это алгоритм (см. [33]) для моделирования несжимаемых турбулентных течений на неструктурированных сетках, разработанный при участии автора. Данный алгоритм состоит всего из трёх базовых операций, являющихся стандартными операциями линейной алгебры:

1. матрично-векторное произведение;
2. скалярное произведение;
3. линейная комбинация векторов.

Такой набор из стандартных операций, совместимых с потоковой обработкой и SIMD, делает алгоритм естественным образом легко переносимым на любую вычислительную архитектуру (хотя бы путём переключения с одной библиотеки линейной алгебры на другую).

1.3 Распараллеливание с распределённой памятью и обмен данными

Расчётная область разделяется на подобласти для распределения вычислительной нагрузки между параллельными процессами. К разбиению, как правило, предъявляются такие естественные требования, как:

- сбалансированность, то есть минимизация различий по числу ячеек в подобластях;
- минимизация объёма обмена данными, то есть минимизация числа интерфейсных граней;
- минимизация числа сообщений в обмене данными, то есть максимального числа соседних подобластей для снижения потерь на латентности сети.

Кроме того, декомпозиция расчётной области должна учитывать неоднородности, такие как расчёт граничных условий, препятствий, источников, и т.д., а также специфику алгоритма и получаемую вычислительную стоимость. Простейший пример – квадратную декартову сетку естественно разбить на квадратные подобласти, поскольку квадрат из прямоугольников равной площади имеет наименьшую длину границ. Но для алгоритма [34], где по подобластям используется LU разложение для ленточной матрицы, такое разбиение неэффективно: ширина ленты L и U матриц пропорциональна длине минимальной стороны прямоугольной подобласти, для квадратной подобласти выходит большая вычислительная стоимость.

Для разбиения расчётной области на неструктурированной сетке строится граф, в котором вершинам соответствуют ячейки, ребрам – грани, разделяющие ячейки. При вычислительной неоднородности вершинам графа могут быть заданы веса. Для представления графа подходит, в частности, стандартный построчно-разреженный формат CSR (Compressed Sparse Row) для разреженных матриц. Для получения рационального разбиения может использоваться одна из открытых библиотек, например, [35,36] на основе иерархических алгоритмов, или [37] на основе инкрементного алгоритма декомпозиции и алгоритма рекурсивной координатной бисекции.

1.3.1 Построение схемы обмена данными

Обмен между процессами необходим в случае такой зависимости по данным, когда в одной операции входными данными для обработки некоторого элемента являются выходные данные другой операции, соответствующие элементу другого типа или другому элементу того же типа. Например, для явной конечно-объёмной схемы расчёт потока через грань – операция по набору граней, использует для каждой грани на вход значения с предыдущего временного слоя из ячеек – элементы другого типа, которые она разделяет. В конечно-разностном случае аналогичная операция может быть записана в цикле по узлам, но на вход на каждого узла используются данные из соседних узлов – тип элемента тот же, но сами элементы другие.

Схема обмена данными по заданной декомпозиции, которая описывает для каждого процесса, какие данные каким процессам он должен передать и какие данные от каких

процессов получить, непосредственно определяется из зависимости по входным данным между операциями. Количество уровней соседства в гало определяется пространственным шаблоном численной схемы.

Для построения такой схемы обменов для больших сеток предлагается следующий параллельный алгоритм. На вход подаётся топология сетки и её декомпозиция на P подобластей. Алгоритм выполняется p параллельными процессами.

1. Процесс i берет себе узел сетки, если i = I/((P-1)/p+1), где I – номер владельца узла. В каждом узле заводится список номеров процессов. Первым в список записывается номер владельца, остальные позиции изначально пусты.

2. Каждому узлу в список добавляются все номера из списков соседних узлов (естественно, если номеров ещё нет в списке).

3. Выполняется обмен данными для обновления списков номеров в узлах (схема обменов по гало 1-го уровня для p подобластей следует напрямую из топологии сетки). Для добавления гало следующего уровня – переход на пункт 2, пока нужное число уровней не достигнуто.

4. По полученной маркировке узлов формирование для каждой из P подобластей списков номеров узлов на получение и отправку каждому соседнему процессу.

Ячейки можно переупорядочить, чтобы интерфейсные ячейки располагались в памяти более компактно: сначала внутренние ячейки, затем интерфейсные, затем гало ячейки, сгруппированные по номеру владельца. Для корректного представления расширенной подобласти вводятся три типа нумерации:

1. *глобальная* – исходная нумерация ячеек в расчётной области;
2. *расширенная локальная* – в расширенной подобласти;
3. *локальная* – в подобласти.

Для отображения идентификаторов из одной нумерации в другую формируются соответствующие индексные массивы. Для обновления гало предлагается использовать наиболее простой вариант с неблокирующими обменами MPI_Isend/MPI_Irecv и ожиданием завершения операций MPI_Waitall. Построение графа связей между подобластями, реализация алгоритма раскраски для потактового выполнения обменов блокирующими функциями передачи данных представляется нецелесообразным, поскольку современные библиотеки MPI сами достаточно хорошо планируют обмены, выполняя передачу данных оптимальным образом.

1.3.2 Обмен данными одновременно с вычислениями

Для реализации обмена данными в режиме «overlap», с одновременным выполнением передачи данных и вычислений, необходимо выделить независимые вычислительные и коммуникационные операции. Для этого операцию над набором элементов некоторого типа,

зависящую от операции обмена данными, разделим на два этапа: расчёт внутренних элементов и расчёт интерфейсных элементов. Расчёт внутренних элементов может выполняться одновременно с обменом данными, который нужен только интерфейсным элементам. Пример трансформации графа исполнения в режим «overlap» показан на рис. 1.5.

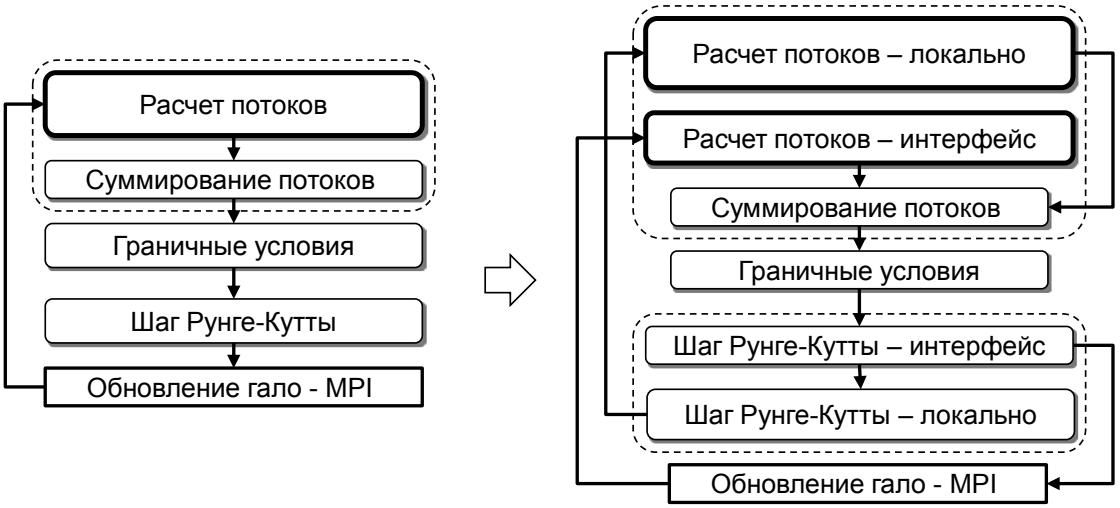


Рисунок 1.5: трансформация графа шага интегрирования по времени в режим «overlap»

Такой режим обычно оправдан только при вычислениях на ускорителях, чтобы скрыть за вычислениями MPI обмены и обмены между хостом и ускорителем. Для упрощения гетерогенной реализации может использоваться инфраструктура типа планировщик задач [30], который по заданному графу зависимостей автоматически обеспечивает выполнение передачи данных одновременно с вычислениями на ускорителе. Схема выполнения обменов данными в режиме «overlap» показана на рис. 1.6.

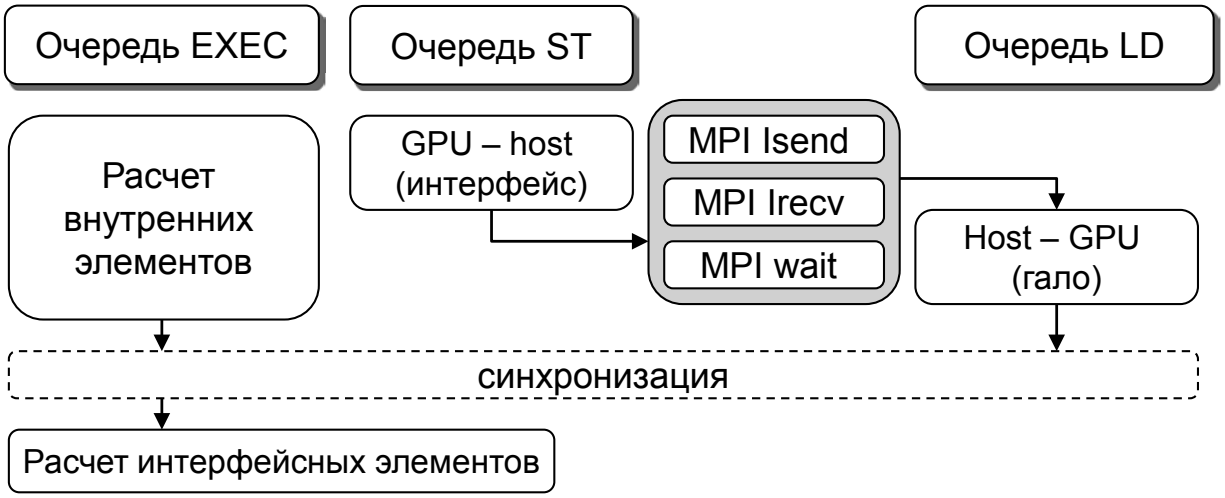


Рисунок 1.6: Схема одновременного обмена данными и вычислений на ускорителе

Эффект от использования режима с одновременным выполнением вычислений и обменов данными показан на рис. 1.7 для конечно-объёмного алгоритма для моделирования сжимаемых

течений на неструктурированных гибридных сетках. Более подробно этот пример применения такого подхода для расчёта на множестве GPU представлен в [6], где демонстрируется высокая эффективность даже для численной схемы с крайне низкой вычислительной стоимостью.

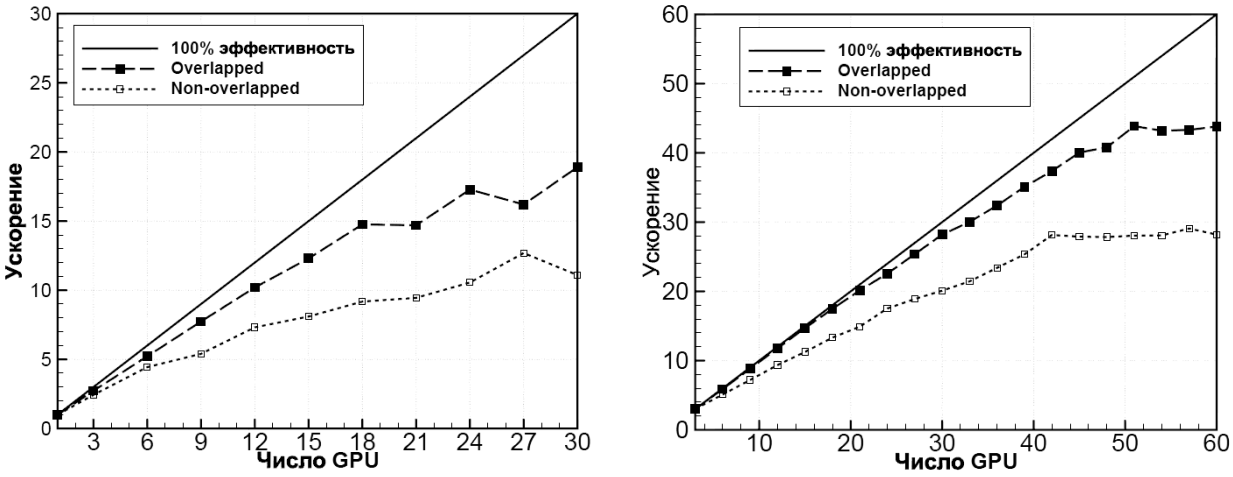


Рисунок 1.7: Ускорение в тестовом расчёте на неструктурированной сетке на суперкомпьютере К-100: слева – ускорение относительно одного GPU, сетка 4 млн тетраэдров, справа – ускорение относительно одного узла с тремя GPU, сетка 16 млн тетраэдров

Режим «overlap» при расчётах на CPU может скрывать потери на групповых обменах (в которые также входит простой из-за вычислительного дисбаланса). Например, в [38] в методе бисопряжённых градиентов скрываются обмены для скалярных произведений.

Для обменов типа «точка-точка» при обновлении гало такой подход на CPU обычно не даёт выигрыша, поскольку MPI библиотека все равно вытеснит вычисляющий процесс с ядра и займет ресурсы для передачи данных.

1.4 Распараллеливание с общей памятью

Описанные в следующем разделе способы адаптации к потоковой обработке, естественно, также применимы для распараллеливания с общей памятью. Поэтому здесь рассмотрим только те способы, которые не подходят для потоковой обработки.

1.4.1 Многоуровневая декомпозиция

Проблемы для распараллеливания операций, представимых, как правило, в виде цикла по набору элементов расчётной области, следуют из зависимости по данным между итерациями цикла. При одновременной обработке итераций, имеющих зависимости, нарушается целостность доступа к данным, возникает состояние гонки, что приводит к некорректному результату. Примеры подобных операций:

– Расчёт суммарных потоков в ячейках. Операция по набору граней, для каждой грани вычисляется поток между ячейками, которые грань разделяет, результат прибавляется в ячейки. Пересечение: на вход текущее значение суммарного потока в ячейке, на выход – обновленное значение этого же потока. При одновременной обработке граней одной ячейки результат непредсказуем.

– Вычисление узловых градиентов. Операция по набору тетраэдров, для каждого тетраэдра находится градиент сеточной функции на тетраэдре, затем вклад от тетраэдра (от пересечения тетраэдра и контрольного объёма вокруг узла) прибавляется в каждый узел тетраэдра. Пересечение аналогично первому случаю.

Такие же зависимости возникают при расчёте потоковых граничных условий, когда в вычислениях по граничным граням сетки результат прибавляется в узлы (вершины граней); при расчёте диссипативных потоков, когда производные определяются по градиентам на тетраэдрах, и так далее.

Для решения этой проблемы предлагается использовать дальнейшую декомпозицию подобласти процесса по нитям на подобласти 2-го уровня. В общем виде рассмотрим операцию по набору элементов расчётной области, связывающей две и более ячейки (грань – 2 ячейки, треугольник – 3, тетраэдр – 4 и т.д.) с записью конечного результата в ячейки. Элементы с ячейками из разных подобластей будут *интерфейсными*. Далее возможны разные способы устранения пересечения по данным.

1. Параллельная обработка внутренних элементов, последовательная обработка интерфейсных элементов. Недостаток – последовательная часть.

2. Параллельная обработка и внутренних и интерфейсных элементов, но запись результата только в свои ячейки. Недостаток – дублирование вычислений по интерфейсным элементам, дополнительные условные переходы.

Эти способы могут быть достаточно эффективными, но лишь на небольшом числе нитей. Современный узел с двумя 8-ядерными процессорами с HyperThreading – это 32 нити. На ускорителях Intel Xeon Phi требуется 240 нитей. Для таких чисел параллельных потоков данный способ недостаточен. Для решения проблемы предлагается продолжить многоуровневое разбиение. Обработка интерфейсных элементов выделяется в отдельный этап, для набора интерфейсных элементов аналогичным образом строится граф связей и выполняется разбиение на подобласти 3-го уровня также по числу нитей (и так далее при необходимости). Этого достаточно для нескольких сотен параллельных потоков. Возьмём, например, подобласть процесса из 119 тыс. узлов, 366 тыс. граней, 632 тыс. тетраэдров (вершинно-центрированная схема). При декомпозиции для 8 нитей получаем 8 тыс. интерфейсных граней и 23 тыс. тетраэдров. На нить приходится 46 тыс. граней, обработка интерфейсной грани дублируется 2 раза, что эквивалентно довеску в 1 тыс. граней, то есть всего 2.2% от общей нагрузки,

для тетраэдров вес интерфейса составит не более 11% (вычисления для тетраэдра могут дублироваться до 4-х раз), что приемлемо. При декомпозиции для 240 нитей число интерфейсных граней и тетраэдров равно уже, соответственно, 52 тыс. и 157 тыс., что будет 14% по граням и до 50% по тетраэдрам. При декомпозиции 3-го уровня число интерфейсных граней уменьшается до 4.5 тыс. (1.2%), тетраэдров – до 26 тыс. (до 16%), что вполне приемлемо.

При таком подходе операция выполняется в 3 этапа с точками синхронизации между ними:

1. расчёт внутренних элементов 2-го уровня,
2. расчёт внутренних элементов 3-го уровня,
3. расчёт интерфейса 3-го уровня.

При этом на первых двух этапах не требуются дополнительные условные переходы на проверку принадлежности ячеек, а накладные расходы на синхронизацию между этапами неизмеримо малы. При декомпозиции необходимо, как и в распараллеливании с распределённой памятью, обеспечивать качественную балансировку.

В работе [5] рассматривается аналогичный, но более общий подход для существенно многопоточного режима, заключающийся в декомпозиции набора сеточных элементов на группы поднаборов, в каждой из которых поднаборы не имеют пересечения по данным между собой, то есть нет элементов, объединяющих ячейки из разных наборов. В таком подходе нити одновременно обрабатывают каждая свой поднабор элементов, а директива OpenMP применяется к дополнительному внешнему циклу по поднаборам. Между обработкой групп поднаборов, соответственно, необходимы точки синхронизации. Такой подход представляется несколько более сложным в реализации, поскольку не ограничивается тремя этапами и требует нетривиального алгоритма распределения элементов по поднаборам, но зато отсутствует этап с обработкой интерфейсных элементов. Также имеется проблема балансировки, поскольку распределяются крупные вычислительные подзадачи.

1.4.2 Оптимизация доступа к оперативной памяти

Для конечно-объёмных и конечно-разностных алгоритмов для моделирования течения жидкости или газа обычно характерна очень низкая удельная вычислительная стоимость на единицу данных из памяти (например, в [17] менее одной операции на байт). При этом число каналов памяти в разы меньше числа параллельных нитей. В связи с этим особое внимание необходимо уделить организации структуры данных и доступу к памяти.

При работе с неструктурированными сетками существенное влияние оказывает нумерация сеточных элементов, ячеек и граней. В [16] влияние нумерации оценивается на конкретном примере для CPU и GPU. Чтобы повысить компактность данных в памяти и снизить число промахов в кэш, можно использовать алгоритм Cuthill-McKee [39] уменьшения ширины ленты разреженных матриц, или другие способы нумерации, обзор которых представлен, например, в [5]. Пример переупорядочивания показан на рис. 1.8 как портрет разреженной матрицы, в которой строки соответствуют ячейкам, позиции портрета – граням. Расстояние от ненулевого

элемента в строке до диагонали пропорционально расстоянию в памяти между ячейками, входящими в шаблон схемы, центрированный в ячейке, соответствующей данной строке. Соответственно, чем ближе к диагонали позиции, тем меньше потери в кэш.

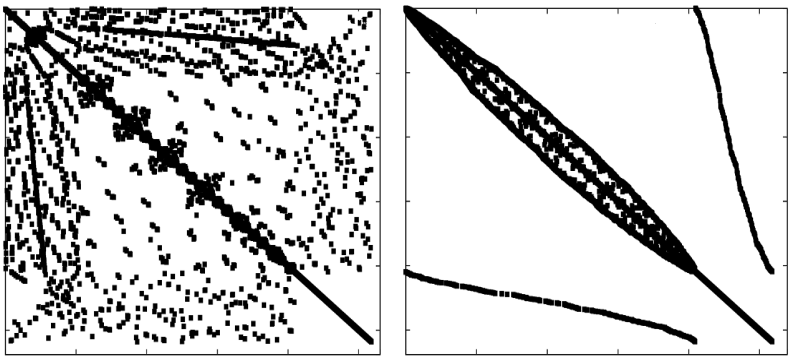


Рисунок 1.8: Пример изменения портрета матрицы при переупорядочивании номеров ячеек для небольшой тетраэдральной сетки: слева – исходная нумерация, справа – после переупорядочивания)

На ускорителях с мощным механизмом сокрытия латентности эффект от переупорядочивания может быть меньше, чем на CPU. Также сетки, построенные качественными сеточными генераторами, обычно имеют достаточно хорошую нумерацию и эффект от переупорядочивания тоже может быть не очень большим.

Далее, по фактической декомпозиции под заданное число нитей, N_T, для повышения эффективности необходимо выполнить лексикографическую сортировку элементов расчётной области по вектору $(p,n_1,n_2,\dots)$, где $n_1,n_2,\dots$ – номера ячеек, связанных с элементом, а старший индекс p обозначает номер подобласти: если элемент внутренний для декомпозиции второго уровня, то $0 \le p < N_T$, если внутренний для 3-го уровня, то $N_T \le p < 2N_T$, если интерфейсный, то $p = 2N_T$. Таким образом, упорядочивание для каждой подобласти задаёт непрерывный диапазон элементов и не потребуется проматывать итерации или использовать непрямую адресацию к элементам, а упорядочивание по номерам ячеек снижает влияние непрямой адресации к ячейкам.

Также для уменьшения затрат на доступ к памяти на CPU помогает режим HyperThreading, обеспечивающий аппаратную поддержку выполнения двух нитей на одном ядре. На рассматриваемых алгоритмах наблюдался выигрыш до 15-20%.

1.5 Адаптация к потоковой обработке

Для потоковой обработки деление на подобласти не подходит, необходимо свести вычислительные операции к набору независимых однородных подзадач. В общем виде рассмотрим операцию по набору элементов расчётной области, связывающей две и более ячейки с записью конечного результата в ячейки.

1.5.1 Устранение зависимости по данным

Устранить зависимость по данным можно различными способами.

1. Разделение операции на такты с точкой синхронизации между тактами: исходный набор элементов делится на поднаборы, в которых нет элементов, имеющих общую ячейку; затем возможна потоковая обработка каждого поднабора. Недостаток – снижение эффективности доступа к памяти, накладные расходы на запуск множества тактов и синхронизацию.

2. Разделение операции на два этапа: расчёт по элементам и хранение промежуточного результата в массиве по элементам; в цикле по ячейкам сборка результатов с элементов. Недостаток – расход памяти на промежуточный массив и на обратную топологию, определяющую для каждой ячейки список связанных с ней элементов.

В первом случае число тактов не может быть меньше, чем максимальное число элементов, имеющих общую ячейку. Для вершинно-центрированной схемы это особенно плохо, поскольку число тактов не ограничено, оно определяется сеткой и обычно составляет 15–30. Такое разделение на поднаборы исключает у варпа (группы нитей, одновременно выполняющейся на мультипроцессоре ускорителя) доступ на чтение в одну ячейку, то есть исключено слияние доступа и повторное использование данных, что приводит к снижению скорости чтения данных и росту промахов в кэш. Преимущество второго подхода в случае его применимости показано в [40].

Для расчёта потоков через грани второй подход наиболее эффективен, как в случае потоковой обработки, так и в случае распараллеливания с общей памятью. Но, например, для операции расчёта узловых градиентов он неприменим, поскольку требуется промежуточный массив размером, эквивалентным 270 массивам по ячейкам (тетраэдров в среднем в 6 раз больше, чем узлов, в каждом надо хранить 45 значений – градиенты от 15 переменных по трем пространственным направлениям).

1.5.2 Оптимизация доступа к памяти ускорителя

Для вычислений на ускорителе доступ к памяти также имеет большое значение, поскольку отношение пиковой производительности к пропускной способности памяти составляет порядка 30 раз (для вещественных чисел двойной точности), при этом латентность доступа к памяти на ускорителе в несколько раз больше.

Оптимизация доступа к памяти, помимо той же проблемы уменьшения промахов в кэш, сводится также к выравниванию данных и слиянию доступа. Выравнивание доступа подразумевает, что группа нитей варпа начинает чтение с позиции, выровненной по кэш линии. Для выравнивания применяется так называемый “padding” – заполнение блоков данных нулями, чтобы следующий блок данных был выровнен по нужной границе. Если на CPU выигрыш от выравнивания может компенсироваться увеличением числа промахов в кэш за счёт увеличения

расстояния между считываемыми данными, то на ускорителе выравнивание может давать преимущество.

Слияние доступа, “coalescing”, подразумевает, что нити варпа одновременно обращаются к данным, попадающим в одну кэш линию. Пример выигрыша на слиянии доступа виден на простом примере размещения в памяти блочных векторов. Каждой ячейке соответствует набор из 5 значений сеточных функций физических переменных (плотность, компоненты скорости, давление). Данные могут быть развернуты в линейный массив путём записи NE блоков по 5 значений, где NE – число ячеек. В этом случае адресация j-й переменной для i-й ячейки будет иметь вид V[i*5+j]. Другой вариант – транспонировать данные в 5 блоков по NE значений, что соответствует адресации V[i+j*NE]. Пусть каждая нить считывает из ячейки все 5 переменных. Для архитектуры CPU предпочтительнее первый вариант, поскольку расстояние в памяти между наборами данных двух ячеек будет в 5 раз меньше, следовательно, меньше промахи в кэш. Для архитектуры GPU предпочтительнее второй вариант в силу SIMD параллелизма. Нити варпа все одновременно сначала считывают 1-ю переменную, затем 2-ю, и т.д. Поэтому слияние доступа имеет большее значение, чем промах в кэш. Во втором случае чтение значений для двух соседних нитей попадает в соседние позиции в памяти. В [40] показан выигрыш до 1.5 раз за счёт транспонирования блочных векторов.

1.5.3 Оптимизация под архитектуру ускорителя

Для GPU характерно очень небольшое количество локальной памяти, доступной нитям на потоковом процессоре, и очень небольшое число регистров (63 для NVIDIA Fermi). Нехватка регистров приводит к эмуляции регистров через глобальную память, что существенно снижает производительность. Поэтому приоритетной задачей является минимизация ресурсоёмкости операций для исключения эмуляции регистров. В связи с этим необходимо минимизировать зерно параллелизма, то есть элементарные задания для нитей. В [17] на примере операции вычисления коэффициентов для полиномиальной реконструкции показано, как уменьшение зерна параллелизма с обработки ячейки одной нитью, что характерного для CPU, до обработки одной ячейки 32-мя нитями, существенно повысило коэффициент загрузки мультипроцессора. Для уменьшения расхода локальной памяти и регистров также предлагается по возможности разделять базовые операции на несколько этапов с сохранением промежуточных данных в дополнительных массивах. Экономия регистровой памяти может компенсировать затраты на чтение промежуточных данных и накладные расходы на запуск вычислительных ядер. В [17] показано такое разделение на этапы операции вычисления конвективных потоков, существенно повысившее эффективность.

Естественно, в рамках SIMD модели необходимо обеспечить, чтобы нити варпа не имели расхождений в ветвлениях вычислительного ядра и выполняли идентичные задания. В случае неструктурированных сеток, где число связей у элементов различно, необходимо предпринимать дополнительные меры, в частности, переупорядочивание элементов и группировку по

количеству связей (или по объёму вычислений) таким образом, чтобы в пределах варпов нити имели одинаковые задания. Это, например, группировка ячеек по числу элементов в шаблоне, группировка узлов по числу соседей, и т.д. Такой подход хорошо проиллюстрирован в [41], где рассматриваются вопросы эффективного представления разреженных матриц, имеющих в строках различное число элементов.

Кроме того, само по себе наличие в вычислительных ядрах циклов и ветвлений, даже если ветвления совпадают для всех нитей варпа, снижает производительность вычислений существенно больше, чем в случае CPU с мощными механизмами предсказания ветвлений, упреждающей выборки и т.д. Возможность компиляция OpenCL ядер во время выполнения CPU программы (в отличие от CUDA) позволяет широко использовать директивы препроцессора. Предлагается все ветвления, известные на момент выполнения CPU кода, выносить на уровень препроцессора; все известные скалярные переменные записывать в виде констант препроцессора, особенно переменные, задающие границы циклов (это позволяет компилятору разворачивать циклы). Таким образом, исходный код OpenCL ядер состоит из двух частей – основной части, хранящейся в файле, и добавляемый перед компиляцией фрагмент с определением констант препроцессора. Файл исходного кода считывается из файла в строковую переменную, к которой в начало прибавляется сгенерированный фрагмент с определениями. Подробно такой подход демонстрируется в [32], где в константы также вынесены смещения для ускорения позиционирования по данным, то есть расстояния в памяти между соседними узлами шаблона в каждом пространственном направлении.

Существенным действием по оптимизации на ускорителе также является минимизация числа операций деления. Деление требует примерно в 5 раз больше тактов, чем умножение, к тому же, делитель на ускорителе вряд ли конвейеризован, и разница в скорости между умножением и делением ещё больше, чем на CPU. Естественно, то же самое касается и математических функций.

В [16, 17] показано, что с помощью OpenCL может быть создана универсальная реализация, эффективная на различных ускорителях, если выбирать наиболее слабую архитектуру и в первую очередь оптимизировать под неё. Так, в случае с GPU AMD и NVIDIA, оптимизация под NVIDIA даёт ядро, эффективное на AMD, но не наоборот. На операции расчёта конвективных потоков с полиномиальной реконструкцией отмечалась разница в скорости более 10 раз между AMD 7970 и NVIDIA C2050, которая после оптимизации под NVIDIA сократилась в несколько раз. Для универсальной реализации вычислительного ядра архитектурно-зависимая оптимизация сводится к выбору параметров запуска – размеров глобальной и локальной рабочей группы и т.д.

1.6 Эффективное проведение вычислительного эксперимента

1.6.1 Постановка вычислительного эксперимента

Рассматривается моделирование нестационарного течения жидкости или газа в расчётной области, геометрия которой не изменяется во времени. Вычислительная стоимость расчёта естественным образом зависит от размеров расчётной области, от разрешения по пространству, от разрешения по времени, от времени выхода течения на установившийся статистически однородный режим, от длительности периода интегрирования по времени для осреднения полей течения и получения спектральных характеристик. Основным же фактором, определяющим эффективность расчёта, является корректность постановки вычислительного эксперимента: ошибки в постановке граничных условий, в размере расчётной области, необходимом сеточном разрешении приводят к большим потерям процессорного времени и необходимости повторения дорогостоящих вычислений.

Для обеспечения корректности постановки и обоснования выбора основных параметров расчёт предлагается выполнять в три этапа на последовательности сгущающихся сеток. Необходимое пространственное разрешение определяется по числу Рейнольдса, Re, исходя из оценки шага сетки в пристенной области и в области отрывного течения из Колмогоровского масштаба. На начальном этапе исследования моделируемой конфигурации строится сетка, имеющая примерно в 4 раза меньшее разрешение по пространству (в ~ 64 раза меньшее число узлов), чем предполагаемая конечная расчётная сетка. В первом приближении размеры расчётной области выбираются с большим запасом по всем пространственным направлениям.

По полученным предварительным средним полям течения определяется размер основных структур течения, области вихрей, обратного течения и так далее, определяющие картину течения. Размер расчётной области выбирается таким образом, чтобы границы не задевали эти области, иначе возникнет нежелательная зависимость результата от размера расчётной области. Иными словами, расчётная область должна быть, с одной стороны, как можно меньше, с другой, достаточно большой, чтобы её дальнейшее увеличение не влияло существенным образом на результаты в области интереса. Также по анимации из моментальных картин течения оценивается влияние границ на область интереса, эффект от нежелательных отражений и т.д. Если присутствует периодическое направление, выполняется анализ двухточечной корреляции для оценки минимального размера по периодическому направлению. Необходимо, чтобы этот размер был больше длины корреляции турбулентных структур.

На рис. 1.9 показаны примеры ключевых параметров размера расчётной области для типичных вычислительных экспериментов фундаментального характера методом прямого численного моделирования. Например, в [42] для моделирования падающей струи в канале было показано, что выполненные ранее физические эксперименты имели недостаточные размеры каналов: граница каналов разрезает основную зону вихревого течения, что делает результаты

экспериментов ненадежными, сложно воспроизводимыми и неприменимыми для валидации моделей турбулентности.

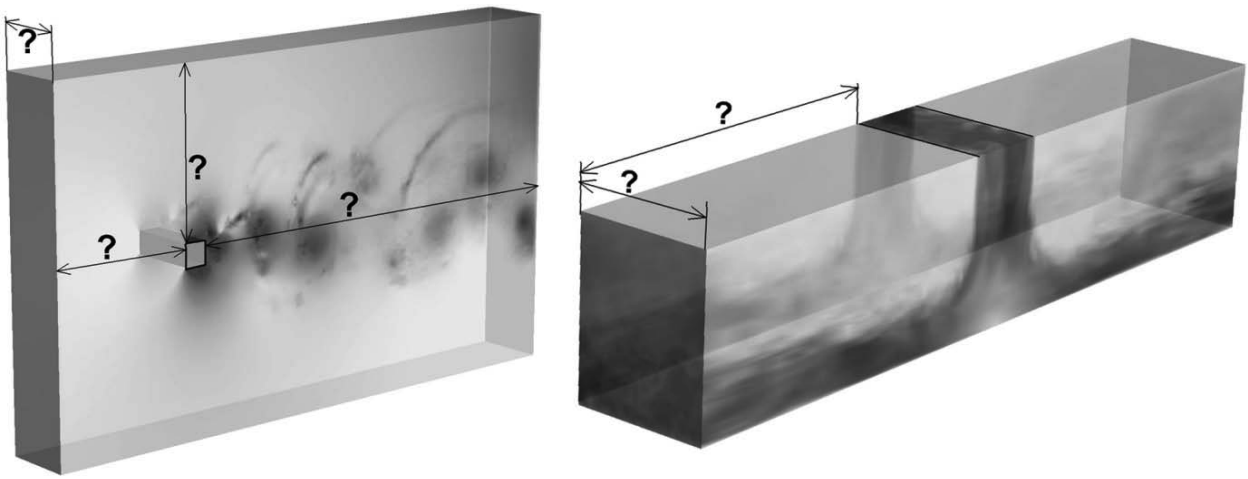


Рисунок 1.9: Примеры ключевых параметров размеров расчётной области для задачи об обтекании бесконечного квадратного цилиндра (слева) и падающей струи (справра)

Размеры расчётной области минимизируются согласно результатам предварительного расчёта посредством анализа картины течения, полученной в заведомо достаточно большой расчётной области. Например, по среднему полю течения и картине линий тока делается вывод, на каком расстоянии от обтекаемого тела наблюдается существенное влияние тела вверх по потоку, до какого места протянуты основные системы вихрей вниз по потоку и т.д. Выбор размера следует делать с запасом, поскольку на подробной сетке возможны изменения в структуре течения.

Для проверки корректности новых размеров выполняется повторный расчёт в новой геометрии, которым подтверждается, что изменение не повлияло на результат в области интереса (если результат изменился, то следует выполнить уточнённый анализ, расширить границы течения и повторить расчёт). По полученному предварительному полю выполняется оптимизация параметров сгущения сетки. На уточнённой сетке выполняется расчёт до получения средних полей течения. Апостериори анализируется развитие течения во времени, по эволюции во времени контрольных величин (например, подъемной силы, пульсаций давления в турбулентном следе, числа Нуссельта и т.д.) определяется момент установления режима течения, по сходимости средних полей определяется необходимый интервал осреднения до достижения необходимого качества осреднения. На выходе данного этапа имеется предварительная геометрия расчётной области, адаптированная к решению сетка, оценка необходимого временного интервала.

На втором этапе, выполняется равномерное сгущение сетки в 2 раза в каждом направлении, или равномерное измельчение сеточных элементов в случае неструктурированной сетки, дополненное при необходимости операцией сглаживания к твёрдой поверхности, перемещающей добавленные на твёрдые границы узлы на заданную поверхность. Для сглаживания в случае простой геометрии твёрдых тел могут применяться подходы на основе параметризации тел аналитически заданными поверхностями. В случае сложной геометрии

твёрдого тела вместо аналитической параметризации может, например, использоваться подробная триангуляция поверхности. Такая поверхностная сетка должна иметь сопоставимую или большую разрешающую способность, чем у самой подробной расчётной сетки (для треугольной поверхностной сетки можно позволить намного более высокую разрешающую способность, чем для трехмерной тетраэдральной сетки). Более подробно методика равномерного измельчения со сглаживанием описана, например, в [43].

Для ускорения установления режима течения в качестве начального поля используется моментальная картина течения, полученная интерполяцией с предыдущего этапа. Выигрыш от интерполяции решения с грубой сетки зависит от типа течения. Например, в случае естественной конвекции на вертикальных стенках в бесконечной каверне выход на установление течения занимает очень продолжительное время. При переносе решения с грубой сетки нарушается тонкий режим течения, происходит отрыв на стенке намного ниже по потоку, разрушается глобальная структура течения, и до установления режима все равно требуется много времени. При этом выигрыш от интерполяции все равно присутствует. Но, например, для конфигурации падающей струи в канале выигрыш от интерполяции может быть заметно больше. Появляющиеся в начальный момент времени вихри (рис. 1.10) достаточно долго растягиваются, пока устанавливается протяженная зона обратного течения. Перенос решения с грубой сетки ускоряет этот процесс.

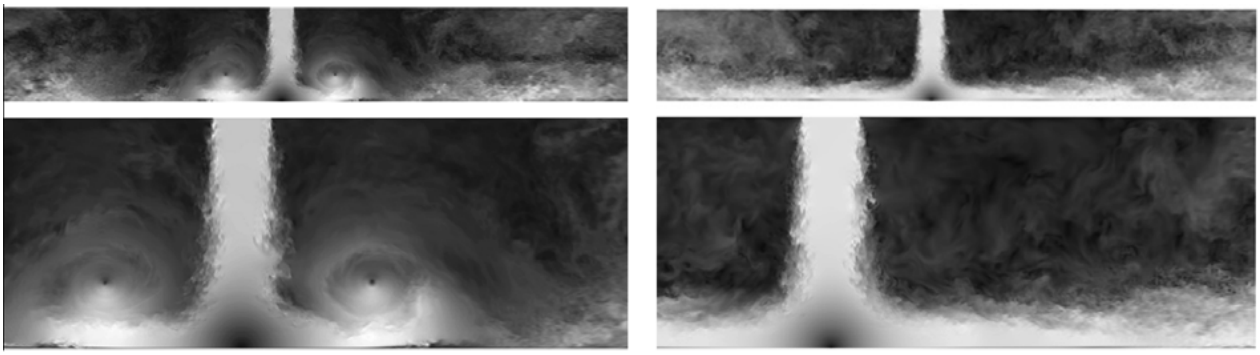


Рисунок 1.10: Начальная стадия течения (слева) и установившийся режим (справа) для падающей струи в канале (показано поле модуля скорости)

Выполняется расчёт до получения средних полей течения, попутно подбираются и проверяются оптимальные управляющие параметры расчёта (шаг по времени, невязка решателя СЛАУ и т.д.), апостериори определяется длительность периода установления режима течения при старте с полей, перенесённых с грубой сетки. Уточняется необходимый период интегрирования по времени, уточняется распределение узлов сетки согласно полученному среднему полю течения, ещё раз проверяется достаточность размеров области. По завершении данного этапа имеется обоснованный выбор геометрии и параметров сгущения сетки, подобраны и проверены параметры шага по времени и невязки решателя и т.д., известен примерный момент начала осреднения и необходимый интервал осреднения, имеется уточнённая оценка потребной вычислительной стоимости для выполнения расчёта на подробной сетке. Далее, как и на

предыдущем этапе, выполняется равномерное сгущение сетки в два раза в каждом направлении, полученные ранее моментальные поля течения на установившемся режиме интерполируется на новую сетку, основной расчёт готов к запуску.

Такой поэтапный подход существенно повышает шансы правильно поставить вычислительный эксперимент, позволяет более точно оценить необходимые вычислительные затраты и ресурсы, оптимизировать основные параметры. Наличие решений на трёх последовательно сгущённых сетках позволят оценить сходимость решения и погрешность расчёта. Кроме того, имея результаты на грубых сетках, можно выполнять расчёты на тех же сетках с использованием моделей турбулентности и показывать напрямую улучшение качества решения за счёт модели. Все это достигается сравнительно небольшой вычислительной стоимостью. Например, пять расчётов на грубой сетке и два расчёта (с учётом возможной коррекции) на средней сетке, что по опыту обычно достаточно, укладываются в 14% от стоимости расчёта на подробной сетке. Это значение следует из оценки, что расчёт на грубой сетке по стоимости составляет примерно 0.4% ($1/4^4$ – в 4 раза меньше узлов в каждом направлении, шаг по времени пропорционален шагу сетки), а на средней сетке – 6% ($1/2^4$) от стоимости расчёта на подробной сетке.

Следует отметить, что кроме вычислительной стоимости, то есть суммарно затраченного процессорного времени, расчёт имеет стоимость, выраженную в трудозатратах. Естественно, построение последовательности сеток и выполнение серии расчётов более трудоёмко. Эти дополнительные трудозатраты представляются незначительными, если речь идет о больших расчётах порядка миллионов процессорных часов. Трудозатраты на построение сетки по большей части состоят в представлении геометрии твёрдых тел и расчётной области, подборе параметров сгущения сетки и т.д. Этот достаточно трудоемкий процесс может выполняться один раз. Коррекция размеров расчётной области в сеточном генераторе существенно менее трудоёмка, а равномерное измельчение сетки вообще выполняется автоматически. Трудозатраты на обработку и анализ результатов также могут быть незначительны, если программный комплекс имеет автоматизированные средства обработки результата.

1.6.2 Выполнение вычислительного эксперимента

Определив корректную постановку задачи, включая проверенные размеры расчётной области и начальное течение, необходимо обеспечить надежное и эффективное выполнение расчёта в характерных условиях вычислительной системы общего пользования. К таким условиям можно отнести:

1. существенное время ожидания задачи в очереди при каждом запуске;
2. ограниченный квант, то есть максимальное время, доступное для одного запуска;
3. возможность остановки задачи в произвольный момент времени (по истечению кванта, в случае сбоя или внезапной профилактики);
4. ограниченную дисковую квоту;

5. относительно низкую надежность файловой системы временного хранения данных.

Из условия 1) следует необходимость минимизировать число постановок задачи в систему очередей. Из 2) и 3) следует необходимость записи точек восстановления, то есть сохранение моментальных полей течения для продолжения прерванного расчёта, с учётом того, что выполнение может быть остановлено и в момент записи точки восстановления. Из 4) следует необходимость минимизировать объём данных, хранящихся на файловой системе, а из 5) – необходимость регулярной выгрузки данных для восстановления расчёта на внешнюю файловую систему.

Представленная далее методика удовлетворяет всем указанным требованиям. Для минимизации числа постановок задачи в очередь необходимо обеспечить отсутствие аварийных остановок за счёт механизма контроля корректности и восстановления вычислений без прерывания выполнения задачи (об этом далее в разделе 4), обеспечить возможность изменения параметров расчёта без остановки задачи. Динамическое изменение параметров реализуется путём периодического повторного чтения файла параметров, в который могут вноситься новые значения для ключевых параметров, определяющих производительность и корректность вычислений.

Для надежного восстановления вычислений, остановленных в произвольный момент времени, записываются и хранятся две сменяющие друг друга промежуточные точки восстановления, то есть записи необходимых моментальных полей течения. Хранение двух записей необходимо на случай остановки задачи во время записи.

Необходимо также правильно выбрать интервал записи точек восстановления: слишком частая запись приведет к излишним затратам времени на запись файлов, слишком редкая – на повторный пересчёт потерянных при остановке шагов. Интервал записи точек восстановления N_R (т.е. количество шагов по времени, через которые сохраняется запись) предлагается выбирать исходя из кванта задачи Q, времени записи одной точки восстановления на диск t_r и времени выполнения одного шага интегрирования по времени t_c следующим образом: $N_R \approx \sqrt{(2Qt_r)}/t_c$ (предполагая, что время записи мало по сравнению со временем вычисления одного интервала).

Запись точки восстановления предлагается хранить в распределённом формате, где необходимые для восстановления расчёта моментальные поля хранятся в отдельном бинарном файле для каждой подобласти MPI процесса. Дополнительно, общая информация, содержащая номер шага по времени и другие необходимые данные, записывается в отдельный заголовочный файл записи в текстовом формате, удобном для чтения пользователем. Если осуществляется осреднение полей течения, то точка восстановления также должна содержать средние поля. Осреднённые поля предлагается хранить отдельно от моментальных полей, поскольку обрабатываются эти данные разными программными средствами. Таким образом, требуемый размер дисковой квоты для расчёта должен вмещать 2 записи восстановления моментальных полей течения и 3 записи средних полей, две из которых для восстановления осреднения, одна – для записи законченного интервала осреднения. Записи интервалов осреднения выгружаются на

внешнюю файловую систему для последующей обработки. Схема записи точек восстановления и переноса данных на внешнюю файловую систему показана на рис. 1.11.

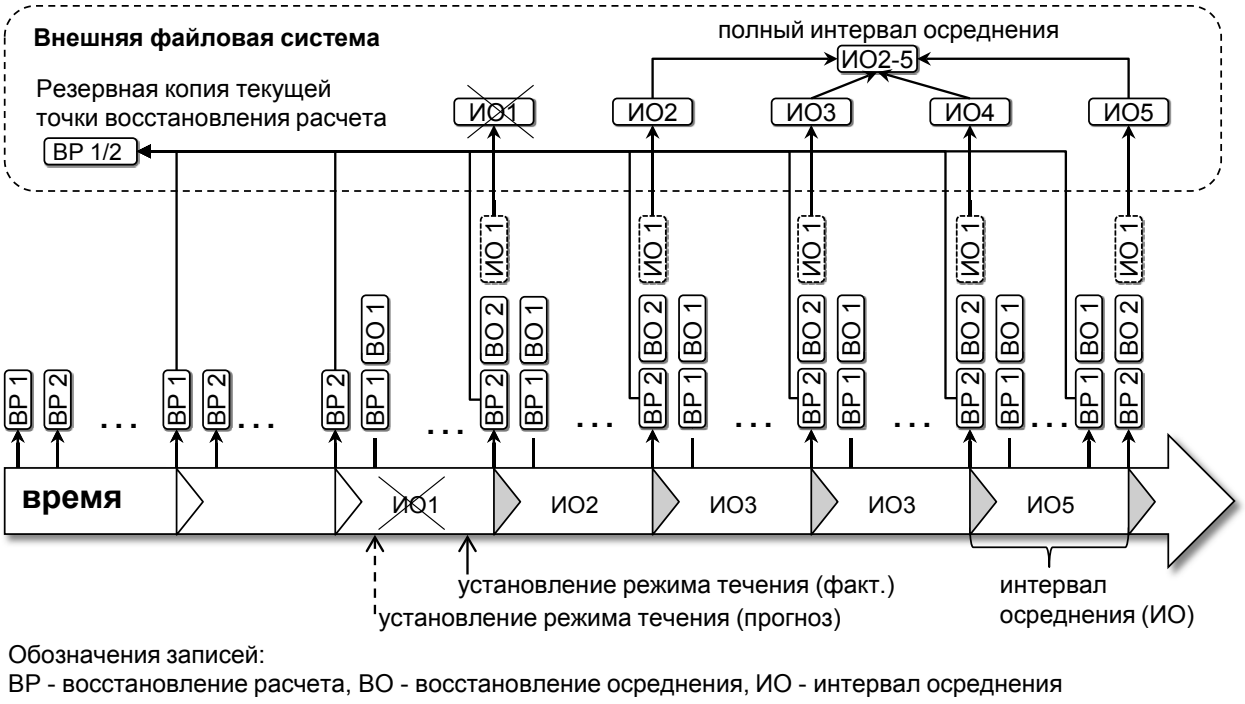


Рисунок 1.11: Схема записи точек восстановления и интервалов осреднения

Для автоматического восстановления расчёта по заголовочным файлам проверяется, какая из двух записей новее, затем проверяется, цела ли наиболее новая запись. Проверку целостности записи предлагается осуществлять простым и быстрым способом – сравнивая размер файла с размером предыдущей записи. Если у какого-то процесса размер более новой записи меньше предыдущей (или файл отсутствует), то запись считается повреждённой (была остановка задачи в момент записи) и восстановление расчёта выполняется с предыдущей записи.

1.6.3 Контроль и автоматическое управление параметрами расчёта

Для обеспечения бесперебойности вычислений, отсутствия аварийных остановок и для автоматического выбора оптимальных параметров расчёта применяется реализация быстрого восстановления расчёта из точек восстановления, хранящихся в оперативной памяти. В случае нарушения корректности вычислений такой подход позволяет выполнить возврат на более раннее время и скорректировать параметры расчёта, чтобы избежать аварийной остановки и, как следствие, длительного ожидания в очереди при последующем запуске. Чем полнее реализованы средства диагностики и критерии нарушения корректности вычислений, тем надежнее будет защищен вычислительный алгоритм. К таким критериям можно отнести:

– некорректные значения в полях течения: выход за допустимый диапазон (например, отрицательное полное давление или плотность), неопределённые значения NaN (нечисловое значение, от англ. not a number) и inf (бесконечность);

- сильное ухудшение сходимости итерационного процесса (для неявной схемы);
- отсутствие или сильное ухудшение сходимости итерационного метода решения СЛАУ (для неявной схемы или несжимаемых течений).
- нарушение законов сохранения по суммарным значениям энергии, момента и т.д.
- нарушение условия бездивергентности (для несжимаемых течений).

Данные индикаторы показывают либо необходимость аварийной остановки, либо её неизбежность в ближайшее время. В случае нарушения условий корректности выполняется отступление на точку восстановления в оперативной памяти для автоматической коррекции параметров расчёта. К таким параметрам, непосредственно влияющим на производительность вычислений и качество результата, можно отнести:

- величину шага по времени;
- критерий невязки ньютоновского процесса (для неявной схемы);
- критерий невязки решателя СЛАУ (для неявной схемы или несжимаемых течений);
- весовой коэффициент между бездиссипативной центрально-разностной схемой и противопотоковой схемой с численной диссипацией (при LES моделировании).

В зависимости от используемых методов список параметров может варьироваться.

Для работы режима автоматической коррекции вычислений предлагается использовать два интервала отступления и, соответственно, 4 точки восстановления в оперативной памяти: по две точки для короткого интервала, сменяющие друг друга каждые n_s шагов по времени, и по две точки для длинного интервала из n_l шагов. Тут две записи на интервал нужны, чтобы всегда был доступен полный интервал, и не возникало ситуации, когда некуда отступать. Величины интервалов выбираются эмпирически из практики расчётов для конкретного алгоритма, но исходя из условий $n_s \ll n_l \ll n_q$, где n_q – число шагов, выполняемое за один квант запуска задачи.

Алгоритм работы состоит в возвращении на ближайший доступный короткий интервал для коррекции параметров расчёта (уменьшение шага по времени, уменьшение невязки решателя СЛАУ и т.д.) и попытке продолжить вычисления. Если восстановление с короткого интервала не получается, выполняется отход на длинный интервал. Естественно, для исключения ненужного расхода процессорного времени на бесконечном повторении коррекции параметров, необходим рациональный критерий остановки расчёта для случая, если восстановление все-таки невозможно. Таким образом, при нарушении корректности вычислений, алгоритм автоматически пытается скорректировать параметры с короткого интервала, если не удаётся, то с длинного интервала, и если опять не удаётся, то уже происходит аварийная остановка.

Параметры расчёта могут автоматически оптимизироваться, меняясь со временем в сторону снижения вычислительной стоимости (например, регулярно растет шаг по времени, уменьшается невязка решателя и т.д.), и, если нарушается корректность, автоматически выполняется возврат и коррекция параметров.

Режим автоматического отступления с коррекцией параметров также применяется при “старте с протяжки”, когда начальное распределение полей течения, не соответствующее картине установившегося течения, сначала “протягивается” и разглаживается численной схемой с большой диссипацией или стационарным RANS методом, чтобы избежать больших градиентов сеточных функций и аварийной остановки алгоритма в начальные моменты времени. Поскольку заранее неизвестно, в какой момент начальное приближение достаточно разглажено, автоматически выполняются попытки переключиться на схему повышенного порядка и сэкономить время на “протяжке” течения.

1.6.4 Статистика течения

Осреднение полей течения должно выполняться после момента установления статистически однородного режима течения, который сложно точно определить априори. Осреднение осуществляется путём интегрирования по времени, то есть суммирования значений полей, умноженных на шаг по времени. Для экономии времени вычислений накопление может осуществляться не на каждом шаге по времени, а через некоторое число шагов, если это не ухудшает качество результата. При этом если усредняются производные по времени, значения производных должны, естественно, браться от ближайших шагов по времени (без пропусков), чтобы не терялась точность.

Осреднение предлагается выполнять частичными интервалами, чтобы избежать порчи данных из-за попадания в осреднение неустановившегося режима течения: потом, когда будет определен момент выхода течения на статистически однородный режим, испорченные интервалы можно отбросить. Обработка записей осреднения в распределённом формате выполняется в параллельном режиме в следующем порядке:

1. сборка осреднения: суммирование частичных интервалов в обратном порядке, исключение интервалов, попавших в неустановившийся режим;
2. получение средних значений путём деления на величину интервала по времени;
3. расчёт дополнительных полей, вычислимых по имеющимся средним полям;
4. пространственное осреднение по изотропному направлению (если есть);
5. пространственное осреднение по симметрии постановки задачи (если есть).

Осреднение по пространству в присутствии периодики уменьшает размерность полей и существенно улучшает качество стационарных характеристик течения. Осреднение с учётом симметрии течения также существенно повышает качество. Одна пространственная симметрия примерно эквивалентна сокращению вдвое интервала интегрирования по времени при таком же качестве результата. На рис. 1.12 показан пример пространственного осреднения для течения в квадратной трубе: осреднение по периодике, осреднение по четырём направлениям симметрии.

В случае неструктурированных сеток для упрощения осреднения по пространству при наличии симметрии предлагается стоить сетку для одной части симметричной области, затем

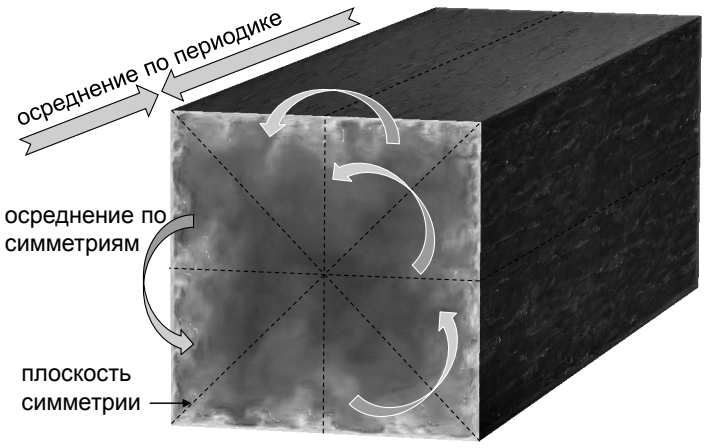


Рисунок 1.12: Осреднение по пространству для течения в бесконечной квадратной трубе

отражением части сетки и сшивкой получать сетку для всей области. Так получится прямое соответствие симметричных узлов друг другу и будет исключена необходимость интерполяции.

1.6.5 Динамические данные

Динамические данные предназначены для получения спектральных характеристик течения и представляют собой запись эволюции во времени:

1. физических величин в контрольных точках;
2. глобальных величин (подъемная сила, коэффициент сопротивления, моменты и т.д.);
3. поля течения на контрольных поверхностях – в задачах аэроакустики (для моделей распространения акустических возмущений в дальнем поле).

Обычно записываются пульсации скорости, которые характеризуют турбулентные характеристики течения, и пульсации давления, которые дают информацию о генерируемом аэродинамическом шуме.

Для получения качественных спектральных характеристик требуется достаточно большая выборка, то есть длительный период интегрирования по времени, длина которого обратно пропорциональна нижней границе необходимого частотного диапазона. Экономия достигается за счёт получения результата необходимой точности за как можно более короткий интервал интегрирования по времени. Техника обработки сигналов основана на методе периодограмм [44]. Из требований к необходимому частотному разрешению спектра выбирается подходящая оконная функция [45]. Далее исходный сигнал делится на части с возможными перекрытиями. Из каждого сегмента вычитается интегральное среднее значение и/или линейный тренд. На следующем этапе полученные части умножаются на выбранную оконную функцию, после чего выполняется преобразование Фурье. В конце производится осреднение полученных данных для каждой дискретной частоты.

В задачах аэроакустики для расчёта шума в дальнем поле дополнительно используется интегральный метод Фокса – Уильямса и Хокингса FW/H [46], основанный на акустической

аналогии Лайтхилла. В рамках данного подхода получение пульсаций давления в точках дальнего поля происходит на основе нестационарных данных с контрольной поверхности, накапливаемых в течение расчёта. Реализация такого подхода на неструктурированных сетках представлена, например, в [47]. Необходимо отметить, что качество получаемых спектров достаточно сильно зависит от правильного выбора поверхности для интегрирования (см., например, [48]), а также от точности численного поверхностно-временного интегрирования.

1.7 Выбор конфигурации распараллеливания

1.7.1 Выбор количества вычислительных ресурсов

Число процессоров (и ускорителей) выбирается, исходя из следующих факторов: зависимость времени ожидания в очереди от запрошенного числа процессоров; ограничение снизу объёмом оперативной памяти узла или ускорителя; ограничение сверху фактической параллельной эффективностью расчётного кода на данной задаче и на данной системе. Также учитывается приоритет между сроками выполнения расчёта и экономией процессорного времени. Затем необходимо выбрать конфигурацию распараллеливания, наиболее подходящую для данной задачи и выбранных вычислительных ресурсов.

1.7.2 Двухуровневое MPI+OpenMP распараллеливание

Конфигурацию MPI+OpenMP распараллеливания и узлов суперкомпьютера предлагается описывать набором параметров, представленных в таблице 1.1.

Параметр	Описание
N_P	– число MPI процессов (Number of Processes)
N_T	– числом OpenMP нитей на MPI процесс (Number of Threads)
P_N	– числом MPI процессов на узел суперкомпьютера (Processes per Node)
N_S	– количество процессоров (Number of Sockets)
N_C	– количество ядер в каждом процессоре (Number of Cores)
T_C	– количество нитей на ядро (Threads per Core)

Таблица 1.1: Параметры MPI+OpenMP распараллеливания и узлов суперкомпьютера

Выбор конфигурации состоит в балансе между MPI и OpenMP составляющими. При запуске одного MPI процесса на многопроцессорный узел, все ядра узла задействуются средствами OpenMP. С одной стороны, это может быть плохо, поскольку, как правило, происходит падение производительности из-за неоднородного доступа к общей памяти между процессорами и накладных расходов на поддержание когерентности кэш; с другой стороны, хорошо, поскольку снижается объём обмена данными за счёт уменьшения числа MPI процессов. Выбор, соответственно, зависит от того, какой из факторов окажет большее влияние. Чем больше узлов суперкомпьютера используется, то есть чем больше вес обменов данными, тем

более выгодно становится использовать OpenMP распараллеливание, сокращающее количество обменов данными. Однако на небольшом числе процессоров MPI распараллеливание может работать более эффективно. Этот баланс факторов следует проверять эмпирически для конкретного программного комплекса на конкретном типе вычислительной системы. Для упрощения предлагается использовать на выбор 3 конфигурации, в которых размещается по одному MPI процессу:

1. на ядро: число нитей $N_T = 1$, число процессов на узел $P_N = N_S \times N_C$;
2. на процессор: $N_T = N_C$, $P_N = N_S$;
3. на узел суперкомпьютера: $N_T = N_S \times N_C, P_N = 1$.

Первая конфигурация подходит для небольших расчётов (особенно для неявной схемы), вторая – для расчётов на большем числе процессоров, третья – для расчётов на предельных числах процессоров близко к исчерпанию параллелизма. При использовании режима одновременной многопоточности ядер (CPU с Hyper-threading – 2 нити на ядро, Intel Xeon Phi – 4 нити) N_T увеличивается в T_C раз.

На рис. 1.13 показан пример выбора конфигурации распараллеливания на суперкомпьютере МВС-10П в зависимости от общего числа ядер, которое планируется задействовать. Небольшой тестовый расчёт кодом [49] моделирует обтекание обратного уступа на сетке из 1.3 млн узлов, неявная схема, число Рейнольдса 28000. Показано процессорное время на расчёт одного шага по времени, то есть фактическое время, умноженное на число задействованных ядер. Для удобства время нормировано по времени наиболее быстрой конфигурации для одного узла, которое принято за единицу. Рост процессорного времени с увеличением числа процессоров отображает рост накладных расходов распараллеливания, в частности, на обмен данными. Видно, что на меньшем числе узлов лучше конфигурация 1, на 4-х узлах (64 ядра) конфигурации 1 и 2 совпадают, на большем числе узлов лучше конфигурация 2.

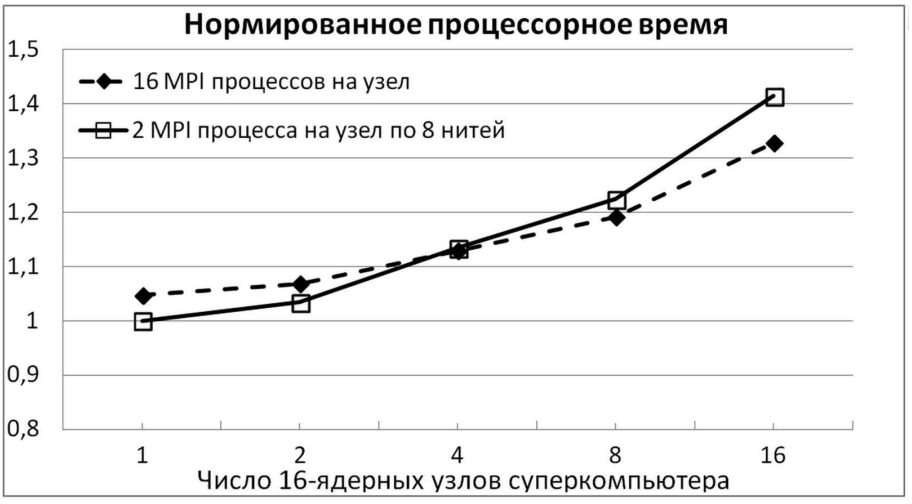


Рисунок 1.13: Суммарное процессорное время (число ядер, умноженное на время выполнения задачи), нормированное по времени лучшей конфигурации на одном узле

1.7.3 Гибридные вычисления на массивно-параллельных ускорителях

Специфика при расчётах на суперкомпьютере с ускорителями Intel Xeon Phi в симметричном режиме заключается в необходимости балансировки загрузки между процессором и ускорителем, в некотором усложнении MPI распараллеливания, а также в изменениях при чтении/записи данных на диск. Для загрузки гибридного узла предлагается размещать по одному процессу на CPU и по одному (или более) процессу на Phi. Обмен данными для процесса на ускорителе существенно замедляется, поскольку передача данных выполняется сначала копированием на хост по шине PCI-E, а затем уже по сети суперкомпьютера. В связи с этим предлагается использовать двухуровневую декомпозицию подобласти: сначала подобласть разделяется по узлам суперкомпьютера, затем каждая подобласть узла разбивается на подобласти второго уровня по вычислительным устройствам узла. Внутри процессоров и ускорителей работает OpenMP распараллеливание. При таком подходе объём обмена данными по сети между узлами будет таким же, как если бы загрузка была по одному процессу на узел, а большая часть обменов данными останется внутри узла. На втором уровне можно минимизировать обмены по сети для Phi, разбив подобласть узла так, чтобы подобласти 2-го уровня, соответствующие ускорителям, имели как можно меньше внешних границ или не имели их вовсе. Для неструктурированной сетки это можно, например, сделать при декомпозиции взвешенного графа подобласти узла, задав очень большие веса ребрам, соединяющим интерфейсные вершины, то есть вершины, соответствующие ячейкам сетки, связанным шаблоном численной схемы с ячейками из подобластей других узлов суперкомпьютера. Тогда алгоритм декомпозиции будет избегать разрезания таких ребер, то есть стараться не выпускать внутренние подобласти наружу.

Балансировка загрузки сводится к получению подобластей с разным числом ячеек, пропорционально соотношению производительности процессора и ускорителя. Проблема состоит в том, что зачастую разные операции алгоритма имеют разное соотношение производительности. Если между такими операциями присутствуют точки синхронизации в виде обмена данными (в матрично-векторном произведении, скалярном произведении и т.д.), то потери на дисбалансе неизбежны, поскольку устройство, быстрее выполняющее операцию, будет простаивать, ожидая обмена.

Аналогичная ситуация и в случае графических процессоров GPU, если задействовать в вычислениях одновременно ускорители и процессоры. Проблема решается также декомпозицией второго уровня с учётом балансировки между CPU и ускорителями. В отличие от Phi, GPU не выполняет независимый MPI процесс, а управляется из MPI процесса, выполняющегося на CPU. Поэтому число MPI процессов на узел берется равным числу GPU на узле, далее на втором уровне работа распределяется между ядрами процессора и ускорителем. Другой вариант – загружать GPU и CPU разными процессами: одни процессы используют одно ядро CPU и одно GPU устройство, а другие используют все оставшиеся CPU ядра.

Чтение и запись на диск с ускорителя Phi выполняется существенно медленнее, чем с процессора. Для ускорения чтения/записи данных на диск предлагается объединять

процессы, выполняющиеся на одном узле, в подгруппу, и записывать файлы полей течения в распределённом формате для декомпозиции первого уровня. Главный процесс подгруппы собирает данные с остальных процессов путём обмена данными внутри узла, и выполняет запись общего файла для подобласти первого уровня. На системах с GPU вычисления на ускорителе в любом случае управляются из MPI процесса, выполняющегося на CPU, и запись данных идет через хост.

Выводы по главе

В главе была представлена технология разработки и программной реализации в рамках многоуровневой параллельной модели конечно-объёмных и конечно-разностных алгоритмов для математического моделирования задач механики сплошной среды на широком спектре вычислительных архитектур, включая гибридные суперкомпьютеры. Кратко её можно сформулировать следующим образом. Сначала алгоритм разделяется на базовые операции. Затем, исходя из зависимости по данным между операциями, на первом уровне реализуется традиционное MPI распараллеливание. Далее, работая с операциями в отдельности, выполняется адаптация к потоковой обработке и различным вычислительным архитектурам, оптимизация структур данных и доступа к памяти. Совокупность представленных подходов для каждого из этих этапов формирует готовый рецепт создания эффективной программной реализации для гетерогенных вычислений. Эффективность понимается в широком смысле, как с точки зрения производительности и параллельной эффективности, так и с точки зрения надежности и удобства работы с кодом. Технология, применение которой демонстрируется далее в следующих главах, подходит для широкого класса численных методов для моделирования задач аэродинамики, аэроакустики, тепломассопереноса.

В главе в деталях рассмотрена комплексная методика выполнения крупных суперкомпьютерных расчётов задач газовой динамики на широком спектре вычислительных систем, включая гибридные суперкомпьютеры. Совокупность представленных подходов, среди которых есть и очевидные, и широко известные, и оригинальные, формирует готовый рецепт эффективного вычислительного эксперимента. Представленная технология отработана и успешно применена в расчётах различных задач, моделирующих сжимаемые и несжимаемые турбулентные течения. Выполненным по этой технологии расчётам посвящена 6-я глава данной работы.

Глава 2

Параллельный алгоритм повышенной точности для расчётов задач аэродинамики и аэроакустики на неструктурированных сетках

Вводные замечания

Данная глава посвящена формулировке параллельного вычислительного алгоритма из фиксированного набора базовых операций, каждая из которых на верхнем уровне совместима с MIMD параллелизмом как распределённой, так и с общей памятью, а на нижнем уровне совместима с парадигмой потоковой обработки и SIMD параллелизмом.

В основе представленного параллельного алгоритма лежат существующие математические модели и численные методы повышенного порядка аппроксимации на неструктурированных гибридных сетках. Ключевым в них является использование экономичных схем повышенного порядка с квазиодномерной реконструкцией, в формулировках как с определением сеточных функций в узлах, так и с определением сеточных функций в центрах сеточных элементов. В этом контексте раздел 2.1, описывающий существующие математические модели и численные методы, лежащие в основе алгоритма, не является результатом работы, а является представлением исходных данных для задачи построения параллельного алгоритма. Раздел 2.1 использует материал, представленный в работах разработчиков численных схем – соавторов программного комплекса NOISETTE Т. К. Козубской, И. В. Абалакина, П. А. Бахвалова, А. П. Дубеня. При этом везде по тексту присутствуют ссылки на исходные работы.

2.1 Математическая модель и численный метод

2.1.1 Уравнения Навье-Стокса

Течение сжимаемого вязкого газа описывается системой безразмерных уравнений Навье-Стокса, которая в дивергентной форме (в виде законов сохранения) относительно вектора консервативных переменных $\mathbf{Q} = (\rho,\, \rho u,\, \rho v,\, \rho w,\, E)^T$- плотности, трёх компонент импульса и полной энергии, имеет вид

$$\frac{\partial \mathbf{Q}}{\partial t} + \frac{\partial \mathbf{F}_1(\mathbf{Q})}{\partial x} + \frac{\partial \mathbf{F}_2(\mathbf{Q})}{\partial y} + \frac{\partial \mathbf{F}_3(\mathbf{Q})}{\partial z} = \frac{\partial \mathbf{F}_1^{NS}(\mathbf{Q})}{\partial x} + \frac{\partial \mathbf{F}_2^{NS}(\mathbf{Q})}{\partial y} + \frac{\partial \mathbf{F}_3^{NS}(\mathbf{Q})}{\partial z}. \qquad (2.1)$$

Конвективные потоки:

$$\mathbf{F}_1(\mathbf{Q}) = (\rho u, \rho u^2 + p, \rho uv, \rho uw, u(E+p))^T,$$
$$\mathbf{F}_2(\mathbf{Q}) = (\rho v, \rho vu, \rho v^2 + p, \rho vw, v(E+p))^T,$$
$$\mathbf{F}_3(\mathbf{Q}) = (\rho w, \rho wu, \rho wv, \rho w^2 + p, w(E+p))^T,$$

где полная энергия $E = \rho(u^2+v^2+w^2)/2 + \rho\varepsilon$, ε – внутренняя энергия.

Вязкие потоки:

$$\mathbf{F}_1^{NS} = (0, \tau_{xx}, \tau_{xy}, \tau_{xz}, u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + q_x)^T,$$
$$\mathbf{F}_2^{NS} = (0, \tau_{yx}, \tau_{yy}, \tau_{yz}, u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + q_y)^T,$$
$$\mathbf{F}_3^{NS} = (0, \tau_{zx}, \tau_{zy}, \tau_{zz}, u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + q_z)^T.$$

Компоненты вязкого тензора напряжений τ_{ij} и вектора теплового потока q_i имеют вид:

$$\tau_{xx} = \frac{\mu}{\mathrm{Re}}\left(2\frac{\partial u}{\partial x} - \frac{2}{3}\mathrm{div}\,\mathbf{u}\right), \quad \tau_{yy} = \frac{\mu}{\mathrm{Re}}\left(2\frac{\partial v}{\partial y} - \frac{2}{3}\mathrm{div}\,\mathbf{u}\right), \quad \tau_{zz} = \frac{\mu}{\mathrm{Re}}\left(2\frac{\partial w}{\partial z} - \frac{2}{3}\mathrm{div}\,\mathbf{u}\right),$$

$$\tau_{xy} = \tau_{yx} = \frac{\mu}{\mathrm{Re}}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right), \quad \tau_{xz} = \tau_{zx} = \frac{\mu}{\mathrm{Re}}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right), \quad \tau_{yz} = \tau_{zy} = \frac{\mu}{\mathrm{Re}}\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right),$$

$$q_x = \frac{\gamma}{Pr(\gamma-1)}\frac{\partial T}{\partial x}, \quad q_y = \frac{\gamma}{Pr(\gamma-1)}\frac{\partial T}{\partial y}, \quad q_z = \frac{\gamma}{Pr(\gamma-1)}\frac{\partial T}{\partial z},$$

где $\mathbf{u} = (u,v,w)$.

Система уравнений (2.1) замыкается уравнением состояния совершенного газа – $p = \rho\varepsilon\,(\gamma-1)$, где γ есть показатель адиабаты.

Граничные условия:

На твёрдой поверхности ставятся граничные условия прилипания $\mathbf{u} = 0$ и условия адиабатической $\left(\dfrac{\partial T}{\partial \mathbf{n}_B} = 0\right)$ или изотермической $(T = T_w)$стенки. На плоскостях симметрии - граничные условия отражения $(\mathbf{u}\cdot\mathbf{n}_B = 0)$. Значение T_w – заданная температура стенки и $\mathbf{n_B}$ – внешний вектор нормали к границе.

2.1.2 Конечно-объёмная пространственная дискретизация

Согласно [50], пространственная дискретизация системы уравнений Навье-Стокса выполняется на двухмерной или трёхмерной неструктурированной гибридной сетке. В двухмерном случае в качестве сеточных элементов будем допускать треугольники и четырёхугольники, в трёхмерном случае – тетраэдры, треугольные призмы, четырёхугольные пирамиды и гексаэдры. Для аппроксимации законов сохранения расчётная область представляется в виде множества контрольных объёмов – *ячеек*, в каждом из которых задан набор сеточных переменных. Общая граница между двумя ячейками будет называться *сегментом*. Проинтегрировав 2.1 по i-й ячейке, и перейдя по теореме Гаусса-Остроградского к поверхностному интегралу, закон сохранения для системы 2.1 имеет вид:

$$\frac{d\overline{\mathbf{Q}}_i}{dt} + \frac{1}{|C_i|} \oint\limits_{\partial C_i} (\mathcal{F} - \mathcal{F}^{\mathcal{NS}}) \cdot \mathbf{n}\, ds = 0, \tag{2.2}$$

где C_i обозначает i-ю ячейку, $|C_i|$ – её объём, ∂C_i – поверхность границы ячейки, $\mathbf{n}$ – внешняя нормаль к поверхности, $\overline{\mathbf{Q}}_i$ – среднее интегральное значение $\mathbf{Q}$ в ячейке, $\mathcal{F} = (\mathbf{F_1}, \mathbf{F_2}, \mathbf{F_3})$ – конвективные потоки, $\mathcal{F}^{\mathcal{NS}} = \left(\mathbf{F_1^{NS}}, \mathbf{F_2^{NS}}, \mathbf{F_3^{NS}}\right)$ – вязкие потоки.

Для элементно-центрированной схемы, то есть когда значения сеточных функций определяются в центрах сеточных элементов, ячейками являются сами сеточные элементы. Как правило, в качестве центра элемента выбирается его центр масс. Сегментами являются грани сеточных элементов. Сегмент ij – это общая грань i-го и j-го элементов.

В случае вершинно-центрированной схемы, то есть когда значения сеточных функций определяются в узлах сетки, ячейки строятся вокруг сеточных узлов. Сегментам в этом случае сопоставлены ребра сетки. Сегмент ij – это общая граница контрольных объёмов i-го и j-го узла, которая может иметь многогранную форму.

Ячейки, соединённые сегментом, будем называть соседними. Обозначим как $\mathbf{S}_{ij}$ вектор площади, который вычисляется как векторная сумма ориентированных в сторону j-й ячейки площадей граней, составляющим сегмент между i-й и j-й ячейками.

В случае вершинно-центрированной схемы ячейки могут строиться различными способами. Например, для тетраэдральной сетки каждый из тетраэдров определённым образом разбивается на 4 части, которые включаются в 4 ячейки, соответствующие вершинам тетраэдра. Тетраэдральная сетка представляет собой разбиение расчётной области Ω с границей Γ на тетраэдры: $\Omega = \bigcup\limits_{j=1}^{N_T} T_j$, где N_T – число тетраэдров. Контрольные объёмы C_i строятся вокруг каждого узла сетки i, причём $\Omega = \bigcup\limits_{j=1}^{N_P} C_j$, где N_P – число узлов сетки, следующим образом. Каждое ребро сетки разрезается на две части по середине. Далее, в каждом тетраэдре некоторым образом, который зависит от конкретного способа построения ячейки, выбирается центр тетраэдра, также на каждой грани тетраэдра определённым образом выбирается центр грани. Например, ячейки могут быть следующих типов:

– медианные, если в качестве центра треугольника и тетраэдра выбирается центр масс;

– ортоцентрические – центр описанной окружности и сферы (если эта точка лежит внутри элемента, иначе берется центр наибольшего ребра или грани, соответственно).

Центр тетраэдра является общей точкой границ четырёх его контрольных объёмов, а отрезки, соединяющие центр тетраэдра и центры его граней – общей границей трёх контрольных объёмов, построенных вокруг вершин грани. У двух тетраэдров, имеющих общую грань, центр общей грани должен совпадать. Рассмотрим один из тетраэдров с вершиной в узле i. Разделим его поверхностью, которая образуется пересечением плоскостей проходящих через следующие точки (рис. 2.1 слева) M_1, M_2, M_3 – середины ребер тетраэдра, содержащих узел i; G_1, G_2, G_3 – центры граней, содержащих узел i; G – центр тетраэдра. Разбиение такой поверхностью определяет многогранник с вершиной в узле i. Тогда контрольный объём вокруг узла i представляет собой объединение таких многогранников, имеющих общий узел i.

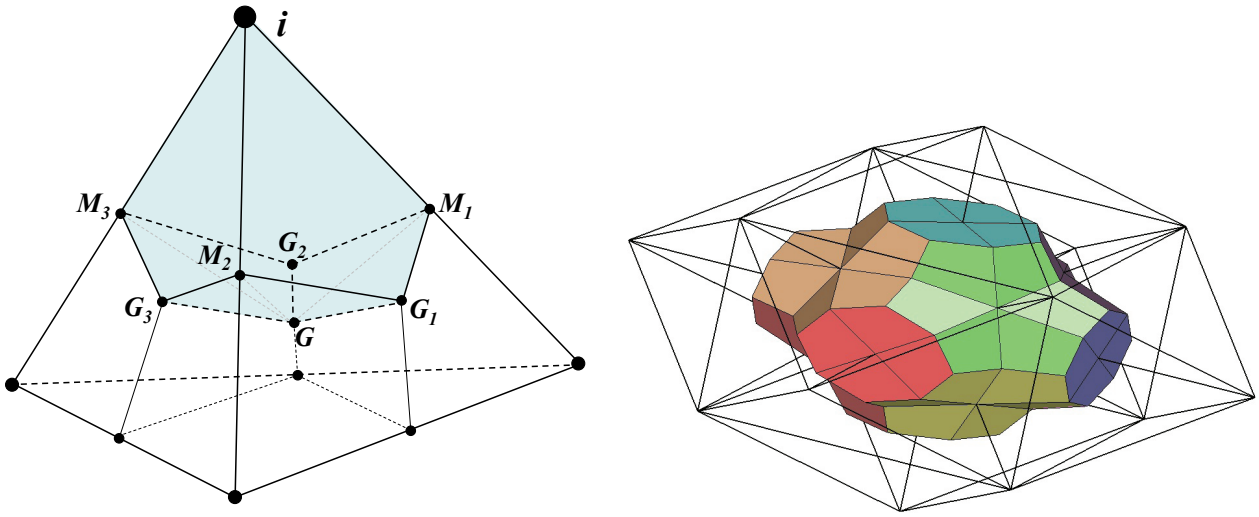


Рисунок 2.1: Часть ячейки C_i вокруг j-го узла (слева) и вид медианной ячейки (справа)

Поверхность ячейки может быть представлена как объединение сегментов, инцидентных ячейке, $\partial C_i = \bigcup\limits_{k \in N_1(i)} \partial C_{ik}$, где $N_1(i)$ – это набор ячеек, соседних с данной, а ∂C_{ik} обозначает поверхность сегмента ik.

Для аппроксимации (2.2) заменим интегральное значение $\overline{\mathbf{Q}}_i$ на точечное значение $\mathbf{Q_i}$, определённое в центре ячейки. Далее, поверхностные интегралы от конвективных и вязких потоков представим в виде суммы по сегментам:

$$\oint\limits_{\partial C_i} \left(\mathcal{F} - \mathcal{F}^{\mathcal{NS}}\right) \cdot \mathbf{n}\, ds = \sum_{k \in N_1(i)} \int\limits_{\partial C_{ik}} \left(\mathcal{F} - \mathcal{F}^{\mathcal{NS}}\right) \cdot \mathbf{n}\, ds \approx \sum_{k \in N_1(i)} \left(\mathbf{h}_{ik} - \mathbf{h}_{ik}^{NS}\right),$$

где $\mathbf{h}_{ik}$ и $\mathbf{h}_{ik}^{NS}$– это численные потоки, аппроксимирующие конвективные и вязкие потоки через сегмент ik, соответственно. Способ вычисления этих потоков определяется конкретной используемой численной схемой. Численные потоки $\mathbf{h}_{ik} = \mathcal{F}_{ik} \cdot \mathbf{n}_{ik}$, а в элементно-

центрированном случае и $\mathbf{h}_{ik}^{NS} = \mathcal{F}_{ik}^{NS} \cdot \mathbf{n}_{ik}$, берутся в точке, которая в случае вершинно-центрированной схемы лежит на середине ребра, соединяющего узлы i и k, а в случае элементно-центрированной схемы – на центре масс грани, разделяющей ячейки i и k. Здесь $\mathbf{n}_{ik} = \int\limits_{\partial C_{ik}} \mathbf{n}\, ds$ – это ориентированная площадь сегмента ik. Численный поток $\mathbf{h}_{ik}^{NS}$ для вершинно-центрированного случая далее будет рассмотрен отдельно.

Таким образом, численная схема и в элементно-центрированном и в вершинно-центрированном случае приводится к общему виду

$$\left(\frac{d\mathbf{Q}}{dt}\right)_i = -\frac{1}{|C_i|} \sum_{k \in N_1(i)} \left(\mathbf{h}_{ik} - \mathbf{h}_{ik}^{NS}\right). \tag{2.3}$$

2.1.3 Определение конвективного численного потока

Конвективный поток $\mathcal{F}_{ij}$ через сегмент ij вычисляется путём решения задачи Римана о распаде произвольного разрыва по схеме годуновского типа относительно предраспадных реконструированных значений “слева” и “справа” от центра сегмента, который в вершинно-центрированном случае находится в центре ребра между узлами i и j, а в элементно-центрированном случае – на центре грани, разделяющей ячейки i и j. В качестве схемы для решения задачи Римана может, в частности, использоваться схема Роу [51] или противопотоковая схема [52].

Для реконструкции значений “слева” и “справа”‘ от центра сегмента, обозначенных $\mathbf{Q}_{ij}$ и $\mathbf{Q}_{ji}$ на рис. 2.2, соответственно, используется подход квазиодномерной реконструкции. Так называемые EBR схемы (edge-based reconstruction) используют одномерный 6-точечный шаблон, находящийся на прямой, проходящей через ребро между узлами i и j в вершинно-центрированном случае (рис. 2.2). Две точки шаблона берутся напрямую из ячеек i и j, а для нахождения остальных 4-х значений используется некоторый способ интерполяции, зависящий от типа схемы.

Например, в многопараметрической схеме [31, 53] шаблон реконструкции для интерполяции значений использует два “противопотоковых” сеточных элемента – треугольника или тетраэдра, содержащих узлы, разделяемые сегментом, и пересекаемые линией реконструкции (см. рис. 2.3 слева). Также используются узловые градиенты, рассчитанные в каждом узле этих элементов путём суммирования градиентов по всем сеточным элементам, содержащим данный узел. Более подробно о данном типе схем см. в [31, 50, 53]. В модификации этой схемы в [12, 54] вместо узловых градиентов используются “противопотоковые” элементы 2-го уровня, т.е. элементы, содержащие узлы тех двух “противопотоковых” элементов 1-го уровня, отличные от i и j и пересекаемые линией, параллельной линии реконструкции, исходящей из узлов в направлении от сегмента ij (см. рис. 2.3 по центру). Вид такой двухуровневой интерполяционной конструкции в трёхмерном случае показа на рис. 2.4. Более новая экономичная версия EBR схемы [55] использует по 2 грани элементов с каждой стороны, как показано на рис. 2.3 справа.

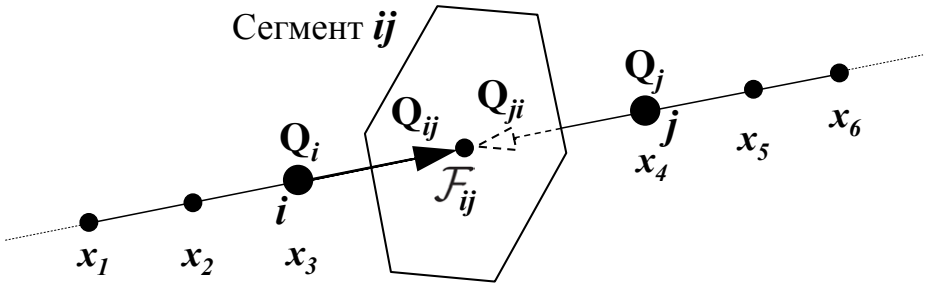


Рисунок 2.2: Квазиодномерная реконструкция значений “слева” и “справа” от центра сегмента по 6-точечному шаблону вершинно-центрированной EBR схемы

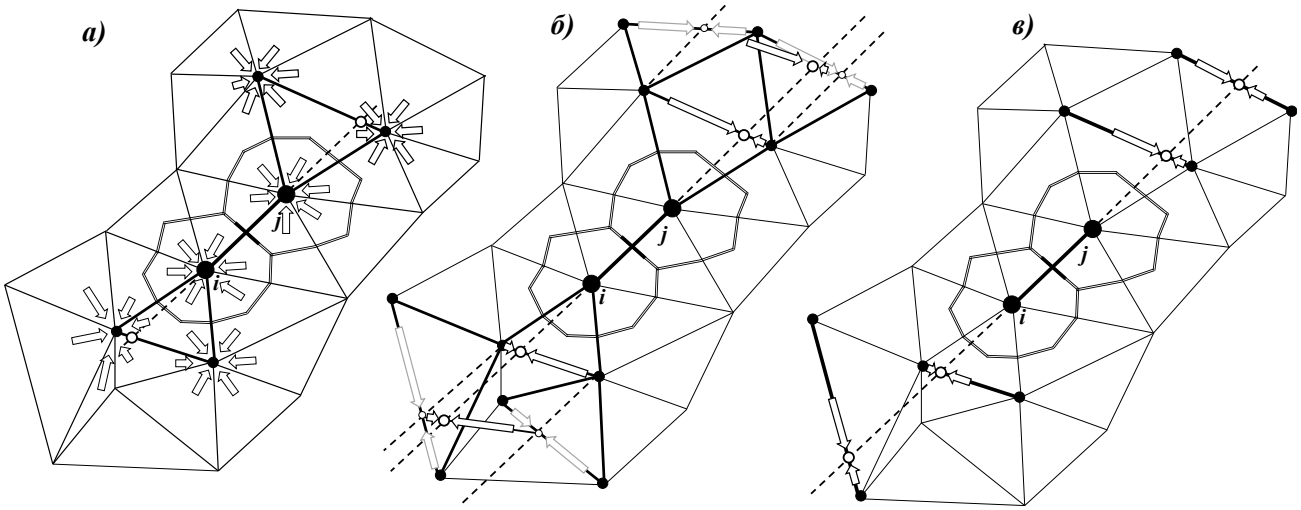


Рисунок 2.3: Расширенный шаблон для схем высокого порядка точности для EBR схемы с использованием узловых градиентов [31] (слева), для схемы с двухуровневой интерполяционной конструкцией [12] (по центру), для экономичной EBR схемы [55] (справа)

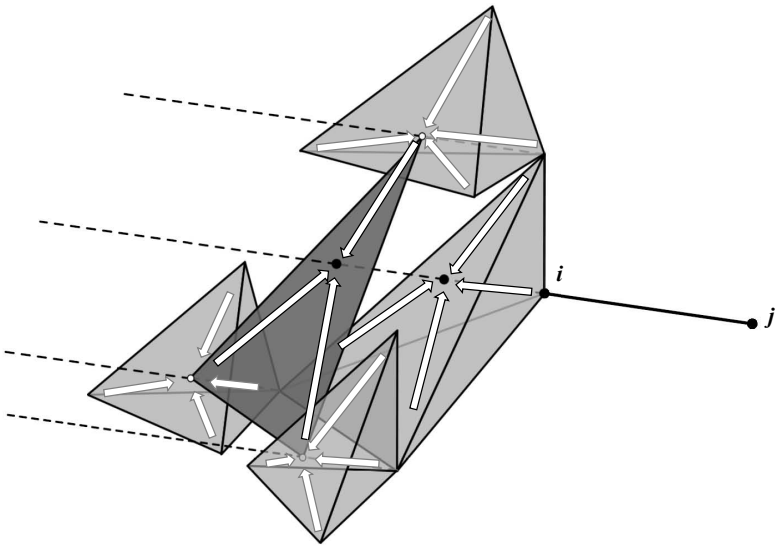


Рисунок 2.4: Двухуровневая интерполяционная конструкция на тетраэдральной сетке

В элементно-центрированном применяется аналогичный подход, но используются две прямые реконструкции: одна прямая, проходящая через центр ячейки j и центр сегмента (грани) ij, используется для нахождения $\mathbf{Q}_{ij}$, вторая прямая, проходящая через центр ячейки j и центр сегмента (грани) ij – для нахождения $\mathbf{Q}_{ji}$ (рис. 2.5).

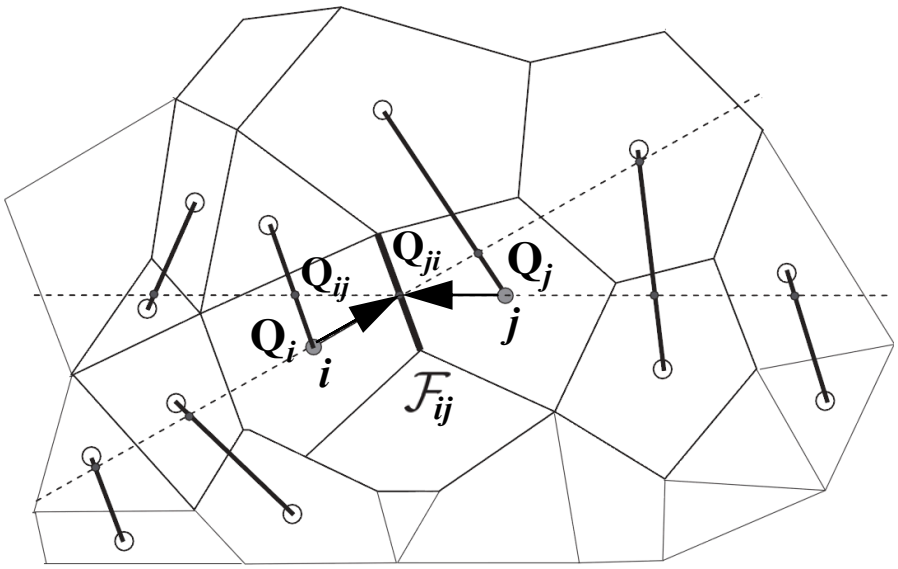


Рисунок 2.5: Квазиодномерная реконструкция значений "слева"и "справа"от центра сегмента по двум направлениям реконструкции в случае элементно-центрированной EBR схемы на двухмерной гибридной сетке, согласно [13, 56]

2.1.4 Определение вязкого численного потока

В случае вершинно-центрированной схемы интеграл от вязкого потока представим в виде суммы значений, посчитанных по инцидентным сеточным элементам:

$$\oint\limits_{\partial C_i} \left(\mathcal{F}^{\mathcal{NS}}\right) \cdot \mathbf{n}\, ds = \sum_{l \in E(i)} \int\limits_{\Gamma_{il}} \left(\mathcal{F}_l^{NS}\right) \cdot \mathbf{n}\, ds = \sum_{l \in E(i)} \mathcal{F}_l^{NS} \hat{\mathbf{n}}_{il},$$

где $E(i)$ обозначает множество сеточных элементов, инцидентных узлу i, $\mathcal{F}_l^{NS}$ – вектор вязкого потока, определённый на сеточном элементе, Γ_{il} – часть поверхности i-й ячейки, лежащая внутри элемента l, $\hat{n}_{il} = \int\limits_{\Gamma_{il}} \mathbf{n}\, \mathbf{ds}$ – её ориентированная площадь. Вектор вязкого потока зависит от физических переменных и их производных. В случае симплициальной сетки производные от физических переменных будем вычислять путём дифференцирования линейного интерполянта по его вершинам. В случае гибридной сетки используется более сложный алгоритм, основанный на локальном разбиении элемента на симплексы. Поскольку данный метод является консервативным, а его шаблон, то есть множество ячеек, с которых берутся значения, совпадает с $N_1(i)$, можно перегруппировать слагаемые в виде суммы по сегментам. Таким образом, в общем виде схема приводится к виду (2.3).

2.1.5 Определение потоков на границе расчётной области

Пусть ячейка с номером i является граничной, т.е. в вершинно–центрированном случае i-й узел лежит на границе расчётной области Γ, а в элементно-центрированном случае ячейка имеет хотя бы одну грань, принадлежащую Γ. Обозначим $\mathbf{n}^\Gamma$ внешнюю нормаль к Γ либо в i-м узле, либо на этой внешней грани, соответственно. Рассмотрим основные типы граничных условий.

1. *Условия отражения (непротекания) на твёрдой стенке*. В дифференциальной постановке это равенство нулю нормальной к поверхности компоненты скорости. В рассматриваемом случае условия задаются следующим способом. Если в формулу для нормальной компоненты потока $\mathbf{F}_1 n_x + \mathbf{F}_2 n_y + \mathbf{F}_3 n_z$ подставить нулевую нормальную компоненту скорости $un_x^\Gamma + vn_y^\Gamma + wn_z^\Gamma = 0$, то граничный поток $\mathbf{\Phi}_i^\Gamma$ в приграничной ячейке i будет иметь следующий вид: $\mathbf{\Phi}_i^\Gamma = \left(0, p_i \mathbf{n}^\Gamma, 0\right)^T$, который можно использовать в качестве потокового граничного условия. Здесь p_i обозначает значение давления в i-й ячейке в вершинно-центрированном случае, а в элементно-центрированном случае берется интерполированное значение на внешней грани.

2. *Условия прилипания на твёрдой стенке.* В случае вершинно-центрированной схемы в граничной точке задаются нулевые значения скорости, а поток массы через все грани приграничной ячейки задаётся нулевым. Для адиабатической стенки $\partial T / \partial \mathbf{n} = 0$ зануляется поток энергии на внешних гранях приграничной ячейки, а для изотермической стенки задаётся $\frac{\partial p}{\partial t} = T \frac{\partial \rho}{\partial t}$. В элеметно-центрированном случае условие прилипания задаётся также, как и условие отражения, а равенство нулю скоростей учитывается при расчёте вязких потоков.

3. *Входные и выходные условия.* Для задания потоков на входных и выходных границах используется такое же характеристическое расщепление, как и во внутренней области, то есть решается задача распада разрыва между значением $\mathbf{Q}_i$ в граничной ячейке и значением с внешней стороны от граничного сегмента $\mathbf{Q}_\infty$, которое может задаваться различным образом. В зависимости от числа характеристик, входящих через данный сегмент в расчётную область, часть значений держится постоянными, а оставшиеся берутся из внутренней ячейки. Например, на сверхзвуковых входных условиях $\mathbf{Q}_\infty$ держится постоянным и не зависит от $\mathbf{Q}_i$; на дозвуковом выходе задаётся давление, а остальные переменные сносятся по характеристикам из ячейки. Более подробно см. в [57].

2.1.6 Дискретизация по времени

Полудискретная аппроксимация уравнений Навье-Стокса, применяя метод линий, представляется виде

$$\left(\frac{d\mathbf{Q}}{dt}\right)_i = -\Psi_i\left(\mathbf{Q}_{\Omega_i}\right), \tag{2.4}$$

где $\Psi_i\left(\mathbf{Q}_{\Omega_i}\right)$ обозначает некоторую пространственную аппроксимацию в i-й ячейке. Ω_i обозначает шаблон схемы, центрированный в этой ячейке, а $\mathbf{Q}_{\Omega_i}$ – множество значений в ячейках шаблона, соответственно.

Явная схема интегрирования по времени

В качестве явной схемы используются схемы семейства Рунге-Кутты. Для каждой ячейки сетки (2.4) можно рассматривать как систему обыкновенных дифференциальных уравнений по времени и записать явную N-шаговую линейную схему Рунге-Кутты [58]:

$$\begin{cases} \mathbf{Q}_i^{(0)} = \mathbf{Q}_i^n \\ \mathbf{Q}_i^{(k)} = \mathbf{Q}_i^{(0)} - \alpha_k \Delta t \;\; \Psi\left(\mathbf{Q}_{\Omega_i}^{(k-1)}\right), \quad k = 1, \ldots, N. \\ \mathbf{Q}_i^{n+1} = \mathbf{Q}_i^{(N)} \end{cases} \tag{2.5}$$

Выбор значений $\alpha_1 = 0.11, \alpha_2 = 0.2766, \alpha_3 = 0.5, \alpha_4 = 1$ в 4-шаговой схеме даёт схему 2-го порядка точности, устойчивую до числа Куранта $CFL = 2$ [59].

Для 4-го порядка точности используется "классическая"4-шаговая схема Рунге-Кутты

$$\begin{cases} \mathbf{k}_1 = \Psi\left(\mathbf{Q}_{\Omega_i}^n\right), & \mathbf{k}_2 = \Psi\left[\left(\mathbf{Q}_i^n + \frac{1}{2}\mathbf{k}_1\right)_{\Omega_i}\right], \\ \mathbf{k}_3 = \Psi\left[\left(\mathbf{Q}_i^n + \frac{1}{2}\mathbf{k}_2\right)_{\Omega_i}\right], & \mathbf{k}_4 = \Psi\left[\left(\mathbf{Q}_i^n + \mathbf{k}_3\right)_{\Omega_i}\right], \\ \mathbf{Q}_i^{n+1} = \mathbf{Q}_i^n + \frac{1}{6}\left(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4\right) \end{cases} \tag{2.6}$$

Неявная схема интегрирования по времени

Полудискретная аппроксимация (2.4) имеет вид

$$V_i \left(\frac{d\mathbf{Q}}{dt}\right)_i = -\Psi_i\left(\mathbf{Q}_{\Omega_i}\right), \tag{2.7}$$

где V_i – объём i-й ячейки, $\mathbf{Q}_{\Omega_i}$ – аппроксимация поверхностного интеграла в методе конечного объёма (см. раздел 2.1.2). Функцию $\mathbf{Q}_{\Omega_i}$, которая зависит от значений в точках пространственного шаблона Ω_i, можно записать в виде суммы конвективных потоков Ψ_2^c, вязких потоков Ψ_2^d и источника S:

$$\Psi_i\left(\mathbf{Q}_{\Omega_i}\right) = \Psi_2^c + \Psi_2^d + S$$

Рассмотрим сеточную функцию $\mathbf{Q}_i^n$, аппроксимирующую $\mathbf{Q}_i(t^n)$ на временном слое t^n. Полностью неявная схема 1-го порядка по времени для (2.7) на слое $t^{n+1} = t^n + \Delta t^n$ будет иметь вид:

$$\frac{V_i}{\Delta t^n}\left(\mathbf{Q}_i^{n+1} - \mathbf{Q}_i^n\right) = -\Psi_i\left(\mathbf{Q}_{\Omega_i}^{n+1}\right), \tag{2.8}$$

а схема 2-го порядка

$$
\begin{aligned}
&V_i\left(a\mathbf{Q}_i^{n+1}+b\mathbf{Q}_i^{n}+c\mathbf{Q}_i^{n-1}\right)=-\Psi_i\left(\mathbf{Q}_{\Omega_i}^{n+1}\right)\\
&a=\frac{1}{\Delta t^n}\frac{2\tau+1}{\tau+1},\quad b=-\frac{1}{\Delta t^n}\left(\tau+1\right),\quad c=\frac{1}{\Delta t^n}\frac{\tau^2}{\tau+1},\quad \tau=\frac{\Delta t^n}{\Delta t^{n-1}}.
\end{aligned}
\tag{2.9}
$$

Рассмотрим схему 2-го порядка (2.9) и построим ньютоновский итерационный процесс для нелинейного уравнения $F\left(\mathbf{Q}_i^{n+1}\right)=0$, где функция F определяется следующим образом

$$
F\left(\mathbf{Q}_i^{n+1}\right)=V_i a\mathbf{Q}_i^{n+1}+\Psi\left(\mathbf{Q}_i^{n+1}\right)-V_i b\mathbf{Q}_i^{n}-V_i c\mathbf{Q}_i^{n-1}. \tag{2.10}
$$

Ньютоновский итерационный процесс для нахождения $\mathbf{Q_i}_i^{n+1}$ имеет вид

$$
\begin{aligned}
&\overset{(0)}{\mathbf{Q}_i}^{n+1}=\mathbf{Q}_i{}^{n},\\
&\overset{(s+1)}{\mathbf{Q}_i}^{n+1}=\overset{(s)}{\mathbf{Q}_i}^{n+1}-\left[F'\left(\overset{(s)}{\mathbf{Q}_i}^{n+1}\right)\right]^{-1}F\left(\overset{(s)}{\mathbf{Q}_i}^{n+1}\right),
\end{aligned}
\tag{2.11}
$$

где s – номер ньютоновской итерации, а коэффициенты якобиан определяются следующим образом:

$$
F'_{ij}=\frac{\partial F_i}{\partial \mathbf{Q}_j}(\overset{(s)}{\mathbf{Q}_i^{n+1}})=aV_i\delta_{ij}+\frac{\partial\Psi(\Omega_i)}{\partial\mathbf{Q}_j}(\overset{(s)}{\mathbf{Q}_i^{n+1}}),\quad i,j\in\Omega_i. \tag{2.12}
$$

Подставляя (2.10) и (2.12) в (2.11) и обозначая приращение на s-ой итерации через $\Delta\overset{(s)}{\mathbf{Q}}{}^{n+1}=\overset{(s+1)}{\mathbf{Q}}{}^{n+1}-\overset{(s)}{\mathbf{Q}}{}^{n+1}$ получаем линеаризованную версию схемы (2.9)

$$
\left[aV+\frac{\partial\Psi}{\partial\mathbf{Q}_i}\left(\overset{(s)}{\mathbf{Q}_i}^{n+1}\right)\right]\Delta\overset{(s)}{\mathbf{Q}_i}^{n+1}=-\Psi\left(\overset{(s)}{\mathbf{Q}_i}^{n+1}\right)-V\left(a\,\overset{(s)}{\mathbf{Q}_i}^{n+1}+b\mathbf{Q}_i{}^{n}+c\mathbf{Q}_i{}^{n-1}\right). \tag{2.13}
$$

Обращая оператор в левой части (2.13), получаем

$$
\Delta\overset{(s)}{\mathbf{Q}_i}^{n+1}=\mathfrak{M}^{-1}\left[-\Psi\left(\overset{(s)}{\mathbf{Q}_i}^{n+1}\right)-V\left(a\,\overset{(s)}{\mathbf{Q}_i}^{n+1}+b\mathbf{Q}_i{}^{n}+c\mathbf{Q}_i{}^{n-1}\right)\right].
$$

Вместо ньютоновского процесса (2.11) можно использовать упрощённый вариант, в котором вычисление коэффициентов якобиана происходит только на нулевой итерации $\left[aV+\frac{\partial\Psi}{\partial\mathbf{Q}_i}\left(\overset{(0)}{\mathbf{Q}_i}^{n+1}\right)\right]\Delta\overset{(s)}{\mathbf{Q}_i}^{n+1}=-\Psi\left(\overset{(s)}{\mathbf{Q}_i}^{n+1}\right)-V\left(a\,\overset{(s)}{\mathbf{Q}_i}^{n+1}+b\mathbf{Q}_i{}^{n}+c\mathbf{Q}_i{}^{n-1}\right)$.

Для дальнейшего упрощения вычислений и уменьшения потребности памяти для хранения якобиана можно брать поток, конвективная часть которого подсчитана на более узком шаблоне, чем шаблон Ω_i, например, по базовой схеме 1-го порядка. При этом, с одной стороны, улучшаются свойства матрицы и существенно сокращается количество ненулевых элементов, но, с другой стороны, ухудшается сходимость ньютоновских итераций (сходимость уже не квадратична) и возникают определённые ограничения по числу Куранта.

2.1.7 Моделирование турбулентности

Для моделирования турбулентных течений в контексте данной работы могут использоваться как вихреразрешающие подходы, так и RANS (Reynolds averaged Navier-Stokes) подходы, подразумевающие решение осреднённых по Рейнольдсу уравнений Навье-Стокса с той или иной моделью замыкания. Используемые вихреразрешающие подходы включают в себя: прямое численное моделирование – DNS (direct numerical simulation), которое предполагает разрешение пространственной и временной дискретизацией турбулентных структур всех релевантных масштабов; моделирование методом крупных вихрей – LES (Large Eddy Simulations) [7], при котором учёт вихрей меньше определённого масштаба внутри инерционного интервала описывается с помощью той или иной модели турбулентности; гибридные подходы [60], эффективно сочетающие RANS и LES подходы.

В RANS подходах для замыкания осреднённых по Рейнольдсу уравнений Навье-Стокса используются дифференциальные модели с одним или двумя уравнениями: модель Спаларта–Аллмараса с одним уравнением [61]; К-Омега (Вилкокса) с двумя уравнениями [62]; модель К-Эпсилон с двумя уравнениями [63]; модель SST Ментера с двумя уравнениями [64].

При моделировании методом LES используются следующие алгебраические модели турбулентности: модель Смагоринского [65]; модель WALE с адаптирующейся к стенке вихревой вязкостью [66]; модель Времана [67]; модель Верстаппена [68]; σ-модель [69]; модели семейства S3 – S3PQ, S3QR, S3PR [70]. Более подробная информация об этих моделях представлена, например, в [70].

Из гибридных RANS-LES подходов используются следующие незонные подходы семейства DES (Detached eddy simulations – моделирование отсоединённых вихрей): DES-97 [71]; DDES [72]; IDDES (Improved DDES – улучшенный DDES) [8]; DES с ускорением “численного” перехода в слоях смешения [73].

Работа модели турбулентности в общем виде заключается в вычислении так называемой турбулентной вязкости, учитываемой в вязких потоках в системе уравнений Навье-Стокса. С точки зрения построения параллельного алгоритма основной вычислительной операцией, необходимой как для алгебраических, так и дифференциальных моделей турбулентности, является расчёт в ячейках градиентов скоростей по трем пространственным направлениям. В случае вершинно-центрированной схемы для этого используется операция расчёта узловых градиентов, вычисляемых в каждом узле путём суммирования градиентов по всем сеточным элементам, содержащим данный узел (см. [31, 53], где такая же операция используется для реконструкции с повышенным порядком). В элементно-центрированном случае для расчёта градиентов применяется метод наименьших квадратов по соседним ячейкам (коэффициенты вычисляются на этапе инициализации при запуске задачи).

Более подробно используемые модели и подходы к моделированию турбулентности представлены в [74, 75].

2.2 Распараллеливание базовых операций

2.2.1 Состав расчётной области

Расчётная область на неструктурированной гибридной сетке определяется следующими основными наборами элементов:

- набором сеточных узлов, заданным списком координат узлов сетки;
- набором сеточных элементов (тетраэдров, четырёхугольных пирамид, треугольных призм, гексаэдров), каждый из которых задан списком номеров составляющих его узлов;
- набором ячеек (контрольных объёмов), который в случае вершинно-центрированной схемы соответствует набору узлов, а в случае объёмно-центрированной схемы – набору сеточных элементов.
- набором сегментов, которые в случае вершинно-центрированной схемы соответствуют ребрам сетки, а в случае объёмно-центрированной схемы – граням сеточных элементов.
- набором внешних граней, то есть граней сеточных элементов, инцидентных только одному сеточному элементу.

Назовем *прямой топологией* (или просто топологией) описание связей набора элементов расчётной области некоторого типа с набором ячеек: топология по сеточным элементам – структура данных, в которой каждому сеточному элементу сопоставлен список его узлов; топология по сегментам – структура данных, в которой каждому сегменту сопоставлены номера ячеек, которые он разделяет; и т.д.

Назовем *обратной топологией* описание связей набора элементов расчётной области некоторого типа с набором ячеек: обратная топология по сеточным элементам – структура данных, в которой каждому узлу сопоставлен список сеточных элементов, в состав которых он входит (вершинно-центрированный случай); обратная топология по сегментам – структура данных, в которой ячейке сопоставлен список номеров сегментов, которые составляют ячейку; обратная топология по связям ячеек – структура данных, в которой ячейке сопоставлен список ячеек, с которыми она имеет общую поверхность; и т.д.

Согласно технологии, представленной в первой главе, расчётная область посредством рациональной декомпозиции графа сетки разделяется между параллельными процессами на подобласти MPI процессов (подобласти 1-го уровня), для каждой ячейки сетки определяется номер процесса-владельца. *Собственными* являются те ячейки, которые принадлежат данному процессу и составляют его подобласть. Две ячейки являются *соседними*, если они связаны шаблоном численной схемы, то есть шаблон, центрированный в одной ячейке или её грани, включает вторую ячейку. Набор *чужих* ячеек, не принадлежащих подобласти, но соседних с ячейками данной подобласти, будем называть *гало*. Чужие ячейки, непосредственно граничащие с собственными ячейками – это гало 1-го уровня. Гало элементы 2-го уровня – чужие ячейки, соседние с гало ячейками 1-го уровня, и т.д. Назовем *расширенной подобластью* объединение множества собственных и гало ячеек. *Интерфейсные ячейки* – собственные ячейки, имеющие

соседей из гало. *Внутренние ячейки* – это собственные ячейки, имеющие связи только с собственными ячейками. Аналогичным образом определяются наборы элементов расчётной области других типов – сеточных элементов, граней и т.д.. *Интерфейсные грани* разделяют собственную и чужую ячейки, внутренние – разделяют собственные ячейки.

2.2.2 Типы операций, входных и выходных данных

Под *входными данными* операций алгоритма интегрирования по времени будем понимать только те данные, использующиеся на чтение в данной операции, которые могут изменяться во время расчёта (в том числе в других операциях алгоритма). Данные, не изменяющиеся во время расчёта, как, например, в случае статической сетки, данные, определяющие геометрию контрольных объёмов, топологию связей ячеек и т.д., не учитываем в качестве входных данных. *Выходными данными* операции алгоритма интегрирования по времени являются данные, в которые операция выполняет запись.

Алгоритм будет составлен из *базовых операций*, то есть операций, представимых в виде цикла, итерации которого могут быть выполнены в произвольном порядке, по набору элементов расчётной области некоторого типа. Входные данные операции соответствуют набору элементов какого-то одного типа, Выходные данные также соответствуют набору элементов какого-то одного типа. При этом тип набора, по которому выполняется операция, тип, которому соответствуют входные данные, и тип, которому соответствуют выходные данные, могут отличаться и быть представлены тремя различными типами. Тип набора, по которому выполняется операция, будем называть *типом операции*; тип набора, которому соответствуют входные данные – *типом входных данных*; тип набора, которому соответствуют выходные данные, – *типом выходных данных*.

Итерацией в данном контексте будем называть обработку одного элемента расчётной области в рамках данной операции (по аналогии с итерацией цикла). *Элементарным заданием* будем называть минимальный объём работы, распределяемый между параллельными потоками (предполагая, что одна итерация может в принципе выполняться более чем одним параллельным потоком).

Соответственно, базовая операция с точки зрения распараллеливания характеризуется следующим:

1. типом элементов расчётной области, которому соответствует набор входных данных операции;
2. типом операции, то есть типом элементов расчётной области, по набору которых выполняется операция (по которым ведется цикл);
3. типом элементов расчётной области, которым соответствуют выходные данные операции;
4. могут ли входные данные итерации включать более одного элемента;
5. могут ли выходные данные итерации включать более одного элемента.

Распараллеливание базовой операции обусловливается зависимостью по входным данным, во-первых, между разными операциями, составляющими алгоритм, и, во-вторых, внутренней зависимостью по входным данным между итерациями. Первый тип зависимости необходимо учитывать при распараллеливании с распределённой памятью, второй тип – при распараллеливании с общей памятью, при распараллеливании для потоковой обработки и векторизации. Зависимость по данным следует из сочетания типов операции, входных и выходных данных. Обмен данными между параллельными процессами, например, может быть необходим, если тип входных данных не совпадает с типом операции или, если выполнено условие 4 вышеописанного списка. Пересечение по данным между итерациями при параллельной обработке множественными потоками возникает, когда тип операции не совпадает с типом выходных данных или когда выполнено условие 5, соответственно.

Для краткости классификации базовой операции, будем описывать её в виде [<тип элемента входных данных> ⇒ <тип элемента операции> ⇒ <тип элемента выходных данных>]. Выполнение условия 4 будет выражено множественным числом типа входных данных, условия 5 – выходных данных, соответственно.

Например, операция расчёта потока через грани (сегменты) ячеек, использующая на вход значения сеточных функций из центров ячеек, и на выходе записывающая поток через грань, имеет вид [ячейки ⇒ грань ⇒ грань]; операция суммирования потоков с граней в центры ячеек в цикле по ячейкам использует значения потоков с граней и записывает результат в ячейки – [грани ⇒ ячейка ⇒ ячейка]. Другой важный пример – операция матрично-векторного произведения, когда строки матрицы соответствуют ячейкам, ненулевые позиции в строке соответствуют граням (или связям между ячейками шаблоном численной схемы). Операция такого типа, которой соответствует обозначение [ячейки ⇒ ячейка ⇒ ячейка], в частности, используется в решателе СЛАУ при обращении якобиана. Далее рассмотрим более подробно типы операций, которые будут использоваться при составлении расчётного алгоритма.

2.2.3 Базовые операции

Для устранения зависимости по данным между итерациями будет использоваться граф связей, в котором вершинам сопоставлены элементы того типа, которому соответствуют выходные данные операции, а рёбрам – элементы того типа, по которому выполняется операция.

Тип T1: [ячейка ⇒ ячейка ⇒ ячейка]

Данный тип операции может быть представлен в виде цикла по ячейкам, где на каждой итерации используются данные из той же ячейки, и результат записывается в ту же ячейку. Такой тип операции не представляет проблемы для распараллеливания, поскольку не имеет зависимости по данным между итерациями цикла. Такая операция выполняется по набору ячеек, соответствующему расширенной подобласти, чтобы данные в гало ячейках оставались актуальными и не требовали обмена данными между параллельными процессами.

Если в операции по набору ячеек выполняется редукция глобальных величин – поиск максимума, минимума или суммы по всем ячейкам, будем обозначать такую операцию **T1R** (R – reduction). Операция с редукцией выполняется по подобласти, т.е. только по собственным ячейкам (не включая гало) и требует распараллеливания. Для распараллеливания с распределённой памятью операция дополняется соответствующим групповым обменом данными – MPI_Allreduce. При распараллеливании с общей памятью либо глобальные величины с редукцией указываются в явном виде соответствующей директивой для автоматической редукции, либо редукция делается вручную путём размножения глобальных величин по нитям и последующей сборкой результата главной нитью. Для потоковой обработки распараллеливание выполняется стандартным образом путём разделения операции на два этапа: на первом этапе каждая локальная рабочая группа выполняет редукцию в своей локальной памяти и записывает свой результат в глобальную память; на втором этапе одиночная нить выполняет сборку результатов локальных групп в глобальной памяти.

Тип T2: [ячейки ⇒ ячейка ⇒ ячейка] или [сегменты ⇒ ячейка ⇒ ячейка]

Данный тип операции может быть представлен в виде цикла по ячейкам, где на каждой итерации на чтение используются данные из той же ячейки, а также из ячеек с которыми данная ячейка связана шаблоном численной схемы (или из сегментов ячейки, соответственно). Дополнительно предполагается, что у данной операции нет зависимости по данным между итерациями, то есть выходной результат операции не используется на чтение в качестве входных данных для других ячеек. Такая операция выполняется по набору ячеек, соответствующему подобласти (то есть только по собственным ячейкам, принадлежащим процессу).

С точки зрения распараллеливания с распределённой памятью, использование в расчёте по ячейке данных из других ячеек может требовать обмена данными между процессами для обновления значений в гало ячейках, которые могут использоваться для расчёта в собственных ячейках. То есть для данной операции требуются актуальные данные в гало ячейках. Количество уровней соседства в гало для каждой интерфейсной ячейки определяется шаблоном численной схемы, центрированным в данной ячейке. Результаты данной операции делают значения в гало ячейках неактуальными. Для последующего использования этих данных, соответствующих результатам данной операции, необходим обмен данными между параллельными процессами.

Поскольку у операции отсутствует зависимость по данным между итерациями, она не представляет проблемы для многопоточного распараллеливания в рамках модели с общей памятью. OpenMP распараллеливание выполняется путём распределения итераций цикла между параллельными потоками (нитями) с использованием статической планировки (*#pragma omp for schedule(static)*) в случае однородных по вычислительной нагрузке итераций, и с использованием динамической планировки (*#pragma omp for schedule(dynamic,k)*). Размер распределяемых блоков итераций, k, выбирается эмпирически, исходя из числа итераций, числа нитей, и вычислительной

стоимости итераций. В данном случае будет использоваться $k = MAX(1,N/(10N_t))$, где N – число итераций, N_t – число нитей.

Для распараллеливания в рамках парадигмы потоковой обработки проблему представляет неоднородность элементарных заданий: поскольку у ячеек может быть разное число связей с другими ячейками возможен дисбаланс вычислительной нагрузки, приводящий к простою параллельных потоков. Чтобы избежать неоднородности элементарных заданий ячейки можно сгруппировать по числу связей и обрабатывать набор по однородным группам.

Тип T3: [ячейки ⇒ сегмент ⇒ сегмент]

Операция, представленная в виде цикла по сегментам, на каждой итерации использует на чтение данные из соседних ячеек, записывает результат в данные, сопоставленные данному сегменту.

Распараллеливание с распределённой памятью может требовать обмен данными между процессами для обновления значений в гало ячейках, которые могут использоваться для расчёта по сегментам интерфейсных ячеек. Данный тип операции требует актуальные данные в гало ячейках. Количество уровней соседства в гало для сегмента интерфейсной ячейки определяется шаблоном численной схемы, центрированным на данном сегменте.

Поскольку у операции отсутствует зависимость по данным между итерациями, она не представляет проблемы для многопоточного распараллеливания в рамках модели с общей памятью. OpenMP распараллеливание и распараллеливание в рамках парадигмы потоковой обработки выполняется аналогично предыдущему типу T2.

Тип T4: [ячейки ⇒ сегмент ⇒ ячейки]

Операция, представленная в виде цикла по граням, на каждой итерации использует на чтение данные из соседних ячеек, записывает результат в инцидентные ячейки.

Для распараллеливания с распределённой памятью, аналогично типу T1, данной операции необходимы обновленные данные в гало ячейках. Результаты данной операции делают значения в гало ячейках неактуальными. Для последующего использования этих данных, соответствующих результатам данной операции, необходим обмен данными между параллельными процессами.

У операции данного типа присутствует пересечение по данным между итерациями, поскольку на разных итерациях может осуществляться доступ на запись в одну и ту же ячейку. Для OpenMP распараллеливания, согласно технологии, представленной в первой главе, рассматривается два подхода.

(a) Разделить операцию на два этапа, представленных операциями типа T3 и T2, согласно разделу 1.5.1. Сначала выполняется вычислительно-ёмкая часть операции в цикле по сегментам (тип T3), которая записывает промежуточный результат дополнительный массив по сегментам. Затем после точки синхронизации потоков выполняется вторая часть

операции, представленная типом T3, в которой в цикле по ячейкам (тип T2, версия с входными данными по сегментам) осуществляется сборка результатов из промежуточного массива по сегментам в ячейки.

(b) Использовать многоуровневую декомпозицию (см. раздел 1.4.1).

Если первый подход применим, то есть такое разделение возможно, то выбор данного способа определяется объёмом памяти, необходимым для хранения промежуточных данных. Слишком большой объём делает такой подход нецелесообразным.

Для распараллеливания в рамках парадигмы потоковой обработки необходимо устранение пересечения по данным между итерациями. Если операция разделена на два этапа типов T3 и T2, то она уже адаптирована к потоковой обработке и эти этапы реализуются соответственно типам.

Если разделение на две операции, совместимых с потоковой обработкой, неприменимо, то, согласно разделу 1.5.1, используется разделение операции на такты с точкой синхронизации между тактами: исходный набор сегментов делится на поднаборы, в которых нет сегментов, имеющих общую ячейку; затем возможна потоковая обработка каждого поднабора. Для разделения на поднаборы используется граф связей, в котором вершины соответствуют ячейкам, а ребра – сегментам. Раскраска графа выполняется таким образом, чтобы у ячеек не было двух ребер одного цвета. Затем выполняется поэтапная обработка сегментов, соответствующих ребрам одного цвета. Если у данной операции присутствует неоднородность элементарных заданий (если, например, в шаблон реконструкции входит разное число ячеек), то возможен дисбаланс вычислительной нагрузки, приводящий к простою параллельных потоков. Чтобы избежать неоднородности элементарных заданий, сегменты в рамках поднаборов можно группировать по числу связей в шаблоне и обрабатывать однородные подгруппы.

Чтобы сопоставить типу операции тип распараллеливания, дополним тип буквенным обозначением:

T4a – операция типа 4, для который применяется разделение операции на два этапа;

T4b – операция типа 4, для который применяется многоуровневое распараллеливание с общей памятью и потактовое выполнение для потоковой обработки.

Тип T5: [ячейки ⇒ сеточный элемент ⇒ ячейки]

Данная операция присутствует только в случае численной схемы с определением значений сеточных функций в узлах, то есть ячейкам соответствуют узлы сетки. Операция, представленная в виде цикла по сеточным элементам, на каждой итерации использует на чтение данные из узлов, составляющих сеточных элемент, записывает результат в эти же узлы.

Подход к распараллеливанию данной операции полностью аналогичен предыдущему типу T4 (соответственно, у данного типа операции будут два обозначения – **T5a** и **T5b**), с той лишь разницей, что число инцидентных элементу узлов (ячеек) не два, а может быть от 4 (тетраэдр) до 8 (гексаэдр).

Тип T6: [ячейки ⇒ внешняя грань ⇒ ячейки]

Данный тип соответствует граничным условиям в случае численной схемы с определением значений сеточных функций в узлах. Распараллеливание данной операции полностью аналогично предыдущему типу T5 (соответственно, у данного типа операции будут два обозначения – **T6a** и **T6b**), с той лишь разницей, что число инцидентных грани узлов (ячеек) может быть от 3 (треугольные грани) до 4 (четырехугольные грани).

2.3 Вычислительный алгоритм

Алгоритм разделен на две части: инициализация вычислений – выполняется один раз при запуске программы; интегрирование по времени – основная часть, выполняется до достижения критерия завершения расчёта или до остановки программы извне.

2.3.1 Инициализация вычислений

Инициализационная часть не требует повышенного внимания к производительности вычислений, однако вычислительно-ёмкие этапы требует распараллеливания.

1. Чтение основного набора входных данных:
   (a) Чтение параметров расчёта;
   (b) Чтение заголовка сеточных данных, содержащего число узлов, элементов, граничных граней;
   (c) Выделение памяти и создание структур данных для хранения расчётной области;
   (d) Чтение координат узлов;
   (e) Чтение топологии сетки;
   (f) Чтение списка граничных граней;
   (g) Чтение списка соответствия типа граничной поверхности типу граничных условий;
   (h) Чтение топологии периодических граничных условий (если присутствуют);
   (i) Чтение расстояний до твёрдых поверхностей;
2. Инициализация топологии
   (a) Инициализация декомпозиции 2-го уровня
   (b) Построение обратной топологии для сеточных элементов в CSR формате, определяющей для каждого узла список содержащих его сеточных элементов;
   (c) Построение обратной топологии для узлов в CSR формате, определяющей для каждого узла список его соседних узлов;
   (d) Построение списка внутренних граней (однозначно соответствующих ребрам сетки) контрольных объёмов, определяющего для каждой грани узлы, которые она разделяет.
   (e) Построение обратной топологии для граней в CSR формате, определяющей для каждого узла список граней.
3. Инициализация геометрии

(a) Построение геометрии контрольных объёмов: вычисление площади и нормали для каждой грани, расчёт вклада от сеточных элементов в контрольные объёмы их узлов, вычисление величины контрольного объёма, и т. д.
(b) Построение интерполяционных конструкций повышенного порядка: поиск пересечения прямой ребра с соседними элементами;
(c) Вычисление минимальных высот сеточных элементов;

4. Инициализация данных
   (a) Выделение памяти под хранение сеточных функций
   (b) Инициализация параметров обезразмеривания
   (c) Инициализация схемы интегрирования по времени
   (d) Инициализация модели турбулентности (если используется)
   (e) Инициализация источниковых членов (если присутствуют)
   (f) Инициализация средних полей течения
   (g) Инициализация основных полей течения (или пульсационных составляющих, в зависимости от модели)
   (h) Инициализация параметров и вспомогательных данных для расчёта вязкости.

При запуске вычислительного ядра выполняется инициализация всех подключенных в расчёте модулей, загружается сетка, параметры пользовательского ввода, коммуникационная схема и т. д. Затем управление передаётся главному циклу интегрирования по времени.

2.3.2 Алгоритм интегрирования по времени

Составим алгоритм из базовых операций шести вышеописанных типов. Составные операции, не являющиеся базовыми, отмечены полужирным шрифтом.

Шаг по времени имеет следующий алгоритм:

1. расчёт величины шага по времени – T1R;
2. **расчёт значений на новом временном слое**;
3. **обработка результатов**.
4. переход на новый временной слой – T1;

Обработка результата состоит из следующих основных операций:

1. проверка корректности полей течения – T1;
2. обработка статистики течения – набор операций типа T1;
3. проверка корректности полей течения – T1;
4. ввод-вывод: обновление параметров расчёта, запись результата, выдача диагностики;

Расчёт значений на новом временном слое существенно различен для явной и неявной схемы.

Явная схема интегрирования по времени

1. цикл многошагового метода Рунге-Кутты:
   (a) **расчёт конвективных потоков**;
   (b) **расчёт вязких потоков**;
   (c) расчёт узловых граничных условий – T1 (только для вершинно-центрированной схемы: условия прилипания на твёрдой стенке, неотражающие условия Тама [76]);
   (d) расчёт потоковых граничных условий – T6b;
   (e) расчёт источников – T1;
   (f) шаг интегрирования Рунге-Кутты – T1;

Расчёт потоков зависит от типа численной схемы. Для многопараметрической вершинно-центрированной схемы [53] алгоритм расчёта конвективной части потоков имеет вид

1. заполнение потоковых функций в узлах – T1;
2. расчёт узловых градиентов – T5b;
3. расчёт потоков через сегменты ячеек – T4a;
   (a) реконструкция и расчёт потоков через сегменты – T3;
   (b) суммирование потоков с сегментов в ячейки – T2;

Расчёт вязких потоков имеет вид

1. расчёт кинематической вязкости – T5B;
2. расчёт вклада модели турбулентности – T5b;

Для вершинно-центрированной EBR схемы [12, 31] алгоритм расчёта конвективной части потоков имеет вид

1. заполнение потоковых функций в узлах – T1;
2. расчёт потоков через грани ячеек – T4a;
   (a) реконструкция и расчёт потоков через грани – T3;
   (b) суммирование потоков с граней в ячейки – T2;

Расчёт вязких потоков имеет вид

1. расчёт кинематической вязкости – T5b или T4a;
2. расчёт вклада модели турбулентности – T5b;

Для EBR схемы [13] с определением переменных в центрах сеточных элементов алгоритм расчёта конвективной части потоков имеет вид

1. заполнение потоковых функций;
2. расчёт потоков через грани ячеек – T4a;
   (a) реконструкция и расчёт потоков через грани – T3;
   (b) суммирование потоков с граней в ячейки – T2;

Расчёт вязких потоков имеет вид

1. расчёт кинематической вязкости – T4b;
2. расчёт вклада модели турбулентности – T4b;

Неявная схема интегрирования по времени на основе линеаризации по Ньютону

В случае неявной схемы шаг по времени имеет следующий алгоритм:

1. цикл итераций по Ньютону:
   (a) **расчёт конвективных потоков и вклада в якобиан**;
   (b) **расчёт вязких потоков и вклада в якобиан**;
   (c) расчёт узловых граничных условий и вклада в якобиан – T1;
   (d) расчёт потоковых граничных условий и вклада в якобиан – T6b;
   (e) расчёт источников и вклада в якобиан – T1;
   (f) **решение СЛАУ**, включающее операции:
      i. матрично-векторное произведение – T2,
      ii. скалярное произведение – T1R,
      iii. линейная комбинация векторов – T1.
2. расчёт невязки ньютоновского процесса – T1R;

Состав операций для расчёта конвективных и вязких потоков аналогичен явной схеме, но каждая операция дополняется записью соответствующего вклада в коэффициенты матрицы якобиана. Необходимость заполнения якобиана исключает вариант (a) для операций типа T4, и в отличие от явной схемы используется вариант T4b. Блок-схема алгоритма для вершинно-центрированной EBR схемы показана на рис. 2.6.

2.3.3 Мультисистемный решатель СЛАУ для мелкоблочных разреженных матриц

При использовании неявной схемы с линеаризацией по Ньютону на каждой итерации Ньютоновского процесса требуется решение СЛАУ с якобианом. Матрица представляется в блочном построчно-разреженном формате CSR (compressed sparse row). Размер блока равен числу переменных, которое может варьироваться от 4 до 7: 4 основных переменных в двухмерном случае и 5 основных переменных в трёхмерном случае (плотность, компоненты скорости, энергия); переменные дифференциальной модели турбулентности, если используется, от 1-й до 2-х. В данной численной схеме турбулентные переменные в матрице СЛАУ не имеют связи с основными переменными, т.е. в строках, соответствующих турбулентным переменным, нет ненулевых позиций в столбцах, соответствующих основным переменным, и наоборот. Блоки системы представляются в виде двух диагональных подблоков размерности соответственно числу основных и турбулентных переменных. Вне диагональных подблоков нет ненулевых элементов.

Таким образом, когда используется дифференциальная модель турбулентности, фактически решается две независимых системы. Основные и турбулентные переменные хранятся вместе в блочных векторах из соображений эффективности доступа к памяти и использования кэш. Системы, соответственно, сгруппированы в одну матрицу мелкоблочной структуры, в которой

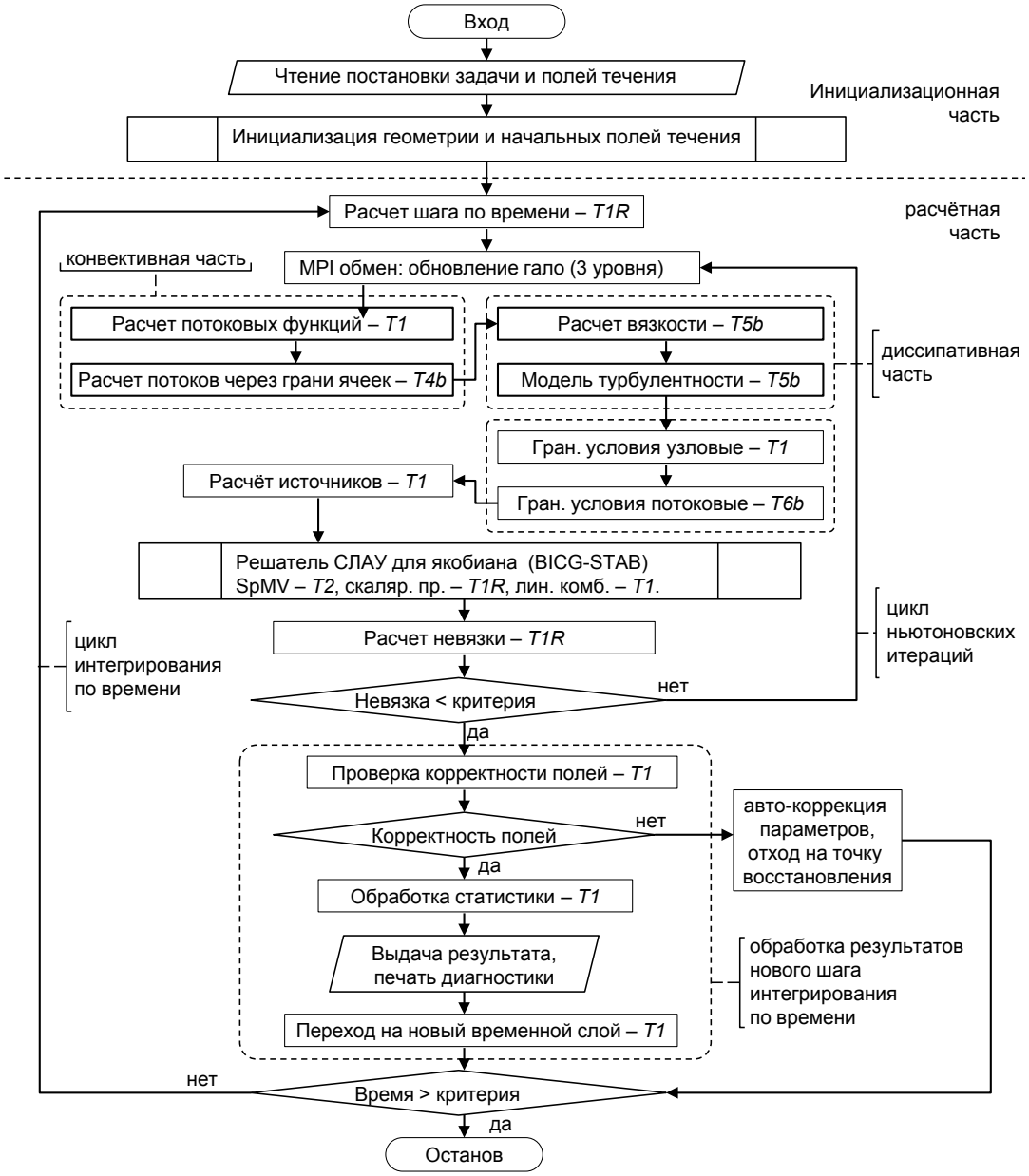


Рисунок 2.6: Блок-схема параллельного алгоритма для неявной схемы интегрирования по времени и вершинно-центрированной EBR схемы дискретизации по пространству

блоки состоят из двух диагональных подблоков. Такое объединение делается для повышения компактности данных и эффективности использования памяти.

Для решения СЛАУ используется стандартный стабилизированный метод бисопряженных градиентов BiCGSTAB (см. [77]). Оригинальным решением является реализация метода в виде мультисистемного решателя для матрицы вышеописанной блочной структуры. Объединение решения двух систем в один итерационный процесс позволяет объединить обмены данными, необходимые для матрично-векторного произведения и для скалярного произведения и избежать таким образом потерь на латентность обмена данными. Когда критерий сходимости для одной из систем удовлетворен, соответствующая ей часть отключается.

Реализация мультисистемного решателя подразумевает реализацию операций линейной алгебры (матрично-векторное произведение, скалярное произведение, линейная комбинация векторов) для блочных векторов с возможностью отключения обработки частей, соответствующих основным или турбулентным переменным.

В алгоритм решателя входят следующие базовые операции

1. матрично-векторное произведение с блочной разреженной матрицей – T2;
2. скалярное произведение – T1R;
3. линейная комбинация векторов – T1;
4. обращение диагональных блоков (для построения блочного диагонального предобуславливателя Якоби) – T1;

Выводы по главе

На базе существующих математических моделей и численных схем сформулирован универсальный параллельный алгоритм, совместимый с вычислительными системами различной архитектуры, в том числе гибридными суперкомпьютерами с массивно-параллельными ускорителями. Разработан набор типовых базовых операций и способов их распараллеливания в рамках 1) модели с распределённой памятью, 2) модели с общей памятью, 3) потоковой обработки. Из базовых операций сформулирован параллельный алгоритм для различных типов численных схем. Все операции адаптированы к потоковой обработке и полученный алгоритм полностью совместим с современными гибридными суперкомпьютерами, включая архитектуры на основе ускорителей GPU NVIDIA, GPU AMD, Intel Xeon Phi, процессоров и ускорителей архитектуры ARM.

Глава 3

Параллельный программный комплекс NOISETTE для крупномасштабных расчётов задач аэродинамики и аэроакустики на неструктурированных сетках

Вводные замечания

Программный комплекс NOISEtte, основанный на схемах повышенной точности для неструктурированных сеток, позволяет моделировать задачи газовой динамики и аэроакустики, используя десятки тысяч процессорных ядер суперкомпьютера. Программный комплекс разрабатывается коллективом, в который помимо автора данной работы входят: Т. К. Козубская, И. В. Абалакин, П. А. Бахвалов, А. П. Дубень, В. Г. Бобков, Н. С. Жданова. В NOISEtte также используются параллельные средства обработки сеточных данных большого объёма, разработанные С. А. Суковым [78]. NOISEtte отличает высокая степень научной новизны – в программном комплексе использованы результаты 5 диссертационных работ:

- И. В. Абалакин – численные схемы и методы, к. ф.-м. н. [79];
- П. А. Бахвалов – экономичные схемы повышенного порядка, к. ф.-м. н. [80];
- А. В. Горобец – параллельные технологии, к. ф.-м. н. [81];
- А. П. Дубень – моделирование турбулентности, к. ф.-м. н. [74];
- Т. К. Козубская – численные схемы и методы, д. ф.-м. н. [50];

История развития кода показана на рис. 3.1. Работа по распараллеливанию NOISEtte начата автором в 2004-м году в рамках кандидатской диссертации. В начальный момент NOISEtte представлял из себя небольшой последовательный код на языке FORTRAN 77, с помощью которого можно было моделировать двухмерные задачи на треугольных сетках. Результатом

работы по кандидатской диссертации было создание параллельного программного комплекса на языке FORTRAN 95 с MPI распараллеливанием для моделирования сжимаемых течений в трёхмерной расчётной области, представленной неструктурированной тетраэдральной сеткой (2006 г.).

Далее, с 2008-го года начинается данная диссертационная работа, в рамках которой было реализовано многоуровневое MPI+OpenMP распараллеливание и создан новый параллельный программный комплекс на языке C++ (стандарта C++03 2003 года). Областями ответственности автора и, соответственно, результатами работы являются:

- применение разработанной параллельной технологии для реализации программного комплекса;
- программная архитектура и общая структура данных;
- производительность вычислений;
- параллельные вычисления – многоуровневое распараллеливание и параллельная инфраструктура;
- встроенные средства профилирования и отладки.

Ключевым этапом в развитии кода является отказ от использования FORTRAN. В 2011-м году начата реализация NOISEtte на C++ и интенсивное развитие программного комплекса. С этого момента автором осуществлялось руководство программной реализацией.

На начало данной работы NOISEtte представлял из себя небольшой код с явной схемой и MPI распараллеливанием для нескольких десятков процессоров. В результате в рамках данной работы NOISEtte стал полноценным программным комплексом с многоуровневым распараллеливанием на десятки тысяч ядер, с множеством численных схем, методов, моделей, подходов к моделированию турбулентности и аэроакустики, с обширной параллельной инфраструктурой. В основе данной главы лежит подготовленный лично автором материал, представленный в статье [49] и программной документации NOISEtte.

Программный комплекс NOISEtte предназначен для решения задач вычислительной газовой и вычислительной аэроакустики, которые являются актуальными во многих отраслях промышленности, где присутствуют высокоскоростные турбулентные течения – в авиационной промышленности, турбомашиностроении и авиационном двигателестроении, ветроэнергетике, автомобилестроении и т. д.

Акцент на задачах аэроакустики делается, чтобы подчеркнуть более широкую область применимости NOISEtte, включающую, помимо стандартных задач аэродинамики, особый класс нестационарных расчётов высокоточными методами с применением дополнительных моделей, которые позволяют получать также данные по аэроакустическим свойствам моделируемого течения. Актуальность таких задач, в частности, в связи с постоянно ужесточающимися требованиями по шуму летательных аппаратов на местности, все больше возрастает интерес к разработке моделей и методов, направленных на изучение механизмов генерации шума в турбулентных течениях. В этой области типы задач, решаемых с помощью математического

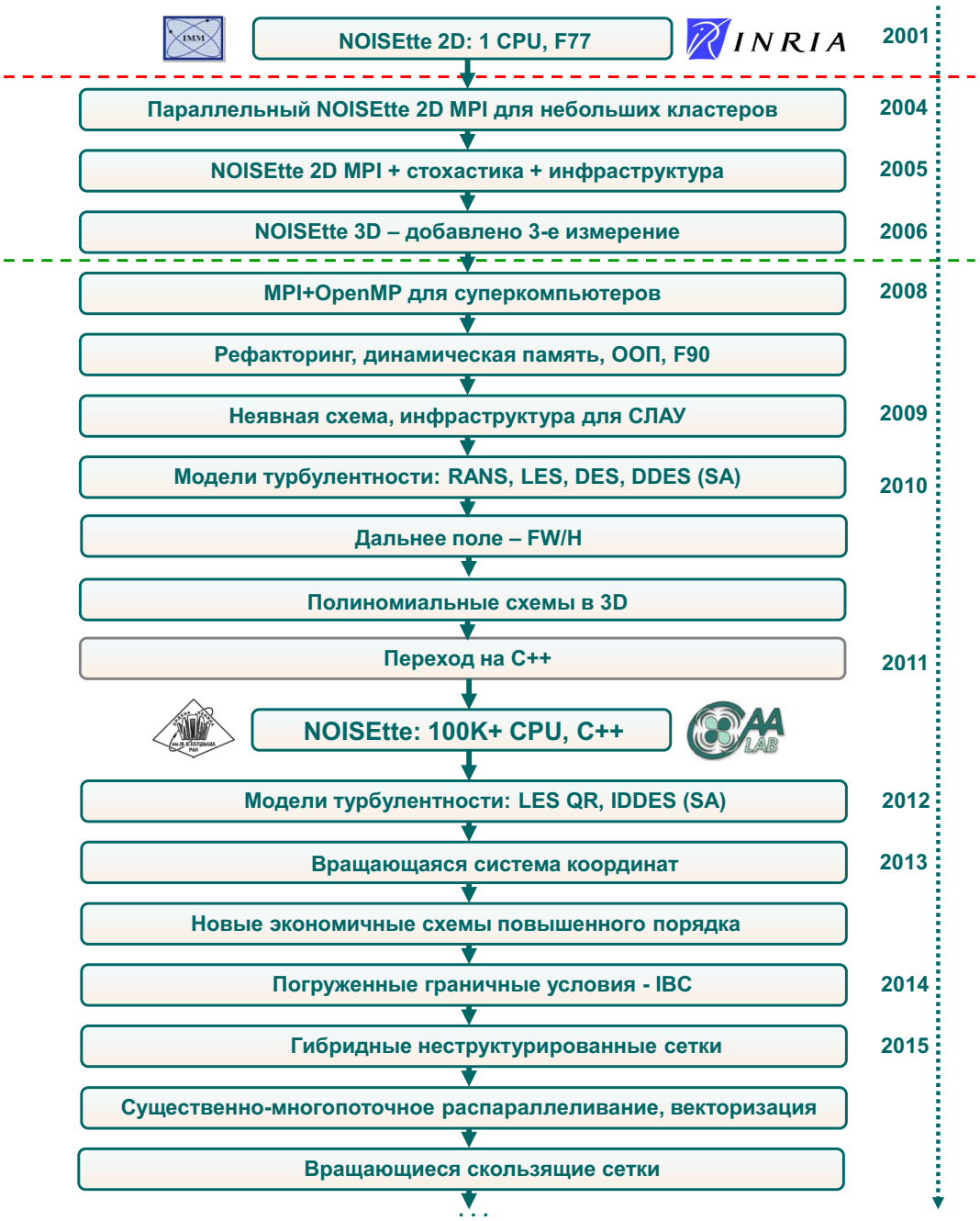


Рисунок 3.1: История развития программного комплекса NOISEtte

моделирования, можно условно разделить на три категории, соответствующие основным источником шума самолетов: 1) генерация шума высокоскоростной струей – шум от реактивной струи авиадвигателя; 2) шум при внешнем обтекании тел сложной геометрии, в частности, шум турбулентного следа и шум от взаимодействия турбулентных структур с твердым телом, – аэродинамический шум при обтекании планера, механизации крыла, шасси; 3) моделирование акустических резонаторов – оптимизация звукопоглощающих конструкций, специальных панелей, расположенных в различных частях мотогондолы и двигателя для снижения шума от вентилятора турбореактивного двигателя. Этот список можно дополнить также специфическими задачами по моделированию лопаток вентилятора, в частности, с целью снижения интенсивности ударных волн, возникающих на кромках лопаток, движущихся

со сверхзвуковой скоростью. Более подробно с состоянием дел в области вычислительной аэроакустики можно ознакомиться, например, в [82].

Моделирование задач аэроакустики представляет собой нестационарный газодинамический расчёт, дополненный вычислениями, обеспечивающими получение акустических возмущений в дальнем поле течения на основе интегральных методик. Данные задачи имеют несколько особенностей, обусловливающих высокую вычислительную стоимость. Во-первых, это необходимость использования численных схем повышенного порядка точности, чтобы корректно воспроизводить и турбулентное течение, и акустические волны, имеющие очень небольшую амплитуду по сравнению с турбулентными пульсациями. Во-вторых, необходимость высокого разрешения по пространству, особенно в области ближнего поля, где происходят процессы генерации шума. В-третьих, необходимость высокого разрешения по времени для получения широкополосных спектров, в сочетании с необходимостью интегрирования по длительному временному интервалу для получения качественного осреднения и, в частности, корректного спектрального анализа.

Программный комплекс включает в себя численные методики решения систем уравнений Эйлера и Навье-Стокса для сжимаемого газа, а также численную реализацию моделей, построенных на основе системы уравнений Эйлера и используемых в вычислительной аэроакустике [83]. Для расчёта турбулентных течений в комплексе программ предусмотрены современные методы моделирования турбулентности – RANS, LES, DES, DDES, IDDES.

Основу вычислительного алгоритма составляют экономные схемы повышенной точности для неструктурированных гибридных сеток, что позволяет моделировать обтекание геометрически сложных объектов. Реализованы схемы как с определением переменных в узлах, так и с определением переменных в центрах сеточных элементов.

Двухуровневое распараллеливание MPI+OpenMP и инфраструктура для работы с большими неструктурированными сетками даёт возможность задействовать для одного расчёта несколько десятков тысяч процессорных ядер и использовать сетки с числом элементов более миллиарда. Необходимость применения сложной параллельной модели обусловлена особенностями современной архитектуры суперкомпьютеров с многоядерными вычислительными модулями.

Как известно, в коммерческих пакетах, широко используемых в промышленности, реализованы уже существующие и хорошо отработанные решения. В этом состоит принципиальное отличие от исследовательского кода, в котором постоянно реализуются и отрабатываются новые методы. Одной из основных отличительных особенностей NOISEtte, в существенной степени определяющей выбор подходов к программной реализации, является его двойное назначение: код ориентирован одновременно как на исследовательскую работу, решение фундаментальных задач, отработку новых моделей, численных методов и вычислительных технологий, так и на решение прикладных промышленных задач, что подразумевает удобство использования, высокую производительность и надежность. С обзором по наиболее известным европейским исследовательским кодам в области газовой динамики читатель может ознакомиться, например, в [84].

Из отечественных комплексов программ наиболее близким NOISEtte по типу реализованных алгоритмов представляется LOGOS-TVD (входит в состав пакета LOGOS, разрабатывается в РФЯЦ ВНИИЭФ под руководством Ю. Н. Дерюнига) для решения задач аэродинамики и аэроакустики, в котором реализованы схемы повышенной точности на неструктурированных сетках, подходы к моделированию турбулентности RANS, LES, DES. Программный комплекс NESVETAY-3D [85] (В. А. Титарев, ВЦ им. А. А. Дородницына РАН) для моделирования сжимаемых турбулентных течений также использует схемы повышенного порядка на неструктурированных сетках. Другим примером, близким по области применения, является программный комплекс CABARET-RS (разрабатывается в ИБРАЭ РАН и МГУ им. М. В. Ломоносова под руководством В. М. Головизнина), который в недавнее время был обобщён на неструктурированные сетки, но интегрирование по времени ограничено явными схемами.

В качестве наиболее близкого аналога в классе алгоритмов повышенной точности на блочных структурированных сетках для расчёта сложных турбулентных течений и явлений аэроакустики, в первую очередь, необходимо отметить код NTS [86] (разрабатывается в СПбПУ и компании NTS под руководством М. Х. Стрельца). В классе пакетов, работающих на блочных структурированных сетках, можно отметить и пакет HSFlow (разрабатывается в ЦАГИ и ФАЛТ МФТИ под руководством И. В. Егорова). Особенностью пакета VP2/3 (разрабатывается под руководством С. А. Исаева в СПбГУГА) является эффективное использование многоблочных вычислительных технологий с использованием пересекающихся разномасштабных сеток. Пакет EWT-ЦАГИ (разрабатывается в ЦАГИ и ФАЛТ МФТИ под руководством С. М. Боснякова) ориентирован на численное воспроизведение экспериментов в аэродинамических трубах. Он создан на основе конечно-объёмного метода на структурированных сетках с использованием различных типов схем интегрирования по времени: явные схемы, схемы с дробным шагом, неявные схема, а также многосеточные ускорители сходимости. Широкой областью применения и большим набором реализованных методов отличается пакет SigmaFlow (разрабатывается в ИТ СО РАН под руководством А. А. Дектерева). Также в классе методов повышенной точности на структурированных сетках можно отметить программный комплекс JET3D (Д. А. Любимов, ЦИАМ им. П. И. Баранова) для расчёта сложных турбулентных дозвуковых и сверхзвуковых течений. Для полноты картины стоит отметить и блочно-структурированный код SINF (разрабатывается в СПбПУ по руководством Е. М. Смирнова) для моделирования несжимаемых течений (который по области применения ближе к коду STG-CFD&HT, представленному в 5-й главе данной работы).

В классе методов на адаптивных сетках типа восьмеричное дерево лидером является отечественный коммерческий пакет FlowVision, разрабатываемый компанией Тесис. Также методы на адаптивных сетках такого типа, но для несжимаемых течений (что ближе к 5-й главе данной работы), используются, например, в программном комплексе [87] (ИВМ РАН).

Основными отличительными особенностями разработанного кода NOISEtte относительно перечисленных выше программных комплексов является: 1) применение численных схем на основе квазиодномерной реконструкции переменных на гибридных неструктурированных

сетках, что обеспечивает повышенную точность результата при относительно низкой вычислительной стоимости; 2) универсальность единого вычислительного ядра, совмещающего без потери эффективности методы для 2D и 3D задач на основе как вершинно-, так и элементно-центрированных схем; 3) высокая степень параллелизма и многоуровневое распараллеливание, рассчитанное на десятки тысяч процессоров, что позволяет выполнять крупномасштабные расчёты нестационарных задач на сетках порядка миллиарда элементов.

3.1 Математические модели и численная реализация

3.1.1 Математические модели

Математические модели, лежащие в основе параллельного алгоритма программного комплекса, более подробно описаны в предыдущей главе. Здесь только кратко перечислены основные модели, методы и подходы, реализованные в программном комплексе.

В NOISEtte за основу взята модель течения сжимаемого газа, описываемая полными уравнениями Навье-Стокса. Как частные случаи этой модели, в NOISEtte реализовано семейство моделей, построенных на основе уравнений Эйлера [83]: полные уравнения Эйлера; линеаризованные уравнения Эйлера; уравнения Эйлера, нелинейные для пульсаций.

Базовая система уравнений Навье-Стокса дополняется необходимыми источниковыми членами и уравнениями для моделирования турбулентных течений в рамках подходов:

- DNS (direct numerical simulation) – прямое численное моделирование;
- RANS (Reynolds averaged Navier-Stokes) – осреднённые по Рейнольдсу уравнения Навье-Стокса;
- LES (large eddy simulation) – моделирование крупных вихрей;
- гибридные RANS-LES подходы семейства DES (detached eddy simulation) – моделирование отсоединённых вихрей.

Для расчёта стационарных турбулентных течений с большими числами Рейнольдса в качестве замыкания RANS уравнений используются модели турбулентности:

- Спаларта-Аллмараса (SA) [88];
- К-Омега (Вилкокса) [62];
- К-Эпсилон [63];
- SST Ментера [64].

В частности, выбор модели SA обусловлен следующими причинами: относительная простота модели, включающей одно дополнительное дифференциальное уравнение, физически корректное моделирование различных типов течений (пограничные слои, струи, отрывные течения и т.д.), удобство использования в гибридных RANS-LES подходах.

Для моделирования турбулентных течений методом LES реализован набор алгебраических моделей турбулентности:

- модель Смагоринского [65];
- модель WALE с адаптирующейся к стенке вихревой вязкостью [66];
- модель Времана [67];
- модель Верстаппена [68];
- σ-модель [69];
- модели семейства S3 – S3PQ, S3QR, S3PR [70], в которых улучшена точность вблизи твёрдых поверхностей (более подробная информация об этих моделях представлена, например, в [70]).

При моделировании с использованием RANS во многих случаях невозможно достичь необходимой точности в случае нестационарных течений, а использование LES часто требует слишком высокой вычислительной стоимости. Наиболее подходящими для расчёта высокорейнольдсовых турбулентных течений являются гибридные методы, в частности, методы семейства DES [89]. Преимущество данного подхода заключается в сочетании сильных сторон RANS и LES методов – высокой точности и низкой вычислительной стоимости полуэмпирической модели RANS в областях присоединённого пограничного слоя и достаточной универсальности LES метода в отрывных областях течения [7]. Из гибридных RANS-LES подходов в NOISEtte реализованы следующие незонные подходы семейства DES:

- DES-97 [71];
- DDES [72], улучшающий исходную версию DES, которая некорректно работает при расчёте толстых пограничных слоев и узких отрывных зон, где в областях перехода между RANS и LES внутри присоединённого пограничного слоя из-за низкого сеточного разрешения может возникать нефизический отрыв пограничного слоя;
- IDDES (Improved DDES – улучшенный DDES) [8] с улучшенным LES моделированием вблизи твёрдой поверхности;
- DES с ускорением “численного” перехода в слоях смешения [73].

Для моделирования распространения акустических возмущений в дальнем поле в NOISEtte применяется метод Фокса-Уилльямса-Хокингса (FW/H). Из расчётной сетки вырезается поверхность, окружающая течение в источниковой зоне (ближнем поле). По пространственно-временному распределению газодинамических переменных на ней расчитывается отклик в дальнем поле как поверхностный интеграл с запаздыванием. С параллельной реализацией метода можно ознакомиться в [47].

Для моделирования твёрдых тел, поверхность которых не согласована с пространственной сеткой, используется метод погруженных граничных условий IBC (immersed boundary condition). Данный метод может применяться, в частности, для моделирования подвижных объектов, для которых не так критично разрешение пограничных слоев. К таким задачам можно, например, отнести моделирование отделение груза от летательного аппарата (ЛА), когда погранслойная сетка строится вокруг ЛА, а груз реализуется методом IBC; моделирование вертолета, (сетка

строится вокруг винта, а фюзеляж реализуется методом IBC во вращающейся системе координат); моделирование выпуска дефлекторов и т. д. Для реализации IBC используются два метода: метод Бринкмана штрафных функций [90], характеристический метод штрафных функций (CBVP) [91].

Для генерации входных сигналов и турбулентных полей используются стохастические методы: белый шум или шум по заданному спектру, модель SNGR (Stochastic Noise Generation and Radiation) – стохастическая генерация шума и излучение [92].

3.1.2 Численная реализация

Численная реализация более подробно описана в предыдущей главе, здесь только кратко перечислены основные численные схемы и методы, реализованные в программном комплексе.

Аппроксимация по пространству

Для пространственной дискретизации системы уравнений Навье-Стокса используется метод конечного объёма на неструктурированных гибридных сетках. Формулировка метода конечных объёмов требует разбиения расчётной области на контрольные объёмы – ячейки. В случае элементно-центрированной схемы с определением значений сеточных функций в центрах сеточных элементов, ячейками являются сами сеточные элементы. В вершинно-центрированном случае, когда значения сеточных функций заданы в узлах сетки, контрольный объём строится вокруг узла сетки, с использованием центров рёбер, центров граней сеточных элементов, опирающихся на этот узел, и центров самих элементов. Выбор центров граней и элементов может осуществляться различными способами в зависимости от особенностей сетки и желаемой аппроксимации конвективного оператора. В частности, для тетраэдральной сетки в NOISEtte определяются два типа контрольных объёмов – барицентрический (медианный) и ортоцентрический [93]. Для построения барицентрического объёма, выбираются центры масс тетраэдра, центры масс граней и середины ребер тетраэдра (пример барицентрического контрольного объёма показан на рис. 3.2). При построении ортоцентрического контрольного объёма берется центр центр описанной сферы (для тетраэдра) и окружности (для грани тетраэдра). Если центры описанной окружности или сферы не принадлежат симплексу (тетраэдру или треугольнику), то в качестве центра выбирается центр его наибольшей грани (или ребра).

В рамках конечно-объёмного подхода объёмный интеграл от производных преобразуется в сумму поверхностных интегралов по границам (сегментам) ячеек. Сегментом ∂C_{ij} расчётной ячейки называется общая граница, разделяющая соседние ячейки между i и j. Понятие сегмента вводится в дополнение к понятию грани ячейки, поскольку, в общем случае эта граница может не лежать в одной плоскости, а представлять собой объединение множества граней, лежащих в

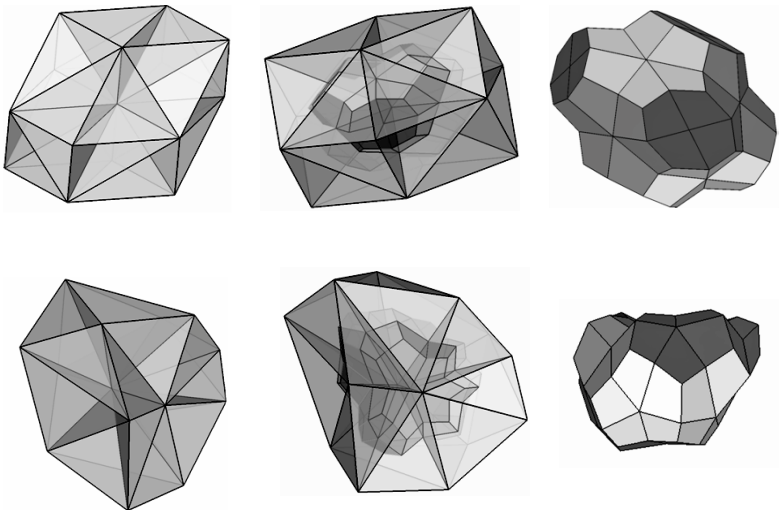


Рисунок 3.2: Барицентрические контрольные объёмы. Сверху – для сетки, полученной разбиением декартовой сетки на тетраэдры, снизу – для произвольной неструктурированной сетки. Слева направо: тетраэдры, содержащие центр контрольного объёма, объём внутри тетраэдров, объём в отдельности

разных плоскостях. В этом случае в качестве нормали к сегменту выбирается среднее значение $\mathbf{n}_{ij} = \iint\limits_{\partial C_{ij}} \mathbf{n}\, d\sigma$.

Нормальный газодинамический поток на сегменте заменяется на численный поток. Конвективный поток через сегмент ∂C_{ij} вычисляется путём решения задачи Римана о распаде разрыва по схеме годуновского типа относительно предраспадных реконструированных значений “слева” и “справа” от центра сегмента. Для определения численного потока реализовано несколько схем:

- схема Роу [51] (в предположении постоянства газодинамических переменных на соседних ячейках);
- противопоточная схема Хуанг [52] (в предположении постоянства газодинамических потоков);
- схема Годунова [94].

Для повышения порядка точности численного потока производится замена кусочно-постоянных газодинамических или потоковых переменных на кусочно-полиномиальное распределение. Для реконструкции значений “слева” и “справа”‘ от центра сегмента реализованы различные численные схемы повышенного порядка аппроксимации, как вершинно-, так и элементно-центрированные. Для повышения порядка в реализованных семействах схем используются два основных подхода:

- квазиодномерная реконструкция – EBR схемы (edge-based reconstruction);
- многомерная полиномиальная реконструкция.

Реализованные схемы до 6-го порядка аппроксимации с квазиодномерной реконструкцией, или так называемые EBR схемы, включают нижеследующие схемы.

– Вершинно-центрированная многопараметрическая схема [31, 53]. Шаблон реконструкции (рис. 3.3) для интерполяции значений использует два “противопотоковых” сеточных элемента, а также используется расчёт узловых градиентов.
– Вершинно-центрированная схема в [12, 54], в которой вместо узловых градиентов используются “противопотоковые” элементы 2-го уровня.
– Вершинно-центрированная экономичная EBR схема [55], которая использует по 2 грани с двух уровней противопотоковых элементов с каждой стороны.
– Элементно-центрированная экономичная EBR схема [13, 56] с двумя лучами реконструкции.

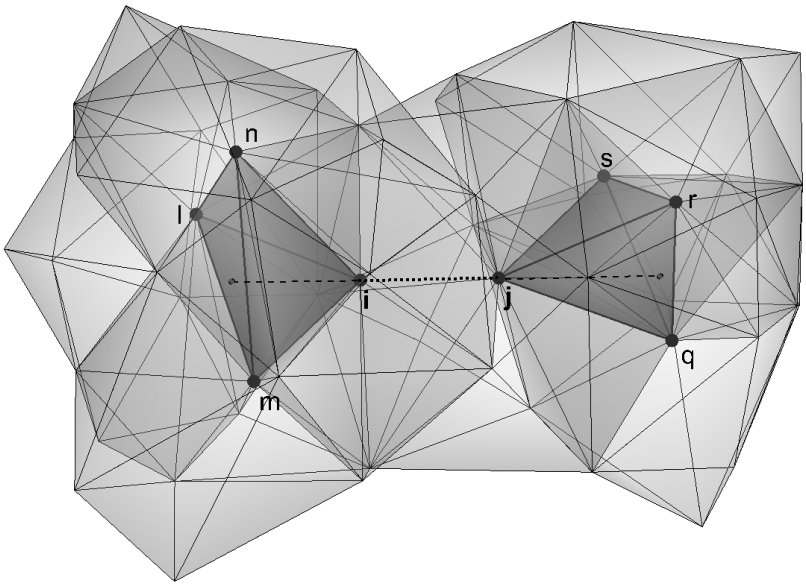


Рисунок 3.3: Шаблон схемы для расчёта конвективной части потоков через грань контрольного объёма между узлами i и j. Базовая часть шаблона содержит узлы левого и правого противопотоковых тетраэдров, а для расчёта узловых градиентов используются все соседние с ними узлы

В качестве дополнительных расширений для моделирования низкоскоростных сжимаемых течений используется предобуславливатель Туркеля [95], а для моделирования течений с разрывами (высокоскоростные сжимаемые течения с ударными волнами) используются специальные версии EBR схем с ограничителями: WENO-EBR и MUSCL-TVD-EBR [96, 97].

Аппроксимация по времени

Для дискретизации по времени используются как явные, так и неявные схемы:

– явные схемы семейства Рунге-Кутты [58, 59] до 4-го порядка точности;
– двухслойная неявная схема 1-го с линеаризацией по Ньютону;
– трёхслойная неявная схема 2-го с линеаризацией по Ньютону.

Реализованные неявные с линеаризацией по Ньютону имеют как полные как полные, так и упрощённые версии, в которых вычисление коэффициентов якобиана происходит только на нулевой итерации ньютоновского процесса. Для дальнейшего упрощения вычислений и уменьшения потребности памяти для хранения якобиана берется поток, конвективная часть которого подсчитана на более узком шаблоне, чем шаблон схемы повышенного порядка, например, по базовой схеме 1-го порядка. За счёт этого улучшаются свойства матрицы и многократно уменьшается количество ненулевых элементов, но, с другой стороны, ухудшается сходимость ньютоновских итераций (сходимость уже не квадратична) и возникают ограничения по числу Куранта. Получающаяся в результате линеаризации линейная система уравнений решается стабилизированным методом бисопряженных градиентов (BiCGSTAB) [77], который более подробно описан в предыдущей главе.

Постановка граничных условий

В NOISEtte реализованы различные типы граничных условий (ГУ), включая нижеследующие.

- ГУ прилипания (равенство нулю скоростей на границе) на твёрдой стенке: задание температуры стенки (изотермическая поверхность), задание на стенке потока тепла, и, как частный случай, адиабатическая поверхность (равенство нулю теплового потока). При такой постановке потоки массы и импульса равны нулю. Поток полной энергии вычисляется либо через заданный поток тепла, либо через температуру стенки. Значение газодинамических параметров на границе определяется начальными условиями и потоками через грани ячейки, не лежащими на границе.
- Условия симметрии (или непротекания – твёрдая стенка при отсутствии вязкости), при которых нормальная к поверхности компонента скорости равна нулю, что приводит к нулевым потокам массы и полной энергии и ненулевым потокам импульса; Потоки импульса вычисляется через значение давления на границе, умноженного на соответствующую компоненту вектора внешней нормали к граничной поверхности.
- Условия Дирихле – задание полного набора газодинамических параметров на границе. При таком подходе потоки на границе определяются точным невязким потоком, вычисленным по заданным скоростям, плотности и давлению. Применение ГУ такого типа ограничено случаем сверхзвуковых входных условий.
- ГУ на дозвуковых входных и выходных границах определяются потоками, расщеплёнными по знаку характеристических скоростей на границе поверхности [57]. Сами потоки вычисляются с использованием значений газодинамических параметров на границе и значений вне расчётной области, которые определяются либо сносом по характеристикам изнутри расчётной области, либо задаются.
- Неотражающие ГУ Тама [76], основанные на асимптотическом решении линейных уравнений Эйлера.

- Неотражающие ГУ Л. Дородницына [98].
- Периодические ГУ по пространственным направлениям с топологическим замыканием краев расчётной области.
- Периодические ГУ по углу вращения с топологическим замыканием краев расчётной области (например, чтобы свести моделирования многолопастного винта к расчёту одной лопасти).

3.2 Особенности программной реализации и процесса разработки

3.2.1 Требования к программному комплексу

Исходный код NOISEtte написан на языке C++ (стандарта C++03 2003 года) с использованием интерфейсов прикладного программирования MPI (стандарт 1.3) для параллельной модели с распределённой памятью, и OpenMP (стандарт 3.0) для параллельной модели с общей памятью. Выбор подходов к программной реализации был обусловлен следующими требованиями переносимости:

- совместимость с различными операционными системами (ОС), в частности, ОС семейства Windows и Linux;
- совместимость с различными типами вычислительных систем от рабочих станций до крупных суперкомпьютеров;
- совместимость с разными архитектурами процессоров, в частности, Intel, AMD, IBM (имеющей обратный порядок байтов);
- совместимость с различными компиляторами C++, в том числе Microsoft VS C++, Intel C++, GNU C++, IBM XL C++;
- автономность – базовая конфигурация может работать без подключения каких либо дополнительных библиотек, отличных от стандартных библиотек C++.

Переносимость и совместимость подразумевают, что базовая конфигурация не использует никаких программных решений, зависящих от типа ОС и конкретной архитектуры вычислительной системы. Однако, это не исключает расширений базовой конфигурации, в которых используются средства конкретных ОС, настройка под конкретный тип процессора или возможности конкретного компилятора. Аналогичным образом, автономность не исключает дополнения базовой конфигурации функционалом внешних библиотек.

Также естественными требованиями к программному комплексу являются удобство работы с исходным кодом, надежность и защищённость кода от внесения ошибок, высокая производительность вычислений. Высокая степень параллелизма обеспечивается разработкой алгоритма и программной реализацией по технологии, представленной в первой главе.

NOISEtte рассчитан на вычислительные системы с числом ядер порядка сотни тысяч. Кроме того, все операции лежащего в основе параллельного алгоритма совместимы с SIMD параллелизмом и парадигмой потоковой обработки. В соответствии с требованием высокой производительности вычислений особое внимание уделяется эффективности реализации алгоритмических операций и, в частности, эффективности доступа к памяти, поскольку данному классу алгоритмов характерна низкая вычислительная стоимость на единицу данных и, как следствие, производительность ограничена в первую очередь пропускной способностью оперативной памяти.

Производительность вычислений при использовании явной схемы составляет порядка $12 \sim 15\%$ от пиковой производительности CPU ядра, для неявной схемы этот показатель несколько ниже, порядка $10 \sim 12\%$, из-за операции матрично-векторного произведения с разреженной матрицей в решателе СЛАУ, которая имеет крайне низкий предел теоретически достижимой производительности из-за ограничений пропускной способности памяти. Данные показатели представляются сравнительно высокими для такого класса задач с нерегулярным доступом к памяти и низкой удельной вычислительной стоимостью на единицу данных. Удельный расход памяти для схемы повышенного порядка с включенной моделью турбулентности (т.е. присутствуют дополнительные переменные) составляет порядка 2 КБ на ячейку для явной схемы и $4 \sim 5$ КБ для неявной.

3.2.2 Особенности программной архитектуры

NOISEtte состоит из вычислительного ядра (с помощью которого непосредственно выполняется расчёт) и обширной инфраструктуры, включающей в себя средства для постановки задачи, подготовки сеточных данных, обработки результатов. Вычислительное ядро реализовано в виде библиотеки статической компоновки, от которой питаются исполнимые файлы расчётного кода и инфраструктуры. Средства инфраструктуры разделены на средства препроцессора (то есть программы, выполняющиеся до начала расчёта) и средства постпроцессора (то есть программы обработки результатов).

Особенностью программной архитектуры является совмещённая реализация (как на уровне вычислительного ядра, так и на уровне средств инфраструктуры) для выполнения 2D и 3D расчётов. Для переключения между 2D и 3D задачами не требуется перекомпиляция кода, при этом 2D расчёты выполняются на “честной” двухмерной сетке (а не на фиктивной трёхмерной сетке с минимальным количеством слоев по одному направлению), что многократно повышает скорость выполнения расчёта. Аналогично совмещены вершинно- и элементно-центрированные схемы. Переход с одного типа дискретизации на другой не требует ни перекомпиляции ни и изменения сеточных данных. Для представления сетки используется общий унифицированный формат. При реализации вычислений для задач различной размерности и для различных типов дискретизации максимизируется повторная используемость кода. Компоненты библиотеки

вычислительного ядра и структуры данных максимально унифицированы, при том таким образом, что не наносится ущерб производительности вычислений.

Другой особенностью, влияющей на программную архитектуру является многостороннее предназначение NOISEtte: код ориентирован одновременно как на исследовательскую работу, решение фундаментальных задач, отработку новых моделей, численных методов и вычислительных технологий, так и на решение прикладных промышленных задач. Исследовательская работа подразумевает возможность выполнения крупномасштабных расчётов фундаментального характера, в которых в высокоточном численном эксперименте на подробных сетках исследуются физические явления и процессы. Для разработки новых методов и моделей необходимо удобство и легкость внесения изменений в код, добавления новых структур данных и изменения существующих, а также высокая защищённость от внесения ошибок. Решение промышленных задач подразумевает удобство пользования программным комплексом, высокую производительность, надежность и устойчивость.

3.2.3 Среда разработки

Работа с кодом ведется в интегрированной среде разработки (IDE) Microsoft Visual Studio 8.0 (MVS) и выше, а также в редакторах для работы с исходным кодом под управлением ОС Linux. Для коллективной работы с кодом и контроля за внесением изменений используется система управления версиями Subversion. Для отслеживания ошибок и управления задачами используется веб-приложение Redmine (в связке с SVN). Отладчик и статический анализатор – входящие в состав MVS. Для профилирования и поиска ошибок по доступу к памяти используется Intel Parallel Studio. Под управлением ОС Linux для этих целей используются средства Valgrind и gdb. NOISEtte также имеет свои внутренние средства контроля, профилирования и отладки (система таймирования, контроль стека вызовов функций, счётчик фактической производительности, менеджер памяти). Для визуализации результатов в основном используется программное обеспечение Tecplot и Origin. Средства постпроцессора NOISEtte позволяют создавать файлы в бинарном формате Tecplot.

Контроль качества включает в себя:

- модуль точных решений, который содержит набор аналитических решений и функции расчёта норм ошибки;
- набор из десятков коротких тестов и средства для автоматизированного регрессионного тестирования, при котором в тестах на небольшой сетке выполняется несколько шагов интегрирования по времени от моментальной картины течения, хранящейся в файле записи восстановления счёта, и выдаются нормы расхождений с результатами предыдущей проверенной версии;
- набор верификационных тестов для более полной проверки качества результатов и правильности конечного решения, которые выполняются полностью и на существенно более подробных сетках.

Документация программного комплекса включает в себя подробную документацию по математической части, в которой описываются используемые математические модели, методы и подходы; описание структуры вычислительного ядра и средств инфраструктуры; набор шаблонов постановок задач и входных файлов; обучающие примеры. Информация для разработчика реализована в виде комментариев по исходному коду, которые в сочетании со средствами интегрированной среды разработки представляются более удобными, чем отдельный текстовый документ. Системы автоматической документации не применяются.

NOISEtte не имеет своего графического интерфейса, поскольку использование графических средств для постановки задачи и управления расчётом представляется неэффективным и нецелесообразным. Для генерации сеток и визуализации результатов используются графические средства, входящие в состав соответствующего внешнего программного обеспечения. Постановка задачи задаётся набором входных текстовых файлов, имеющих единую упрощённую структуру. Формат XML в файлах входных данных не применяется.

3.3 Средства препроцессора

3.3.1 Структура

Основу препроцессора составляют параллельные средства, реализованные коллективом разработчиков NOISEtte, для работы с неструктурированными сетками большого объёма (с числом элементов более миллиарда). Общая схема средств препроцессора показана на рис. 3.4.

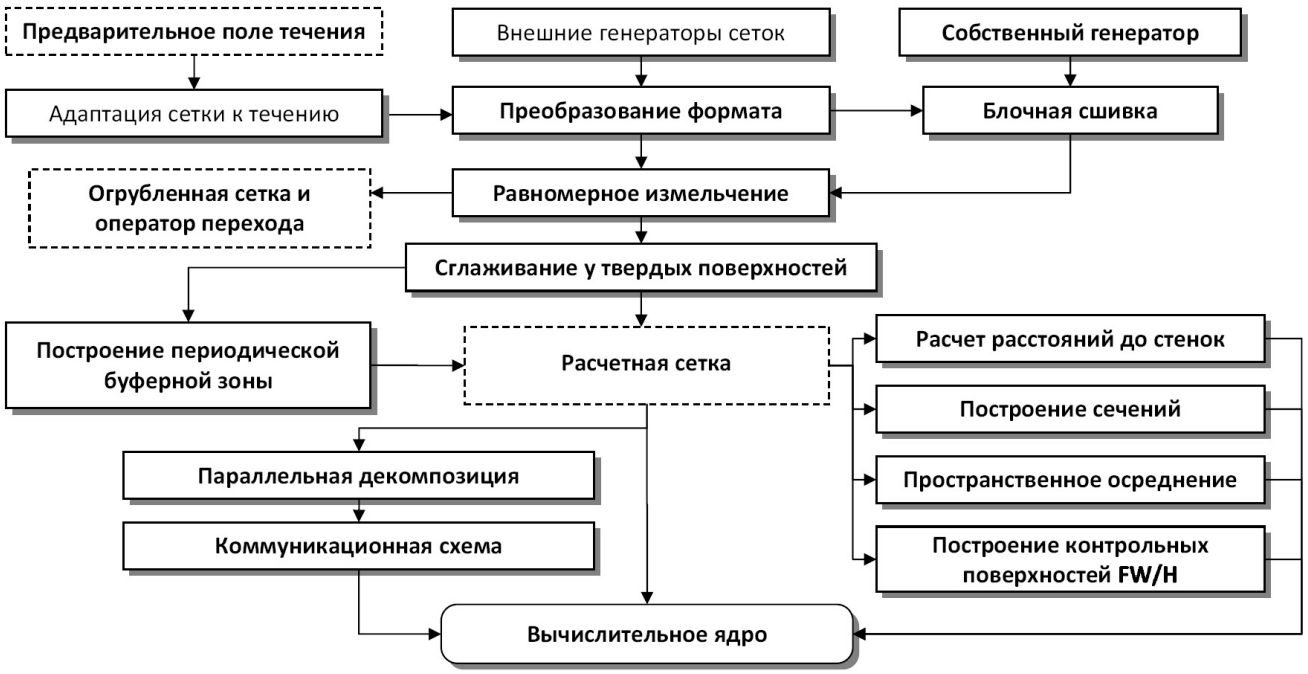


Рисунок 3.4: Схема работы средств обработки сеточных данных

3.3.2 Построение сетки

Для генерации сеток используются либо коммерческие генераторы, либо собственный сеточный генератор. Ограничения последовательных генераторов не позволяют получать сетки с большим числом элементов (порядка 10^8 – 10^9). Поэтому приходится либо сшивать сетку из множества блоков, либо равномерно измельчать, либо и то и другое. Необходимая функциональность реализована в следующих программах.

- Преобразователь сеток из внешних форматов коммерческих генераторов и кодов, включая Gambit, Icem, CFX (В. Г. Бобков).
- Генератор двухмерных сеток путём сшивки блоков, топологически эквивалентных декартовой решётке, и разбиением на треугольники.
- Генератор трёхмерных тетраэдральных сеток из двухмерных треугольных вытягиванием с постоянным шагом по третьей оси и разбиением получающихся призматических элементов на тетраэдры.
- Сшивка сетки из блоков с совпадающими границами.
- Параллельное средство для равномерного измельчения сетки (С. А. Суков). Исходная сетка распределяется между параллельными процессами, процессы выполняют измельчение, разделяя каждый тетраэдр на 8 тетраэдров, затем выполняется сшивка подобластей. Результирующая сетка имеет примерно в 2 раза большее пространственное разрешение в каждом из трёх направлений. В дополнение к сетке записывается индексный массив, ставящий в соответствие узлы исходной и измельченной сетки, необходимый для оператора перехода с одной сетки на другую. Оператор перехода с грубой сетки на подробную используется для генерации начального поля течения из результатов предварительного расчёта на грубой сетке. Переход с подробной на грубую применяется для визуализации.
- Сглаживание сетки у твёрдых поверхностей. Операция равномерного измельчения не улучшает качество представления геометрии расчётной области, поскольку новые узлы добавляются на существующие ребра тетраэдров. Для повышения качества геометрии, новые узлы, принадлежащие поверхностям расчётной области, перемещаются в соответствии с параметризацией геометрии. Внутренняя реализация позволяет работать только с простейшими поверхностями, заданными параметрически.
- Построение буферной зоны для периодических граничных условий методом топологического замыкания краев сетки фиктивными тетраэдрами (П. А. Бахвалов).
- Параллельная программа расчёта расстояний до твёрдых поверхностей. Для работы модели турбулентности в каждом узле сетки вычисляется расстояние до ближайшей твёрдой поверхности (А. П. Дубень).
- Вырезка параметрически заданной поверхности из трёхмерной расчётной области (для расчёта дальнего поля или визуализации). На выходе – двумерная сетка и оператор перехода с расчётной сетки для быстрой выгрузки данных.

3.3.3 Декомпозиция расчётной области

Для параллельных вычислений неструктурированная сетка должна быть разбита на подобласти MPI процессов. К разбиению предъявляются следующие очевидные требования: минимизация площади границ подобластей – для уменьшения объёма обмена данными, минимизация максимального числа соседних подобластей – для уменьшения количества пересылок, балансировка загрузки. После разбиения должна быть построена коммуникационная схема для конкретного шаблона численной аппроксимации, то есть для каждого узла сетки необходимо определить, какому процессу, помимо процесса-владельца, он требуется для вычислений по своим узлам. Эту задачу решают следующие программы:

- программа декомпозиции на основе библиотек Metis, ParMetis [35];
- параллельная программа декомпозиции GridSpider [37] ИПМ РАН;
- параллельная программа построения коммуникационной схемы для заданной ширины шаблона пространственной аппроксимации.

3.4 Структура вычислительного ядра

3.4.1 Основные вычислительные модули

Вычислительное ядро программного комплекса реализует алгоритм, представленный предыдущей главе. Алгоритм составлен из базовых операций нескольких типов, каждая из которых имеет MPI+OpenMP распараллеливание и адаптирована к парадигме потоковой обработки. Программная реализация выполнена согласно технологии, представленной в первой главе.

Исходный код библиотеки вычислительного ядра условно разделен на следующие функциональные модули:

- модуль граничных условий: содержит обработчик внешних граней расчётной области и набор граничных условий;
- модуль точных решений: содержит обработчик точных решений в узлах сетки и набор точных решений для различных модельных задач;
- геометрический модуль: построение геометрии контрольных объёмов, расчёт площади и нормали сегментов, расчёт объёмов сеточных элементов и ячеек, поиск противопотоковых элементов и так далее.
- модуль формирования начальных данных: содержит набор начальных полей течения для различных типов задач;
- модуль неявного интегрирования по времени: реализует неявные схемы с линеаризацией по Ньютону;

- модуль линейной алгебры: содержит набор алгебраических операций для блочных разреженных неструктурированных матриц и решатель СЛАУ на основе метода бисопряжённых градиентов;
- модуль ввода-вывода: интерпретатор пользовательского ввода, функции чтения и записи полей в бинарном и текстовом формате, функции записи динамических данных;
- модуль пространственной аппроксимации: расчёт величины шага по времени, расчёт потоков через грани контрольных объёмов, расчёт узловых и тетраэдральных градиентов, реконструкция потоков, и т.д.;
- модуль вязкости: вычисление вязких потоков в уравнении Навье-Стокса;
- параллельный модуль: содержит дополнительные структуры данных и инфраструктуру для параллельных вычислений с использованием MPI и OpenMP;
- модуль источников: задаёт источниковый член в правой части уравнения для различных типов задач;
- турбулентный модуль: реализует набор моделей и методов для моделирования турбулентности;
- модуль IBC: реализует действие погруженных граничных условий;
- модуль скользящих интерфейсов: для использования расчётной области из множественных вращающихся друг относительно друга подобластей, соединённых через общие плоские и/или цилиндрические поверхности;
- модуль автокоррекции: с помощью набора критериев корректности расчёта и посредством хранения точек восстановления счёта в памяти обеспечивает автоматическое восстановление вычислений с коррекцией параметров (уменьшение числа Куранта, уменьшение критерия невязки решателя СЛАУ, увеличение веса противопотоковой схемы и т.д.);
- модуль вычислительной инфраструктуры: решатель СЛАУ малой размерности, алгоритмы сортировки и т.д.
- модуль профилирования и отладки: встроенные средства инструментального профилирования – профилирование, измеритель фактической производительности, встроенный менеджер памяти, встроенная диагностика стека вызовов функций.

3.4.2 Инфраструктура вычислительного ядра

Менеджер памяти – промежуточный интерфейс, через который динамически выделяется память. Он контролирует количество затраченной памяти, чтобы избежать проблем с попаданием в режим подкачки (swap), на некоторых системах являющийся проблематичным. Менеджеру может передаваться текстовая метка, идентифицирующая потребителя, что

позволяет разработчику контролировать распределение памяти по конкретным структурам данных. Для этого реализована печать диагностической таблицы по всем потребителям памяти.

Измерение времени вычислений – встроенная система ручного инструментального профилирования, измеряющая время для множества участков программы (каналов), идентифицируемых по текстовым меткам. Модуль рассчитан на гибридный параллельный режим и может выполнять измерения для разных нитей и процессов, что даёт возможность измерять дисбаланс загрузки. Измерения усредняются на заданном числе шагов по времени для повышения точности измерений и накопления статистики по каналам.

Интерпретатор пользовательского ввода обрабатывает набор входных текстовых файлов параметров, имеющих единую структуру. Для удобства работы с постановкой задачи формат пользовательского ввода был максимально упрощен: каждый файл содержит список пар – именованных параметров и их значений. Было решено отказаться от популярного в таких случаях языка XML. Каждый параметр может иметь свой тип данных и свойство обязательности. Если не задан обязательный параметр, происходит выдача соответствующей диагностики и остановка, а для необязательного параметра берется значение по умолчанию (также с выдачей диагностики). Несмотря на то, что число параметров достаточно велико, большинство параметров имеют значения по умолчанию и их можно не задавать. Такой подход позволяет добиться обратной совместимости при добавлении новых параметров. Входные параметры расчёта состоят из основного файла параметров и дополнительных файлов параметров для отдельных модулей. В основном файле определяются тип численного алгоритма, тип модели турбулентности, типы контрольных объёмов, время расчёта, интервалы записи данных и так далее. В дополнительных файлах задаются параметры обезразмеривания, параметры неявной схемы, параметры начальных данных и среднего поля, параметры осреднения, параметры записи результатов, дополнительные параметры граничных условий, параметры модели дальнего поля и так далее. Модули вычислительного ядра могут иметь свои конфигурационные файлы, что позволяет формировать постановку задачи из готовых блоков.

Счётчик FLOPs измеряет получаемую полезную производительность. Для каждого блока каждой из подпрограмм вычислительного ядра подсчитано фактическое количество операций с плавающей точкой. Количество выполненных операций накапливается в глобальном счётчике и усредняется на заданном числе шагов по времени. Такой подход позволяет достаточно точно оценить реально получаемую производительность, отбросив все сопутствующие операции, не входящие в алгоритм.

Диагностика стека вызовов функций реализуется добавлением к описанию функции одной строки обращения к макросу, в которой передаётся текстовая метка функции. Аналогичным образом можно отмечать и отдельные структурные блоки внутри функции. Макроподстановка создаёт экземпляр сервисного объекта, который в конструкторе вносит данные в таблицу стека вызовов, а в деструкторе, соответственно, автоматически удаляет внесённую в таблицу стека запись. По запросу или в случае аварийной остановки программы может выдаваться информация о текущем состоянии стека, что облегчает локализацию ошибки.

Параллельный решатель СЛАУ на основе метода бисопряженных градиентов с локальным предобуславливателем [77], применяется при неявном интегрировании по времени. В случае неявной схемы, получающаяся в результате линеаризации линейная система уравнений решается стабилизированным методом бисопряженных градиентов (BiCGSTAB) с локальным предобуславливателем (не требующим обмена данными между MPI процессами). Предложен оригинальный вариант решателя BiCGSTAB, адаптированный к структуре матрицы,состоящей из блоков размерности числа переменных в узлах, каждый из которых состоит из двух диагональных подблоков (для основных переменных и для переменных модели турбулентности) По сути, сразу решается две системы – для основной и турбулентной части, при этом все операции сгруппированы, что позволяет объединить обмены данными и избежать потерь на латентность сети. Более подробно алгоритм работы решателя описан в предыдущей главе.

3.5 Средства постпроцессора

Результаты записываются параллельными процессами в распределённые бинарные файлы специального формата. Файлы состоят из фрагментов данных, для каждого фрагмента указан размер для проверки целостности файла и контрольный код для определения порядка байтов (для переносимости данных между системами с разным порядком байтов, например между архитектурами Intel и IBM). Реализованы следующие типы записей: 1) распределённая запись набора полей на расчётной сетке или заданной подсетке – моментальные поля, осреднённые поля, записи восстановления счёта; 2) скалярные динамические данные – эволюция во времени выбранных значений в контрольных точках или глобальных величин (сопротивление, подъемная сила и т. д.); 3) динамические данные на поверхности интегрирования FW/H для моделирования акустики в дальнем поле [47]. Для обработки большого объёма данных (запись 1-го типа может занимать десятки гигабайт, 3-го типа – более терабайта) реализован набор параллельных средств:

- суммирование интервалов осреднения: осреднённые поля записываются интервалами, поскольку заранее не известен момент выхода течения на статистически однородный режим; после анализа данных и выбора момента начала интегрирования, плохие интервалы отбасываются, а хорошие – объединяются;
- расчёт статистики течения – дополнительных полей, которые можно вычислить на этапе обработки по основным полям, записанным в процессе счёта;
- сборка записи из распределённого формата (каждый MPI процесс записывает данные в свой файл по своей подобласти, каждая запись хранится в таком распределённом формате) – формирует единый файл в бинарном формате Tecplot для всей расчётной области или для заданной подсетки (сечения или огрублённой сетки);
- сборка интервалов записи динамических данных с автоматическим удалением перекрытий и интерполяцией на постоянный шаг по времени для Фурье анализа;

– расчёт дальнего поля по эволюции на контрольной поверхности для заданных позиций наблюдателя (П. А. Бахвалов);

– средства Фурье анализа для построения спектров, диаграмм направленности, и т.д (А. П. Дубень).

Структура постпроцессора показана на рис. 3.5

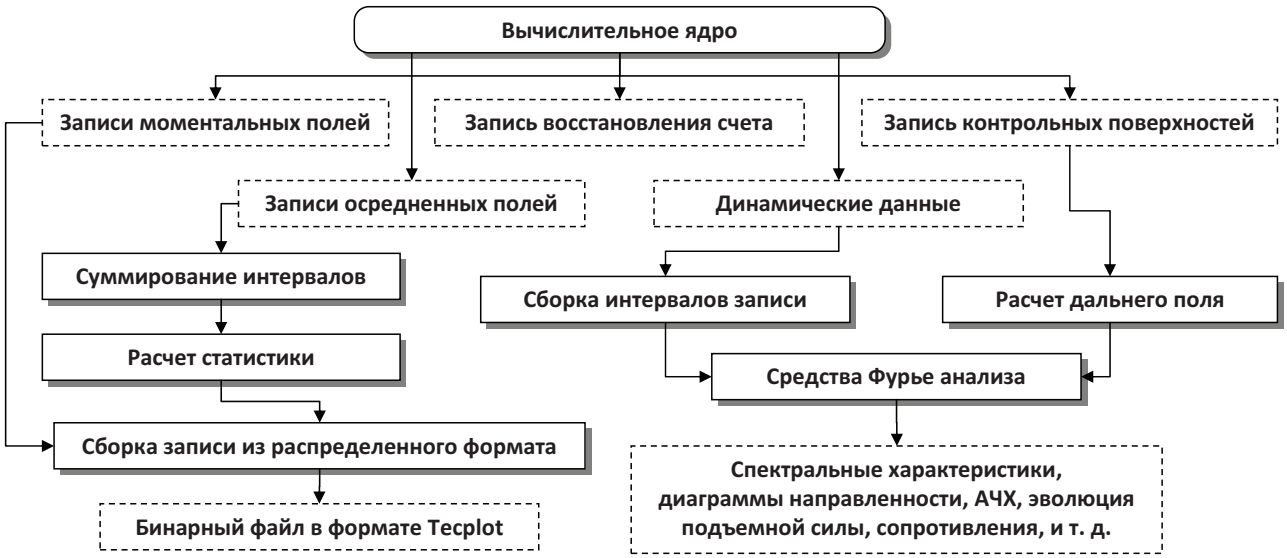


Рисунок 3.5: Схема работы средств обработки результатов расчёта

3.6 Представление данных расчётной области

Расчётная область на неструктурированной тетраэдральной сетке с заданием значений сеточных функций в узлах определяется в NOISEtte списком координат узлов сетки, списком сеточных элементов (топологией сетки), списком внешних граней сетки. Топология задаётся перечислением индексов узлов (вершин), составляющих каждый сеточный элемент. Вокруг узлов сетки выстраиваются контрольные объёмы, каждому ребру сетки соответствует грань (или группа граней) контрольного объёма. Таким образом, ребра и узлы сетки составляют граф связанности контрольных объёмов.

Расчётная область на неструктурированной тетраэдральной сетке с заданием значений сеточных функций в центрах сеточных элементов дополнительно имеет список координат центров сеточных элементов. Граням контрольных объёмов в данном случае соответствуют грани сеточных элементов. Графом связей контрольных объёмов является дуальный граф сетки, в котором вершинам соответствуют сеточные элементы, а ребрам – их грани.

В состав расчётной области входят следующие элементы: ячейка – это либо узел сетки и его контрольный объём, либо сеточный элемент; сеточный элемент – от 4-х до 6-ти граней; сегмент (грань) контрольного объёма; а также внешняя грань – треугольник или четырехугольник. Соответственно, возникает три основных типа наборов данных: данные по узлам, сеточным

элементам и сегментам. Данные представляются в виде блочных векторов размерности N по M элементов в блоке, где N – число элементов, M – число величин, заданных в элементе. Например, поля основных физических переменных представляются в виде блочного вектора с размером блока 5 – плотность, три компоненты скорости и давление.

Блочные вектора было бы естественно представить в виде двухмерных массивов размерности $N \times M$. Для этих целей реализован шаблонный класс, представляющий собой простейшую "обертку" над одномерным вектором данных: оператор *[]* возвращает указатель на заданный блок: *inline ValueType *operator[](int i){return V+i*M;}*. Здесь ValueType – тип данных, i – номер блока, V – указатель на начало вектора данных. Такой способ отличается низкими накладными расходами на позиционирование и высокой скоростью доступа к данным. Также для класса реализованы операторы присваивания и арифметические операции, что позволяет обращаться с блочными векторами как со скалярными переменными (по аналогии с возможностями языка Fortran). Конструктор и функция выделения памяти интегрированы с внутренним менеджером памяти NOISEtte.

Для представления блочных данных с переменным размером блока (например, топология гибридной сетки), реализован аналогичный класс с интерфейсом двухмерного массива. В данном блочном векторе с переменным размером блока дополнительно содержится массив указателей на начало каждого блока.

Характерно, что для реализации элементов расчётной области не используются сложные структуры данных в виде классов. Все данные расчётной области размещаются в блочных векторах, чтобы избежать потерь в производительности и затратах памяти. При этом оригинальная реализация доступа к данным с использованием макроподстановки позволяет имитировать доступ к свойствам класса. Рассмотрим, например, сегмент контрольного объёма, для описания которого нужен класс, обладающий следующими свойствами: номера "левой" и "правой" ячеек, которые разделяет сегмент, номера "левого" и "правого" противопотоковых элементов (только для EBR схем повышенного порядка), номера противопотоковых элементов второго уровня слева и справа (только для определённых версий EBR схемы повышенного порядка), нормированные компоненты вектора нормали и площадь, 6 коэффициентов разложения вектора реконструкции по естественному базису противопотоковых элементов (только для некоторых схемы повышенного порядка), значения потоков через грань (только для явной схемы для OpenMP распраллеливания) – 5 штук без дифференциальной модели турбулентности и 6 (и более) с моделью. Представить сегмент в виде структуры или класса C++ было бы неэффективно из-за того, что в зависимости от типа численной схемы или параллельного режима существенно меняется состав свойств объекта. Большое число свойств оказались бы не задействованными, что увеличило бы затраты памяти, потери в кэш и снизило бы скорость доступа. Вместо этого данные размещаются в блочном векторе, при этом размерность блока определяется по фактическим параметрам расчёта. Над блочным вектором реализована некоторая надстройка для позиционирования внутри блока. В общем виде доступ к некоторому свойству Value осуществляется по следующей схеме: метод *inline ValueType*

**VALUE(){return ((ValueType*)V + ValueOffst);}*, где V – указатель на начало блока, ValueType – тип данных свойства, и ValueOffst – смещение свойства в блоке, выполняет позиционирование. Макроподстановка *#define Value VALUE()* имитирует доступ к полю структуры. При этом тип ValueType может быть различным для различных свойств, преобразование типа указателя осуществляется методом позиционирования. При таком подходе доступ, например, к номеру левого узла i-й грани выглядит следующим образом: *Face[i].Node[0]* или *Face[i].NodeL* через макросы *#define Node NODE()* и *#define NodeL NODE()[0]* соответственно, где NODE – соответствующий метод класса, производного от блочного вектора.

Основные блочные вектора, определяющие расчётную область: координаты узлов, индексы узлов сеточных элементов – топология сетки, сегменты контрольных объёмов, поля физических переменных. Топология сетки ставит в соответствие сеточным элементам набор составляющих их узлов (вершин) сетки. Обратная топология для каждого узла сетки хранит содержащие его элементы в стандартном виде построчной разреженной матрицы CSR (Compressed Sparse Row). С помощью обратной топологии осуществляется быстрый доступ к соседним ячейкам, к сеточным элементам, содержащим узел, или к сегментам контрольного объёма.

3.7 Параллельные вычисления

Двухуровневое распараллеливание MPI+OpenMP, реализованное в NOISEtte, соответствует современной архитектуре суперкомпьютеров с многоядерными вычислительными модулями (узлами). Суперкомпьютер представляет собой систему с распределённой памятью, так как узлы имеют свое адресное пространство оперативной памяти. При этом, на узлах установлены многоядерные процессоры, и узлы сами по себе являются параллельной системой с общей памятью. В многоуровневой модели на первом уровне используется MPI для объединения узлов в рамках модели с распределённой памятью. На втором уровне внутри узлов применяется OpenMP в рамках модели с общей памятью.

3.7.1 Распараллеливание MPI+OpenMP

MPI распараллеливание NOISEtte выполнено на основе традиционного геометрического параллелизма. Расчётная область разбивается на подобласти MPI процессов. Для обмена данными между соседними подобластями неструктурированной сетки используется неблокирующая пересылка типа точка-точка (MPI_Isend, MPI_Irecv). Более подробно с MPI распараллеливанием можно ознакомиться в [99]. Использование OpenMP в дополнение к MPI позволяет повысить эффективность при расчётах на большом числе процессоров. Число MPI процессов, P_p, уменьшается в P_t раз при том же числе задействованных ядер $P = P_p \times P_t$, где P_t – число нитей. Суммарный объём обмена данными уменьшается примерно в P_t раз пропорционально размеру границ между подобластями, сокращаются потери на латентность за счёт уменьшения числа процессов в групповых обменах, повышается скорость обмена,

поскольку процессам не приходится разделять коммуникационные ресурсы узла. Однако при этом приходится иметь дело с более сложной параллельной моделью и особенностями OpenMP, которые необходимо учитывать. Типичные проблемы при применении OpenMP в случае неструктурированных сеток, подробно рассматривающиеся, например, в [5], в основном связаны с целостностью доступа к общей памяти, состояниями гонки и так далее.

3.7.2 Производительность вычислений и параллельная эффективность

Производительность вычислений измеряется встроенным в код счётчиком операций с плавающей точкой и встроенной системой таймирования. При расчётах по явной схеме “чистая” производительность составляет $12 \sim 15\%$ от пиковой производительности CPU ядра (в тестах на суперкомпьютере К-100 измерялась фактическая производительность вычислений), в случае неявной схемы примерно $10 \sim 12\%$, что является высоким показателем для данного класса газодинамических алгоритмов на неструктурированных сетках, которым характерна, с одной стороны, очень низкая удельная вычислительная стоимость на единицу данных и, с другой стороны, нерегулярный доступ к памяти с неизбежными потерями в кэш в силу неструктурированной топологии. Более низкие показатели для неявной схемы связаны с тем, что существенную часть времени вычислений суммарно потребляет операция матрично-векторного произведения с разреженной матрицей нерегулярной структуры. Данной операции характерна очень низкая удельная вычислительная стоимость (обычно теоретический достижимый предел для такой операции составляет менее 10%).

Для демонстрации работы OpenMP распараллеливания был выполнен тест на суперкомпьютере Ломоносов на разделе с 8-ядерными узлами (процессоры Intel Xeon X5570) для MPI групп размера 32 и 128 процессов. Специально использовалась небольшая сетка, содержащая всего 2 миллиона узлов. Число OpenMP нитей варьируется от 1 до 8, задействуется суммарно от 32 до 1024 ядер. Как видно из результатов на рис. 3.6, эффективность по OpenMP для 8 нитей составила 67% для группы 32 процесса и 63% для группы 128 процессов. Такое отклонение от линейного ускорения связано с тем, что по мере ускорения вычислений относительный вклад накладных расходов MPI увеличивается в несколько раз. Это подтверждается заметной разницей в 4% между группами 32 и 128 процессов. Без MPI эффективность по OpenMP составила бы около 80%, но такой режим не представляет практического интереса. По распределению вычислительной нагрузки получены следующие данные: расчёт узловых градиентов примерно 15% от общего времени на шаг, расчёт конвективной части потоков через грани – примерно 50% и расчёт диссипативной части примерно 11% (остальное время тратится на граничные условия, интегрирование по времени, обработку статистики течения, MPI обмены и так далее).

Ускорение MPI также показано на тесте со сравнительно небольшой сеткой, содержащей 16 миллионов узлов, чтобы вывести код на близкий к предельному режим, когда параллелизм практически исчерпывается. Тест выполнен на суперкомпьютере Ломоносов на разделе с 8-

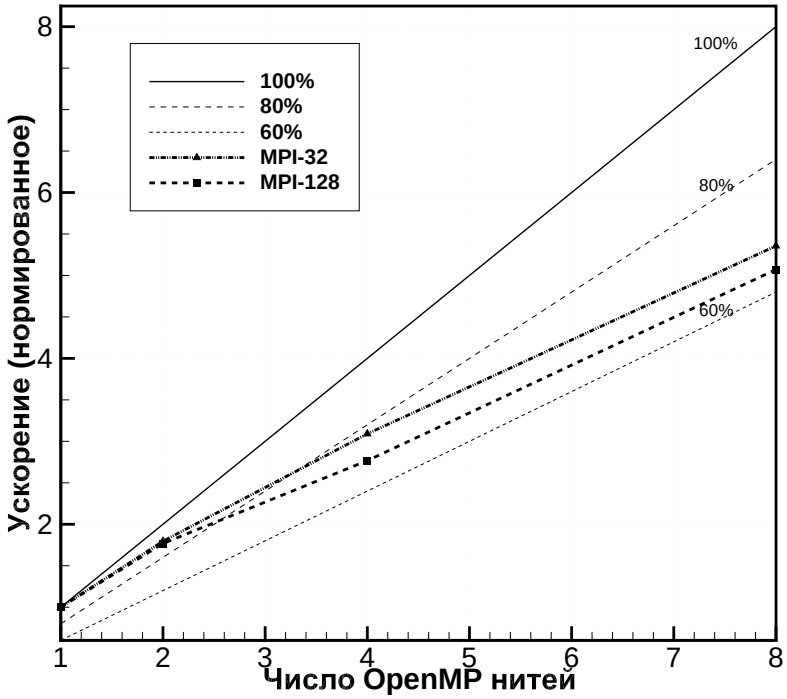


Рисунок 3.6: Ускорение на суперкомпьютере Ломоносов. Слева ускорение с OpenMP для групп 32 из 128 MPI процессов

ядерными узлами, библиотека MPI OpenMPI 1.4. Задействовано до 24000 ядер, что соответствует загрузке всего 670 узлов сетки на ядро, близкой к минимальной (в [100] приводятся тесты NOISEtte до 500 узлов на ядро). Тест соответствует расчёту реальной задачи – обтекание цилиндра дозвуковой струей, выполненному в рамках совместных с ЦАГИ исследований источников шума в турбулентном следе [101]. Результаты полученного ускорения при переходе с 1024 до 24000 ядер показаны на 3.7. Время вычислений на 1 шаг метода Рунге-Кутты составило 0.65 сек. на 1024 ядрах и 0.04 сек. на 24000 ядрах, что соответствует ускорению 16 и параллельной эффективности 65%.

Аналогичные тесты были выполнены на суперкомпьютере МВС-10П. Для оценки распараллеливания с распределённой памятью в качестве тестовой задачи для измерений показателей базового алгоритма взят расчёт вязкого турбулентного течения вокруг клиновидного тела с обратным уступом. В расчёте используется гибридная модель турбулентности DES (задействованы подходы RANS и LES). Для пространственной дискретизации используется сетка из 13 миллионов узлов и 78 миллионов тетраэдров. Результаты измерения эффективности распараллеливания базового алгоритма показаны на Рис. 3.8. Из результатов видно, что показатели параллельной эффективности составляют примерно 90% на 2048 ядрах.

Тестирование ускорения при распараллеливании с общей памятью для одиночного MPI процесса показано на рис. 3.9, сетка 1.3 млн узлов, задача о сверхзвуковом обтекании конуса. Ускорение на 8-ядерном процессоре составило примерно 7, на двух процессорах 12 их 16-ти.

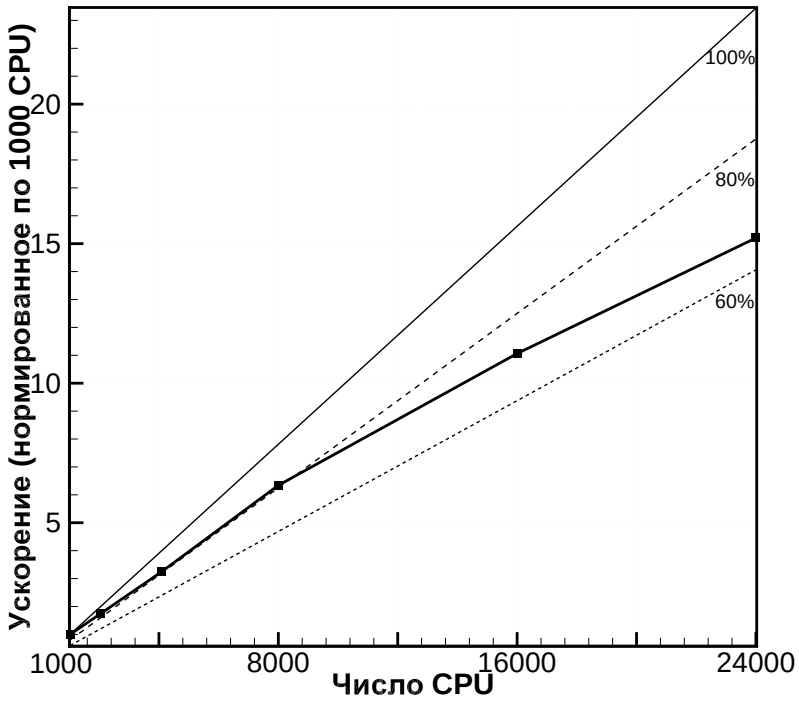


Рисунок 3.7: Ускорение на суперкомпьютере Ломоносов. Ускорение с MPI, 8 OpenMP нитей на процесс

3.8 Приложения

3.8.1 Область применения

Комплекс программ NOISEtte предназначен для расчёта стационарных и нестационарных задач аэродинамики и аэроакустики. С его помощью могут решаться как модельные задачи, направленные на фундаментальные исследования, так и задачи, связанные с моделированием реальных турбулентных течений в сложных геометрических областях. Благодаря реализованным алгоритмам повышенной точности, NOISEtte обеспечивает возможность численного воспроизведения нестационарных режимов течения и окружающих акустических полей, что является отличительной особенностью комплекса. Соответственно, основным типом задач, на которые ориентирован NOISEtte, является расчёт сложных нестационарных турбулентных течений, требующих высокого пространственного разрешения и точных численных методов.

Помимо аэродинамических и аэроакустических приложений, NOISEtte может применяться для других задач, поддающихся математическому описанию при помощи уравнений газовой динамики для идеального сжимаемого газа. Также существует возможность расширения для моделирования динамики реальных газов.

3.8.2 Верификация и валидация

Верификация численных алгоритмов, реализованных в программном комплексе NOISEtte, проводилась на множестве тестовых задач с известными точными или эталонными решениями.

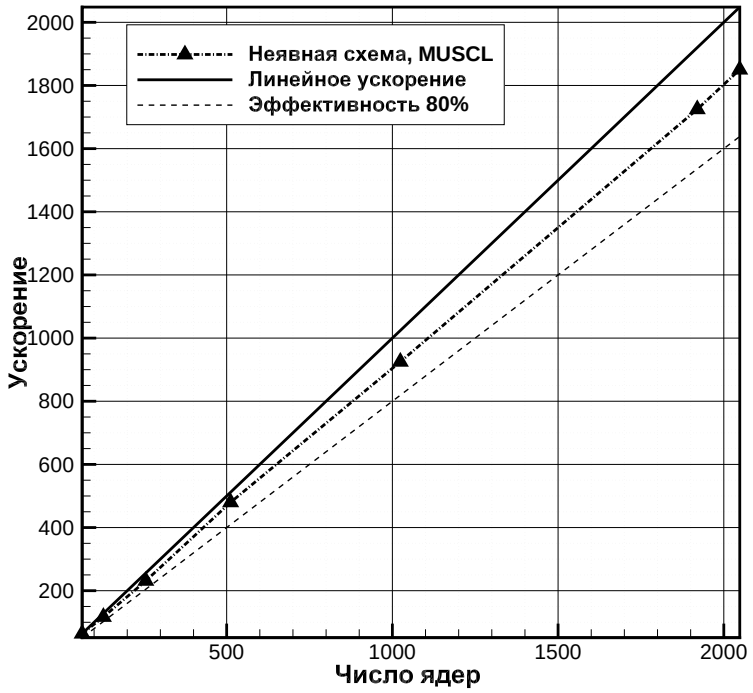


Рисунок 3.8: Ускорение с MPI на суперкомпьютере МВС-10П, сетка 13 млн. узлов, неявная схема, моделирование турбулентности DES

Среди таких задач: распространение начального возмущения в виде Гауссового импульса при наличии постоянного фонового течения [76], расчёт мощности акустического излучения в дальнем поле от монопольного точечного источника в дозвуковом потоке [102], рассеяние акустических волн на цилиндре [103], рассеяние акустических волн на единичном вихре (вихрь Рэнкина) [104], [105], распад произвольного разрыва [106], течения в канале, сверхзвуковое течения вокруг прямого уступа [107], расчёт характеристик сверхзвукового течения газа в плоском канале с клином [108].

Валидация комплекса осуществлялась, в основном, на моделировании турбулентных течений путём сравнения результатов с известными эмпирическими законами и/или имеющимися экспериментальными данными. В частности, корректность результатов, получаемых при помощи комплекса NOISEtte, подтверждена на решениях следующих задач: моделирование изотропной турбулентности [109], ламинарного и турбулентного пограничных слоев на пластине [110], турбулентного течения в канале и турбулентного течения вокруг обратного уступа в канале [111], расчёт стационарных аэродинамических характеристик профилей (NACA0012, NACA23012 и др.), крыла ONERA M6 [112], модели DPW4 [113], и других.

3.8.3 Исследовательские расчёты

Комплекс программ NOISEtte использовался при решении ряда исследовательских задач, ориентированных на реальные приложения в аэродинамике и аэроакустике. Примером одной из таких задач является численное исследование поглощения акустической энергии падающих волн при прохождении звукопоглощающих конструкций резонансного типа, широко применяющихся

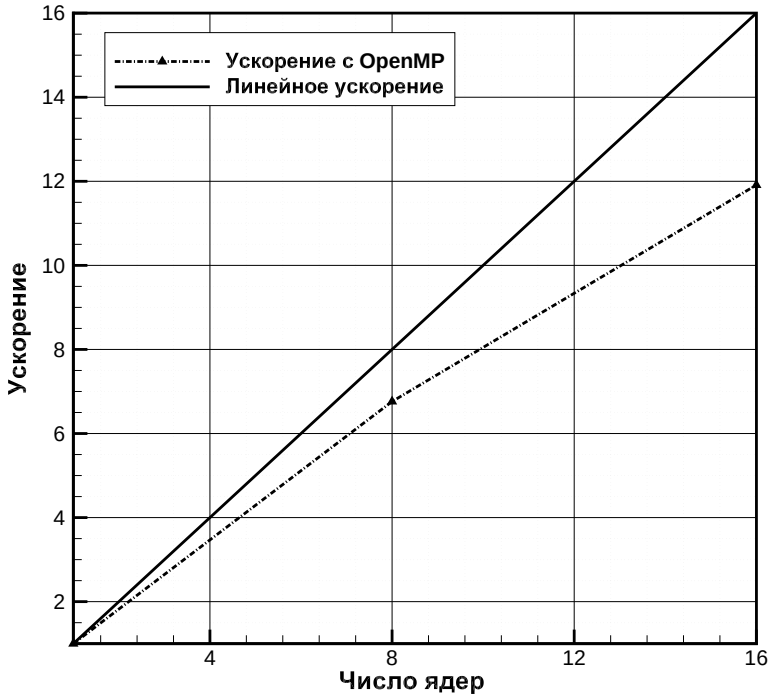


Рисунок 3.9: Ускорение с OpenMP на суперкомпьютере МВС-10П, сетка 1.3 млн. узлов, явная схема

для снижения шума газотурбинных двигателей. В процессе исследования ставился специальный вычислительный эксперимент на микро-уровне, а именно, рассматривался один резонатор реальной конфигурации, “вмонтированный” в стенку волновода [114]. На вход волновода подавалось звуковое излучение в виде плоских волн на фоне входящего дозвукового течения (или без него). На основе измерений мощности акустических возмущений до и после резонатора проводился расчёт характеристик исследуемого резонатора – коэффициента прохождения и акустического импеданса. На рис. 3.10 сверху представлена схема расчёта задачи с перфорированным резонатором, имеющим 11 выходящих в волновод отверстий. Нижний рис. 3.10 показывает поля возмущения плотности в окрестности резонаторных отверстий на разные моменты времени. Из других примеров исследований аэроакустических задач с помощью NOISEtte можно привести исследования акустического течения в горле резонатора [115] и рассеяния акустических волн изолированными вихревыми структурами [116].

NOISEtte применялся в совместном с ЦАГИ численном исследовании структуры распределённого акустического источника, формирующегося в турбулентном следе за цилиндром, и возможностей влияния на него с целью уменьшения шума в дальнем поле. В основе данного исследования лежит идея, предложенная учеными ЦАГИ, о снижении акустической мощности источника в результате отсечения сектора с тыльной стороны цилиндра [101]. Фундаментальное исследование имеет явную прикладную направленность, так как цилиндры при рассматриваемых режимах течения имитируют стойки шасси самолета, шум при обтекании которых на этапе посадки негативно влияет на экологию аэропортов и прилегающих к ним территорий.

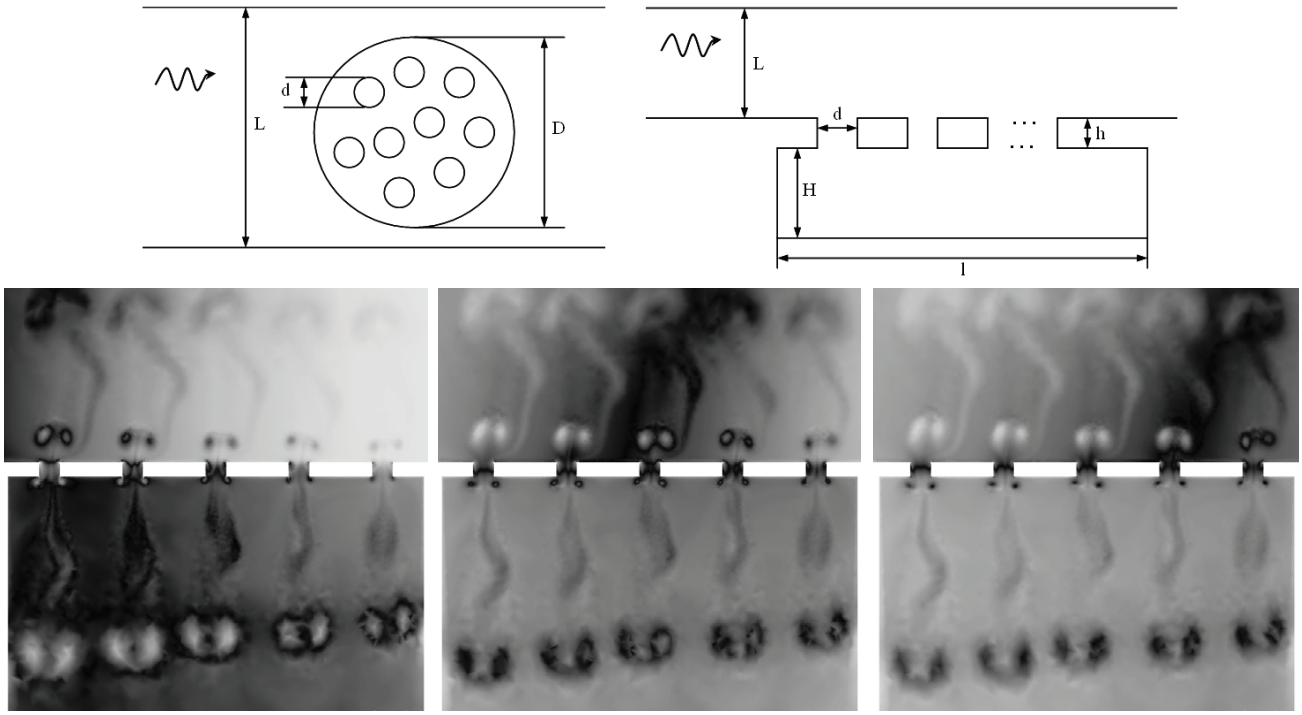


Рисунок 3.10: Сверху – схема расчёта задачи с перфорированным резонатором, снизу – поле плотности в центральном сечении в разные моменты времени при прохождении акустической волны

В численном эксперименте моделировалось поперечное обтекание круглого цилиндра круглой дозвуковой струей. Такая постановка соответствует проводимому в ЦАГИ физическому эксперименту. На рис. 3.11 показан вид огрублённой неструктурированной тетраэдральной сетки, демонстрирующий области сгущения к поверхности цилиндра и краю струи. В расчётах использовались сетки до 16 миллионов узлов и 100 миллионов тетраэдров. На рис. 3.11 справа сверху показано поле модуля скорости в системе “струя–цилиндр”, которое даёт представление о сложном характере турбулентного течения. На рис. 3.12 сверху показано поле плотности при обтекании круглого цилиндра однородным потоком.

NOISEtte использовался для расчётов в рамках работ по проекту 7-й рамочной программы Евросоюза VALIANT, в котором исследовались механизмы возникновения аэродинамического шума при заходе самолета на посадку. Моделировались две конфигурации: обтекание цилиндров, воспроизводящее механизмы генерации шума при обтекании стоек шасси, и турбулентность от зазоров в конструкции планера. На рис. 3.12 снизу показано поле плотности при обтекании тандема квадратных цилиндров. Для постановок имеются экспериментальные данные, полученные в NLR (Нидерланды) и ЦАГИ. По результатам расчётов выполнялось сравнение с экспериментом и другими расчётами. Первая конфигурация состоит их двух квадратных цилиндров, выставленных по одной линии. Расстояние между центрами цилиндров шириной D=0.04м составляет S=0.16м. Скорость потока составляет M = 0.2, Re=182000 (по ширине цилиндров), периодика вдоль оси цилиндров, сетки до 13 млн узлов, 72 млн тетраэдров. На рис. 3.13 в качестве примера показано сравнение с экспериментом для осреднённых полей

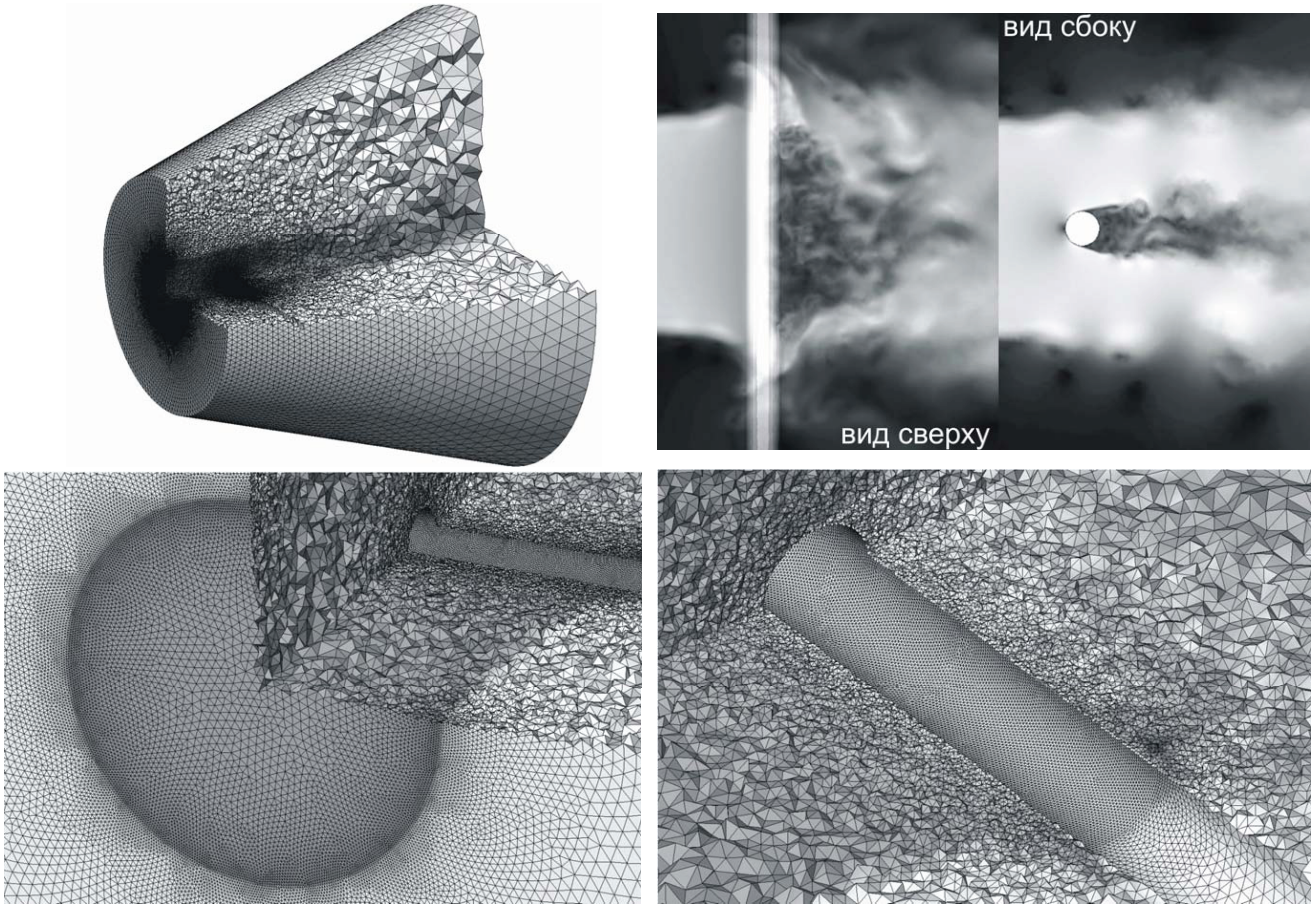


Рисунок 3.11: Пример тетраэдральной сетки для сложной конфигурации – сгущение одновременно к границе круглой струи и к поверхности поперечного цилиндрического препятствия. Справа сверху показана картина течения (поле модуля скорости) при обтекании цилиндра струей

течения, для спектра пульсаций поверхностного давления на препятствиях и для спектра в дальнем поле для одной из множества позиций наблюдателя. Более подробно расчёт представлен в 6-й главе данной работы. Вторая конфигурация – зазор между бесконечными пластинами (рис. 3.14), периодические условия вдоль зазора, отношение ширины зазора и толщины пластин 2:1, M = 0.175, Re=40000 (по ширине зазора). Результаты представлены в [117].

С помощью NOISEtte выполнена серии расчётов, в которых исследуется течение за обратным уступом в различных конфигурациях, включая клиновидное тело с обратным уступом (рис. 3.15), Re=7.2 млн (по длине пластины перед уступом), M=0.913, сетки до 26 млн узлов, 149 млн тетраэдров. Физический эксперимент в соответствующей постановке выполнен в ЦНИИМАШ. Исследуются механизмы генерации повышенных пульсаций давления при трансзвуковых режимах обтекания. Результаты представлены в работе [118].

Выводы по главе

Представленный в главе программный комплекс NOISEtte, основанный на схемах повышенной точности на неструктурированных сетках, представляет собой, с одной

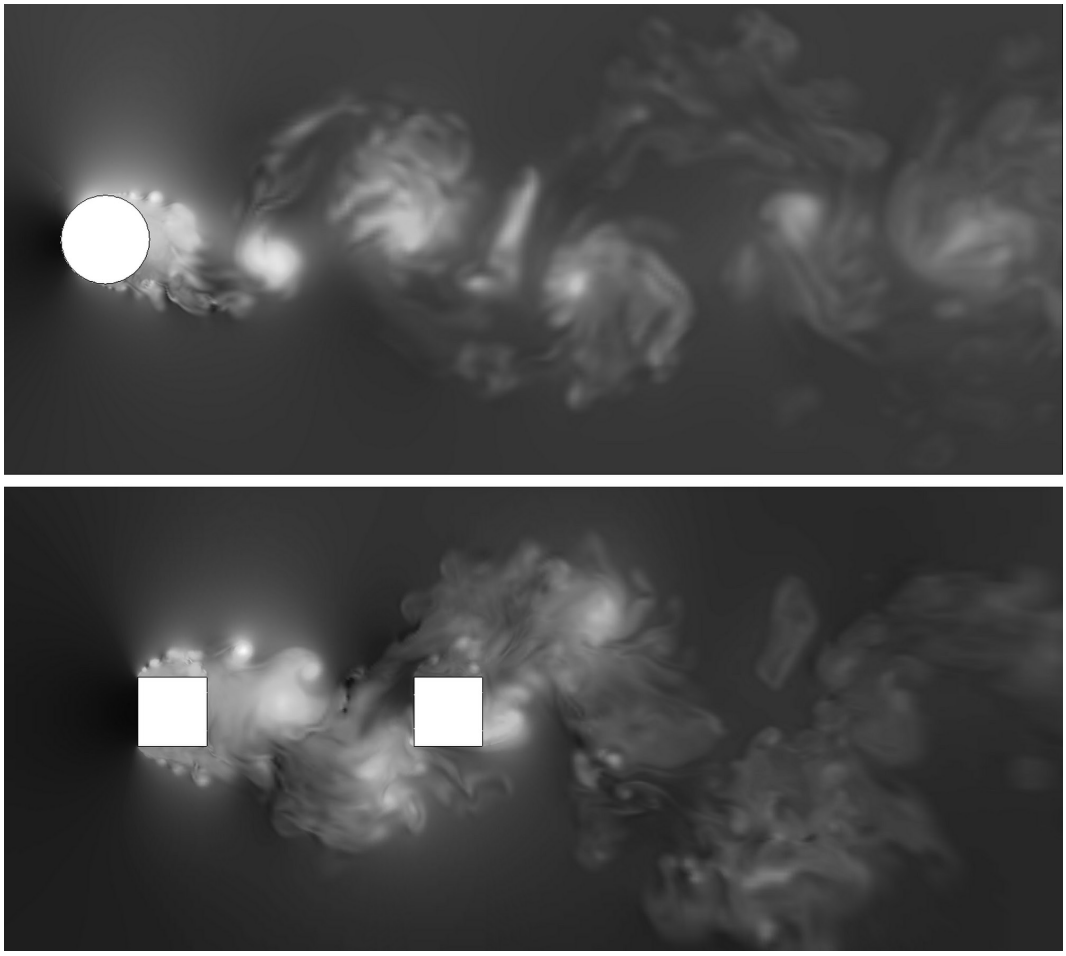


Рисунок 3.12: Поле плотности при обтекании круглого цилиндра (сверху) и системы из двух цилиндров с квадратным сечением (снизу)

стороны, инструмент для решения исследовательских фундаментальных задач, выполнения крупномасштабных суперкомпьютерных расчётов, отработки новых методов и алгоритмов, и, с другой стороны, средство для реальных промышленных приложений. Программный комплекс NOISEtte использовался при проведении совместных исследований с ОАО «ОКБ Сухого», ОАО «Авиадвигатель», ЦНИИМаш, ЦАГИ, ФАЛТ МФТИ и ОАО «Камов». Он также был задействован в европейском проекте FP7 VALIANT, где совместно с ЦАГИ, ONERA, NLR, TUB и другими партнерами из 6 европейских стран проводились исследования потенциала современных численных методов и подходов для моделирования аэродинамического шума. Многоуровневое распараллеливание MPI+OpenMP позволяет задействовать десятки тысяч процессорных ядер суперкомпьютеров, параллельные средства инфраструктуры позволяют выполнять расчёты на неструктурированных сетках с числом элементов более миллиарда.

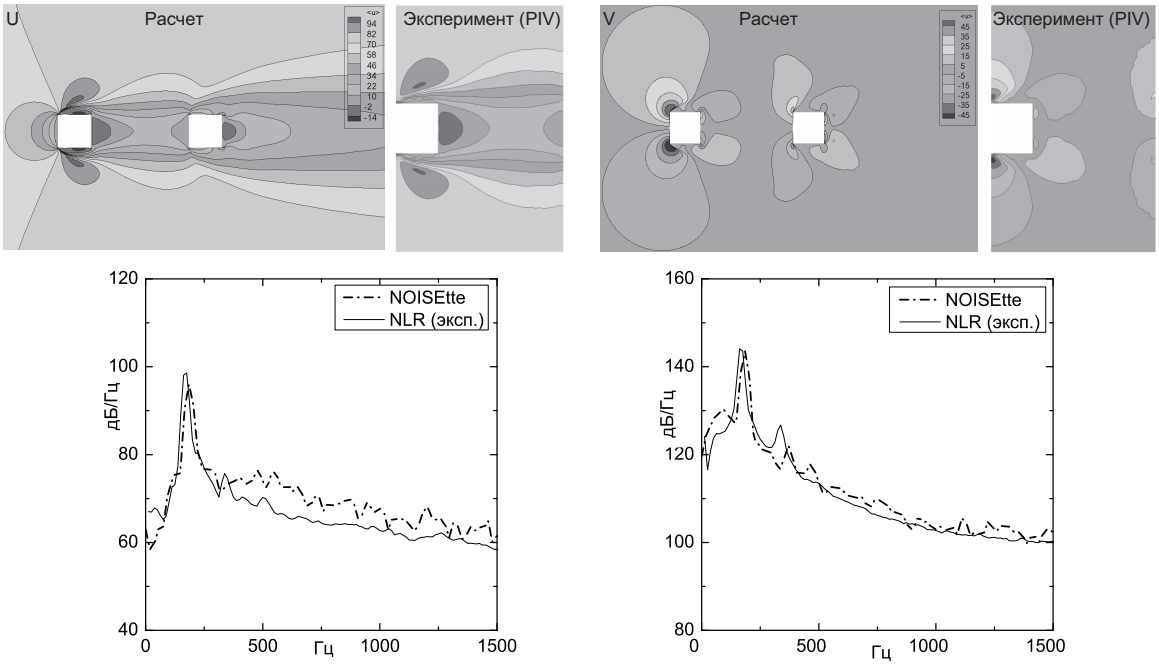


Рисунок 3.13: Сравнение с экспериментом. Сверху – осреднённые поля продольной (слева) и вертикальной (справа) компоненты скорости в центральном сечении. Снизу – акустический спектр в дальнем поле (слева) и спектр пульсаций поверхностного давления (справа)

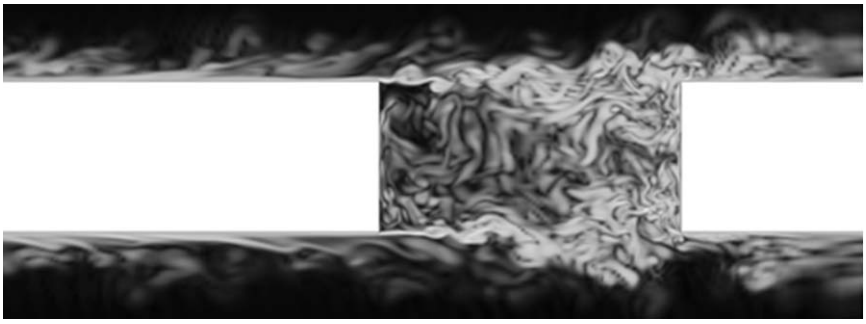


Рисунок 3.14: Течение в зазоре между бесконечными пластинами [117]: показано поле завихренности в продольном сечении

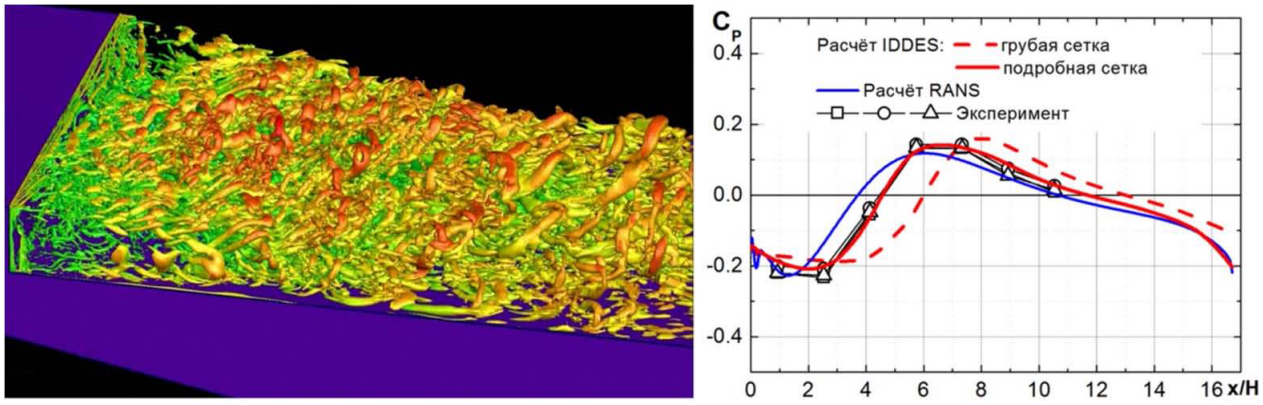


Рисунок 3.15: Обтекание клиновидного тела с обратным уступом [118]: слева – течение за уступом (Q-критерий), справа – сравнение с экспериментом давления на поверхности по продольной центральной линии за уступом

Глава 4

Параллельный решатель уравнения Пуассона для моделирования несжимаемых турбулентных течений на десятках тысяч процессоров и на гибридных системах

Вводные замечания

Эта глава посвящена разработке по предложенной в первой главе параллельной технологии эффективного параллельно алгоритма для прямого численного моделирования несжимаемых турбулентных течений на крупных суперкомпьютерах различной архитектуры, включая гибридные системы с массивно-параллельными ускорителями. Данная глава основана на материале, представленном в статьях [32, 119].

В рамках проекционного метода дробного шага на каждом шаге по времени необходимо решать уравнение Пуассона, для того чтобы спроецировать поле скоростей на бездивергентное пространство. В связи с нелокальной природой решения, эта эллиптическая система является наиболее сложной для распараллеливания и вычислительно ёмкой частью алгоритма.

Наличие однородного направления с периодическими граничными условиями существенно сказывается на вычислительной стоимости расчёта, позволяя диагонализировать блоки матрицы, соответствующие связям шаблоном численной схемы в этом пространственном направлении.

Наиболее быстрые и дешевые методы на основе быстрого преобразования Фурье (БПФ) или полиномов Чебышева применяются в так называемых спектральных кодах для задач, имеющих периодические условия по всем трём осям. Классическими задачами данного типа является моделирование гомогенной изотропной турбулентности. В случае трёх периодических направлений матрица полностью диагонализируется быстрыми методами. В связи с этим рекордные DNS расчёты [120] (68 миллиардов узлов, 2004 г.) и новый расчёт [121] (1.8 триллиона узлов, 2015 г.) принадлежат к этому особому классу упрощённых задач. Сами

спектральные коды также как правило являются достаточно простыми. Автор непосредственно участвовал в разработке спектрального кода объёмом менее тысячи строк (за счёт использования библиотеки БПФ FFTW3), применявшегося в [70], с помощью которого можно выполнять расчёты трёхмерных задач на сетках из миллиардов узлов.

Следующим по сложности классом задач являются задачи с двумя периодическими направлениями. Диагонализация по двум направлениям приводит к небольшим “одномерным” системам, решение которых также имеет небольшую стоимость. Классической задачей этого типа является течение между двумя параллельными бесконечными пластинами. Рекордные расчёты задач с двумя периодическими направлениями – [122] (20 миллиардов узлов, 2007 г.) и [123] (242 миллиарда степеней свободы, 2013 г.).

Алгоритм, представленный в данной главе, ориентирован на более сложный класс задач – с одним периодическим направлением. Сложность решения для данного класса задач уже сопоставима с “полностью трёхмерными” задачами, имеющими твёрдые поверхности во всех трёх пространственных направлениях. Однако, выигрыш от диагонализации во многих случаях оказывается достаточно существенным, чтобы использовать более узко специализированные решатели для задач с периодикой вместо универсальных решателей СЛАУ, применимых к полностью трёхмерным задачам.

Для трёхмерных задач без периодики наиболее распространёнными и перспективными с точки зрения эволюции вычислительной архитектуры представляются методы на основе подпространств Крылова и многосеточные методы, примеры которых могут быть найдены в [9] и [10, 11], соответственно. Под перспективностью понимается то, что решатели могут быть реализованы на базе операций, совместимых с потоковой обработкой и SIMD параллелизмом, то есть быть применимы на массивно-параллельных ускорителях различной архитектуры.

Решатель уравнения Пуассона, представленный в данной главе, предназначен для задач с одним однородным периодическим направлением. В тестовых сравнения с универсальными решателями на таком типе задач наблюдался выигрыш около двух раз. Сравнение выполнялось с многосеточным решателем [124] на основе библиотеки Hypre и с данными по производительности многосеточного метода, представленными в [125].

Специальное расширение, также представленное в этой главе, позволяет применять метод и для трёхмерных задач без периодики, хотя в этом случае метод уже может заметно проигрывать по производительности вышеуказанным универсальным решателям.

Метод решения уравнения Пуассона, названный для краткости KSFD (Krylov-Schur-Fourier decomposition) [126], сочетает три метода: 1) диагонализация с помощью БПФ, 2) прямой метод на основе дополнений Шура [127], 3) итерационный предобусловленный метод сопряжённых градиентов. Первый компонент разделяет исходную “трёхмерную” систему уравнений на набор независимых “двухмерных” систем, иначе говоря, плоскостей; второй компонент используется для решения нескольких таких плоскостей, соответствующих низким частотам в пространстве Фурье, наиболее проблематичных для итерационного метода; третий компонент – для решения остальных двухмерных систем, для которых итерационный метод имеет хорошую сходимость.

Предыдущая версия решателя [126], представленная в кандидатской диссертации [81], была предназначена для одноядерных процессоров и использовала только параллельную модель с распределённой памятью посредством MPI. Максимальное число процессоров, на котором метод был эффективно применим в реальных DNS расчётах, было ограничено диапазоном $2000 \sim 6000$ ядер в зависимости от размера сетки и порядка численной схемы. Ограничения связаны с двумя факторами: 1) ограничения прямого метода на основе дополнений Шура, связанные с обращением интерфейсной матрицы; 2) распараллеливание по периодическому направлению, использующее групповой обмен данными в одномерных подгруппах для восстановления подвекторов по периодическому направлению. Более подробно ограничения масштабируемости KSFD решателя описаны в [126].

Решатель, представленный в данной работе, имеет тот же принцип работы и основан на тех же численных методах, но имеет многоуровневое распараллеливание для расчётов на десятках тысяч процессоров и гибридных системах с массивно-параллельными ускорителями различной архитектуры.

Появление архитектур с существенно-многоядерными вычислительными модулями, объединяющими множество процессорных ядер в одном адресном пространстве оперативной памяти, мотивировало переход на двухуровневое MPI+OpenMP распаралленивание, в котором на втором уровне используется параллельная модель с общей памятью.

Двухуровневое распараллеливание позволило существенно расширить границы эффективного диапазона масштабируемости. Решатель был успешно применен на числе ядер до 12800 на сетках до миллиарда узлов. Представленные оценки на основе полученных результатов позволяют утверждать, что решатель может успешно применяться в расчётах с использованием более ста тысяч ядер.

Появление гибридных архитектур на основе массивно-параллельных ускорителей потребовало адаптации алгоритма к парадигме потоковый обработки. Третий уровень распараллеливания посредством открытого вычислительного стандарта OpenCL, который представлен в следующей главе, предназначен для использования ускорителей различных архитектур, включая GPU AMD и NVIDIA, ускорители Intel Xeon Phi и т.д.

4.1 Математическая модель и численный метод

Рассматривается моделирование несжимаемого турбулентного течения ньютоновской жидкости в расчётной области, представляющей собой параллелепипед высотой L_z, шириной L_y и длиной L_x. Периодические граничные условия заданы в направлении оси x.

Поле скоростей, $\boldsymbol{u}$, и температуры, θ, описывается безразмерными несжимаемыми уравнениями Навье-Стокса (НС) и уравнением переноса температуры:

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0, \tag{4.1}$$

$$\partial_t \boldsymbol{u} + \mathcal{C}\left(\boldsymbol{u}, \boldsymbol{u}\right) = \mathcal{D}\boldsymbol{u} - \boldsymbol{\nabla} p + \boldsymbol{f}, \tag{4.2}$$

$$\partial_t \theta + \mathcal{C}\left(\boldsymbol{u}, \theta\right) = Pr^{-1}\mathcal{D}\theta, \tag{4.3}$$

где конвективный член определен как $\mathcal{C}\left(\boldsymbol{u}, \phi\right) = \left(\boldsymbol{u} \cdot \nabla\right)\phi$. В случае естественной конвекции, когда учитывается изменение плотности жидкости с температурой, диффузионный член имеет вид $\mathcal{D}\boldsymbol{u} = PrRa^{-1/2}\boldsymbol{\nabla}^2\boldsymbol{u}$, а вектор массовых сил записан в виде $\boldsymbol{f} = (0{,}0{,}Pr\theta)$ (аппроксимация Буссинеска), где Ra – число Рэлея (по высоте расчётной области, L_z). В случае вынужденной конвекции изменением плотности пренебрегается, температура становится пассивным скаляром, вектор массовых сил $\boldsymbol{f} = 0$, а диффузионный член имеет вид $\mathcal{D}\boldsymbol{u} = Re^{-1}\boldsymbol{\nabla}^2\boldsymbol{u}$, где Re – безразмерное число Рейнольдса.

4.1.1 Дискретизация по времени

Для дискретизации по времени конвективных и диффузионных членов уравнений (4.2-4.3) используется явная схема второго порядка. Уравнение (4.2) может быть записано в виде

$$\partial_t \boldsymbol{u} = \mathbf{R}\left(\boldsymbol{u}\right) - \boldsymbol{\nabla} p, \tag{4.4}$$

где $\mathbf{R}(\boldsymbol{u})$ представляет собой суммарный вклад конвективной части, диффузионной части и вектора массовых сил в уравнение момента и имеет вид

$$\mathbf{R}(\boldsymbol{u}) = \mathcal{D}\boldsymbol{u} - \mathcal{C}\left(\boldsymbol{u}, \boldsymbol{u}\right) + \boldsymbol{f}. \tag{4.5}$$

$\mathbf{R}(\boldsymbol{u})$ дискретизован по времени с использованием однопараметрической полностью явной схемы второго порядка Адамса–Башфора. Для дискретизации градиента давления применяется явная обратная схема Эйлера первого порядка. Ограничение несжимаемости не может быть представлено в явном виде.

Дискретизованные по времени уравнения (4.1-4.3) имеют вид

$$\boldsymbol{\nabla} \cdot \boldsymbol{u}^{n+1} = 0, \tag{4.6}$$

$$\frac{\left(\kappa + 0.5\right)\boldsymbol{u}^{n+1} - 2\kappa\boldsymbol{u}^{n} + \left(\kappa - 0.5\right)\boldsymbol{u}^{n-1}}{\Delta t} = \mathbf{R}\left(\left(1+\kappa\right)\boldsymbol{u}^{n} - \kappa\boldsymbol{u}^{n-1}\right) - \boldsymbol{\nabla} p^{n+1}, \tag{4.7}$$

$$\frac{\left(\kappa + 0.5\right)\theta^{n+1} - 2\kappa\theta^{n} + \left(\kappa - 0.5\right)\theta^{n-1}}{\Delta t} = Pr^{-1}\mathcal{D}\left(\left(1+\kappa\right)\theta^{n} - \kappa\theta^{n-1}\right) - \tag{4.8}$$

$$\mathcal{C}\left(\left(1+\kappa\right)\boldsymbol{u}^{n} - \kappa\boldsymbol{u}^{n-1}, \left(\left(1+\kappa\right)\theta^{n} - \kappa\theta^{n-1}\right)\right),$$

где параметр κ вычисляется на каждом шаге по времени, чтобы адаптировать линейную область устойчивости схемы интегрирования по времени к моментальной картине течения и обеспечить использование максимально возможного шага по времени Δt. Более подробно метод интегрирования по времени описан в [128]. Уравнение для температуры дискретизовано аналогичным образом:

Классический проекционный метод дробного шага [129, 130] используется для связи между скоростью и давлением. Для решения нестационарных уравнений Навье-Стокса сначала выполняется продвижение по времени поля скорости $\boldsymbol{u}$ без учёта уравнения (4.6) – в явном виде вычисляется предиктор скорости $\boldsymbol{u}^p$, затем выполняется восстановление соленоидальности для получения результирующего поля скорости $\boldsymbol{u}^{n+1}$, для которого выполнено условие $\boldsymbol{\nabla} \cdot \boldsymbol{u}^{n+1} = 0$. Такая проекция получается на основе теоремы Гельмгольца–Ходжа о разложении вектора [131], из которой следует, что скорость $\boldsymbol{u}^p$ может быть единственным образом разложена на соленоидальный вектор $\boldsymbol{u}^{n+1}$, и безвихревой вектор, выраженный как градиент скалярного поля, $\boldsymbol{\nabla}\tilde{p}$:

$$\boldsymbol{u}^p = \boldsymbol{u}^{n+1} + \boldsymbol{\nabla}\tilde{p}, \tag{4.9}$$

где предиктор скорости $\boldsymbol{u}^p$ имеет вид

$$\boldsymbol{u}^p = \frac{2\kappa\boldsymbol{u}^n - (\kappa - 1/2)\,\boldsymbol{u}^{n-1}}{\kappa + 1/2} + \frac{\Delta t}{\kappa + 1/2}\mathbf{R}\left((1+\kappa)\,\boldsymbol{u}^n - \kappa\boldsymbol{u}^{n-1}\right), \tag{4.10}$$

а псевдо-давление $\tilde{p} = \Delta t/\,(\kappa + 1/2)\,p^{n+1}$.

Применив оператор дивергенции к (4.9) получим уравнение Пуассона для $\tilde{p}$

$$\boldsymbol{\nabla}^2\tilde{p} = \boldsymbol{\nabla}\cdot\boldsymbol{u}^p\,. \tag{4.11}$$

Что касается граничных условий в уравнении для давления, то на дискретном уровне при дискретизации на разнесённой сетке с заданными граничными условиями для скоростей условие несжимаемости выполняется естественным образом и никакие специфические граничные условия для давления не требуются, как было показано в [132].

Когда решение уравнения (4.11) получено, выполняется коррекция поля скорости

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^p - \boldsymbol{\nabla}\tilde{p}\,. \tag{4.12}$$

4.1.2 Дискретизация по пространству

Дискретизация по пространству уравнений Навье-Стокса (4.2) выполняется на декартовой разнесённой сетке с помощью численной схемы, сохраняющей симметрию [133]. Конечно-

объёмная пространственная дискретизация уравнений (4.1-4.3) в операторной форме имеет вид:

$$\begin{aligned} \mathsf{M}\boldsymbol{u}_h &= \mathbf{0}_h, & (4.13)\\ \Omega\frac{d\boldsymbol{u}_h}{dt} &= -\mathsf{C}\left(\boldsymbol{u}_h\right)\boldsymbol{u}_h + \mathsf{D}\boldsymbol{u}_h - \mathsf{M}^t\boldsymbol{p}_h + \boldsymbol{f}_h, & (4.14)\\ \Omega\frac{d\theta_h}{dt} &= -\mathsf{C}\left(\boldsymbol{u}_h\right)\theta_h + Pr^{-1}\mathsf{D}\theta_h. & (4.15) \end{aligned}$$

Диффузионная матрица D, симметричная и положительная полуопределённая, представляет собой интеграл диффузионного потока через грани контрольных объёмов. Диагональная матрица Ω описывает размеры контрольных объёмов. Матрица M представляет дискретный оператор дивергенции. Конвективный поток дискретизован как предложено в [133]. Результирующая конвективная матрица, $\mathsf{C}\left(\boldsymbol{u}_h\right)$, является косо-симметричной, т. е. $\mathsf{C}\left(\boldsymbol{u}_h\right) = -\mathsf{C}^t\left(\boldsymbol{u}_h\right)$. Для поля температуры θ_h заданы в центре контрольного объёма.

Дискретное уравнение Пуассона для $\boldsymbol{p}_h^{n+1}$, которое необходимо решать на каждом шаге по времени:

$$\mathsf{L}\boldsymbol{p}_h^{n+1} = \mathsf{M}\boldsymbol{u}_h^p, \tag{4.16}$$

где дискретный оператор Лапласа $\mathsf{L} = -\mathsf{M}\Omega^{-1}\mathsf{M}^t$ представлен симметричной отрицательной полу-определённой матрицей.

Этапы шага интегрирования по времени представлены в Алгоритме 1.

Алгоритм 1 Алгоритм шага интегрирования по времени

1. Вычислить $\mathbf{R}\left(\left(1+\kappa\right)\boldsymbol{u}_h^n - \kappa\boldsymbol{u}_h^{n-1}\right)$.
2. Получить в явном виде предиктор скорости $\boldsymbol{u}_h^p$ из уравнения (4.10).
3. Вычислить в явном виде поле температуры.
4. Вычислить дивергенцию для предиктора скорости $\boldsymbol{\nabla}\cdot\boldsymbol{u}_h^p$
5. Решить дискретное уравнение Пуассона (4.16).
6. Получить поле скорости на новом шаге по времени из уравнения (4.12).

4.2 Метод решения уравнения Пуассона

Исходная СЛАУ, соответствующая дискретному уравнению Пуассона (4.16) имеет вид:

$$\mathbf{A}^{3D}\mathbf{x}^{3D} = \mathbf{b}^{3D} \tag{4.17}$$

где $\mathbf{A}^{3D} \in \mathbb{R}^{N\times N}$ и $N = N_x \times N_y \times N_z$ – общее число узлов сетки. Для удобства знак системы изменен, чтобы матрица $\mathbf{A}^{3D}$ была положительной полуопределённой. Верхний индекс $3D$ обозначает, что неизвестные связаны шаблоном численной схемы с соседними неизвестными во всех трёх пространственных направлениях. Упорядочив неизвестные таким образом, что внутренней нумерацией была нумерация по периодической оси (т. е. соседям по этой оси соответствуют соседние элементы в векторе неизвестных), вектора $\mathbf{x}^{3D} \in \mathbb{R}^N$ и $\mathbf{b}^{3D} \in \mathbb{R}^N$ могут быть разделены на $N_{yz} = N_y \times N_z$ подвекторов, каждый из которых по N_x компонент: $\mathbf{x}^{3D} \equiv \left(\mathbf{x}_{1,1}, \mathbf{x}_{2,1}, \cdots, \mathbf{x}_{j,k}, \cdots, \mathbf{x}_{N_yN_z}\right)^t$ где $\mathbf{x}_{j,k} = \left(\mathbf{x}_{1,j,k}, \mathbf{x}_{2,j,k}, \cdots, \mathbf{x}_{N_x,j,k}\right)^t \in \mathbb{R}^{N_x}$.

С таким разделением и построчным лексикографическим упорядочиванием узлов сетки матрица системы $\mathbf{A}^{3D}$ представляется в блочном ленточном виде.

Используя тензорное произведение, матрица может быть записана в виде

$$\mathbf{A}^{3D} = \left(\mathbf{A}^{2D} \otimes \Omega_x\right) + \left(\Omega^{2D} \otimes \mathbf{A}_x\right) \,, \tag{4.18}$$

где матрица $\Omega^{2D} \in \mathbb{R}^{N_{yz}\times N_{yz}}$ – это диагональная матрица, представляющая площади проекции контрольных объёмов на двухмерной сетке (трёхмерная сетка с одним однородным периодическим направлением строится путём вытягивания по оси x двухмерной сетки с постоянным шагом), и матрица $\mathbf{A}^{2D} \in \mathbb{R}^{N_{yz}\times N_{yz}}$ представляет систему, возникающую при дискретизации оператора Лапласа на этой двухмерной сетке. Матрица $\Omega_x = \Delta x \mathbf{I}_{N_x}$, где $\mathbf{I}_{N_x} \in \mathbb{R}^{N_x\times N_x}$ – это единичная матрица, Δx – это шаг сетки по периодическому направлению, $\mathbf{A}_x$ – это симметричная циклическая матрица [134], имеющая вид

$$\mathbf{A}_x \equiv \begin{pmatrix} a^p & a^{e_1} & a^{e_2} & \dots & a^{e_1} \\ a^{e_1} & a^p & a^{e_1} & \dots & a^{e_2} \\ a^{e_2} & a^{e_1} & a^p & \dots & a^{e_3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a^{e_1} & a^{e_2} & a^{e_3} & \dots & a^p \end{pmatrix} \in \mathbb{R}^{N_x\times N_x} \,, \tag{4.19}$$

где число ненулевых коэффициентов $a^{e_r}_{j,k}$ зависит от размера шаблона численной схемы. Дальнейшие подробности о дискретизации уравнения Пуассона могут быть найдены в [127,135].

Решатель Пуассона подробно описан в предыдущих работах [81, 126]. Здесь представлено краткое описание основных принципов работы. Используя те же обозначения, сначала определим $\mathbf{Q}_\mathbb{R} \in \mathbb{R}^{N_x\times N_x}$ как матрицу обратного преобразования Фурье, т. е. $\mathbf{x}_{j,k} = \mathbf{Q}_\mathbb{R}\hat{\mathbf{x}}_{j,k}$. Затем, применяя смену базиса $(\mathbf{I}_{N_{yz}} \otimes \mathbf{Q}_\mathbb{R})$, исходная система (4.17) разделяется на набор из N_x независимых между собой систем,

$$\hat{\mathbf{A}}^{2D}_i \hat{\mathbf{x}}^{2D}_i = \hat{\mathbf{b}}^{2D}_i, \qquad i = 1, \cdots, N_x \,. \tag{4.20}$$

где $\hat{\mathbf{x}}_{j,k} = \mathbf{Q}^{-1}_\mathbb{R}\mathbf{x}_{j,k}$, $\hat{\mathbf{b}}_{j,k} = \mathbf{Q}^{-1}_\mathbb{R}\mathbf{b}_{j,k}$ и $\hat{\mathbf{A}}^{2D}_i = (\Delta x)\mathbf{A}^{2D} + \lambda_i\Omega^{2D}$. λ_i – собственные значения $\mathbf{A}_x$ имеющие вид $\lambda_i = a^p + 2\sum_{r=1}^{N_x/2-1} a^{e_r} cos\left(r\alpha_i\right)$, где $\alpha_i = \left[2\pi\left(i-1\right)\right]/N_x$. Поскольку $\lambda_i = \lambda_{N_x-i+2}$,

матрицы по парам совпадают, $\hat{\mathbf{A}}_i^{2D} = \hat{\mathbf{A}}_{N_x-i+2}^{2D}$, и набор матриц систем (4.20) сокращается до $N_x/2+1$ различных матриц. Матрицы $\hat{\mathbf{A}}_1^{2D}$, соответствующая нулевой частоте, и $\hat{\mathbf{A}}_{N_x/2+1}^{2D}$, соответствующая самой высокой частоте, не имеют пары.

$$\hat{\mathbf{A}}_1^{2D}\hat{\mathbf{x}}_1^{2D} = \hat{\mathbf{b}}_1^{2D} \tag{4.21}$$

$$\hat{\mathbf{A}}_i^{2D}(\hat{\mathbf{x}}_i|\hat{\mathbf{x}}_{N_x-i+2}) = (\hat{\mathbf{b}}_i|\hat{\mathbf{b}}_{N_x-i+2}) \qquad i=2,\ldots,N_x/2 \tag{4.22}$$

$$\hat{\mathbf{A}}_{N_x/2+1}^{2D}\hat{\mathbf{x}}_{N_x/2+1}^{2D} = \hat{\mathbf{b}}_{N_x/2+1}^{2D} \tag{4.23}$$

где матрицы $\hat{\mathbf{A}}_i^{2D}$ упорядочены по убыванию числа обусловленности, т. е. $\kappa(\hat{\mathbf{A}}_i^{2D}) > \kappa(\hat{\mathbf{A}}_{i+1}^{2D})$. В [126], было показано, что для упрощённого случая с постоянным шагом сетки и дискретизацией по пространству второго порядка точности, спектральное число обусловленности, ограничено сверху

$$\kappa\left(\hat{\mathbf{A}}_i^{2D}\right) \leq 1 + \frac{2}{sin^2\left(\xi\pi/2\right)}, \tag{4.24}$$

где $\xi\left(i,N_x\right) \equiv 2(i-1)/N_x$ – это относительный номер плоскости. Это соотношение даёт примерное представление о там, насколько хорошо обусловлены системы.

Поскольку матрицы $\hat{\mathbf{A}}_i^{2D}$ по построению симметричные и положительно определённые, предобусловленный метод сопряжённых градиентов PCG (preconditioned conjugate gradient) [77] хорошо подходит для решения этих систем.

Тем не менее, когда значение ξ близко к 0, системы становятся плохо обусловленными, и, следовательно, PCG алгоритм нужно заменить чем-то более эффективным.

Как было предложено в предыдущей работе [126], набор систем разделятся на два поднабора разделителем D. Системы из первый поднабора, т. е. $1 \leq i \leq D$, решаются с помощью прямого метода DSD (Direct Schur decomposition) [127] на основе дополнений Шура, а системы из второго поднабора, т. е. $D < i \leq N_x/2+1$, решаются итерационным методом PCG. Алгоритм решателя приведен ниже.

Алгоритм 2 Алгоритм KSFD решателя

1. Преобразовать вектор правой части, $\hat{\mathbf{b}}_{j,k} = \mathbf{Q}_{\mathbb{R}}^{-1}\mathbf{b}_{j,k}$.
2. Решить первые D систем $\hat{\mathbf{A}}_i^{2D}(\hat{\mathbf{x}}_i|\hat{\mathbf{x}}_{N_x-i+2}) = (\hat{\mathbf{b}}_i|\hat{\mathbf{b}}_{N_x-i+2})$, где $i \in \{1,\ldots,D\}$, используя параллельный прямой DSD решатель [127].
3. Решить $N_x/2+1-D$ систем $\hat{\mathbf{A}}_i^{2D}(\hat{\mathbf{x}}_i|\hat{\mathbf{x}}_{N_x-i+2}) = (\hat{\mathbf{b}}_i|\hat{\mathbf{b}}_{N_x-i+2})$, где $i \in \{D+1,\ldots,N_x/2+1\}$, используя итерационный PCG метод.
4. Восстановить вектор решения $\mathbf{x}_{j,k} = \mathbf{Q}_{\mathbb{R}}\hat{\mathbf{x}}_{j,k}$.

И прямое и обратное преобразование Фуре на этапах 1 и 4 алгоритма выполняются с помощью стандартного быстрого преобразования Фурье (см. [136, 137]). Соответственно,

стоимость этих операций $\mathcal{O}\left(N_x \left(log_2 N_x\right) N_y N_z\right)$ незначительна по сравнению со стоимостью решения N_x СЛАУ.

4.2.1 Ограничения MPI распараллеливания

Распараллеливание решателя основано на принципе геометрического параллелизма. Расчётная область разделяется на P подобластей, по одной на каждый из параллельных процессов. Разбиение выполняется путём разделения области по периодическому направлению на P_x частей и путём разделения плоскостей, ортогональных периодической оси, на P_{yz} частей. Общее число подобластей $P = P_x \times P_{yz}$. Распараллеливание алгоритма разделено на две части: 1) распараллеливание этапов 1 и 4, на которых происходит смена базиса из физического пространства в спектральное и обратно; 2) распараллеливание этапов 2 и 3, на которых выполняется решение независимых систем методами DSD и PCG.

Рассмотрим распараллеливание по периодической оси. Шаги 1 и 4 представляют собой ни что иное как N_{yz} взаимно независимых преобразований Фурье для подвекторов размера N_x. Распараллеливание с распределённой памятью применительно к БПФ даже не рассматривалось, тем более для таких коротких подвекторов. Поэтому БПФ было размножено внутри P_x-групп. Это требует дополнительного группового обмена данными чтобы собрать целиком подвектора вдоль периодической оси внутри P_x-групп. Затем каждый процесс выполняет свой набор БПФ, при этом сами БПФ выполняются последовательно. К счастью, стоимость дублирующихся вычислений БПФ достаточно мала, чтобы не влиять существенным образом на эффективность распараллеливания. Однако, стоимость дополнительного группового обмена растет линейно с ростом числа P_x. В предыдущей работе [126] было показано, что приемлемая параллельная эффективность сохраняется до P_x=8.

С другой стороны, число P_{yz} также ограничено в силу ограничений масштабируемости DSD решателя, который используется на этапе 2 алгоритма. Основную проблему представляет решение интерфейсной системы, которая растет как с числом процессов, так и с размером сетки. Факторов, ограничивающих распараллеливание два: 1) размер группового обмена для редукции вклада интерфейсной части растёт как $\mathcal{O}(\log_2(P)\sqrt{IN_{yz}P})$, 2) затраты оперативной памяти растут как, $\mathcal{O}(I^2 N_{yz})$, где I – ширина интерфейса, пропорциональная порядку схемы. Кроме того, на этапе препроцессора происходит обращение интерфейсной матрицы, для чего используется параллельный метод на основе блочного исключения Гаусса. Этот этап также становится ограничивающим фактором. Более подробную информацию можно найти в [81,126]. По опыту применения решателя в CFD приложениях, приемлемая эффективность использования DSD метода сохраняется до P_{yz} примерно $200 \sim 300$ для схемы 4-го порядка ($I = 3$). Для схемы второго порядка ($I = 1$), этот диапазон шире, порядка $500 \sim 800$.

С учётом этих ограничений и с учётом опыта применения этого решателя уравнения Пуассона на практике, MPI распараллеливание работает эффективно до примерно $2000 \sim 6000$ процессов (в зависимости от размера сетки и порядка схемы). Чтобы выйти за эти границы,

необходимо было дополнить MPI распараллеливание распараллеливанием с общей памятью, что и было сделано в рамках данной работы.

4.3 Двухуровневое MPI+OpenMP распараллеливание

Двухуровневое MPI+OpenMP распараллеливание лучше соответствует современной архитектуре суперкомпьютеров с многоядерными процессорами: MPI используется для объединения узлов суперкомпьютера в рамках модели с распределённой памятью, а OpenMP используется для распараллеливания с общей памятью внутри многоядерного узла суперкомпьютера.

Предположим, что параллельная система имеет по P_t ядер на каждом узле. Тогда MPI группа использует $P_x \times P_{yz}$ узлов, в то время как OpenMP выполняет P_t нитей на каждом узле. Таким образом, суммарно задействуется $P = P_x \times P_{yz} \times P_t$ процессорных ядер. OpenMP распараллеливание также разделено на две части: 1) распараллеливание этапов 1 и 4 алгоритма 2 решателя Пуассона; 2) распараллеливание этапов 2 и 3. В обоих случаях каждая MPI подобласть разделяется на втором уровне на P_t частей.

Отличие состоит в направлениях разбиения: разбиение по оси z используется в первом случае при небольшом числе нитей и по осям y и z при большом, во втором случае разбиение выполняется по периодической оси x (см. рис. 4.1): независимые плоскости динамически распределяются между параллельными потоками.

Таким образом, размер MPI группы, задействующей P CPU ядер, уменьшается в P_t раз, что даёт следующие преимущества.

1. Размер матрицы дополнения Шура (интерфейсной системы) в DSD решателе пропорционален числу P_{yz}. Следовательно, общие затраты памяти, затраты времени на этапе препроцессора, внутренние групповые обмены существенно сокращаются с уменьшением P_{yz}.
2. Аналогично, существенно повышается параллельная эффективность с уменьшением числа P_x за счёт сокращения числа процессов, участвующих в дорогом групповом обмене данными в одномерных подгруппах.
3. Кроме того, сокращается общий объём MPI обмена данными для обновления гало, пропорциональный $P_x P_{yz}$, примерно в P_t раз.
4. Наконец, за счёт сокращения числа MPI процессов ускоряются и сами обмены, поскольку множественным процессам не приходится разделять между собой ограниченные коммуникационные ресурсы узла.

Резюмируя вышесказанное, использование OpenMP на втором уровне распараллеливания позволяет решателю эффективно задействовать в P_t раз большее число CPU ядер относительно только MPI распараллеливания.

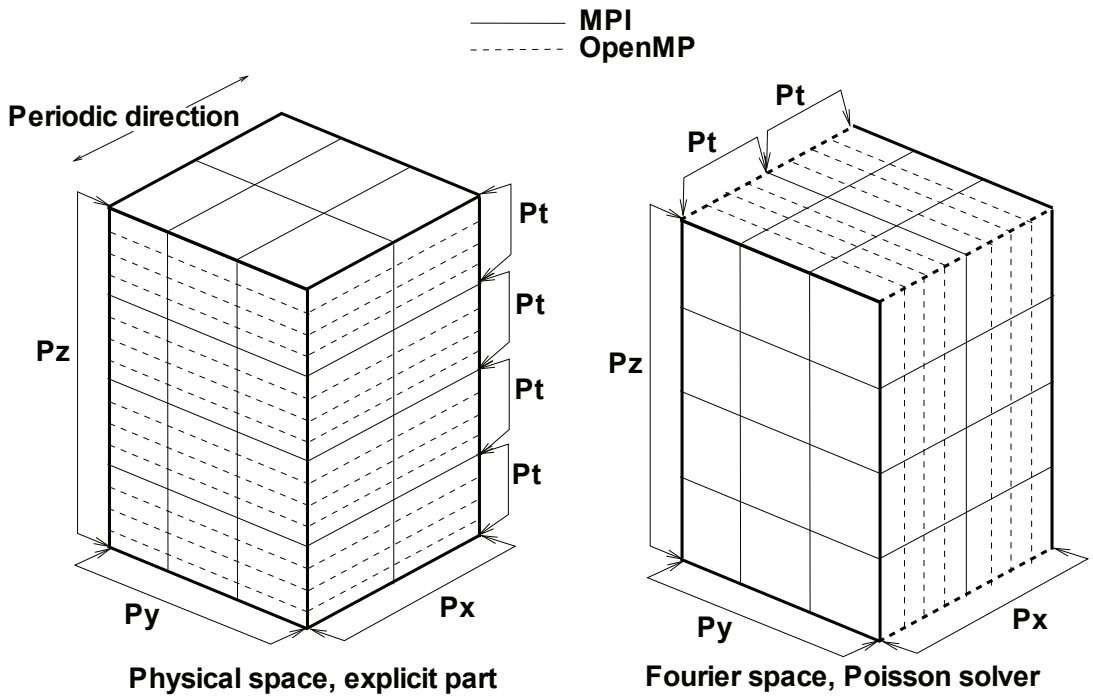


Рисунок 4.1: Декомпозиция расчётной области для MPI+OpenMP распараллеливания

Другое полезное преимущество, что с OpenMP число задействованных ядер можно легко изменять при выполнении расчёта без выполнения дорогого препроцесса на другое число подобластей и без трансформирования распределённой записи восстановления вычислений под новое разбиение расчётной области. Всё это позволяет легче адаптироваться к различным архитектурам вычислительной системы, ограничениям системы очередей и состоянию загруженности системы.

4.3.1 Подробности распараллеливания с общей памятью

Все операции CFD алгоритма в целом и решателя Пуассона в частности хорошо подходят для распараллеливания с общей памятью. Распараллеливание явной части алгоритма 1 (все этапы кроме решения уравнения Пуассона) выполняется путём декомпозиции циклов. Вычисления по подобласти в явной части организованы в виде трёх вложенных циклов по по осям z, y и x. Внутренний цикл соответствует периодическому направлению. Итерации внешнего цикла по оси z распределяются между нитями (см. рис. 4.1 слева). При большом числе нитей выполняется декомпозиция двух внешних циклов, т.е. включая цикл по оси y. К циклам напрямую не применяются директивы OpenMP по распараллеливанию циклов. Вместо этого пересчитываются границы циклов для каждой нити таким образом, что каждая нить получает свой набор итераций, не имеющий пересечений с наборами других нитей, и разница в числе итераций между наборами не превосходит одной итерации. MPI обмены и операции ввода-вывода выполняются только главной нитью.

Этапы 1 и 4 алгоритма 2 решателя уравнения Пуассона распараллеливаются аналогичным образом. Набор независимых БПФ разделяется между нитями, сами же БПФ при таком подходе не требуют распараллеливания.

На этапах 2 и 3 решается набор взаимно независимых $2D$ систем (плоскостей). Эти этапы распараллеливаются простым и естественным образом: между нитями распределяются эти плоскости (см. рис. 4.1, справа). Отдельно распределяются плоскости, для которых используется прямой метод – этап 2, и отдельно плоскости, для которых используется итерационный метод– этап 3. Каждая нить работает со своим набором данных без каких либо пересечений по памяти.

Следует отметить, что в больших расчётах прямым методом решаются только несколько первых плоскостей, а в наиболее масштабируемой конфигурации всего одна. В этом случае получается, что на этапе 2 между нитями нечего распределять. Чтобы улучшить производительность в этой ситуации, прямой DSD решатель также имеет своё внутренне OpenMP распараллеливание. В частности, в параллельном режиме выполняется матрично-векторное произведение с плотной матрицей на этапе решения интерфейсной системы.

На этапе 3 для решения 2D систем используется итерационный метод PCG. Здесь также имеются некоторые тонкости в распараллеливании. Итерации метода сопряжённых градиентов выполняются сразу для всех плоскостей, для которых ещё не удовлетворен критерий сходимости. Базовые операции метода, в частности, скалярное произведение, матрично-векторное произведение, линейная комбинация векторов, локальный предобуславливатель (блочное неполное LU разложение, или неполная обратная матрица, или диагональный метод Якоби) выполняются одновременно для всех решаемых систем. Такая мультисистемная реализация решателя используется для группировки MPI обменов, необходимых операциям скалярного и матрично-векторного произведения. Группировка обменов позволяет избежать излишних потерь на латентность коммуникаций. Естественно, если решение для какой-то системы удовлетворяет критерию по невязке, эта система исключается из процесса.

Операции скалярного и матрично-векторного произведения, линейная комбинация векторов реализованы в виде двух вложенных циклов по осям z и y. Эти операции распараллеливаются аналогично операциям явной части CFD алгоритма путём статической декомпозиции циклов.

Распараллеливанию предобуславливателя требуется уделить особое внимание. Рассмотрим локальный блочный предобуславливатель на основе лентночного неполного LU разложения. Операция выполняется в виде цикла по набору LU для оставшихся систем. Итерации этого цикла распределяются между нитями. Однако, в этом случае итерации цикла существенно неоднородные: разные плоскости могут иметь разный размер блока, отличается и заполненность L и U матриц. Поэтому при декомпозиции цикла напрямую используется директива OpenMP по распараллеливанию цикла с динамической планировкой (*#pragma omp for schedule(dynamic)*) для обеспечения балансировки между нитями. Вместо LU в гетерогенной версии для блоков аналогичным образом может использоваться неполная обратная матрица.

4.4 Расширение масштабируемости за счёт симметрии расчётной области

Дальнейшее расширение масштабируемости, то есть возможность выполнять расчёты на сетках с ещё большим числом узлов и на ещё большем числе процессоров, может быть получено за счёт симметрий расчётной области, характерной для DNS расчётов. Рассмотрим для простоты сетку с одной пространственной симметрией. В этом случае алгоритм решателя уравнения Пуассона может быть модифицирован простой сменой базиса:

$$\mathbf{x}^{\pm} = \mathbf{S}\mathbf{x}, \qquad \text{где} \quad \mathbf{S} = \frac{1}{\sqrt{2}} \begin{pmatrix} \mathbf{I}_{N/2} & \mathbf{I}_{N/2} \\ \mathbf{I}_{N/2} & -\mathbf{I}_{N/2} \end{pmatrix} \in \mathbb{R}^{N \times N}, \tag{4.25}$$

$\mathbf{I}_{N/2} \in \mathbb{R}^{N/2 \times N/2}$ – единичная матрица. Вектора $\mathbf{x}^{\pm} \in \mathbb{R}^N$ и $\mathbf{x} \in \mathbb{R}^N$ могут быть разделены на два подвектора из $N/2$ компонент каждый: $\mathbf{x} \equiv (\mathbf{x}_0, \mathbf{x}_1)^t$ и $\mathbf{x}^{\pm} \equiv (\mathbf{x}^+, \mathbf{x}^-)$, соответствующие численному решению одной части подобласти и симметричной ей другой части. Таким образом, $\mathbf{x}^+ = 1/\sqrt{2}(\mathbf{x}_0 + \mathbf{x}_1)$ представляет симметричную часть $\mathbf{x}$, а $\mathbf{x}^- = 1/\sqrt{2}(\mathbf{x}_0 - \mathbf{x}_1)$ представляет косо-симметричную часть. Следует отметить, что в (4.25) подразумевается одинаковая лексикографическая сортировка для обеих частей. Это означает, что симметричные друг другу узлы сетки будут иметь одинаковые позиции в векторах $\mathbf{x}_0$ и $\mathbf{x}_1$.

Рассмотрим следующую систему уравнений

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \quad \text{with} \quad \mathbf{A} = \begin{pmatrix} \mathbf{A}_P & \mathbf{A}_N \\ \mathbf{A}_N & \mathbf{A}_P \end{pmatrix} \tag{4.26}$$

где $\mathbf{A}_P = \mathbf{A}_{0,0} = \mathbf{A}_{1,1}$ и $\mathbf{A}_N = \mathbf{A}_{0,1} = \mathbf{A}_{1,0}$.

$\mathbf{A}_{i,j}$ представляет связь между поднаборами i и j. Матрица $\mathbf{A}$ симметричная, $\mathbf{A}_N = \mathbf{A}_N^T$. Применяю смену базиса $\mathbf{S}$ к СЛАУ (4.26) получим блочную диагональную систему

$$\mathbf{A}^{\pm}\mathbf{x}^{\pm} = \mathbf{b}^{\pm} \quad \text{with} \quad \mathbf{A}^{\pm} \equiv \mathbf{S}\mathbf{A}\mathbf{S}^{-1} = \begin{pmatrix} \mathbf{A}^+ & \mathbf{0} \\ \mathbf{0} & \mathbf{A}^- \end{pmatrix} \tag{4.27}$$

где $\mathbf{A}^+ = \mathbf{A}_P + \mathbf{A}_N$, $\mathbf{A}^- = \mathbf{A}_P - \mathbf{A}_N$, $\mathbf{x}^{\pm} = \mathbf{S}\mathbf{x}$ и $\mathbf{b}^{\pm} = \mathbf{S}\mathbf{b}$, соответственно. Отметим, что $\mathbf{S}^{-1} = \mathbf{S}$.

В итоге, операции для решения исходной системы (4.26) представлены в Алгоритме 3.

Алгоритм 3

1. Трансформировать вектор правой части, $\mathbf{b}^{\pm} = \mathbf{S}\mathbf{b}$.
2. Решить две разделённые системы: $\mathbf{A}^+\mathbf{x}^+ = \mathbf{b}^+$ и $\mathbf{A}^-\mathbf{x}^- = \mathbf{b}^-$
3. Реконструировать решение исходной системы $\mathbf{x} = \mathbf{S}^{-1}\mathbf{x}^{\pm}$.

Новые системы имеют в два раза меньше неизвестных, чем исходная система. Если у сетки более одной симметрии, этот метод может быть применен рекурсивно, и размер исходной системы сократится в 2^S раз, где S – число симметрий сетки. Действуя таким способом, вместо исходных систем из набора (4.20), задействуя P_{yz} процессоров на N_{yz} неизвестных, можно одновременно решать 2^S систем с $N_{yz}/2^S$ неизвестными подгруппами по $P_{yz}/2^S$ процессоров. Такая декомпозиция может быть применена только для тех 2D систем, для решения которых используется прямой DSD решатель, имеющий ограничения по масштабируемости. Таким образом, размер группы MPI процессов, задействованных в DSD уменьшается в 2^S раз и, следовательно, суммарно может быть задействовано 2^S больше процессоров.

Дополнительными накладными расходами при таком подходе являются два обмена данными на шагах 1 и 3 алгоритма 3, на которых выполняется смена базиса (умножение на $\mathbf{S}$). Однако, эти обмены достаточно дешевые, поскольку имеют тип “точка-точка” и используются только для нескольких первых плоскостей. По сравнению с групповыми обменами по периодическому направлению, вклад в общее время от этих обменов является несущественным и не влияющим на общую параллельную эффективность.

4.5 Расширение для полностью трёхмерных задач

4.5.1 Расширение применимости

Представленный в данной главе масштабируемый параллельный алгоритм для уравнения Пуассона хорошо подходит для широкого спектра вычислительных систем, от небольших кластеров со слабой сетевой инфраструктурой до крупных суперкомпьютеров с десятками тысяч процессоров. Однако, использование БПФ для диагонализации блоков матрицы ограничивает область применимости решателя только задачами с присутствием однородного периодического направления. По той же причине расчётная область не может иметь твёрдых поверхностей, ортогональных периодическому направлению. Такой тип геометрии расчётной области будем условно называть 2.5D геометрией, которая по сути является 2D геометрией, вытянутой по третьему направлению с постоянным шагом сетки. Пример такой геометрии показан на рис. 4.2 слева.

Данный раздел посвящён расширению применимости решателя для расчётов полностью трёхмерных задач: расчётная область без периодики, наличие твёрдых поверхностей по всем трём направлениям.

Например, конфигурация с бесконечным квадратным цилиндром, вмонтированным в стену канала, может быть эффективно смоделирована с использованием KSFD решателя. Но если вместо цилиндра поставить куб (рис. 4.2 справа), то KSFD решатель будет неприменим: всем трём направлениям будут ортогональны твёрдые поверхности и потребуется сгущать сетку вокруг препятствия по всех трёх направлениях.

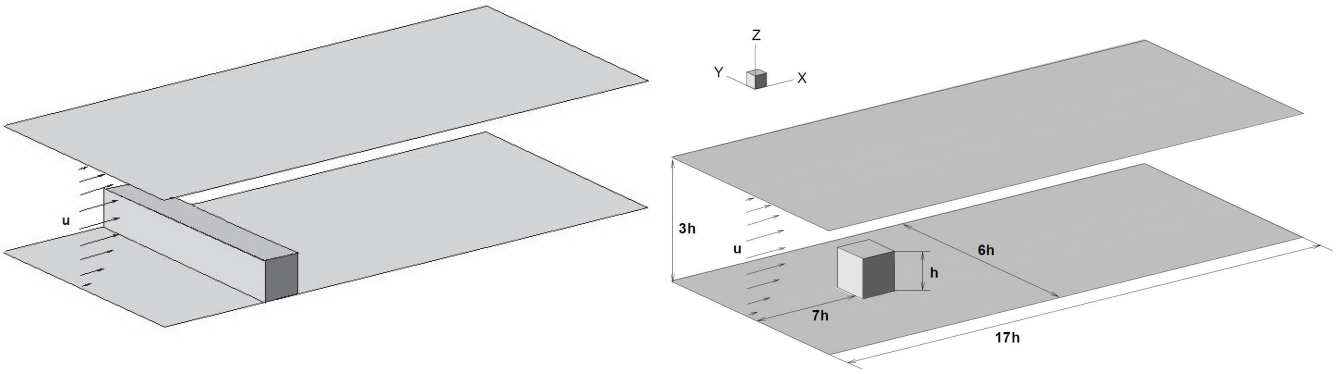


Рисунок 4.2: Вмонтированный в стену квадратный цилиндр в канале – пример 2.5D геометрии (слева), вмонтированный в стену куб – пример 3D геометрии (справа)

4.5.2 Алгоритм многосеточного расширения

Расширение основано на многосеточном подходе и представляет собой по сути многосеточный метод с двумя уровнями. В качестве приближения для полностью трёхмерной задачи на втором уровне используется 2.5D постановка, совместимая с KSFD решателем. Эта 2.5D постановка решается с помощью KSFD метода, а решение исходной 3D системы обеспечивается внешним итерационным методом. Рассматривалось два варианта: внешняя надстройка на основе многосеточного подхода, такое расширение решателя будем обозначать MG-KSFD (от Multigrid-KSFD); внешний итерационный PCG метод для 3D системы, в котором 2.5D постановка используется в качестве предобуславливателя. Первый вариант показал лучшую сходимость и выбор был сделан в пользу MG подхода Общая идея MG-KSFD метода состоит в том, что низкие частоты ошибки, наиболее проблематичные для итерационного метода, эффективно убираются на втором уровне KSFD решателем, а высокие частоты эффективно уничтожатся сглаживателем на первом уровне.

Исходная полностью трёхмерная система

$$\mathbf{A}^{3D}\mathbf{x}^{3D} = \mathbf{b}^{3D} \tag{4.28}$$

и соответствующая ей 2.5D система

$$\tilde{\mathbf{A}}\mathbf{x} = \mathbf{b}, \tag{4.29}$$

где $\mathbf{A}^{3D}$ и $\tilde{\mathbf{A}}$ – симметричные положительно определённые матрицы.

MG надстройка имеет следующий алгоритм на i-й итерации:

1. Сглаживатель: получить приближённое решение $\mathbf{x}_i^{3D}$ системы (4.28) используя метод CG с локальным предобуславливателем.
2. Вычислить невязку $\mathbf{r}_i^{3D}$ для системы (4.28).
3. Трансформировать вектор невязки на второй уровень $\mathbf{r}_i = \mathbf{Q}\mathbf{r}_i^{3D}$
4. Решить уравнение для ошибки $\tilde{\mathbf{A}}\mathbf{z}_i = \mathbf{r}_i$, используя на втором уровне метод KSFD.

5. Преобразовать полученное решение для ошибки со второго 2.5D уровня на первый полностью 3D: $\mathbf{z}_i^{3D} = \mathbf{P}\mathbf{z}_i$

6. Выполнить коррекцию $\mathbf{x}_{i+1}^{3D} = \mathbf{x}_i^{3D} + \mathbf{z}_i^{3D}$

Здесь матрицы $\mathbf{Q}$ и $\mathbf{P}$ представляют собой операторы перехода между уровнями.

4.5.3 Решение на втором уровне

В качестве приближения на втором уровне используется 2.5D сетка с тем же числом неизвестных, координаты узлов по y и z направлениям совпадают, отличие состоит в том, что по оси x на втором уровне используется постоянный шаг сетки. Присутствие твёрдых тел внутри расчётной области на втором уровне попросту игнорируется.

Для реализации граничных условий твёрдой стенки по оси x необходимо модифицировать шаблон схемы в граничных узлах. Помимо этого никаких изменений в KSFD решателе не требуется.

Поскольку число узлов не изменяется, в качестве операторов перехода между уровнями может использоваться единичная матрица. На тестовых 3D задачах было показано, что использование консервативной интерполяции не даёт преимуществ по сравнению с переносом вектора решения без изменений, даже более того, интерполяция ухудшает сходимость.

Если исходная 3D задача имеет периодические граничные условия, то матрица $\tilde{A}$ на втором уровне (4.29) строится аналогично, как и матрица $\mathbf{A}^{3D}$ исходной системы (4.28), с той разницей, что игнорируется присутствие твёрдых тел в потоке. KSFD решатель применяется без каких либо изменений.

В случае, если по оси x в исходной 3D задаче используются условия твёрдой стенки (на внешних границах), необходимо использовать другое БПФ преобразование и модифицировать шаблон схемы в граничных узлах. Модификация шаблона схемы у границ нужна, чтобы сделать матрицу совместимой с БПФ. Матрица $\tilde{\mathbf{A}}$ имеет блочную структуру, блоки соответствуют связям по оси x. Исходно блоки (в случае схемы 2-го порядка аппроксимации) имеют вид (4.30).

$$\mathbf{A}_{j,k}^{p} = \begin{pmatrix} a_p + a_{ew} & a_{ew} & & & & \\ a_{ew} & a_p & a_{ew} & & & \\ & \ddots & \ddots & \ddots & & \\ & & a_{ew} & a_p & a_{ew} & \\ & & & a_{ew} & a_p + a_{ew} \end{pmatrix} \in \mathbb{R}^{N_x \times N_x}. \tag{4.30}$$

А чтобы быть совместимыми с БПФ блоки должны быть вида (4.31).

$$\tilde{\mathbf{A}}^{p}_{j,k} = \begin{pmatrix} a_p & a_{ew} & & & \\ a_{ew} & a_p & a_{ew} & & \\ & \ddots & \ddots & \ddots & \\ & & a_{ew} & a_p & a_{ew} \\ & & & a_{ew} & a_p \end{pmatrix} \in \mathbb{R}^{N_x \times N_x} \tag{4.31}$$

Граничные узлы в системе для уравнения Пуассона отсоединены и могут иметь любые значения, поскольку не учитываются. Это делается модификацией шаблона у границы – ножка, соединяющая приграничный узел с граничным удаляется (зануляется соответствующий коэффициент). В исходном варианте на первом уровне 4.30 значение удалённого коэффициента прибавляется к диагональному элементу (который имеет противоположный знак). Таким образом вклад ножки шаблона, соединяющей внутренний узел с граничным, устраняется. На втором уровне нельзя изменить диагональный коэффициент из-за БПФ, блок получается вида 4.31. Рис. 4.3 показывает различие в построении шаблона у твёрдой границы.

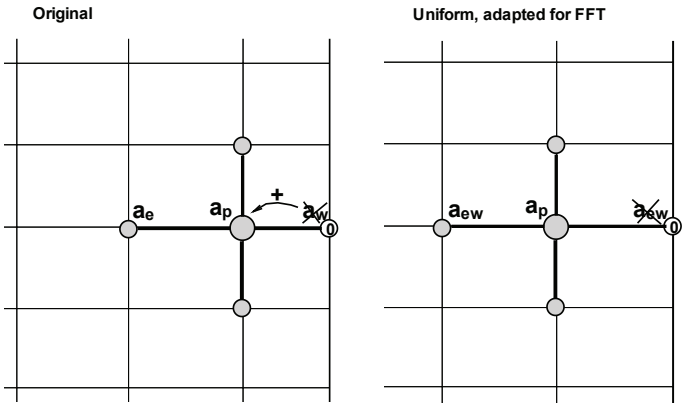


Рисунок 4.3: Модификация шаблона у твёрдой границы

Также для условий твёрдой стенки нужно использовать другое БПФ и собственные значения по сравнению с периодическим случаем. Преобразование Фурье

$$y_j = \frac{1}{n}\left(y_0 + 2\sum_{i=1}^{n-1} y_i cos(ji\frac{\pi}{n}) + (-1)^j y_n\right), \quad j = 0,...,n. \tag{4.32}$$

Обратное преобразование Фурье

$$y_j = \frac{1}{2}\, y_0 + \sum_{i=1}^{n-1} y_i cos(ji\frac{\pi}{n}) + \frac{1}{2}(-1)^j y_n, \quad j = 0,...,n, \tag{4.33}$$

где $n = N_x - 1$, а N_x – число узлов в направлении применения БПФ.
Собственные значения:

$$\lambda_j = a_p + \sum_{i=1}^{m} a_{ew}^i \cos(\frac{\pi}{n} i), \quad j = 0,...,n, \tag{4.34}$$

где m – это размер шаблона.

Матрицы на первом и втором уровне $\mathbf{A^{3D}}$ и $\tilde{\mathbf{A}}$ являются сингулярными. В случае граничного условия твёрдой стенки образ исходной матрицы $im(\mathbf{A}^{3D})$ отличается от образа матрицы второго уровня $\tilde{\mathbf{A}}$. Из-за этого получается, что система второго уровня несовместная, что сильно ухудшает сходимость MG. Чтобы этого избежать, матрица $\tilde{\mathbf{A}}$ должна быть подправлена для устранения сингулярности. Делается это изменением одного диагонального элемента первой плоскости после выполнения БПФ: $\tilde{a}_p^i = 1.1 * a_p^i$. Из-за этого меняется невязка исходной системы в этой точке, но это не портит общую сходимость.

4.5.4 Сходимость и производительность

Производительность MG-KSFD решателя зависит от множества нижеследующих факторов, рассмотрим которые на примере самой проблематичной задачи из тех, для решения которых решатель применялся, – моделирование течения вокруг куба, вмонтированного в стену канала (рис. 4.2 справа).

1. Размер сетки – число итераций растет с числом узлов. На тестах с кубом в канале при увеличении числа узлов в 60 раз с 200 тыс. до 12 млн число итераций вырастало примерно в 4.5 раза.
2. Увеличение неравномерности шага сетки по направлению, в котором применяется БПФ, то есть увеличение сгущения к твёрдым поверхностям, естественно, ухудшает сходимость, а после определённого предела сгущения сходимость вообще пропадает. В данном расчёте этот предел был заметно дальше, чем необходимо для решения задачи. В расчёте куба для сгущения сетки используется функция, распределяющая узлы на единичном интервале со сгущением к краям следующим образом: $x_k = \frac{1}{2}\left(1 + \frac{tanh\{\gamma(2(k-1)/N-1)\}}{tanh\gamma}\right), \quad k = 1,\ldots,N+1$, где γ – это коэффициент сгущения. Интервалы соответствующим образом масштабируются и стыкуются с таким условием, что шаги сетки на краях стыкуемых интервалов совпадают. На рис. 4.4 показан вид сетки, БПФ применяется по оси y, используются 2 интервала $y1, y2$, интервал $y1$ разделяется пополам и стыкуется с двух сторон к интервалу $y2$. При коэффициенте $\gamma = 1$, использовавшемся в расчёте при числе Рейдольдса $Re = 7235$ на сетке из 16 млн узлов [138], наблюдалась устойчивая сходимость, требовалось прядка $5 \sim 7$ внешних итераций (30 итераций сглаживателя, критерий невязки порядка 10^{-4} относительно нормы правой части). При коэффициенте $\gamma = 2$ и более, сходимость пропадала, слишком много узлов оказывалось в узкой зоне постановки препятствия.
3. Выбор числа итераций сглаживателя и точности решения на 2-м уровне, естественно, влияют на число внешних итераций и общее время вычислений.

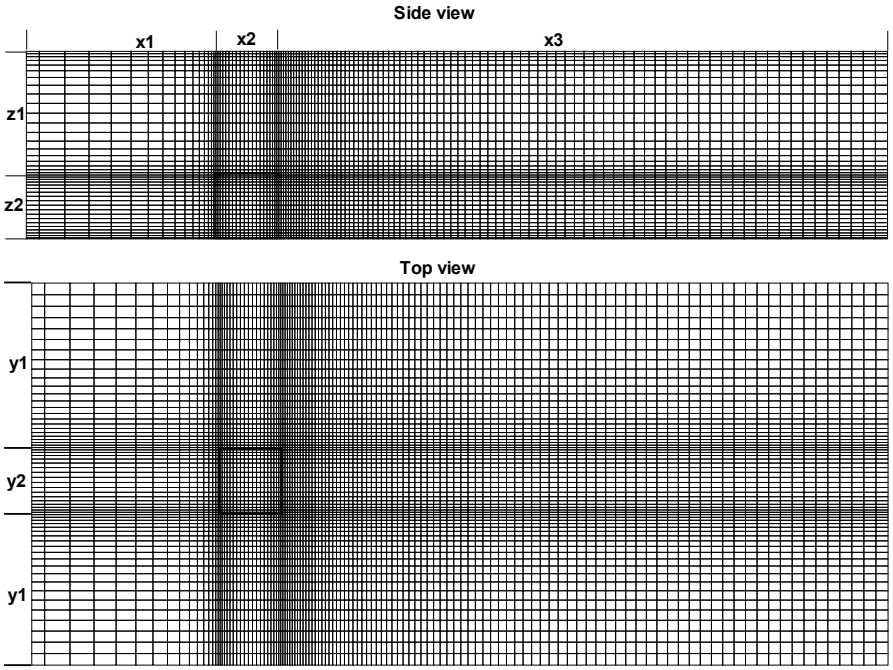


Рисунок 4.4: Вид огрублённой сетки для конфигурации куб в канале

Данный пример является предельным случаем применимости решателя, из-за того, что при увеличении сгущения узлы группируются в узкой зоне расположения препятствия. При использовании равномерной сетки по направлению применения БПФ в той же задаче решателю в среднем требуется менее 2-х итераций (только на учёт влияния твёрдого тела в потоке).

Гораздо более удобный пример – это расчёт течения в закрытой каверне с вертикальными разнонагретыми стенками [139] на сетке из 100 млн узлов с коэффициентом сгущения $\gamma = 1.5$ решателю требовалось в среднем менее 2-х итераций. Более подробная информация о MG-KSFD решателе может быть найдена в [140].

4.6 Общий алгоритм шага интегрирования по времени

Алгоритм для моделирования несжимаемых турбулентных течений с учётом явлений теплопереноса, основанный на представленном выше решателе уравнения Пуассона, составляется, согласно технологии, представленной в первой главе, из сокращённого набора базовых операций.

Набор базовых операций состоит из шести следующих типов:

(T1) линейный оператор (матрично-векторное произведение);

(T2) нелинейный оператор;

(T3) линейная комбинация векторов вида $\mathbf{y} = \sum_{i=1}^{n} a_i * \mathbf{x}_i + c$, где $\mathbf{y}$ и $\mathbf{x}_i$ – сеточные функции (поля), a_i – скалярные коэффициенты, и число слагаемых, n, может варьироваться от 2 до 4.

(T4) БПФ (и обратное БПФ);

(T5) матрично-векторное произведение для набора разреженных матриц;

(T6) редукция – скалярное произведение для набора векторов.

В этих обозначениях с данным набором базовых операций алгоритм моделирования несжимаемых турбулентных течений на основе представленного решателя Пуассона имеет следующий вид.

1. Уравнение момента:
   - 1.1 оператор диффузии – 3 вызова линейного оператора для кожной из компонент вектора скорости (тип T1);
   - 1.2 оператор конвекции – 3 вызова нелинейного оператора (тип T2);
   - 1.3 член массовых сил – 1 вызов линейного оператора (тип T1);
   - 1.4 суммирование членов (три операции типа T3 с тремя слагаемыми).
2. Уравнение переноса температуры:
   - 2.1 оператор диффузии – 1 вызов линейного оператора (тип T1);
   - 2.2 оператор конвекции – 1 вызов нелинейного оператора (тип T2);
   - 2.3 суммирование членов (тип T3 с тремя слагаемыми).
3. Блок промежуточных операций:
   - 3.1 получить поля предиктора скорости – набор операций типа T3;
   - 3.2 граничные условия (производные от T1 операции);
   - 3.3 вычисление поля дивергенции для получения правой части уравнения Пуассона – 3 вызова линейного оператора (тип T1).
4. Решатель уравнения Пуассона:
   - 4.1 трансформировать подвектора правой части в спектральное пространство посредством БПФ, $\hat{\mathbf{b}}_{j,k} = \mathbf{Q}_{\mathbb{R}}^{-1}\mathbf{b}_{j,k}$, (тип T4);
   - 4.2 решить D низкочастотных независимых систем $\hat{\mathbf{A}}_i^{2D}(\hat{\mathbf{x}}_i|\hat{\mathbf{x}}_{N_x-i+2}) = (\hat{\mathbf{b}}_i|\hat{\mathbf{b}}_{N_x-i+2})$ с $i \in \{1,\ldots,D\}$ с использованием прямого метода на основе дополнений Шура DSD;
   - 4.3 решить оставшиеся системы $\hat{\mathbf{A}}_i^{2D}(\hat{\mathbf{x}}_i|\hat{\mathbf{x}}_{N_x-i+2}) = (\hat{\mathbf{b}}_i|\hat{\mathbf{b}}_{N_x-i+2})$ with $i \in \{D+1,\ldots,N_x/2+1\}$ используя итерационный PCG метод (операции типа T3, T5, T6);
   - 4.4 восстановить подвекторы вектора решения $\mathbf{x}_{j,k} = \mathbf{Q}_{\mathbb{R}}\hat{\mathbf{x}}_{j,k}$, (тип T4).
5. Блок промежуточных операций:
   - 5.1 вычисление градиента – 3 вызова линейного оператора (тип T1);
   - 5.2 найти скорость на новом временном слое – набор операций типа T3.
6. Переход на новый временной слой:
   - 6.1 обработка осреднённых полей – набор операций типа T3;
   - 6.2 переключение полей на новый временной слой – набор операций типа T3.
7. Обмен данными для обновления гало.

Общая блок-схема алгоритма показана на рис. 4.5.

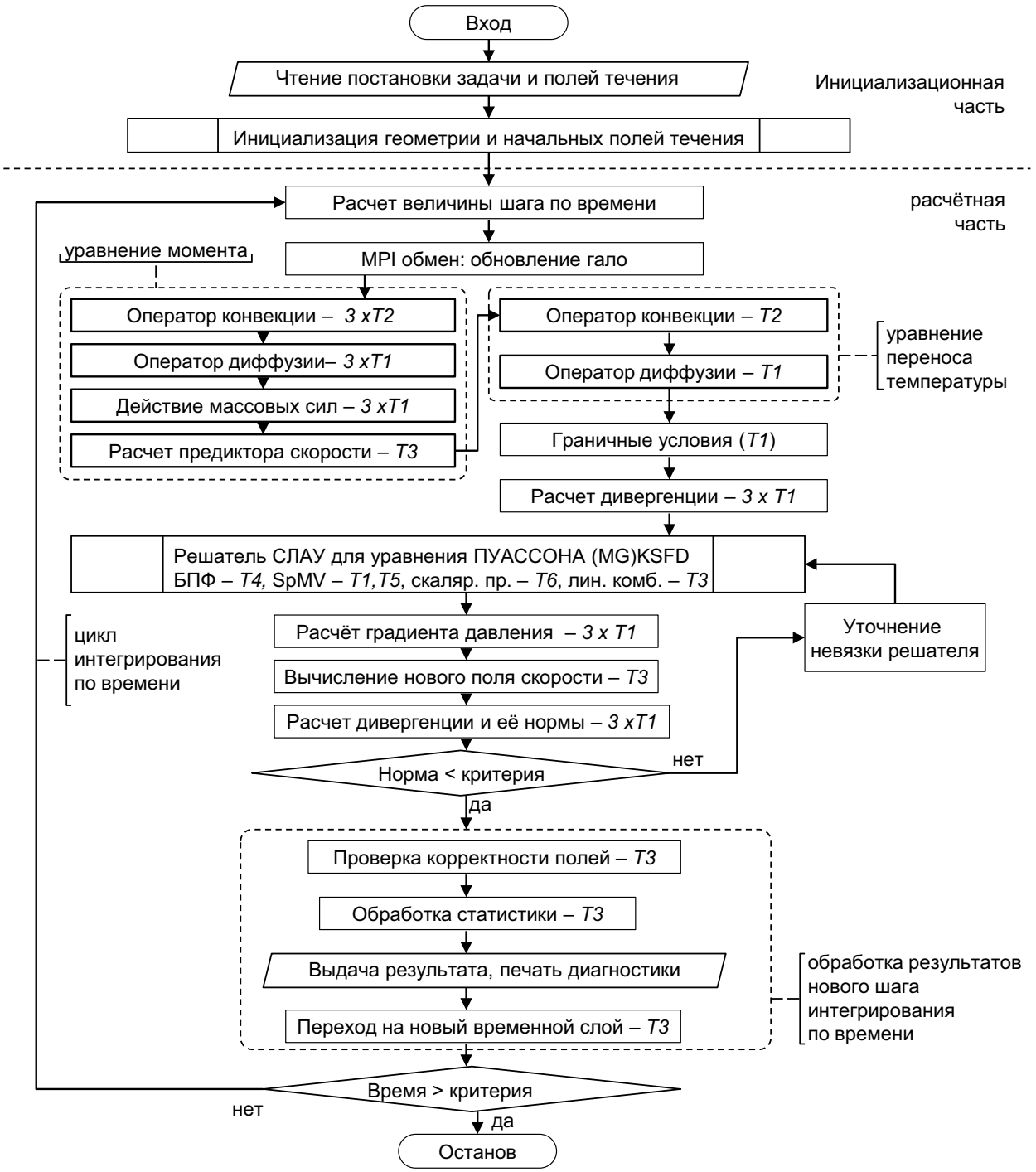


Рисунок 4.5: Блок-схема параллельного алгоритма шага интегрирования по времени

4.6.1 Параллельная эффективность с MPI+OpenMP

В качестве тестовой задачи для измерения параллельной эффективности используется DNS турбулентного течения при естественной конвекции в каверне с вертикальными стенками разной температуры. Подробности по этой конфигурации могут быть найдены в [141]. Размер сетки варьируется от 11 миллионов до 1 миллиарда узлов (см. таблицу 4.1). Измеряется «время настенных часов» на выполнение 100 шагов по времени. Критерий точности фиксирует уровень уменьшения нормы дискретной дивергенции

$$\frac{\|\mathsf{M}\boldsymbol{u}_h^{n+1}\|_2}{\|\mathsf{M}\boldsymbol{u}_h^{p}\|_2} \leq \epsilon \tag{4.35}$$

где $\mathsf{M}\boldsymbol{u}_h^p$ – это значение дискретной дивергенции поля скорости до коррекции. В данном случае критерий установлен $\epsilon = 10^{-4}$. Достаточность этого уровня показана в [126], где также присутствуют подробности о производительности решателя при различных уровнях невязки.

Тестирование параллельной эффективности выполнено на двух различных системах с узлами типичной многоядерной архитектуры. OpenMP распараллеливание настраивалось и тестировалось на суперкомпьютере MVS-100K, затем более крупные тесты были выполнены на суперкомпьютере Ломоносов. Системы имели следующие конфигурации на момент выполнения расчётов:

- МВС-100К МСЦ РАН, 11680 CPU ядер, 8-ядерные узлы с двумя четырехъядерными CPU Intel Xeon E5450 3. 0ГГц, сеть Infiniband DDR 4x.
- Ломоносов, НИВЦ МГУ, 35360 CPU ядер, 8-ядерные узлы с двумя четырехъядерными CPUs Xeon 5570 2.93 GHz, сеть Infiniband QDR.

Тест на сравнение MPI распараллеливания по периодической оси против OpenMP распараллеливания был выполнен на суперкомпьютере Ломоносов на сравнительно грубой сетке (Mesh1 в таблице 4.1) из 1.1×10^7 узлов ($128 \times 192 \times 462$) со схемой 4-го порядка, сохраняющей симметрию. Результаты, представленные в [126], показывают хорошую масштабируемость как минимум до $P_{yz} \approx 128$. Затем, чтобы задействовать большее число процессоров, можно увеличивать либо P_t (OpenMP) и/или P_x (MPI). Сравнение конфигурации $P_{yz} = 128, P_x = 1, P_t = 8$ и $P_{yz} = 128, P_x = 8, P_t = 1$, что соответствует 1024 CPU, показало, что OpenMP конфигурация обгоняет MPI с $P_x = 8$ на примерно 20%. Аналогичные результаты были получены для других сеток и разбиений. Следовательно, режим с OpenMP более эффективен, чем использование больших значений P_x на пределе масштабируемости.

	N_x	N_y	N_z		N	Ra	Pr	Порядок
Mesh1	128	192	462	$\approx$	11.4×10^6	10^{11}	0.71	4
Mesh2	256	800	1600	$\approx$	327.7×10^6	10^{11}	0.71	2
Mesh3	256	1400	2800	$\approx$	1003.5×10^6	10^{11}	0.71	2

Таблица 4.1: Физические и численные параметры тестовых задач.

Следующий тест на суперкомпьютере МВС-100К показывает ускорение с OpenMP для решателя уравнения Пуассона отдельно и для всего CFD алгоритма. Используется та же сетка, что и в предыдущем тесте (Mesh1), число MPI процессов фиксировано, варьируется число нитей P_t. Тест выполнен для MPI групп разного размера: (i) $P_{yz} = 64, P_x = 1$ (рис. 4.6, слева) и (ii) $P_{yz} = 128, P_x = 1$ (рис. 4.6, справа). Видно, что полученное с OpenMP ускорение меньше для большей группы $P_{yz} = 128$. Это связано с тем, что вклад MPI обменов данными в общее

время вычислений растет с увеличением числа MPI процессов. А поскольку только главная нить выполняет обмены, то для обмена данными нет параллельного ускорения. Тем не менее, обмены ускоряются при использовании OpenMP, поскольку на узлах суперкомпьютера находится меньше процессов, разделяющих ограниченные коммуникационные ресурсы узла. Например, обмен данными точка-точка в PCG решателе (нужен для матрично-векторного произведения) ускоряется примерно в 2.5 раза при $P_t = 8$ относительно $P_t = 1$. Вычисления при этом ускоряются заметно сильнее, примерно в 5 раз. Общее ускорение составило 5.5 для группы из 64-х MPI процессов и 4.8 для группы из 128 процессов. Время вычислений для одного вызова решателя Пуассона (и время расчёта одного шага по времени для всего алгоритма) составило 0.92 (2.60) и 0.11 (0.29) секунды на 64 и 1024 CPU ядрах, соответственно.

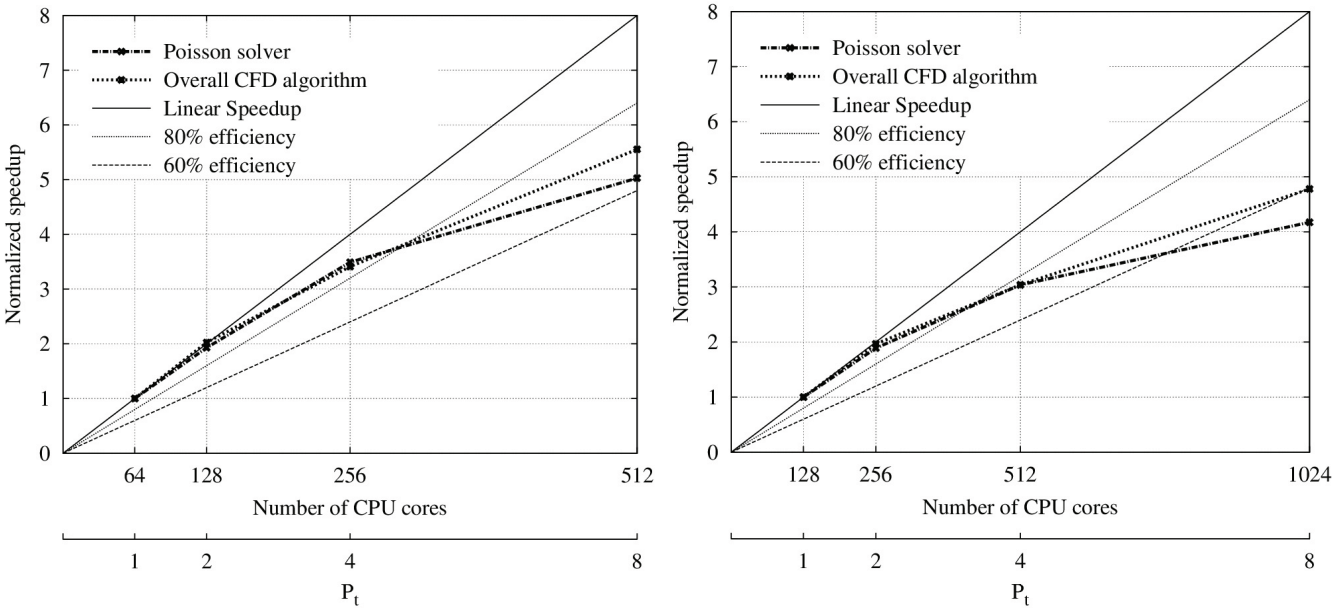


Рисунок 4.6: Нормированное ускорение с OpenMP для решателя Пуассона и для всего CFD алгоритма (P_t варьируется от 1 до 8, P_{yz} и P_x фиксированы) на суперкомпьютере МВС-100К, сетка Mesh1. MPI группы $P_{yz} = 64$, $P_x = 1$ (слева) и $P_{yz} = 128, P_x = 1$ (справа)

Следующие тест выполнен на суперкомпьютере Ломоносов для демонстрации применимости алгоритма на большем числе процессоров. Тесты выполнены для двух сеток: (i) $256 \times 800 \times 1600 \approx 3.3 \times 10^8$ и (ii) $256 \times 1400 \times 2800 \approx 10^9$ узлов, обозначенные Mesh2 и Mesh3 в таблице 4.1, соответственно. Хотя ускорение для схемы 4-го прядка обычно несколько лучше, чем для схемы 2-го порядка, за счёт большей вычислительной стоимости (см. [126]), чтобы сэкономить ограниченное машинное время на этапе препроцесса, тесты были выполнены для схемы 2-го порядка.

Число $P_{yz} = 200$ и $P_t = 8$ фиксированы, число P_x варьируется от 1 (1600 CPU ядер) до 8 (12800 CPU ядер). В такой конфигурации данный тест доходит до исчерпания параллелизма по периодической оси. Разделитель $D = 1$, что означает, что только для одной плоскости, соответствующей самой низкой частоте в пространстве Фурье, используется прямой DSD метод. Время вычислений для решателя Пуассона уменьшается с 0.78 до 0.22 секунд и с 3.82 до 1.00 секунд для сеток (Mesh2) и (Mesh3), соответственно. Ускорение для решателя Пауссона и для

всего CFD алгоритма показано на рис. 4.7. Измерения показывают эффективность удвоения числа процессоров порядка 80% при изменении P_x (на сетке Mesh3).

Естественно, в данном случае увеличить P_{yz} вместо P_x было бы заметно более эффективно. Но в данном тесте моделируется предельная ситуация, когда P_{yz} уже взято «маскимальным» и параллелизм исчерпан. На самом деле P_{yz} в этом тесте можно было бы взять в несколько раз больше, данное значение $P_{yz} = 200$ выбрано исходя из максимального числа ядер, доступных для теста $P = 12800$, с учётом максимальных значений $P_x = 8$ и $P_t = 8$.

Также было отмечено, что хотя ускорение для решателя уравнения Пуассона несколько лучше для более подробной сетки 3.82 (Mesh3) против 3.54 (Mesh2), общее ускорение CFD кода немного лучше для более грубой сетки Mesh2. Это связано с тем, что относительный вклад уравнения Пауссона в общее время вычислений больше для более подробной сетки. В тесте на 12800 CPU ядрах, уравнение Пуассона занимает 56% (Mesh2) и 75% (Mesh3) от общего времени.

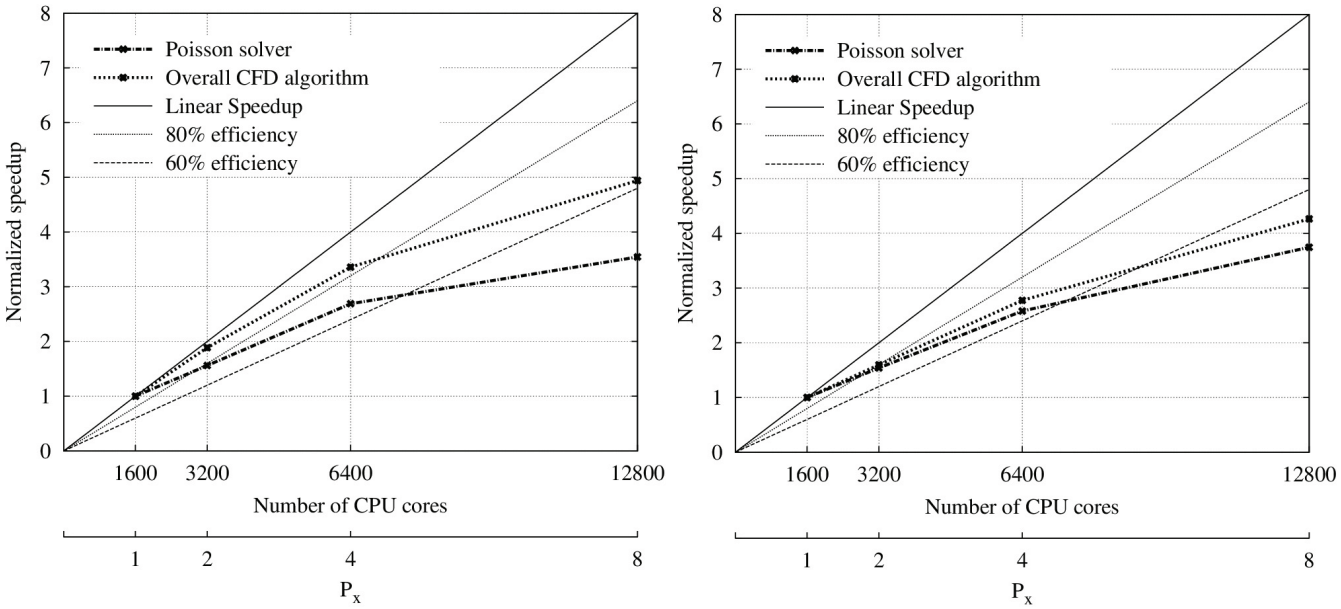


Рисунок 4.7: Нормированное ускорение для решателя Пуассона и для всего CFD алгоритма с MPI+OpenMP распараллеливанием (P_x варьируется от 1 до 8, $P_{yz} = 200$ и $P_t = 8$ фиксированы) на суперкомпьютере Ломоносов, сетки Mesh2 (left) и Mesh3 (right)

Следующий аналогичный тест демонстрирует параллелизм отдельно по числу P_{yz}. На рис. 4.8 показано ускорение для сеток Mesh2 и Mesh3 , конфигурация $P_x = 1$, $P_t = 8$, разделитель $D = 1$. Число P_{yz} варьируется от 200 до 800. На более грубой сетке Mesh2решатель начинает терять параллельную эффективность и достигает исчерпания параллелизма, в то время как на более подробной сеткеMesh3 ещё продолжает наблюдаться ускорение, близкое к линейному. На больших числах P_{yz} ограничивающим фактором уже становится этап препроцесса, на котором выполняется обращение интерфейсной матрицы в DSD методе. Для дальнейшего расширения масштабируемости по числу P_{yz} уже требуется модернизация этой части алгоритма.

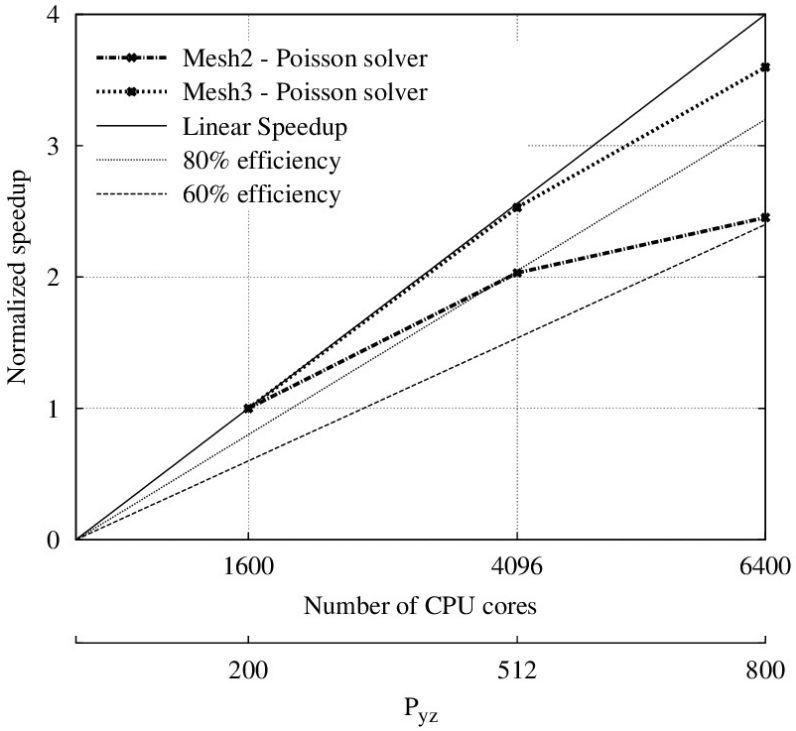


Рисунок 4.8: Нормированное ускорение для решателя Пуассона с MPI+OpenMP распараллеливанием (P_{yz} варьируется от 200 до 800, $P_x = 1$ и $P_t = 8$ фиксированы), сетки Mesh2 и Mesh3

4.6.2 Оценка эффективного диапазона числа процессоров

В приведённых выше тестах на доступных на момент тестирования 12800 CPU ядрах было показано, что передел для MPI распараллеливания составляет не менее $P_x = 8$ и $P_{yz} = 800$. Это соответствует общему числу 6400 MPI процессов. С использованием OpenMP распараллеливания общее число задействованных CPU ядер может быть увеличено в P_t раз, где P_t ограничено числом ядер на узлах суперкомпьютера. Для 8-ядерных узлов это соответствует пределу примерно 50 тысяч ядер ($6400 \times 800 = 51200$). для сетки с числом узлов 1 миллиард и более. Современные суперкомпьютеры имеют двухпроцессорные узлы с 8-ядерными процессорами. Для таких систем диапазон применимости составляет уже порядка ста тысяч ядер, соответственно. Для уже существующих 18-ядерных процессоров Intel Xeon границы составят уже около 240 тысяч.

Применение симметрического расширения позволяет отодвинуть границу до 4-х раз (при наличии двух пространственных симметрий), что позволяет обоснованно ожидать приемлемой параллельной эффективности при числе ядер около 920 тысяч. Учитывая, что, согласно представленному выше тесту, на $P_{yz} = 800$ параллелизм не исчерпывается для больших сеток, это число можно взять и немного больше, можно говорить об эффективном диапазоне применимости порядка **миллиона** процессорных ядер для систем на основе существующих процессоров. Таким образом, можно сделать вывод, что представленный решатель Пуассона и CFD алгоритм в целом принципиально применимы на системах с числом ядер порядка

миллиона. Хотя, конечно, нельзя быть уверенным, что на практике на таком экстремальном числе ядер не возникнут какие-то новые ограничивающие факторы.

Выводы по главе

В главе представлен новый решатель уравнения Пуассона, предназначенный для задач с одним однородным периодическим направлением. С помощью специального нового расширения, также представленного в этой главе, решатель может с некоторыми ограничениями использоваться для задач без периодики. Предложено расширение масштабируемости решателя для задач, имеющих пространственную симметрию. Новая версия параллельного решателя имеет многоуровневое MPI+OpenMO распараллеливание, предназначенное для расчётов на десятках тысяч процессорных ядер. На основе решателя сформулирован параллельный алгоритм для моделирования несжимаемых турбулентных течений с учётом явлений теплопереноса, применимый также на вычислительных системах гибридной архитектуры с массивно-параллельными ускорителями.

Глава 5

Параллельный программный комплекс STG-CFD&HT для крупномасштабных расчётов несжимаемых турбулентных течений

Вводные замечания

В данной главе представлен программный комплекс STG-CFD&HT, основанный на сохраняющих симметрию схемах повышенной точности, который позволяет моделировать задачи газовой динамики и теплопереноса, используя десятки тысяч процессорных ядер суперкомпьютера и массивно-параллельные ускорители различной архитектуры. В основе данной главы лежит материал, представленный в статье [32].

Программный комплекс разрабатывается коллективом, в который помимо автора данной работы входят: Ф.-Х. Триас, М. Сориа, Ф. Даббаг. STG-CFD&HT отличает высокая степень научной новизны – в программном комплексе использованы результаты двух диссертационных работ:

- Ф. Х. Триас – регуляризация, LES модели, схемы повышенного порядка (Ph.D.) [142];
- А. В. Горобец – параллельные технологии и KSFD решатель (к.ф.-м.н.) [81];

История развития кода показана на рис. 5.1. Разработка KSFD метода и параллельного программного комплекса начаты автором в 2004-м году в рамках кандидатской диссертации. Эта часть работы выполнялась в Технологическом центре тепломассопереноса (CTTC) Политехнического университета Каталонии (UPC) в рамках сотрудничестве между ИММ РАН и CTTC. Исходный программный комплекс представлял из себя небольшой параллельный код на языке C, с помощью которого можно было выполнять расчёты на небольших кластерах с использованием 10-20 процессоров и сеток с числом узлов до 10 миллионов. Прямой метод на основе дополнений Шура DSD [127], применявшийся в тот момент для решения уравнения

Пуассона, имел существенные ограничения масштабируемости, связанные с ограничениями по оперативной памяти и групповыми обменами. На первом этапе был модифицирован исходный прямой решатель: за счёт подхода с использованием смешанной точности чисел (32 и 64 бит), применения неполного LU разложения в DSD, изменения распределения строк интерфейсной матрицы между процессами в несколько раз был сокращен объём требуемой памяти и почти в два раза уменьшен объём группового обмена данными. Затем с помощью разработанного KSFD метода границы масштабируемости были расширены ещё как минимум на порядок, стали доступны расчёты на более чем тысяче процессоров и сетках из сотен миллионов узлов. Результатом работы по кандидатской диссертации было создание параллельного программного комплекса на языке C с MPI распараллеливанием для моделирования несжимаемых течений в трёхмерной расчётной области, представленной разнесённой структурированной декартовой сеткой (2007 г.).

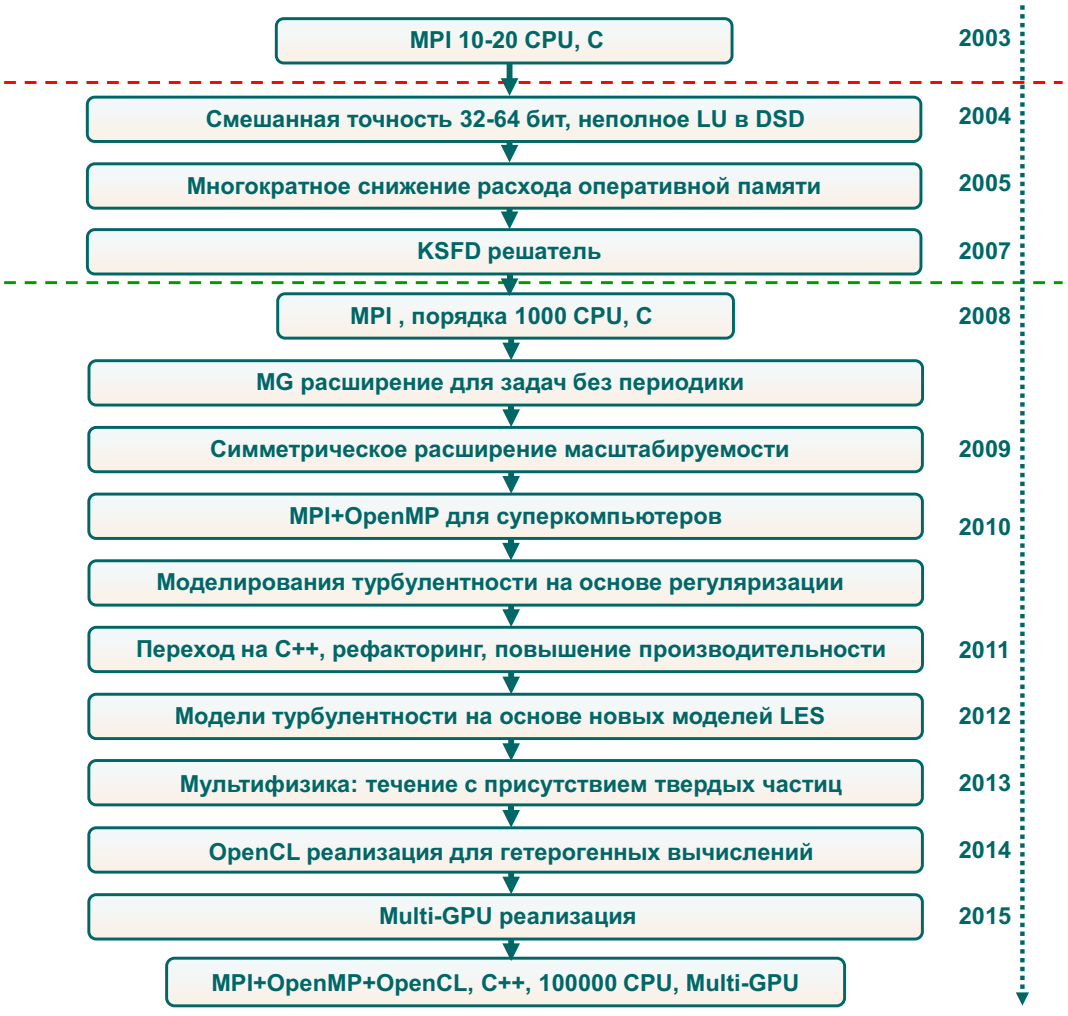


Рисунок 5.1: История развития программного комплекса STG-CFD&HT

Далее, с 2008-го года начинается данная диссертационная работа, в рамках которой областями ответственности автора являются:

– применение разработанной параллельной технологии для реализации программного комплекса;

- программная архитектура и общая структура данных;
- производительность вычислений;
- расширение возможностей KSFD метода: симметричное расширение, многосеточное расширение;
- параллельные вычисления: многоуровневое распараллеливание, гетерогенные вычисления;
- встроенные средства профилирования и отладки.

С этого момента автором осуществлялось руководство программной реализацией.

На начало данной работы исходный программный комплекс имел одноуровневое MPI распараллеливание и мог задействовать порядка тысячи процессоров. В результате в рамках данной работы был создан STG-CFD&HT , который стал полноценным гетерогенным программным комплексом с многоуровневым распараллеливанием на десятки тысяч ядер, с возможностью использовать массивно-параллельные ускорители различной архитектуры, с набором моделей, подходов к моделированию турбулентности на основе регуляризации и LES, с обширной параллельной инфраструктурой. Код программного комплекса написан на языке C++ (стандарта C++03 2003 года) с использованием интерфейсов прикладного программирования MPI (стандарт 1.3) для параллельной модели с распределённой памятью, и OpenMP (стандарт 3.0) для параллельной модели с общей памятью, открытый вычислительный стандарт OpenCL (версия 1.1).

Программный комплекс STG-CFD&HT предназначен для моделирования несжимаемых турбулентных течений с учётом явлений теплопереноса, как в условиях вынужденной конвекции, так и в условиях естественной конвекции. Основу вычислительного алгоритма составляют схемы повышенной точности на структурированных разнесённых сетках, сохраняющие симметрию [133], и специализированный масштабируемый решатель уравнения Пуассона, представленный в предыдущей главе.

Многоуровневое MPI+OpenMP+OpenCL распараллеливание и обширная инфраструктура для распределённой обработки результатов расчёта большого объёма даёт возможность задействовать для одного расчёта несколько десятков тысяч процессорных ядер, а также задействовать множественные массивно-параллельные ускорители, в том числе графические процессоры. Программный комплекс позволяет выполнять расчёты на сетках с числом узлов до нескольких миллиардов. Необходимость применения сложной параллельной модели обусловлена особенностями современной архитектуры суперкомпьютеров с многоядерными вычислительными модулями. Для моделирования турбулентного течения может использоваться прямое численное моделирование, подходы к моделированию турбулентности на основе регуляризации, моделирование LES с обширным набором моделей турбулентности.

Код ориентирован на исследовательскую работу, решение фундаментальных задач, отработку новых моделей, численных методов и вычислительных технологий, что подразумевает удобство внедрения новых методов, моделей и алгоритмов, высокую эффективность использования вычислительных ресурсов, высокую производительность вычислений и надежность.

5.1 Математическая модель и численная реализация

В программном комплексе STG-CFD&HT за основу взята модель течения несжимаемого газа, описываемая полными уравнениями Навье-Стокса. Рассматривается моделирование несжимаемого турбулентного течения ньютоновской жидкости, для зависимости плотности от температуры используется аппроксимация Буссинеска. Для моделирования турбулентности реализовано семейство методов на основе регуляризации [143], семейство LES моделей [70], а также гибридные подходы, сочетающие регуляризацию и моделирование турбулентной вязкости. Для дискретизации по пространству используется конечно-объёмный подход на структурированных декартовых разнесённых сетках на основе семейства сохраняющих симметрию численных схем [133] произвольного порядка точности. Для дискретизации по времени используется проекционный метод дробного шага [129, 130]. Реализованы различные типы граничных условий (ГУ), определяемые заданием потоков на граничных поверхностях расчётной области, включающие:

1. ГУ прилипания (равенство нулю скоростей на границе) на твёрдой стенке: задание температуры стенки (изотермическая поверхность), задание на стенке потока тепла, и, как частный случай, адиабатическая поверхность (равенство нулю теплового потока);
2. условия Дирихле – задание полного набора газодинамических параметров на границе, возможность использования результатов другого расчёта в качестве входных граничных условий;
3. конвективные выходные граничные условия (с коррекцией для точного сохранения массы);
4. условия Неймана.

5.2 Структура программного комплекса

5.2.1 Средства разработки и требования к коду

Исходный код написан на языке C++ стандарта C++03, выбор подходов к программной реализации был обусловлен следующими требованиями:

- совместимость с различными операционными системами (ОС), в том числе Windows XP/Vista/7, Linux;
- совместимость с различными типами вычислительных систем от рабочих станций до крупных суперкомпьютеров, включая гибридные системы с ускорителями различной архитектуры (Intel Xeon Phi, GPU AMD, GPU NVIDIA);
- совместимость с различными компиляторами C++, в том числе Microsoft VS C++, Intel C++, GNU C++, IBM XL C++;
- автономность – базовая конфигурация может работать без подключения каких либо дополнительных библиотек, отличных от стандартных библиотек C++.

Исследовательское предназначение накладывает дополнительные требования, такие как удобство работы с исходным кодом, простота внедрения новых методов и моделей, надежность и защищённость кода от внесения ошибок, высокая производительность вычислений. Переносимость и совместимость подразумевают, что базовая конфигурация не использует никаких программных решений, зависящих от типа ОС и конкретной архитектуры вычислительной системы. Однако, это не исключает расширений базовой конфигурации, в которых используется средства конкретных ОС, настройка под конкретный тип процессора и использование возможностей конкретного компилятора. Аналогичным образом, автономность не исключает дополнения базовой конфигурации функционалом внешних библиотек, в частности, в случае наличия на вычислительной системе используется библиотека FFTW3 [137].

Параллельная конфигурация использует MPI для модели с распределённой памятью, версия стандарта 1.3; OpenMP для модели с общей памятью, стандарт 3.0; OpenCL, версия стандарта 1.1 для вычислений с использованием ускорителей.

STG-CFD&HT состоит из вычислительного ядра и обширной инфраструктуры, включающей в себя средства для постановки задачи и обработки результатов. Вычислительное ядро реализовано в виде библиотеки статической компоновки, от которой питаются исполнимые файлы расчётного кода и инфраструктуры. Набор программных средств условно разделен на 3 категории: препроцессор – программы, выполняющиеся перед началом расчёта; вычислительное ядро – библиотека расчётных функций алгоритма интегрирования по времени и внутренние вспомогательные средства; постпроцессор – программы обработки результатов расчёта.

5.2.2 Инфраструктура препроцессора

В состав препроцессора входят следующие средства:

- препроцесс DSD решателя: вычисление обратной матрицы для интерфейсной матрицы в методе дополнений Шура, используется параллельный метод блочного исключения Гаусса, для блоков используется LU разложение;
- перенос полей течения, записанных в распределённом формате, с одной сетки на другую с использованием консервативная интерполяции;
- перенос полей течения, записанных в распределённом формате, с одной декомпозиции сетки на другую;
- оптимизатор сетки – подбор оптимальных параметров сгущения сетки по набору моментальных картин течения.

5.2.3 Структура вычислительного ядра

Вычислительное ядро программного комплекса реализует алгоритм, представленный предыдущей главе. Алгоритм составлен из базовых операций 6-ти типов, каждая из которых имеет MPI+OpenMP распараллеливание и адаптирована к парадигме потоковой обработке.

Согласно технологии, представленной в первой главе, каждая базовая операция имеет исходную реализацию, ориентированную на центральный процессор, переходную реализацию на C++ в структуре данных, ориентированной на ускоритель, и OpenCL реализацию для вычислений на ускорителях. Исходный код библиотеки вычислительного ядра условно разделен на следующие функциональные модули:

- геометрический модуль: реализует построение контрольных объёмов и геометрических коэффициентов шаблона численной схемы;
- конвективный модуль: расчёт конвективного члена в уравнении Навье-Стокса и в уравнении переноса температуры;
- диффузионный модуль: расчёт диффузионного члена в уравнении Навье-Стокса и в уравнении переноса температуры;
- модуль источников: реализует расчёт действия массовых сил (аппроксимация Буссинеска);
- модуль внешних источников: реализует связь с другими программными комплексами для расчёта мультифизических задач;
- модуль граничных условий: реализует набор граничных условий – твёрдой стенки, входные условия, выходные условия с выполнением законов сохранения массы, внешние динамические условия (на вход подаются данные, полученные из внешнего расчёта, например, турбулентное течение из канала);
- модуль MMS (method of manufactured solutions) верификации;
- модуль (MG-)KSFD решателя уравнения Пуассона;
- модуль статистики течения – накопление осреднения, запись значений в контрольных точках;
- модуль фильтров и регуляризации;
- модуль LES моделей турбулентности.

5.2.4 Средства обработки результатов расчёта – постпроцессор

Результаты записываются параллельными процессами в распределённые бинарные файлы специального формата. Файлы состоят из фрагментов данных, для каждого фрагмента указан размер для проверки целостности файла и контрольный код для определения порядка байтов (для переносимости данных между системами с разным порядком байтов, например между архитектурами Intel и IBM). Реализованы следующие типы записей: 1) распределённая запись набора полей на расчётной сетке (или заданной подсетке – сечения, огрублённая сетка, фрагмент сетки) – моментальные поля, осреднённые поля, записи восстановления счёта; 2) скалярные динамические данные – эволюция во времени выбранных значений в контрольных точках или глобальных величин (сопротивление, подъемная сила и т. д.); Для обработки большого объёма данных реализован набор параллельных средств, включающий:

- суммирование интервалов осреднения (согласно технологии выполнения расчёта, осредненные поля записываются интервалами, поскольку заранее не известен момент выхода течения на статистически однородный режим; после анализа данных и выбора момента начала интегрирования, плохие интервалы отбрасываются, а хорошие – объединяются);
- расчёт статистики течения – дополнительных полей, которые можно вычислить на этапе обработки по основным полям, записанным в процессе счёта;
- осреднение полей по пространственным симметриям;
- уменьшение размерности – осреднение полей по однородному пространственному направлению;
- сборка записи из распределённого формата (каждый MPI процесс записывает данные в свой файл по своей подобласти) – формирует единый файл для всей расчётной области или для заданной подсетки (сечения или огрублённой сетки);
- изменение декомпозиции – запись в распределённом формате для другого разбиения сетки;
- интерполяция записи с одной сетки на другую;
- получение одномерных профилей из записи полей на многомерной сетке;
- средство визуализации – построение файлов в формате TecPlot и ParaView;
- сборка интервалов записи динамических данных с автоматическим удалением перекрытий и интерполяцией на постоянный шаг по времени (для Фурье анализа);
- средства Фурье анализа для построения спектров.

5.3 Особенности программной реализации

5.3.1 Представление данных расчётной области

Рассматривается многоуровневое распараллеливание, в котором вычислительные устройства (процессоры или ускорители), работают с подобластями расчётной области, полученными из декомпозиции с верхнего уровня.

Структуры данных, соответствующие расчётной области, учитывают следующее разделение, согласно технологии из первой главы: сеточные узлы, которые принадлежат вычислительному устройству, составляют его *подобласть*; *гало* узлы – это узлы из других подобластей, связанные с узлами этой подобласти шаблоном численной схемы; *расширенная подобласть* – это объединение подобласти и её гало узлов. Каждое вычислительное устройство работает с расширенной подобластью. Структура расширенной подобласти показана на рис. 5.2.

Основными данными, с которыми оперирует численный алгоритм, являются: описание декомпозиции сетки, описание геометрии контрольных объёмов, поля переменных (сеточные функции).

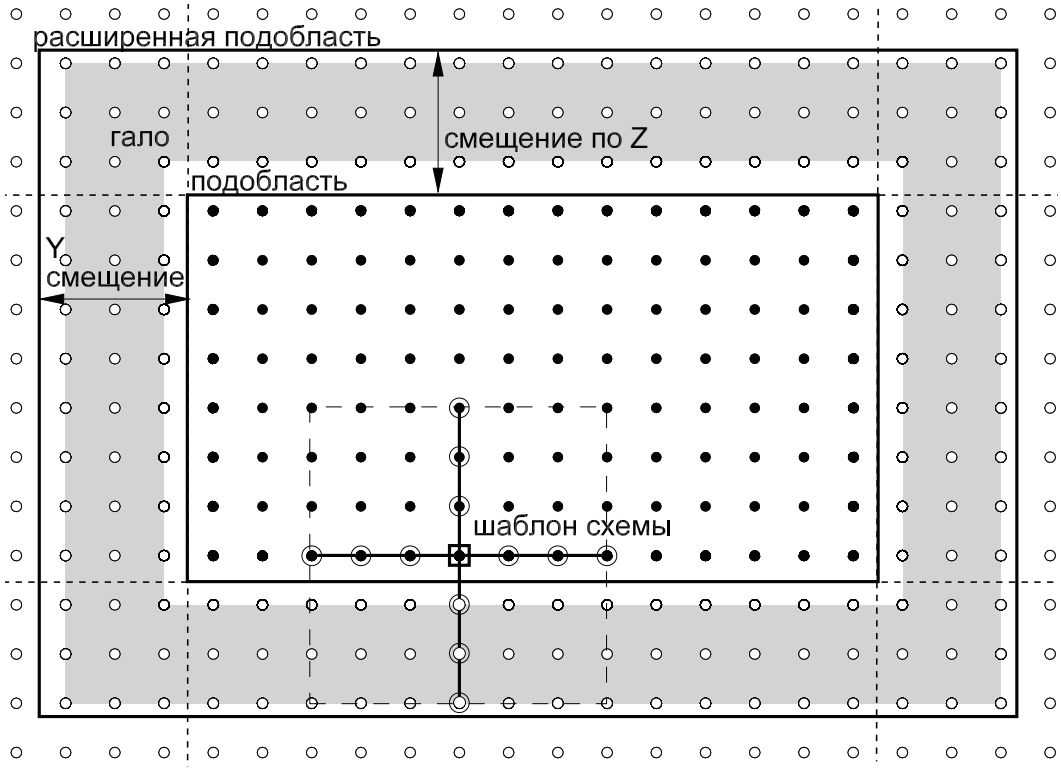


Рисунок 5.2: Расчётная область вычислительного устройства (процессора или массивно-параллельного ускорителя) в рамках декомпозиции верхнего уровня (для простоты показано 2D сечение, ортогональное оси x)

Представление сеточных функций

Поля физических переменных (сеточные функции) размещаются в памяти в виде 1D векторов данных для расширенной подобласти. При этом в вычисления в большинстве операций выполняются только для узлов подобласти, то есть исключая гало. Это несколько усложняет доступ к памяти, поскольку возникает необходимость пропускать гало узлы.

Каждый узел подобласти идентифицируется тремя индексами (k,j,i), исчисляемыми от нуля, которые позиционируют точку в подобласти по трем пространственным направлениям z, y, x, соответственно. В одномерном векторе данных узлы располагаются в лексикографическом порядке. Позиция значения сеточной функции в одномерном векторе данных для заданного узла (k,j,i) задаётся следующим определением препроцессра: (для 3D сетки и для 2D y-z сечения, соответственно):

```
#define _P3D(i,j,k) ((i+OX) + (j+OY)*TX + (k+OZ)*TXY)
#define _P2D(j,k)   ((i+OY) + (j+OZ)*TY)
```

где OX, OY, OZ обозначают гало смещения подобласти внутри расширенной подобласти в x, y, z направлениях, соответственно; TX и TY – это фактические размеры расширенной подобласти (включая гало) по x и y направлениям, TXY= TX*TY.

Номер по x-направлению (периодическое направление) является внутренним в этой нумерации, т.е. соседним узлам по этому направлению соответствуют соседние узлы в векторе

данных. Таким образом, вектор можно рассматривать как блочный с размером блока, равным TX. В этот размер TX могут также входить фиктивные элементы, обеспечивающие выравнивание блоков в памяти по нужной границе (см. рис. 5.3).

Помимо сеточных функций, в виде полей представляются и другие данные: диагонали матриц, коэффициенты шаблона численной схемы, и т.д.

Структура поля имеет следующие свойства:

- указатель на одномерный вектор данных;
- смещённость – три целых числа, описывающих по каждому из трёх направлений, где задана переменная – в узле или в центре грани (0 – узел, 1 – грань)

Представление декомпозиции расчётной области

Структура, описывающая расчётную область, представленную структурированной декартовой сеткой, и её декомпозицию:

- число узлов сетки по трем пространственным направлениям – три целых числа;
- ширина шаблона численной схемы – одно целое число;
- количество подобластей по каждому из трёх направлений – три целых числа;
- номера этой и соседних подобластей по всем направлениям – 27 целых чисел (блок подобластей $3 \times 3 \times 3$);
- границы индексов (k,j,i) для подобласти;
- границы индексов (k,j,i) для расширенной подобласти;
- число узлов подобласти по трем пространственным направлениям;
- число узлов расширенной подобласти по трём пространственным направлениям.

Представление геометрии

Структура, описывающая сеточную геометрию и контрольные объёмы:

- пространственные размеры сетки – три числа с плавающей точкой;
- пространственные границы сетки – шесть чисел с плавающей точкой;
- число контрольных объёмов по каждой оси – три целых числа;
- три указателя на массивы координат узлов сетки по трем пространственным направлениям для всей расчётной области, размера N_x, N_y, N_z, соответственно;
- три указателя на такие же массивы, но для смещённых координат;
- постоянный шаг сетки по оси x в случае наличия однородного периодического направления.

5.4 Реализация базовых операций

5.4.1 Линейный оператор – T1

Линейный оператор представляет собой по сути матрично-векторное произведение с разреженной матрицей, имеющей $1 + (m - 1) * 6$ диагоналей, где m – порядок схемы [133]. Основные схемы, использующиеся в расчётах, имеют либо 2-й либо 4-й порядок. Коэффициенты матрицы линейного оператора не изменяются в течение интегрирования по времени.

В случае, когда имеется однородное периодическое направление (по оси x), матрица состоит из N_x одинаковых “2D” блоков, где N_x – число узлов по оси x. Естественно, в таком случае в памяти хранится только один “2D” блок. В случае полностью трёхмерной задачи, с сеткой, неравномерной по всем трём направлениям, матрицу приходится хранить полностью.

Матрица шаблона схемы, реализующей дискретный линейный оператор, хранится по диагоналям: каждая диагональ хранится как поле (сеточная функция), заданное в каждом узле 2D y-z сечения расширенной подобласти.

Для каждой диагонали значение поля в i-м узле соответствует значению на этой диагонали в i-й строке матрицы. Таким образом, линейный оператор представляется в виде 7 или 19 полей, для схемы 2-го и 4-го порядка, соответственно, хранящихся в виде одномерных векторов данных для расширенной подобласти. Такая структура выбрана из соображений слияния доступа к памяти на ускорителях GPU. Если группа нитей запрашивает коэффициент диагонали для последовательного блока строк, соответствующие значения расположены последовательно в памяти и соседние нити считывают соседние 64-битные позиции, что позволяет нескольким нитям получить значения за одну транзакцию доступа к памяти.

Альтернативный вариант – хранить коэффициенты так, чтобы значения всех ненулевых позиций i-й строки находились последовательно в памяти. Такой вариант подходит для центрального процессора, поскольку уменьшает потери в кэш за счёт компактного расположения данных, обрабатываемых на итерации внутреннего цикла. В случае ускорителя, если бы данные располагались таким способом, расстояние между соседними коэффициентами было бы 448 байт (для схемы 2-го порядка).

Поля коэффициентов шаблона схемы (диагонали матрицы) хранятся в виде одномерных векторов данных одно за другим. Размер поля выбирается с учётом выравнивания, то есть вектор данных может дополняться фиктивными элементами, чтобы следующий за ним вектор начинался с нужной границы. Поля коэффициентов перебираются одно за другим путём прибавления размера поля к указателю на вектор данных полей. Реализация обобщена на случай, если шаблон схемы имеет разное число узлов в разных пространственных направлениях, что может быть применимо, например, в сильно анизотропных задачах. Структура линейного оператора показана на рис. 5.3.

При реализации на OpenCL базовых операций по возможности циклы разворачиваются или их границы заменяются на `#define` константы, `if` ветвления заменяются `#if` определениями

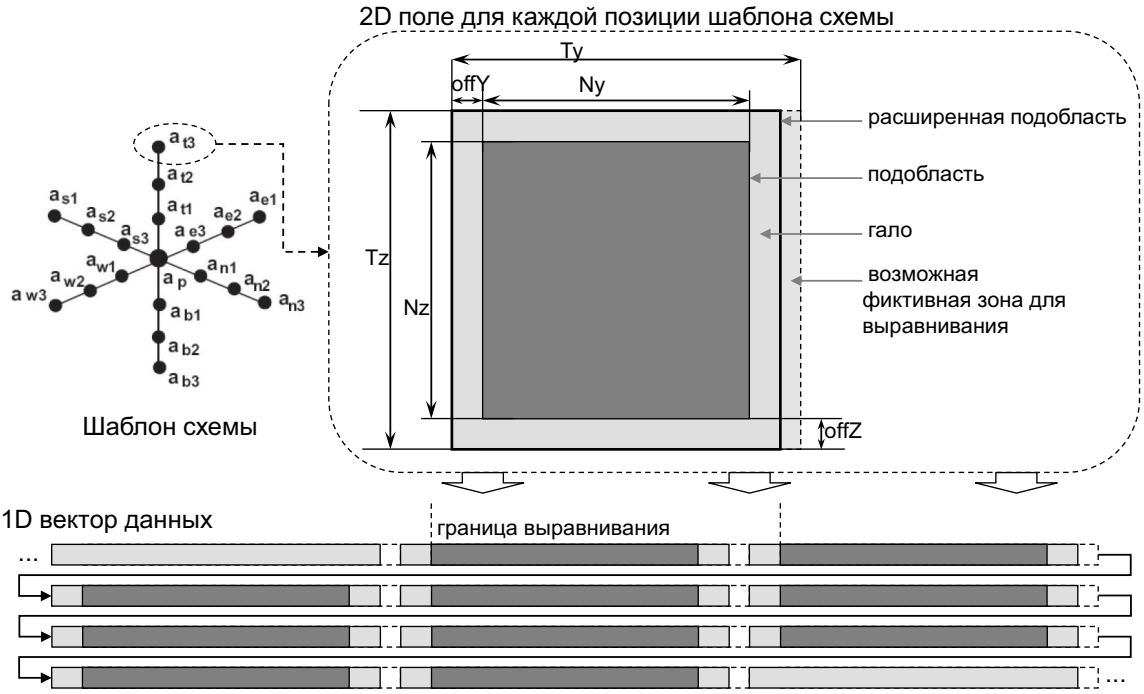


Рисунок 5.3: Схема структуры данных для дискретизации линейного оператора

препроцессора. Это возможно, поскольку компиляция OpenCL ядер происходит во время выполнения CPU программы. Такая замена позволяет избежать накладных расходов на ветвления и организацию циклов, достаточно существенных на акселераторах, и тем самым повысить производительность.

Элементарным заданием в данном случае является обработка одного узла сетки. Каждая нить независимо обрабатывает свой узел. Нити собраны в локальные рабочие группы по x направлению – блок нитей обрабатывает узлы с одинаковой позицией по y и z осям. Если оптимальный размер блока больше, чем N_x, то блок может обрабатывать несколько соседних линий. Если N_x больше, чем оптимальный размер блока, то операция разделяется на несколько вызовов операции для разных частей по x оси (это более эффективно, чем иметь цикл внутри ядра).

Рассмотрим реализацию линейного оператора более подробно. Все вхождения в коде, которые не определены в теле ядра, являются `#define` константами, подставляемыми в начале исходного кода перед компиляцией во время выполнения программы на CPU.

```
__kernel void LinearStencilProduct(
  __global double *inField, /* input field */
  __global double *outField,/* output field*/
  __global double *st       /* stencil data*/){

//позиция узла по оси X
  const int lID = get_global_id(0)%ls;
//позиция узла в плоскости, ортогональной оси X
```

```
  const int I   = get_global_id(0)/ls;

  if((I>=NYZ)||(lID>=NX)) return;

//индексы узла по Y и Z осям
  const int i1 = I%NY+offY;
  const int i2 = I/NY+offZ;

//соотвествующая позиция в 2D и 3D полях
  const int p2d  = _P2D(i1,i2);
  const int p3d0 = _P3D(offX+lID,i1,i2);

//сначала коэффициент на главной диагонали
  double pr = stencil[p2d]*(inField[p3d0]);
  stencil+=fieldSize; //переходим на следующую диагональ

  MULSTENCIL(0); // шаблон схемы по оси X
  MULSTENCIL(1); // шаблон схемы по оси Y
  MULSTENCIL(2); // шаблон схемы по оси Z

  outField[p3d0]=pr; //запись результата
}
```

Шаблон численной схемы по трём пространственным направлениям обрабатывается следующим макро определением:

```
// перебираем узлы "слева" и "справа" по данной оси
// шаблона схемы aa (0-x,1-y,2-z)
#define MULSTENCIL(aa)                                    \
for(int s=1; s<=st_ssize##aa; s++){                       \
  const double buf1=st[p2d]; /*коэффициент слева*/        \
  stencil+=fieldSize;/*переход на следующее поле*/        \
  const double buf2=st[p2d]; /*правый коэффициент.*/      \
  stencil+=fieldSize;/*переход на следующее поле*/        \
  /*расстояние в памяти между соседними узлами по этой оси*/ \
  const int offst3Daa = offst3D##aa*s;                    \
  pr +=buf1*inField[p3d0-offst3Daa] /*вклад слева*/       \
      +buf2*inField[p3d0+offst3Daa];/*вклад справа*/}
```

5.4.2 Нелинейный оператор – T2

Представление нелинейного оператора более сложное, поскольку коэффициенты шаблона зависят от текущих полей компонент вектора скорости (адвекции). Вычисления выполняются

таким образом, что сначала в каждом узле сетки по геометрическим коэффициентам, неизменным в процессе интегрирования по времени, и полю адвекции, вычисляются коэффициенты шаблона. Затем шаблон применяется к полю входных данных. Эти два этапа объединены в одном программном ядре, чтобы избежать необходимости записи и повторного чтения коэффициентов шаблона.

Для ускорения генерации коэффициентов шаблона, на этапе препроцесса вычисляется набор геометрических коэффициентов в каждом узле сетки. Эти вычисления выполняются один раз при запуске расчёта и полученные геометрические коэффициенты напрямую применяются к соответствующем позициям поля адвекции.

Число геометрических коэффициентов, приходящихся на узел шаблона, в общем случае переменно и может быть меньше или равно длине ножки шаблона. Поскольку используются разнесённые сетки, требуется много логических операций для определения соответствующей позиции в поле скорости (в зависимости от того, является ли выходное поле смещённым в конкретном направлении, является ли данный компонент скорости смещённым в направлении ножки шаблона и т.д.). Получается более эффективным просто использовать дополнительный массив, содержащий позиции в поле скорости, соответствующие геометрическим коэффициентам. В таком CSR-подобном формате геометрические коэффициенты хранятся вместе с индексами соответствующих им позиций в полях адвекции. Данная структура показана на рис. 5.4.

После того, как выполнено позиционирование на наборы геометрических коэффициентов и индексов позиций в полях адвекции, следующим макро-определением вычисляются коэффициенты шаблона:

```
//bcoef - позиция набора геометрических коэффициентов,
//bi - позиция набора индексов,
//nr - размер набора, padv - поле адвекции (компонента в данном направлении)
#define GET_BCPV(padv)                    \
 (((nr>0) ? bcoef[0]*padv[bi[0]] : 0.0) + \
  ((nr>1) ? bcoef[1]*padv[bi[1]] : 0.0) + \
  ((nr>2) ? bcoef[2]*padv[bi[2]] : 0.0) + \
  ((nr>3) ? bcoef[3]*padv[bi[3]] : 0.0))
```

Такая конструкция с использованием тернарного оператора позволяет избежать цикла с переменными границами и ветвлений в коде, которые несут большие накладные расходы. Когда вычислены коэффициенты шаблона, результат находится аналогично линейному оператору.

5.4.3 Линейная комбинация – T3

Операция типа T3 вида $\mathbf{y} = \sum_{i=1}^{n} a_i * \mathbf{x}_i + c$ реализована для расширенной подобласти. Такой способ с излишними вычислениями в гало оказался более эффективным, поскольку из-за очень низкой вычислительной стоимости это выгоднее, чем делать вычисления по подобласти (без

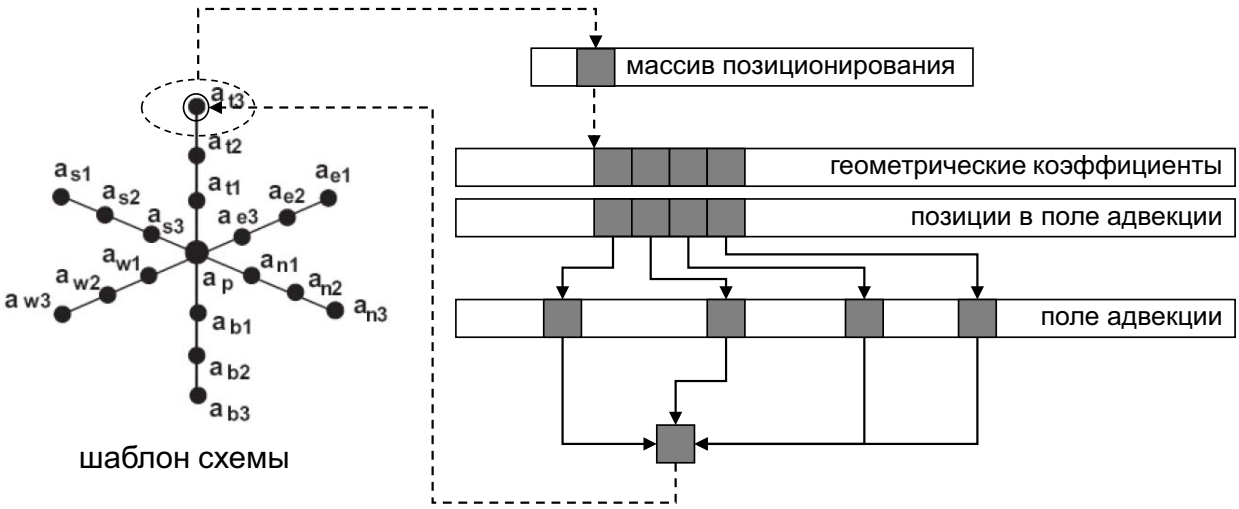


Рисунок 5.4: Схема генерации коэффициентов шаблона схемы для нелинейного оператора

гало), нарушая непрерывность вычислений и вводя дополнительные смещения и логические операции. Вычисления по расширенной подобласти оперируют с полным 1D вектором данных и реализация предельно упрощается: каждая нить независимо вычисляет результат для своего узла.

5.4.4 БПФ – T4

Каждому узлу из y-z плоскости соответствует вектор из N_x элементов. Для каждого из этих векторов независимо выполняется идентичное БПФ. Операция с выполнением набора из большого количества независимых БПФ распараллеливается в рамках потоковой обработки простейшим образом. Элементарным заданием является одно БПФ. Каждая нить независимо выполняет БПФ для своего вектора. Все коэффициенты БПФ вычисляются заранее и разделяются между локальными рабочими группами.

5.4.5 Матрично-векторное произведение – T4

Реализация матрично-векторного произведения с “3D” матрицей, которое используется на верхнем уровне в MG-расширении KSFD решателя (раздел 4.5), идентична типу T1.

В KSFD решателе реализован мультисистемный PCG метод, в котором решение ищется одновременно для N_x независимых “2D” систем (плоскостей). Поэтому данная операция выполняется для набора из N_x матриц и векторов. Реализация для гетерогенных вычислений ограничивается схемами 2-го и 4-го порядка аппроксимации, которым соответствуют 5- и 13-диагональные матрицы, соответственно. Данная операция имеет существенные отличия от “3D” версии.

Поскольку “плоскости” обладают разными числами обусловленности, для разных плоскостей требуется разное число итераций решателя. Реализация матрично-векторного произведения

для множественных матриц и векторов также должна позволять отключать матрицы по мере достижения сходимости. С этой целью, входные данные операции дополняются массивом флагов размера N_x, который хранит для каждой плоскости статус сходимости. Если сходимость до заданного критерия достигнута, флаг устанавливается и матрица выбывает из вычислительных операций.

Ещё одна особенность состоит в том, что “2D” матрицы отличаются только главной диагональю, все остальные ненулевые позиции одинаковы, поскольку БПФ только убирает связи по периодическому направлению и не затрагивает внедиагональные элементы, соответствующие связям по другим направлениям. Поэтому узлы с одинаковой y-z позицией имеют одинаковые внедиагональные коэффициенты. Чтобы использовать этот факт, нити группируются по x направлению, чтобы разделять общие внедиагональные коэффициенты.

Матрицы хранятся определённым образом в единой структуре данных в виде непрерывного одномерного массива в памяти. Главные диагонали всех матриц хранятся подряд как N_x 2D полей. затем подряд хранятся другие диагонали в виде 4 или 12 полей, соответствующих узлам 5-ти или 13-точечного шаблона схемы 2-го или 4-го порядка, соответственно. Все входные и выходные вектора также объединяются в единую структуру данных и хранятся один за другим как непрерывный одномерный массив в памяти.

Элементарным заданием в данной операции является вычисление одной позиции в одном из N_x векторов. Следующий исходный код иллюстрирует реализацию данной операции для схемы 2-го порядка.

```
__kernel void SpMV_multi_GPU(
  __global double *a, //массив коэффициентов всех матриц
  __global double *InV,  //массив всех входных векторов
  __global double *OutV, //массив всех выходных векторов
  __global int *finished /*флаги сходимости*/){

  int I=get_global_id(0)/NYZ; //номер плоскости
  int p=get_global_id(0)%NYZ; //номер узла в плоскости
  if((I>=NX)||finished[I]/*сходимость достигнута*/)return;

  //позиционирование в плоскости
  I *= M/*размер одного 2D поля*/;
  InV += I;
  //позиция узла в векторе данных с учётом гало
  p = (p%NY+offY) + TY*(p/NY+offZ);

  double avd=a[I+p]*InV[p]; //главная диагональ
  a+=NX*M; //переход на поля побочных диагоналей
  avd += InV[p-1     ]/*AW*/*a[p]; //j-1 (y)
  a+=M //переход на следующее поле;
```

```
  avd += InV[p+1     ]/*AE*/*a[p]; //j+1 (y)
  a+=M //переход на следующее поле;
  avd += InV[p-offstz]/*AS*/*a[p]; //k-1 (z)
  a+=M //переход на следующее поле;
  avd += InV[p+offstz]/*AN*/*a[p]; //k+1 (z)

  OutV[I+p]=avd; //запись результата
}
```

5.4.6 Скалярное произведение – T6

Операции редукции (скалярное произведение, нормы) реализованы стандартным образом: каждая локальная рабочая группа выполняет редукцию в локальной общей памяти, затем одна нить выполняет редукцию результатов локальных рабочих групп в глобальной памяти. Особенность состоит в том, что операция может выполняться для набора векторов, для чего входные данные может дополнять массив флагов сходимости.

5.4.7 Операции решателя Пуассона

KSFD решатель включает базовые операции 4 типов – БПФ, матрично-векторное произведение с разреженной матрицей, редукция (скалярное произведение, нормы), линейная комбинация векторов ($\mathbf{y} = a_1 * \mathbf{x}_1 + a_2 * \mathbf{x}_2$). Из этих операций формируется PCG решатель для 2D систем с предобуславливателем Якоби или неполной обратной матрицы – SAI (sparse approximate inverse).

PCG метод применяется к набору независимых систем (плоскостей) одинакового размера, отличающихся только главной диагональю. Поскольку число обусловленности систем существенно отличается, для разных систем требуется заметно разное число итераций.

Для группировки обменов данными итерационный метод применяется сразу ко всем плоскостям. Соответственно, все операции метода работают с множественными наборами данных. Например, обмен данными для матрично-векторного произведения для набора матриц, требующий обновления гало для входных векторов, группируется и выполняется за одно одну операцию, уменьшая до N_x раз потери на латентности сети.

KSFD решатель работает в гетерогенном режиме, используя одновременно CPU и ускоритель. Прямой метод DSD, который применяется для нескольких систем, соответствующих наиболее низким частотам в пространстве Фурье, а в больших расчётах только для одной системы из сотен, работает на CPU. Итерационный PCG метод, применяющийся для всех остальных систем работает на GPU. Реализация DSD метода на GPU нецелесообразна, поскольку существенные затраты времени уходят в этом методе не на вычисления, а на групповой обмен данными на этапе суммирования вкладов подобластей в интерфейсную часть. Обмен данными при переносе DSD на ускоритель только замедлится и получить какое-либо ускорение вряд ли

удастся. Схема работы в гетерогенном режиме показана на рис. 5.5. После БПФ вектора правой части для систем, решаемых прямым методом, передаются с ускорителя на CPU (одновременно с вычислениями на ускорителе). Затем DSD методом находится решение для данных систем, и выполняется копирование решения с CPU на ускоритель.

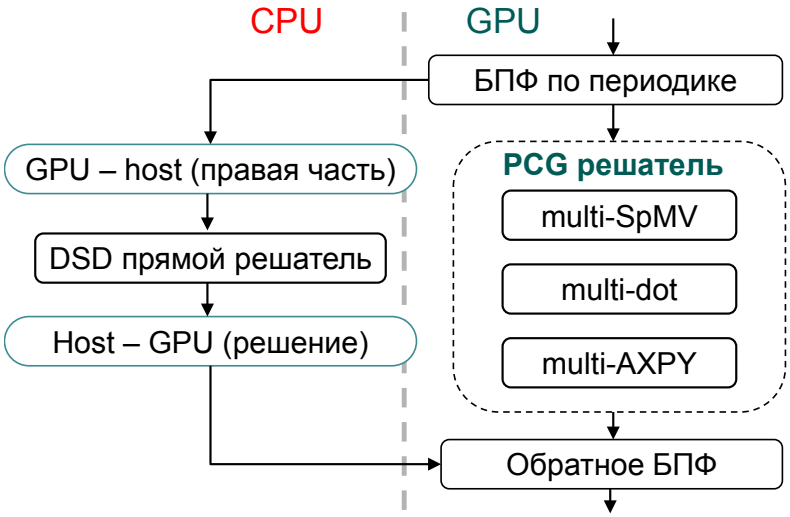


Рисунок 5.5: Схема работы KSFD решателя в гетерогенном режиме

5.5 Производительность базовых операций

Тестирование производительность вычислений (числа с плавающей точкой двойной точности) выполнено для схемы 4-го порядка аппроксимации на сетке, содержащей 1.28 миллиона узлов, $N_x = 64$. В тестовой задаче моделируется естественная конвекция в каверне с противоположными вертикальными стенками разной температуры, аналогично [141].

В тестовых расчётах использовалось следующее оборудование:

- 6-ядерный CPU Intel Xeon X5670 2.93GHz, 12 GFLOPS на ядро, (Суперкомпьютер К-100 ИПМ им. М.В. Келдыша РАН);
- GPU NVIDIA Tesla C2050, 515 GFLOPS, 144 Gb/s (Суперкомпьютер К-100 ИПМ им. М.В. Келдыша РАН);
- GPU AMD Radeon 7970, 947 GFLOPS, 264Gb/s (рабочая станция с GPU).

Для компиляции CPU кода использовался GNU g++ компилятор с полной оптимизацией. Для реализации программных ядер базовых операций использовался OpenCL 1.1. Для устройстве AMD использовался драйвер OpenCL версии 898.1 (оказалось, что производительность заметно меняется в зависимости от версии драйвера). При расчётах на различных GPU использовался один и тот же OpenCL код программных ядер. На уровне кода не было специфической адаптации программы к конкретной модели GPU. Единственный настраиваемый параметр – это размер локальной рабочей группы.

Таблица 5.1 показывает ускорение базовых операций, сравнивая время вычислений на одном ядре CPU и на одном GPU ускорителе. Также в таблице приводится ускорение для уравнения

Операция	время на CPU, сек.	NVIDIA C2050 ускорение, раз	AMD 7970 ускорение, раз
диффузия (T1)	0.08	22.5	45.5
конвекция (T2)	0.25	17.6	43.0
ур-е момента (T1,T2,T3)	0.39	18.3	47.0
лин. комб. (T3)	0.01	18.0	96
м.-в. п. (T4)	0.042	19.2	44.7
БПФ (T5)	0.035	7.7	12.7
скалярное пр. (T6)	0.002	7.6	11.6

Таблица 5.1: Ускорение различных операций, сравнивая одно GPU устройство с одним ядром CPU Intel Xeon (сетка 1.28 млн узлов, схема 4-го порядка).

Операция	1 Xeon core		NVIDIA C2050		AMD 7970	
	GFLOPS	% от пика	GFLOPS	% от пика	GFLOPS	% от пика
диффузия (T1)	2.4	20	42	8.1	85	9
конвекция (T2)	2.0	17	36	7	88	9.3
лин. комб. (T3)	1.1	9	19.3	3.7	143	15.1
м.-в. п. (T4)	0.76	6.3	14.6	2.8	34	3.6
БПФ (T5)	1.5	12	11.5	2.2	19	2.0
скалярное пр. (T6)	1.2	10	9	1.7	14	1.5

Таблица 5.2: Достигаемая фактическая производительность и % от пика для базовых операций.

момента, которое состоит из оператора диффузии (3 вызова линейного оператора для каждого пространственного направления), оператора конвекции (3 вызова нелинейного оператора) и операции типа T3.

Результаты по достигнутой фактической производительности и проценту от пиковой производительности устройств приведены в таблице 5.2. Число операций с плавающей точкой подсчитано напрямую из алгоритма.

Представленные результаты показывают сравнение с одним ядром CPU, чтобы сравнивать одно GPU устройство с одним 6-ядерным CPU Intel Xeon, полученное ускорение следует поделить примерно на 4 (характерное среднее ускорение с OpenMP на данном 6-ядерном процессоре для всего алгоритма).

Как видно из результатов, фактическая производительность, извлекаемая из GPU, достаточно далека от пиковой производительности с точки зрения вычислений с плавающей точкой. Однако, основным ограничением для рассматриваемых операций, имеющих низкую удельную вычислительную стоимость на единицу данных, является пропускная способность памяти. Для GPU NVIDIA C2050 соотношение вычислительной производительность 515 GFLOPS и пропускной способности 144Gb/s составляет 3.6. Примерно такое же соотношение и у AMD 7970 GPU. Для чисел с плавающей точкой двойной точности размером 8 байт это соотношение следует умножить на 8. Это означает, что для достижения производительности, сопоставимой с пиковой, должно быть около 30 операций с плавающей на один аргумент двойной точности из

глобальной памяти GPU устройства. В рассматриваемых базовых операциях это соотношение примерно на порядок меньше.

Наглядное сравнение показателей производительности для базовых операций показано на рис.5.6–5.8 Сравнение фактически извлекаемой производительности из CPU и различных GPU показано на рис. 5.6 в процентах от пиковой производительности устройств и в GFLOPS на рис. 5.7 Рис. 5.8 показывает полученное ускорение на базовых операциях в сравнении с одним ядром CPU на различных GPU.

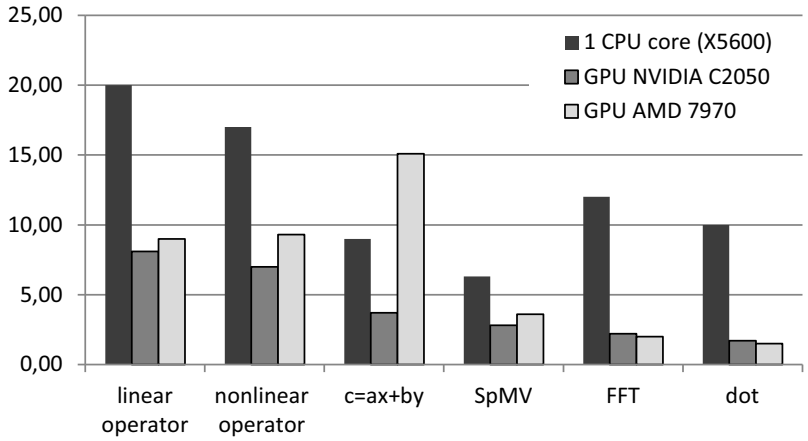


Рисунок 5.6: Сравнение фактической производительности в процентах от пиковой производительности устройств на базовых операциях

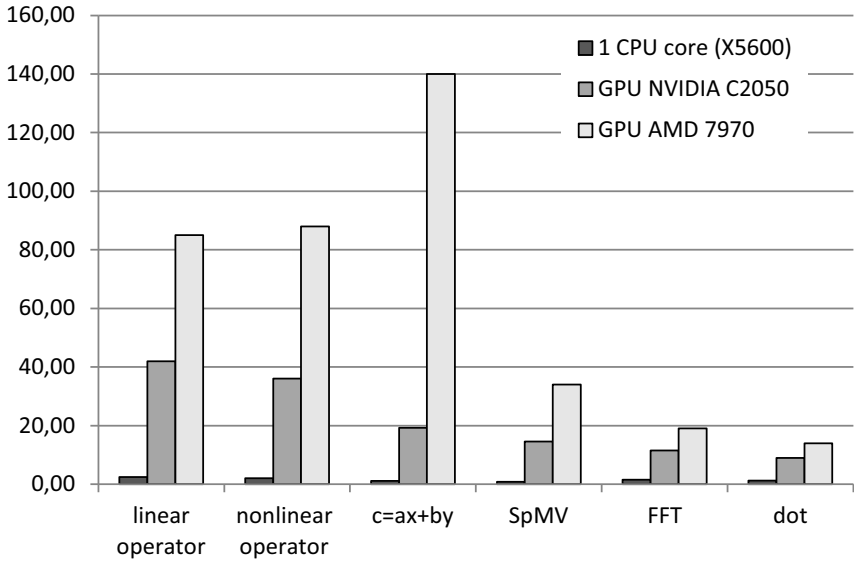


Рисунок 5.7: Сравнение фактически достигаемой производительности на базовых операциях, GFLOPS

Сравнение производительности всего алгоритма на различных многоядерных процессорах и ускорителях показано на рис. 5.9. За единицу взята производительность 6-ядерного CPU Intel Xeon X5670. Из результатов видно, что получаемая производительность на ускорителе примерно в 2.5–3 раза выше, чем на CPU того же поколения. Это заметно меньше, чем

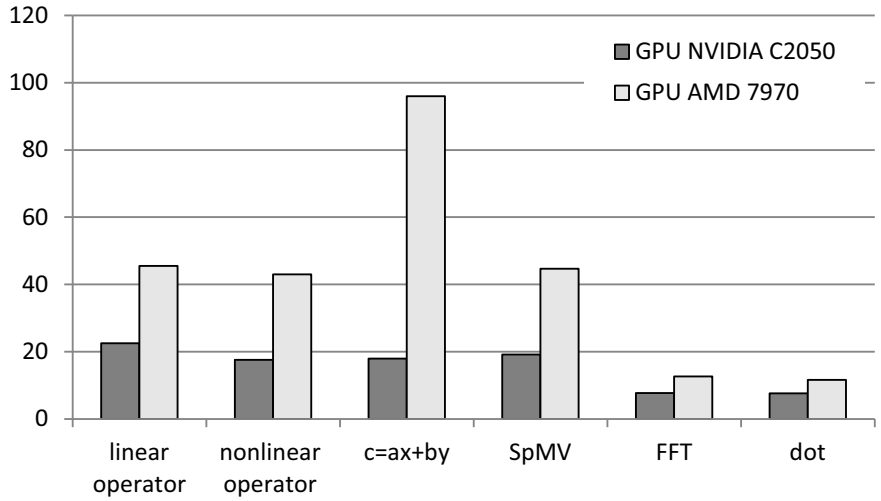


Рисунок 5.8: Ускорение на базовых операциях в сравнении с одним ядром CPU

ускорение отдельных базовых операций. Ожидаемое ускорение исходя из среднего ускорения базовых операций составляет около 5 раз. Однако, полученные результаты вполне предсказуемы, поскольку в реальном расчёте операции KSFD решателя в основном выполняются не для всего набора плоскостей, а только для нескольких первых плоскостей, соответствующих низким Фурье частотам. Для большинства плоскостей решение быстро сходится и они выбывают из итерационного процесса. Графики распределения числа итераций по плоскостям представлены в [126]. Поскольку сильно уменьшается объём вычислений, то, с одной стороны, увеличивается вес обмена данными, а, с другой стороны, сильно снижается коэффициент загрузки мультипроцессора, в следствие чего невозможно достигать такого ускорения на GPU, как на отдельных операциях для полного набора плоскостей.

Результаты по ускорению и масштабированию на множественных GPU показаны на рис. 5.10. Тесты выполнены на суперкомпьютере К-100 на расчёте турбулентного течения при естественной конвекции схемой 4-го порядка.

Выводы по главе

Представленный в главе новый программный комплекс STG-CFD&HT, основанный на схемах повышенной точности на структурированных разнесённых сетках, представляет собой инструмент для решения исследовательских фундаментальных задач, выполнения крупномасштабных суперкомпьютерных расчётов, отработки новых методов, моделей и алгоритмов. Программный комплекс использовался при проведении совместных исследований с Техническим университетом Каталонии, Испания; Университетом Гронингена, Нидерланды; компанией TermoFluids, Испания; Школой машиностроения при Xiangtan University, Китай. Многоуровневое распараллеливание MPI+OpenMP+OpenCL позволяет задействовать десятки тысяч процессорных ядер суперкомпьютеров, а также гибридные системы с массивно-

параллельными ускорителями различной архитектуры. Параллельные средства инфраструктуры позволяют выполнять расчёты на сетках с числом элементов более миллиарда.

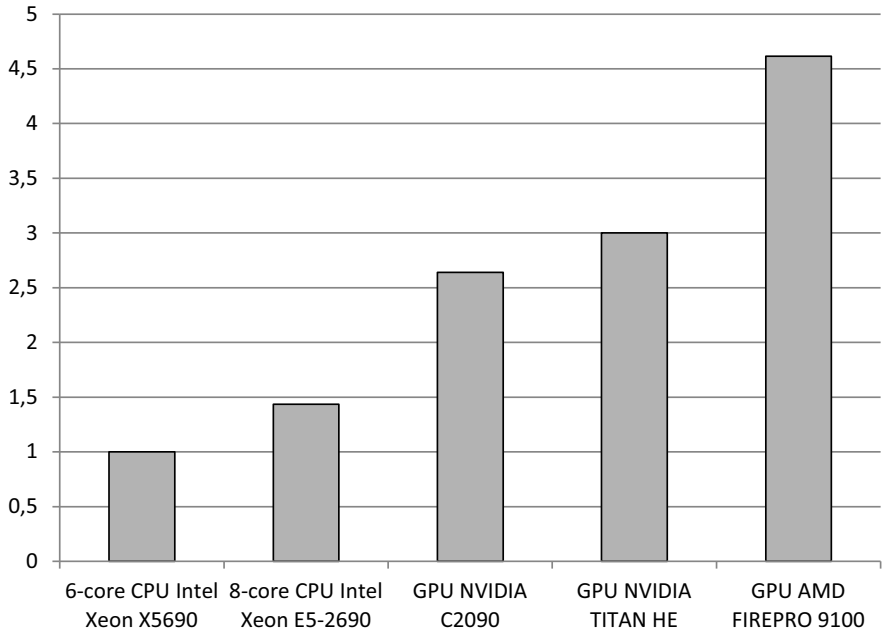


Рисунок 5.9: Сравнение производительности всего CFD алгоритма на расчёте турбулентного течения при естественной конвекции, сетка 1.3 млн узлов, схема 4-го порядка (за единицу взята производительность 6-ядерного CPU)

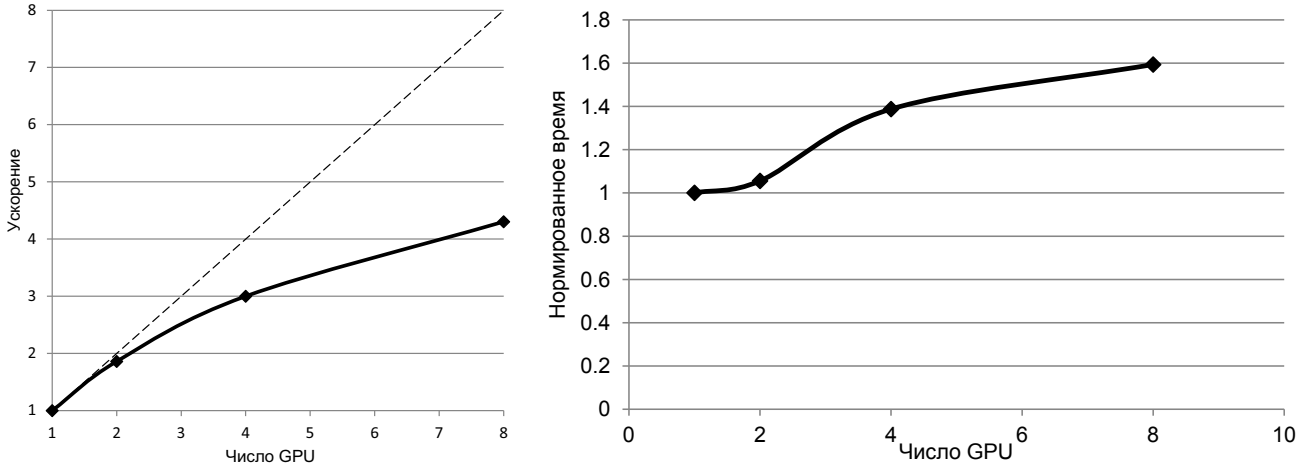


Рисунок 5.10: Ускорение вычислений на множественных GPU, сетка 2 млн узлов, схема 4-го порядка (слева); масштабирование на множественных GPU, сетка 1 млн узлов на одно GPU устройство, схема 4-го порядка (справа)

Глава 6

Крупномасштабные расчёты турбулентных течений

Вводные замечания

В этой главе представлены крупномасштабные расчёты ряда фундаментальных задач по моделированию турбулентных течений, выполненные лично автором по разработанной технологии с помощью программных комплексов, представленных в данной работе. В главе используется материал, опубликованный в статьях, посвящённых представленным расчётам.

6.1 Дозвуковое турбулентное обтекание тандема цилиндров с квадратным сечением

6.1.1 Постановка задачи

В модельной задаче рассматривается течение сжимаемого газа вокруг тандема квадратных цилиндров, выставленных по одной линии вдоль по потоку. Расчёт выполнен с помощью представленного в 3-й главе программного комплекса NOISETTE. Исследуются источники звука от взаимодействия турбулентных структур с твёрдым телом и источники в турбулентном следе. Воспроизводятся базовые механизмы аэродинамического шума при обтекании стоек шасси самолета в режиме захода на посадку. Для задачи имеются подробные экспериментальные данные, полученные в NLR (Аэрокосмический центр, Нидерланды) в рамках работ по Европейскому проекту FP7 VALIANT [144, 145].

Конфигурация состоит их двух квадратных цилиндров, выставленных по одной линии. Расстояние между центрами цилиндров составляет S=0.16м, рассматриваются два угла атаки: $\alpha = 0^\circ$ и $\alpha = 10^\circ$ (рис. 6.1).

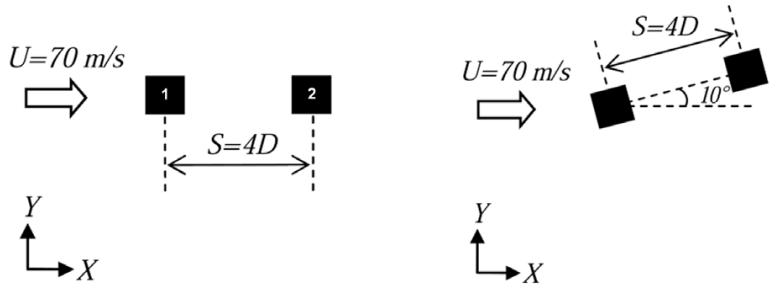


Рисунок 6.1: Конфигурация из двух квадратных цилиндров

Оба цилиндра имеют ширину D=0.04м. Скорость невозмущённого воздушного потока составляет 70 м/с (M = 0.2) при нормальном атмосферном давлении и комнатной температуре. Число Рейнольдса (по ширине цилиндров) равно 182,000.

Расчётная область представляет собой параллелепипед с периодическими граничными условиями по одной оси и цилиндрами, установленными вдоль этой оси (рис. 6.2). Ось цилиндров направлена вдоль периодической оси, таким образом, воспроизводится конфигурация бесконечных цилиндров в однородном потоке.

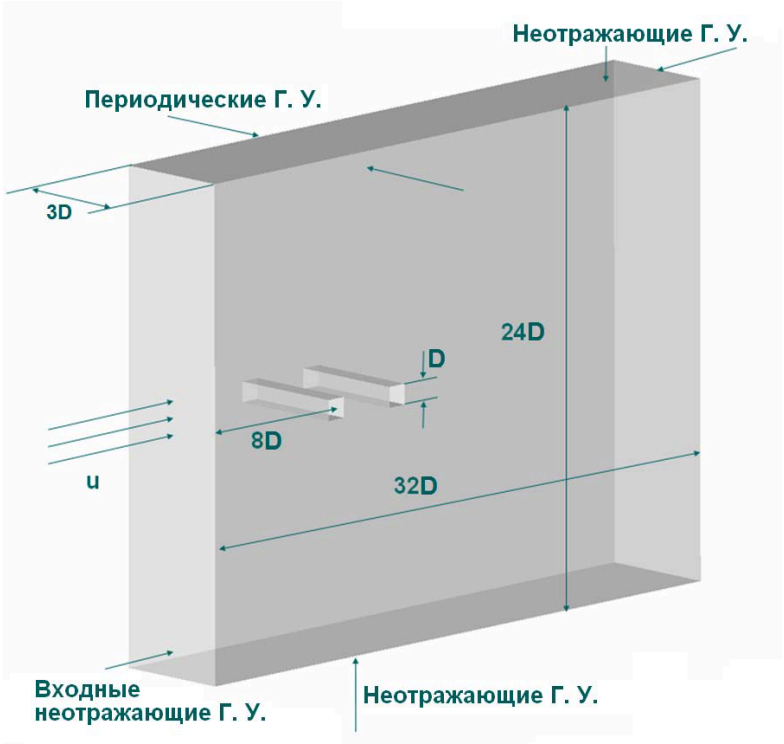


Рисунок 6.2: Конфигурация расчётной области

На выходе из численного и физического экспериментов представлены следующие типы данных:

- осреднённые по времени (и по пространству вдоль периодической оси в случае численного эксперимента) поля компонент скорости в центральном сечении;
- осреднённые по времени профили поверхностного давления в центральном сечении;

- распределение интенсивности турбулентности в центральном сечении;
- спектр пульсаций давления (спектральная плотность мощности в дБ/Гц по линейной шкале) на каждой из четырёх сторон каждого из цилиндров;
- акустический спектр в дальнем поле (спектральная плотность мощности в дБ/Гц по линейной шкале) в различных точках на окружности радиуса 2м с центром на оси первого (по потоку) цилиндра.

Численная задача решается в трёхмерной декартовой системе координат. Расчётная область представляет собой параллелепипед с двумя твёрдыми телами в форме квадратных цилиндров с одинаковым поперечным сечением. Характеристическая длина для обезразмеривания – сторона цилиндра, D. На поверхности цилиндров заданы граничные условия твёрдой адиабатической стенки без проскальзывания: $\boldsymbol{u} = 0$, $\partial T/\partial \mathbf{n} = 0$. Входные граничные условия воспроизводят однородное плоскопараллельное течение, заданное следующими безразмерными параметрами: $\rho = 1,\ u = 1,\ v = 0,\ p = 1/\gamma \mathrm{M}^2,\ \nu = \mu = 1/\mathrm{Re}$, где ν и μ – турбулентная и молекулярная вязкость, соответственно. На выходных границах заданы неотражающие граничные условия. Периодические граничные условия заданы на границах, ортогональных оси цилиндров. Для пространственной дискретизации используется тетраэдральная неструктурированная сетка, содержащая 13 миллионов узлов и 72 миллиона тетраэдров. Вдоль периодической оси сетка имеет 90 равномерных шагов. Для аппроксимации уравнений используется вершинно-центрированная EBR схема [31]. Для дискретизации по времени применяется неявная трёхслойная схема 2-го порядка с линеаризацей по Ньютону. Для моделирования сжимаемого вязкого турбулентного течения используется гибридный DDES (Delayed Detached Eddy Simulation) подход [72], сочетающий RANS и LES подходы. Для замыкания RANS уравнений используется модель турбулентности Спалларта-Аллмараса (SA). Расчёт выполнялся на суперкомпьютере Ломоносов НИВЦ МГУ.

6.1.2 Осреднённая по времени статистика течения

Данные для двухмерных осреднённых полей течения получены путём осреднения по времени и по пространству вдоль периодической оси (поскольку задача однородна в этом пространственном направлении).

Угол атаки 0^o

Рисунки с 6.3 по 6.7 отображают осреднённые по времени различные поля течения и соответствующие им экспериментальные данные, полученные PIV (particle image velocimetry) измерениями.

На Рис. 6.8 и Рис. 6.9 показано сравнение с экспериментом осреднённых профилей различных величин вдоль линии, соединяющей центры граней цилиндров.

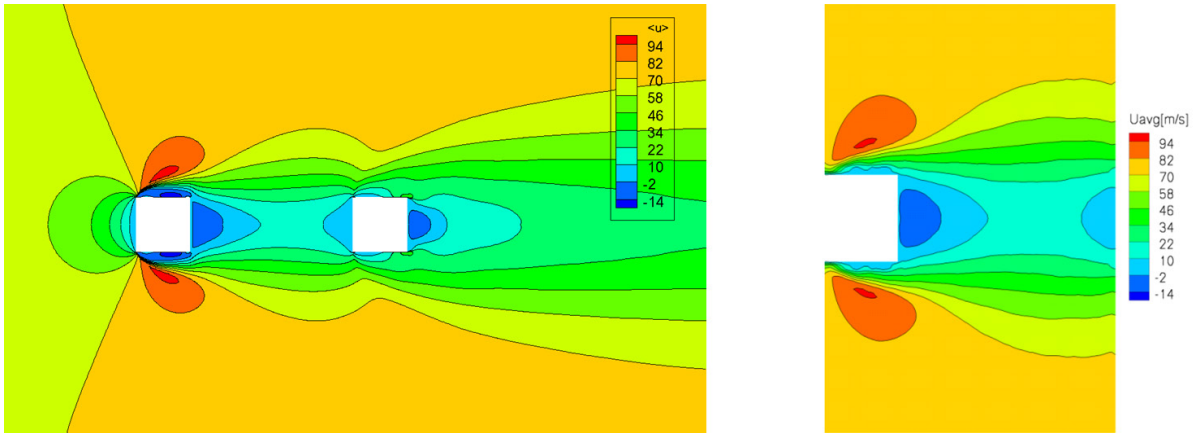


Рисунок 6.3: Осреднённое поле продольной компоненты скорости по результатам численного моделирования (слева) и экспериментальных измерений (справа)

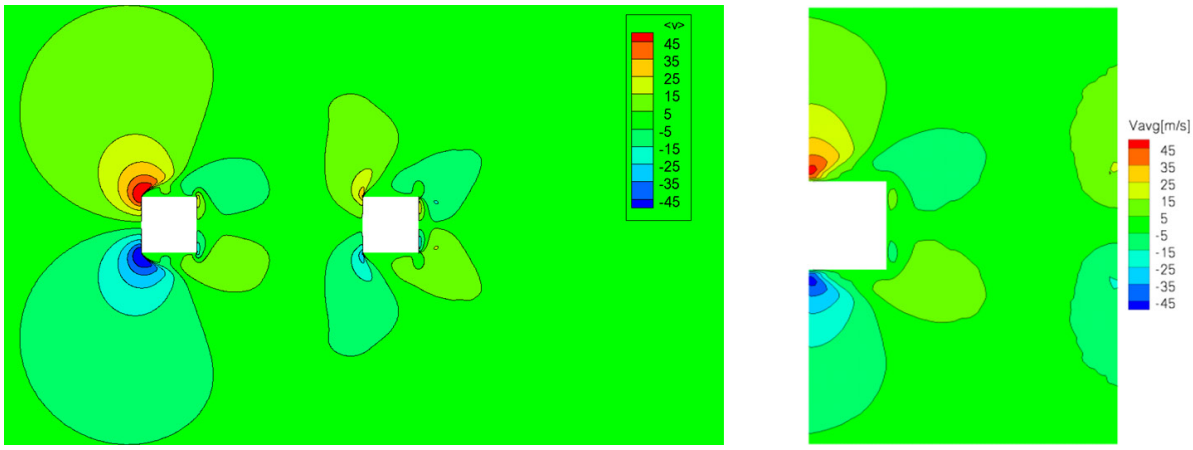


Рисунок 6.4: Осреднённое поле вертикальной компоненты скорости по результатам численного моделирования (слева) и экспериментальных измерений (справа)

Рис. 6.10 показывает сравнение осреднённого по времени распределения коэффициента поверхностного давления. Численные результаты показаны сплошной линией, экспериментальные данные отмечены точечными маркерами.

По оси абсцисс отложена поверхностная координата, идущая по линии вдоль поверхности от центра передней грани цилиндра (0м) до центра задней грани цилиндра (0.08м). Обозначение PS (pressure side) соответствует обходу поверхности сверху, обозначение SS (suction side) – обходу снизу. Линии для обхода сверху и снизу должны в идеальном случае совпадать, поскольку угол атаки равен 0, однако имеются некоторые расхождения в силу конечного периода интегрирования.

Угол атаки 10^o

Рисунки с 6.11 по 6.15 отображают осреднённые по времени различные поля течения и соответствующие им экспериментальные данные, полученные PIV измерениями.

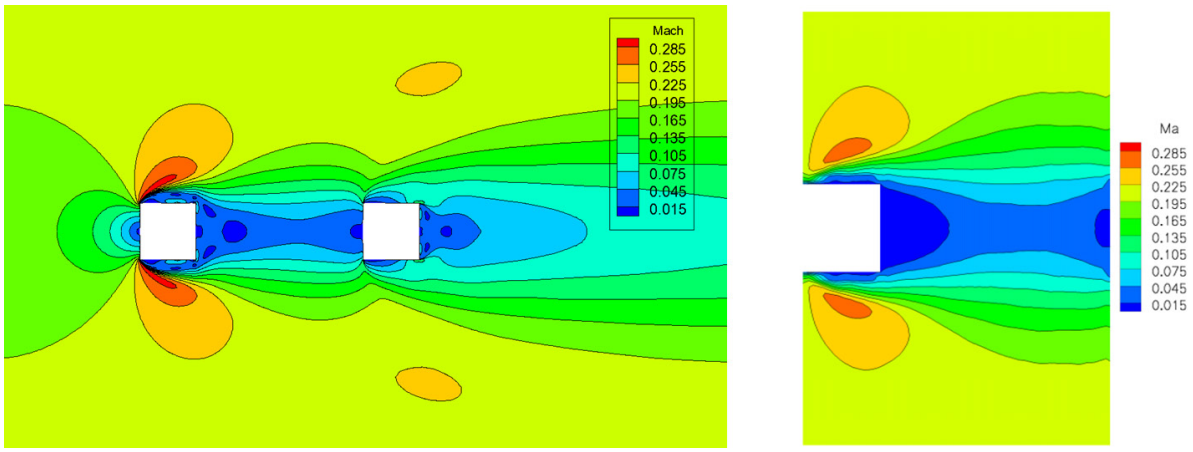


Рисунок 6.5: Осреднённое поле локального числа Маха по результатам численного моделирования (слева) и экспериментальных измерений (справа)

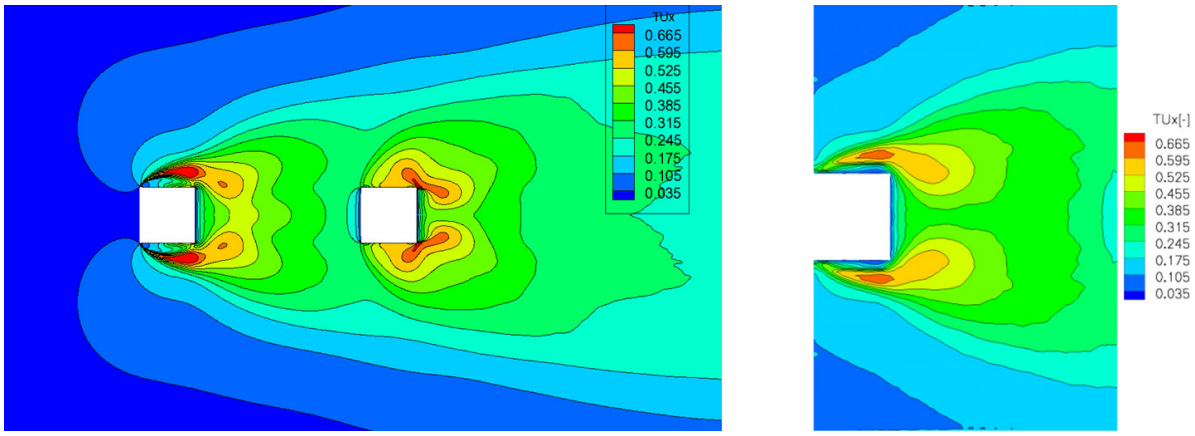


Рисунок 6.6: Осреднённое поле интенсивности турбулентности в продольном направлении ($TU_x = \sqrt{< u'u' >}/u_{ref}$) по результатам численного моделирования (слева) и экспериментальных измерений (справа)

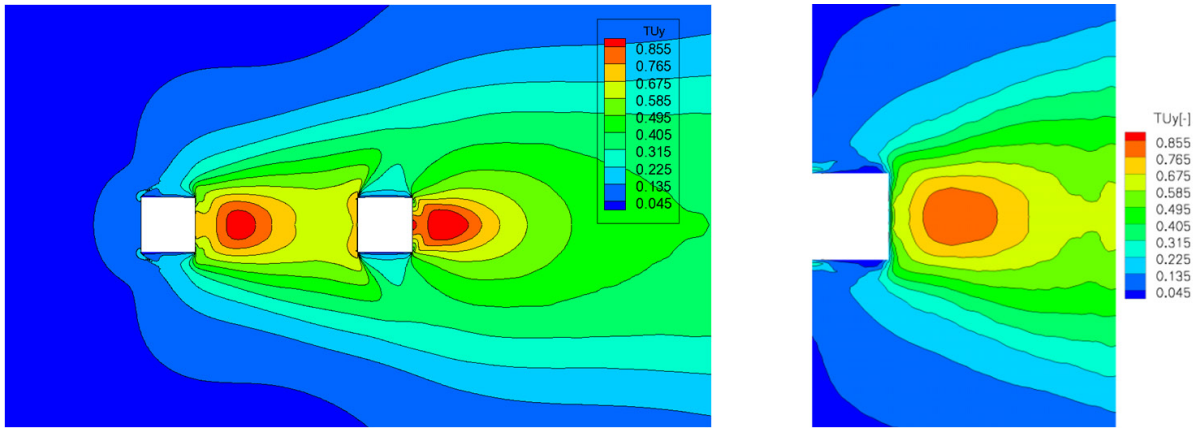


Рисунок 6.7: Осредненное поле интенсивности турбулентности в вертикальном направлении ($TU_y = \sqrt{< v'v' >}/u_{ref}$) по результатам численного моделирования (слева) и экспериментальных измерений (справа)

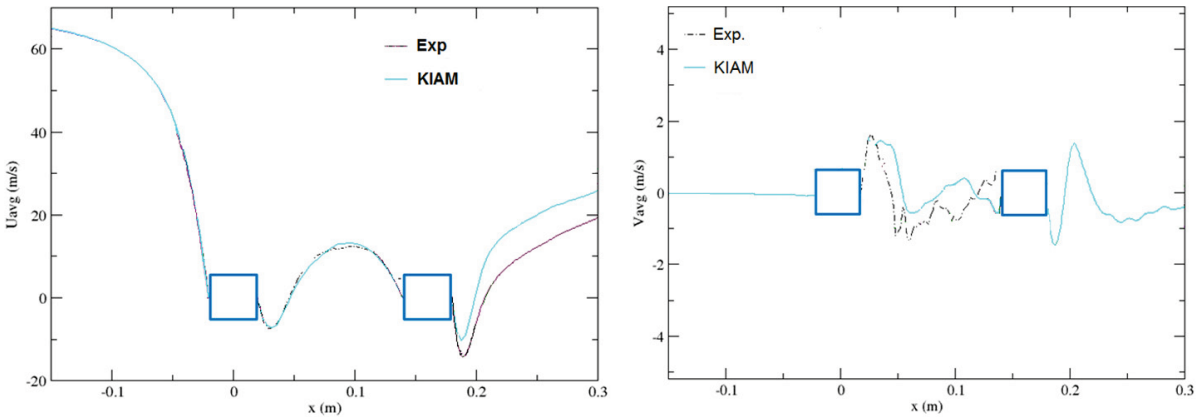


Рисунок 6.8: Сравнение осреднённых профилей продольной (слева) и вертикальной (справа) компоненты скорости

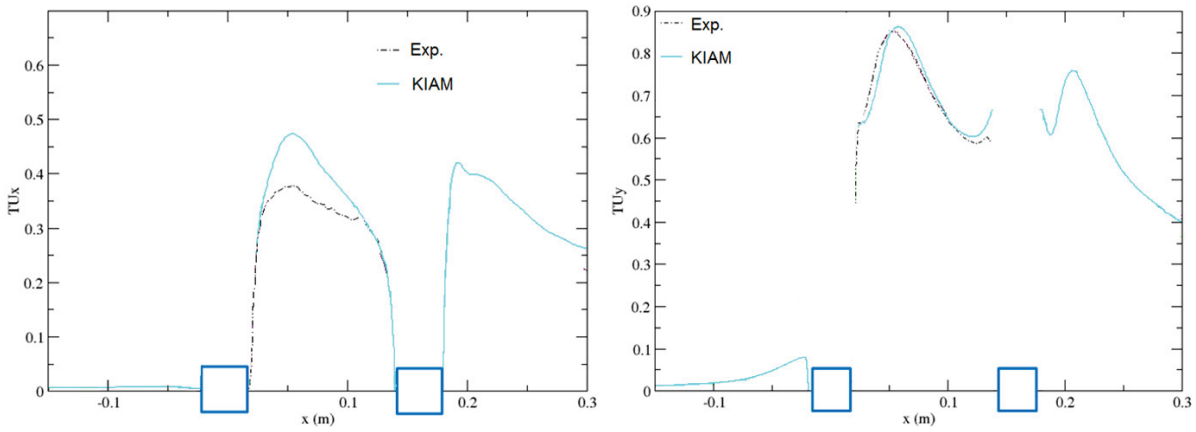


Рисунок 6.9: Сравнение осреднённых профилей интенсивности турбулентности в продольном (слева) и вертикальном (справа) направлении

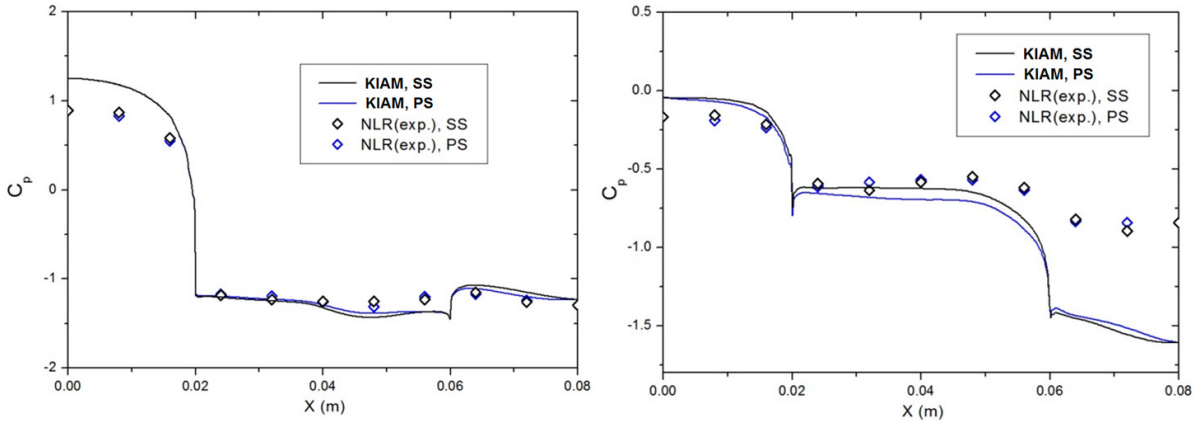


Рисунок 6.10: Сравнение распределения коэффициента давления по поверхности первого (слева) и второго (справа) цилиндров

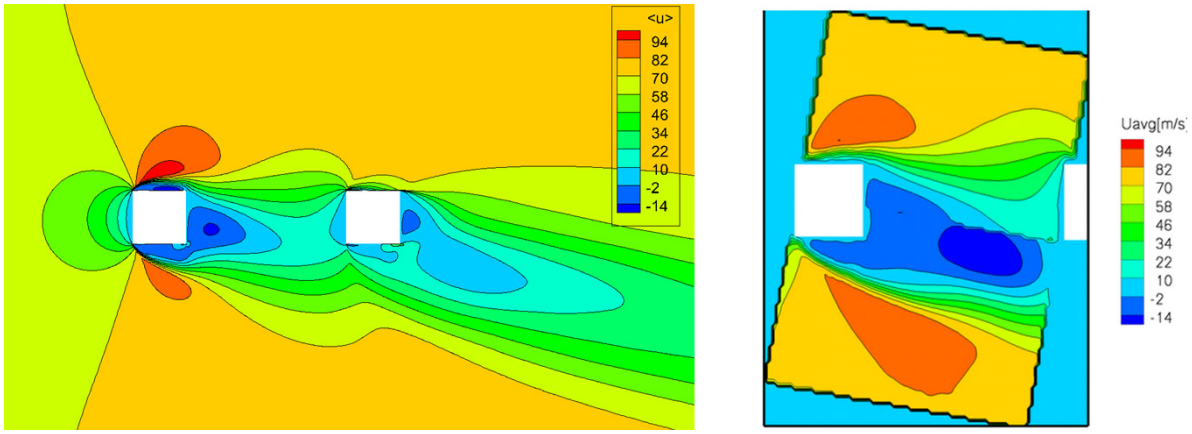


Рисунок 6.11: Осреднённое поле продольной компоненты скорости по результатам численного моделирования (слева) и экспериментальных измерений (справа)

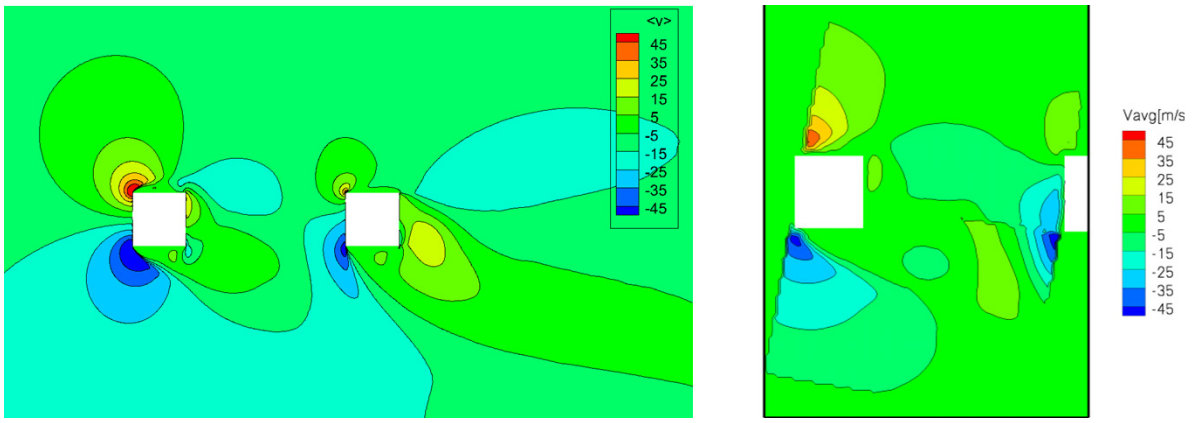


Рисунок 6.12: Осреднённое поле вертикальной компоненты скорости по результатам численного моделирования (слева) и экспериментальных измерений (справа)

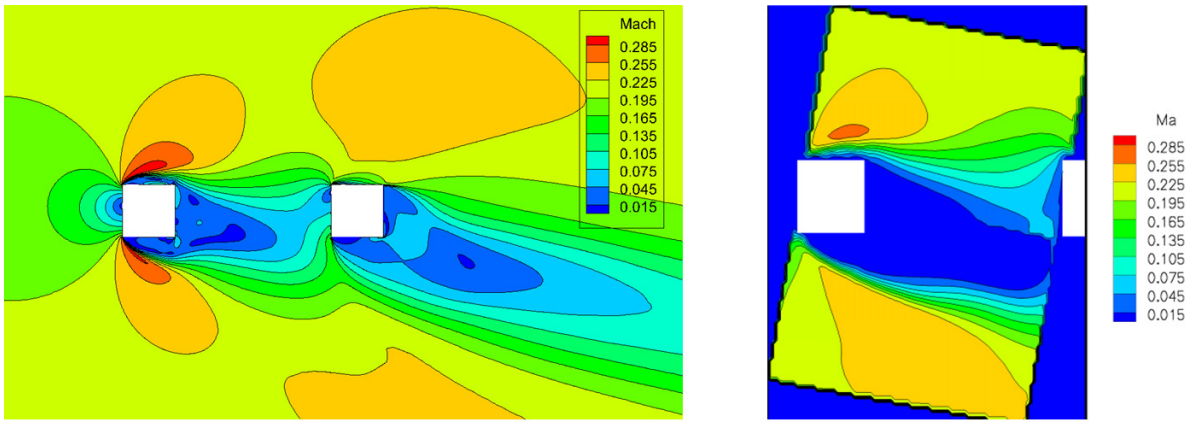


Рисунок 6.13: Осреднённое поле локального числа Маха по результатам численного моделирования (слева) и экспериментальных измерений (справа)

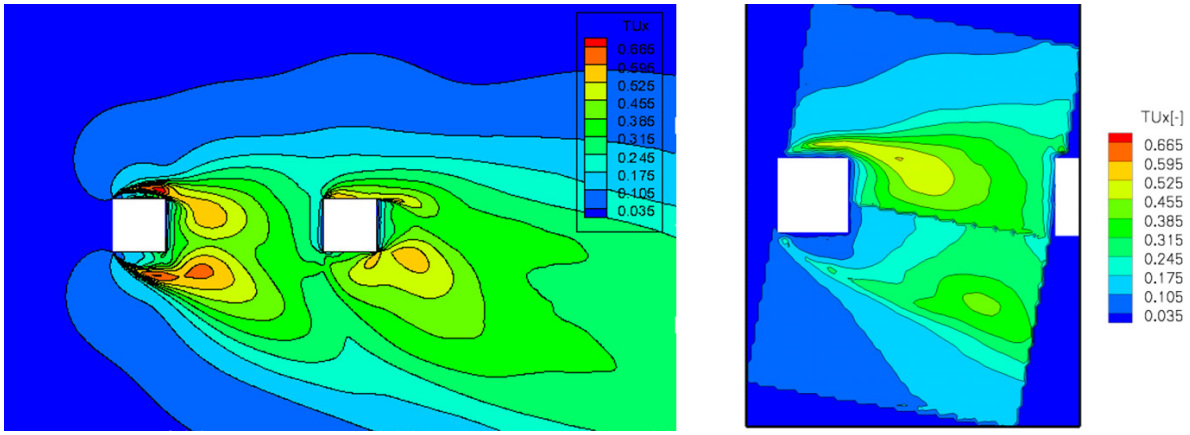


Рисунок 6.14: Осреднённое поле интенсивности турбулентности в продольном направлении ($TU_x = \sqrt{< u'u' >}/u_{ref}$) по результатам численного моделирования (слева) и экспериментальных измерений (справа)

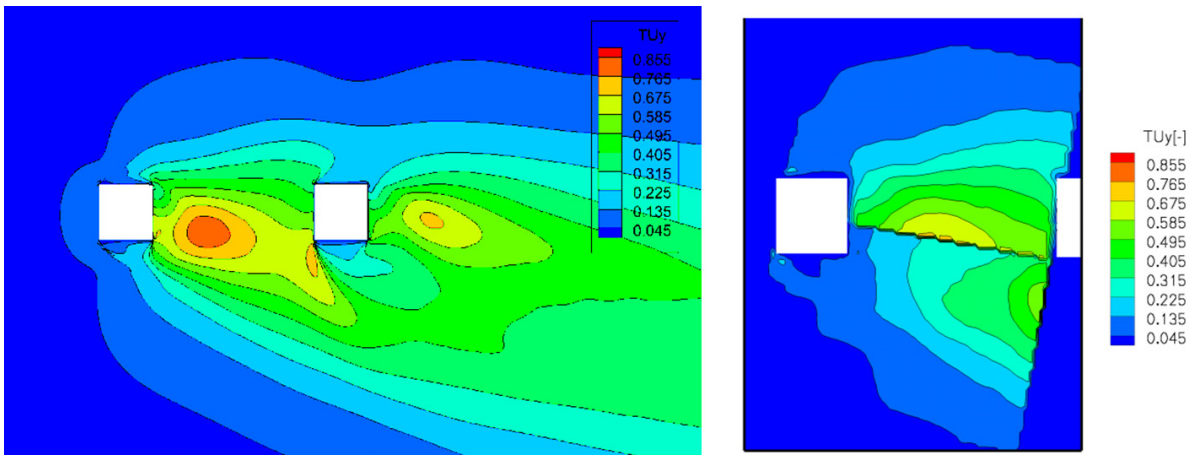


Рисунок 6.15: Осреднённое поле интенсивности турбулентности в вертикальном направлении ($TU_y = \sqrt{< v'v' >}/u_{ref}$) по результатам численного моделирования (слева) и экспериментальных измерений (справа)

Рис. 6.16 показывает сравнение осреднённого по времени распределения коэффициента поверхностного давления. Численные результаты показаны сплошной линией, экспериментальные данные отмечены точечными маркерами. По оси абсцисс отложена поверхностная координата, идущая по линии вдоль поверхности от центра передней грани цилиндра (0м) до центра задней грани цилиндра (0.08м). Обозначение PS соответствует обходу поверхности сверху, обозначение SS – обходу снизу.

6.1.3 Мгновенные поля течения

Моментальные картины течения позволяют оценить уровень турбулентности, разрешаемый численным моделированием. Рис. 6.17 показывает трёхмерную картину течения, на которой можно наблюдать образование крупномасштабных когерентных квазидвумерных вихревых структур, которые сочетаются с мелкомасштабными хаотическими трёхмерными турбулентными структурами. Вихри сносятся средним течением, сталкиваются со вторым цилиндром,

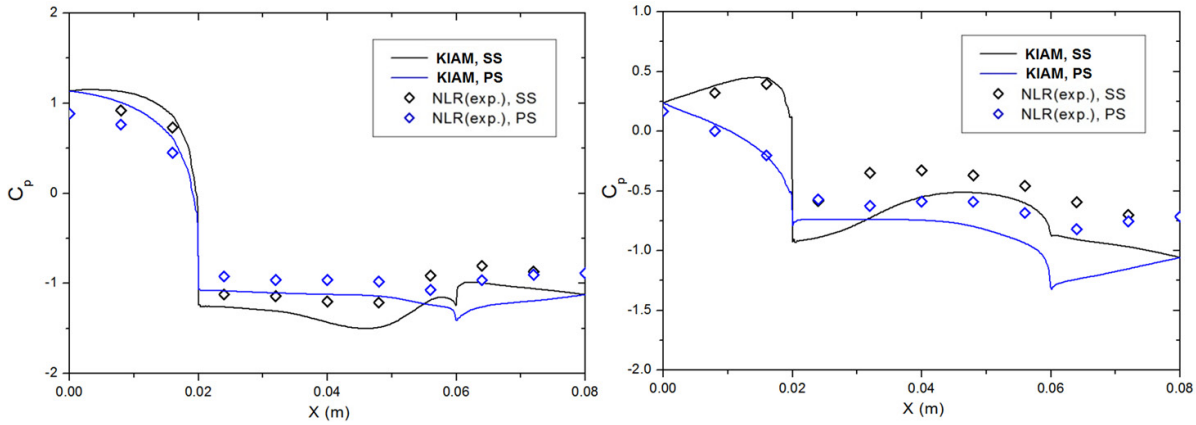


Рисунок 6.16: Сравнение распределения коэффициента давления по поверхности первого (слева) и второго (справа) цилиндров

разбиваются и медленно угасают далее по потоку. Разбиение вихревого следа от первого цилиндра о второй цилиндр хорошо видно на рис. 6.17 и 6.18.

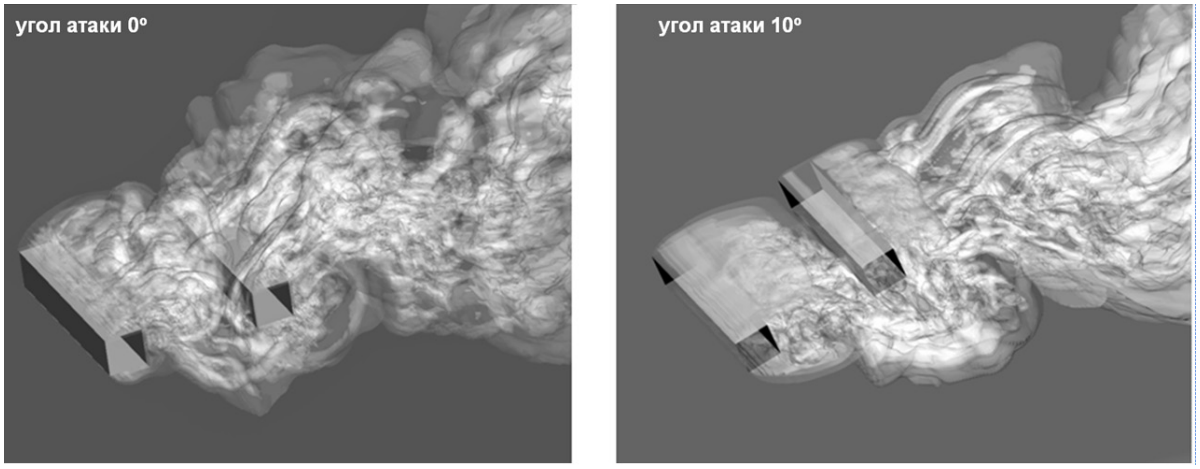


Рисунок 6.17: Моментальные картины течения. Показаны изоповерхности модуля скорости для угла атаки 0^o (слева) и 10^o (справа)

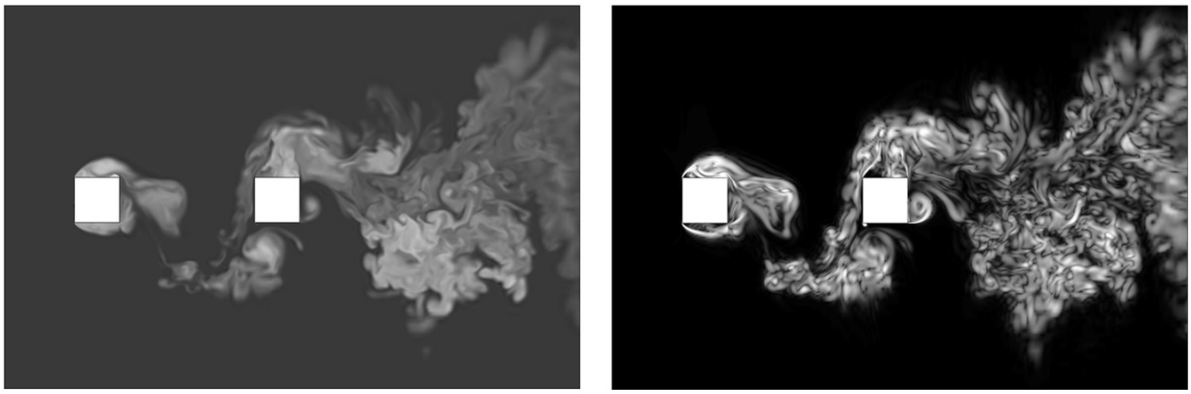


Рисунок 6.18: Моментальные картины течения в центральном сечении для угла атаки 0^o. Показаны поле плотности (слева) и поле модуля завихренности (справа)

6.1.4 Спектры пульсаций поверхностного давления

Графики спектральной плотности мощности пульсаций поверхностного давления получены с использованием окна Хэннинга с коэффициентом перекрытия 50%. Построение спектра имеет следующие параметры: 7 осреднений быстрого преобразования Фурье (БПФ), размером выборки 5436 с частотой дискретизации 18.5 Гц.

Угол атаки 0^o

На рис. 6.19 и 6.20 показано сравнение спектров, полученных по результатам расчёта и в экспериментальных измерениях.

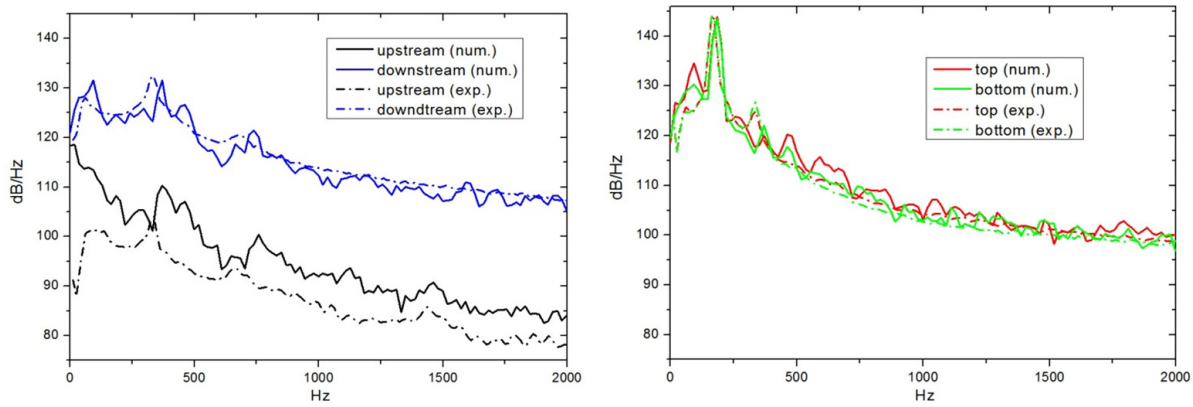


Рисунок 6.19: Сравнение спектров пульсаций давления в центрах передней и задней граней (слева) и в центрах верхней и нижней граней (справа) для первого цилиндра

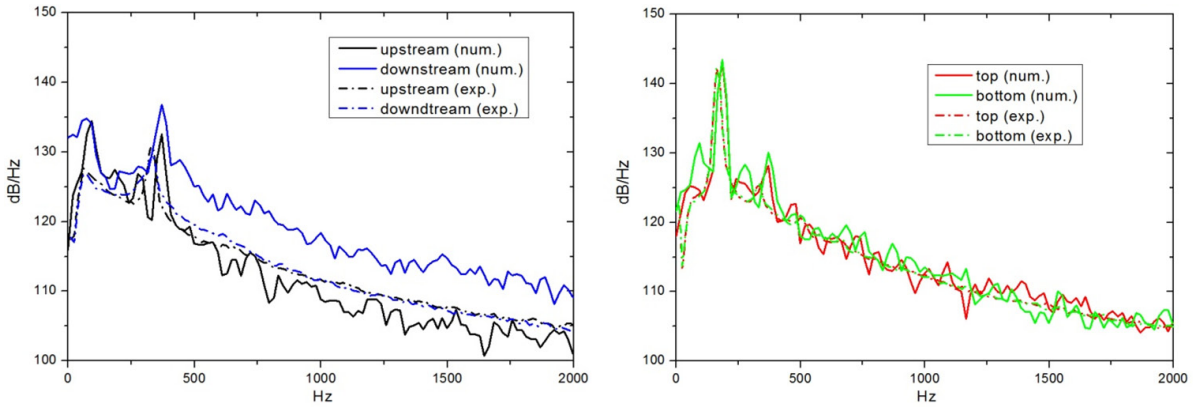


Рисунок 6.20: Сравнение спектров пульсаций давления в центрах передней и задней граней (слева) и в центрах верхней и нижней граней (справа) для второго цилиндра

Угол атаки 10^o

На рис. 6.21 и 6.22 показано сравнение спектров, полученных по результатам расчёта и в экспериментальных измерениях.

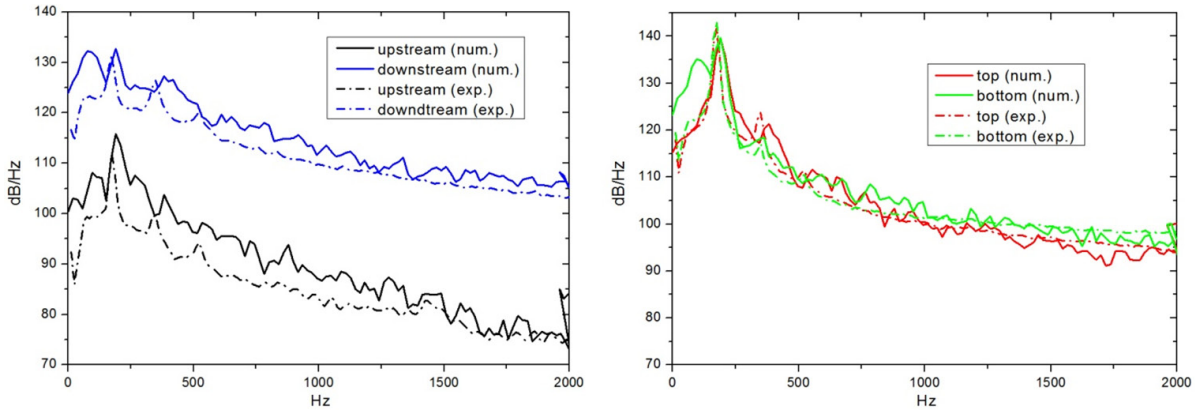


Рисунок 6.21: Сравнение спектров пульсаций давления в центрах передней и задней граней (слева) и в центрах верхней и нижней граней (справа) для первого цилиндра

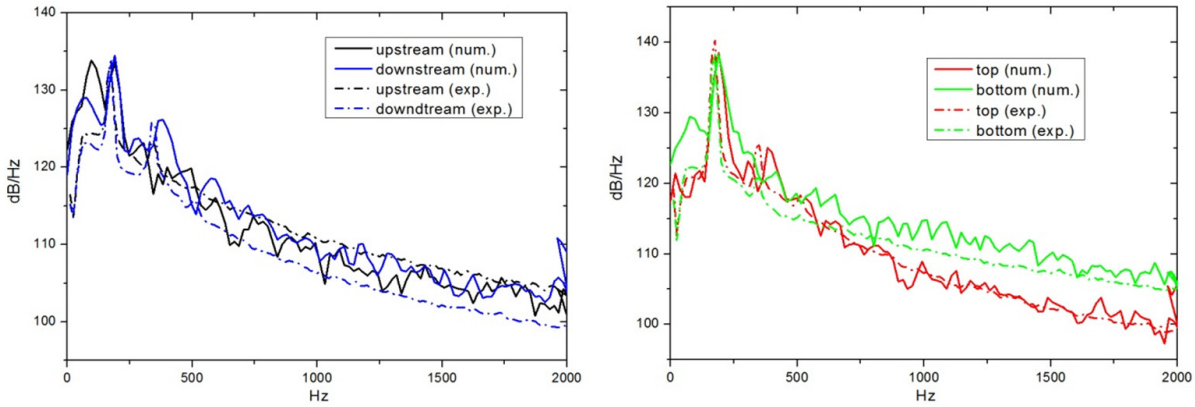


Рисунок 6.22: Сравнение спектров пульсаций давления в центрах передней и задней граней (слева) и в центрах верхней и нижней граней (справа) для второго цилиндра

6.1.5 Акустика в дальнем поле

В рамках эксперимента акустика в дальнем поле измерялась в 10-ти контрольных точках. По 5 контрольных точек расположено с верхней и с нижней стороны на окружности радиусом 2 метра с центром, находящимся на оси первого цилиндра. Контрольные точки, как показано на рис. 6.23, расположены под углами 105° (наблюдатель 1), 90° (наблюдатель 2), 75° (наблюдатель 3), 60° (наблюдатель 4), 45° (наблюдатель 5). Угол отсчитывается от направления течения (вверх и вниз соответственно). Все графики спектральной плотности мощности пульсаций давления в дальнем поле получены с использованием окна Хэннинга с коэффициентом перекрытия 50%. Построение спектра имеет следующие параметры: 5 осреднений БПФ, размер выборки 5436 с частотой дискретизации 18.5 Гц.

Угол атаки 0^o

Далее на графиках 6.24–6.26 показаны спектры для различных позиций наблюдателей. Поскольку течение симметрично, показаны результаты только для одной группы из 5 позиций.

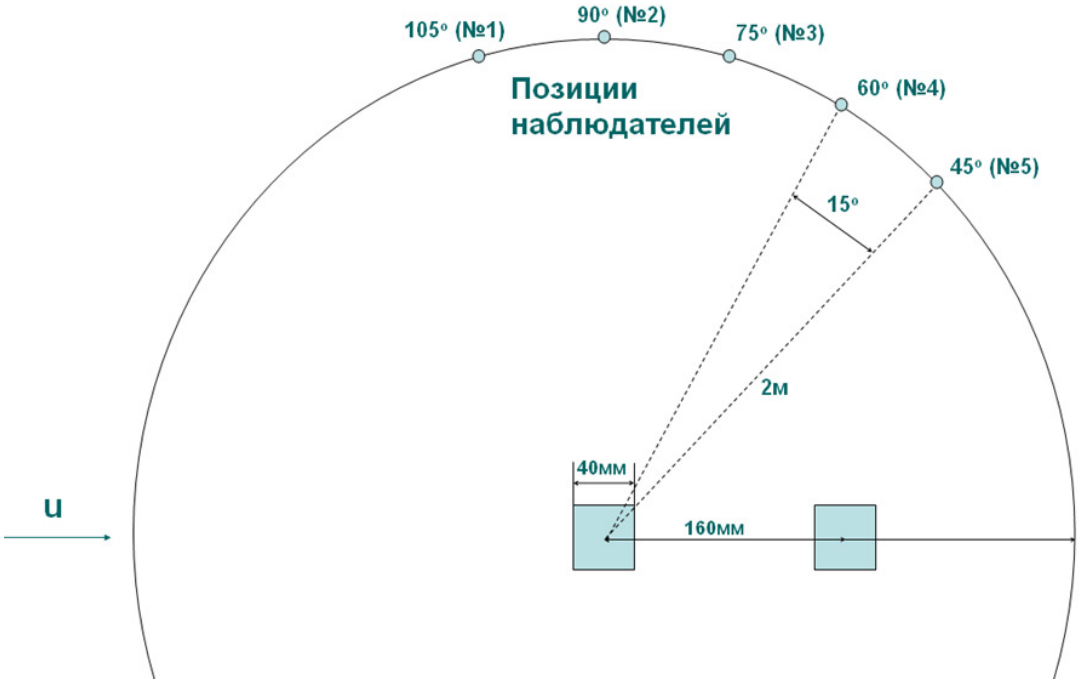


Рисунок 6.23: Позиции наблюдателей (микрофонов) для измерений в дальнем поле

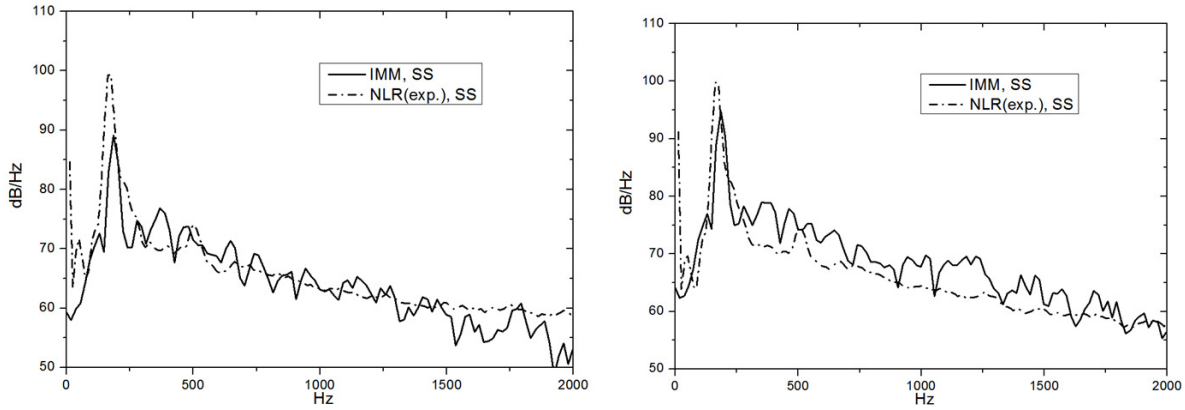


Рисунок 6.24: Сравнение спектров пульсаций давления в дальнем поле для позиций 1 (слева) и 2 (справа)

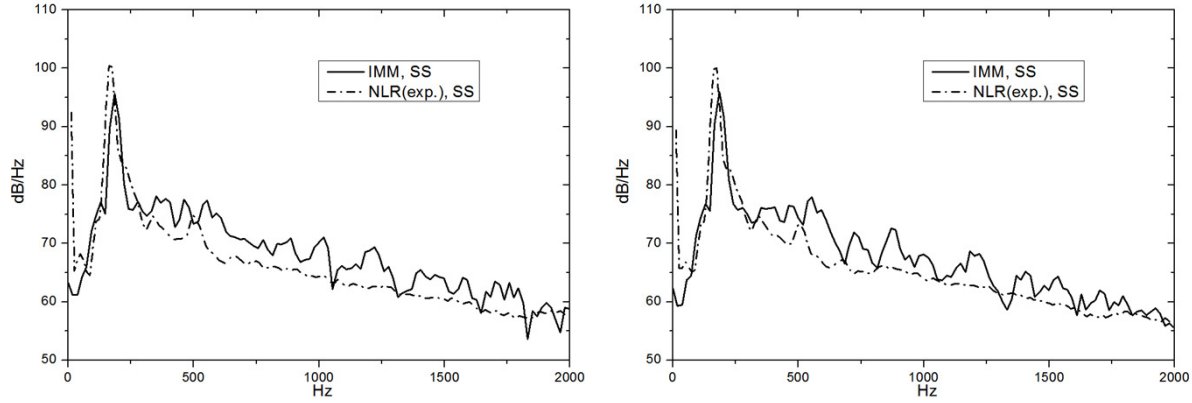


Рисунок 6.25: Сравнение спектров пульсаций давления в дальнем поле для позиций 3 (слева) и 4 (справа)

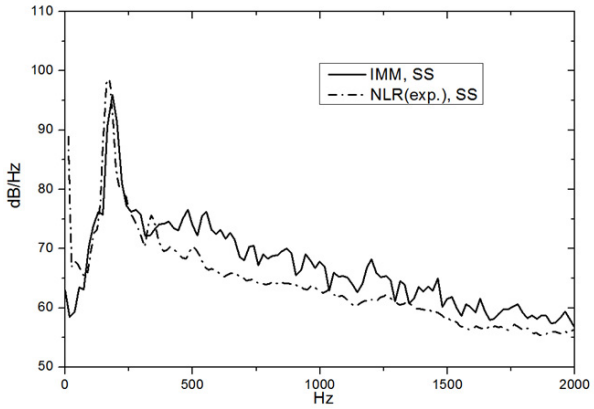


Рисунок 6.26: Сравнение спектров пульсаций давления в дальнем поле для позиции 5

Угол атаки 10^o

Далее на графиках 6.27–6.29 показаны спектры для различных позиций наблюдателей.

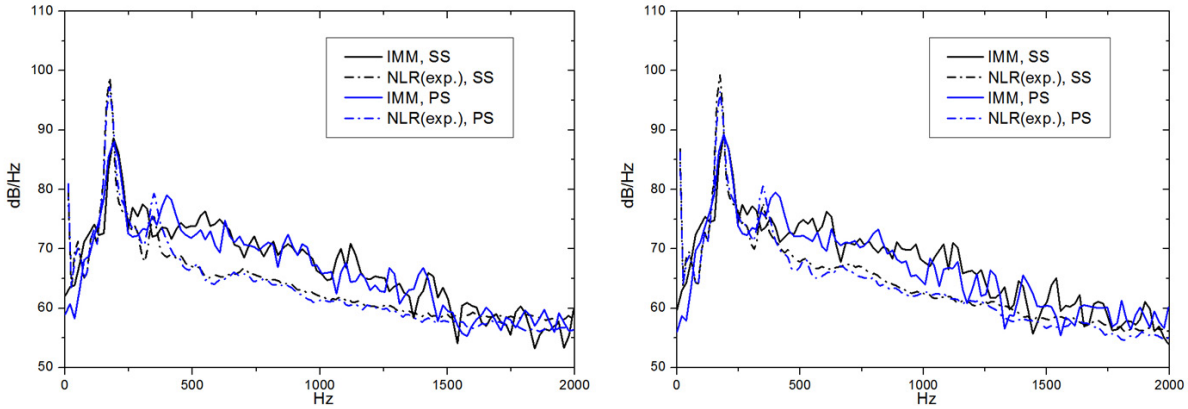


Рисунок 6.27: Сравнение спектров пульсаций давления в дальнем поле (для наблюдателей как сверху, так и снизу) для позиций 1 (слева) и 2 (справа)

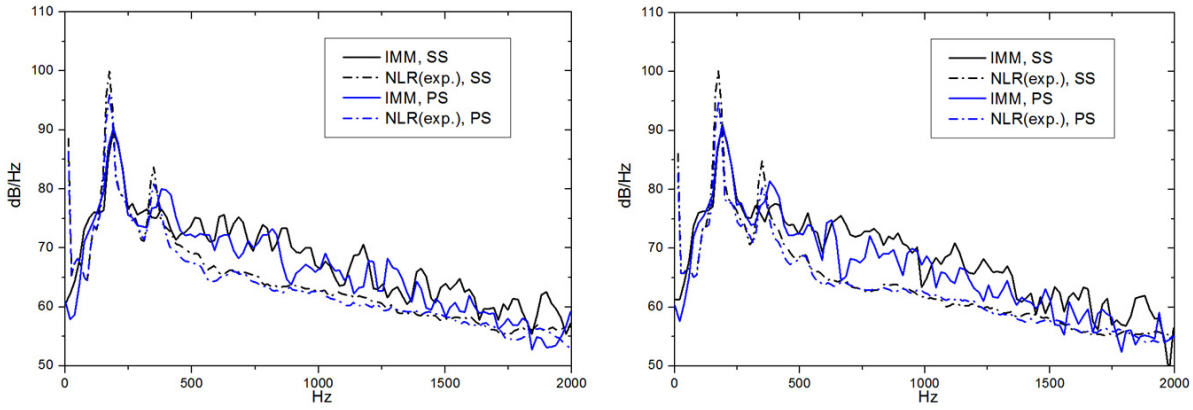


Рисунок 6.28: Сравнение спектров пульсаций давления в дальнем поле (для наблюдателей как сверху, так и снизу) для позиций 3 (слева) и 4 (справа)

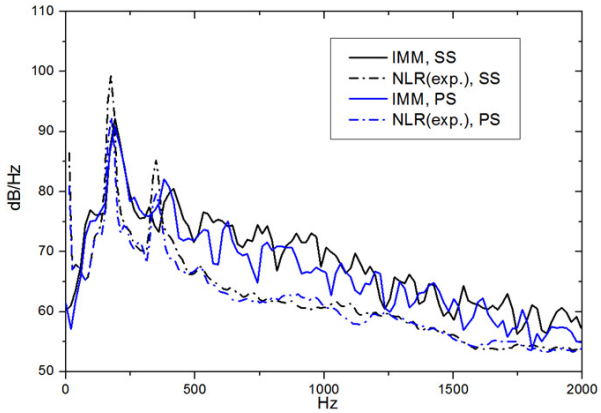


Рисунок 6.29: Сравнение спектров пульсаций давления в дальнем поле (для наблюдателей как сверху, так и снизу) для позиции 5

Осреднённые поля течения, полученные в численном эксперименте, показывают хорошее согласование с физическим экспериментом. Отмечены небольшие расхождения в профиле поверхностного давления на задней грани 2-го цилиндра. Также наблюдаются отличия в полях интенсивности турбулентности в зоне между цилиндрами для угла атаки 10^o. Основные пики на доминантной частоте в спектре пульсаций поверхностного давления в целом хорошо воспроизведены для обоих углов атаки, максимальное расхождение составило около 5 дБ по амплитуде и 20 Гц по частоте. Спектры для дальнего также хорошо воспроизводят основную частоту, результаты имеют расхождения того же порядка, что и в случае пульсаций поверхностного давления.

6.2 DNS течения в каверне с вертикальными разнонагретыми стенками с соотношением высоты к ширине 5:1

6.2.1 Постановка задачи

Естественная конвекция в каверне с вертикальными стенками разной температуры, DHC (differentially heated cavity), является известным типом задач, численному моделированию которых было посвящено много исследований. Этот тип задач соотносится с такими инженерными приложениями, как вентиляция помещений, охлаждение электронных устройств, конвекционные потоки в зданиях и т.д. Конвекционные течения в закрытых кавернах служили прототипами для разработки численных алгоритмов, примеры которых могут быть найдены, например, в [146–149].

Данный расчёт [150] на сетке из 35 млн узлов схемой 4-го порядка моделирует каверну с соотношением сторон 5:1, число Рэлея $Ra = 4.5 \times 10^{10}$. Такая постановка исследуется с середины 80-х, в том числе экспериментально, и достаточно широко используется для валидации моделей. Схема данной DHC конфигурации показана на рис. 6.30 (слева). Точное предсказание структуры течения и теплообмена в такой конфигурации представляет большой

интерес и, несмотря на большие усилия, посвящённые этой проблеме (см., например, [151–155]), точное моделирование турбулентности в этой конфигурации всё ещё сопряжено с большими сложностями. В основном проблема состоит в сложном сочетании картин течения (см. рис. 6.30 справа): пограничный слой остаётся ламинарным до точки, где волны, двигающиеся вниз по потоку, достигают достаточной амплитуды, чтобы разрушить погранслойное течение и привести к формированию крупных нестационарных вихревых структур. Перемешивание, вызываемое этими вихрями, приводит к почти изотермическим горячей верхней и холодной нижней зонам и провоцирует большой перепад температуры в небольшой зоне центре каверны. Поэтому точное предсказание точки перехода является критическим для получения структуры течения в каверне. В то же время, чувствительность пограничного слоя к внешним возмущениям делает очень сложным правильное предсказание точки перехода. Всё это делает конфигурацию DHC очень сложной для моделей турбулентности, поскольку присутствуют и взаимодействуют между собой области с совершенно разными режимами течения.

Данная конфигурация имеет следующие параметры: число Праднтля $Pr = 0.71$ (воздух), соотношение высоты к ширине $A_z = 5$, число Рэлея $Ra = 4.5 \times 10^{10}$ (по высоте каверны L_z). Эта конфигурация воспроизводит один из первых экспериментов [156] в середине 80-х. Эти результаты были широко использованы для валидации моделей турбулентности (см [154, 155, 157–162], например). В этой связи, получение подробного эталонного численного решения для данной конфигурации представляет большой интерес. С этой целью выполнено прямое численное моделирование данной конфигурации с использованием программного комплекса STG-CFD&HT, представленного в предыдущей главе.

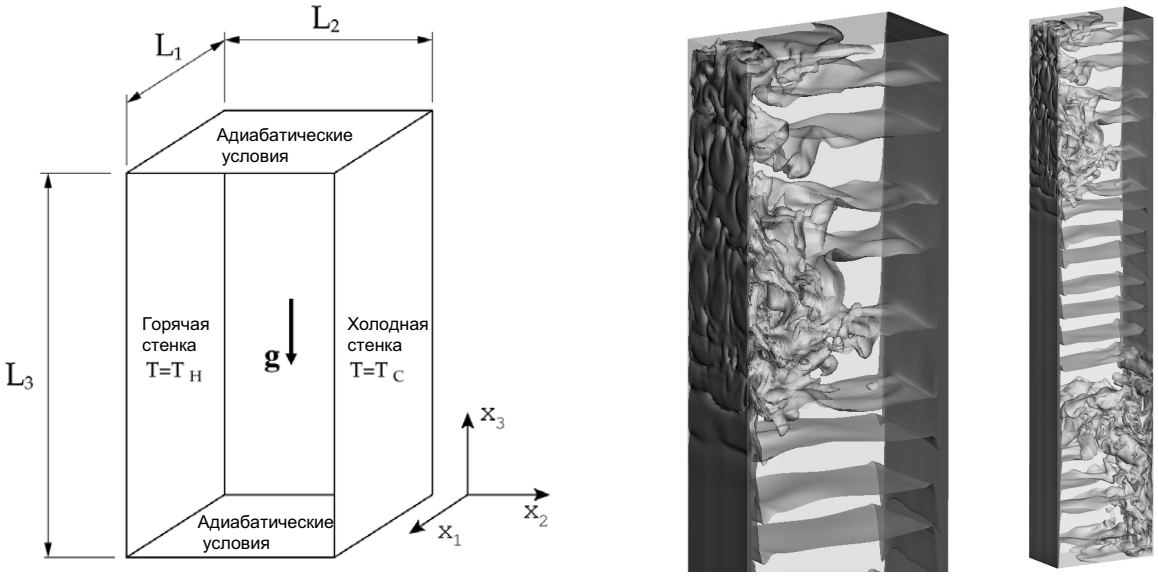


Рисунок 6.30: Схема DHC конфигурации (слева) и мгновенная картина течения (изоповерхности температуры) из расчёта(справа)

6.2.2 Верификация расчёта

Результаты осредняются по статистически инвариантным преобразованиям: по времени, по периодической оси x и по пространственной симметрии. Поскольку в расчёте не используется модель турбулентности подсеточного масштаба, разрешение сетки должно быть достаточным для всех релевантных масштабов течения. Размер расчётной области по периодическому направлению L_x должен быть достаточно большим, чтобы не влиять на численное решение. Период интегрирования по времени должен быть достаточно длинным и начинаться после того, как течение вышло на статистически однородный режим.

Согласно технологии, представленной в первой главе, были выполнены предварительные расчёты на последовательности сгущающихся сеток (см. таблицу 6.1). Предварительный расчёт на сетке Mesh2 был выполнен для достаточно широкой расчётной области по периодическому направлению. Соотношение глубины (по периодике) к ширине было выбрано $A_x = 0.2$. На рис. 6.31 показан анализ двухточечной корреляции для компоненты скорости, R_{uu}, в пяти различных (y,z) позициях. Из полученных результатов для всех контрольных точек видно, что $A_x = 0.1$ достаточно для того, чтобы турбулентные флуктуации были некоррелированы на расстоянии половины периода, $A_x/2$.

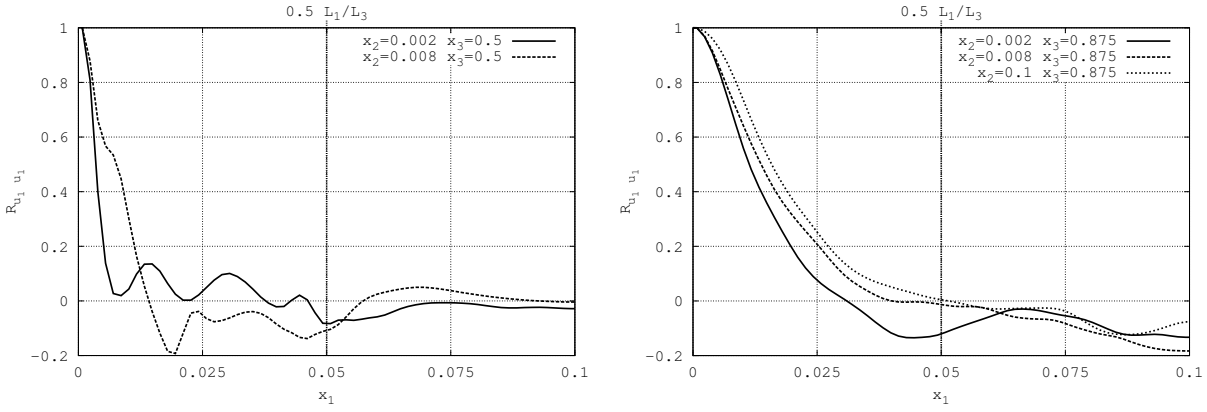


Рисунок 6.31: Двухточечная корреляция поперечной компоненты скорости u в 5 позициях

Аналогичные результаты были получены для других (y,z) позиций и для других переменных. Полученные на сетке Mesh2 средние поля течения показали, что переход вертикального пограничного слоя может происходить в точках дальше по потоку, чем наблюдалось в эксперименте [163] и численных исследованиях [154, 155]. Точное предсказание этой точки является ключевым моментом для корректного определения конфигурации всего течения в каверне (более подробно об этом в [164]). Чтобы подтвердить результаты, полученные на сетке Mesh2, было выполнено DNS на подробной сетке $128 \times 318 \times 862$ (Mesh1) с $A_x = 0.1$ (см. таблицу 6.1). Использовалась численная схема четвёртого порядка [133]. Шаг сетки по периодической оси x постоянный, а по направлениям, ортогональным твёрдым стенкам, координаты узлов сетки задаются следующей функцией сгущения сетки к краям отрезка (задается длина отрезка L, критерий сгущения γ и число точек на отрезке N):

Сетка	N_x	N_y	N_z	A_x	γ_y	γ_z	$(\Delta y)_{min}$	$(\Delta y)^+_{min}$	Δt	Общее время	Время осреднения
Mesh1	128	318	862	0.1	2.0	0.0	4.67×10^{-5}	$\lesssim 0.4$	2.85×10^{-4}	420	215
Mesh2	128	160	432	0.2	2.0	0.0	9.33×10^{-5}	$\lesssim 0.8$	1.38×10^{-3}	425	180
Mesh3	32	80	216	0.1	2.0	0.0	1.87×10^{-4}	$\lesssim 1.6$	4.57×10^{-3}	800	400

Таблица 6.1: Физические и численные параметры расчётов

$$x_k = x_0 + \frac{L}{2}\left(1 + \frac{tanh\left\{\gamma\left(2\left(k-1\right)/N-1\right)\right\}}{tanh\gamma}\right), \quad k = 1, \ldots, N+1. \tag{6.1}$$

Пространственное разрешение по этим двум направлением определялось путём систематической процедуры на основе последовательного сгущения сетки (см. [128]). Параметры концентрации сетки γ_y и γ_z выбирались так, чтобы минимизировать градиенты течения в вычислительном пространстве для набора репрезентативных моментальных картин течения. Естественно, наиболее чувствительной к разрешению сетки областью является зона около изотермических стенок. В расчёте на сетки (Mesh1) первый узел сетки расположен в $y^+ \lesssim 0.4$ (см. таблицу 6.1). Сдвиговая скорость u_τ вычисляется по локальному касательному напряжению на стенке.

Шаг сетки по периодической оси должен обеспечивать хорошее разрешение мелких масштабов течения. Для проверки достаточности сеточного разрешения $\Delta x = A_x/N_x$ использовался одномерный энергетический спектр в нескольких контрольных точках.

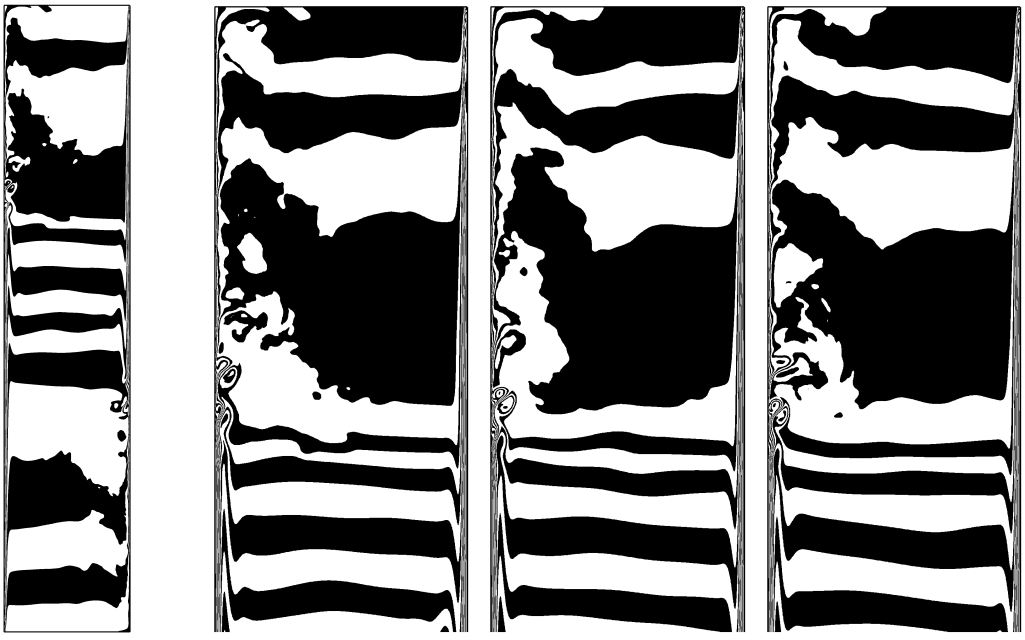


Рисунок 6.32: Моментальное поле температуры в сечении, ортогональном периодической оси. Изотермы однородно распределены между значениями -0.5 и 0.5, соответствующими температуре холодной и горячей стенки. Слева: общий вид каверны. Справа: временная последовательность картин течения в верхней части каверны

6.2.3 Результаты расчёта

Моментальные поля температуры, показанные на рис. 6.32, иллюстрируют характерную сложность течения в данной конфигурации. А именно, погранслой остаётся ламинарным до точки, где волны, двигающиеся вниз по потоку, достигают достаточной амплитуды, чтобы разрушить погранслойное течение и привести к формированию крупных нестационарных вихревых структур. Турбулентные возмущения существенны в погранслое только ниже по потоку после точки перехода по оси z (см. рис. 6.33 справа).

Эту точку перехода достаточно сложно правильно предсказать, в частности, из-за высокой чувствительности погранслоя к внешним возмущениям (см. [154], например). По этой точке в литературе присутствуют заметные расхождения. Результаты эксперимента [163] показывают, что точка перехода на горячей стенке находится около $z \approx 0.2$. Однако, недавние численные исследования [154, 155] показывают, что полученные численные решения сильно зависят от параметров сетки и модели турбулентности. Тем не менее, по этим результатам точка перехода получается намного выше по потоку, чем получено в DNS расчёте на подробной сетке Mesh1.

Осредненные поля температуры и линии тока осредненного течения показаны на рис. 6.33 вместе с турбулентной статистикой течения. Из этих результатов чётко видна ключевая роль определения точки перехода на вертикальных погранслоях. Точка перехода z^{Tr} соответствует позиции $\sigma\left(Nu\right)_{max}$ по оси z на вертикальной горячей стенке (см. рис. 6.34 справа). Из этого следует, что $z^{Tr} \approx 0.674$ для сетки Mesh1 (0.697 для Mesh2), существенно ниже по потоку, чем наблюдалось в эксперименте и предыдущих численных исследованиях. Аналогичные расхождения наблюдались и для DHC с соотношением сторон 4 при сравнении результатов RANS расчётов [155] с DNS результатами [128]. Аналогично, для соотношения сторон 4, экспериментальные исследования [165] были выполнены с Ra до 1.2×10^{11}. Для самого высокого Ra точка перехода в эксперименте $z^{Tr} \approx 0.3$ (см. рис. 11 в [165]), в то время как в DNS для $Ra = 10^{11}$ [166] точка перехода была гораздо ниже по потоку ($z^{Tr} \approx 0.61$, см. рис. 1 в [166]). Такое расхождение не может быть связано со сравнительно небольшим отличием в числе Рэлея. Причина такого расхождения до сих пор полностью не выяснена и требует дальнейших исследований. В случае эксперимента [156], расхождение можно объяснить эффектами за пределами аппроксимации Буссинеска. Считается, что для комнатной температуры аппроксимация Буссинеска работает до перепада температуры $\Delta T \lesssim 20K$ [165, 167]. Однако, перепад в эксперименте был намного больше, $\Delta T \approx 46K$ [156]. Эти эффекты с учётом теплового излучения и сопряжённого теплообмена следует учитывать при прямом сравнении с экспериментом.

Сравнение тепловой стратификации в центре каверны между численными и экспериментальными результатами (см. детальный обзор в [168]) для широкого диапазона отношений высоты к ширине показывает совершенно различные результаты. Из экспериментов следует безразмерная стратификация около 0.5, а из расчётов следуют значения, близкие к 1. Недавно было показано, что причиной этих расхождений являются эффекты теплового

излучения передней и задней стенок [169]. В данном расчёте это значение также близко к единице (см. рис. 6.34 слева и таблицу 6.2).

Осреднённое локальное число Нуссельта и его среднеквадратическое отклонение для сеток Mesh1 и Mesh2 показано на рис. 6.34 справа. Как и ожидалось, флуктуации в погранслое значительны только снизу по потоку, а верхняя часть остаётся ламинарной. Общее число Нуссельта для сетки Mesh1 равно 154.5 (155.7 для Mesh2). Это значение хорошо согласуется со значением 154.8, которое предсказывает степенное соотношение, предложенное в [166]. Более подробно результаты расчёта представлены в [150].

Case	$\overline{Nu}$	μ_{Nu}	z^{Tr}		C	f_{BV}
Mesh1	154.5	0.156	0.674	0.148	1.002	0.133
Mesh2	155.7	0.155	0.697	0.144	1.038	0.136

Таблица 6.2: Число Нуссельта и соотношения

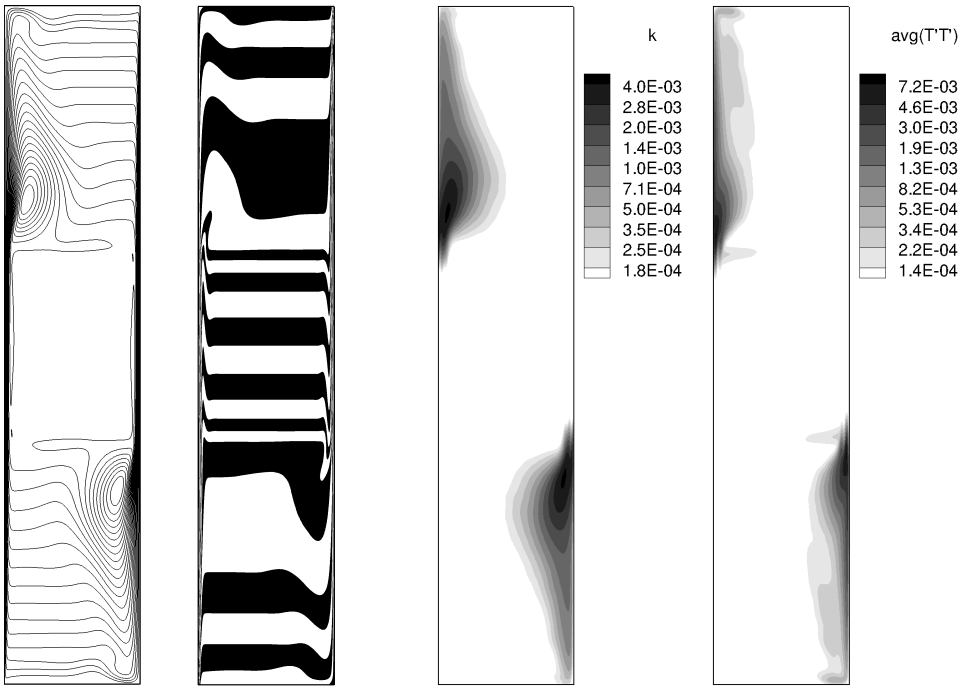


Рисунок 6.33: Осреднённые решения слева направо: линии тока, $\overline{\theta}$, кинетическая энергия турбулентности и $\overline{\theta'\theta'}$ (изотермы равномерно распределены между -0.5 и0.5)

6.3 DNS падающей на плоскую пластину струи

6.3.1 Постановка задачи

Падающие струи часто встречаются в промышленных приложениях, связанных с обогревом, охлаждением, процессами высыхания, охлаждением лопаток турбин, электронных компонентов. Особенностью такого течения является более локализованный перенос тепла и массы, по

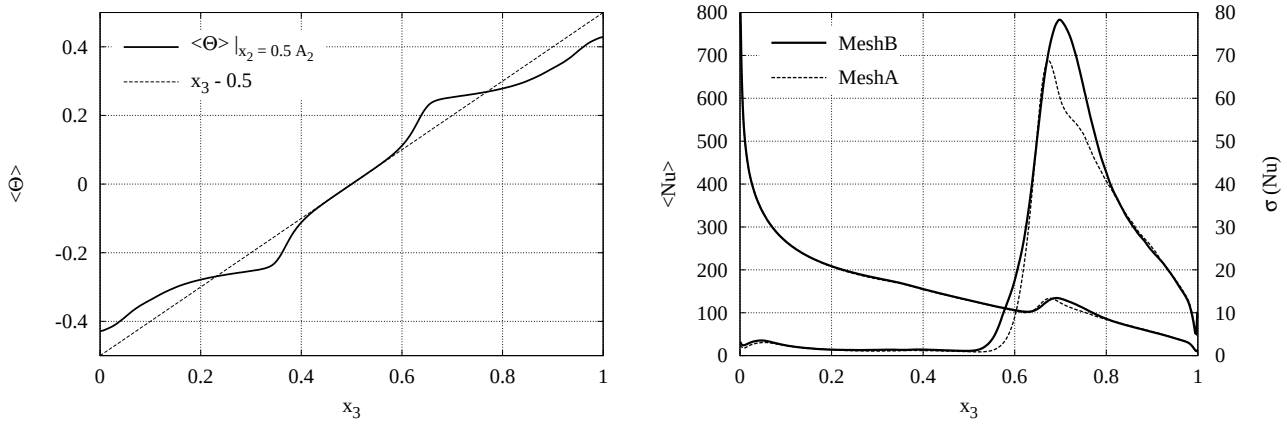


Рисунок 6.34: Осреднённый вертикальный профиль температуры по центру каверны (слева), распределение локального числа Нуссельта и его среднеквадратическое отклонение для сеток Mesh1 и Mesh2 (справа)

сравнению с течением параллельно твёрдой поверхности. Большинство приложений имеют турбулентный режим течения со сложными характеристиками, несмотря на простоту геометрии. Точное предсказание переноса тепла в случае падающей струи является важным во многих промышленных приложениях.

Турбулентная плоская падающая струя достаточно много исследовалась в различных конфигурациях. Экспериментальные исследования динамики течения включают Ashforth-Frost и др. [170] – измерение скорости и турбулентных характеристик для щелевой струи с числом Рейнольдса 20000 и двумя пропорциями 4 и 9.2. Zhe и Modi [171] также исследовали поле скорости для различных чисел Рейнольдса и пропорций. Задача активно исследовалась численно с использованием RANS и LES подходов. В [42] представлен более подробный обзор исследований по этой конфигурации.

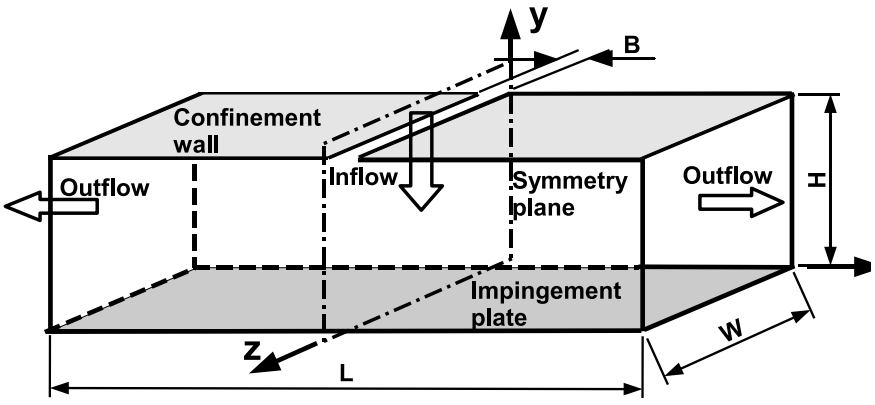


Рисунок 6.35: Схема расчётной области

В данном расчёте на сетке из 104 млн узлов схемой 4-го порядка моделируется одиночная турбулентная воздушная струя, падающая вниз из прямоугольной щели под прямым углом на плоскую пластину. Течение ограничено двумя параллельными плоскостями, как показано на

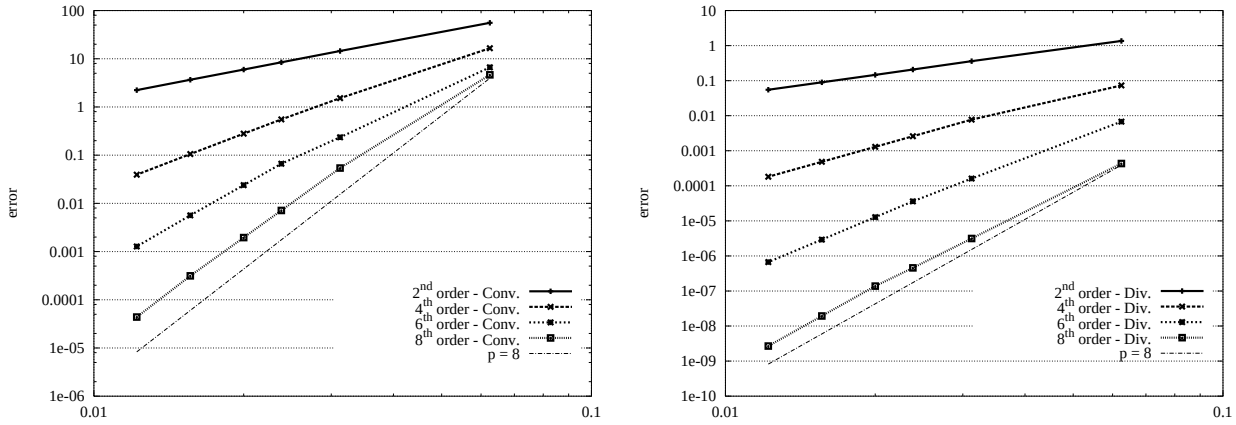


Рисунок 6.36: Численные ошибки для операторов конвекции (слева) и дивергенции (справа) для схем 2-го, 4-го, 6-го и 8-го порядка

рис. 6.35. С обоих сторон от струи образуются зоны рециркуляции около верхней пластины. Число Рейнольдса равно 20000, обезразмеривание выполнено по скорости входящего течения и по ширине щели. Пропорция H/B, то есть соотношение расстояния между пластинами и ширины щели, равна 4. Число Прандтля равно 0.71 (воздух). Результаты данного DNS расчёта подробно представлены в [42], где в сравнении с результатами DNS исследовались и оценивались множество RANS моделей турбулентности. Расчёт выполнялся на суперкомпьютере MareNostrum, BSC, Испания.

6.3.2 Верификация расчёта

Расчёт выполнен с помощью программного комплекса STG-CFD&HT, представленного в 5-й главе. Верификация программной реализации и дискретных операторов выполнена с помощью метода MMS (Method of Manufactured Solutions) [172], и с помощью тестирования на модельных задачах. На рис. 6.36 показаны тесты на сходимость для конвективного оператора и оператора дивергенции для численных схем от 2-го до 8-го порядка аппроксимации. Результаты показывают хорошее согласование с ожидаемым порядком сходимости. В качестве дополнительного критерия используется проверка точного выполнения баланса глобальной кинетической энергии.

Результаты осредняются по трем статистически инвариантным преобразованиям: по времени, по периодической оси z и по пространственной симметрии. Поскольку в расчёте не используется модель турбулентности подсеточного масштаба, разрешение сетки должно быть достаточным для разрешения всех релевантных масштабы течения.

Кроме того, размер расчётной области по периодическому направлению L_z и по направлению течения между пластинами L_x должны быть достаточно большими, чтобы не влиять на численное решение. Наконец, период интегрирования по времени должен быть достаточно длинным и начинаться после того, как течение вышло на статистически однородный режим.

Re_B	L/B	H/B	W/B	N_x	N_y	N_z	$\Delta x_{min}/B$	$\Delta y_{min}/B$	$\delta t_{avg}/t_{ref}$	t/t_{ref}
20000	100	4	2π	1680	368	168	9.23×10^{-4}	5.28×10^{-3}	3.63×10^{-3}	~ 3700

Таблица 6.3: Физические и численные параметры

Кроме того, необходимо оценить влияние входного профиля на результаты расчёта. Физические и численные параметры расчёта даны в таблице 6.3.

Расчётная область должна быть достаточно большой в направлении течения вдоль пластин, чтобы выходные условия не влияли на численное решение в области интереса. Для определения L_x был выполнен расчёт на более грубой сетке, согласно технологии, представленной в первой главе, с заведомо достаточно большим размером по этому направлению. Для определения места, где можно ставить выходные условия, анализировалось среднее поле течения и проверялось, где ортогональная пластине компонента скорости $\overline{v}$ пропадает (см. рис. 6.37), то есть, где линии тока становятся параллельными стенкам (см. рис. 6.38). Из этого критерия следует, что выходные граничные условия должны отстоять от центра струи на примерно $x/B = \pm 50$ (в 50 раз больше ширины щели).

Сразу следует отметить, что в предыдущих численных [173, 174] и экспериментальных исследованиях [170, 171] выход располагался на расстоянии $x/B = \pm 10 \sim 15$, а, следовательно, не воспроизводилась достаточно хорошо обратное течение у верхней пластины, которое является одной из основных особенностей данной конфигурации.

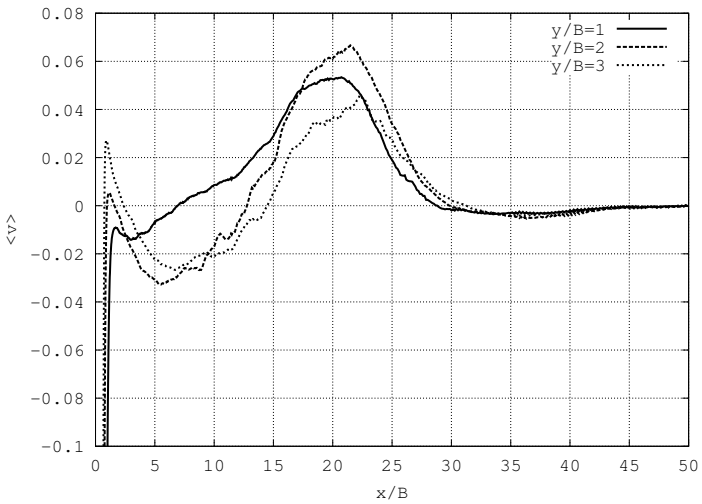


Рисунок 6.37: Осреднённый профиль компоненты скорости, ортогональной пластине, вдоль оси y на уровнях $y/B = 1$, 2 и 3

Глубина расчётной области по периодическому z направлению определялась исходя из требования, что турбулентные флуктуации не коррелированы на расстоянии половины периода. Для проверки этого условия использовалась двухточечная корреляция, $R_{\phi\phi}(x,y,r_z) = \overline{\phi'(x,y,z)\phi'(x,y,z+r_z)}/\overline{\phi'^2(x,y,z)}$, в различных (x,y)-позициях. На рис. 6.39 показана двухточечная корреляция для компоненты скорости в периодическом направлении, R_{ww}, в 6 контрольных точках по (x,y). Во всех случаях, значения корреляции падают до нуля для

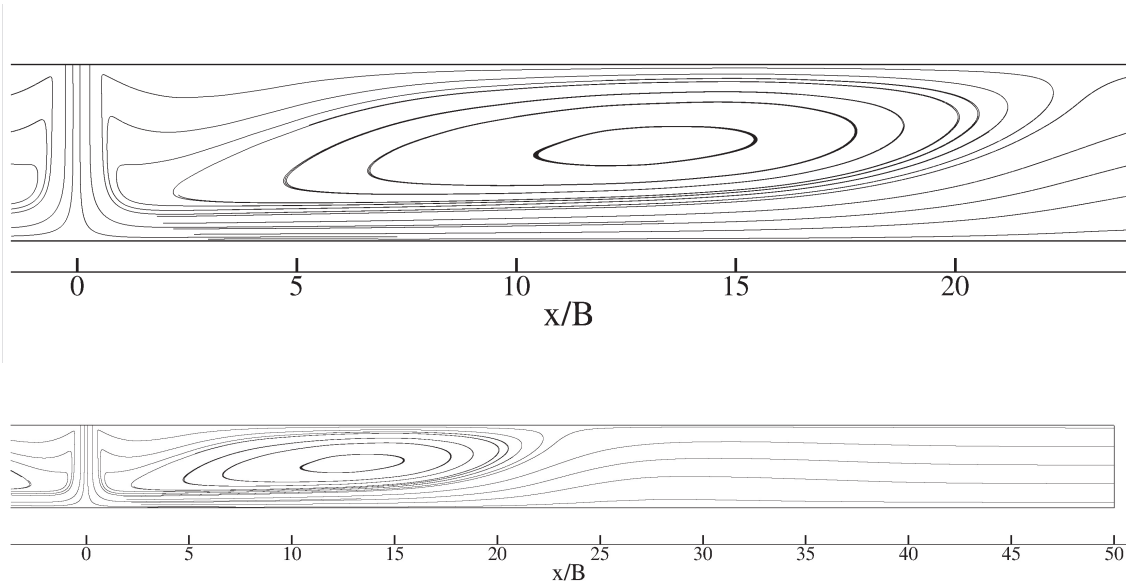


Рисунок 6.38: Линии тока осреднённого течения

разделения меньше половины периода $W/2 = \pi$, подтверждая, что расчётная область в таблице 6.3 имеет достаточную глубину. Аналогичные результаты были получены для других переменных и контрольных точек.

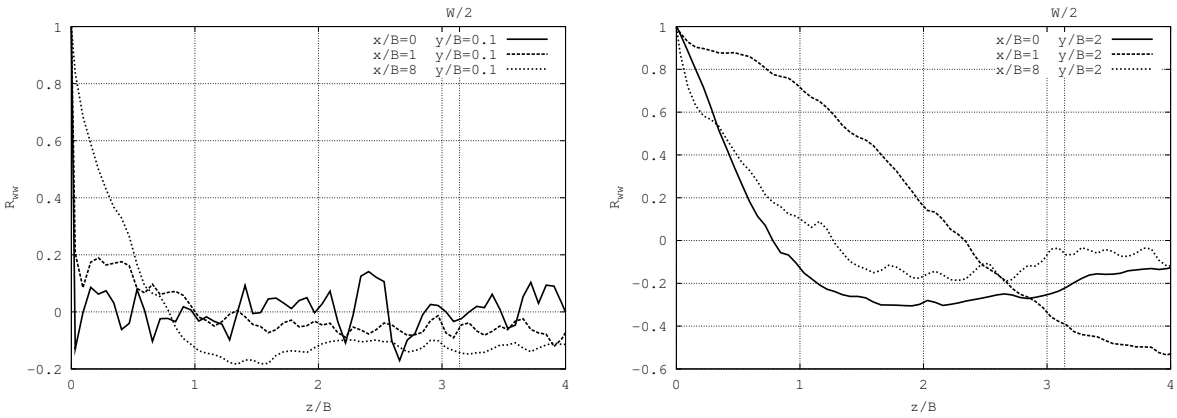


Рисунок 6.39: Двухточечная корреляция компоненты скорости w в 6 контрольных позициях

Период интегрирования по времени составил примерно 3700 безразмерных временных единиц (t_{ref}), из них последние $1000t_{ref}$ использовались для накопления статистики течения. Подробности о процедуре определения периода интегрирования могут быть найдены в [175].

Наконец, было изучено влияние входных граничных условий. С этой целью была выполнена дополнительная верификация путём прямого сравнения DNS расчёта с DNS результатами Hattori и Nagano [174] для плоской падающей струи с пропорцией 2 и числом Рейнольдса 4560. Была показана сильная зависимость решения от входных условий. В [174] авторы использовали полностью развитое течение в канале в качестве входных условий. Только в случае точного воспроизведения этих входных условий было получено хорошее согласование результатов. Для других входных условий, включая плоский профиль, осреднённый профиль течения в канале, были существенные расхождения. Из этих результатов был сделан вывод, что входные

условия в эталонном DNS расчёте должны быть максимально чётко определены и просты для воспроизведения. Из этих соображений был выбран плоский ламинарный профиль $(u,v,w) = (0,\overline{v}_{in},0)$ – он максимально прост и большинство экспериментов имеют близкие к этому профилю параметры входных условий с относительно низкой интенсивностью турбулентности ($I \leq 2\%$).

Для выбора пространственного разрешения и параметров сгущения сетки была выполнена аналогичная предыдущему расчёту процедура. В данном расчёте первый узел около нижней пластины расположен на расстоянии $y^+ \leq 1.7$. По одномерному энергетическому спектру $E_{\phi\phi}(x,y,k_3) = \hat{\phi}_{k_3}(x,y)\hat{\phi}^*_{k_3}(x,y)$ аналогичным образом проверялась достаточность пространственного разрешения по z направлению в тех же 6 контрольных точках ($(\cdot)^*$ обозначает комплексно сопряжённое число). Поскольку не наблюдается накачки энергии на высоких волновых числах и амплитуда плотности энергии между наименьшими и наибольшими волновыми числами падает на несколько порядков, энергетический спектр на рис. 6.40 показывает, что сетка имеет достаточное разрешение.

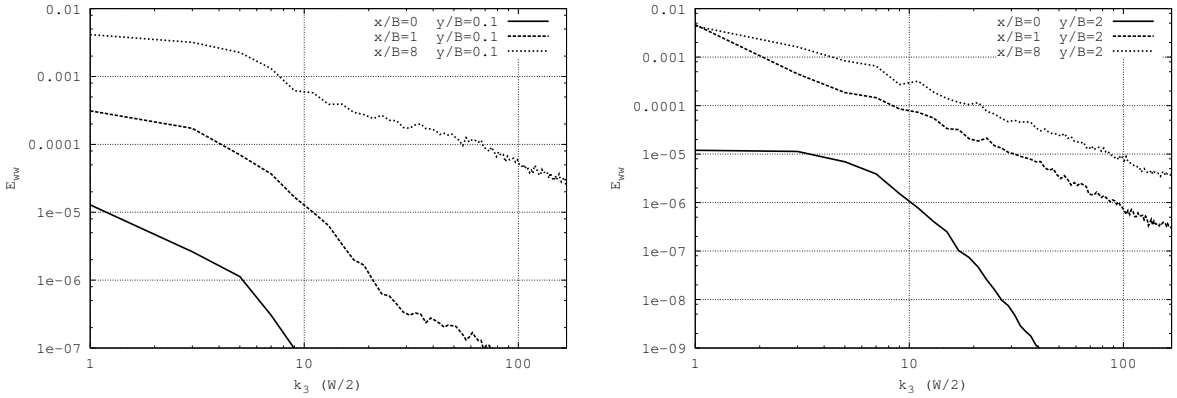


Рисунок 6.40: Энергетический спектр компоненты скорости w в 6 контрольных позициях

6.3.3 Результаты расчёта

Тепловое поле

Один из основных параметров в данной конфигурации, представляющих тепловое поле, – это локальное число Нуссельта вдоль нижней пластины: $Nu = \frac{\dot{q}_w}{\dot{q}} = \frac{-[\partial T/\partial y]_w}{(T_w - \bar{T}_{in})/B}$, где T_w – это температура пластины, а $\bar{T}_{in}$ – температура входной струи. График числа Нуссельта полученный в DNS расчёте показан на рис. 6.41 (в месте с результатами расчётов по некоторым RANS моделям). Пик Nu из результатов DNS составляет 63.5 в точке стагнации, что согласуется со значением 61.2, которое даёт эмпирическое соотношение предложенное в [176]. На рис. 6.42 показан безразмерный ортогональный стенке турбулентный тепловой поток в 5 различных позициях по оси x.

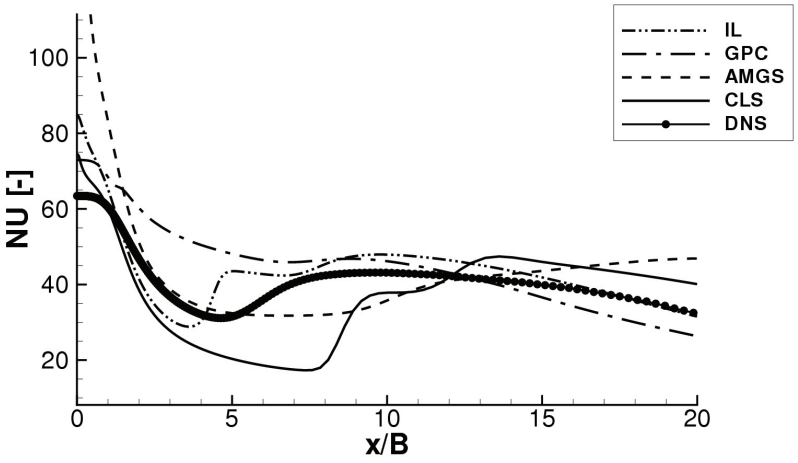


Рисунок 6.41: Локальное число Нуссельта на нижней пластине

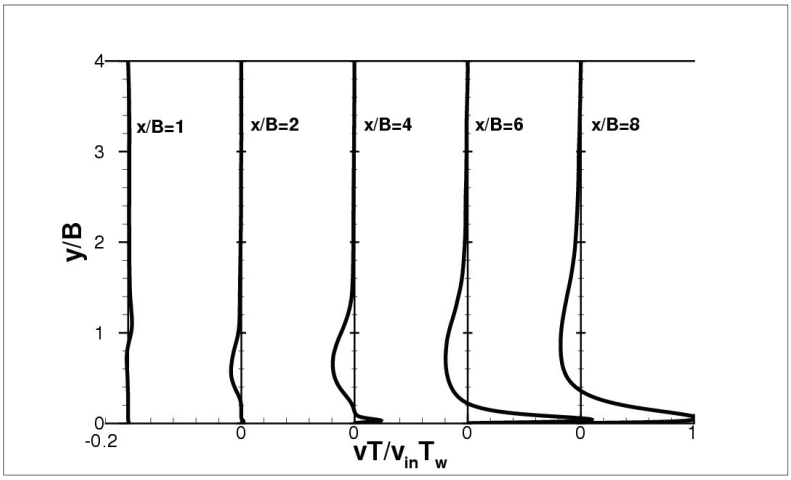


Рисунок 6.42: Профили ортогонального стеке турбулентного теплового потока

Поле течения

Моментальные картины течения показаны на рис. 6.43 и 6.44 для поля модуля скорости и поля температуры, соответственно. На рис. 6.45 показано распределение u компоненты скорости в пяти позициях по оси x. На рисунке видно, как ускорение потока около стенки приводит к быстрому росту u/u_{in} с y/B в области до $x/B \approx 2$. После $x/B \approx 4$ развивается пристенная струя. На рис. 6.46 показаны профили безразмерного среднего квадратического значения пульсаций u компоненты скорости, u'/v_{in}. Это значение показывает рост во внешней области развивающегося погранслоя с максимумом около $y/B = 0.8$. На рисунке этого не видно, но эти высокие значения сохраняются до конца зоны рециркуляции, где пристенная струя распадается. Кроме того, u'/v_{in} достигает вторичного максимум очень близко к стенке на диапазоне $6 < x/B < 10.4$, что хорошо согласуется с экспериментальными данными [177]. Профили безразмерного среднего квадратического значения v компоненты скорости, v'/v_{in}, показаны на рис. 6.47. Профили сдвигового напряжения Рейнольдса показаны на рис. 6.48.

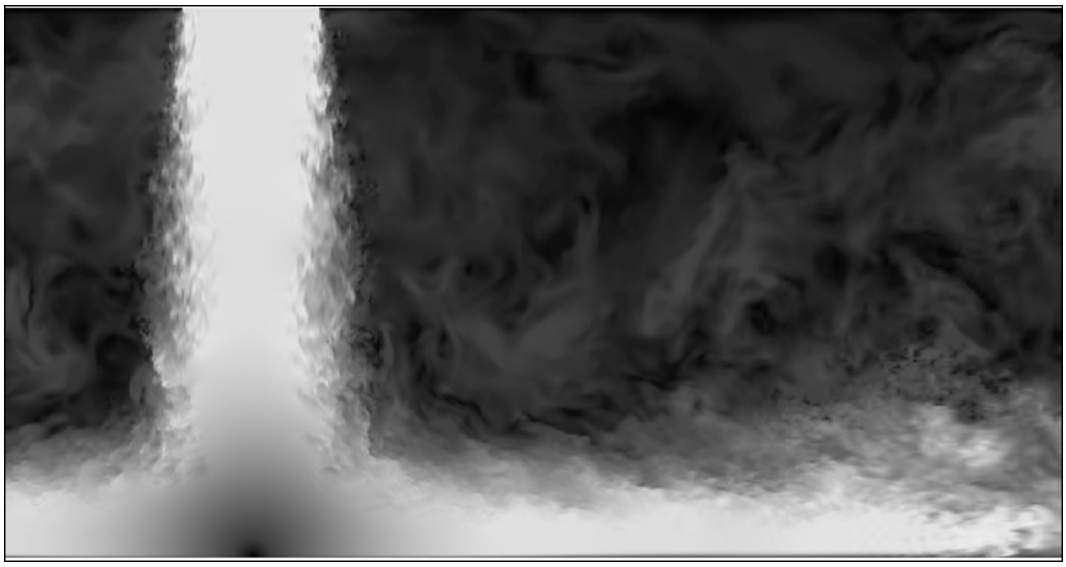


Рисунок 6.43: Моментальная картина течения (поле модуля скорости) в зоне струи

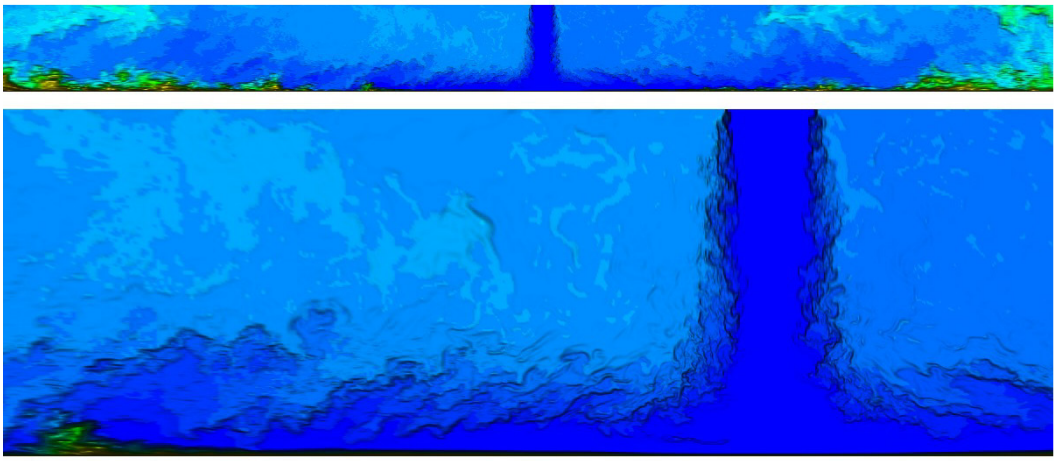


Рисунок 6.44: Моментальные картины течения (поле температуры)

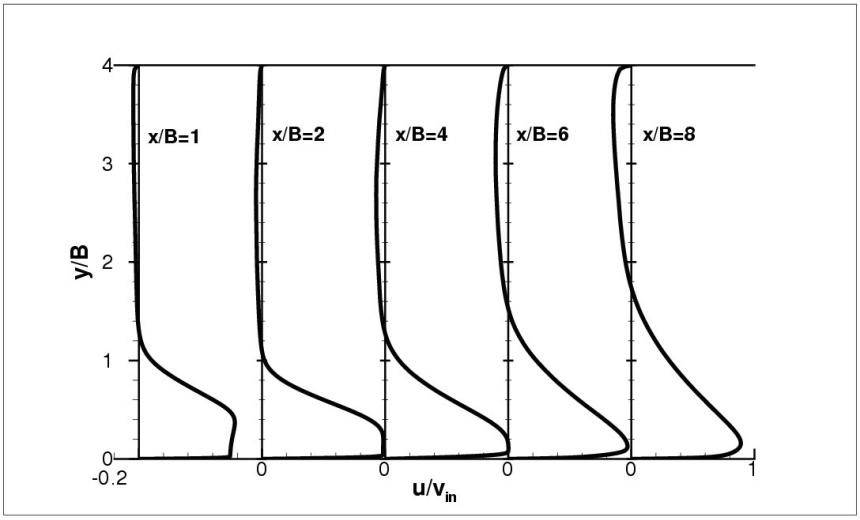


Рисунок 6.45: Осреднённые профили u компоненты скорости

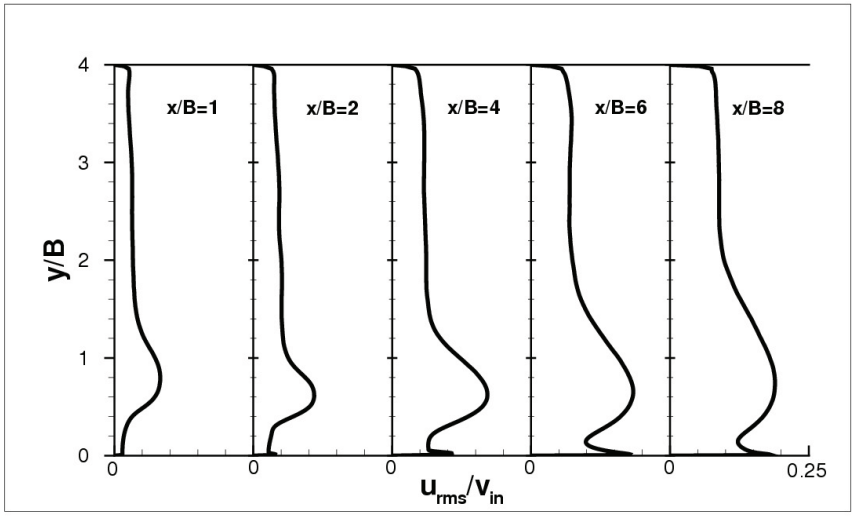


Рисунок 6.46: Осреднённые профили среднего квадратического значения u

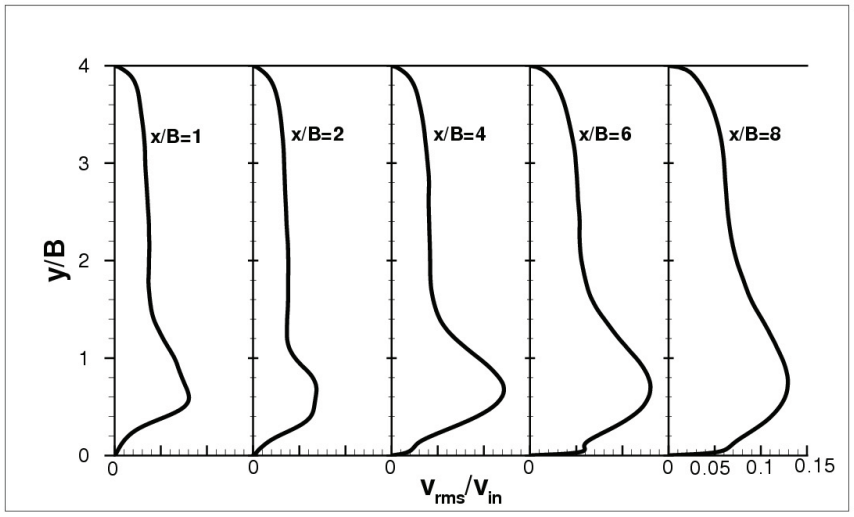


Рисунок 6.47: Профили среднего квадратического значения v компоненты скорости

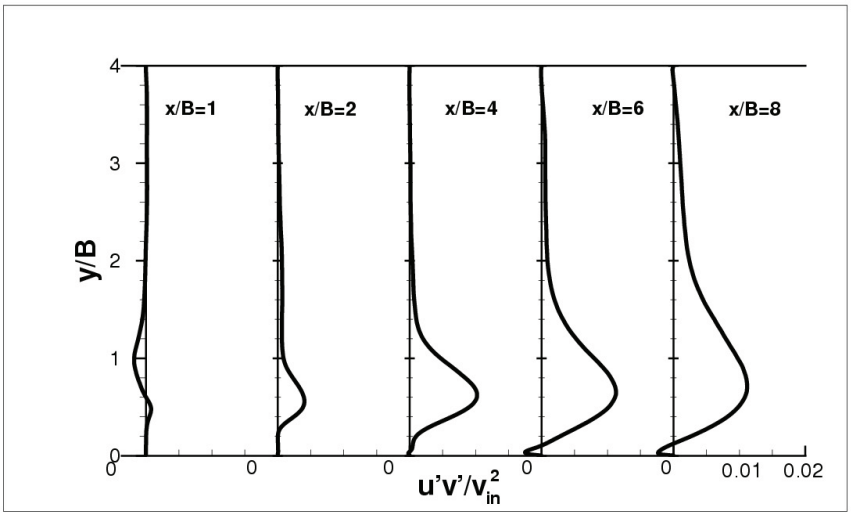


Рисунок 6.48: Профили сдвигового напряжения Рейнольдса

6.4 DNS течения вокруг квадратного цилиндра

6.4.1 Постановка задачи

Течение вокруг цилиндров с прямоугольным сечением широко изучалось в последние десятилетия и экспериментально и численно. Подробный обзор представлен в [178]. Кроме того, исчерпывающий обзор численных результатов из ERCOFTAC [179,180] имеется в [181]. Данный тип течения принципиально отличается от течения вокруг круглого цилиндра тем, что отрыв потока происходит на острых углах, и, следовательно, точки отрыва фиксированы.

В данном расчёте с помощью программного комплекса STG-CFD&HTна сетке из 322 млн узлов схемой 4-го порядка моделируется течение несжимаемой ньютоновской жидкости в расчётной области, схема которой показана на рис.6.49. Расчёт выполнялся на суперкомпьютере MareNostrum, BSC, Испания. Число Рейнольдса, $Re = UD/\nu$, берется по скорости входного потока U и толщине цилиндра D. В безразмерном виде размеры расчётной области $30.5D \times 54D \times \pi D$ в направлении течения, в поперечном направлении и в направлении оси цилиндра, соответственно. Передняя грань цилиндра находится на расстоянии $10D$ от входной границы, цилиндр расположен по центру в поперечном (вертикальном) направлении. Начало координат находится в центре цилиндра. Граничные условия на входе – постоянный профиль скорости, $\boldsymbol{u} = (U, 0, 0)$, на выходе – конвективные граничные условия. В поперечном направлении, то есть сверху и снизу – условия Неймана. В направлении оси цилиндра используются периодические граничные условия. На поверхности цилиндра условия соответствую твёрдой стенке без проскальзывания. Результаты представляются в безразмерной форме с характерной длиной, скоростью, временем и кинематическим давлением, соответственно, D, U, D/U, $U^2/2$.

Для дискретизации используется схема 4-го порядка [133]. В дополнение к базовой процедуре верификации кода, описанной в предыдущих задачах, выполнено сравнение с предыдущими численными исследованиями [182–184].

6.4.2 Верификация расчёта

В целом процедура верификации аналогична предыдущим расчётам. Размер по периодической оси, L_z, должен быть достаточно большим, чтобы не влиять на решение, шаг сетки Δz в этом направлении достаточным для воспроизведения всех релевантных масштабов течения. Кроме того, цилиндр должен быть расположен достаточно далеко от границ расчётной области. Наконец, момент начала осреднения и продолжительность осреднения должны быть выбраны соответствующим образом, чтобы корректно воспроизвести статистику течения.

В предварительном расчёте использовалась более грубая сетка $128 \times 708 \times 708$ (Mesh2, см. таблицу 6.4). Глубина по периодике $L_z = 4$, такое же значение, как в исходной задаче, предложенной на семинарах ERCOFTAC [179, 180], и использовавшейся в большинстве последующих численных исследований. Этот размер должен быть достаточно большим, чтобы турбулентные флуктуации были не коррелированы на разделении половины периода $L_z/2$.

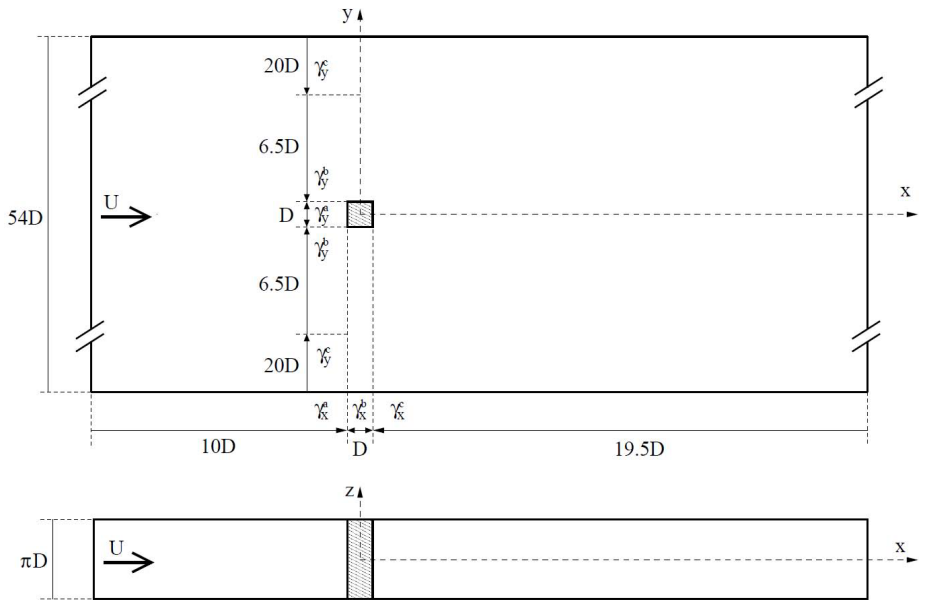


Рисунок 6.49: Схема задачи, зоны сгущения сетки (стрелками показаны направления сгущения)

Case	$N_x \times N_y \times N_z$	L_z	γ_x^a, γ_x^b, γ_x^c, γ_y^a, γ_y^b, γ_y^c	$(\Delta x)^+_{min}$	$(\Delta y)^+_{min}$	Total time	Average time
Mesh1	$1272 \times 1174 \times 216$	π	2.2, 0.5, 2, 0.5, 2, 1.7	$\lesssim 0.6$	$\lesssim 0.15$	690	550
Mesh2	$708 \times 708 \times 218$	4	2.2, 0.5, 2, 0.5, 2, 1.7	$\lesssim 0.9$	$\lesssim 0.25$	300	200

Таблица 6.4: Физические и численные параметры расчёта.

Рис. 6.50 показывает анализ двухточечной корреляцию в периодическом направлении, R_{ww}, в пяти различных (x,y)-позициях. Видно, что значения корреляции падают до нуля. Результаты показали, что можно уменьшить размер по периодике с целью снижения вычислительных затрат. Для основного расчёта было выбрано $L_z = \pi$. В продольном и поперечном направлениях размеры области соответствуют исходно предложенной на ERCOFTAC [179,180] за исключением дополнительных буферных зон размера $20D$, добавленных сверху и снизу (см. рис. 6.49). В предварительных расчётах наблюдалось влияние размера в поперечном направлении на результаты в области интереса, из-за этого был применен этот подход.

Процедура определения сеточного разрешения аналогична предыдущим расчётам. Шаг сетки по периодическому направлению постоянный, по остальным направлениям, ортогональным твёрдым поверхностям, используется сгущение с использованием кусочных функций гиперболического тангенса. Распределение узлов на отрезке со сгущением к обоим концам отрезка задаётся выражением (6.1). Там, где сгущение требуется только для одного конца отрезка, берется распределение для в два раза большего отрезка и отрезается половина. Параметры сгущения γ_x^a, γ_x^b и γ_x^c определяются из соображений минимизации градиентов течения в вычислительном пространстве на наборе репрезентативных моментальных картин течения. Общее число точек по оси x равно сумме отрезков $N_x^a + N_x^b + N_x^c = N_x$, а размеры контрольных объёмов на стыке отрезков равны. Распределение по оси y сделано аналогичным образом. Значения параметров сгущения и параметров расчёта представлены в

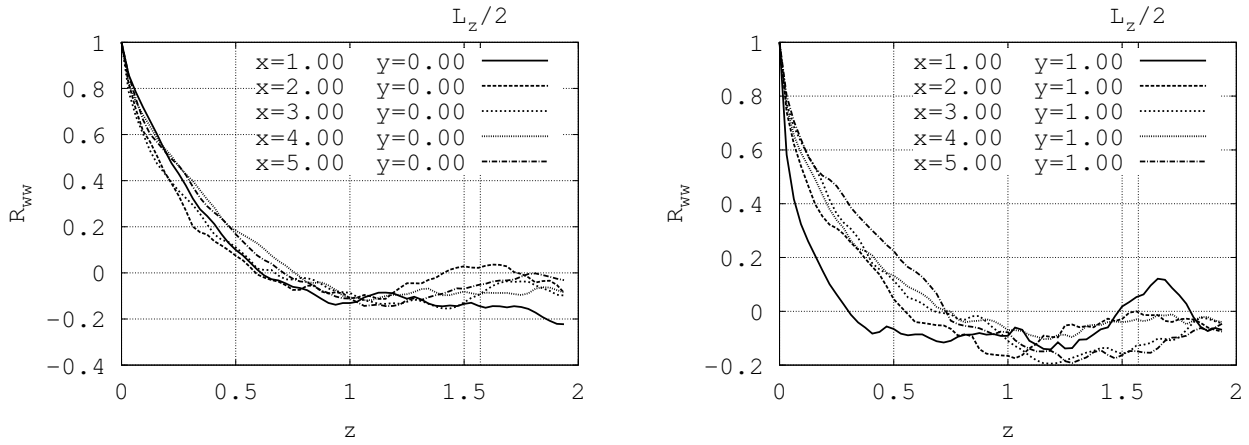


Рисунок 6.50: Двухточечная корреляция в периодическом направлении для компоненты скорости w в пяти контрольных позициях

таблице 6.4. Сдвиговая скорость, u_τ, вычисляемая по локальному сдвиговому напряжению, падает до значений меньше единицы для всех четырёх стенок препятствия, что свидетельствует о достаточности сеточного разрешения.

Интервал интегрирования по времени составил 690 безразмерных временных единиц. Из низ последние 550 использовались для накопления статистики течения, что соответствует примерно 75 циклам шеддинга, что намного больше, чем в предыдущих расчётах. В [180] был предложен период 13 циклов. Недавние LES результаты [185] также использовали такую длительность. В этом расчёте наблюдалось, что требуется значительно больший период осреднения, для получения точных результатов.

6.4.3 Результаты расчёта

Среднее поле

Для получения статистики течения выполнялось осреднение по трем статистически инвариантным преобразованиям (время, периодическое направление, плоскость симметрии) для всех полей. $\langle\cdot\rangle$ используется для обозначения процедуры осреднения. Осреднённое поле давления и линии тока показаны на рис. 6.51 и 6.52, соответственно. Распределение давления по поверхности цилиндра показано на рис. 6.53. Коэффициенты лобового сопротивления и подъемной силы, C_D и C_L, получены путём интегрирования давления и сдвигового напряжения на поверхности цилиндра. Осреднённое значение коэффициента лобового сопротивления составило $\langle C_D\rangle = 2.23$. Хотя это интегральная величина, её достаточно сложно корректно предсказать. В работе [181] собраны почти все результаты из ERCOFTAC [179, 180]. Численные результаты показали большие расхождения по $\langle C_D\rangle$, варьируясь от 1.9 до почти 2.8. В таблице 6.5 представлены результаты из различных численных исследований и экспериментов. Более подробное описание среднего поля течения дают рис. 6.54 и 6.55, где показаны профили осреднённой скорости около поверхности цилиндра.

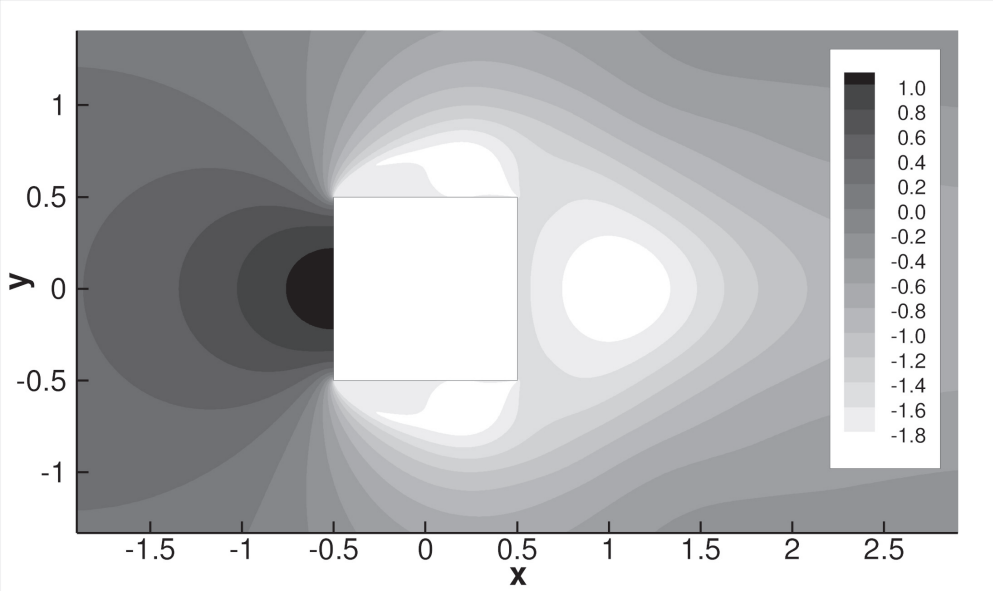


Рисунок 6.51: Осреднённое поле давления

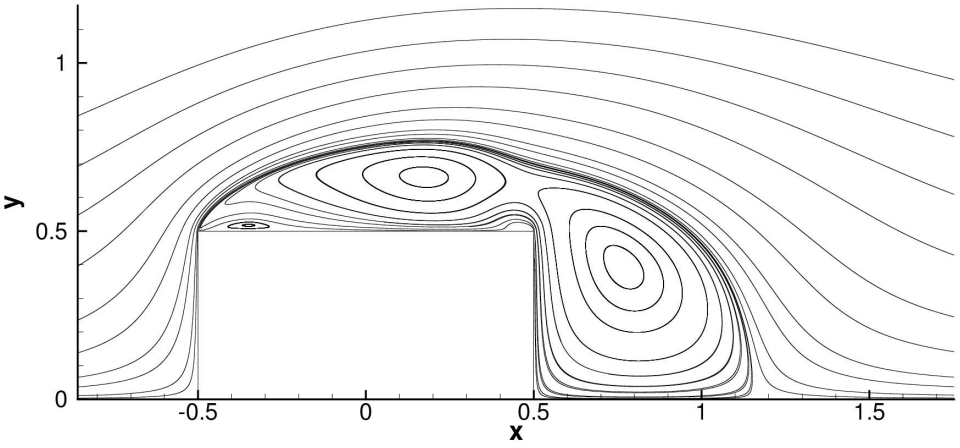


Рисунок 6.52: Осреднённые линии тока

Динамика течения

Главной особенностью течений такого типа является образование вихревой дорожки Кармана. Это явление создаёт воздействие сил на цилиндре. На рис. 6.56 показана эволюция во времени лобового сопротивления и подъемной силы на интервале 90 временных единиц. Коэффициент подъемной силы чётко показывает доминантную частоту, соответствующую формированию вихрей. Пик в нормированном энергетическом спектре (см. рис. 6.57, справа) находится на значении 0.131. Полученное число Струхаля St хорошо согласуется с экспериментальными данным (см. таблицу 6.5). В заключении, моментальные картины течения показаны на рис. 6.58 и 6.59.

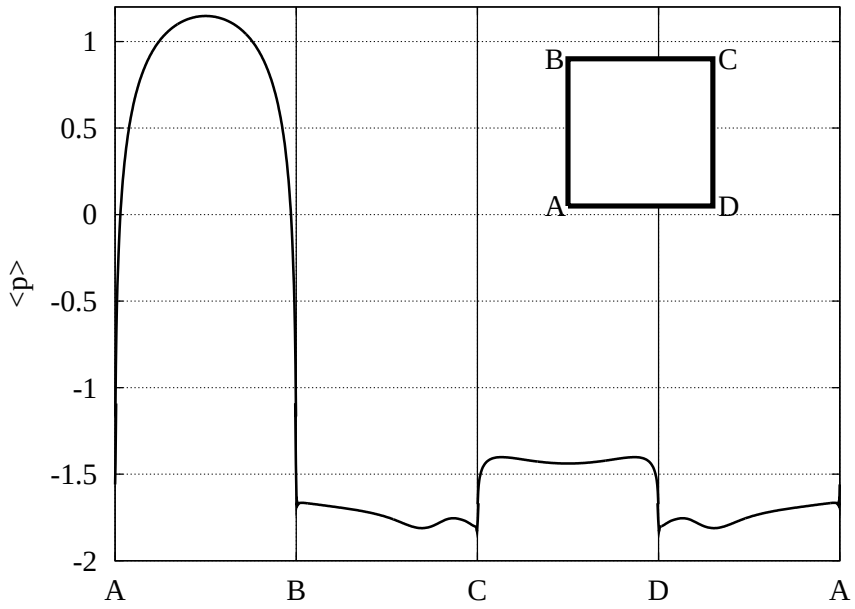


Рисунок 6.53: Осреднённое распределение давления по поверхности цилиндра

Case	St	$\langle C_D \rangle$	C_D^{rms}	C_L^{rms}	l_R
DNS	0.132	2.18	0.162	1.52	1.04
Minguez *et al.* [185]	0.130	2.1	–	–	
Lyn *et al.* [186, 187]	0.133	2.1	–	–	
Lee [188]	-	2.05	$0.16 - 0.23$	$0.68 - 1.32$	
Vickery [189]	0.133	2.1	–	–	
Minguez *et al.* [185]	0.141	2.2	–	–	1.28
Ochoa & Fueyo [190]	0.139	2.01	0.22	1.4	–
Sohankar *et al.* [191]	$0.126 - 0.132$	$2.03 - 2.32$	$0.16 - 0.20$	$1.23 - 1.54$	–
Verstappen & Veldman [181]	0.133	2.09	0.178	1.45	–
Rodi [192]	0.13	2.3	0.14	1.15	1.46
LES results [179, 180]	$0.09 - 0.15$	$2.02 - 2.77$	$0.14 - 0.27$	$1.15 - 1.79$	$0.94 - 1.68$
RANS results [179, 180]	$0.134 - 0.159$	$1.64 - 2.43$	$\approx 0 - 0.27$	$0.31 - 1.49$	$0.98 - 2.80$

Таблица 6.5: Сравнение с предыдущими экспериментами и численными результатами. Слева направо: число Струхаля, осреднённый коэффициент лобового сопротивления, среднее квадратическое значение пульсаций коэффициентов лобового сопротивления и подъемной силы, длина повторного присоединения. Вверху вниз: результаты данного DNS расчёта, экспериментальные результаты [185–189], результаты LES расчётов [185, 190–192] и более грубых DNS [181], диапазоны значений в расчётах методами LES и RANS, представленных в ERCOFTAC [179, 180].

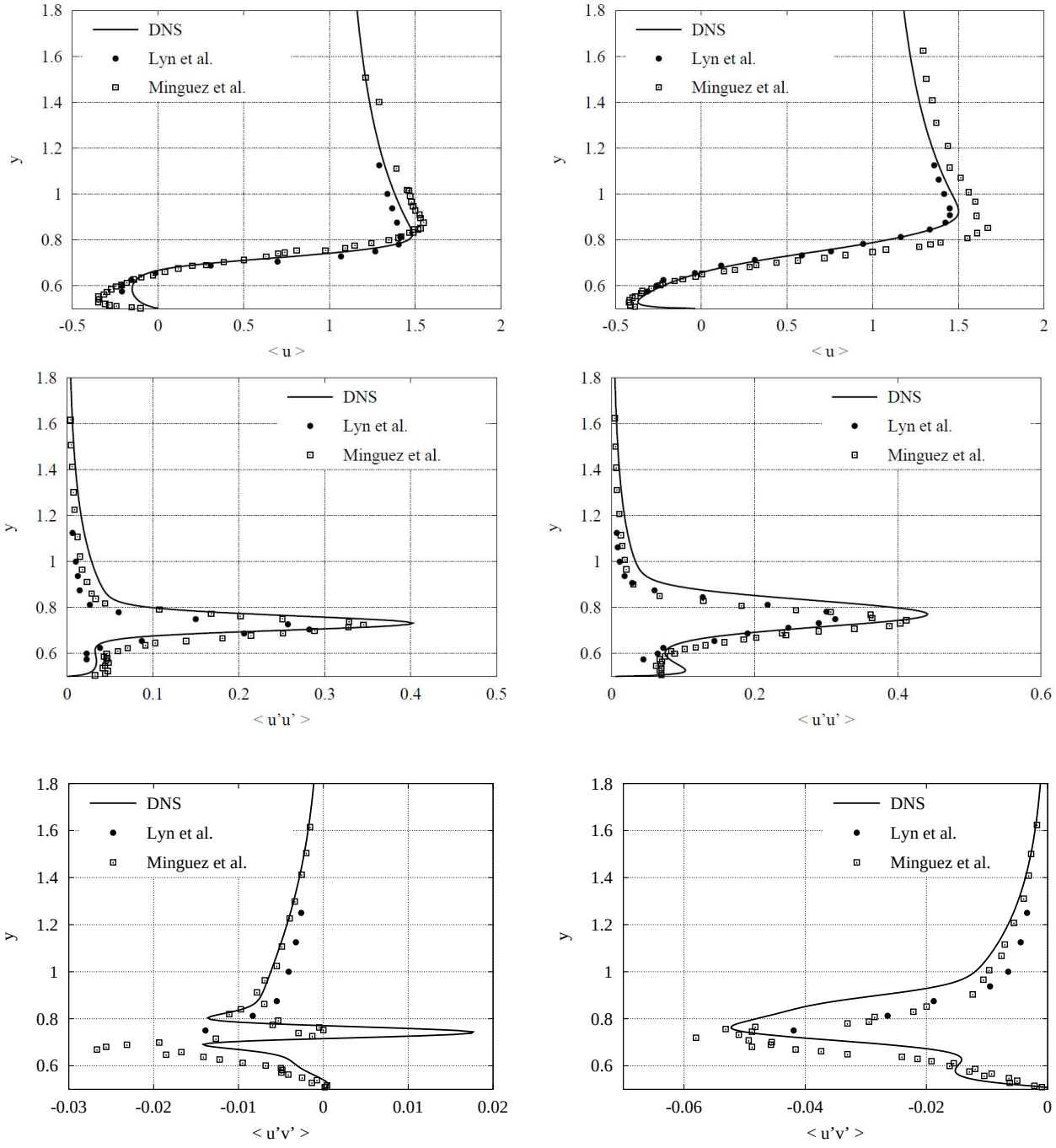


Рисунок 6.54: Профили осредненных величин $\langle u_x \rangle$ (сверху), $\langle u'_x u'_x \rangle$ (по середине) и $\langle u'_x u'_y \rangle$ (снизу) в двух позициях: $x = -0.125$ (слева) и $x = 0.125$ (справа). в сравнении с экспериментальными результатами [187] и [185]

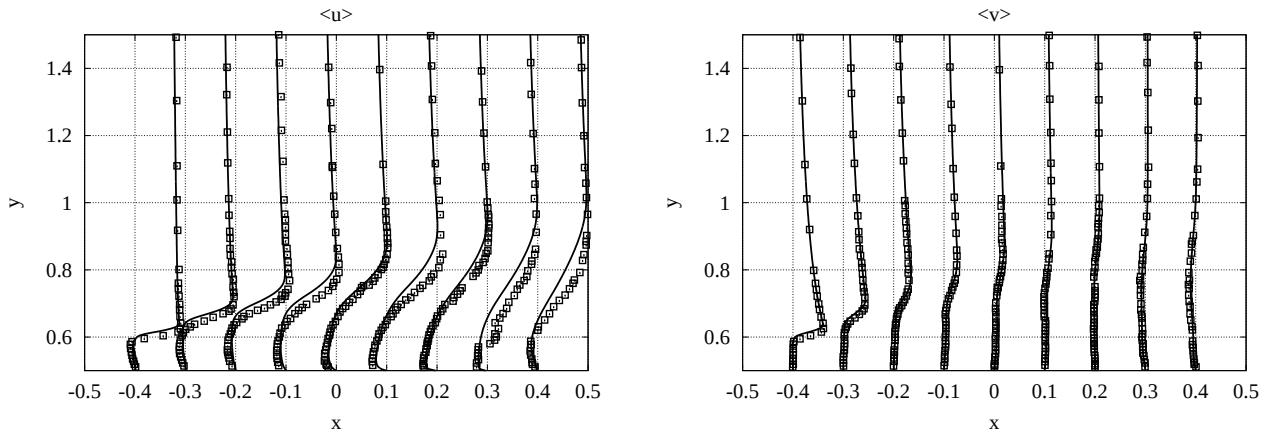


Рисунок 6.55: Профили осреднённых величин $\langle u_x \rangle$ и $\langle vvel \rangle$ в области около цилиндра в сравнении с экспериментальными результатами [185].

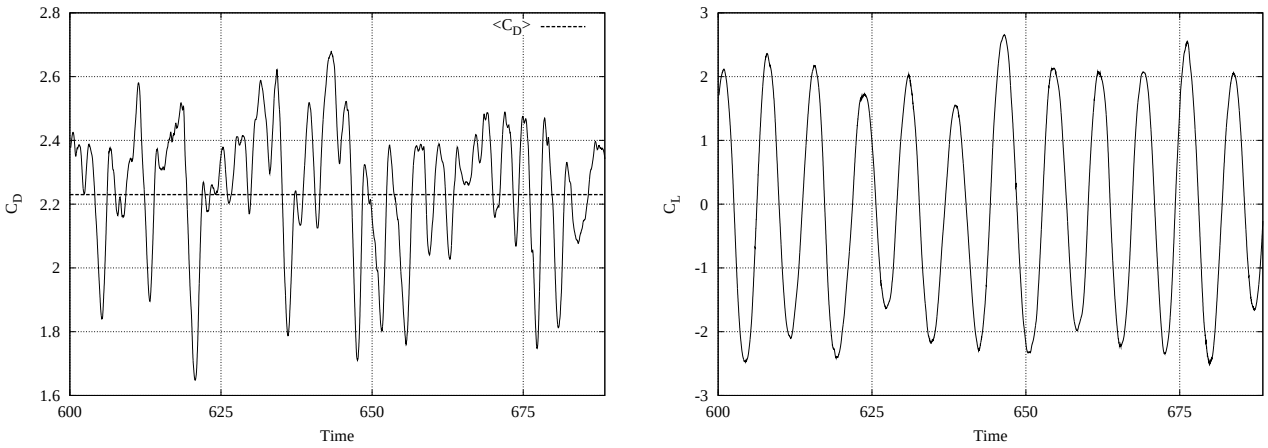


Рисунок 6.56: Эволюция во времени коэффициента лобового сопротивления (слева) и подъемной силы (справа) на интервале 90 временных единиц

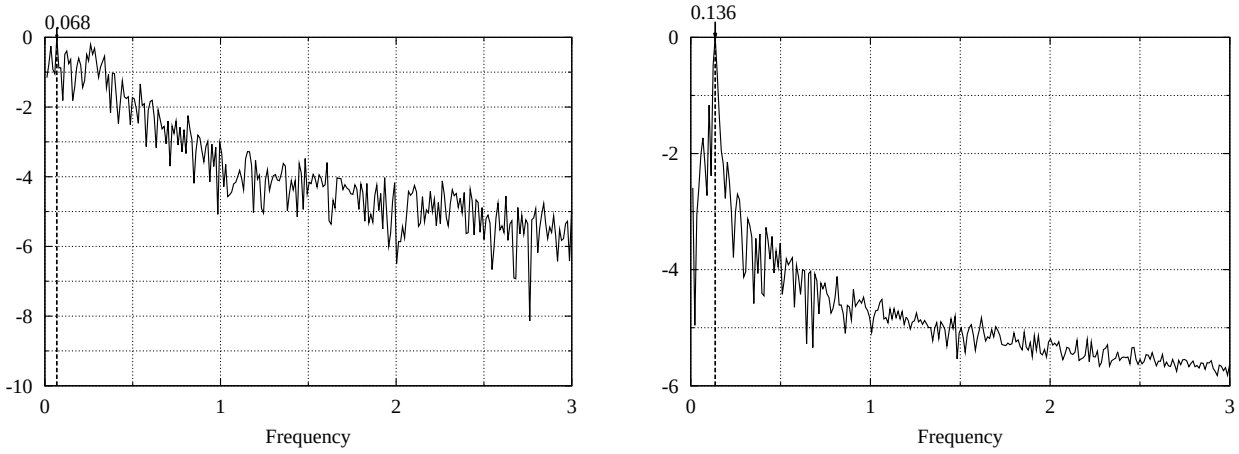


Рисунок 6.57: Нормированный энергетический спектр пульсаций коэффициента лобового сопротивления (слева) и подъемной силы (справа)

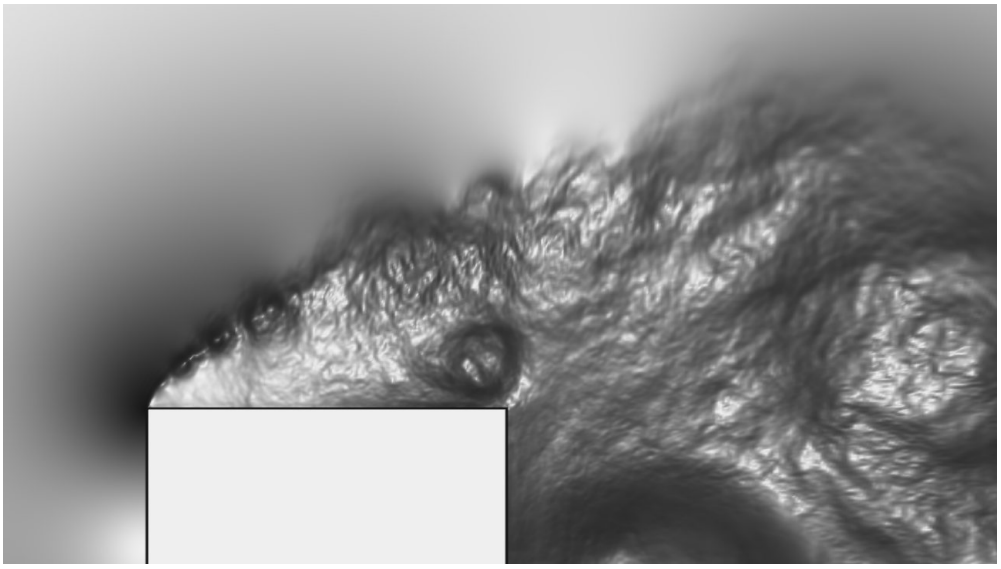


Рисунок 6.58: Визуализация структур Кельвина-Гельмгольца (моментальная картина модуля градиента давления)

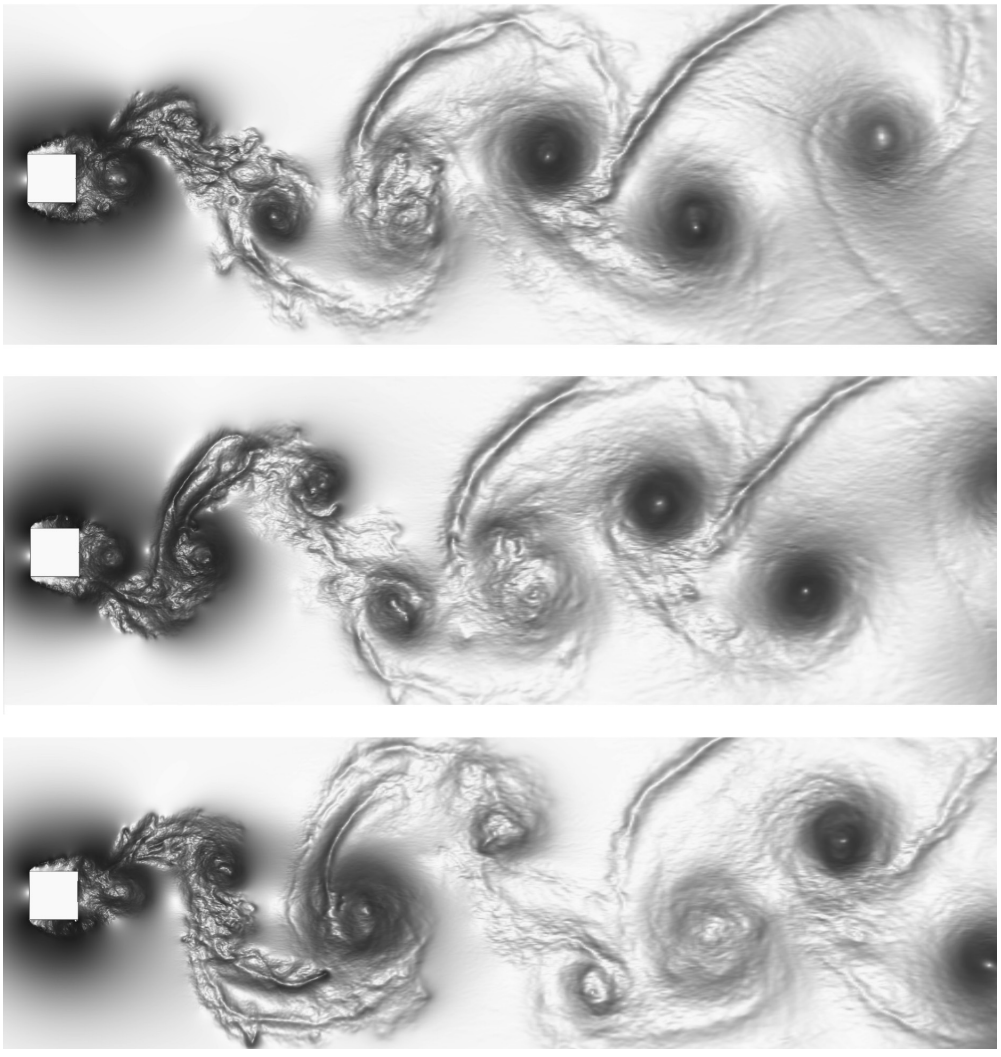


Рисунок 6.59: Временная последовательность моментальных картин турбулентного следа (модуль градиента давления)

Re_τ	L_x/h	$N_x \times N_y \times N_z$	L_x^+	L_y^+	Δx^+	Δy^+_{min}	γ
300	2π	$160 \times 128 \times 128$	1885	300	11.78	0.224	1.85
600	2π	$320 \times 256 \times 256$	3770	600	11.78	0.216	1.85
900	2π	$480 \times 384 \times 384$	4050	900	11.78	0.215	1.85
1200	2π	$640 \times 512 \times 512$	7540	1200	11.78	0.214	1.85

Таблица 6.6: Физические и численные параметры расчётов течения в квадратной трубе.

6.5 DNS течения в квадратной трубе

6.5.1 Постановка задачи

Основной целю данного численного исследования является изучение полностью развитого турбулентного течения в квадратной трубе с числом Re_τ до 1200. В данной серии вычислительных экспериментов выполнены расчёты схемой 4-го порядка для чисел Re_τ 300, 600, 900, и 1200 – наиболее крупный расчёт на сетке из 168 мнл узлов. Размеры расчётной области, схема которой показана на рис. 6.60, равны $L_x \times h \times h$, где h – высота трубы. Для расчёта использовался программный комплекс STG-CFD&HT, представленный в предыдущей главе. Расчёт выполнялся на суперкомпьютере MareNostrum, BSC, Испания.

В целом процедура верификации аналогична предыдущим расчётам. На рис.6.61 показаны данные для двухточечной корреляции по периодическому направлению (продольное направление), исходя из которых выбирался размер L_x расчётной области по этому направлению. Для сгущения сетки использовалась функция (6.1) с параметром сгущения γ. Параметры численных экспериментов представлены в таблице 6.6. Более подробно описание задачи, обзор исследований данной конфигурации и верификация расчёта представлены в [193].

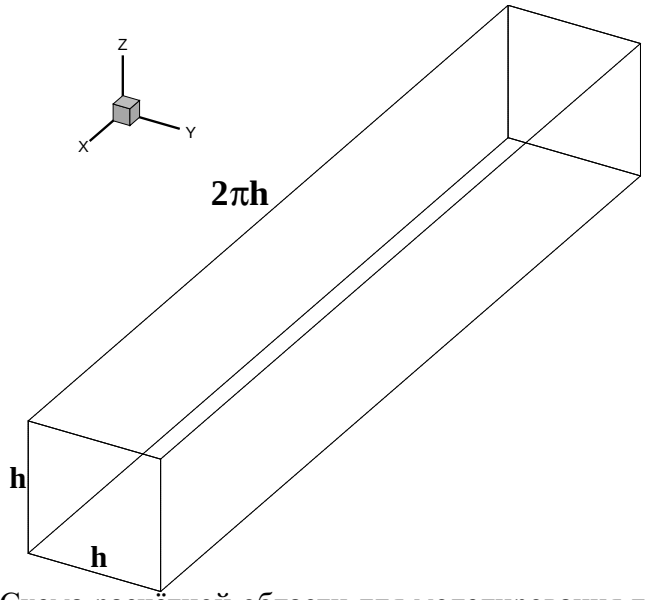


Рисунок 6.60: Схема расчётной области для моделирования течения в квадратной трубе

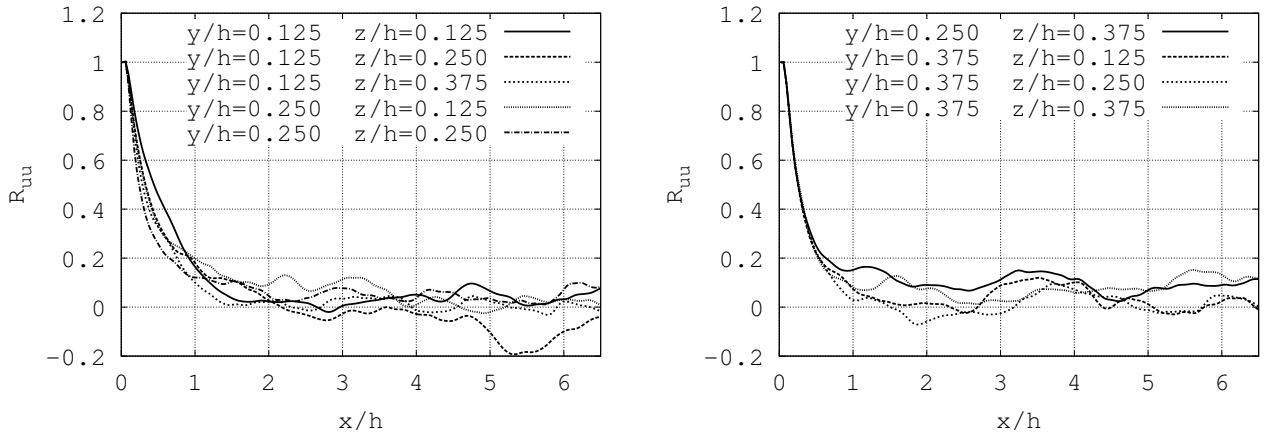


Рисунок 6.61: Двухточечная корреляция компоненты скорости u_x в 9 контрольных позициях. Данные результаты соответствуют предварительному расчёту с $L_x/h = 4\pi$, в 2 раза больше, чем в основных расчётах (см. таблицу 6.6).

6.5.2 Результаты расчёта

Моментальная картина турбулентного течения для $Re_\tau = 1200$ показана на рис. 6.62.

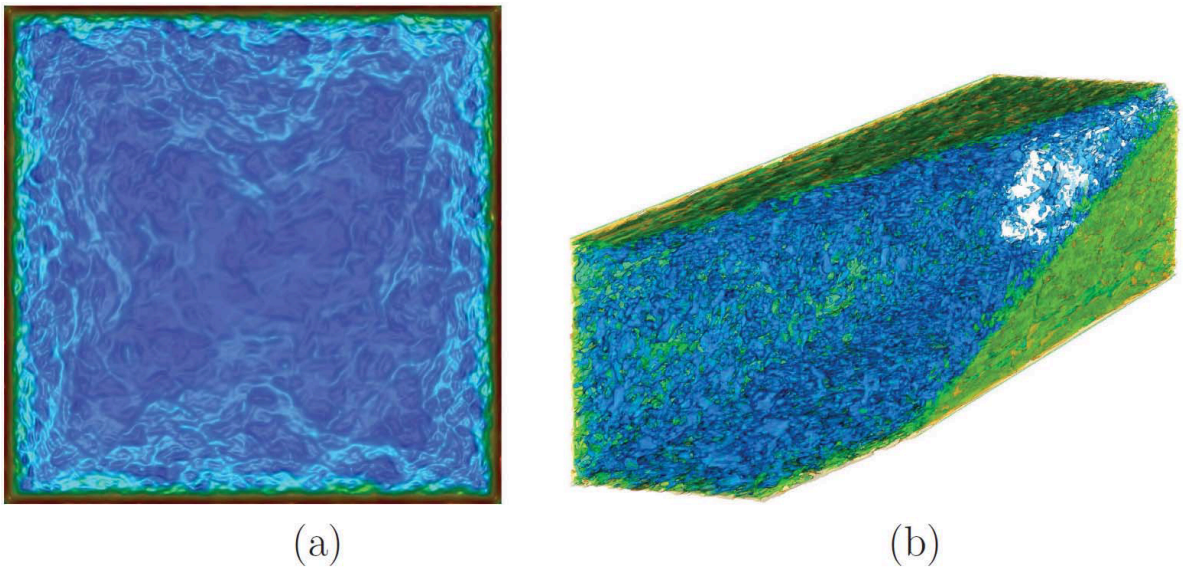


Рисунок 6.62: (a) поперечное 2D сечение и (b) 3D моментальные картины течения – показано распределение модуля завихренности, $Re_\tau = 1200$

Для получения статистики течения выполнялось осреднение по статистически инвариантным преобразованиям: время, периодическое направление, 4 плоскости симметрии. Осреднённая картина вторичного течения показана на рис. 6.63 для 4-х чисел Re_τ. Контуры показаны для продольной компоненты скорости, нормированной по осреднённой скорости по центру трубы $u_c = \langle u(h/2, h/2)\rangle$. В каждой четверти видны пары вихрей, вращающихся в противоположном направлении. Позиции центра нижнего вихря (y/h, z/h): (0.26, 0.11) для $Re_\tau = 300$, (0.31, 0.13) для $Re_\tau = 600$, (0.33, 0.14) для $Re_\tau = 900$ и (0.33, 0.13) для $Re_\tau = 1200$.

Осреднённое распределение напряжения на стенке в сравнении данными Gavrilakis [194] для $Re_\tau = 300$ и Huser и Biringen [195] для $Re_\tau = 600$ показано на рис. 6.64. Результаты

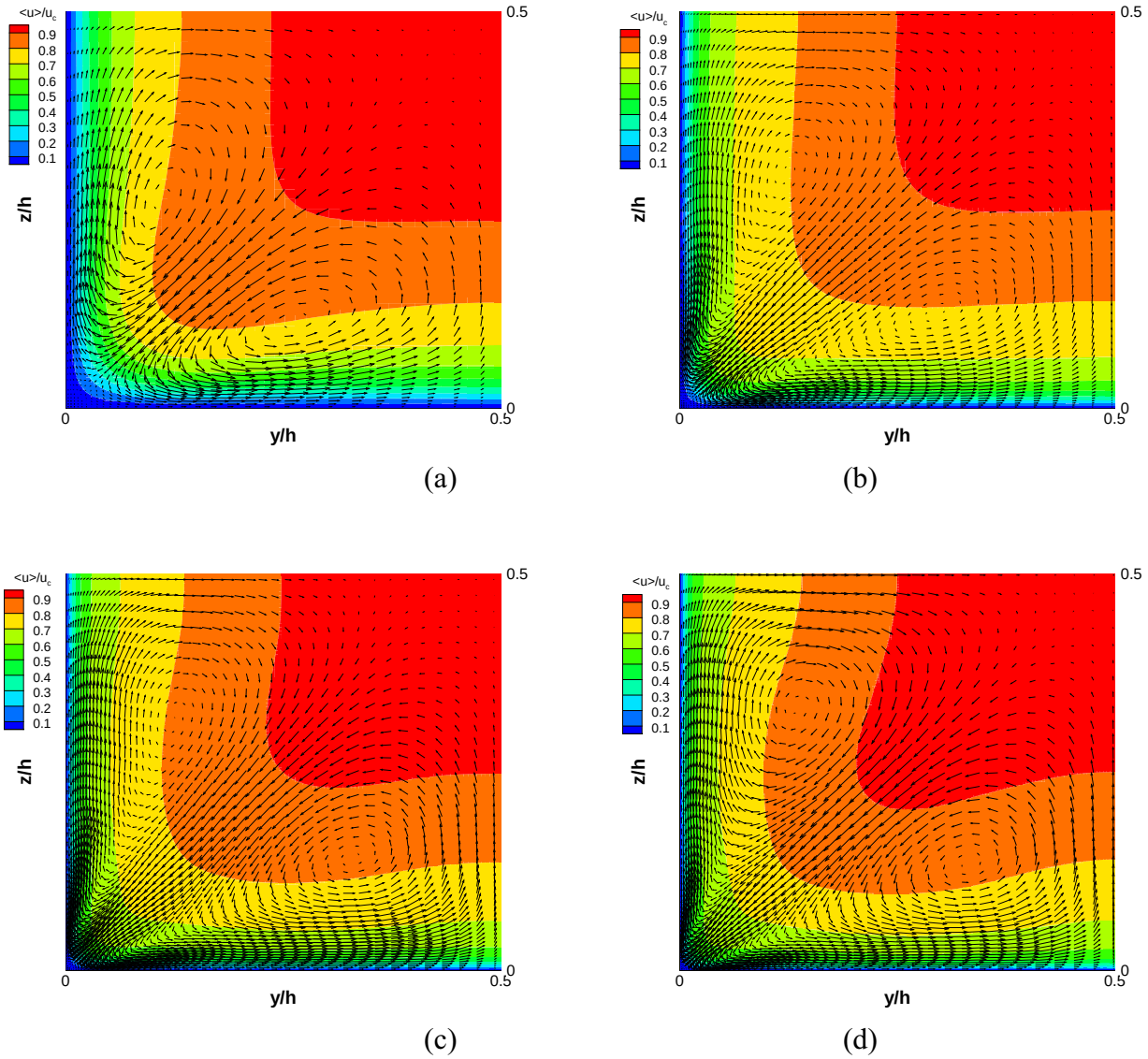


Рисунок 6.63: Линии тока (вторичное течение) и контуры продольной компоненты скорости u_x: (a) $Re_\tau = 300$, (b) $Re_\tau = 600$, (c) $Re_\tau = 900$, (d) $Re_\tau = 1200$

хорошо согласуются для $Re_\tau = 300$, однако наблюдаются расхождения с результатами [195], полученными на многократно более грубой сетке.

В таблице 6.7 приведены численные характеристики течения и сравнение с результатам других авторов, где u_b – суммарная скорость течения, u_{tau} – скорость касательного напряжения на стенке, коэффициент трения $F_f = 8u_\tau^2/u_b^2$.

Осреднённые профили продольной компоненты скорости u_x показаны на рис. 6.65 для 4-х чисел Re_τ, где также для сравнения показаны результаты [194]. Для поперечной компоненты скорости u_y аналогичные профили показаны на рис. 6.66.Распределение интенсивности турбулентности, нормированной по локальной продольной компоненте скорости, показано на рис 6.67 вместе с результатами [194]. Более подробно полученные результаты расчётов и их анализ представлены в [193].

Re_τ	Источник	u_c/u_b	F_f	u_τ/u_b
300	данный расчёт (DNS)	1.33	0.037	0.068
	Gavrilakis (DNS) [194]	1.33	0.037	0.068
	Sharma (DNS) [196]	1.32	0.035	0.066
600	данный расчёт (DNS)	1.30	0.031	0.062
	Hartnett (Experiment) [197]	–	0.030	0.061
	Huser (DNS) [195]	–	0.027	0.058
900	данный расчёт (DNS)	1.27	0.028	0.059
1200	данный расчёт (DNS)	1.26	0.025	0.056

Таблица 6.7: Численные характеристики течения и сравнение с результатам других авторов

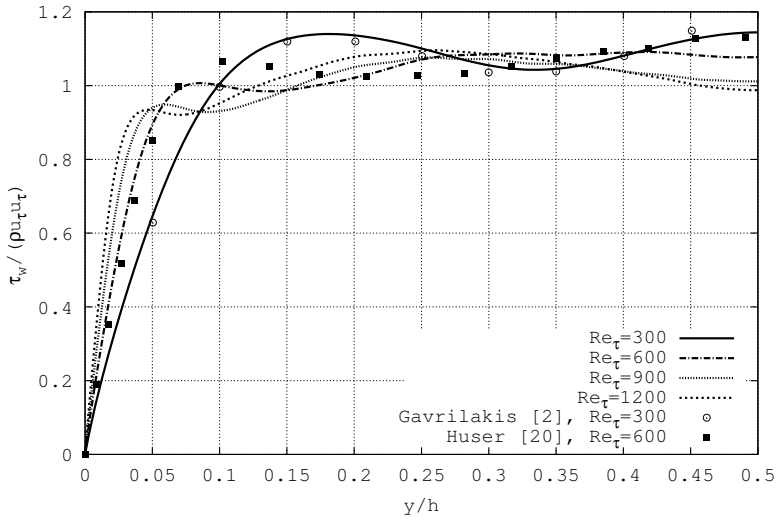
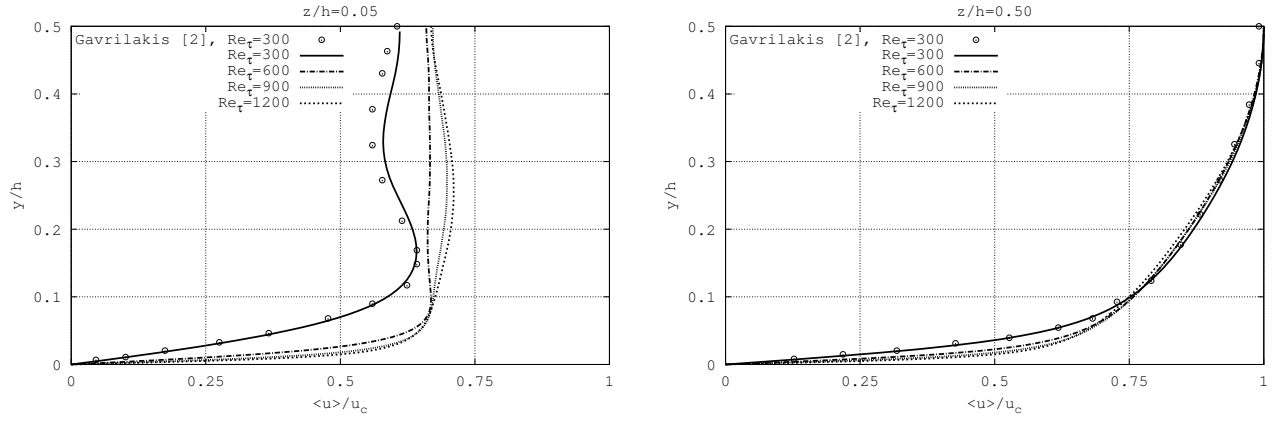
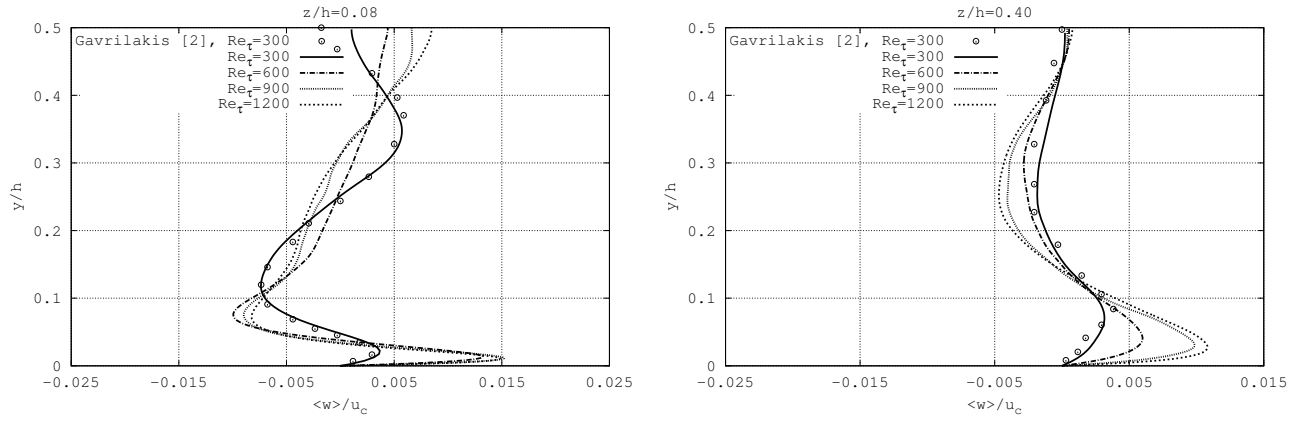


Рисунок 6.64: Осреднённое распределение напряжения на стенке в сравнении с экспериментальными данными Gavrilakis [194], Huser и Biringen [195]



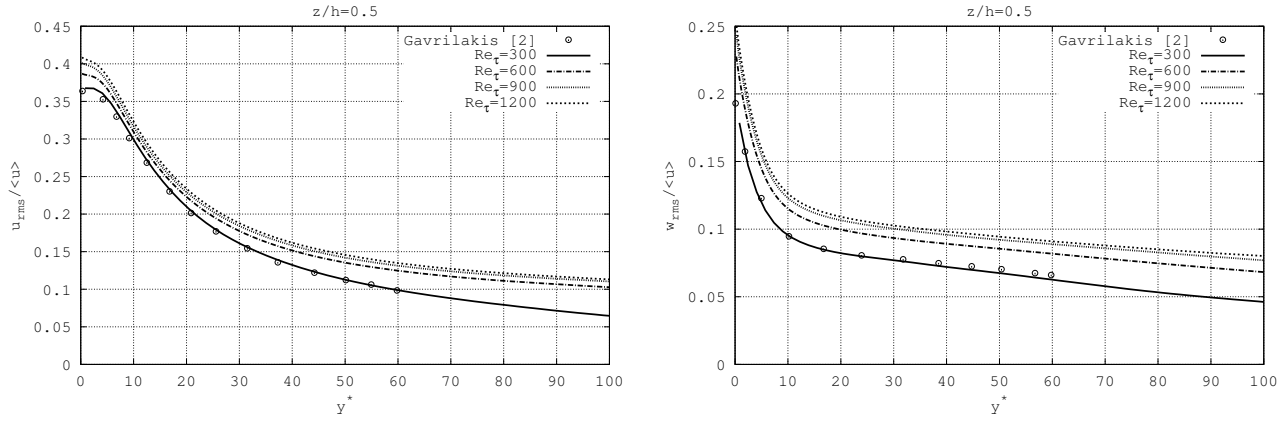
(a) (b)

Рисунок 6.65: Средние профили продольной компоненты скорости u_x: (a) по центру около нижней стенки и (b) по биссектрисе между стенками, в сравнение с данными [194]



(a) (b)

Рисунок 6.66: Средние профили поперечной компоненты скорости u_y: (a) по центру около нижней стенки и (b) по биссектрисе между стенками, сравнение с данными [194]



(a) (b)

Рисунок 6.67: Распределение интенсивности турбулентности, нормированной по локальной продольной компоненте скорости

6.6 DNS течения вокруг куба, вмонтированного в стену канала

6.6.1 Постановка задачи

Турбулентное течения вокруг куба, вмонтированного в стену, было объектом различных численных и экспериментальных исследований. Результаты показали, что течение характеризуется подковообразным вихрём, возникающим за перед кубом, дугообразным вихрем, возникающим за кубом, отрывом течения на передней и боковых гранях куба и образованием вихревого следа. Однако, большинство численных исследований было выполнено с помощью RANS или LES подходов, а DNS немногочисленны и ограничены очень низкими числами Рейнольдса. Более подробный обзор исследований представлен в [198]. Поскольку эта конфигурация используется для валидации моделей турбулентности, наличие качественных данных DNS расчёта было весьма актуальным.

В данном расчёте с помощью программного комплекса STG-CFD&HTна сетке из 16 млн узлов схемой 2-го порядка моделируется турбулентное течение несжимаемой ньютоновской жидкости вокруг куба, вмонтированного в стену канала. Расчёт выполнялся на суперкомпьютере МВС-100К МСЦ РАН. Число Рейнольдса $Re = 7235$ (по высоте куба h и суммарной скорости входного потока). Схема расчётной области показана на рис. 6.68 сверху. Размеры расчётной области составляют $17h \times 6h \times 3h$ продольному, поперечному и вертикальному направлениям, соответственно. Для простоты постановки на входе используется следующий аналитически заданный профиль

$$U^+ = U/u_\tau = min\left(y^+, k \ln y^+ + B\right) \; ; \qquad V^+ = W^+ = 0, \tag{6.2}$$

где $y^+ = (y/H)Re_\tau$, $u_\tau = Re_\tau \nu / H$, $k = 0.25$ и $B = 5.0$. H – это половина высоты канала (в данном случае $H = 1.5h$). Влияние входных условий изучалось на нескольких типах течения: рассматривалось полностью развитое течение в канале (динамические граничные условия, полученные из расчёта течения в канале), осреднённый профиль течения в канале, и профиль (6.2). Было отмечено, что при достаточном удалении от входной границы ($\gtrsim 5h$) не наблюдалось существенных отличий в решении в области интереса. Поэтому был выбран стационарный аналитический профиль. На выходе заданы конвективные граничные условия. На поверхности куба и стенках канала используются условия твёрдой стенки без проскальзывания. В поперечном направлении на границах, достаточно удалённых от препятствия, используются переодические условия.

В данной постановке во всех трёх пространственных направлениях присутствуют твёрдые поверхности, что требует применения расширения решателя MG-KSFD для уравнения Пуассона, представленного в разделе 4.5. Для построения сетки, вид которой показан на рис. 6.68 снизу слева, выполнялась такая же процедура, как и в предыдущих расчётах, для сгущения сетки на отрезках использовалась функция (6.1). Число узлов сетки $400 \times 200 \times 200$ в продольном, поперечном и вертикальном направлениях, соответственно. Процедура верификации аналогична предыдущим расчётам.

6.6.2 Результаты расчёта

Моментальная картина течения показана на рис. 6.68 снизу справа. Для получения статистики течения выполнялось осреднение по статистически инвариантным преобразованиям: время, одна плоскость симметрии. Осреднённые лини тока в различных сечениях показаны на рис. 6.69. Профили продольной компоненты скорости и её среднего квадратического значения по линиям в вертикальном центральном сечении, показанным на рис. 6.70, представлены на рис. 6.71. На этом рисунке также присутствуют результаты расчёта на грубых сетках с использованием регуляризации для моделирования турбулентности. Более подробно результаты расчёта представлены в [138].

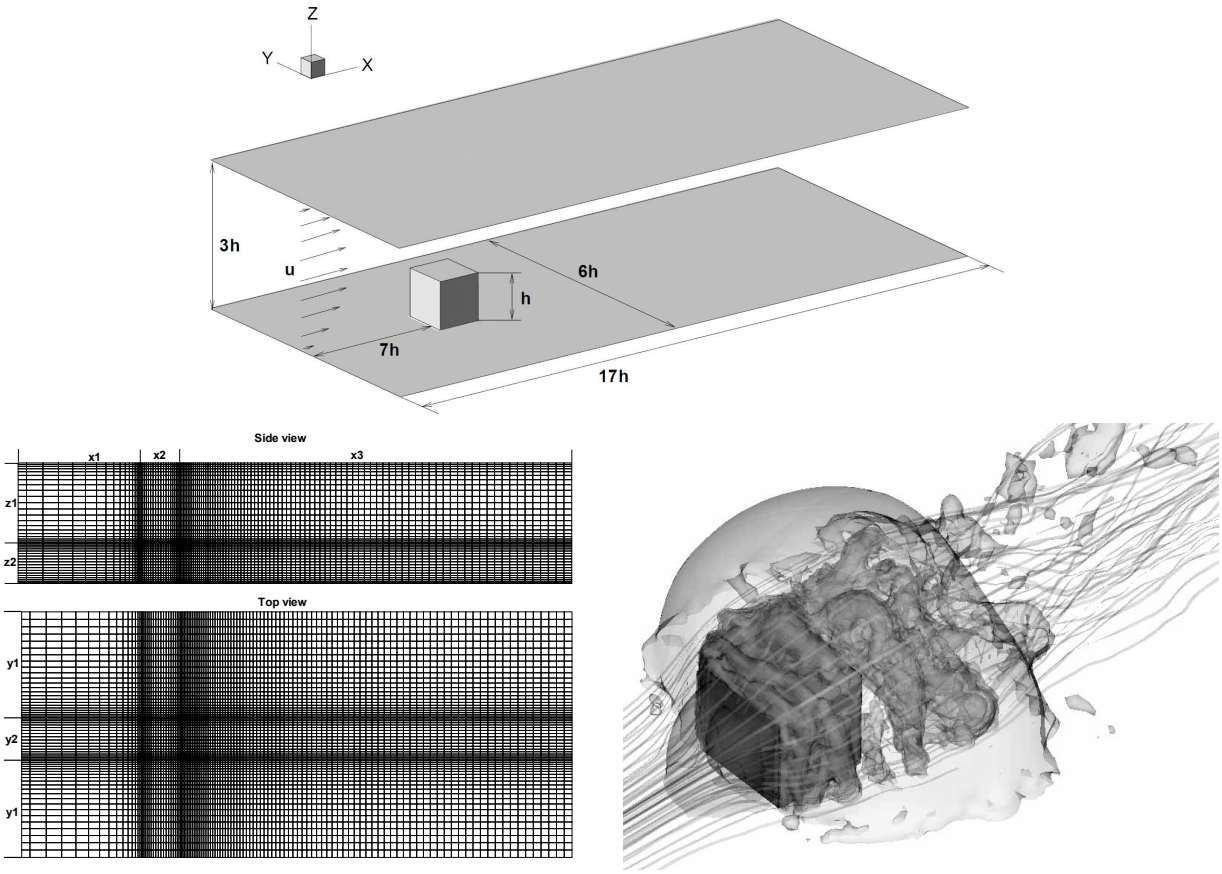


Рисунок 6.68: Схема расчётной области (сверху), вид огрублённой сетки (снизу слева), моментальная картина течения (изоповерхности давления и линии тока – снизу справа)

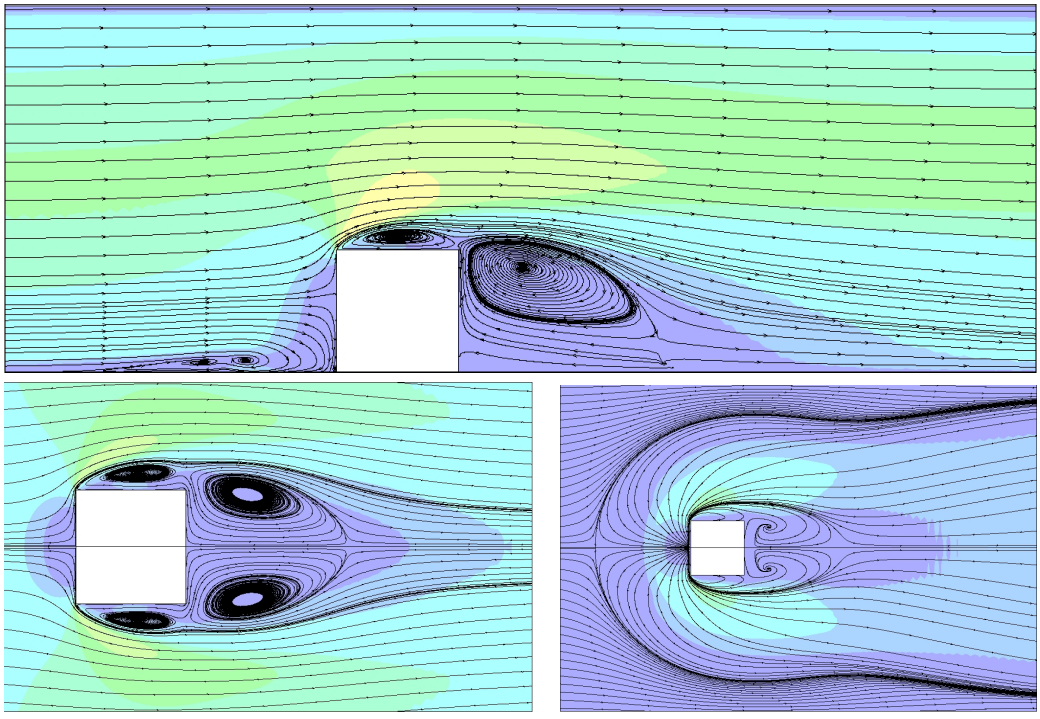


Рисунок 6.69: Линии тока в центральном вертикальном сечении (сверху), в горизонтальных сечениях на высоте середины куба (снизу справа) и вблизи стенки (снизу справа)

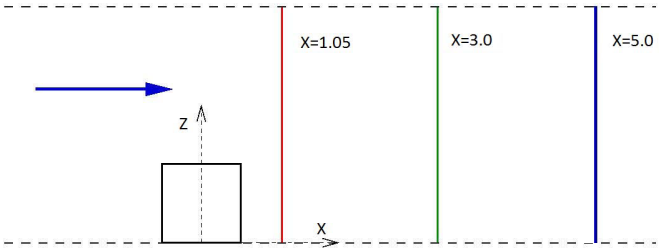


Рисунок 6.70: Позиции вертикальных профилей в центральном сечении

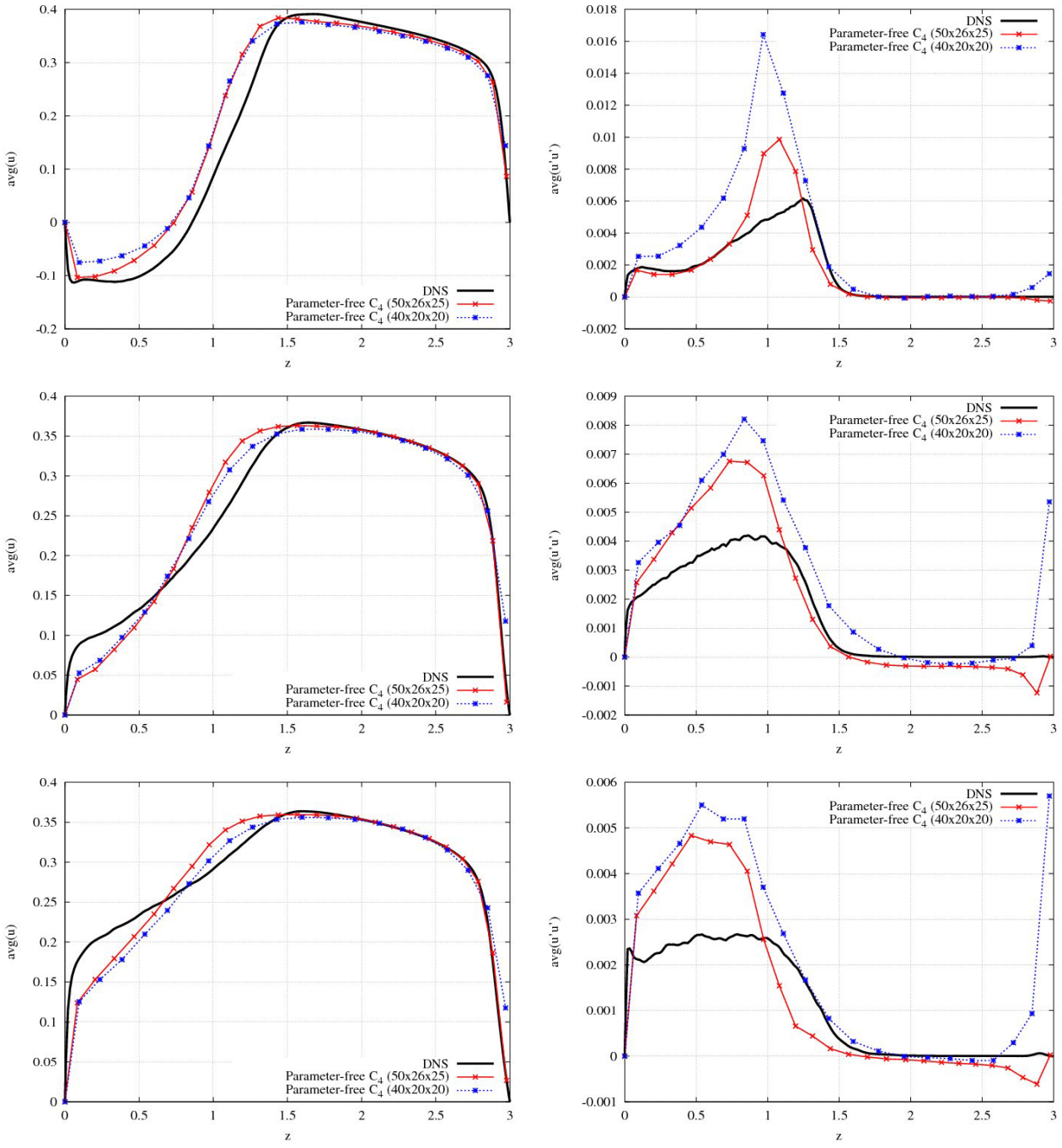


Рисунок 6.71: Профиль продольной компоненты скорости (слева) и её среднего квадратического значения (справа) для позиций (см. рис. 6.70) $x = 1.05$ (сверху), $x = 3$ (по центру), $x = 5$ (снизу)

6.7 Демонстрационные DNS расчёты течений при естественной конвекции

В этом разделе приводятся DNS расчёты, выполняемые автором в настоящее время с помощью программного комплекса STG-CFD&HT, представленного в предыдущей главе. Для этих расчётов на данный момент ещё не завершена обработка результатов и анализ полученных данных. Поэтому в контексте данной работы расчёты приводятся только в качестве примеров, демонстрирующих применимость, производительность и эффективность представленных в работе алгоритмов. Результаты расчётов не входят в результаты данной работы.

6.7.1 DNS течения в закрытой каверне с разнонагретыми стенками

Данный расчёт моделирует аналогичную 6.2 конфигурацию, ограниченную стенками во всех трёх пространственных направлениях. Конфигурация имеет следующие параметры: число Праднтля $Pr = 0.71$ (воздух), соотношение высоты к ширине $A_z = 3.84$, соотношение глубины к ширине $A_x = 0.86$, число Рэлея $Ra = 1.2 \times 10^{11}$ (по высоте каверны L_z). Эта конфигурация соответствует эксперименту [199] и численным исследованиям посредством LES [200]. Целью расчёта было дополнить экспериментальные данные и численные данные, полученные с применением модели турбулентности, данными DNS расчёта на достаточно подробной сетке.

DNS расчёт был выполнен, согласно технологии из 1-й главы, на последовательности сгущающихся сеток, с процедурой верификации постановки задачи аналогично расчёту каверны с периодикой в разделе 6.2. Сетка основного расчёта из 104 млн узлов имела следующие параметры: пространственное разрешение $256 \times 450 \times 900$ и параметры сгущения сетки для функции (6.1) $1.0, 2.0, 1.0$ по глубине, ширине и высоте, соответственно. Общая длина интегрирования по времени составила примерно 600 безразмерных единиц, для накопления статистики течения использовалось примерно 300. Схема расчётной области представлена на рис. 6.72 слева. Расчёт выполнялся на суперкомпьютере МВС-10П МСЦ РАН.

DNS расчёт был успешно завершен, хотя исследование данной конфигурации не в полной мере завершено, в следствие чего здесь приводятся только некоторые предварительные результаты, которые были были представлены в [139]. Исследование подразумевает сопоставление результатов с новым LES расчётом, который ещё не завершен коллегами из LIMSI CNRS, Франция.

В контексте данной работы, расчёт представляет интерес как наиболее крупный пример успешного и достаточно эффективного применения MG-KSFD решателя, представленного в 4-й главе в разделе 4.5. На данной конфигурации MG-расширению требовалось в среднем всего 1.3 итерации для достижения заданной нормы невязки относительно правой части 10^{-3}. Время выполнения одного шага интегрирования по времени на 300 ядрах было менее 3 секунд. Таким образом, производительность решателя для полностью 3D геометрии со сгущением сетки и твёрдыми поверхностями по всем трём осям оказалась сопоставимой с исходным

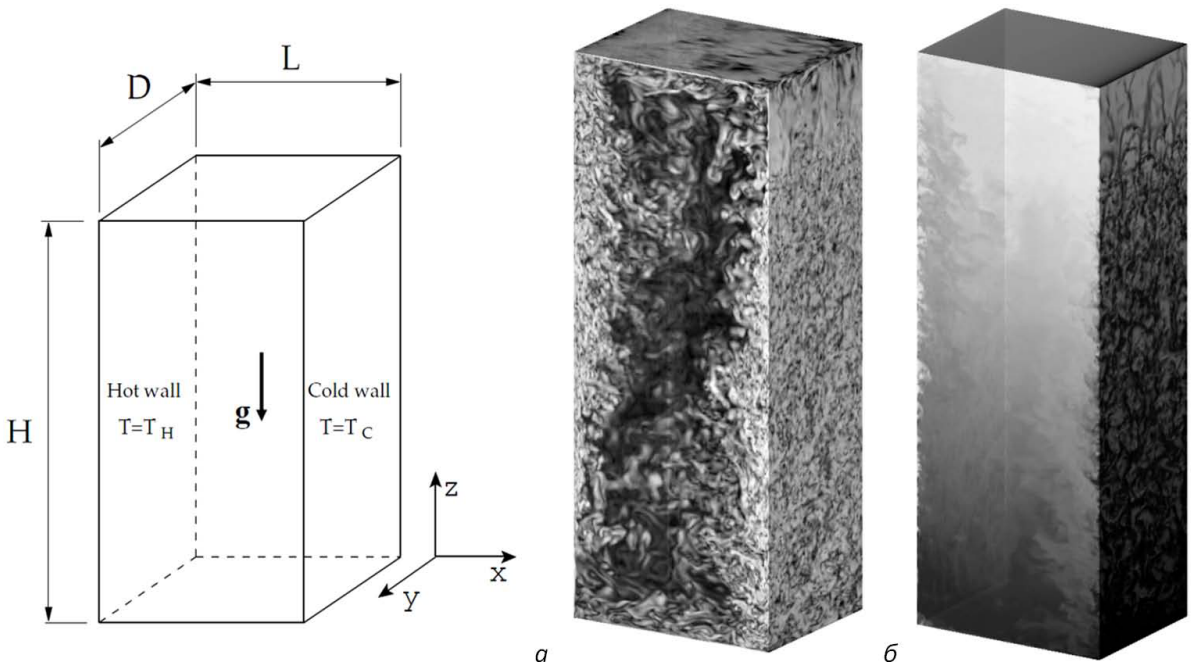


Рисунок 6.72: Схема расчётной области для закрытой каверны со стенками разной температуры (слева) и моментальная картина течения (справа), показаны поля (а) завихренности и (б) температуры

KSFD решателем для задач с однородным периодическим направлением. Существенно более проблематичный для MG-KSFD расчёт был представлен ранее в разделе 6.6.

Моментальная картина течения показана на рис. 6.72 справа. Результаты осреднялись по статистически инвариантным преобразованиям: по времени и по 3-м пространственным симметриям. На рис. 6.73 показано осреднённое поле течения в верхней части каверны. Предварительные результаты расчёта и сопоставление с данными эксперимента [199] и LES [200] показаны на рис. 6.74.

6.7.2 DNS конвекции Рэлея-Бенара

Данный расчёт моделирует естественную конвекцию Рэлея-Бенара. Расчётная область представляет собой параллелепипед с соотношением длины, ширины, высоты $\pi : 1 : 1$. На коротких вертикальных поверхностях заданы периодические граничные условия, на длинных – твёрдая адиабатическая стенка без проскальзывания. Сверху и снизу – холодная и горячая изотермические стенки без проскальзывания, соответственно. Число Рэлея $Ra = 10^10$, число Прандтля равно 0.7. Сетка основного расчёта из 604 млн узлов и имеет следующие параметры: пространственное разрешение $1024 \times 768 \times 768$ по длине, ширине и высоте, соответственно, параметр сгущения сетки по ширине и высоте для функции (6.1) $\gamma = 1.6$. Для расчёта используется программный комплекс STG-CFD&HTи схема 4-го порядка аппроксимации по пространству. В настоящее время достигнут статистически однородный режим течения и выполняется интегрирование по времени для накопления статистики течения. Расчёт выполняется на суперкомпьютере MareNostrum на 784 ядрах Intel Xeon E5-2670 2.6 ГГц

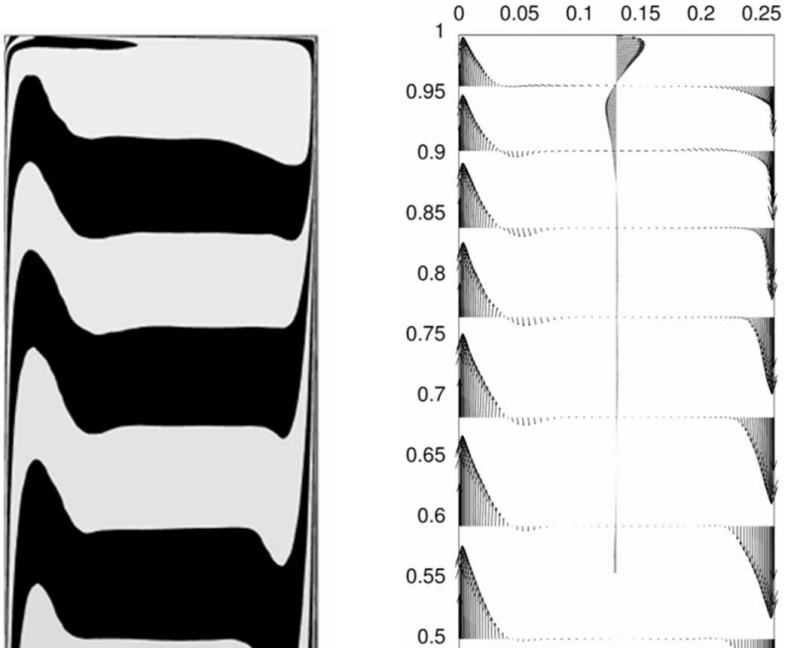


Рисунок 6.73: Осреднённые поля течения в верхней половине каверны: поле температуры (слева) в центральном сечении, изотермы однородно распределены между значениями -0.5 и 0.5, соответствующими температуре холодной и горячей стенки; поле скоростей (справа)

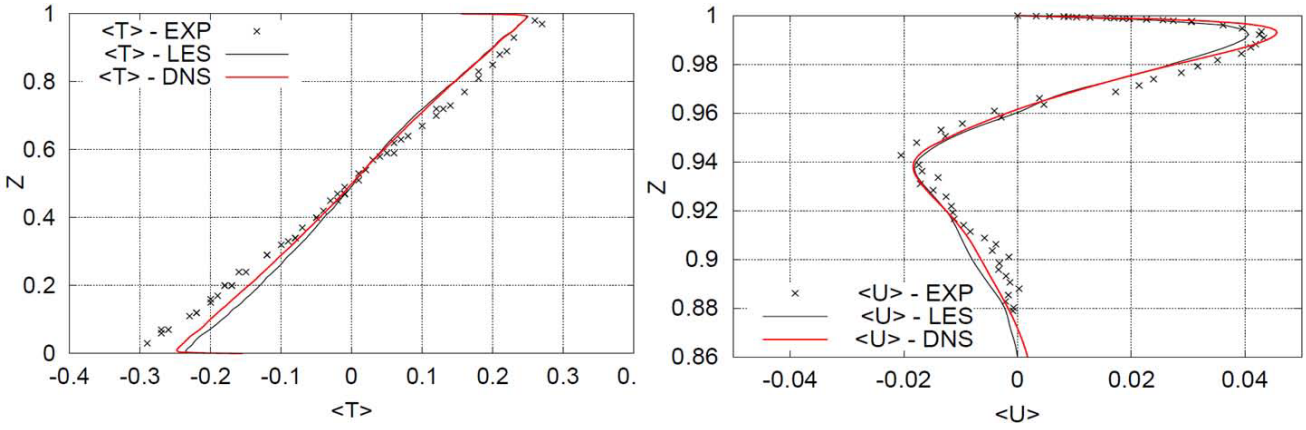


Рисунок 6.74: Осреднённые профили температуры на вертикальной линии по середине горячей стены (слева) и поперечной скорости по вертикальной центральной линии (справа) в сравнении с данными [199, 200] (предоставлено A. Sergent)

(SandyBridge-EP). Среднее время расчёта шага интегрирования времени составляет порядка 4.7 секунд, решение уравнения Пуассона занимает 2.8 секунды (невязка относительно нормы правой части 10^{-3}), из них на решение 1024 плоскостей затрачивается 2.3 сек. Ориентировочная общая вычислительная стоимость расчёта составляет порядка 2.5 млн процессорных часов. Визуализация моментальных картин течения показана на рис. 6.75–6.79 (предоставлено F. Dabbagh). Для визуализации используется ParaView в параллельном режиме на кластере JFF-3 CTTC UPC, я с помощью которого генерируются изображения в том числе высокой чёткости для моментальных картин течения с полным разрешением 604 млн узлов. Для анализа топологии течения использовались 200 моментальных картин течения с полным разрешением (суммарный объём порядка 5 ТБ). Для визуализации динамики течения обработаны 420 моментальных

картин течения (на огрублённой 2 раза по каждому направлению сетке). Отработана техника генерации стереопар и визуализации течения в 3D формате высокой чёткости. Пример вертикальной стереопары для 3D видео формата 16:9 1080p показан на рис. 6.78.

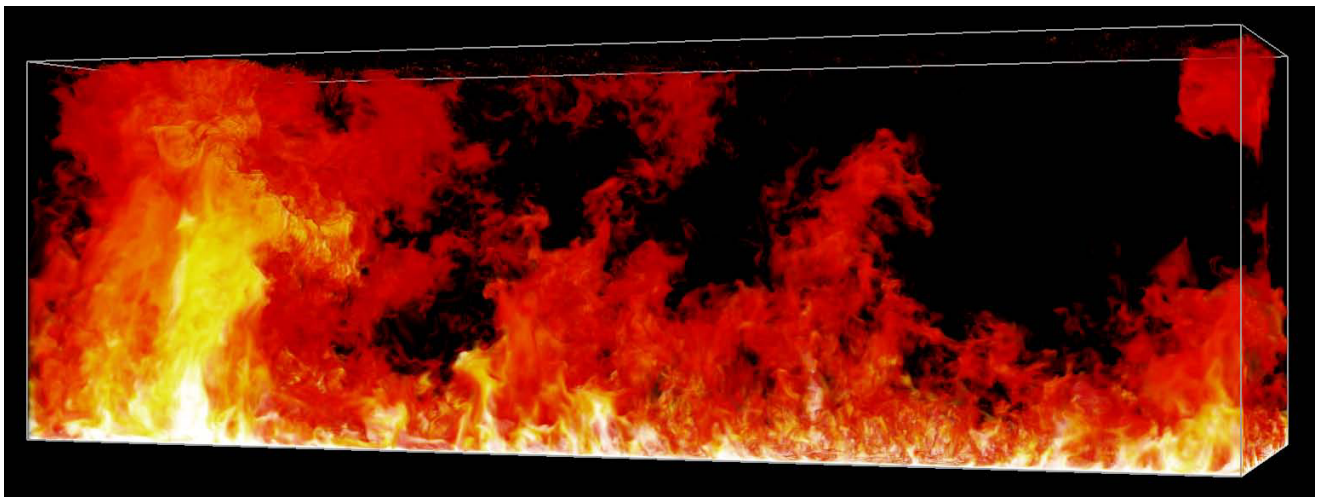


Рисунок 6.75: Визуализация “тёплых” структур течения по скорости $\mathbf{u}$ и температуре T на основе модуля тензора $grad(\mathbf{u}T)$

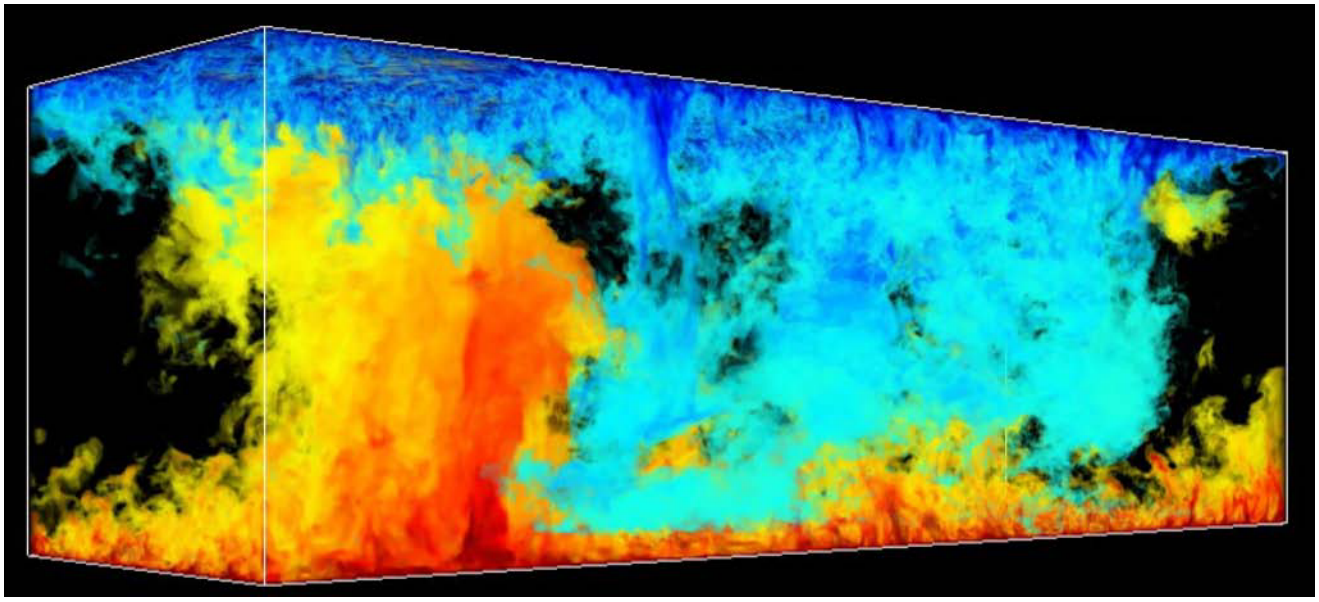


Рисунок 6.76: Аналогичная предыдущему рисунку визуализация тёплой и холодной частей течения

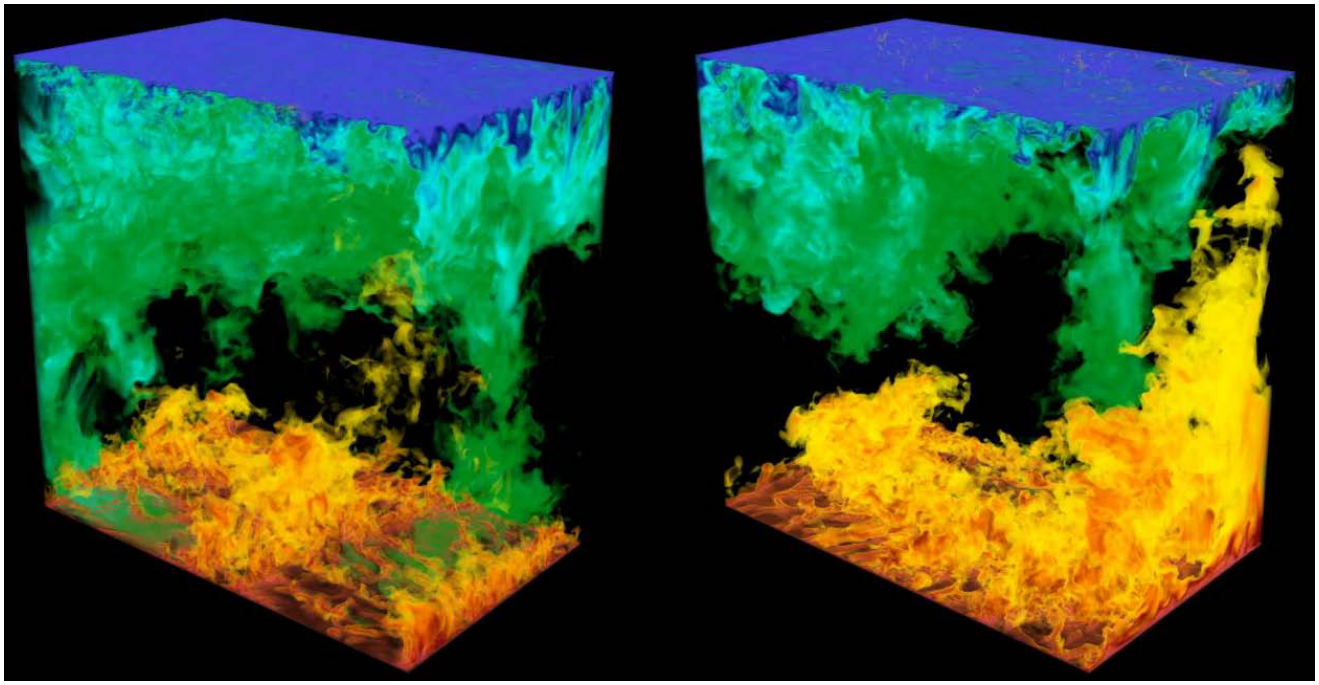


Рисунок 6.77: Теплые и холодные структуры течения в увеличенных фрагментах расчётной области

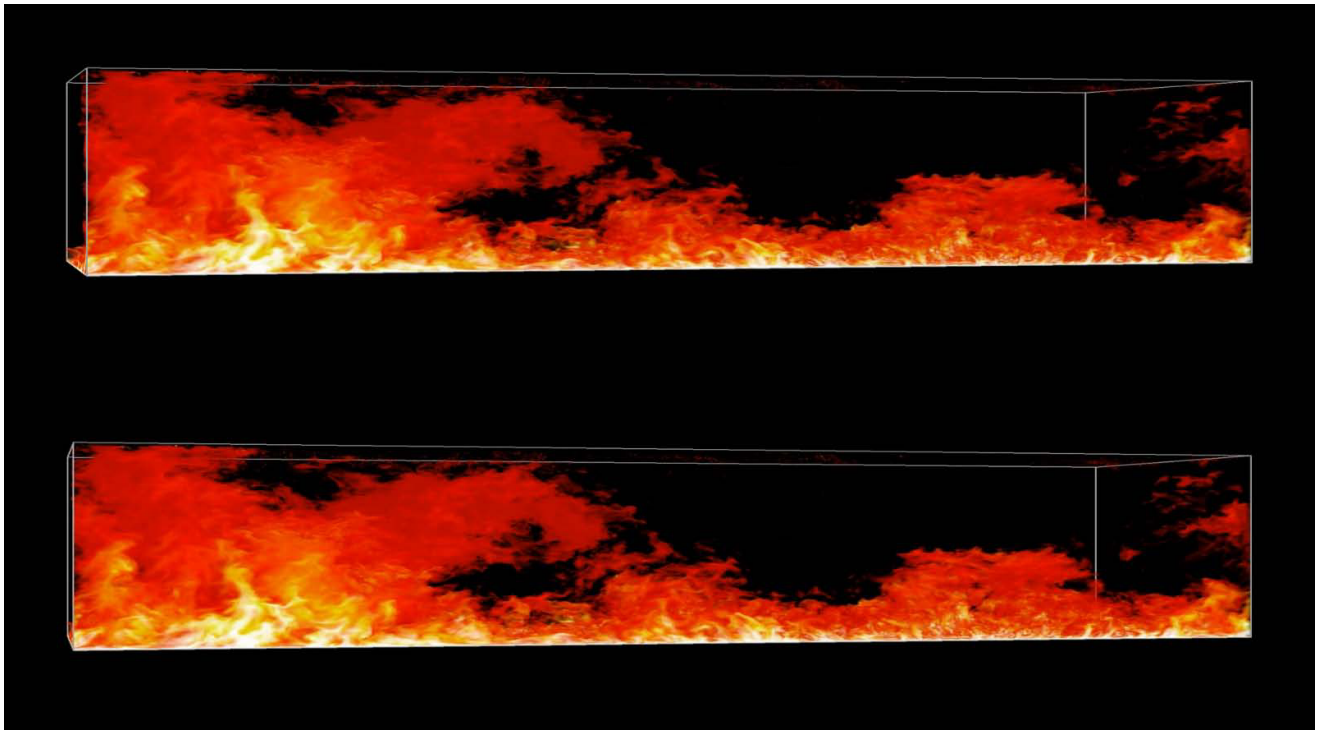


Рисунок 6.78: Пример стереопары для 3D видео

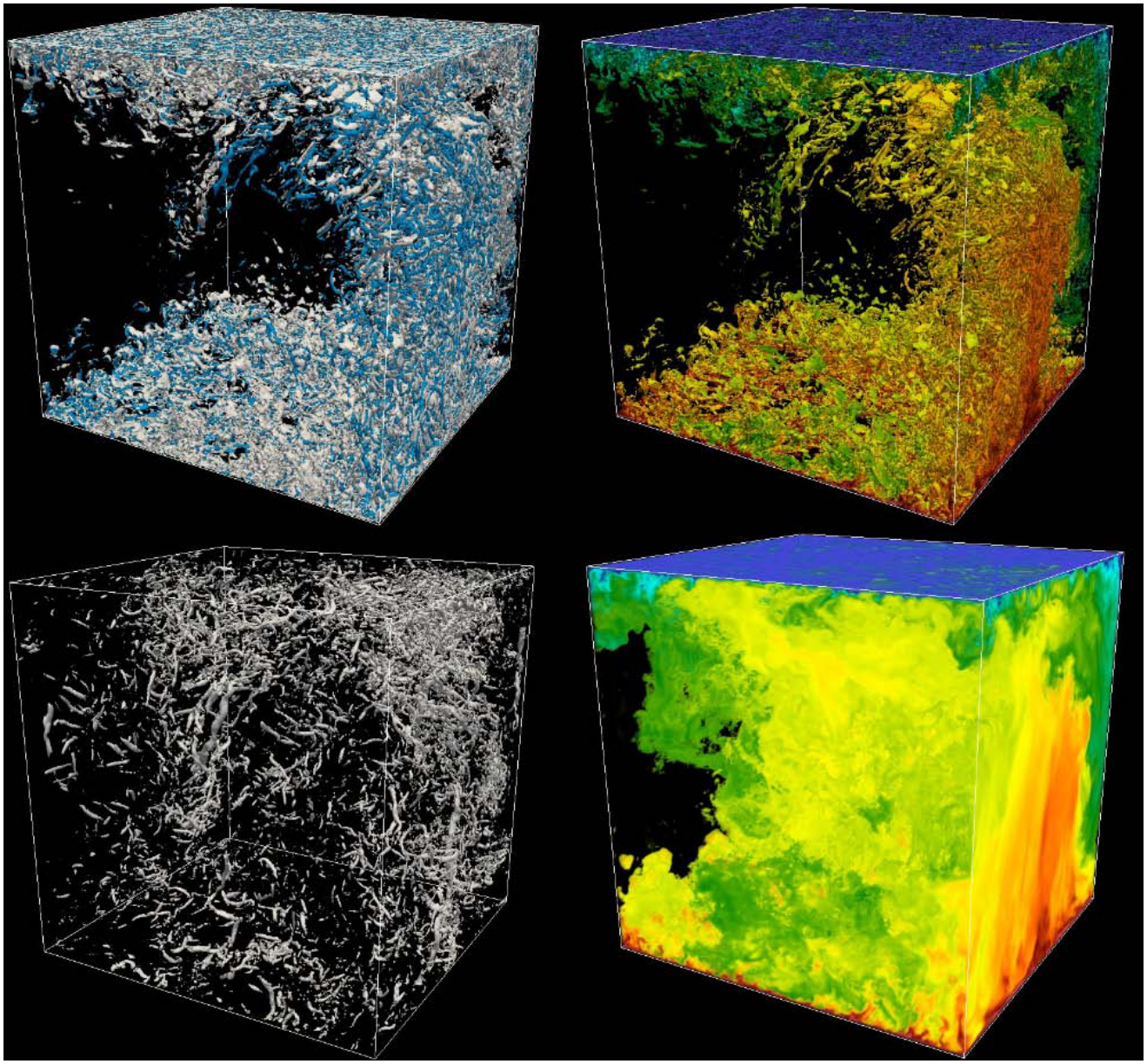


Рисунок 6.79: Визуализация турбулентных структур во фрагменте расчётной области: сверху слева – наибольшие положительные (синий) и отрицательные (white) значения Q-инварианта тензора $grad(\mathbf{u}T)$; сверху справа – тоже самое в раскраске по контурам температуры; снизу слева – наибольшие значения Q-инварианта тензора $grad(\mathbf{u})$; снизу справа – объёмное представление тепловых течений на основе модуля тензора $grad(\mathbf{u}T)$

Заключение

1. Разработан новый комплексный подход к математическому моделированию турбулентных течений на суперкомпьютерах с экстра-массивным параллелизмом. Он включает методику разработки алгоритма в рамках многоуровневой параллельной модели, технологию гетерогенной программной реализации для гибридных суперкомпьютеров с массивно-параллельными ускорителями и технологию выполнения крупных расчётов.

2. Предложен эффективный параллельный алгоритм повышенной точности с многоуровневым распараллеливанием для крупномасштабных расчётов задач аэродинамики и аэроакустики на неструктурированных сетках.

3. Разработан параллельный метод решения уравнения Пуассона для моделирования несжимаемых турбулентных течений, позволяющий проводить расчёты на десятках тысяч процессоров и на гибридных суперкомпьютерах с ускорителями различной архитектуры. Принципиальное отличие нового метода от известных аналогов – высокая эффективность при расчётах задач с одним периодическим направлением при использовании схем произвольного порядка точности.

4. На основе разработанной технологии и алгоритмов созданы программные комплексы для крупномасштабных расчётов с использованием суперкомпьютеров с десятками тысяч процессоров, а также гибридных суперкомпьютеров:

   4.1 NOISEtte для математического моделирования турбулентного течения сжимаемого газа и создаваемых им акустических полей с повышенной точностью на неструктурированных сетках;

   4.2 STG-CFD&HTдля математического моделирования турбулентного течения несжимаемой жидкости и явлений теплопереноса на основе алгоритма повышенной точности на структурированных сетках.

5. Выполнены крупномасштабные, в том числе рекордные, расчёты фундаментальных задач по моделированию сжимаемых и несжимаемых турбулентных течений. Получен широкий набор данных для разработки и валидации перспективных моделей турбулентности.

Литература

1. *A. Ramirez*. The Mont-Blanc architecture // International Supercomputing Conference (ISC 2012). — 2012. — June 17-21. — Hamburg, Germany.

2. *Воеводин В.В. Воеводин Вл.В.* Параллельные вычисления. — СПб.: БХВ-Петербург, 2002. — 608 с.

3. *Grama A. Karypis G. Kumar V. Gupta A.* Introduction to parallel computing (2nd edn). — Addison-Wesley: Reading, MA, 2003.

4. *Василевский Ю. В. Коньшин И. Н. Копытов Г. В. Терехов К. М.* INMOST - программная платформа и графическая среда для разработки параллельных численных моделей на сетках общего вида: Учебное пособие. — М.: Издательство Московского университета, 2013. — 144 с.

5. *Aubry R. Houzeaux G. Vazquez M. Cela J. M.* Some useful strategies for unstructured edge-based solvers on shared memory machines // *International Journal for Numerical Methods in Engineering*. — 2010. — Vol. 85. — Pp. 537–561.

6. *Богданов П.Б. Ефремов А.А. Горобец А.В. Суков С.А.* Применение планировщика для эффективного обмена данными на суперкомпьютерах гибридной архитектуры с массивно-параллельными ускорителями // *Вычислительные методы и программирование*. — 2013. — Т. 85. — С. 122–134.

7. *Волков К.Н. Емельянов В.Н.* Моделирование крупных вихрей в расчетах турбулентных течений. — М.: ФИЗМАТЛИТ, 2008.

8. *Shur M.L. Spalart P.R. Strelets M.Kh. Travin A.K.* A hybrid RANS-LES approach with delayed-DES and wall-modeled LES capabilities // *Int. J. of Heat and Fluid Flow*. — 2008. — Vol. 29(6). — Pp. 1638–1649.

9. *Капорин И. Е. Милюкова О. Ю.* Оптимизация факторизованных предобусловливаний метода сопряженных градиентов для решения систем линейных алгебраических уравнений с симметричной положительно определенной матрицей // *Препринты ИПМ*. — 2013. — № 013. — С. 17.

10. *Жуков В.Т. Новикова Н.Д. Феодоритова О.Б.* Параллельный многосеточный метод для решения эллиптических уравнений // *Матем. моделирование*. — 2014. — Т. 26:1. — С. 55–68.

11. *Baker A.H. Falgout R.D. Kolev Tz.V., U.M. Yang*. Multigrid smoothers for ultraparallel computing // *SIAM Journal on Scientific Computing*. — 2011. — Vol. 33 (5). — Pp. 2864–2887.

12. *Abalakin I.V. Bakhvalov P.A., T.K. Kozubskaya*. Edge-based reconstruction schemes for prediction of near field flow region in complex aeroacoustic problems // *Int. J. of Aeroacoustics*. — 2014. — Vol. 13, no. 3&4. — Pp. 207–234.

13. *А. Бахвалов П.* Схема с квазиодномерной реконструкцией переменных на сетках из выпуклых многоугольников для решения задач аэроакустики // *Матем. моделирование*. — 2013. — Т. 9. — С. 95–108.

14. *В. Горобец А.* Параллельная технология численного моделирования задач газовой динамики алгоритмами повышенной точности // *Журнал вычислительной математики и математической физики*. — 2015. — Т. 55, № 4. — С. 641–652.

15. *В. Горобец А.* Методика выполнения крупномасштабных расчетов задач газовой динамики // *Математическое моделирование. (в стадии рецензирования)*.

16. *Горобец А.В. Суков С.А. Богданов П.Б.* На пути к освоению гетерогенных супервычислений в газовой динамике // *Информационные технологии и вычислительные системы*. — 2013. — Т. 4. — С. 23–34.

17. *Богданов П.Б. Горобец А.В. Суков С.А.* Адаптация и оптимизация базовых операций газодинамического алгоритма на неструктурированных сетках для расчетов на массивно-параллельных ускорителях // *ЖВМ и МФ*. — 2013. — Т. 53, № 8. — С. 1360–1373.

18. Message Passing Interface Forum. — MPI: A Message-Passing Interface Standard, Version 3.0, 2012. — September. — (URL: http://www.mpi-forum.org/docs/mpi-3.0/mpi30-report.pdf).

19. OpenMP Architecture Review Board. — OpenMP Application Program Interface, Version 4.0, 2013. — July. — (URL: http://www.openmp.org/mp-documents/OpenMP4.0.0.pdf).

20. Intel Corporation. — Intel® Cilk™ Plus Language Extension Specification, Version 1.2 (2013-09-06), 2013. — (URL: https://www.cilkplus.org/sites/default/files/open_specifications/Intel_Cilk_plus_lang_spec_1.2.htm).

21. Intel Corporation. — A Guide to Vectorization with Intel C++ Compilers, 2012. — (URL: http://download-software.intel.com/sites/default/files/8c/a9/CompilerAutovectorizationGuide.pdf).

22. The OpenACC™ Application Programming Interface, Version 2.0, 2013. — August. — (URL: http://www.openacc.org/sites/default/files/OpenACC.2.0a_1.pdf).

23. *Maxim Milakov Peter Messmer Thomas Bradley*. Accelerating NEMO with OpenACC // GPU technology conference 2013. —). — 2013. — March 18-21. — San-Jose, California, USA, (URL: http://on-demand.gputechconf.com/gtc/2013/presentations/S3209-Accelerating-NEMO-with-OpenACC.pdf).

24. *Ramanan Sankaran Jackie Chen Ray Grout John Levesque*. Hybridization of a Direct Numerical Simulation Software for Massively Parallel Accelerator-based Architectures // Extreme Scaling Workshop 2012. —). — 2012. — June 15-16. — Chicago, Illinois, USA, (URL: https://www.xsede.org/documents/271087/369160/ExtScale-Sankaran.pdf).

25. NVIDIA Corporation. — NVIDIA CUDA C Programming Guide, Version 6.5, 2014. — (URL: http://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf).

26. Khronos OpenCL Working Group. — The OpenCL Specification Version 2.0, 2014. — (URL: https://www.khronos.org/registry/cl/specs/opencl-2.0.pdf).

27. *C. Augonnet S. Thibault R. Namyst, Wacrenier P.-A.* StarPU: A Unified Platform for Task Scheduling on Heterogeneous Multicore Architectures // Concurrency and Computation: Practice and Experience, Special Issue: Euro-Par 2009. — Vol. 23. — 2011. — February. — Pp. 187–198.

28. *Thomas B. Jablin Prakash Prabhu James A. Jablin Nick P. Johnson Stephen R. Beard, August David I.* Automatic CPU-GPU Communication Management and Optimization // Proceedings of the 32nd ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI). — 2011. — June. — (URL: http://liberty.princeton.edu/Publications/pldi11_cuda.pdf).

29. *Fengguang Song Jack Dongarra*. A scalable framework for heterogeneous GPU-based clusters // SPAA'12 Proceedings of the 24th ACM symposium on Parallelism in algorithms and architectures. — 2011. — Pp. 91–100. — (URL: http://web.eecs.utk.edu/ song/publication/spaa12.pdf).

30. *Bogdanov P. B. Efremov A. A.* Programming infrastructure of heterogeneous computing based on OpenCL and its applications // GPU Technology Conference GTC-2013. — 2013. — March 18-21. — San Jose, California, USA (http://on-demand.gputechconf.com/gtc/2013/presentations/S3410-Programming-Infrastructure-Heterogeneous-Computing-OpenCL.pdf).

31. *Абалакин И.В. Козубская Т.К.* Схема на основе реберно-ориентированной квазиодномерной реконструкции переменных для решения задач аэродинамики и аэроакустики на неструктурированных сетках // *Математическое моделирование*. — 2013. — Т. 25, № 8. — С. 109–136.

32. *Gorobets A. Trias F.X. Oliva A.* A parallel MPI+OpenMP+OpenCL algorithm for hybrid supercomputations of incompressible flows // *Computers and Fluids*. — 2013. — Vol. 88. — Pp. 764–772.

33. *G. Oyarzun R. Borrell A. Gorobets O. Lehmkuhl, Oliva A.* Hybrid MPI-CUDA Implementation of Unstructured Navier-stokes Solver focusing on CG Preconditioners // 26-th Parallel Computational Fluid Dynamics (ParCFD) Conference. — 2014. — May 20–22. — Trondheim, Norway.

34. *Soria M. Perez-Segarra C.D. Oliva A.* A Direct Parallel Algorithm for the Efficient Solution of the Pressure-Correction Equation of Incompressible Flow Problems Using Loosely Coupled Computers // *Numerical Heat Transfer, Part B*. — 2002. — Vol. 41. — Pp. 117–138.

35. *Schloegel K. Karypis G. Kumar V.* Parallel static and dynamic multi-constraint graph partitioning // *Concurrency and Computation: Practice and Experience*. — 2002. — Vol. 14, no. 3. — Pp. 219–240.

36. *F. Pellegrini*. — PT-Scotch and libPTScotch 6.0 User's Guide, 2012. — (URL: https://gforge.inria.fr/docman/view.php/248/8261/ptscotch_user6.0.pdf).

37. *Е.Н. Головченко*. Параллельный пакет декомпозиции больших сеток // *Математическое моделирование*. — 2011. — Т. 23, № 10. — С. 3–18.

38. *B. Krasnopolsky*. The reordered BiCGStab method for distributed memory computer systems // *ICCS 2010, Procedia Computer Science*. — 2010. — Vol. 1. — Pp. 213–218.

39. *Cuthill E. McKee J.* Reducing the bandwidth of sparse symmetric matrices // *ACM '69 Proceedings of the 1969 24th national conference*. — 1969. — Pp. 157–172.

40. *Горобец А.В. Суков С.А. Железняков А.О. Богданов П.Б. Четверушкин Б.Н.* Применение GPU в рамках гибридного двухуровневого распараллеливания MPI+OpenMP на гетерогенных вычислительных системах // *Вестник ЮУрГУ*. — 2011. — Т. 242, № 25. — С. 76–86.

41. *Monakov A. Lokhmotov A. Avetisyan A.* Automatically Tuning Sparse Matrix-Vector Multiplication for GPU Architectures // *High Performance Embedded Architectures and Compilers, Lecture Notes in Computer Science*. — 2010. — Vol. 5952. — Pp. 111–125.

42. *Jaramillo J. E. Trias F. X. Gorobets A. Perez-Segarra C. D. Oliva A.* DNS and RANS modelling of a Turbulent Plane Impinging Jet // *Int. J. of Heat and Mass Transfer*. — 2012. — Vol. 55. — Pp. 789–801.

43. *Yilmaz E. Aliabadi S.* Surface conformed linear mesh and data subdivision technique for large-scale flow simulation and visualization in Variable Intensity Computational Environment // *Computers and Fluids*. — 2013. — Vol. 80. — Pp. 388–402.

44. *P.D. Welch*. The use of fast Fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. // *IEEE Trans. Audio Electroacoust. AU-15*. — 1967. — Pp. 70–73.

45. *Heinzel G. Rudiger A. Schilling R.* Spectrum and spectral density estimation by the Discrete Fourier transform (DFT), including a comprehensive list of window functions and some new at-top windows. — 2002. — (URL: http://hdl.handle.net/11858/00-001M-0000-0013-557A-5).

46. *Ffowcs Williams J.E. Hawkings D.L.* Sound generated by turbulence and surfaces in unsteady motion // *Philosophical Transactions of the Royal Society*. — 1969. — Vol. 1151. — Pp. 321–342.

47. *Бахвалов П.А. Козубская Т.К. Корнилина Е.Д. Морозов А.В. Якобовский М.В.* Технология расчетов акустических пульсаций в дальнем поле течения // *Матем. моделирование*. — 2011. — Т. 23, № 11. — С. 33–47.

48. *Spalart P.R. Shur M.L.* Variants of the Ffowcs Williams - Hawkings equation and their coupling with simulations of hot jets // *International Journal of Aeroacoustics*. — 2009. — Vol. 8, no. 5. — Pp. 477–492.

49. *Абалакин И.В. Бахвалов П.А. Горобец А.В. Дубень А.П. Козубская Т.К.* Параллельный программный комплекс NOISETTE для крупномасштабных расчетов задач аэродинамики и аэроакустики // *Вычислительные методы и программирование*. — 2012. — Т. 13. — С. 110–125.

50. *Т.К. Козубская*. Разработка моделей и методов повышенной точности для численного исследования задач прикладной аэроакустики: диссертация д.ф.-м.н.: 05.13.18. ИПМ им. М.В. Келдыша РАН, Москва. — 2010.

51. *P.L. Roe*. Approximate Riemann Solvers, Parameter Vectors and Difference Schemes // *J. Comput. Phys.* — 1981. — Vol. 43, no. 2. — Pp. 357–372.

52. *L.C. Huang*. Pseudo-unsteady difference schemes for discontinuous solution of steady-state. One-dimensional fluid dynamics problems // *J. Comput. Phys.* — 1981. — Vol. 42, no. 1. — Pp. 195–211.

53. *Абалакин И.В. Козубская Т.К.* Многопараметрическое семейство схем повышенной точности для линейного уравнения переноса // *Матем. моделирование*. — 2007. — Т. 19, № 7. — С. 56–66.

54. *Dervieux A. Debiez C.* Mixed element volume MUSCL methods with weak viscosity for steady and unsteady flow calculation // *Computer and Fluids*. — 2000. — Vol. 29. — Pp. 89–118.

55. *Бахвалов П. А. Козубская Т.К.* Экономичная формулировка схем с квазиодномерной реконструкцией переменных // *Препринты ИПМ им. М.В.Келдыша*. — 2013. — № 89.

56. *П.А.Бахвалов Т.К.Козубская*. Схема с квазиодномерной реконструкцией переменных, определенных в центрах элементов трёхмерной неструктурированной сетки // *Матем. моделирование (принята к публикации)*. — 2016.

57. *Ch. Hirsch*. Numerical Computation of Internal and External Flows. Vol.2: Computational Methods for Inviscid and Viscous Flows. — John Wiley & Sons, 1990.

58. *A. Jameson*. Numerical Solution of the Euler Equations for Compressible inviscid Fluids, Numerical Methods for the Euler Equations of Fluid Dynamics. — SIAM, Philadelphia, 1985. — 199 pp.

59. *M.-H. Lallemand*. Schemas decentres Multigrilles pour la Resolution des Equations D'Euler en Elements Finis. — Ph.D. Thesis, L'Universite de Provence Saint Charles, 1988.

60. *J. Frohlich D. von Terzi*. Hybrid LES/RANS Methods for the Simulation of Turbulent Flows // *Progress in Aerospace Sciences*. — 2008. — Vol. 44. — Pp. 349–377.

61. *Spalart P. R. Allmaras S. R.* A One-Equation Turbulence Model for Aerodynamic Flows // *Recherche Aerospatiale*. — 1994. — no. 1. — Pp. 5–21.

62. *C. Wilcox D.* Formulation of the k-omega Turbulence Model Revisited // *AIAA Journal*. — 2008. — Vol. 46, no. 11. — Pp. 2823–2838.

63. *K.-Y. Chien*. Predictions of Channel and Boundary-Layer Flows with a Low-Reynolds-Number Turbulence Model // *AIAA Journal*. — 1982. — Vol. 20, no. 1. — Pp. 33–38.

64. *R. Menter F.* Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications // *AIAA Journal*. — 1994. — Vol. 32, no. 8. — Pp. 1598–1605.

65. *Smagorinsky J.* General circulation experiments with the primitive equations // *Mon. Weather Rev.* — 1963. — Vol. 91. — P. 99–164.

66. *Nicoud F., Ducros F.* Subgrid-scale stress modelling based on the square of the velocity gradient tensor // *Flow, Turbul. Combust.* — 1999. — Vol. 62, no. 3. — P. 183–200.

67. *Vreman A.W.* An eddy-viscosity subgrid-scale model for turbulent shear flow: Algebraic theory and applications // *Phys. Fluids*. — 2004. — Vol. 16, no. 10. — P. 3670–3681.

68. *Verstappen R.* When does eddy viscosity damp subfilter scales sufficiently? // *J. Sci. Comput.* — 2011. — Vol. 49, no. 1. — P. 94–110.

69. *F. Nicoud H. B. Toda O. Cabrit S. Bose, Lee J.* Using singular values to build a subgrid-scale model for large eddy simulations // *Phys. Fluids*. — 2011. — Vol. 23, no. 8. — P. 085106.

70. *F.X. Trias D. Folch A. Gorobets, Oliva A.* Building proper invariants for eddy-viscosity subgrid-scale models // *Physics of Fluids*. — 2015. — Vol. 27. — P. 065103.

71. *P.R. Spalart W.H. Jou M.Kh. Strelets S.R. Allmaras*. Comments оп the feasibility of LES for wings, and оп a hybrid RANS/LES approach // *Proc. of the First AFOSR Int. Conf. on DNS/LES, Ruston, USA*. — 1997. — Pp. 137–148.

72. *Spalart P.R. Deck S. Shur M. Squires K. Strelets M. Travin A.* A new version of detached-eddy simulation, resistant to ambiguous grid densities // *Theor. Comput. Fluid Dyn.* — 2006. — Vol. 20, no. 3. — Pp. 181–195.

73. *Mikhail L. Shur Philippe R. Spalart Mikhail Kh. Strelets Andrey K. Travin*. An Enhanced Version of DES with Rapid Transition from RANS to LES in Separated Flows // *Flow, Turbulence and Combustion. – DOI: 10.1007/s10494-015-9618-0.* — 2015.

74. *А.П. Дубень*. Численное оделирование сложных пристеночных турбулентных течений на неструктурированных сетках: диссертация к.ф.-м.н.: 05.13.18. ИПМ им. М.В. Келдыша РАН, Москва. — 2014.

75. *А.П. Дубень*. Вычислительные технологии для моделирования сложных пристеночных турбулентных течений на неструктурированных сетках // *Математическое моделирование*. — 2013. — Т. 25, № 9. — С. 4–16.

76. *Tam C.K.W. Webb J.C.* Dispersion-relation-preserving finite difference schemes for computational acoustics // *J. Comput. Phys.* — 1993. — Vol. 107, no. 2. — Pp. 262–281.

77. *Saad Yousef*. Iterative Methods for Sparse Linear Systems (2nd edn). — SIAM, 2003.

78. *Soukov S. A. Iakobovski M.V. Boldyrev S.N.* Big Unstructured Mesh Processing on Multiprocessor Computer Systems // Proc. of the Parallel CFD 2003 Conference. — Elsevier, 2004. — May 13-15. — Pp. 73–79. — Moscow, Russia.

79. *И.В. Абалакин*. Использование кинетического подхода при построении разностных схем газовой динамики: диссертация к.ф.-м.н.: 05.13.18. ИММ РАН, Москва. — 2007.

80. *П.А. Бахвалов*. Развитие схем на основе квазиодномерного подхода для решения задач аэроакустики на неструктурированных сетках: диссертация к.ф.-м.н.: 05.13.18. МФТИ, Москва. — 2013.

81. *А.В. Горобец*. Параллельные алгоритмы повышенной точности для численного моделирования задач газовой динамики и аэроакустики: диссертация к.ф.-м.н.: 05.13.18. ИММ РАН, Москва. — 2007.

82. *B.A. Singer, Y. Guo*. Development of computational aeroacoustics tools for airframe noise calculations // *International Journal of Computational Fluid Dynamics*. — 2004. — Vol. 18, no. 6. — Pp. 455–469.

83. *Abalakin I. Dervieux A. Kozubskaya T.* Computational study of mathematical models for noise DNS // *AIAA Paper 2002-2585*. — 2002.

84. *Vos J.B. Rizzi A. Darracq D., E.H. Hirschel*. Navier-Stokes solvers in European aircraft design // *Progress in Aerospace Sciences*. — 2002. — Vol. 38, no. 8. — Pp. 601–697.

85. *Копьев В.Ф. Титарев В.А. Беляев И.В.* Разработка методологии расчета шума винтов с использованием суперкомпьютеров // *Ученые записки ЦАГИ*. — 2014. — Т. XLV, № 2. — С. 78–106.

86. *Shur M. Strelets M. Travin A.* High-order implicit multi-block Navier-Stokes code: Ten-years experience of application to RANS/DES/LES/DNS of turbulent flows // 7th Symposium on Overset Composite Grids & Solution Technology. — 2004. — Huntington Beach, California, URL: http://cfd.spbstu.ru/agarbaruk/c/document_library/DLFE-42505.pdf.

87. *M. A. Olshanskii K. M. Terekhov Yu. V. Vassilevski*. An octree-based solver for the incompressible Navier-Stokes equations with enhanced stability and low dissipation // *Computers & Fluids*. — 2013. — Vol. 84. — P. 231–246.

88. *Spalart P.R. Allmaras S.R.* A one-equation turbulence model for aerodynamic flows // 30th Aerospace Science Meeting. — AIAA Paper 92-0439, 1992. — May 20–22. — Reno, Nevada.

89. *Haase W. Braza M. Revell A.* DESider — A European Effort on Hybrid RANS-LES Modelling. — Springer, 2009.

90. *Boiron O. Chiavassa G. Donat R.* A high-resolution penalization method for large Mach number flows in the presence of obstacles // *Computers and Fluids*. — 2009. — Vol. 38, no. 3. — Pp. 703–714.

91. *E. Brown-Dymkoski N. Kasimov O.V. Vasilyev*. A Characteristic-Based volume penalization method for arbitrary mach flows around solid obstacles // *Journal of Computational Physics*. — 2014. — Vol. 262. — Pp. 344–357.

92. *C. Bailly P. Lafon S. Candel*. A Stochastic Approach to Compute Noise Generation and Radiation of Free Turbulent Flows // *AIAA Paper 95-092-1-6*. — 1995.

93. *T. Barth*. Numerical methods for conservation laws on structured and unstructured meshes // *VKI for Fluid Dynamics, Lectures series, 2003-03*. — 2003.

94. *Годунов С. К.* Разностный метод численного расчета разрывных решений уравнений гидродинамики // *Матем. сб.* — 1959. — Vol. 47(89). — P. 271–306.

95. *Turkel Eli*. Preconditioned methods for solving the incompressible and low speed compressible equations // *Journal of computational physics*. — 1987. — Vol. 72. — Pp. 277–298.

96. *Kozubskaya T. Abalakin I. Bakhalov P.* Edge-based methods in CAA // VKI lecture series. — 2013. — Von Karman Institute for Fluid Dynamics.

97. *I.V. Abalakin P.A. Bakhvalov T.K. Kozubskaya*. WENO-EBR scheme for solving aerodynamics and aeroacoustics problems on unstructured meshes // International Workshop “Computational Experiment in Aeroacoustics”. — 2014. — September 24-27. — Svetlogorsk, Russia.

98. *Дородницын Л. В.* Неотражающие граничные условия и численное моделирование задач обтекания // *Ж. вычисл. матем. и матем. физ.* — 2011. — Т. 51, № 1. — С. 152–169.

99. *Gorobets A.V. Abalakin I.V. Kozubskaya T.K.* Technology of parallelization for 2D and 3D CFD/CAA codes based on high-accuracy explicit methods on unstructured meshes // *Lecture Notes in Computational Science and Engineering, PCFD 2007*. — 2009. — Vol. 67. — Pp. 253–260.

100. *Савин Г.И. Четверушкин Б.Н. Суков С.А. Горобец А.В. Козубская Т.К. Вдовикин О.И. Шабанов Б.М.* Моделирование задач газовой динамики и аэроакустики с использованием ресурсов суперкомпьютера МВС-100К // *Доклады Академии Наук*. — 2008. — Т. 423, № 3. — С. 312–315.

101. *Kopiev V. Abalakin I. Faranosov G. Gorobets A. Kozubskaya T. Ostrikov N. Zaitsev M.* Experimantal and numerical localization of noise sources for cylinder in round jet // Abstracts of Trilateral Russian-French-German Workshop “Computational experiment in aeroacoustics”. — Moscow: MAKS press, 2010. — September 22-25. — Pp. 75–78. — Svetlogorsk.

102. *Crighton D.G. Dowling A.P. Ffowcs Williams J.E. Heckl M.A. Leppington F.A.* Modern Methods in Analytical Acoustics: Lecture Notes. — Springer, 1992.

103. *P.J. Morris*. Scattering of sound by a sphere: category 1: problems 3 and 4 // Second Computational Aeroacoustics (CAA) Workshop on Benchmark Problems / Ed. by NASA Conference Publication 3352. — 2010. — Pp. 15–17.

104. *Colonius T. Lele S.K. Moin P.* The scattering of sound waves by a vortex: numerical simulations and analytical solutions // *J. Fluid Mech.* — 1994. — Vol. 260. — Pp. 271–298.

105. *Karabasov S.A. Kopiev V.F. Goloviznin V.M.* On a classical problem of acoustic wave scattering by a free vortex: Numerical modelling // 15th AIAA/CEAS Aeroacoustics Conference / Ed. by AIAA Paper 2009-3234. — 2009. — May 11-13. — Pp. 15–17. — Miami, FL.

106. *Жмакин А.И. Фурсенко А.А.* Об одной монотонной разностной схеме сквозного счета // *Ж. вычисл. матем и матем. физ.* — 1980. — Т. 20, № 4. — С. 1021–1031.

107. *Woodward P. Colella P.* The numerical simulation of two-dimensional fluid flow with strong shocks // *J. Comput. Phys.* — 1984. — Vol. 54, no. 1. — Pp. 115–173.

108. *Du X. Corre C. Lerat A.* A third-order finite-volume residual-based scheme for the 2D Euler equations on unstructured grids // *J. Comput. Phys.* — 2011. — Vol. 230, no. 11. — Pp. 4201–4215.

109. *Bunge U. Mockett C. Thiele F.* Guidelines for implementing detached-eddy simulation using different models // *Aerospace Science and Technology*. — 2007. — Vol. 11, no. 5. — Pp. 376—-385.

110. *Л.Г. Лойцянский*. Механика жидкости и газа. — М.: Дрофа, 2003.

111. *Shur M. Spalart P.R. Strelets M. Travin A.* A hybrid RANS-LES approach with delayed-DES and wall-modelled LES capabilities // *International Journal of Heat and Fluid Flow*. — 2008. — Vol. 29, no. 6. — Pp. 1638–1649.

112. *Schmitt V. Charpin F.* — Pressure Distributions on the ONERA-M6-Wing at Transonic Mach Numbers. Experimental Data Base for Computer Program Assessment. — Report of the Fluid Dynamics Panel Working Group 04, AGARD AR 138., 1979. — (URL: http://www.grc.nasa.gov/WWW/wind/valid/m6wing/m6wing.html).

113. *Vassberg J.C. DeHaan M.A. Rivers S.M. Wahls R.A.* Development of a Common Research Model for Applied CFD Validation Studies // *AIAA Paper 2008-6919 (URL: http://aaac.larc.nasa.gov/tsab/cfdlarc/aiaa-dpw/Workshop4/DPW4-geom.html)*. — 2008.

114. *Дубень А.П. Козубская Т.К. Миронов М.А.* Численное исследование резонаторов в волноводе // *Изв. РАН. МЖГ*. — 2012. — № 1. — С. 146–156.

115. *А.П. Дубень Т.К. Козубская С.И. Королев В.П. Маслов А.К. Миронов Д.А. Миронова В.М. Шахпаронов*. Исследования акустического течения в горле резонатора // *Акустический журнал*. — 2012. — Т. 58, № 1. — С. 80–92.

116. *О.А. Доронина Н.С. Жданова*. Численное моделирование рассеяния акустических волн изолированными вихревыми структурами // *Математическое моделирование*. — 2013. — Т. 25, № 9. — С. 85–94.

117. *Kozubskaya T. Duben A. Knacke T. Thiele F. Kopiev V. Zaitsev M.* Joint experimental and numerical study of gap turbulence interaction // *AIAA paper 2013-2214*. — 2013.

118. *Даньков Б.Н. Дубень А.П. Козубская Т.К.* Численное моделирование трансзвукового турбулентного обтекания клиновидного тела с обратным уступом // *Математическое моделирование*. — 2015.

119. *A. Gorobets F. X. Trias R. Borrell O. Lehmkuhl A. Oliva*. Hybrid MPI+OpenMP parallelization of an FFT-based 3D Poisson solver with one periodic direction // *Computers and Fluids*. — 2011. — Vol. 49. — Pp. 101–109.

120. *Kaneda Yukio, Yokokawa Mitsuo*. DNS of Canonical Turbulence with up to 4096^3 Grid Points // Parallel Computational Fluid Dynamics. — Elsevier, 2004. — May. — Pp. 23–32.

121. *T.Ishihara K.Morishita et al.* Direct numerical simulation of high reynolds number turbulence by the K computer // Japan-Russia workshop on supercomputer modeling, instability and turbulence in fluid dynamics (JR SMIT2015) / Keldysh Institute for Applied Mathematics RAS. — 2015. — March 4-6. — Moscow, Russia.

122. *Jimenez J., Moser. R. D.* What are we learning from simulating wall turbulence? // *Phil. Trans. Royal Soc. A*. — 2007. — Vol. 365. — Pp. 715–732.

123. *Myoungkyu Lee Nicholas Malaya, Moser Robert*. Petascale Direct Numerical Simulation of Turbulent Channel Flow on up to 786K Cores // SC '13 Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis. — 2013. — November 17-21. — Denver, Colorado.

124. *Краснопольский Б. Медведев А.* Алгоритмические особенности создания многосеточного решателя СЛАУ на вычислительных системах с графическими ускорителями // *Вестник Нижегородского университета им. Н.И. Лобачевского.* — 2014. — № 2. — С. 210–217.

125. *Жуков В.Т. Новикова Н.Д. Феодоритова О.Б.* Параллельный многосеточный метод для разностных эллиптических уравнений. Часть I. Основные элементы алгоритма // *Препринты ИПМ им. М. В. Келдыша*. — 2012. — № 030. — 32 с.

126. A scalable parallel Poisson solver for three-dimensional problems with one periodic direction / A. Gorobets, F. X. Trias, M. Soria, A. Oliva // *Computers & Fluids*. — 2010. — Vol. 39. — Pp. 525–538.

127. *M. Soria C. D. Pérez-Segarra A.Oliva*. A Direct Schur-Fourier Decomposition for the Solution of the Three-Dimensional Poisson Equation of Incompressible Flow Problems Using Loosely Parallel Computers // *Numerical Heat Transfer, Part B*. — 2003. — Vol. 43. — Pp. 467–488.

128. Direct numerical simulations of two- and three-dimensional turbulent natural convection flows in a differentially heated cavity of aspect ratio 4 / F. X. Trias, M. Soria, A. Oliva, C. D. Pérez-Segarra // *Journal of Fluid Mechanics*. — 2007. — Vol. 586. — Pp. 259–293.

129. *Chorin Alexandre Joel*. Numerical Solution of the Navier-Stokes Equations // *Journal of Computational Physics*. — 1968. — Vol. 22. — Pp. 745–762.

130. *Yanenko N. N.* The Method of Fractional Steps. — Springer-Verlag, 1971.

131. *Chorin A. J.* A Mathematical Introduction to Fluid Mechanics. — Springer-Verlag, 1993.

132. *Kim J., Moin P.* Application of a Fractional-Step Method to Incompressible Navier-Stokes Equations // *Journal of Computational Physics*. — 1985. — Vol. 123. — Pp. 308–323.

133. *R. W. C. P. Verstappen A. E. P. Veldman*. Symmetry-Preserving Discretization of Turbulent Flow // *Journal of Computational Physics*. — 2003. — Vol. 187. — Pp. 343–368.

134. *P.J.Davis*. Circulant Matrices. — Chelsea Publishing, New York, 1994.

135. A Direct Schur-Fourier Decomposition for the Efficient Solution of High-Order Poisson Equations on Loosely Coupled Parallel Computers / F. X. Trias, M. Soria, C. D. Pérez-Segarra, A. Oliva // *Numerical Linear Algebra with Applications*. — 2006. — Vol. 13, no. 4. — Pp. 303––326.

136. Numerical Recipies in C. The art of Scientific Computing / W.H.Press, S.A.Teukolsky, W.T.Vetterling, B.P.Flannery. — Cambridge University Press, Cambridge, 1992.

137. *M. Frigo S. G. Johnson*. The design and implementation of FFTW3 // Proceedings of the IEEE, special issue on “Program Generation, Optimization, and Platform Adaptation”. — 93 (2). — 2005. — Pp. 216–231.

138. *F.X. Trias A. Gorobets R.W.C.P. Verstappen M. Soria, Oliva A.* Turbulent flow around a wall-mounted cube: DNS and regularization modelling // Parallel Computational Fluid Dynamics: Recent Advances and Future Directions. Papers from the 21st International Conference on Parallel Computational Fluid Dynamics. — DEStech Publications, Inc, 2010. — Pp. 483–496. — Moffett Field, California, USA.

139. *A. Gorobets F.X. Trias R. Borrell G. Oyarzun, Oliva A.* Direct numerical simulation of turbulent flows with parallel algorithms for various computing architectures // In Proceedings of the Sixth European Computational Fluid Dynamics (ECCOMAS CFD 2014). — 2014. — Barcelona, Spain.

140. *A. Gorobets F. X. Trias M. Soria C. D. Perez-Segarra, Oliva A.* From extruded-2D to fully-3D geometries for DNS: a Multigrid-based extension of the Poisson solver // *Parallel Computational Fluid Dynamics 2008, Lecture Notes in Computational Science and Engineering*. — 2011. — Vol. 74, no. 6. — Pp. 219–226.

141. Direct numerical simulation of a differentially heated cavity of aspect ratio 4 with Ra-number up to 10^{11} - Part I: Numerical methods and time-averaged flow / F. X. Trias, A. Gorobets, M. Soria, A. Oliva // *International Journal of Heat and Mass Transfer*. — 2010. — Vol. 53. — Pp. 665–673.

142. *F.-X. Trias*. Direct numerical simulation and regularization modelling of turbulent flows on loosely coupled parallel computers using symmetry-preserving discretization: Ph.D. thesis: CTTC UPC, Terrassa, Spain. — 2006.

143. *F.X. Trias A. Gorobets C.D. Perez-Segarra, Oliva A.* Numerical simulation of turbulence at lower cost: Regularization modeling // *Computers & Fluids*. — 2013. — Vol. 80. — Pp. 251–259.

144. *Oerlemans S.* Acoustic wind tunnel tests on two-struts configuration // *VALIANT project report D20, NLR*. — 2011.

145. *Oerlemans S.* Coherence analysis for VALIANT 2-struts experimental benchmark data // *VALIANT memo, NLR*. — 2011.

146. *Kim W. W., Menon S.* An unsteady incompressible Navier-Stokes solver for large eddy simulation of turbulent flows // *International Journal for Numerical Methods in Fluids*. — 1999. — Vol. 31, no. 6. — Pp. 983–1017.

147. *L. Agrawal J. C. Mandal, Marathe A. G.* Computations of laminar and turbulent mixed convection in a driven cavity using pseudo-compressibility approach // *Computers & Fluids*. — 2001. — Vol. 30, no. 5. — Pp. 607–620.

148. *Ertuk E., Gökзöl C.* Fourth-order compact formulation of Navier-Stokes equations and driven cavity flow at high Reynolds numbers // *International Journal for Numerical Methods in Fluids*. — 2006. — Vol. 50, no. 4. — Pp. 421–436.

149. *E.D. dos Santos G. L. Piccoli F. H. R. Franca, Petry A. P.* Analysis of mixed convection in transient laminar and turbulent flows in driven cavities // *International Journal of Heat and Mass Transfer*. — 2011. — Vol. 54(21-22). — Pp. 4585–4595.

150. *F.X. Trias A. Gorobets C.D. Perez-Segarra, Oliva A.* DNS and regularization modeling of a turbulent differentially heated cavity of aspect ratio 5 // *International Journal of Heat and Mass Transfer*. — 2013. — Vol. 57, no. 1. — Pp. 171–182.

151. *A. Sergent P. Joubert, Le Quéré P.* Development of a local subgrid diffusivity model for large-eddy simulation of buoyancy-driven flows: application to a square differentially heated cavity // *Numerical Heat Transfer, part A*. — 2003. — Vol. 44, no. 8. — Pp. 789–810.

152. *Hsieh K. J., Lien F. S.* Numerical modeling of buoyancy-driven turbulent flows in enclosures // *International Journal of Heat and Fluid Flow*. — 2004. — Vol. 25. — Pp. 659–670.

153. *S. Kenjereš S. B. Gunarjo, Hanjalić K.* Contribution to elliptic relaxation modelling of turbulent natural and mixed ceonvection // *International Journal of Heat and Fluid Flow*. — 2005. — Vol. 26. — Pp. 569–586.

154. *Barhaghi D. G., Davidson L.* Natural convection boundary layer in a 5:1 cavity // *Physics of Fluids*. — 2007. — Vol. 19(12). — P. 125106.

155. *Albets-Chico X., A.Oliva, Pérez-Segarra C.D.* Numerical Experiments in Turbulent Natural Convection Using Two-Equation Eddy-Viscosity Models // *Journal of Heat Transfer*. — 2008. — Vol. 130, no. 7. — P. 072501.

156. *R. Cheesewright K.J. King, Ziai S.* Experimental data for the validation of computer codes for the prediction of two-dimensional buoyant cavity flows // *Heat Transfer Division, HTD*. — 1986. — Vol. 60. — Pp. 75–81.

157. *Ince N. Z., Launder B. E.* On the computation of buoyancy-driven turbulent flows in rectangular enclosures // *International Journal of Heat and Fluid Flow*. — 1989. — Vol. 10, no. 2. — Pp. 110–117.

158. *T. J. Heindel S. Ramadhyani, Incropera F. P.* Assessment of turbulence models for natural convection in an enclosure // *Numerical Heat Transfer, Part B: Fundamentals*. — 1994. — Vol. 26, no. 2. — Pp. 147–172.

159. *C. D. Pérez-Segarra A. Oliva M. Costa, Escanes F.* Numerical experiments in turbulent natural and mixed convection in internal flows // *International Journal of Numerical Methods for Heat and Fluid Flow*. — 1995. — Vol. 5, no. 1. — Pp. 13–32.

160. *S. Murakami S. Kato T. Chikamoto D. Laurence, Blay D.* New low-Reynolds-number $k-\epsilon$ model including damping effect due to buoyancy in a stratified flow field // *International Journal of Heat and Mass Transfer*. — 1996. — Vol. 39, no. 16. — Pp. 3483 – 3496.

161. *Liu F., Wen J. X.* Development and validation of an advanced turbulence model for buoyancy driven flows in enclosures // *International Journal of Heat and Mass Transfer*. — 1999. — Vol. 42, no. 21. — Pp. 3967 – 3981.

162. *Peng S. H., Davidson L.* Computation of turbulent buoyant flows in enclosures with low-Reynolds-number $k-\omega$ models // *International Journal of Heat and Fluid Flow*. — 1999. — Vol. 20, no. 2. — Pp. 172 – 184.

163. *Chenoweth D. R., Paolucci S.* Natural Convection in an Enclosed Vertical Air Layer with Large Horizontal Temperature Differences // *Journal of Fluid Mechanics*. — 1986. — Vol. 169. — Pp. 173 – 210.

164. Parameter-free symmetry-preserving regularization modeling of a turbulent differentially heated cavity / F. X. Trias, R. W. C. P. Verstappen, A. Gorobets et al. // *Computers & Fluids*. — 2010. — Vol. 39. — Pp. 1815–1831.

165. *D. Saury N. Rouger F. Djanna, Penot F.* Natural convection in an air-filled cavity: Experimental results at large Rayleigh numbers // *International Communications in Heat and Mass Transfer*. — 2011. — Vol. 38. — Pp. 679–687.

166. Direct numerical simulation of a differentially heated cavity of aspect ratio 4 with Ra-number up to 10^{11} - Part II: Heat transfer and flow dynamics / F. X. Trias, A. Gorobets, M. Soria, A. Oliva // *International Journal of Heat and Mass Transfer*. — 2010. — Vol. 53. — Pp. 674–683.

167. *Gray D. D., Giorgini A.* The validity of the boussinesq approximation for liquids and gases // *International Journal of Heat and Mass Transfer*. — 1976. — Vol. 19, no. 5. — Pp. 545–551.

168. Experimental and numerical investigation of turbulent natural convection in a large air-filled cavity / J. Salat, S. Xin, P. Joubert et al. // *International Journal of Heat and Fluid Flow*. — 2004. — Vol. 25. — Pp. 824–832.

169. Resolving the stratification discrepancy of turbulent natural convection in differentially heated air-filled cavities. Part III: a full convection-conduction-surface radiation coupling / S. Xin, J. Salat, P. Joubert et al. // *International Journal of Heat and Fluid Flow*. — 2013. — Vol. 42. — Pp. 33–48.

170. *Ashforth-Frost S., Jambunathan K., Whitney C. F.* Velocity and Turbulence Characteristics of a Semiconfined Orthogonally Impinging Slot Jet // *Exp. Thermal and Fluid Science*. — 1997. — Vol. 14, no. 1. — Pp. 60–67.

171. *Zhe J., Modi V.* Near Wall Measurements for a Turbulent Impinging Slot Jet // *Journal of Fluids Engineering*. — 2001. — Vol. 123, no. 1. — Pp. 112–120.

172. *Roache Patrick J.* Code Verification by the Method of Manufactured Solutions // *Journal of Fluids Engineering, Transactions of ASME*. — 2002. — Vol. 124. — Pp. 4–10.

173. *Beaubert F., Viazzo S.* Large eddy simulations of plane turbulent impinging jets at moderate Reynolds numbers // *International Journal of Heat and Fluid Flow*. — 2003. — Vol. 24, no. 4. — Pp. 512–519.

174. *Hattori H., Nagano Y.* Direct numerical simulation of turbulent heat transfer in plane impinging jet // *International Journal of Heat and Fluid Flow*. — 2004. — Vol. 25, no. 5. — Pp. 749–758.

175. Direct numerical simulation of a three-dimensional natural-convection flow in a differentially heated cavity of aspect ratio 4 / M. Soria, F. X. Trias, C. D. Pérez-Segarra, A. Oliva // *Numerical Heat Transfer, part A*. — 2004. — April. — Vol. 45. — Pp. 649–673.

176. *Hofmann H., Kind M., Martin H.* Measurements on Steady State Heat Transfer and Flow Structure and New Correlations for Heat and Mass Transfer in Submerged Impinging Jets. // *International Journal of Heat and Mass Transfer*. — 2007. — Vol. 50, no. 19. — Pp. 3957–3965.

177. *Narayanan V., Seyed-Yagoobi J., Page R. H.* An experimental study of fluid mechanics and heat transfer in an impinging slot jet flow // *International Journal of Heat and Mass Transfer*. — 2004. — Vol. 47, no. 8-9. — Pp. 1827–1845.

178. *F.X.Trias A. Gorobets, Oliva A.* Turbulent flow around a square cylinder at Reynolds number 22000: a DNS study // *Computers & Fluids, (submitted)*.

179. Status of large eddy simulation: Results of a workshop / W. Rodi, J. H. Ferziger, M. Breuer, M. Pourquié // *Journal of Fluids Engineering, Transactions of the ASME*. — 1997. — Vol. 119, no. 2. — Pp. 248–262.

180. *Voke P. R.* Flow past a square cylinder: test case LES2 // Direct and Large Eddy Simulation II / Ed. by J.P. Chollet et al. — 1997. — Pp. 355–373.

181. *Verstappen R. W. C. P., Veldman A. E. P.* Spectro-consistent discretization of Navier-Stokes: a challenge to RANS and LES // *Journal of Engineering Mathematics*. — 1998. — Vol. 34. — Pp. 163–179.

182. *Sohankar A., Norberg C., Davidson L.* Low-Reynolds-number flow around a square cylinder at incidence: study of blockage, onset of vortex shedding and outlet boundary condition // *International Journal for Numerical Methods in Fluids*. — 1999. — Vol. 26. — Pp. 39–56.

183. *Sohankar A., Norberg C., Davidson L.* Simulation of three-dimensional flow around a square cylinder at moderate Reynolds numbers // *Physics of Fluids*. — 1999. — Vol. 11. — Pp. 288–306.

184. Accurate computations of the laminar flow past a square cylinder based on two different methods: lattice-Boltzmann and finite-volume / M. Breuer, J. Bernsdorf, T. Zeiser, F. Durst // *International Journal of Heat and Fluid Flow*. — 2000. — Vol. 21. — Pp. 186–196.

185. Experimental and high-order LES analysis of the flow in near-wall region of a square cylinder / M. Minguez, C. Brun, R. Pasquetti, E. Serre // *International Journal of Heat and Fluid Flow*. — 2011. — Vol. 32, no. 3. — Pp. 558–566.

186. *Lyn D., Rodi W.* The flaping shear layer formed by flow separation from the forward corner of a square cylinder // *Journal of Fluid Mechanics*. — 1994. — Vol. 267. — Pp. 353–376.

187. A laser doppler velocimetry study of ensemble-averaged characteristics of the turbulent near wake of a square cylinder / D. Lyn, S. Einav, W. Rodi, J. Park // *Journal of Fluid Mechanics*. — 1995. — Vol. 304. — Pp. 285–319.

188. *Lee B E.* The effect of turbulence on the surface pressure field of square prisms // *Journal of Fluid Mechanics*. — 1975. — Vol. 69. — Pp. 263–282.

189. *Vickery B. J.* Fluctuating lift and drag on a long square cylinder of square cross-section in a smooth and in a turbulent stream // *Journal of Fluid Mechanics*. — 1966. — Vol. 25. — P. 481.

190. *Ochoa J. S., Fueyo N.* Large eddy simulation of the flow past a square cylinder // PHOENICS User Conference 2004. — Melbourne, Australia: 2004.

191. *A Sohankar L Davidson C Norberg.* Large eddy simulation of flow past a square cylinder: comparison of different subgrid scale models // *Journal of Fluids Engineering*. — 2000. — Vol. 122. — Pp. 39–47.

192. *Rodi W.* Comparison of LES and RANS calculations of the flow around bluff bodies // *Journal of Wind Engineering and Industrial Aerodynamics*. — 1997. — Vol. 69-71. — Pp. 55 – 75.

193. *Hao Zhang F. Xavier Trias Andrey Gorobets Yuanqiang Tan Assensi Oliva*. Direct numerical simulation of a fully developed turbulent duct flow up to $Re_\tau = 1200$ // *International journal of heat and fluid flow*. — 2015. — Vol. 54. — P. 258–267.

194. *Gavrilakis S.* Numerical simulation of low-Reynolds-number turbulent flow through a straight square duct // *Journal of Fluid Mechanics*. — 1992. — Vol. 244. — Pp. 101–129.

195. *A. Huser S. Biringen*. Direct numerical simulation of turbulent flow in a square duct // *Journal of Fluid Mechanics*. — 1993. — Vol. 257. — Pp. 65–95.

196. *G. Sharma D. J. Phares*. Turbulent transport of particles in a straight square duct // *International Journal of Multiphase Flow*. — 2006. — Vol. 32, no. 7. — P. 823–837.

197. *Hartnett JP, Koh JCY, McComas ST*. A comparison of predicted and measured friction factors for turbulent flow through rectangular ducts // *Journal of Heat Transfer*. — 1962. — Vol. 84, no. 1. — Pp. 82–88.

198. *A. Yakhot T. Anor H. Liu, Nikitin. N.* Direct numerical simulation of turbulent flow around a wall-mounted cube: spatio-temporal evolution of large-scale vortices // *Journal of Fluid Mechanics*. — 2006. — Vol. 566. — P. 1–9.

199. *D. Saury N. Rouger F. Djanna, Penot F.* Natural convection in an air-filled cavity: Experimental results at large Rayleigh numbers // *International Communications in Heat and Mass Transfer*. — 2011. — Vol. 38. — Pp. 679–687.

200. *A. Sergent P. Joubert S. Xin, Quere P. Le*. Resolving the stratification discrepancy of turbulent natural convection in differentially heated air-filled cavities. Part II: End walls effects // *International Journal of Heat and Fluid Flow*. — 2013. — Vol. 39. — P. 15–27.

Список рисунков

Список таблиц