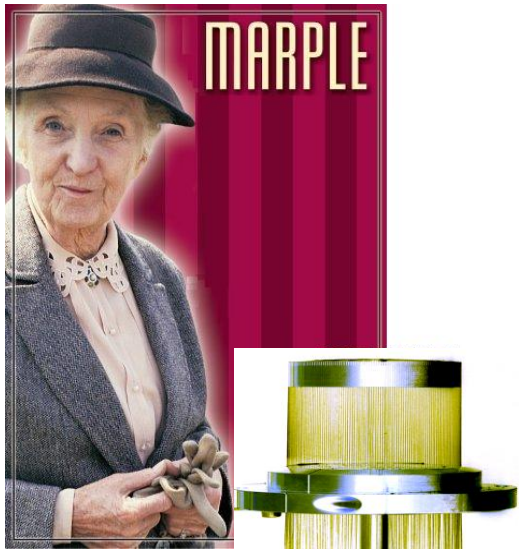
Код MARPLE: решение трехмерных магнитогидродинамических задач и принципы построения кода

А. С. Болдарев, В. А. Гасилов,
О. Г. Ольховская, Г. А. Багдасаров

Проект MARPLE
(Magnetically Accelerated Radiative Plasma Explorer)
Новые вычислительные технологии в вычислительной физике плазмы



- **одножидкостная двухтемпературная магнитогидродинамическая модель** [*С. И. Брагинский, в: Вопросы теории плазмы, Атомиздат, Москва, 1963*]
- **расчет переноса лучистой энергии** сеточно-характеристическим методом [*S. Chandresekhar, Radiative transfer. Oxford, 1951*] и на основе диффузионного приближения.
- **модель "длительного" плазмообразования** [*V.V. Alexandrov,et al., IEEE Transactions on PLASMA SCIENCE, No2 (2002)*]

- **электротехническое уравнение полной цепи** (генератор, подводящие системы и разрядная камера с плазмой)
- **транспортные и кинетические коэффициенты, оптические свойства и уравнения состояния** на базе таблиц [*А.Ф. Никифоров, В.Г. Новиков, В.Б. Уваров , Квантово-статистические модели высокотемпературной плазмы. Физматлит, Москва, 2000*] – код TERMOS (ИПМ им.Келдыша РАН)
- **неструктурированные сетки** с элементами смешанных типов
- **Контрольно-объемные методики повышенной точности** с коррекцией потоков (TVD, ENO - недиссипативная МГД), **интегро-интерполяционные схемы** и **схемы метода Галеркина** с разрывными базисными функциями (диссипативные процессы).

Промышленные и научные коды

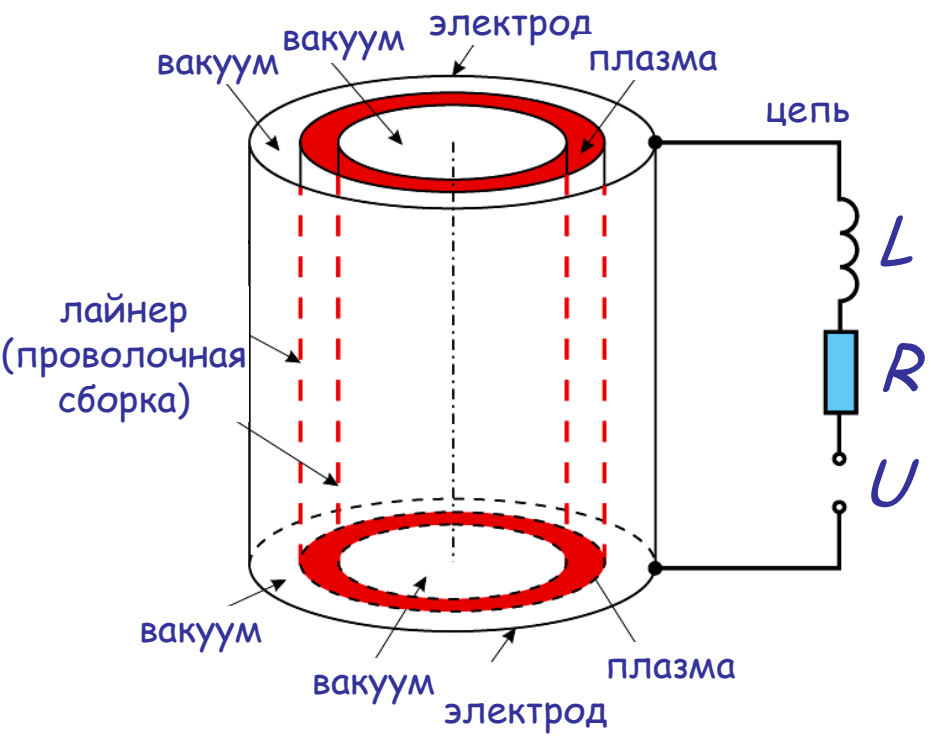
Промышленные коды	Научные коды
Цели: массовые вычисления однотипных вариантов инженерных задач, относящихся к хорошо изученной предметной области	Цели: уникальные расчеты различных вариантов, относящихся к недостаточно исследованным предметным областям; исследование моделей, методов и алгоритмов, численный эксперимент
В постановках задач меняются в основном начальные и граничные условия	Могут меняться начальные и граничные условия, а также модели, методы и алгоритмы
Общие требования: надежность, наличие подробной документации, техническая поддержка	Общие требования: новизна, открытость, точность, возможность исследования и развития моделей, методов и алгоритмов
Используются только известные, изученные и хорошо себя зарекомендовавшие модели, методы и алгоритмы	Могут использоваться любые модели, методы и алгоритмы
Огромные объемы кода, большие коллективы профессиональных разработчиков	Умеренные объемы кода, малые и средние коллективы разработчиков

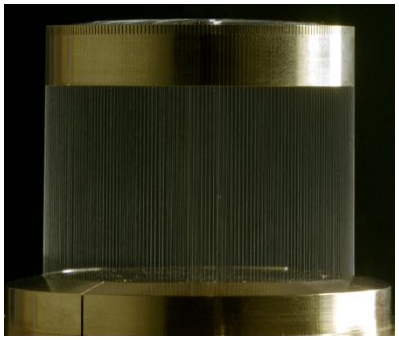
MARPLE - научный код, первоначально предназначенный для поддержки эксперимента в области физики плазмы высокой плотности энергии. При разработке кода мы применяли некоторые подходы, больше характерные для индустриальных кодов:

- Объектно-ориентированное программирование, C++
- Репозиторий текстов с контролем версий, SVN
- Система автоматического документирования Doxygen
- Кроссплатформенная среда разработки CMake

Сжатие лайнера: типичная схема эксперимента

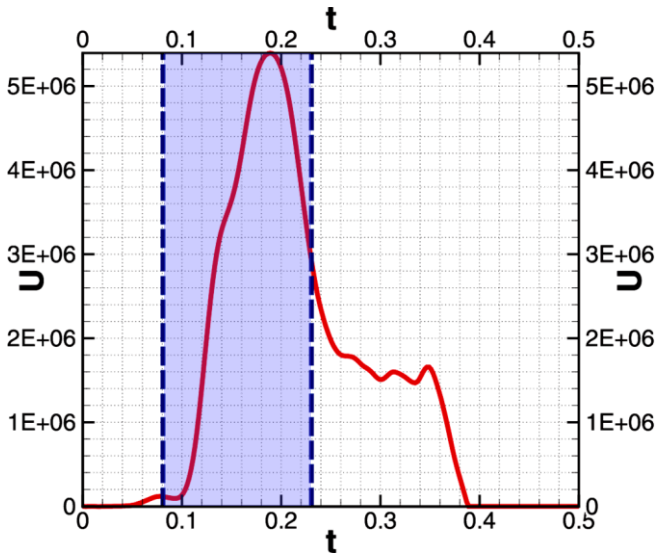
Экспериментальная установка





Цилиндрическая проволочная сборка из 240 вольфрамовых проволок ∅7.5 мкм. Полный диаметр 40 мм.

Профиль напряжения (экспериментальные данные, вариант для 20 МА, ~200 нс)

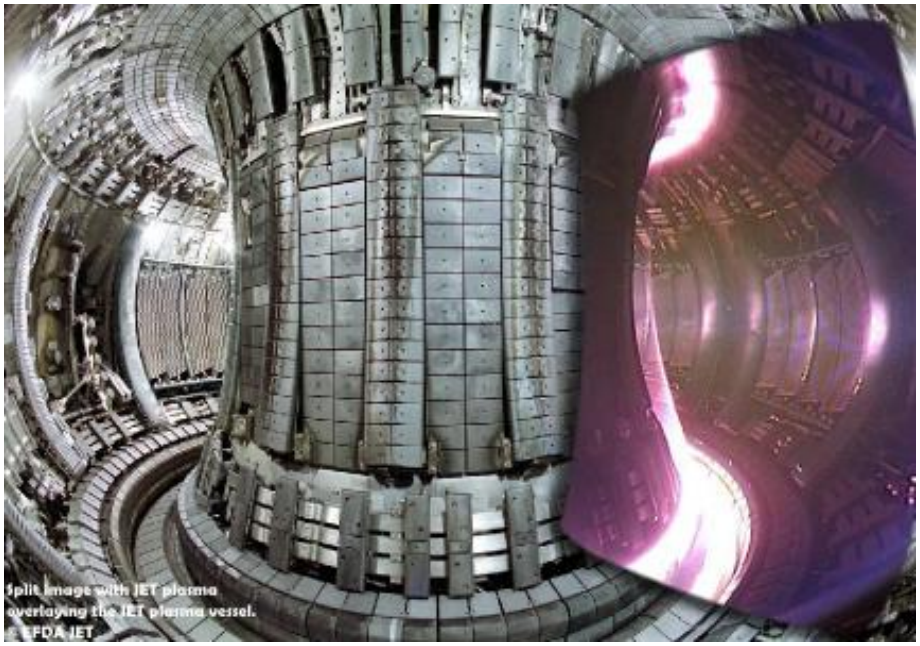


Импульсные установки

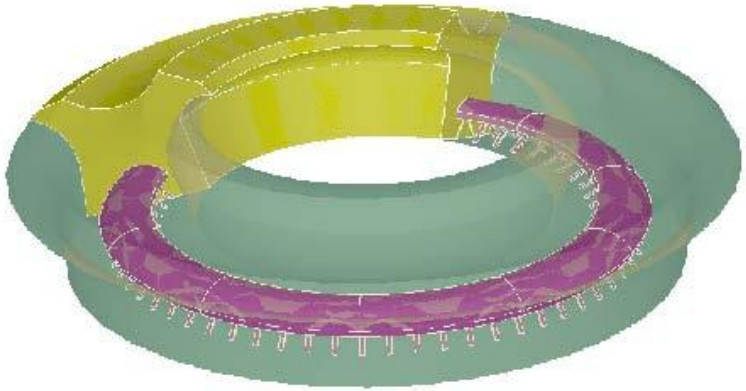
Установка Ангара-5-1,
Троицк, Россия.
Ток разряда до 5 МА,
время нарастания 90 – 100 нс.

Установка ZR, Сандийская Национальная Лаборатория, США.
Ток разряда до 26 МА,
время нарастания 100 нс.

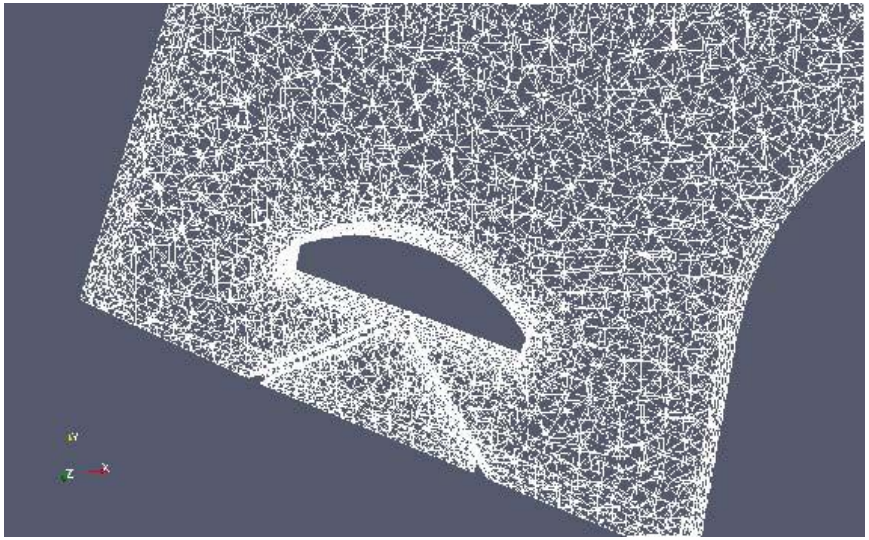
Постановки сложных задач – дивертор токамака



Камера токамака ITER.
Внизу виден дивертор.



Расчетная область для моделирования дивертора

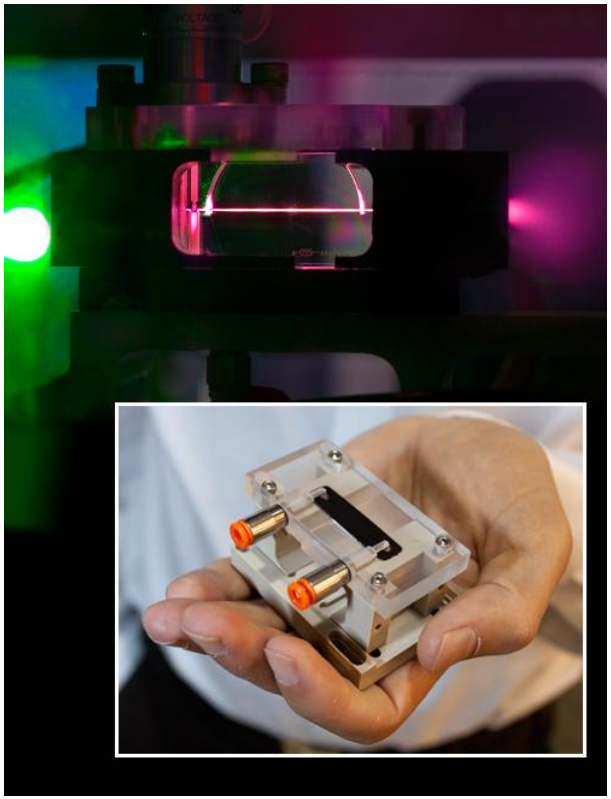


Фрагмент неструктурированной (тетраэдральной) сетки в диверторе

Проект BELLA

BELLA (Berkeley Laboratory Laser Accelerator) [1]:

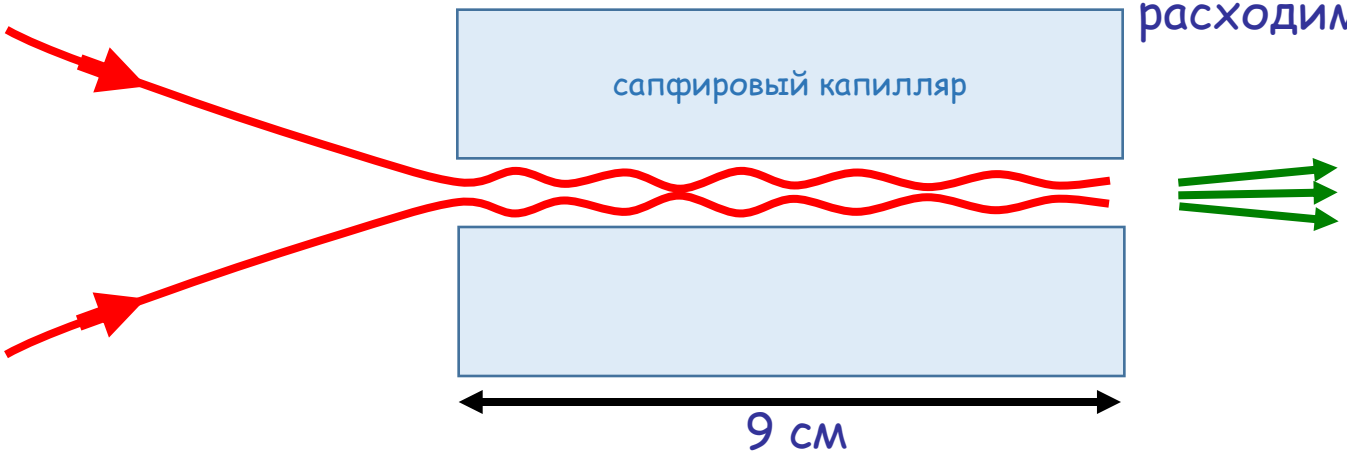
- разработка и применение компактных лазерно-плазменных ускорителей
- ускорение электронного пучка до 10 ГэВ/м



[1] W. P. Leemans et al. // Phys. Rev. Lett **113**, 245002 (2014)

импульс титан-сапфирового лазера: 16 Дж, 40 фс, частота повторений 1 Гц, фокальное пятно 52 мкм, λ = 815 мкм

электронный пучок: $4 \cdot 10^7$ электронов/пучок, 4.2 ГэВ, разброс по энергии 6%, расходимость 0.3 мрад



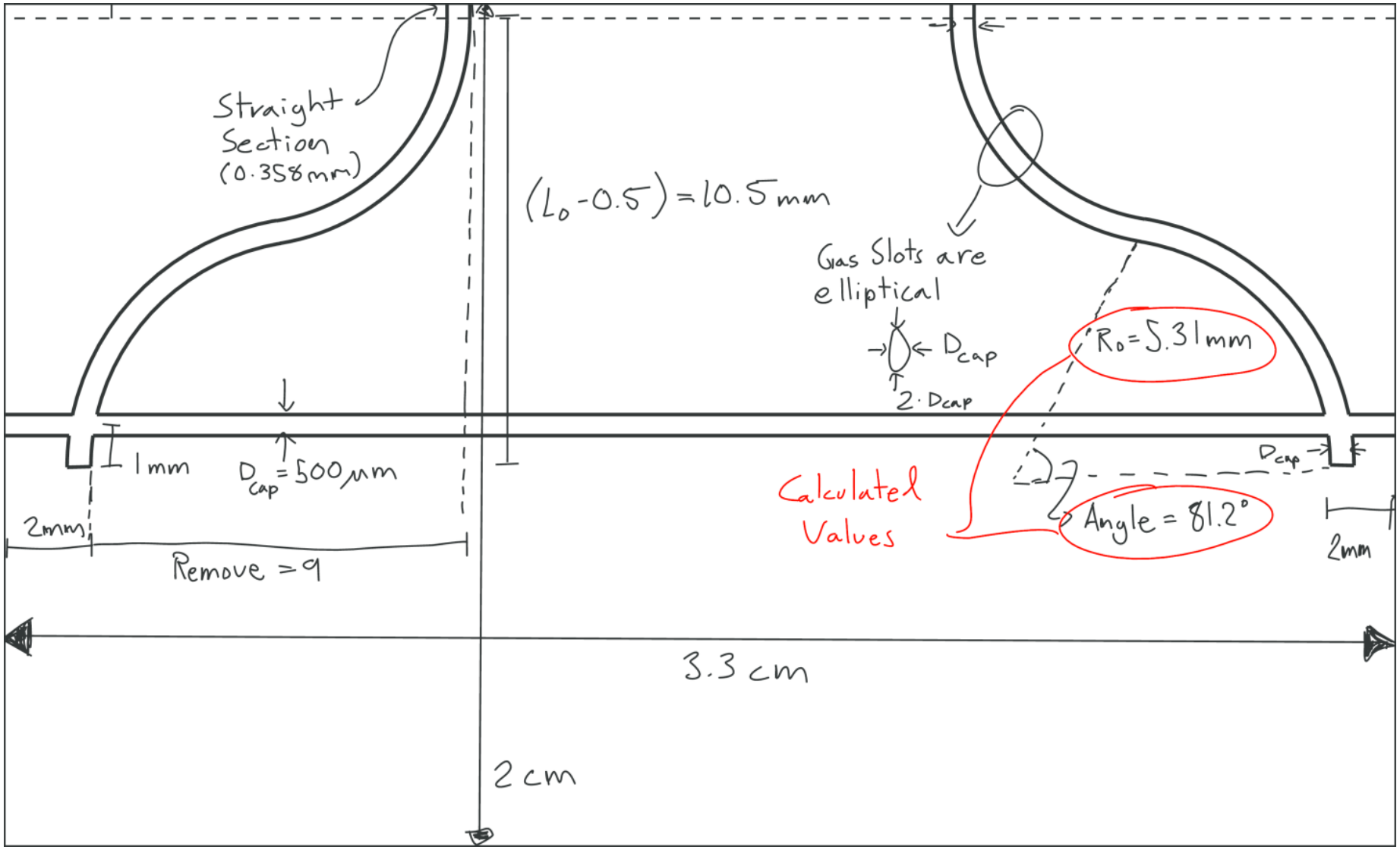
параметры капиллярного разряда:
d = 500 мкм, заполнен водородом, полностью ионизированным,
$n_e = 7 \cdot 10^{17}$ см$^{-3}$
максимальный ток = 250 А,
продолжительность ≈ 90 нс,
диаметр пятна = 70 мкм

Основные этапы эксперимента:

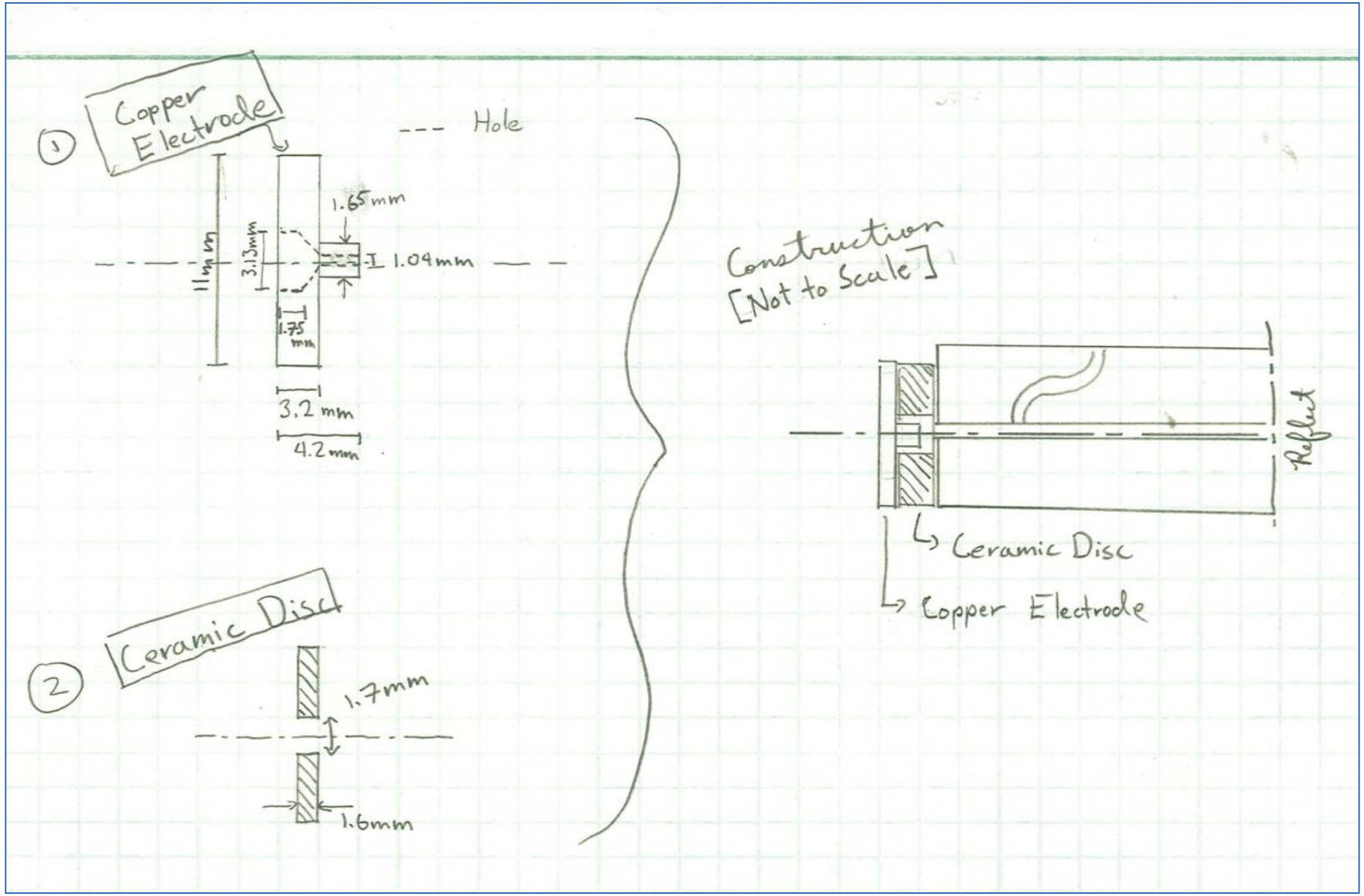
1. Заполнение холодным водородом (гидродинамические эффекты);
2. Капиллярный разряд (МГД);
3. Распространение лазерного импульса.

[1] W. P. Leemans et al. // Phys. Rev. Lett **113**, 245002 (2014)
аналогичные эксперименты: X. Wang et al., Nature Comm. **4**:1988 (2013)

1 стадия: заполнение холодным водородом



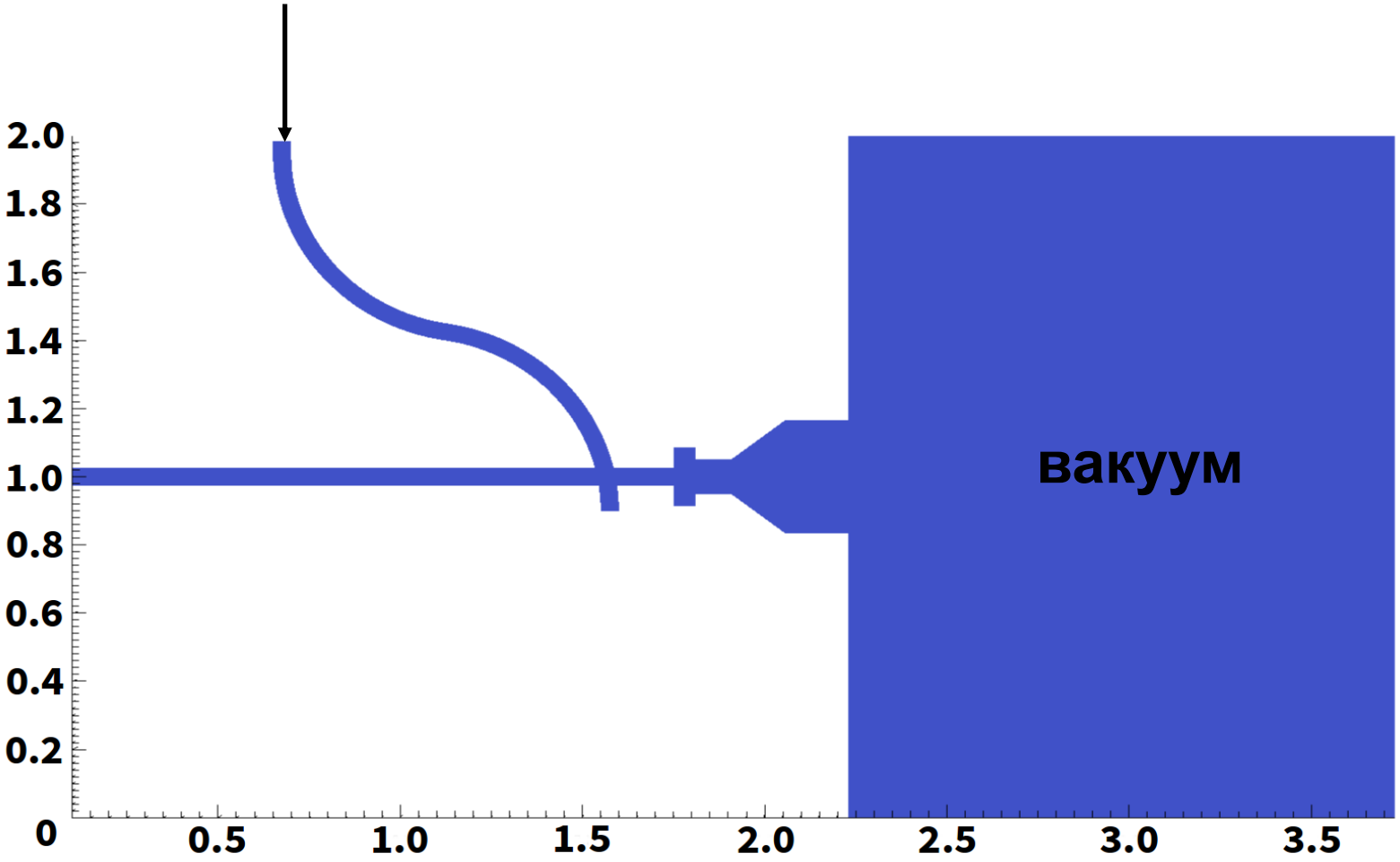
2 стадия: моделирование разряда



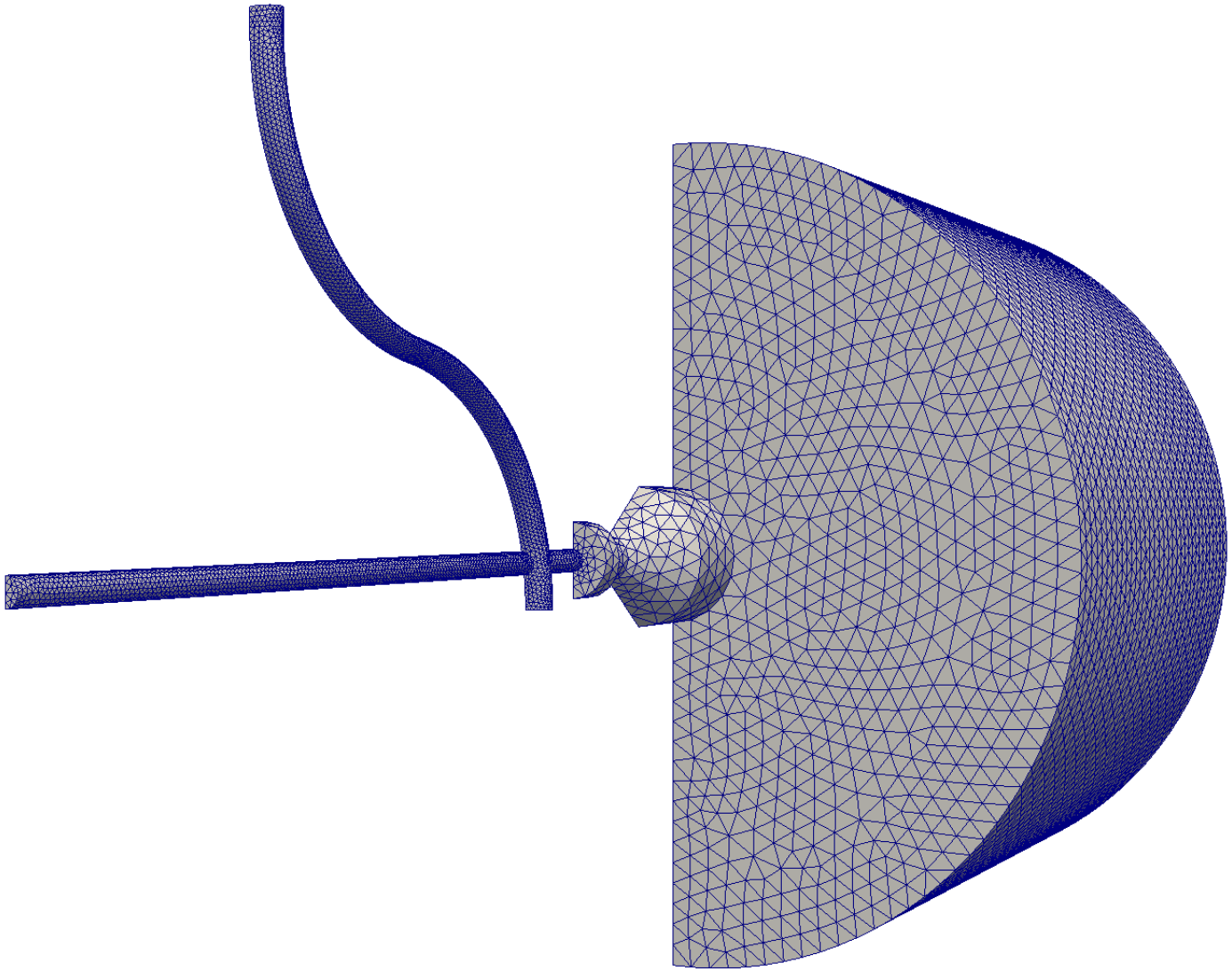
Заполнение капилляра

Гидродинамическое моделирование: $0 \le t \le 150$ мкс

водород, $\rho \approx 4.3 \cdot 10^{-6}$ г/см^3
(~ P = 40 Торр, T = 300 К)

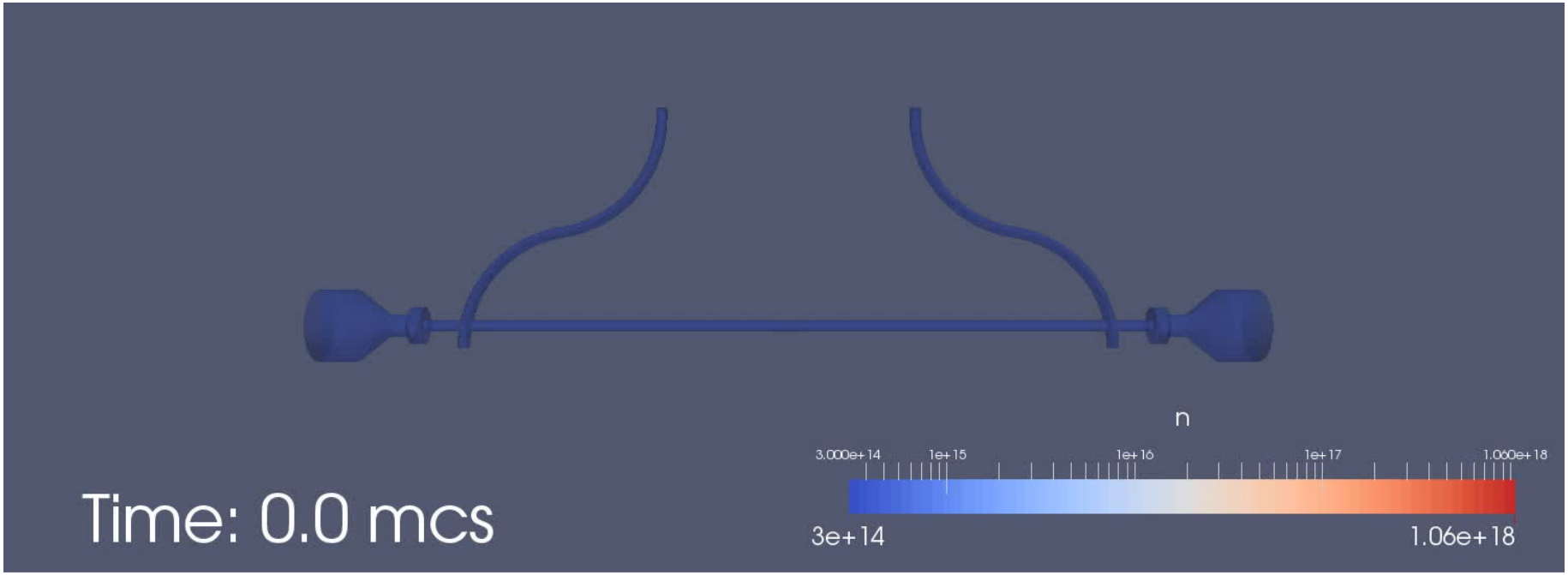


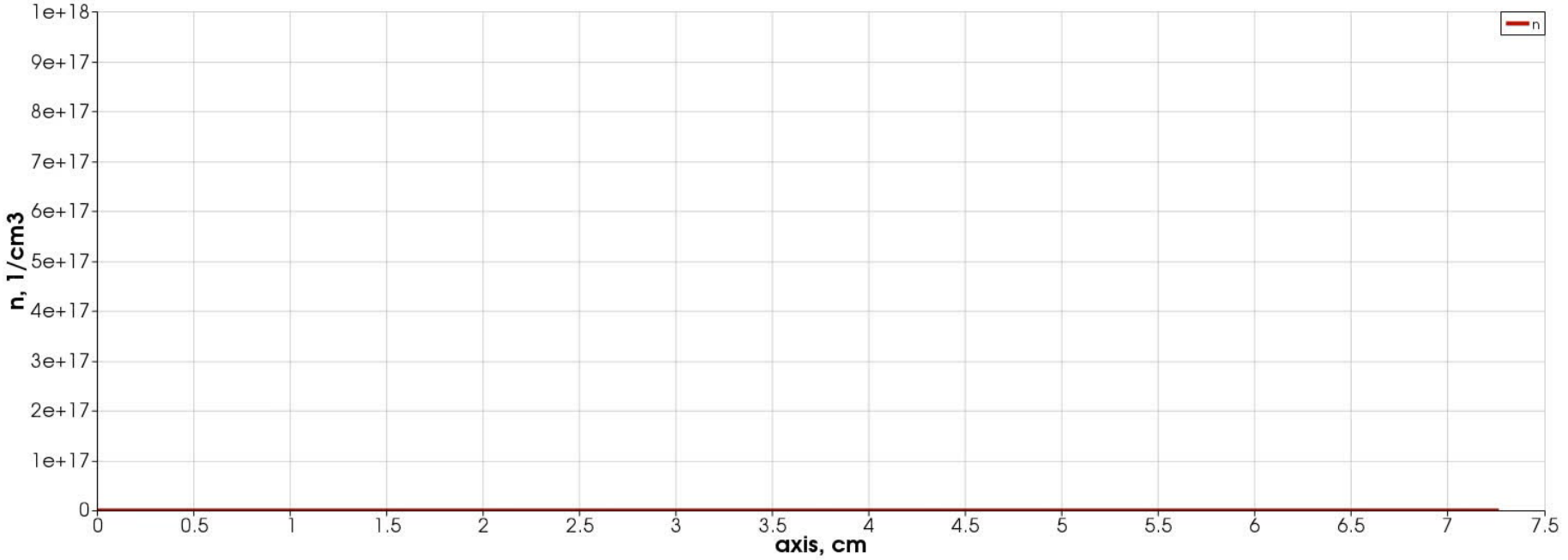
Заполнение капилляра



сетка: 120 тыс. тетраэдральных ячеек
расчет занял ~21 час на 12 ядрах Xeon x5670
(K100, ИПМ им. М. В. Келдыша РАН)

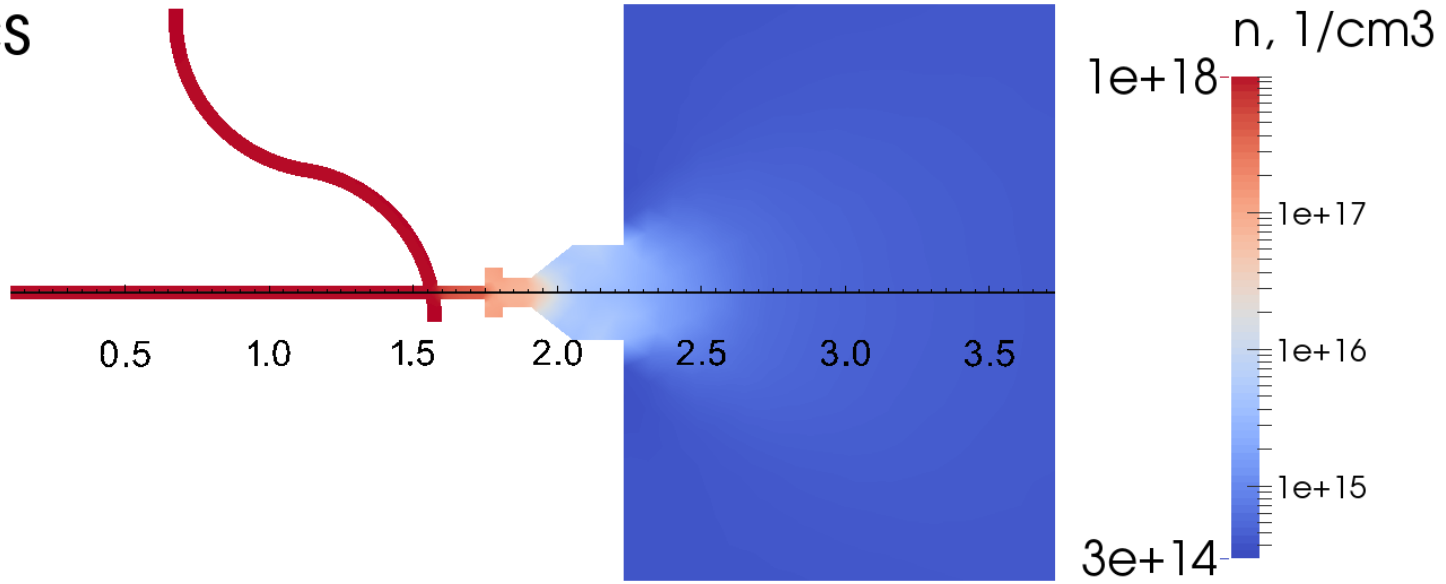
Заполнение капилляра



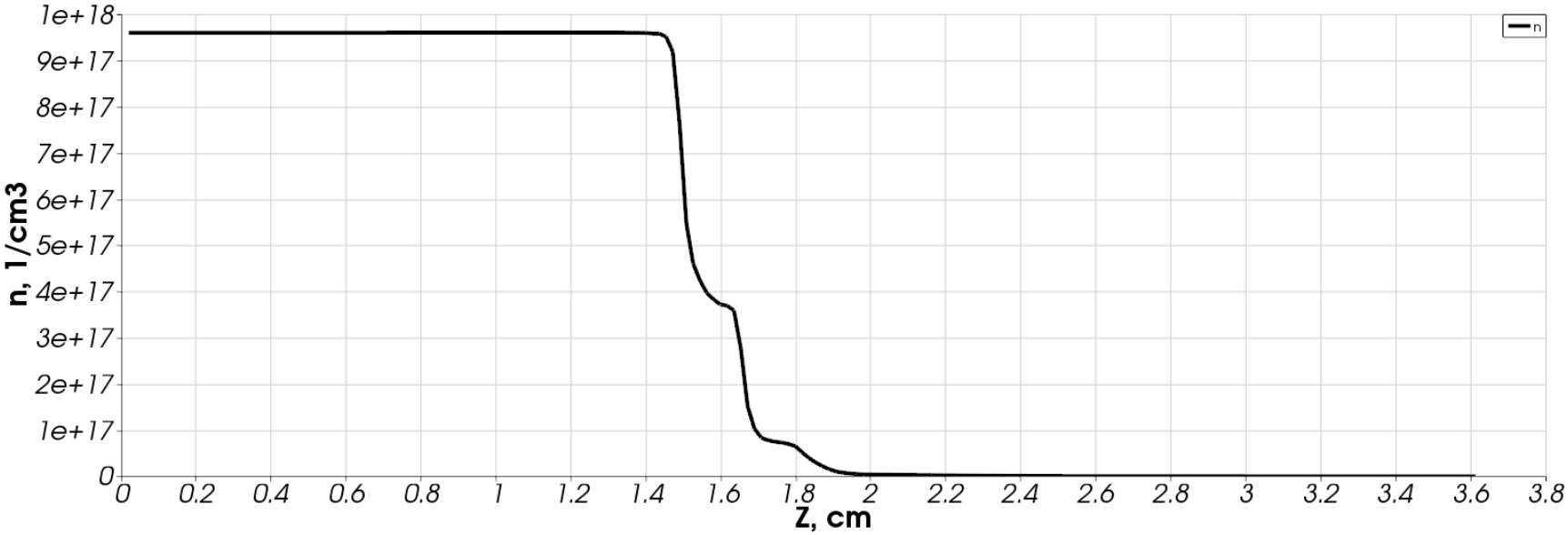


Заполнение капилляра

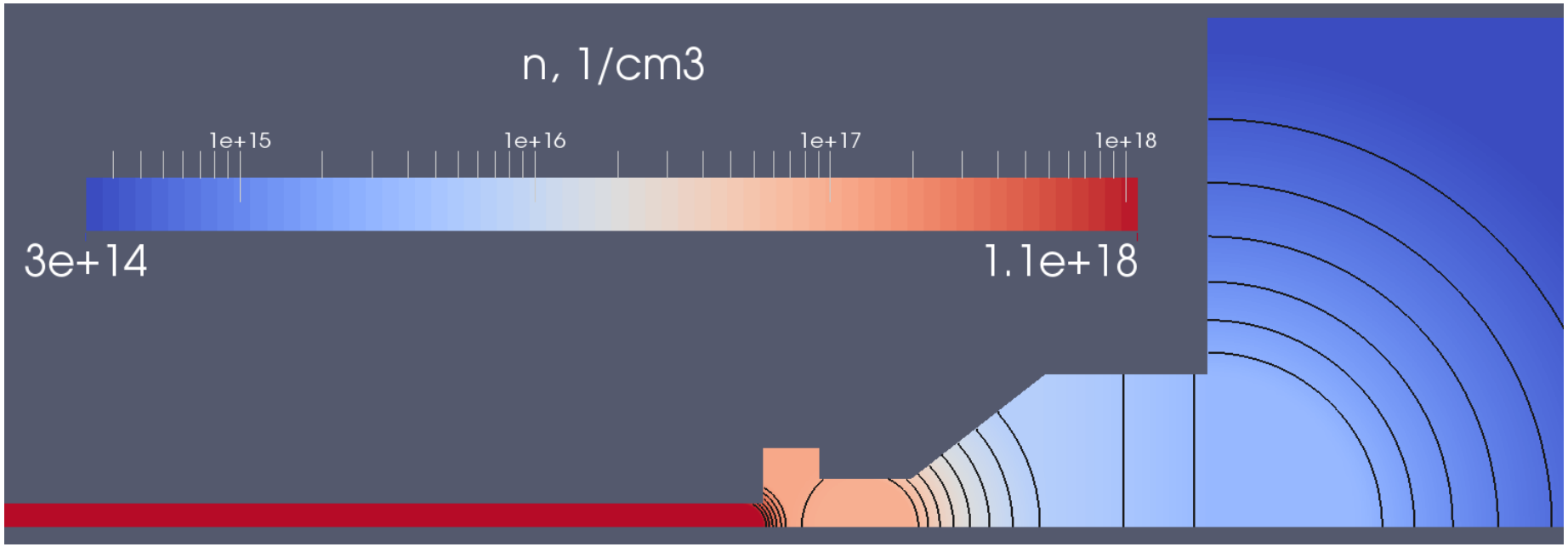
110 mcs



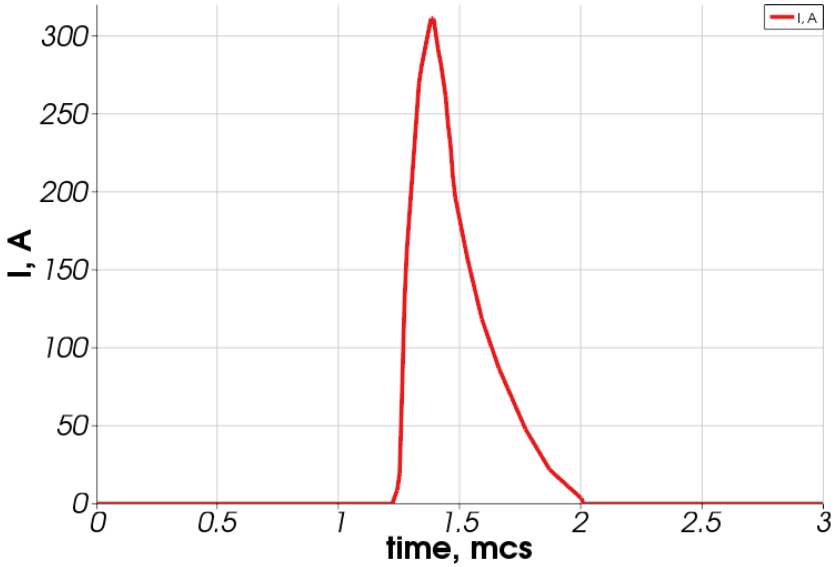
40 torr



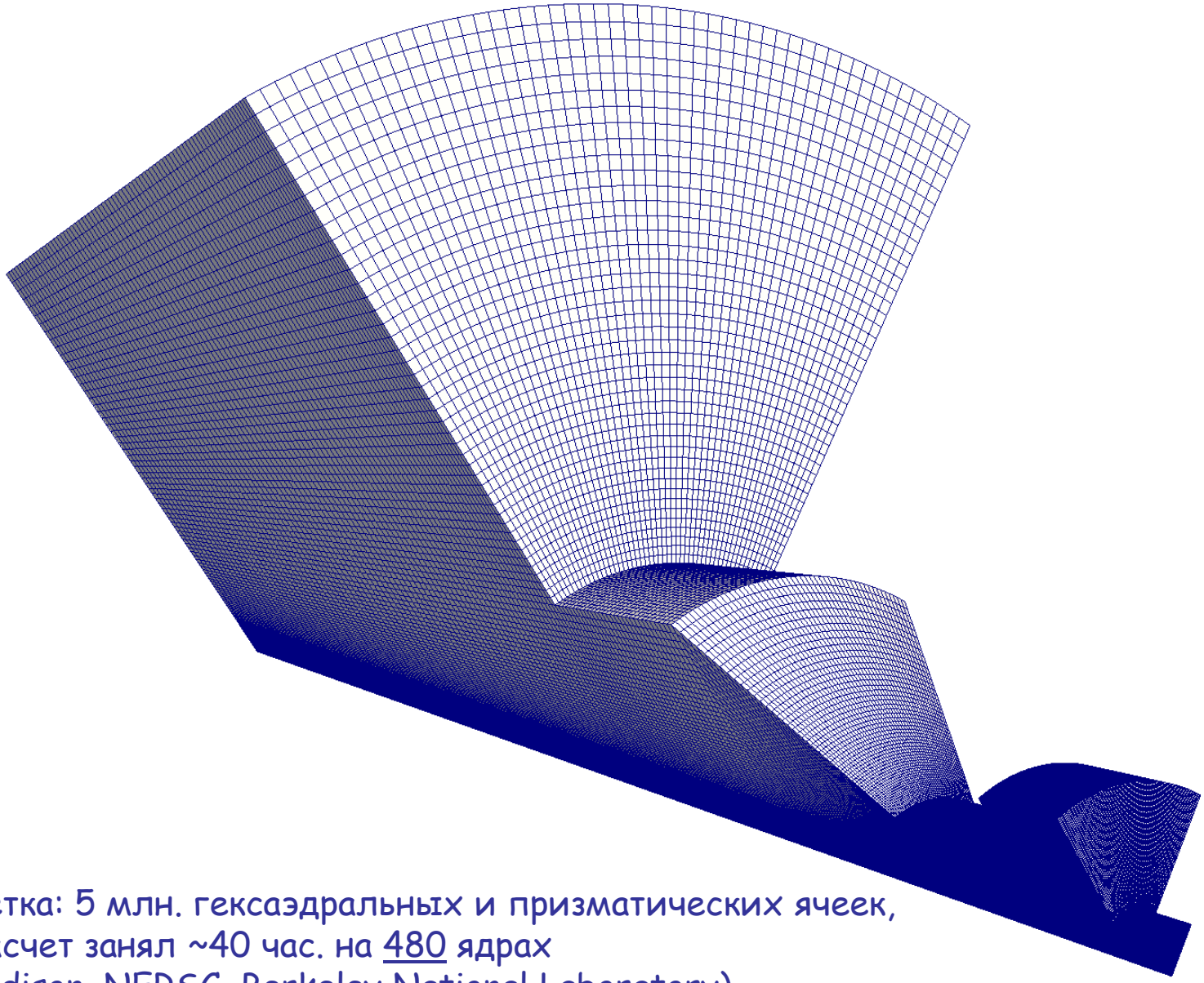
Капиллярный разряд



- МГД моделирование: $0 \le t \le 250$ нс
- начальные условия:
  - распределение плотности из расчета заполнения,
  - T_e = 0.5 эВ.
- граничные условия:
  - $I(t)$ – задано таблицей,
  - I_{max}(170 нс) = 311 А.

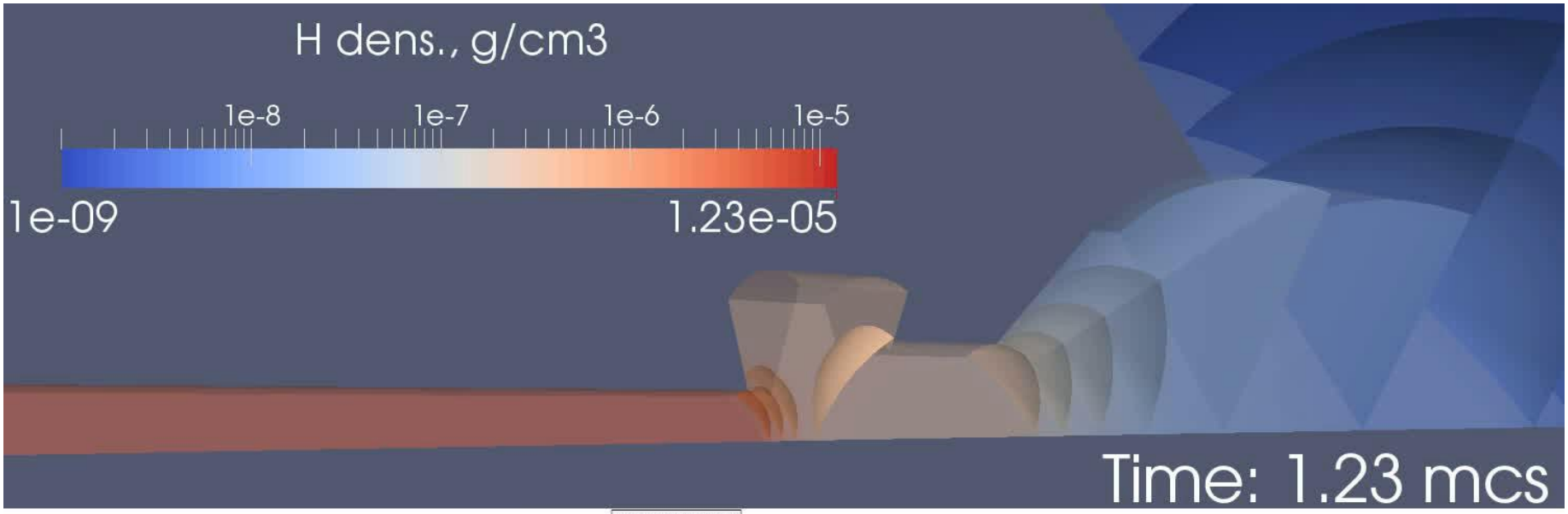


Капиллярный разряд

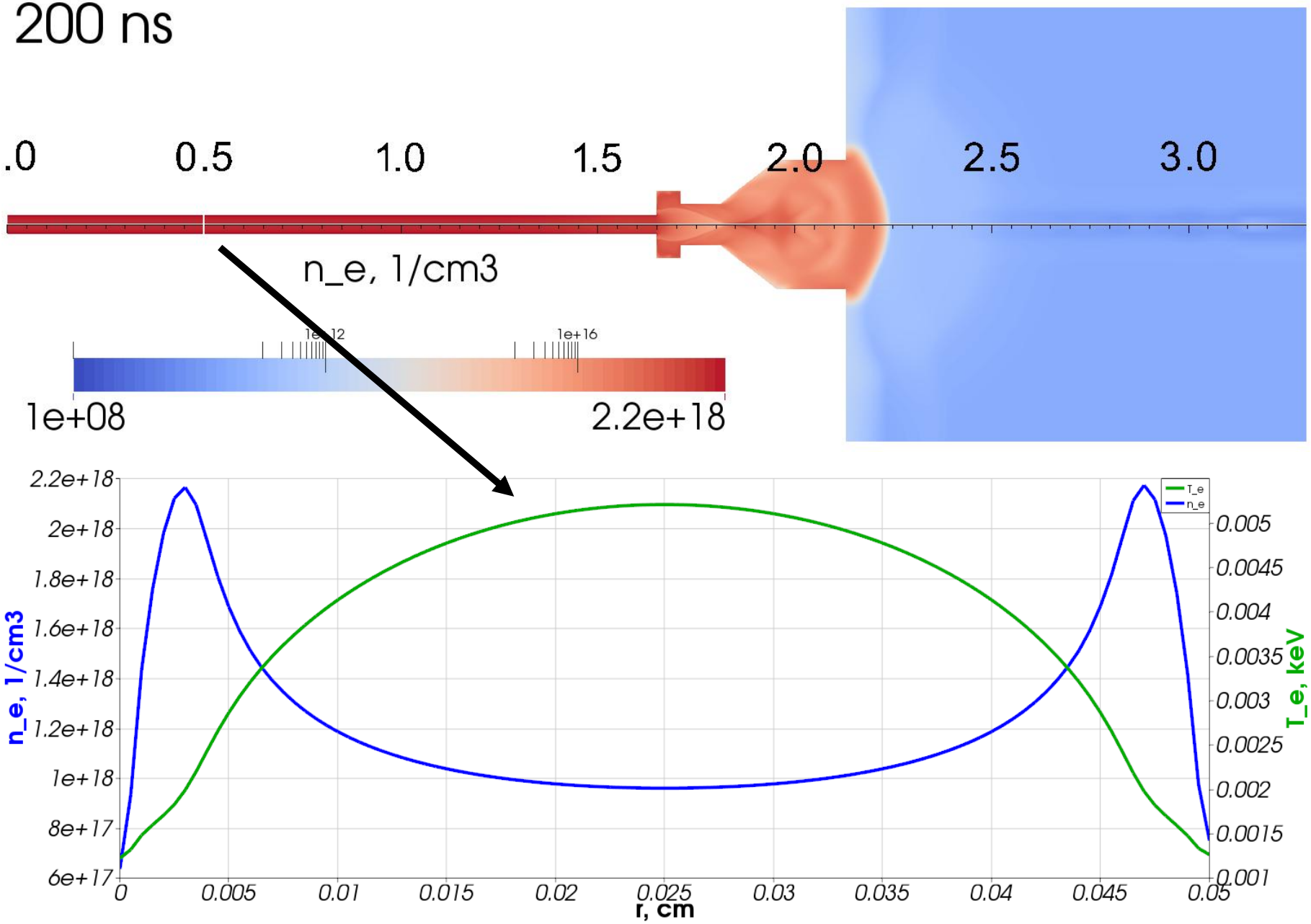


сетка: 5 млн. гексаэдральных и призматических ячеек,
расчет занял ~40 час. на 480 ядрах
(Edison, NERSC, Berkeley National Laboratory)

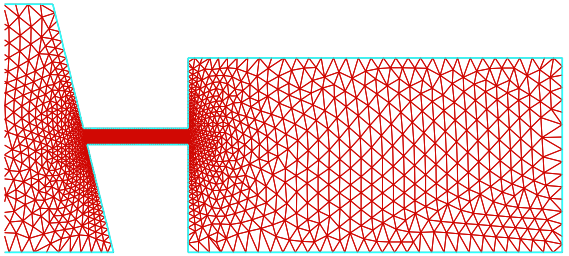
Капиллярный разряд

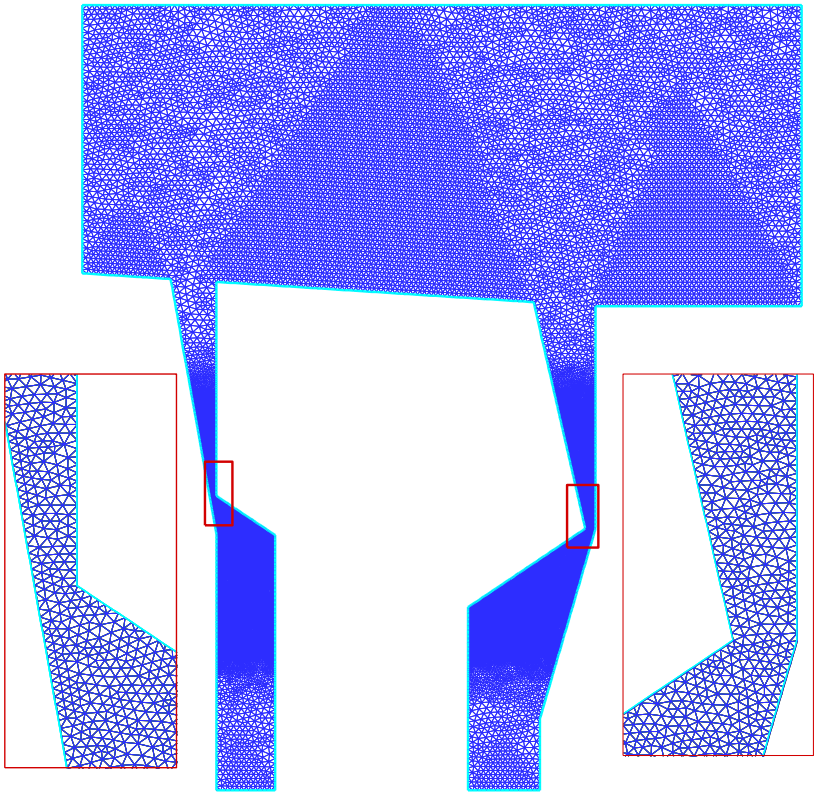


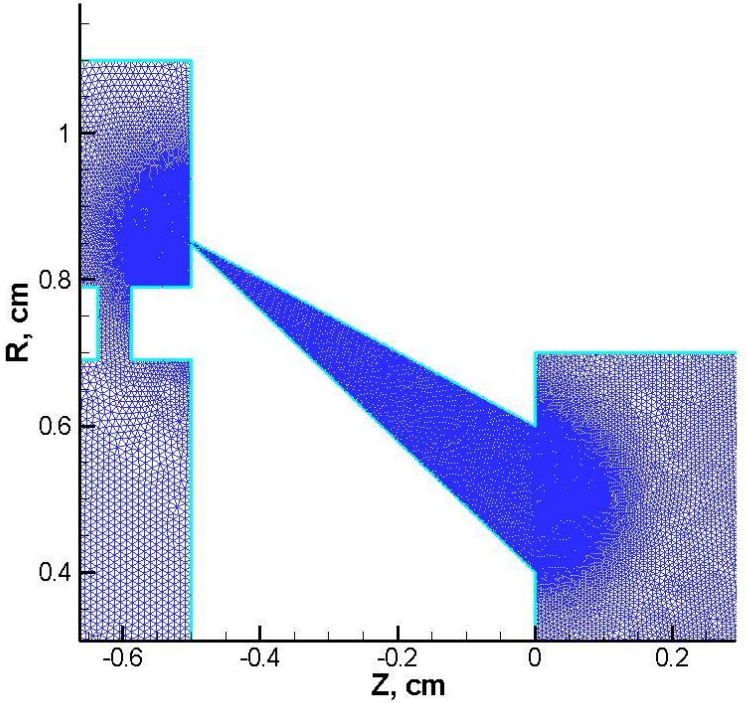
Капиллярный разряд



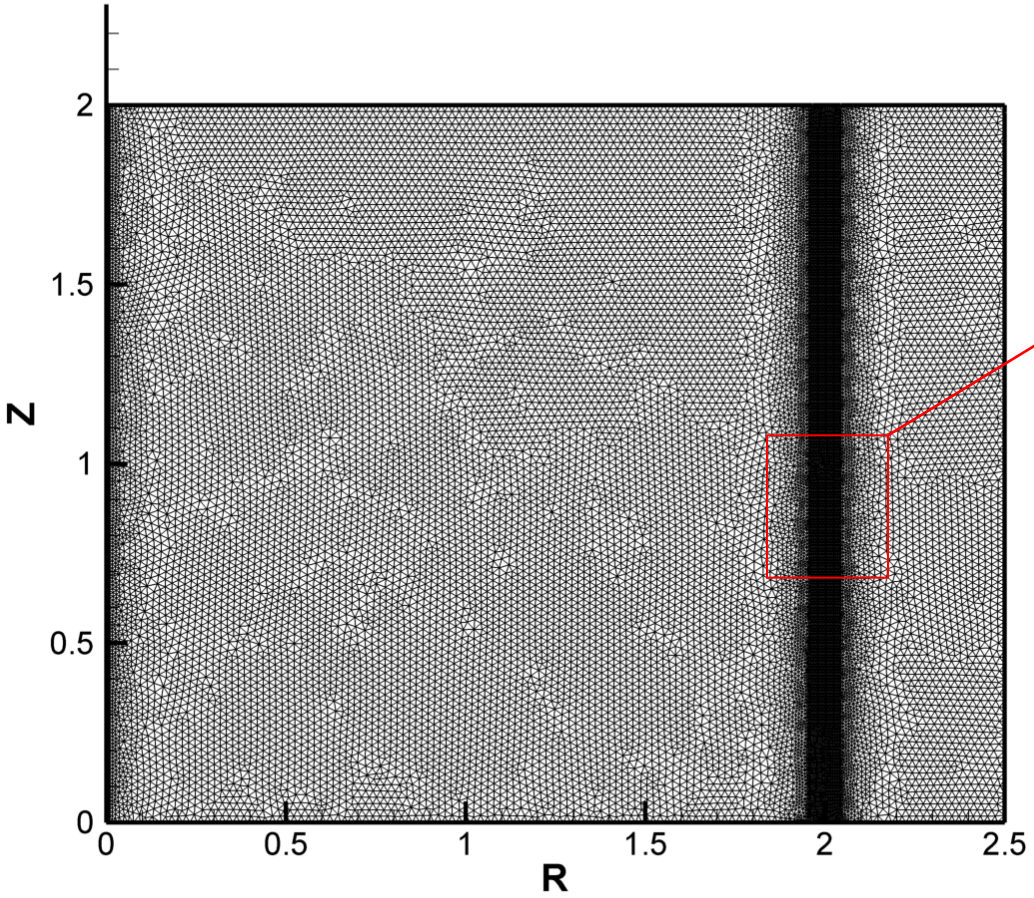
Преимущества неструктурированных сеток – адаптация к геометрии расчетной области

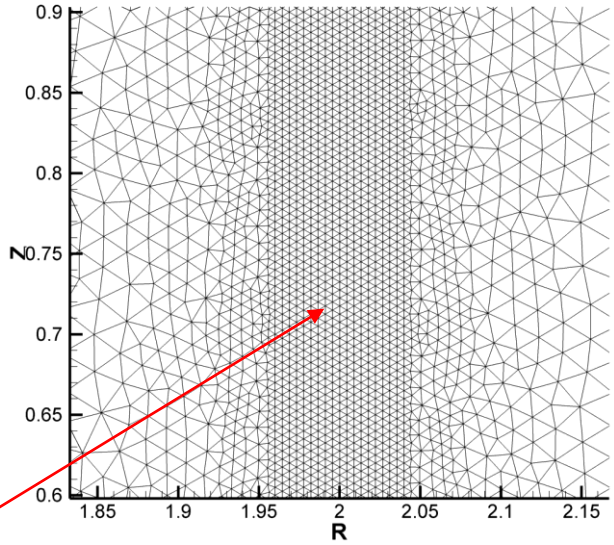




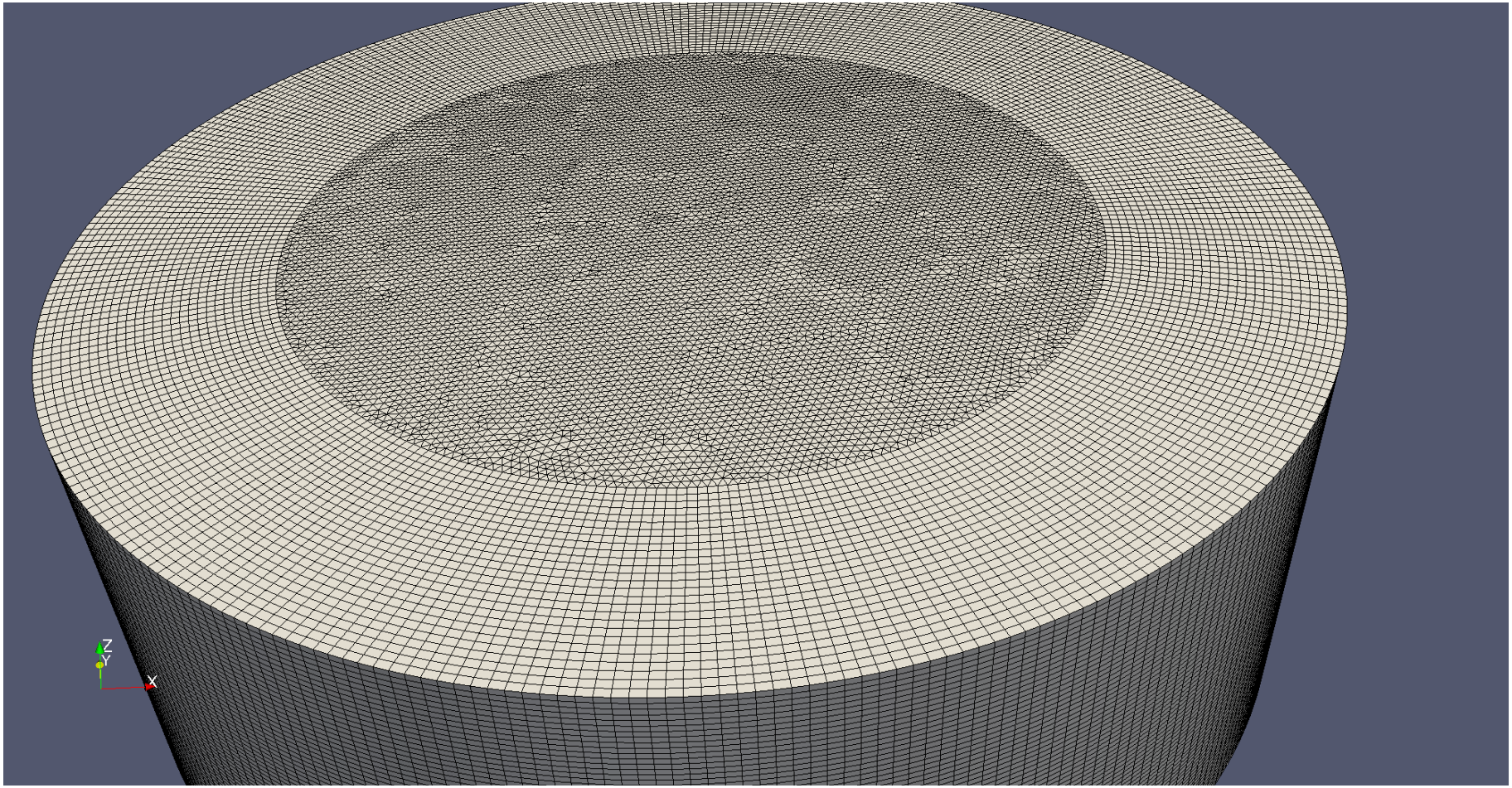


Преимущества неструктурированных сеток – адаптация к особенностям решения





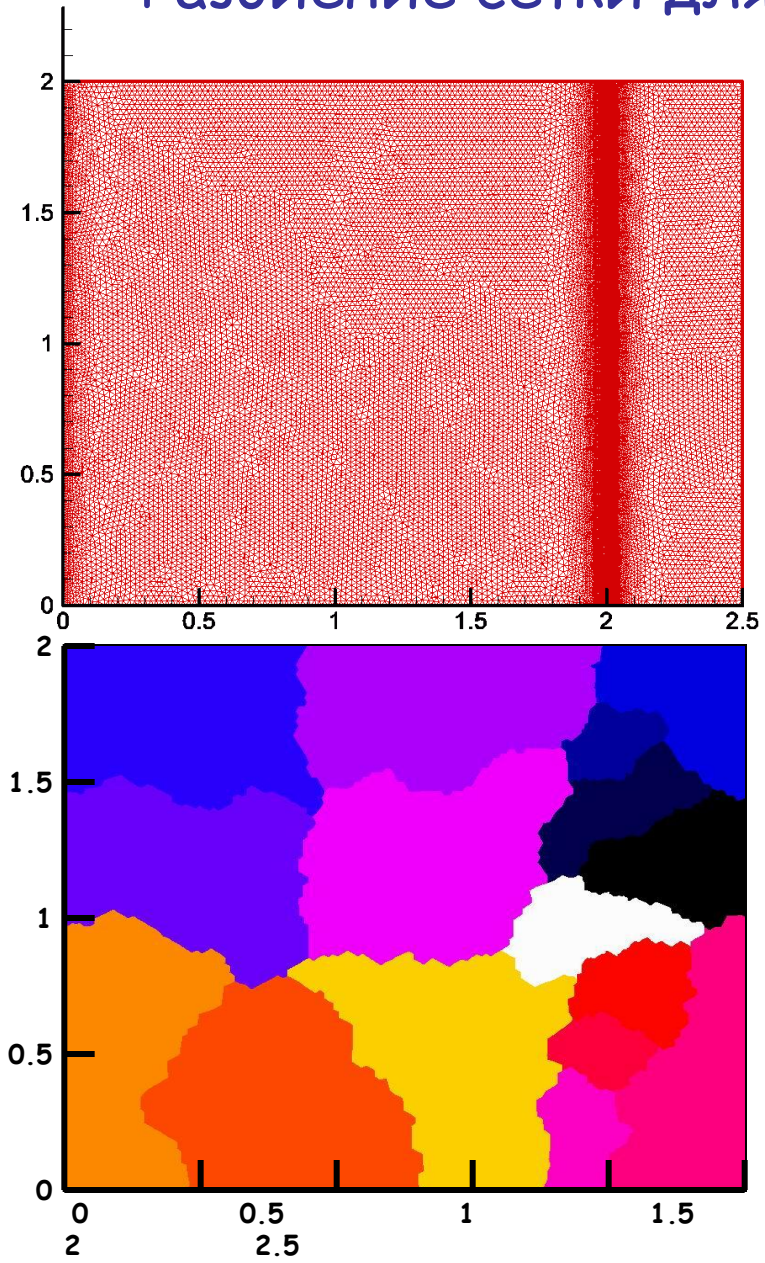
Пример комбинированной трехмерной сетки



Программные средства для обработки неструктурированных сеток

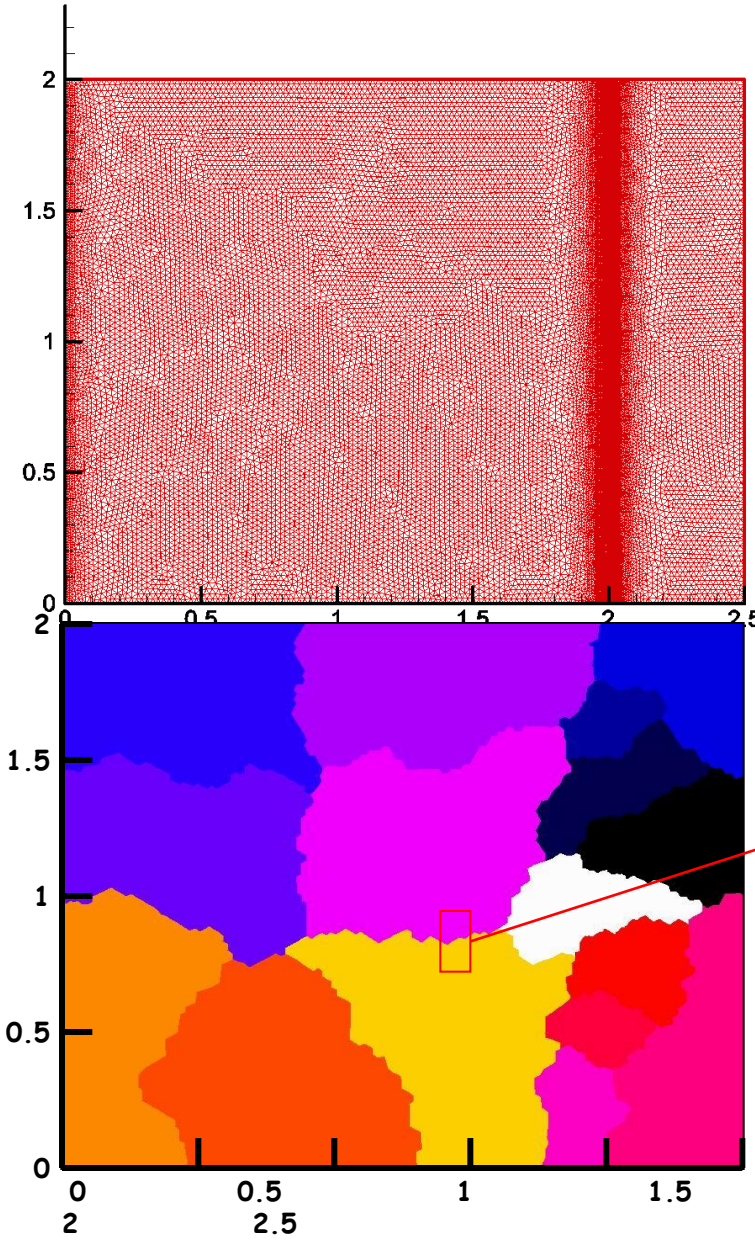
- Независимость от предметной области.
- Поддержка основных операций с сетками, необходимых для построения аппроксимаций и генерации сеток.
- Хранение физических величин в элементах сетки.
- Возможность параллельной (распределенной) обработки.
- Бескомпромиссно высокая производительность.

Разбиение сетки для параллельных вычислений



- Размеры сеток могут исключать их обработку последовательными кодами;
- Распределенные алгоритмы должны использоваться на всех стадиях решения задачи:
  - распределенная генерация сетки
  - разбиение и переразбиение сеток (ParMetis)
  - параллельное решение задачи
  - параллельный анализ результатов (ParaView)
- Распределенность сетки должна учитываться на алгоритмическом уровне.

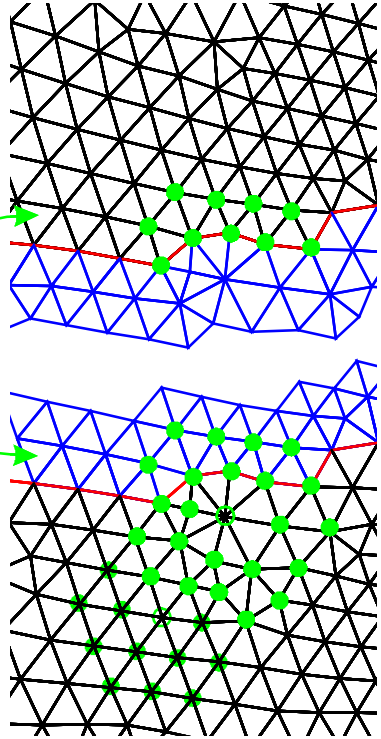
Разбиение сетки для параллельных вычислений



Регулярные сеточные элементы

"Фиктивные" сеточные элементы (поля)

Расчет на распределенной сетке



Хранение физических атрибутов сеточных элементов

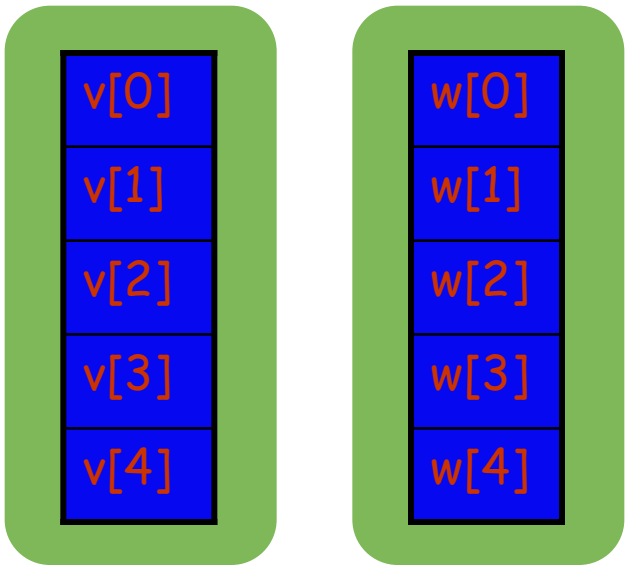
Традиционный подход

Набор пронумерованных объектов (например, сеточных элементов). Ему не соответствует никакой объект в программе.

Удалим второй элемент.

Теперь 4-й элемент становится 2-м, чтобы сохранить непрерывность нумерации.

Массивы атрибутов
std::vector



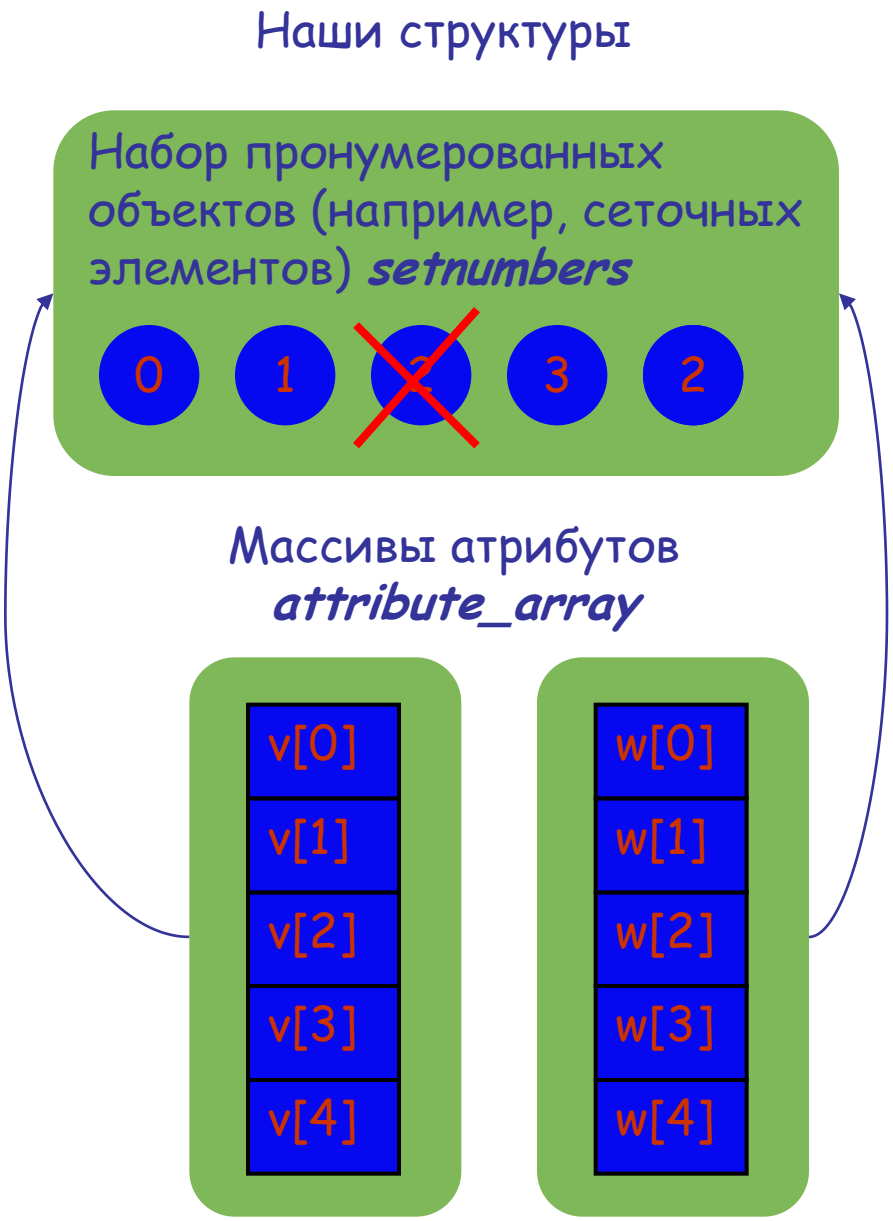
Черт побери! Соответствие между сеточными элементами и атрибутами нарушено.

Хранение физических атрибутов сеточных элементов

Удалим второй элемент.

Теперь 4-й элемент становится 2-м, чтобы сохранить непрерывность нумерации.

Теперь все в порядке. Соответствие между объектами и их атрибутами сохраняется.

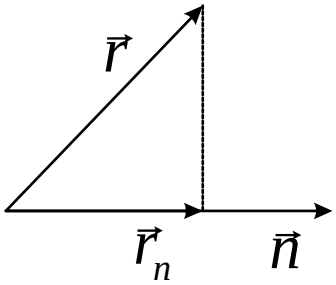


Представление сетки с помощью бинарных отношений

Использование C++

- Особенности архитектуры современных компьютеров (например, кэширование памяти, векторные операции и т. д.) приводят к необходимости возврата к низкоуровневому программированию в стиле Фортрана или ассемблера.
- Между тем, высокоуровневые языки программирования наподобие C++ обеспечивают возможности гораздо более легкой и удобной разработки универсальных и легко сопровождаемых кодов.

Использование C++ - типы данных



$$\vec{r}_n = \frac{(\vec{r} \cdot \vec{n})}{|\vec{n}|^2} \cdot \vec{n}$$

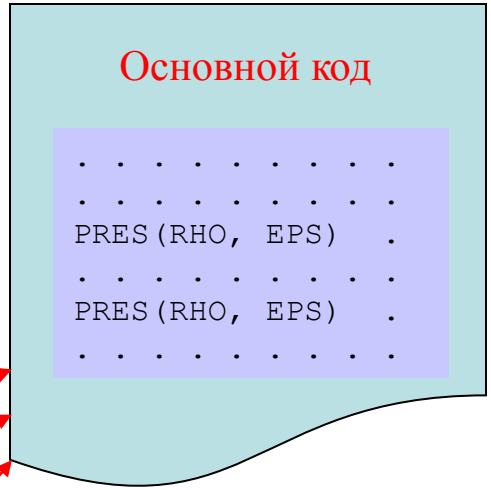
```
      ALNG = (R_X*AN_X + R_Y*AN_Y + R_Z*AN_Z) / (AN_X*AN_X + AN_Y*AN_Y + AN_Z*AN_Z)
      RN_X = ALNG * AN_X
      RN_Y = ALNG * AN_Y
      RN_Z = ALNG * AN_Z
```

```
rn = (r * n) / (n * n) * n;
```

Использование C++ – полиморфные типы. Пример – уравнение состояния

Варианты уравнения состояния $P(\rho,\varepsilon)$:

- Идеальный газ $P=(\gamma-1)\rho\varepsilon$;
- Ван-дер-Ваальса $P=\dfrac{(\gamma-1)(\varepsilon+a\rho)\rho}{1-b\rho}-a\rho^2$;
- Табличные уравнения;
- …

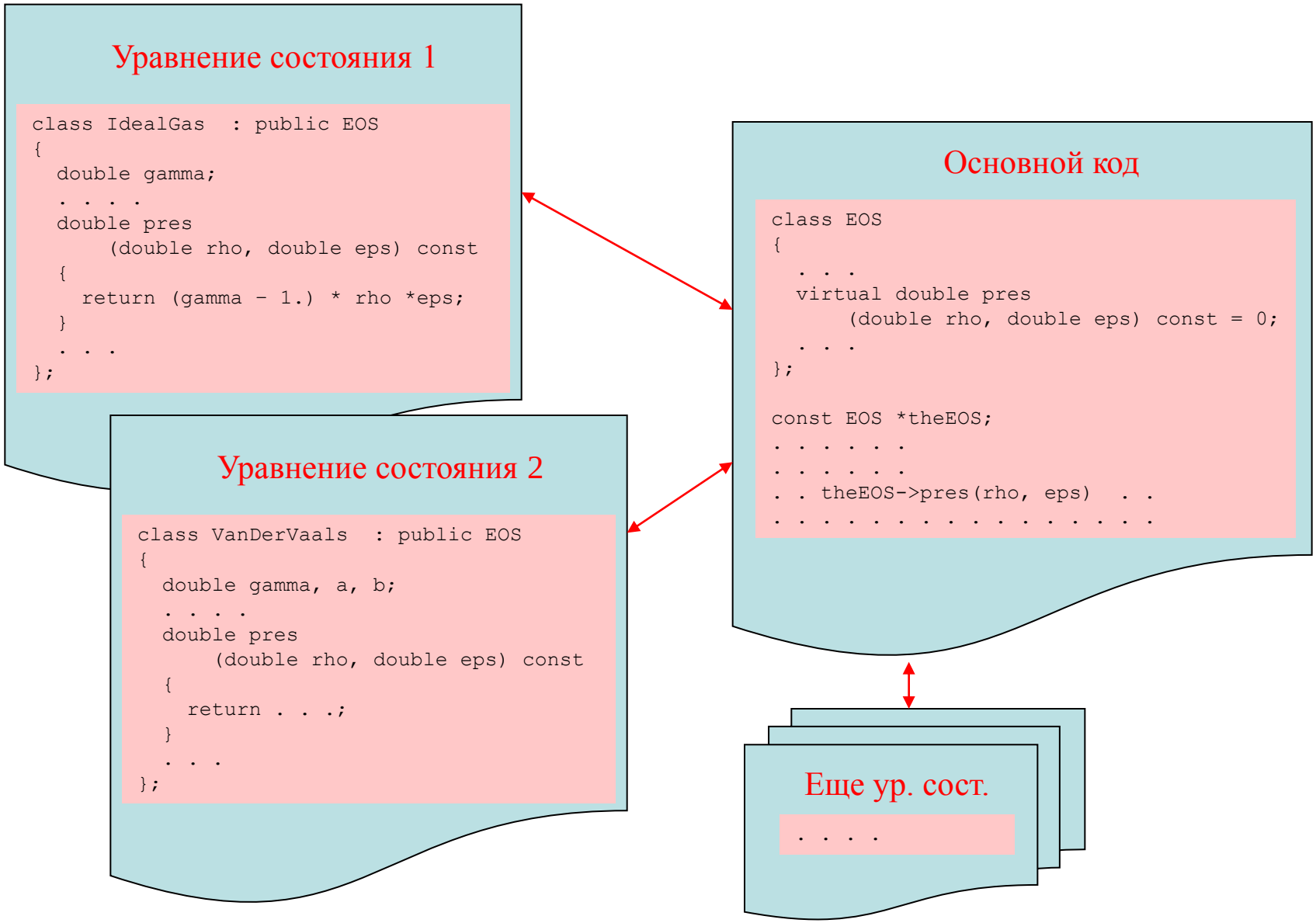



```
      FUNCTION PRESS(RHO, EPS)
      PRESS = (GAMMA - 1.) * RHO * EPS
```

```
      FUNCTION PRESS(RHO, EPS)
C Some other equation of state...
      . . .
      PRESS = . . .
      RETURN
      END
```

```
                                        + A * RHO)
      RETURN
      END
```

Использование C++ – полиморфные типы. Пример – уравнение состояния



Параметризация солверов. Пример – солвер гиперболической системы

Система гиперболических уравнений $\frac{\partial u}{\partial t} + \frac{\partial F(u)}{\partial x} = 0$.
Частные случаи:

- Уравнение переноса
- Газовая динамика
- Магнитогидродинамика
- То или иное может быть одно- или двухтемпературным
- Может быть одно- или многокомпонентным

Чтобы иметь возможность решать все эти задачи, нужно большое количество однотипных (но различных) солверов.
Хотя бы из-за того, что набор величин u различен для разных уравнений.

Параметризация солверов на примере гиперболической системы уравнений.

Уровень аппроксимации

Уравнения баланса

$$\frac{\hat{u}_i - u_i}{\tau} + \frac{1}{h}\left(F_{i+1/2} - F_{i-1/2}\right) = 0$$

```
template <class physicalLevel>
  void step(double tau, double h,
                const std::vector<physicalLevel::dataType> &u,
                std::vector<physicalLevel::dataType> &u1)
  {
     ...
     for (size_t i = ...)
       u1[i] = u[i] – tau/h * (physicalLevel::Flux(u[i], u[i+1]) - physi
  }
```

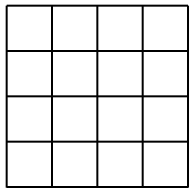
Физический уровень

- Набор хранящихся величин
- Метод вычисления потоков

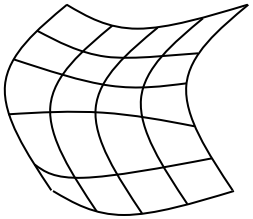
```
struct physicalLevelForSomething
{
   …
   struct dataType
   {
      // Values stored in the mesh elements.
      ...
   };

   static dataType Flux(const dataType &u1, const dataType &u2);
   …
};
```

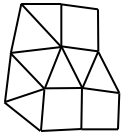
Дальнейшее развитие – типы сеток



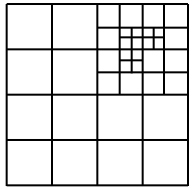
Прямоугольная



Топологически прямоугольная



Неструктурированная (конформная)



Восьмеричное дерево

Способы обеспечения общности:

- Реализуется наиболее сложный тип, остальные рассматриваются как его частные случаи.

Недостаток: реализация простых типов сеток неэффективна.

- Сетка рассматривается как чистая абстракция наподобие концепции итератора в C++ STL библиотеке.

Эффективная обработка различных типов сеток

Солвер для прямоугольных сеток

```
template <class physicalLevel, class meshType>
 void solve(..., const meshType &mesh, ...)
 {
    ...
    // Uses functions for (i, j, k) indexing to find an appropriate element.
    ...
 }
```

Солвер для несруктурированных конформных сеток

```
template <class physicalLevel, class meshType>
 void solve(..., const meshType &mesh, ...)
 {
    ...
    // Uses request functions to find neighboring elements.
    ...
 }
```

Каждый солвер может использовать такой тип сеток, который поддерживает необходимые для этого солвера сеточные операции. Неважно, какой тип имеет сетка на самом деле. Если сетка такого типа не поддерживает какие-либо из необходимых операций, солвер не скомпилируется корректно с такой сеткой.

Спасибо за внимание!