2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

doi 10.26089/NumMet.v26r103 УДК 519.67

Сравнение подходов к динамической адаптации расчетной сетки на распределенной вычислительной системе

С. К. Григорьев
Институт прикладной математики имени М. В. Келдыша РАН (ИПМ РАН),
Москва, Российская Федерация
ORCID: 0009-0005-2012-6609, e-mail: sergejgri@gmail.com

А. А. Бай
Институт прикладной математики имени М. В. Келдыша РАН (ИПМ РАН),
Москва, Российская Федерация
ORCID: 0000-0002-0007-6246, e-mail: bay.aa@phystech.edu

Аннотация: В статье обсуждаются блочный и листовой подходы к динамической адаптации регулярных сеток. Целью работы является сравнение эффективности подходов при решении задач на распределенной вычислительной системе. Приведены описания подходов и особенностей реализации разностных схем с их помощью. Произведено сравнение эффективности подходов по быстродействию и количеству сеточных элементов при выполнении трехмерных расчетов развития неустойчивости типа Рэлея–Тейлора. При моделировании применена схема предиктор-корректор с использованием линейной интерполяции и метода HLL (схема годуновского типа). Адаптация сетки выполнялась в области перемешивания веществ. Проведены расчеты, результаты которых соответствуют априорным оценкам эффективности подходов.

Ключевые слова: динамическая адаптация, octree-сетки, параллельные вычисления.

Благодарности: Мы хотели бы поблагодарить М. В. Якобовского и В. А. Гасилова за полезные и плодотворные обсуждения. Вычисления проводились с помощью гибридного суперкомпьютера K60, установленного в Суперкомпьютерном Центре коллективного пользования ИПМ имени М. В. Келдыша РАН.

Для цитирования: Григорьев С.К., Бай А.А. Сравнение подходов к динамической адаптации расчетной сетки на распределенной вычислительной системе // Вычислительные методы и программирование. 2025. **26**, № 1. 33–49. doi 10.26089/NumMet.v26r103.

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

Comparison of adaptive mesh refinement approaches on distributed computational system

Sergej K. Grigorjev
Keldysh Institute of Applied Mathematics of RAS, Moscow, Russia
ORCID: 0009-0005-2012-6609, e-mail: sergejgri@gmail.com
Anton A. Bay
Keldysh Institute of Applied Mathematics of RAS, Moscow, Russia
ORCID: 0000-0002-0007-6246, e-mail: bay.aa@phystech.edu

Abstract: The article discusses block-based and tree-based approaches to adaptive mesh refinement on regular meshes. Our goal is to compare the efficiency of approaches in solving problems on a distributed computational system. Descriptions of the approaches and features of the implementation of difference schemes using them are given. A comparison of the efficiency of the approaches in terms of speed and number of mesh elements when performing three-dimensional calculations of the development of Rayleigh–Taylor instability is made. The modeling used a predictor-corrector scheme, a Godunov-type scheme using linear interpolation, and the HLL method. Grid adaptation is applied in the area of mixing of substances. Experiments compliance of a priori estimates of the effectiveness of the approaches obtained as a result of the experiments.

Keywords: adaptive mesh refinement, octree-mesh, parallel computations.

Acknowledgements: We would like to thank M. V. Yakobovskiy and V. A. Gasilov for useful and fruitful discussions. The research is carried out using the K60 hybrid supercomputer installed at the Supercomputer Center for Collective Use of the Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences.

For citation: S. K. Grigorjev, A. A. Bay, "Comparison of adaptive mesh refinement approaches on distributed computational system," Numerical Methods and Programming. **26** (1), 33–49 (2025). doi 10.26089/NumMet.v26r103.

1. Введение. Динамическая адаптация сетки (Adaptive Mesh Refinement, AMR) — это технология численного решения задач математической физики, разработанная с целью перестроения расчетных сеток в зависимости от свойств или особенностей структуры решения. Как известно, для многих задач, решаемых численно каким-либо сеточным методом, нет необходимости использования сеток с единым уровнем дискретизации для всей области расчета. В тех случаях, когда моделируются взаимовлияющие нелинейные процессы, протекающие на существенно различающихся масштабах по времени и пространству, заранее бывает трудно или невозможно получить априорную информацию о расположении и/или эволюции областей локализации особенностей решения — больших градиентов, скачкообразных изменений свойств среды и т.п. Динамическая локальная адаптация открывает возможность сохранения уровня точности во всей расчетной области при значительно меньшей вычислительной нагрузке, чем в случае использования статической сетки. Данная сеточная технология была предложена в 80-х гг. прошлого века [1] для моделирования задач гидродинамики и в настоящее время используется для широкого спектра задач математической физики, решаемых методом сеток.

Существующие методы динамической локальной адаптации условно можно разделить на адаптацию структурно-регулярных [1] и нерегулярных (неструктурированных) сеток [2, 3]. В этой статье рассматриваются подходы к динамической адаптации регулярной сетки.

Существует несколько разных подходов к реализации динамической адаптации регулярной сетки: cell-based (он же tree-based, или листовой), block-based (блочный). В cell-based сетках адаптируются только те ячейки, которые помечены критерием адаптации, этот подход дает лучший контроль структуры сетки и наименьшее количество ячеек при одинаковых критериях. Примерами реализации сеток листового типа являются p4est [4], t8code [5]. Подобный подход наиболее применим к задачам, в которых адаптированная область существенно меньше всей расчетной области, к задачам со сложной геометрией

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

особенностей потока. Подход широко применяется для решения задач астрофизики [6, 7]. В block-based сетках адаптация происходит не для каждой ячейки по отдельности, а сразу для всех ячеек некоторого прямоугольного блока. Такой подход позволяет использовать более простые и быстрые структуры данных, но также неизбежно влечет бóльшие размеры сетки. Блочные сетки тоже, вообще говоря, бывают разных типов. Блочные сетки (заранее размеченные на блоки, которые впоследствии могут быть адаптированы) обычно применяются в задачах с заранее известными областями адаптации. Примером сеток такого типа является PARAMESH [8]. Сетки, аналогичные изначально предложенным в [1], сейчас часто называют patch-based, они предполагают построение и перестроение блоков во время расчетов на основании маркированных критериями адаптации ячеек. Подход является более универсальным и в среднем приводит к сеткам меньшего размера, чем в случае закрепленных блоков, но при этом влечет за собой издержки, связанные с более сложными операциями адаптации, балансировки и обработки границ блоков. Подход имеет большое число реализаций, например AMReX [9], Chombo [10], Enzo [11], SAMRAI [12, 13]. Применения этого подхода можно найти для задач астрофизики, динамики плазмы, геофизики и др.

Цель этой статьи — сравнение блочного и листового методов динамической адаптации. Отметим, что на эффективность программных реализаций методов влияют как операции перестройки сетки, так и сопутствующие операции по изменению структуры сеточных данных, балансировки нагрузки вычислительной системы и обновления полей обмена (в случае параллельных реализаций методик адаптации), что приводит к дополнительным вычислительным затратам. Эти факторы изучались путем вычислительных экспериментов, для выполнения которых были взяты библиотеки AMReX[1] (комплекс утилит для работы с сетками типа patch-based) и octreemesh[2] (комплекс утилит для листовой адаптации). Сравнение блочного и листового подходов выполнено на примере решения задачи о развитии гидродинамической неустойчивости типа Рэлея–Тейлора. Похожие исследования для сравнения листовых сеток с блочными в задачах прогнозирования погоды описаны в [14]. Общие результаты этих исследований, связанные с эффективностью, согласуются с приведенными в настоящей статье.

Содержание и структура статьи следующие: во втором разделе приведены основные определения, в следующих двух разделах кратко описаны используемые библиотеки, в пятом разделе рассмотрены реализации численной схемы и их различия, в шестом разделе делаются предположения о производительности методов, в седьмом разделе анализируются результаты расчетов и сравниваются эффективности блочного и листового подходов к адаптации сетки на распределенной вычислительной системе.

2. Определения. Основные операции, выполняемые при адаптации, — это дробление и огрубление. Дробление — это разделение сеточного элемента на несколько элементов меньшего размера, а укрупнение — обратная операция. Будем называть нераздробленные ячейки листовыми, а раздробленные — нелистовыми.

Уровень дробления ячейки — характеристика размера ячейки по отношению к исходной регулярной сетке. Чем больше уровень дробления, тем меньше ячейка и сильнее измельчена сетка.

Критерий дробления сетки — это совокупность параметров, которые отвечают за принятие решения о дроблении и укрупнении ячеек сетки. Критерии делятся на структурные, функциональные и геометрические.

Геометрический критерий дробления — дробление ячеек сетки, обусловленное особенностями заполняемой области, например на границе расчетной области.

Функциональный критерий дробления — адаптация сетки к особенностям решения.

Структурный критерий дробления подразумевает, что две смежные ячейки могут отличаться по уровню дробления не более чем на 1, т.е. принимается необходимость соблюдения соотношения линейных размеров смежных элементов не более чем $1:2$, а также соблюдение заданного количества сеточных элементов одного уровня измельчения между элементами, линейные размеры которых отличаются в четыре раза.

Сеточным доменом будем называть набор ячеек сетки, выделенный из сетки в соответствии с некоторым критерием. На один MPI-процесс приходится один домен. Нелистовые ячейки при определении размера домена не учитываются.

Виртуальные ячейки — множество ячеек на границе домена, принадлежащих другому домену.

Индексное пространство — пространство трехмерных целочисленных векторов, определяемое начальной точкой расчетной области (x_0, y_0, z_0) и линейным размером ячейки h. Будем считать, что в этом

[1] См. https://amrex-codes.github.io/

[2] Разработка института прикладной математики имени М. В. Келдыша РАН (ИПМ РАН)

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

пространстве индекс (i, j, k) относится к ячейке с центром в $\left(x_0 + \left(i + \frac{1}{2}\right) h, y_0 + \left(j + \frac{1}{2}\right) h, z_0 + \left(k + \frac{1}{2}\right) h\right)$. Индексация граней требует также задать направление, так, например, по направлению x индекс (i, j, k) соответствует грани с центром в $\left(x_0 + ih, y_0 + \left(j + \frac{1}{2}\right) h, z_0 + \left(k + \frac{1}{2}\right) h\right)$.

Блок — прямоугольный параллелепипед в индексном пространстве. Определяется двумя векторами индексов (i_1, j_1, k_1), (i_2, j_2, k_2) и содержит в себе все ячейки с индексами (i, j, k) такими, что $i_1 \leqslant i \leqslant i_2$, $j_1 \leqslant j \leqslant j_2$, $k_1 \leqslant k \leqslant k_2$.

MPI-процесс — один экземпляр программы, выполняющейся в параллельном режиме с использованием библиотеки, реализующей стандарт MPI. Каждый MPI-процесс работает независимо, но они могут взаимодействовать друг с другом.

3. Библиотека Octreemesh. Листовой подход к динамической адаптации расчетной сетки, реализованный в библиотеке octreemesh, предполагает независимое измельчение или огрубление любой ячейки сетки. Преимуществом такого подхода является возможность минимизировать количество сеточных элементов, участвующих в расчете.

Значения сеточных функций, соответствующих тем или иным физическим параметрам, могут относиться к ячейкам и вершинам сетки. В силу того, что операция огрубления выполняется только при соблюдении и функционального, и структурного критериев, огрубление ячеек сетки производится в порядке убывания уровня измельчения. Вызов процедур интерполяции значений сеточных функций в ячейках происходит непосредственно во время изменения ячейки, во время вызова соответствующей подпрограммы.

В процессе адаптации в памяти хранятся все сеточные элементы: как листовые, так и нелистовые, а также связывающие их отношения смежности и наследственности (родитель-потомок).

Распределение данных между MPI-процессами выполняется на основе листовых ячеек. В области виртуальных ячеек формируются и листовые, и не листовые элементы. Обмены данными между MPI-процессами синхронные. Формирование очереди обменов выполняется на основе разделения графа обменов.

Перестроение каждого домена выполняется MPI-процессами независимо друг от друга. Для синхронизации топологии виртуальных ячеек после перестроения сетки выполняется операция синхронизации топологии. Особенности реализации этого алгоритма обуславливают ограничение на изменение ячейки сетки не более чем на один уровень измельчения за один вызов процедуры перестроения сетки.

После перестроения сетки выполняется балансировка нагрузки. Для декомпозиции сетки используется PARMETIS [15]. Вызов процедуры формирования глобальной нумерации перед балансировкой обусловлен необходимостью восстановления виртуальных ячеек после перераспределения доменов во время балансировки. По завершении балансировки выполняется повторный вызов алгоритмов сборки мусора и формирования глобальной нумерации сеточных элементов.

Глобальная нумерация вводится отдельно для листовых и отдельно для всех (листовых и нелистовых) элементов. Выполнение перестроения сетки содержит следующие этапы:

1) применение функционального критерия;
2) применение структурного критерия;
3) огрубление ячеек;
4) синхронизация топологии виртуальных ячеек;
5) измельчение ячеек;
6) синхронизация топологии виртуальных ячеек;
7) формирование глобальной нумерации и сборка мусора;
8) балансировка нагрузки и перераспределение ячеек между MPI-процессами;
9) формирование глобальной нумерации и сборка мусора.

Все этапы перестроения сетки скрыты от пользователя API библиотеки, для выполнения адаптации пользователю достаточно задать функции, определяющие процедуру переинтерполяции значений сеточных функций при адаптации.

4. Фреймворк AMReX. Сетки AMReX являются реализацией блочного подхода к динамической адаптации. Аналогично операциям огрубления и дробления для ячеек будем говорить, что огрублением блока (i_1, j_1, k_1), (i_2, j_2, k_2) в индексном пространстве с линейным размером h является блок с параметрами $(i_1/2, j_1/2, k_1/2)$, $(i_2/2, j_2/2, k_2/2)$, принадлежащий индексному пространству с линейным размером $2h$. Дробление определяется таким же образом.

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

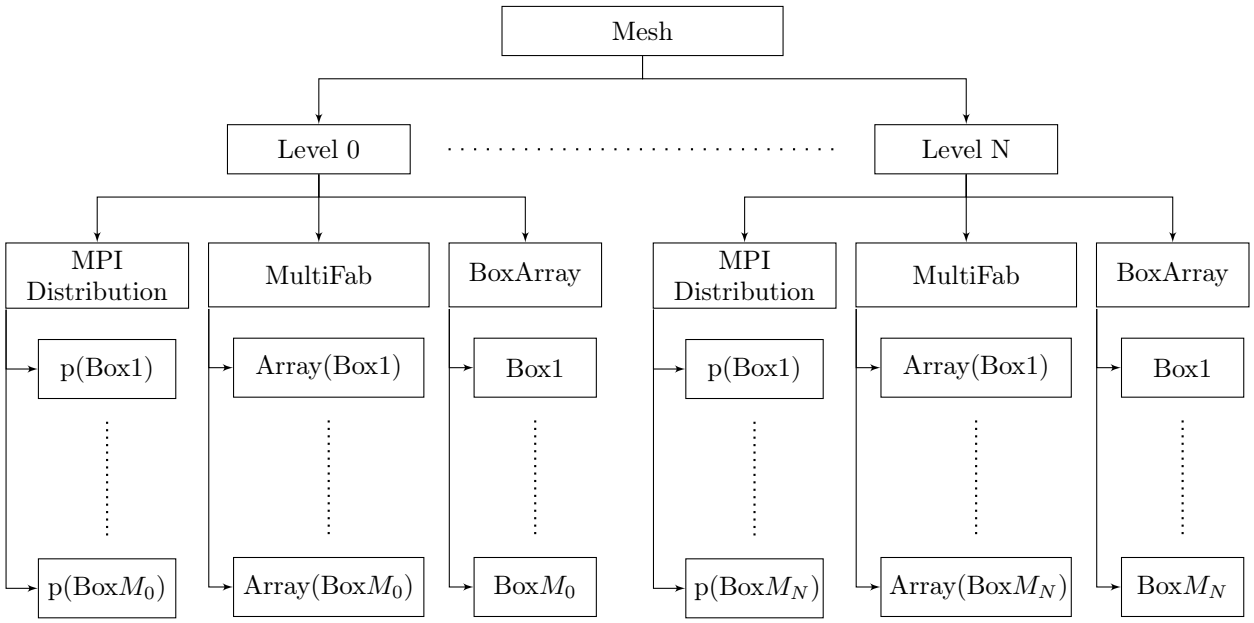

Рис. 1. Структура сетки AMReX

Fig. 1. Structure of the AMReX mesh

Вся сетка представляется в виде сеток для конкретных уровней дробления от нулевого (начальной сетки) до некоторого заданного максимального уровня дробления N (рис. 1). Каждый уровень состоит из своего индексного пространства (для разных уровней отличается линейный размер ячейки), структуры сетки, структур, содержащих данные на этом уровне сетки, таблицы распределения по MPI-процессам.

Структура сетки любого уровня — массив блоков, из которых этот уровень сетки состоит. Блоки внутри одного уровня не могут накладываться друг на друга. Сетка нулевого уровня строится таким образом, чтобы объединение блоков соответствовало всей расчетной области. Сетки первого уровня и выше строятся так, чтобы удовлетворять структурному и функциональному критериям.

Структуры данных могут отличаться в зависимости от того, что необходимо для реализации той или иной схемы расчета. Отличия могут быть как в количестве физических параметров, так и в том, к каким элементам сетки эти параметры относятся — ячейкам, граням или вершинам. Так или иначе, структуры данных для одного уровня подобной сетки будут аналогичны или построены из структур типа `amrex::MultiFab`.

MultiFab определяется структурой сетки уровня, таблицей распределения по MPI-процессам, количеством физических компонент n_{comp}, шириной слоя виртуальных ячеек n_{ghost} и типом элементов сетки, к которым относятся данные (ячейки, грани, вершины). MultiFab состоит из массивов с физическими параметрами. Количество массивов равно количеству блоков в сетке уровня. Каждый массив содержит параметры, относящиеся ко всем ячейкам/граням/вершинам соответствующего ему блока, включая слой виртуальных.

Стратегия распараллеливания с помощью MPI заключается в следующем: поскольку объем памяти, необходимый для описания структуры сетки, пренебрежимо мал по сравнению с объемом памяти, необходимым для хранения данных о физических параметрах в ячейках (блоков значительно меньше, чем ячеек), структура сетки содержится на каждом MPI-процессе. Массивы с физическими параметрами внутри MultiFab'ов хранятся и обрабатываются только на одном MPI-процессе. Распределение блоков записано в таблице распределения по MPI-процессам. Распределение происходит путем сортировки блоков по их размеру и раздаче MPI-процессам по очереди.

Реализация пользователем какой-либо разностной схемы на подобных сетках, как правило, предполагает сначала заполнение виртуальных ячеек блоков и потом обход всех ячеек и применение в них соответствующей схемы. Обход ячеек происходит по алгоритму, подобному алгоритму 1.

Заполнение слоя виртуальных ячеек зависит от типа границы (рис. 2). Если это граница расчетной области, значения в виртуальных ячейках заполняются на основании граничных условий задачи. Если это

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

Алгоритм 1. Пример применения разностной схемы на сетках AMReX

Algorithm 1. Pseudocode example of using difference scheme with AMReX mesh

```
for Level = 0, 1, ..., N do
    for Box = BoxArray(Level).first(), ..., BoxArray(Level).last()
        In = Array from MultiFab with input data
        Out = Array from MultiFab with output data
        for (i, j, k) ∈ Box do
            Using difference scheme, for example:
            Out(i, j, k, 0) = In(i, j, k, 0) + dt/h · (In(i − 1, j, k, 0) − In(i, j, k, 0)) + ...
        end for
    end for
end for
```

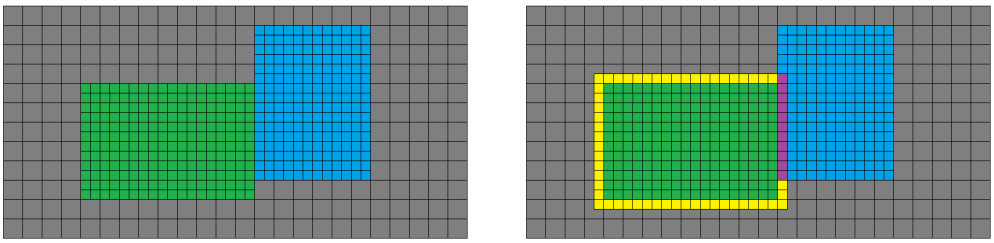

Рис. 2. Иллюстрация слоя виртуальных ячеек с шириной $n_{\text{ghost}} = 1$

Fig. 2. Illustration of a layer of virtual cells with width $n_{\text{ghost}} = 1$

граница между двумя блоками одного уровня, то в виртуальные ячейки записываются значения из ячеек соседнего блока. Если прообраз виртуальной ячейки не принадлежит ни одному блоку того же уровня, то она заполняется интерполяцией из ячеек сетки уровнем ниже.

Отдельное внимание стоит уделить алгоритму адаптации сетки. Процесс адаптации уровней сетки происходит по убыванию, из-за необходимости учета структурного критерия. Пусть задан функциональный критерий адаптации, максимальный линейный размер блока $N_{\max}$ и минимальная допустимая эффективность адаптации $\text{eff}_{\min}$. Последняя определяется как минимальное допустимое значение отношения числа ячеек блока, которые были маркированы критерием адаптации, к общему числу ячеек блока. Таким образом, при увеличении $\text{eff}_{\min}$ в сетке уменьшается общее количество ячеек и увеличивается общее число блоков. Рассмотрим построение (или перестроение) уровня $n + 1$ при существующем уровне n.

1. Обходятся все ячейки уровня n, для каждой проверятся удовлетворение критериям адаптации.
2. На каждом MPI-процессе создается кластер — структура, содержащая массив с индексами маркированных ячеек и параметры минимального блока, который содержит все эти ячейки.
3. Кластеры всех MPI-процессов объединяются в один на нулевом MPI-процессе.

Цель следующих операций — разбить кластер на несколько других кластеров, которые бы содержали все маркированные ячейки, имея при этом сеточную эффективность $\text{eff}_{\text{cl}} \geqslant \text{eff}_{\min}$ и линейные размеры минимального блока меньше $N_{\max}$. Сеточная эффективность кластера определяется отношением количества маркированных ячеек к общему количеству ячеек, содержащемуся в минимальном блоке кластера.

4. Если условия на сеточную эффективность или линейные размеры кластера не выполнены, то запускается процесс разбиения кластера.
   - 4.1. Для каждого направления строится массив весов слоев (слоем блока по некоторому направлению будем называть все ячейки блока с фиксированным индексом по этому направлению). Заполняются эти массивы количеством маркированных ячеек в соответствующем слое.
   - 4.2. Далее ищется лучший разрез кластера. Лучшим разрезом считается разрез по слою, не содержащему маркированных ячеек; следующим по списку идет разрез, на котором изменение второй разности весов максимально; если максимальное изменение оказывается меньше некоторой заданной границы, то предлагается разрез по центру блока. Если разрезов одного типа несколько, выбирается тот, что ближе к середине кластера.

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

4.3. В соответствии с выбранным разрезом из одного кластера формируются два меньших. Для каждого из них происходит пересчет параметров минимального блока. Каждый из них проверяется на удовлетворение условиям минимальной сеточной эффективности и максимального линейного размера блока. В случае неудовлетворения условию процесс разбиения запускается уже для новых кластеров.

5. Минимальные блоки получившихся кластеров дробятся до $n+1$ уровня. Таким образом получается новая структура сетки $n+1$ уровня. Блоки раздаются по MPI-процессам. Новые MultiFab'ы создаются и заполняются либо значениями из старых MultiFab'ов $n+1$ уровня, либо интерполяцией значений из MultiFab'ов уровня n.

5. Методика расчета. Сравнение эффективности работы библиотек выполнено на задаче о моделировании развития неустойчивости типа Рэлея–Тейлора на границе раздела тяжелой и легкой жидкостей в поле силы тяжести. Движение вещества рассчитывается в трехмерной модели на основе системы уравнений Эйлера:

$$\frac{\partial U}{\partial t}+\frac{\partial F_x}{\partial x}+\frac{\partial F_y}{\partial y}+\frac{\partial F_z}{\partial z}=H,$$

$$U=\begin{pmatrix}\rho_1\\ \rho_2\\ \rho u\\ \rho v\\ \rho w\\ E\end{pmatrix},\quad H=\begin{pmatrix}0\\ 0\\ \rho g\\ 0\\ 0\\ \rho u g\end{pmatrix},\quad F_x=\begin{pmatrix}\rho_1 u\\ \rho_2 u\\ \rho u^2+p\\ \rho v u\\ \rho w u\\ (E+p)u\end{pmatrix},\quad F_y=\begin{pmatrix}\rho_1 v\\ \rho_2 v\\ \rho u v\\ \rho v^2+p\\ \rho w v\\ (E+p)v\end{pmatrix},\quad F_z=\begin{pmatrix}\rho_1 w\\ \rho_2 w\\ \rho u w\\ \rho v w\\ \rho w^2+p\\ (E+p)w\end{pmatrix}.$$

Система замыкается уравнениями состояния:

$$p_i=(\gamma-1)\rho_i\varepsilon_i,\quad E_i=\rho_i\left(\varepsilon_i+\frac{u^2+v^2+w^2}{2}\right),$$

$$p=p_1+p_2,\quad \rho=\rho_1+\rho_2,\quad E=E_1+E_2.$$

Перед построением схемы стоит отметить один важный факт о реализации схем на сетках AMReX: соседние ячейки двух разных уровней дробления не могут взаимодействовать друг с другом напрямую, поскольку они будут находиться в разных блоках. Их взаимодействие может быть реализовано только через слой виртуальных ячеек блока верхнего уровня, т.е. ячейки нижнего уровня могут взаимодействовать с ячейками верхнего только через функцию интерполяции, причем шаблон этой интерполяции может включать только ячейки нижнего уровня. Это приводит к тому, что на сетках AMReX плохо реализуется, например, схема интерполяции в ячейке на границе уровней дробления с шаблоном, который по одному направлению должен содержать четыре соседних ячейки уровнем выше. В связи с этим подбиралась такая схема, которая может быть просто реализована и на листовых, и на блочных сетках.

Итак, для дискретизации уравнений газовой динамики используется схема предиктор-корректор, схема годуновского типа с использованием линейной интерполяции и метода HLL [16] для нахождения потоков через грани. Схема может быть записана следующим образом:

$$U_i^{n+\frac{1}{2}}=U_i^n-\frac{\tau}{2V_i}\sum_{j\in\Gamma_i}F_{ij}^n S_{ij},$$

$$U_i^{n+1}=U_i^n-\frac{\tau}{V_i}\sum_{j\in\Gamma_i}F_{ij}^{n+\frac{1}{2}} S_{ij},$$

где Γ_i — множество индексов ячеек, имеющих общую грань с i-й ячейкой. Для вычисления потоков сначала производится линейная интерполяция значений $\boldsymbol{\phi}=(\rho,u,v,w,p)^T$ из ячеек на грань γ_{ij}

$$\boldsymbol{\phi}_{ij}^L=\boldsymbol{\phi}_i+\frac{G_i}{2},\quad \boldsymbol{\phi}_{ij}^R=\boldsymbol{\phi}_j-\frac{G_j}{2},\tag{1}$$

значения G_i, G_j находятся по значениям $\boldsymbol{\phi}$ в четырех точках рядом с гранью (рис. 3):

$$\begin{aligned}G_i&=\operatorname{sign}\left(\boldsymbol{\phi}(p_1)-\boldsymbol{\phi}(p_{-1})\right)\operatorname{absmin}\left(\boldsymbol{\phi}(p_{-1})-\boldsymbol{\phi}(p_{-2}),\boldsymbol{\phi}(p_1)-\boldsymbol{\phi}(p_{-1})\right),\\ G_j&=\operatorname{sign}\left(\boldsymbol{\phi}(p_1)-\boldsymbol{\phi}(p_{-1})\right)\operatorname{absmin}\left(\boldsymbol{\phi}(p_2)-\boldsymbol{\phi}(p_1),\boldsymbol{\phi}(p_1)-\boldsymbol{\phi}(p_{-1})\right).\end{aligned}\tag{2}$$

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

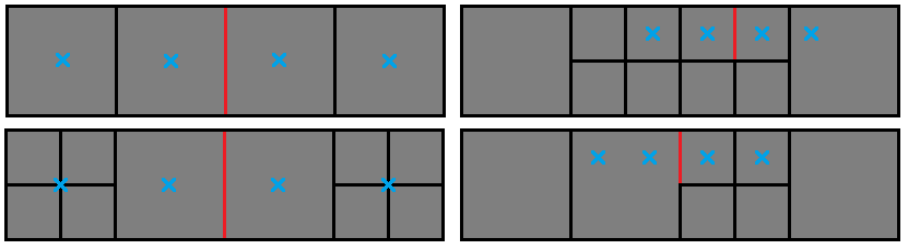

Рис. 3. Шаблон интерполяции в разных случаях. Красным обозначена грань, в середине которой ищется интерполяция $\boldsymbol{\phi}$. Синим слева направо обозначены точки p_{-2}, p_{-1}, p_1, p_2, из которых берутся значения для интерполяции

Fig. 3. Interpolation scheme in different cases. The face in the middle of which the interpolation $\boldsymbol{\phi}$ is sought is marked in red. The points p_{-2}, p_{-1}, p_1, p_2 from which the values for interpolation are taken are marked in blue from left to right

Если точка p_i находится в центре раздробленной ячейки, то значение $\boldsymbol{\phi}(p_i)$ находится осреднением. Во всех остальных случаях $\boldsymbol{\phi}(p_i)$ равно значению в ячейке, к которой относится p_i.

Потоки вычисляются на основании приближенного решения методом HLL задачи о распаде разрыва на границе между веществом с параметрами $\boldsymbol{\phi}_{ij}^L$ и $\boldsymbol{\phi}_{ij}^R$:

$$F_{ij} = F_{\text{HLL}}\left(\boldsymbol{\phi}_{ij}^L, \boldsymbol{\phi}_{ij}^R\right). \tag{3}$$

Доли первого и второго вещества в потоке F_{ij} берутся равными долям первого и второго вещества в ячейке, из которой смесь вытекает.

Алгоритм, реализующий эту схему, будет выглядеть следующим образом: пусть есть входной слой U^n, промежуточный слой $U^{n+\frac{1}{2}}$ и конечный слой U^{n+1}.

1. Шаг предиктора.

1.1. Заполнение виртуальных ячеек.

При реализации на сетках octreemesh здесь происходит заполнение ячеек на границе расчетной области и на границах между доменами MPI-процессов. При реализации на сетках AMReX здесь происходит заполнение слоев виртуальных ячеек всех блоков. Описанная схема не требует интерполяции на границах раздела разных уровней, поэтому в дочерние виртуальные ячейки записываются значения из родительских.

1.2. Обход всех граней сетки. Вычисление потоков через грани.

На сетках octreemesh далее идет определение точек шаблона интерполяции для грани. Пусть ищется поток через грань γ, соединяющую ячейки c_{-1}, c_1, и пусть внутренняя нормаль этой грани относительно c_1 направлена по оси Ox в положительном направлении. Значения $\boldsymbol{\phi}_{-1}$ и $\boldsymbol{\phi}_1$ берутся из ячеек c_{-1} и c_1 соответственно.

Далее, если ячейки одного уровня дробления, то ищутся ячейки соседние с c_1 по направлению $+x$ и ячейки соседние с c_{-1} по направлению $-x$. Если по этим направлениям ячейки того же или меньшего уровня, то значения $\boldsymbol{\phi}_{-2}$ и $\boldsymbol{\phi}_2$ берутся из них. Если по какому-то из этих направлений ячейки уровнем выше, то $\boldsymbol{\phi}$ находится осреднением по ним.

Если уровень c_{-1} больше уровня c_1, то $\boldsymbol{\phi}_2$ берется из c_1, а $\boldsymbol{\phi}_{-2}$ — из ячейки или ячеек, соседних с c_{-1} по направлению $-x$. Случай, когда уровень c_{-1} меньше уровня c_1, рассматривается аналогично.

По формулам (1)–(3) вычисляются интерполяция на грань справа и слева $\boldsymbol{\phi}^L, \boldsymbol{\phi}^R$ и потоки $F_{\text{HLL}}\left(\boldsymbol{\phi}^L, \boldsymbol{\phi}^R\right)$. Полученные потоки добавляются к значениям в ячейке c_1 и отнимаются от значений в ячейке c_{-1} на промежуточном слое.

На сетках AMReX внутри каждого блока определение точек интерполяции и значений в них идет проще, так как внутри одного блока все ячейки одинаковы, а осреднения и интерполяции на границах раздела разных уровней дробления уже проведены. Таким образом, для грани (i, j, k) по направлению x:

$$\boldsymbol{\phi}_{-2} = \boldsymbol{\phi}(i-2, j, k), \quad \boldsymbol{\phi}_{-1} = \boldsymbol{\phi}(i-1, j, k), \quad \boldsymbol{\phi}_1 = \boldsymbol{\phi}(i, j, k), \quad \boldsymbol{\phi}_2 = \boldsymbol{\phi}(i+1, j, k).$$

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

Потоки F_{HLL} также вычисляются по формулам (1)–(3). Но здесь появляется еще одна особенность. На границе раздела двух уровней дробления поток через грань, вычисленный в блоке снизу, и сумма потоков через "дочерние" грани в блоке сверху могут быть неравны, что может приводить к неконсервативности схемы. Во избежание этого необходимо производить коррекцию потоков через грани, покрытые гранями большего уровня. Эта операция может включать в себя межпроцессорные обмены, так как блок сверху и блок снизу не обязаны принадлежать одному и тому же MPI-процессу. Необходимость этой операции также является причиной необходимости содержать отдельно потоки через каждую грань сетки.

1.3. Обход всех ячеек сетки. Вычисление новых значений на промежуточном слое.

На сетках AMReX значения консервативных параметров вычисляются следующим образом:

$$U^{n+\frac{1}{2}}(i,j,k) = U^n(i,j,k) - \frac{\tau}{2h}\cdot\left(F_x(i+1,j,k) + F_y(i,j+1,k) + F_z(i,j,k+1) - \right.$$
$$\left. - F_x(i,j,k) - F_y(i,j,k) - F_z(i,j,k)\right).$$

где индекс при U относится к ячейкам, индекс при F — к граням по соответствующему направлению.

На сетках octreemesh этот шаг уже выполнен в прошлом пункте. Далее на промежуточном слое учитывается поле силы тяжести в виде поправок $\frac{\tau}{2}\rho g$ для значений $(\rho u)^{n+\frac{1}{2}}$ и $\frac{\tau}{2}\rho u g$ для $E^{n+\frac{1}{2}}$ и рассчитываются величины

$$(\rho\varepsilon) = E - \rho\frac{v^2+u^2+w^2}{2}, \quad p = (\gamma-1)\rho\varepsilon, \quad T = \frac{\varepsilon}{C_v}.$$

2. Шаг корректора. Все операции выполняются аналогично шагу предиктора.

Особенностью реализации численной схемы с использованием библиотеки AMReX является необходимость выполнения расчетов в нелистовых элементах сетки.

6. Априорная оценка ожидаемой производительности методов. В блочной модели каждый блок представляет собой небольшую регулярную сетку и адаптивную сетку как набор связанных между собой небольших регулярных сеток. Листовой подход рассматривает адаптивную сетку как частный случай неструктурированной сетки. Разница в подходах к представлению сетки позволяет предположить выигрыш по времени выполнения одной итерации расчета для блочного подхода в силу возможности оптимизации работы с памятью и возможности использования прямой адресации при обращении к смежным сеточным элементам. При этом листовой подход позволяет измельчать не области, а конкретные ячейки, что может дать выигрыш в количестве элементов для листового подхода. Особенность постановки задачи, в которой возмущения расположены вдоль плоскости, перпендикулярной оси Ox, позволяет предположить небольшую сравнительную величину выигрыша по количеству ячеек для листового подхода.

Листовой подход содержит затратную по времени и числу операций процедуру перестроения и балансировки сетки, что позволяет предположить меньшее время на выполнение процедуры адаптации и балансировки в блочном подходе.

7. Сравнение расчетов.

7.1. Модельная задача. Сравнение подходов производилось на примере модельной задачи (см. раздел 5) в следующей постановке: в расчетной области $[0,1]\times[0,1]\times[0,1]$ см^3 находятся два вещества, характеристики которых в начальный момент времени представлены в табл. 1. Функция, описывающая неровности границы раздела веществ:

$$f(y,z) = 0.005\sin(25z) + 0.007\cos(18y) + 0.0025\sin(60(y+z)).$$

Движение слоев веществ происходит под действием силы тяжести с ускорением $g = 0.15\frac{\text{см}}{\text{мкс}^2}$, направленным по оси Ox.

7.2. Результаты расчетов. Вычислительные эксперименты выполнены с целью сравнить следующие характеристики библиотек: общее количество не виртуальных сеточных элементов, участвующих в расчете, время выполнения одной итерации, время выполнения операций перестроения и балансировки сетки.

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

Таблица 1. Характеристики веществ в начальный момент времени

Table 1. Parameters of substances at the initial moment of time

Substance No.	x, cm	ρ, g/cm^3	p, g/(cm·us^2)	γ	C_v, 10^5 J/(g·keV)
1	$x < 0.5 + f(y,z)$	2.0	2.0	1.4	2.5
2	$x > 0.5 + f(y,z)$	1.0	2.0	1.4	5

Первая серия расчетов выполнялась со следующими параметрами:

1. Начальная сетка: $100 \times 100 \times 100$.
2. Максимальный уровень адаптации: 2.
3. Критерий адаптации: по концентрации первого вещества $0.001 < \dfrac{\rho_1}{\rho} < 0.999$. При этом в начальный момент времени раздроблены ячейки с $0.48 + f(y,z) < x < 0.52 + f(y,z)$.
4. Ширина слоя виртуальных ячеек блоков и доменов: 2.
5. Параметры построения сеток AMReX: максимальный линейный размер блока — 32, минимальная сеточная эффективность $\text{eff}_{\text{min}} = 0.8$.
6. Частота адаптаций: каждые 200 шагов.
7. Вычислительные ресурсы: 112 MPI-процессов в однопоточном режиме, моделирование производилось на вычислительном кластере K-60 [17].

Расчеты этой серии проводились до $t = 6$ мкс с фиксированным шагом по времени $\tau = 5 \cdot 10^{-4}$ мкс. Таким образом, каждый расчет был выполнен за 12000 итераций.

На рис. 4 представлено распределение концентрации первого вещества и структура сетки в обоих расчетах. На рис. 5, 6 представлены графики для сравнения эффективности использования сеток по количеству сеточных элементов и времени, потраченному на расчет. На всех графиках синие линии — AMReX, оранжевые — octreemesh.

Во второй серии проводилось 3 расчета: один расчет на сетках octreemesh и два расчета на сетках AMReX. Общие параметры были следующими:

1. Начальная сетка: $100 \times 100 \times 100$.
2. Максимальный уровень адаптации: 2.
3. Критерий адаптации: по концентрации первого вещества $0.001 < \dfrac{\rho_1}{\rho} < 0.999$. При этом в начальный момент времени раздроблены ячейки с $0.48 + f(y,z) < x < 0.52 + f(y,z)$.

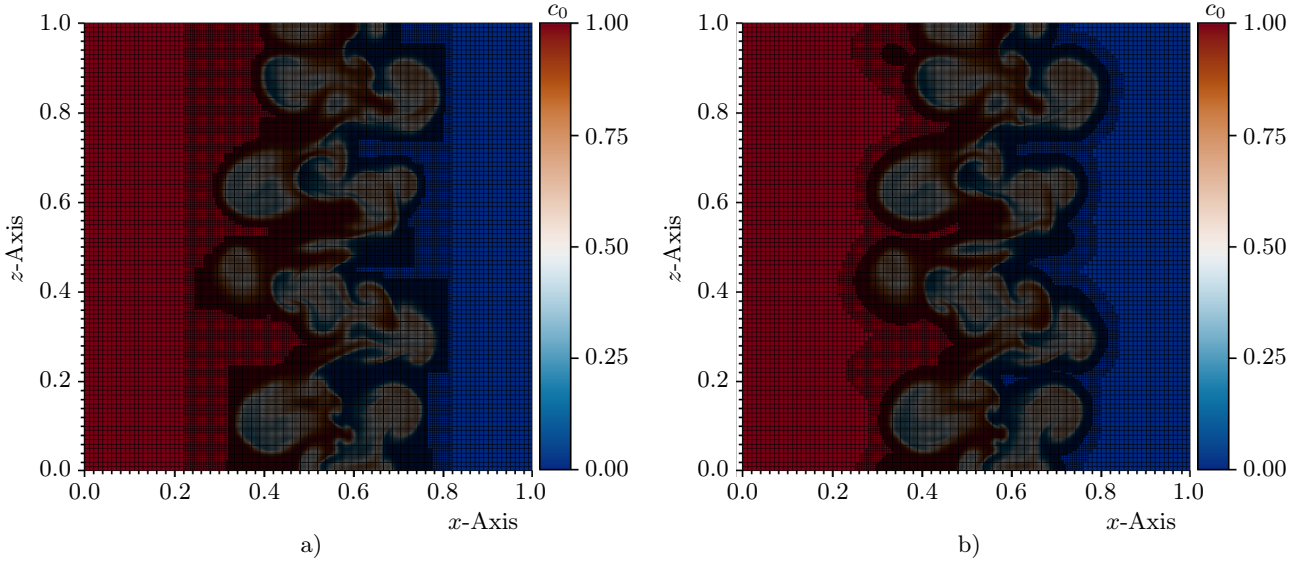

Рис. 4. Концентрация первого вещества на плоскости $y = 0.5$ см в момент времени $t = 6$ мкс: a) AMReX; b) octreemesh

Fig. 4. Concentration of the first substance on the plane $y = 0.5$ cm while $t = 6$ μs: a) AMReX; b) octreemesh

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

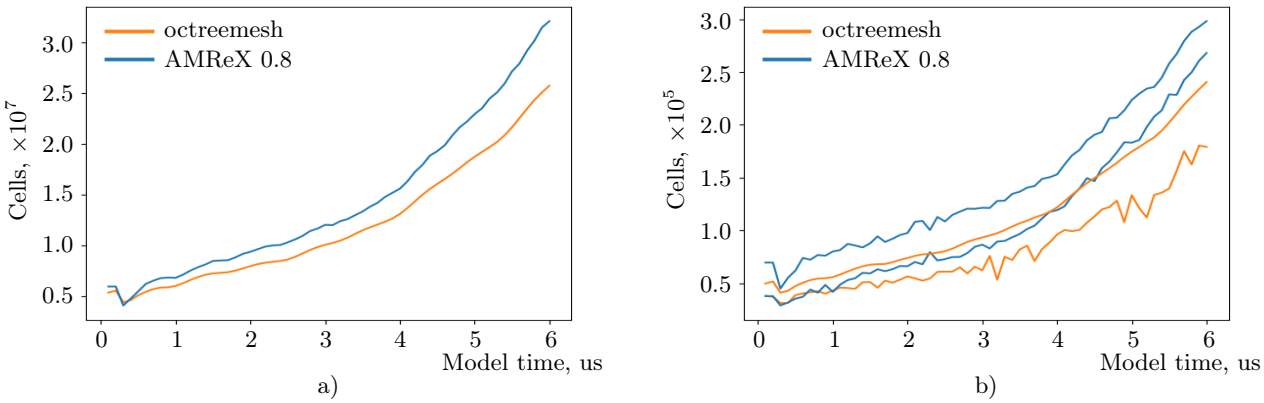

Рис. 5. Эффективность использования сеток: a) полные размеры сетки; b) максимальный и минимальный размер домена

Fig. 5. Mesh efficiency: a) full mesh sizes; b) maximum and minimum domain sizes

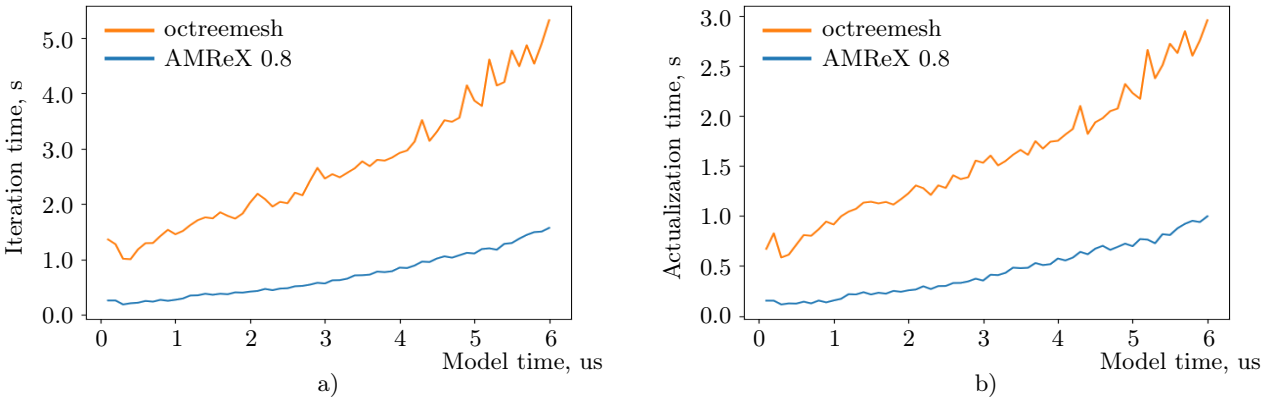

Рис. 6. Эффективность использования сеток: a) среднее время на расчет одного шага по времени; b) среднее время на все операции актуализации данных за расчет одного шага по времени

Fig. 6. Mesh efficiency: a) average time for one iteration; b) average time for all actualization operations done in one iteration

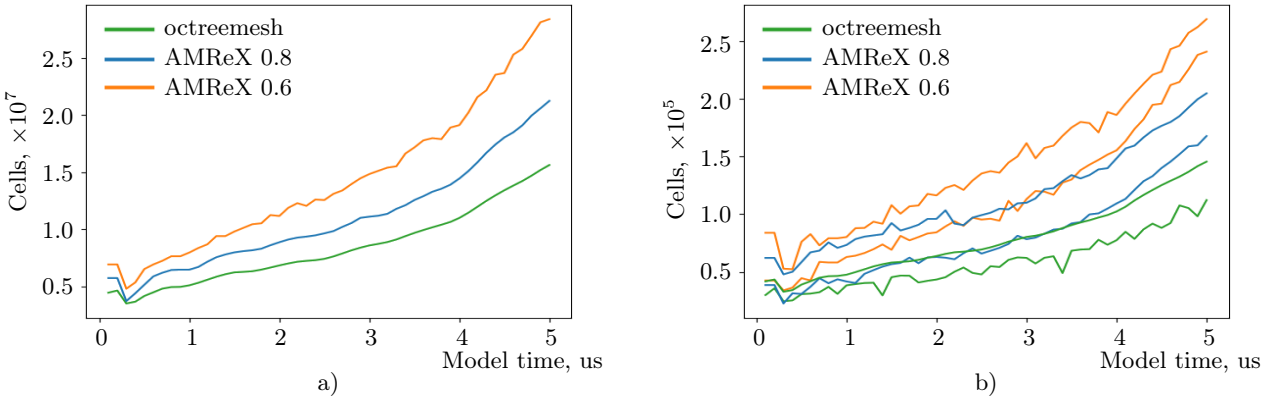

Рис. 7. Эффективность использования сеток: a) полные размеры сетки; b) максимальный и минимальный размер домена

Fig. 7. Mesh efficiency: a) full mesh sizes; b) maximum and minimum domain sizes

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

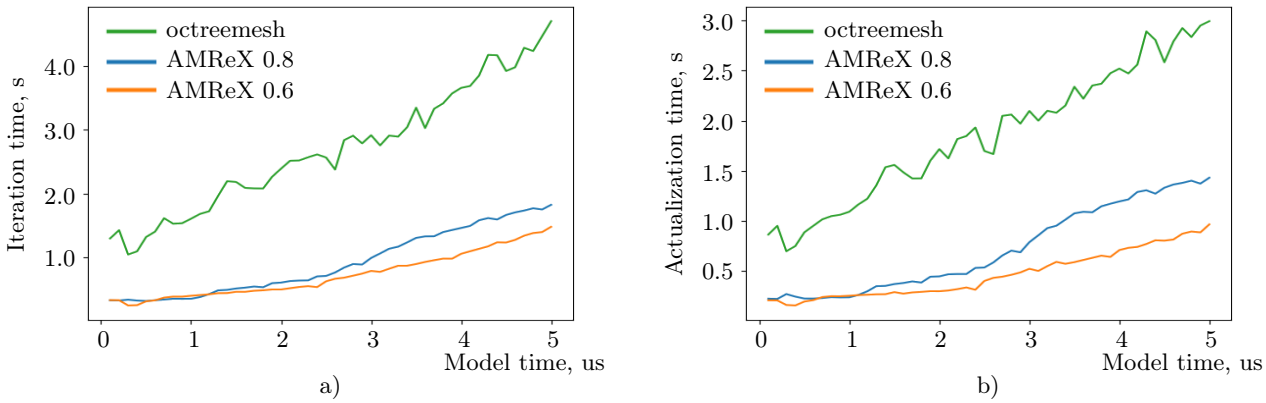

Рис. 8. Эффективность использования сеток: a) среднее время на расчет одного шага по времени; b) среднее время на все операции актуализации данных за расчет одного шага по времени

Fig. 8. Mesh efficiency: a) average time for one iteration; b) average time for all actualization operations done in one iteration

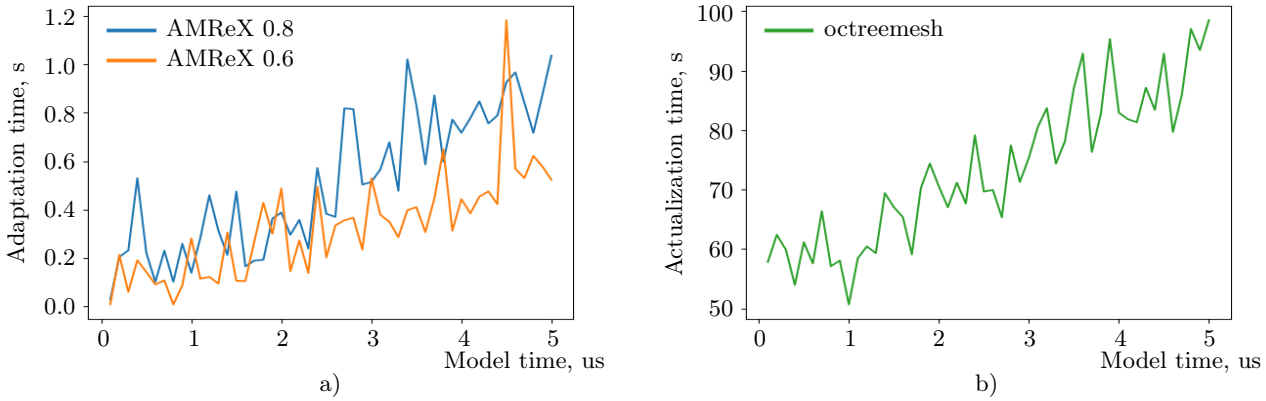

Рис. 9. Время одной адаптации: a) AMReX; b) octreemesh

Fig. 9. Time for one adaptation procedure: a) AMReX, b) octreemesh

4. Ширина слоя виртуальных ячеек блоков и доменов: 3.
5. Максимальный линейный размер блока на сетках AMReX: 32.
6. Частота адаптаций: каждые 250 шагов.
7. Вычислительные ресурсы: 112 MPI-процессов в однопоточном режиме, моделирование производилось на вычислительном кластере K-60 [17].

Расчеты этой серии проводились до $t = 5$ мкс с фиксированным шагом по времени $\tau = 4 \cdot 10^{-4}$ мкс. Отличие двух расчетов на сетках AMReX — в минимальной сеточной эффективности: в одном $\mathrm{eff}_{\min} = 0.8$, в другом $\mathrm{eff}_{\min} = 0.6$.

Для сравнения на графиках (рис. 7–9) зелеными линиями показаны результаты расчетов посредством библиотеки octreemesh, синими линиями — AMReX с $\mathrm{eff}_{\min} = 0.8$, оранжевыми линиями — AMReX с $\mathrm{eff}_{\min} = 0.6$.

График на рис. 10 показывает нарастание амплитуды возмущений в зависимости от времени. До модельного времени $t = 4.0$ мкс полученные результаты хорошо соотносятся с экспоненциальной функцией роста возмущений. Далее, когда амплитуда становится порядка длины волны, скорость роста начинает замедляться.

Количество сеточных элементов, участвующих в расчетах на сетках листовой модели, оказывается меньше, чем на сетках блочной модели. Отношение варьируется в зависимости от параметра минимальной сеточной эффективности и геометрии области, маркированной функциональным критерием. Если в

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

начале расчетов, когда область адаптации хорошо аппроксимируется прямоугольным параллелепипедом, количество сеточных элементов практически не отличается, то к концу расчета, когда вещества смешиваются и граница между ними начинает иметь менее тривиальную форму, отношение размеров сеток может достигать 2:1.

Отдельно стоит отметить такую особенность блочных сеток, как ускорение расчета при уменьшении количества блоков, несмотря на существенное увеличение количества сеточных элементов. Как можно видеть из рис. 11, 12, сетка при $\mathrm{eff}_{\min} = 0.6$ состоит из меньшего количества более крупных блоков, при этом общее время расчета для $\mathrm{eff}_{\min} = 0.8$ больше, чем для $\mathrm{eff}_{\min} = 0.6$.

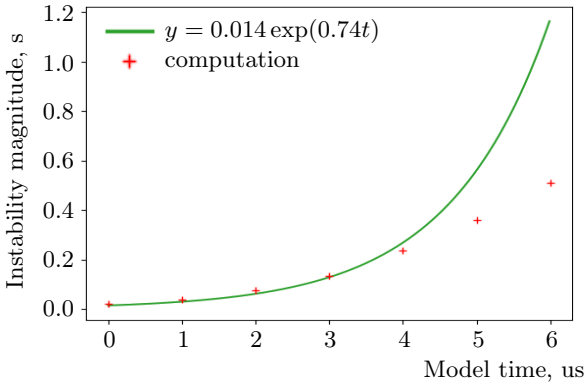

Рис. 10. График зависимости амплитуды возмущений от времени

Fig. 10. Instability amplitude evolution in time

Общее время на одну итерацию занимает меньше времени на блочных сетках. В начале расчета, когда отношение размеров сеток меньше, на сетках AMReX расчеты оказываются в 4–5 раз быстрее, к концу расчета отношение уменьшается примерно до 3:1. Время на актуализацию данных в виртуальных ячейках также меньше на блочных сетках, но при этом листовые сетки оказываются менее подвержены замедлению при увеличении слоя виртуальных ячеек: в расчетах с шириной слоя 2 отношение между временем актуализации на сетках octreemesh и на сетках AMReX с $\mathrm{eff}_{\min} = 0.8$ на 5 мкс равно 3.19:1, в расчетах с шириной 3 это же отношение равно 2.1:1.

Время на адаптацию и балансировку загрузки на сетках octreemesh оказывается значительно больше, чем на сетках AMReX. Если на сетках AMReX эти операции занимают пренебрежимо малое время относительно всего остального расчета, то на сетках octreemesh на адаптацию и балансировку тратится около 10% всего времени.

Для оценки масштабируемости расчетов небольшая часть расчета также была проведена с использованием разного количества MPI-процессов. Сетка в каждом из этих расчетов строилась по одному и тому же геометрическому критерию. Размеры сеток получались близкими к 9, 12, 16 млн для octreemesh, AMReX с $\mathrm{eff}_{\min} = 0.8$ и $\mathrm{eff}_{\min} = 0.6$ соответственно.

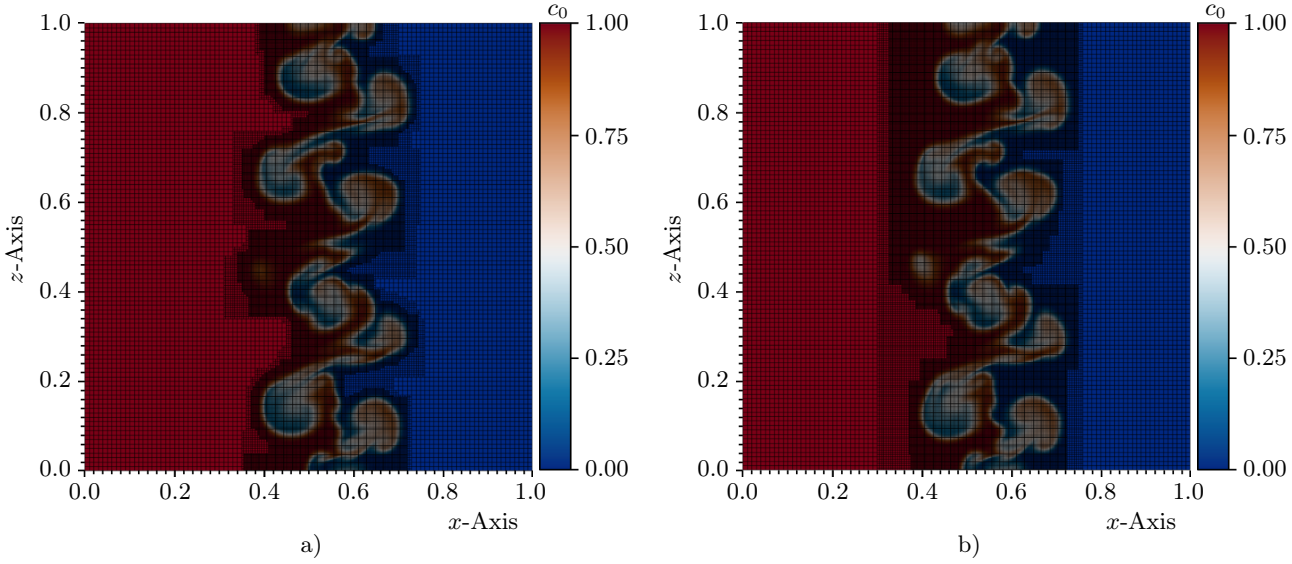

Рис. 11. Концентрация первого вещества на плоскости $y = 0.5$ см в момент времени $t = 5$ мкс: a) AMReX при $\mathrm{eff}_{\min} = 0.8$; b) AMReX при $\mathrm{eff}_{\min} = 0.6$

Fig. 11. Concentration of the first substance on the plane $y = 0.5$ cm while $t = 5$ μs: a) AMReX with $\mathrm{eff}_{\min} = 0.8$; b) AMReX with $\mathrm{eff}_{\min} = 0.6$

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

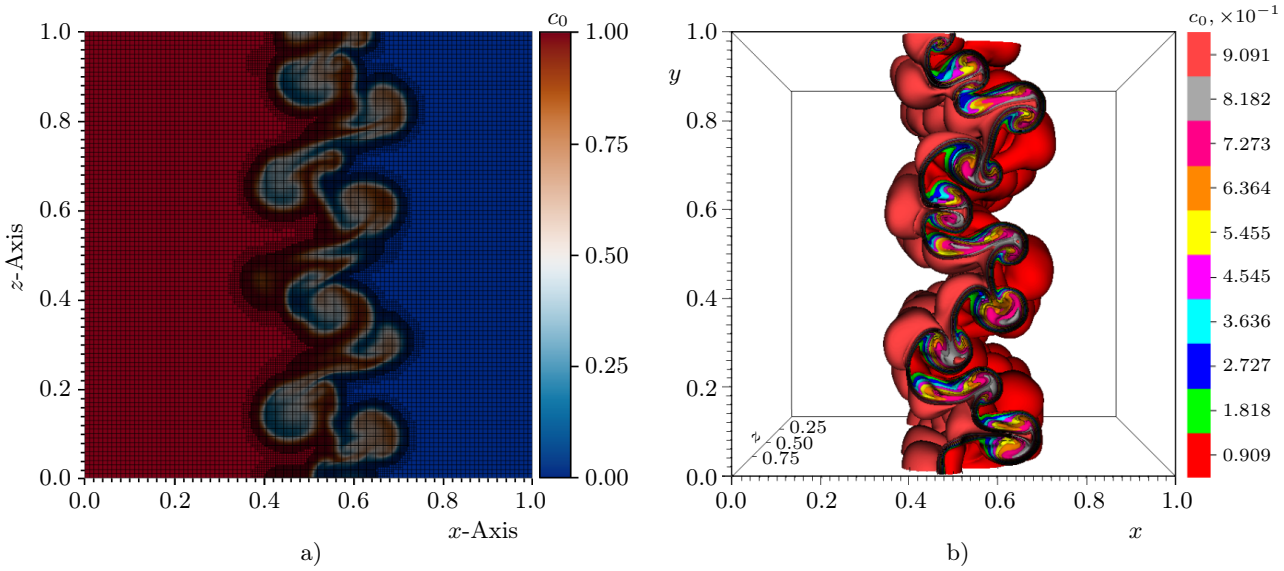

Рис. 12. Расчеты на сетках octreemesh: a) концентрация первого вещества на плоскости $y = 0.5$ см в момент времени $t = 5$ мкс; b) изоповерхности концентрации первого вещества

Fig. 12. Calculations on octreemesh: a) concentration of the first substance on the plane $y = 0.5$ cm at time $t = 5$ us; b) isosurfaces of the concentration of the first substance

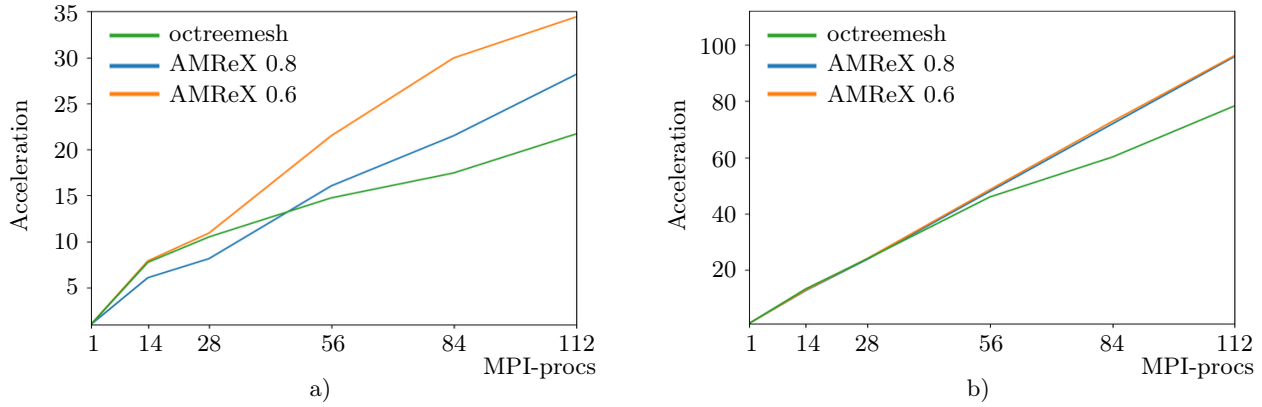

Рис. 13. Ускорение в зависимости от количества MPI-процессов: a) всего расчета; b) без учета операций с обменами

Fig. 13. Acceleration obtained using mpi: a) full iteration; b) excluding communication operations

Как можно видеть из рис. 13, блочные сетки показывают лучшую масштабируемость при использовании одинакового критерия адаптации. При этом ускорение заметно падает при увеличении параметра $\mathrm{eff}_{\min}$ и, как следствие, увеличении общего числа блоков сетки. Отчасти это можно объяснить тем, что сетки листового типа и сетки с большим параметром $\mathrm{eff}_{\min}$ получаются меньше в размере, из-за чего отношение количества ячеек к количеству граничных ячеек на одном MPI-процессе увеличивается и дополнительно при этом растет относительная несбалансированность загрузки MPI-процессов.

8. Заключение. Изначальное предположение о том, что листовая модель позволяет сэкономить количество сеточных элементов за счет увеличения времени обработки одного сеточного элемента в процессе расчета, подтвердилось. Как показывают вычислительные эксперименты, экономия существенно зависит от геометрии адаптируемой области.

Время на выполнение операции адаптации и балансировки загрузки для листового метода оказалось ожидаемо выше, чем для блочного.

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

Блочные сетки быстрее теряют эффективность при увеличении ширины слоя виртуальных ячеек; следовательно, есть причины полагать, что эффективность листовых сеток будет сохраняться лучше при расширении шаблона схемы.

Исследование масштабируемости показало, что блочные сетки в среднем показывают лучшее ускорение расчета при увеличении вычислительных ресурсов, при этом стоит заметить, что ускорение заметно снижается при увеличении числа блоков сетки.

Также можно отметить, что на листовых сетках при расчете взаимодействия ячеек разных уровней дробления можно работать со значениями в этих ячейках напрямую, в то время как на блочных сетках доступны лишь интерполированные или осредненные значения, поскольку каждый блок обязан содержать ячейки одного уровня. Это приводит к ограничению на схемы, которые можно просто и эффективно реализовывать на блочных сетках, в то время как для листовых сеток подобного ограничения не существует.

Все сравнения в данной работе выполнялись с использованием явных схем, построенных методом конечных объемов, которые хорошо подходят для решения вычислительных задач газодинамики и которые, вследствие своей универсальности, эффективно и просто реализуются на адаптивных сетках различных типов. Значительный интерес для последующих исследований представляет изучение эффективности использования octree-сеток различных типов в неявных схемах, так как неявные схемы обладают известными преимуществами и широко используются при моделировании процессов тепломассообмена, распространения теплового излучения или, например, облучения мишеней лазером и разлета лазерной плазмы. Для решения систем линейных уравнений большой размерности, возникающих при моделировании подобных задач, оптимальными могут оказаться матричные, безматричные или мультисеточные методы в зависимости от типа и размера используемой сетки. Актуальной также является разработка параллельной реализации сеточных методов на вычислительных кластерах и графических ускорителях.

Список литературы

1. *Berger M.J., Colella P.* Local adaptive mesh refinement for shock hydrodynamics // J. Comput. Phys. 1989. **82**, N 1. 64–84. doi 10.1016/0021-9991(89)90035-1.
2. *Peraire J., Patera A.T.* Bounds for linear-functional outputs of coercive partial differential equations: local indicators and adaptive refinement // Studies in Applied Mechanics. Vol. 47. Amsterdam: Elsevier, 1998. 199–216.
3. *Суков С.А.* Визуализация результатов расчетов газодинамических течений на смешанных локально-адаптивных сетках // Научная визуализация. 2021. **13**, № 5. 52–64. doi 10.26583/sv.13.5.05.
4. *Burstedde C., Wilcox L.C., Ghattas O.* p4est: scalable algorithms for parallel adaptive mesh refinement on forests of octrees // SIAM J. Sci. Comput. 2011. **33**, N 3. 1103–1133. doi 10.1137/100791634.
5. *Holke J., Burstedde C., Knapp D., et al.* t8code v. 1.0 — modular adaptive mesh refinement in the exascale era // Proc. SIAM Int. Meshing Round Table 2023, Amsterdam, Netherlands, March 6–9, 2023. https://github.com/DLR-AMR/t8code. Cited January 12, 2025.
6. *Wünsch R., Walch S., Dinnbier F., Whitworth A.* Tree-based solvers for adaptive mesh refinement code flash — I: gravity and optical depths // Mon. Not. R. Astron. Soc. 2018. **475**, N 3. 3393–3418. doi 10.1093/mnras/sty015.
7. *Klein Y.Y.* Construction of a multidimensional parallel adaptive mesh refinement special relativistic hydrodynamics code for astrophysical applications // doi 10.48550/arXiv.2310.02331.
8. *MacNeice P., Olson K.M., Mobarry C., et al.* PARAMESH: a parallel adaptive mesh refinement community toolkit // Computer Physics Communications, 2000. **126**, N 3. 330–354. doi 10.1016/S0010-4655(99)00501-9.
9. *Zhang W., Almgren A., Beckner V., et al.* AMReX: a framework for block-structured adaptive mesh refinement // J. Open Source Softw. 2019. **4**, N 37. Article Number 1370. doi 10.21105/joss.01370.
10. *Adams M., Colella P., Graves D.T., et al.* Chombo software package for AMR applications design document. Technical Report LBNL-6616E. Berkeley: Lawrence Berkeley Nat. Lab., 2015. https://crd.lbl.gov/assets/pubs_presos/chomboDesign.pdf. Cited January 12, 2025.
11. *Bryan G.L., Norman M.L., O'Shea B.W., et al.* Enzo: an adaptive mesh refinement code for astrophysics // Astrophys. J. Suppl. Ser. 2014. **211**, N 2. Article Number 19. doi 10.1088/0067-0049/211/2/19.
12. SAMRAI: Structured Adaptive Mesh Refinement Application Infrastructure (2017). https://computing.llnl.gov/projects/samrai. Cited January 12, 2025.

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

13. *Hornung R.D., Kohn S.R.* Managing application complexity in the SAMRAI object-oriented framework // Concurr. Comput. Pract. Exp. 2002. **14**, N 5. 347–368. doi 10.1002/cpe.652.
14. *Abdi D.S., Almgren A., Giraldo F.X., Jankov I.* Comparison of adaptive mesh refinement techniques for numerical weather prediction // doi 10.48550/arXiv.2404.16648.
15. *Karypis G.* METIS and ParMETIS // Encyclopedia of Parallel Computing. Boston: Springer, 2011. 1117–1124.
16. *Toro E.F.* Riemann solvers and numerical methods for fluid dynamics: a practical introduction. Berlin: Springer, 2009. doi 10.1007/b79761.
17. Центр коллективного пользования ИПМ им. М. В. Келдыша РАН. Гибридный вычислительный комплекс К60. https://ckp.kiam.ru Cited January 12, 2025.

Поступила в редакцию
7 ноября 2024 г.

Принята к публикации
10 января 2025 г.

Информация об авторах

Сергей Константинович Григорьев — мл. научн. сотр.; Институт прикладной математики имени М. В. Келдыша РАН (ИПМ РАН), Миусская пл., д. 4, 125047, Москва, Российская Федерация.

Антон Алексеевич Бай — аспирант; Институт прикладной математики имени М. В. Келдыша РАН (ИПМ РАН), Миусская пл., д. 4, 125047, Москва, Российская Федерация.

References

1. M. J. Berger and P. Colella, "Local Adaptive Mesh Refinement for Shock Hydrodynamics," J. Comput. Phys. **82** (1), 64–84 (1989). doi 10.1016/0021-9991(89)90035-1.
2. J. Peraire and A. T. Patera, "Bounds for Linear-Functional Outputs of Coercive Partial Differential Equations: Local Indicators and Adaptive Refinement," in *Studies in Applied Mechanics* (Elsevier, Amsterdam, 1998), Vol. 47, pp. 199–216.
3. S. A. Soukov, "Visualization of the CFD Calculations Results on Adaptive Mixed Meshes," Sci. Vis. **13** (5), 52–64 (2021). doi 10.26583/sv.13.5.05.
4. C. Burstedde, L. C. Wilcox, and O. Ghattas, "p4est: Scalable Algorithms for Parallel Adaptive Mesh Refinement on Forests of Octrees," SIAM J. Sci. Comput. **33** (3), 1103–1133 (2011). doi 10.1137/100791634.
5. J. Holke, C. Burstedde, D. Knapp, et al., "t8code v. 1.0 - Modular Adaptive Mesh Refinement in the Exascale Era," in *Proc. SIAM Int. Meshing Round Table 2023, Amsterdam, Netherlands, March 6–9, 2023* https://github.com/DLR-AMR/t8code. Cited January 12, 2025.
6. R. Wünsch, S. Walch, F. Dinnbier, and A. Whitworth, "Tree-Based Solvers for Adaptive Mesh Refinement Code Flash – I: Gravity and Optical Depths," Mon. Not. R. Astron. Soc. **475** (3), 3393–3418 (2018). doi 10.1093/mnras/sty015.
7. Y. Y. Klein, "Construction of a Multidimensional Parallel Adaptive Mesh Refinement Special Relativistic Hydrodynamics Code for Astrophysical Applications." doi 10.48550/arXiv.2310.02331.
8. P. MacNeice, K. M. Olson, C. Mobarry, et al., "PARAMESH: A Parallel Adaptive Mesh Refinement Community Toolkit," Comput. Phys. Commun. **126** (3), 330–354 (2000). doi 10.1016/S0010-4655(99)00501-9.
9. W. Zhang, A. Almgren, V. Beckner, et al., "AMReX: A Framework for Block-Structured Adaptive Mesh Refinement," J. Open Source Softw. **4** (37), Article Number 1370 (2019). doi 10.21105/joss.01370.
10. M. Adams, P. Colella, D. T. Graves, et al., *Chombo Software Package for AMR Applications Design Document*, Technical Report LBNL-6616E (Lawrence Berkeley Nat. Lab., Berkeley, 2015). https://crd.lbl.gov/assets/pubs_presos/chomboDesign.pdf. Cited January 12, 2025.
11. G. L. Bryan, M. L. Norman, B. W. O'Shea, et al., "Enzo: An Adaptive Mesh Refinement Code for Astrophysics," Astrophys. J. Suppl. Ser. **211** (2), Article Number 19 (2014). doi 10.1088/0067-0049/211/2/19.
12. SAMRAI: Structured Adaptive Mesh Refinement Application Infrastructure (2017). https://computing.llnl.gov/projects/samrai. Cited January 12, 2025.

2025, **26** (1), 33–49. doi 10.26089/NumMet.v26r103

13. R. D. Hornung and S. R. Kohn, "Managing Application Complexity in the SAMRAI Object-Oriented Framework," Concurr. Comput. Pract. Exp. **14** (5), 347–368 (2002). doi 10.1002/cpe.652.
14. D. S. Abdi, A. Almgren, F. X. Giraldo, and I. Jankov, "Comparison of Adaptive Mesh Refinement Techniques for Numerical Weather Prediction." doi 10.48550/arXiv.2404.16648.
15. G. Karypis, "METIS and ParMETIS," in *Encyclopedia of Parallel Computing* (Springer, Boston, 2011), pp. 1117–1124.
16. E. F. Toro, *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction* (Springer, Berlin, 2009). doi 10.1007/b79761.
17. Center for Collective Use of the Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences. Hybrid Supercomputer K60. https://ckp.kiam.ru. Cited January 12, 2025.

Received
November 7, 2024

Accepted for publication
January 10, 2025

Information about the authors

Sergej K. Grigorjev — Junior Scientist; Keldysh Institute of Applied Mathematics of RAS, Miusskaya ploshchad', 4, 125047, Moscow, Russia.

Anton A. Bay — Postgraduate Student; Keldysh Institute of Applied Mathematics of RAS, Miusskaya ploshchad', 4, 125047, Moscow, Russia.