ВНИИА
РОСАТОМ

Платформа PyPHIA

Python parallel PHysics Integrated API

С.А. Дьячков, С.Ю. Григорьев, В.В. Жаховский, П.П. Захаров, А.А. Козырев, И.С. Меньшов, Р.В. Муратов, С.А. Мурзов, А.Н. Нимаков, А.Н. Паршиков, Г.Д. Рублев, А.А. Сережкин

ФГУП «ВНИИА» им. Н.Л. Духова

12-й профессиональный слёт разработчиков отечественных CFD-кодов

CFD WEEKEND-2025

Предназначена для моделирования комплексных физических процессов

ВНИИА
РОСАТОМ

МЕХАНИКА СПЛОШНЫХ СРЕД

- Кумулятивные явления и пробитие
- Ударные и детонационные волны
- Неустойчивые течения
- Взаимодействие лазера с веществом
- Гетерогенные среды
- Вязкость и капиллярные явления

ФИЗИКА ПЛАЗМЫ

- Лазерный термоядерный синтез
- Взаимодействие лазера с плазмой
- Плазменные потоки в атмосфере
- Горение в газовых потоках
- Астрофизика, магнитосфера планет Плазменные источники излучений

ФИЗИКА ЭЛЕКТРОМАГНИТНЫХ ВОЛН

- Распространение в неоднородных средах
- Взаимодействие с объектами
- Распространение в атмосфере
- Детектирование

ЯДЕРНАЯ И РАДИАЦИОННАЯ **ФИЗИКА**.

- Транспорт частиц и излучения в сложной геометрии
- Транспорт частиц и излучения
- Радиационное воздействие на вещество
- Радиационная стойкость приборов

СВОЙСТВА МАТЕРИАЛОВ

- Уравнения состояния
- Прочность и пластичность
- Разрушение и откол
- Перенос тепла, излучения
- Диффузия и вязкость
- Химическая и плазменная кинетика

ЭЛЕКТРОНИКА

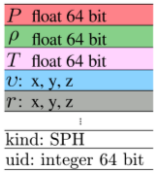
- Газовые разряды
- Источник Пеннинга
- Эффект Холла
- Электровакуумные источники нейтронов (нейтронные трубки)
- Полупроводники

Платформа PyPHIA | Python parallel PHysics Integrated API

ВНИИА РОСАТОМ

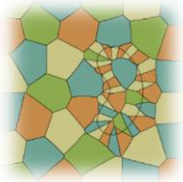
Data

Единое представление данных для всех модулей



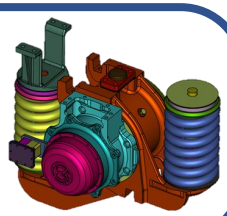
Decomposition

Распределение данных между вычислительными узлами при параллельных расчетах



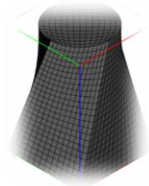
Geometry

Интерактивное и скриптовое построение геометрических моделей



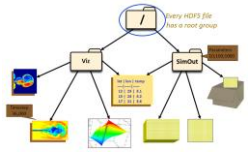
Mesh

Построение расчетных сеток для геометрических моделей и вспомогательные алгоритмы



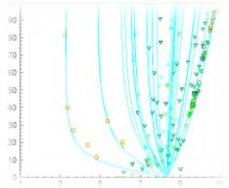
Input/Output

Ввод/вывод данных в различных форматах



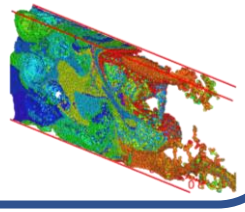
Matter

Библиотека моделей материалов



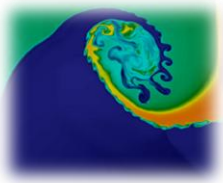
Render

Визуализация результатов расчета в интерактивном и скриптовом режиме



Solver

Библиотека решателей: FVM, FEM, SPH, MD, TPT, DDM, PIC, EM и др.



pyphia.data | Модуль представления данных

ВНИИА
РОСАТОМ

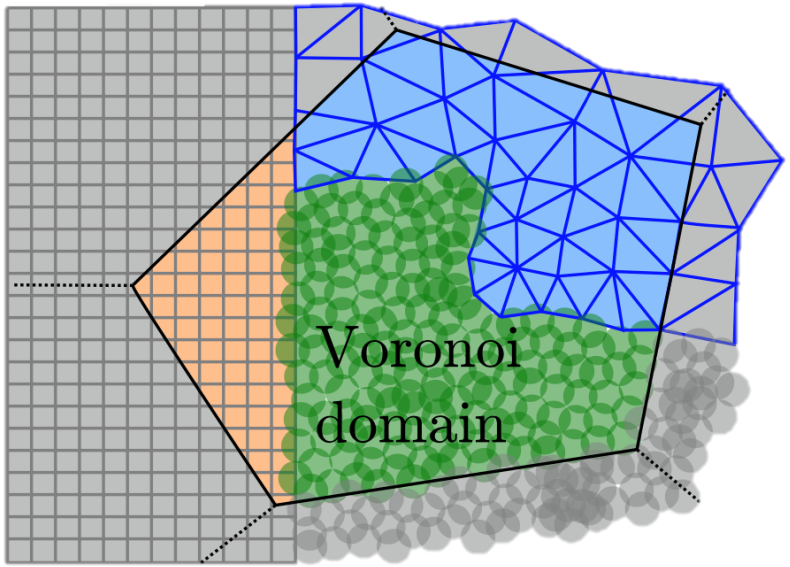
```
data::Storage storage(
{ // список полей из имеющихся
        "coords",
        "velocity",
        "size"
    },
    100 // число элементов массива
);
for (int i = 0; i < 100; ++i) {
storage[i] [coords].x    = 5;
storage[i] [velocity].x  = 5;
storage[i] [size].value  = 5;
}
```

```
from pyphia.data.types import *
from pyphia.data import Storage

storage = Storage(
    np.dtype([
        coords,
        velocity,
        size
    ]), 10
)
storage.data["coords"] ["x"]    = 5.0
storage.data["velocity"] ["x"] = 5.0
storage.data["size"] ["value"] = 5.0
```

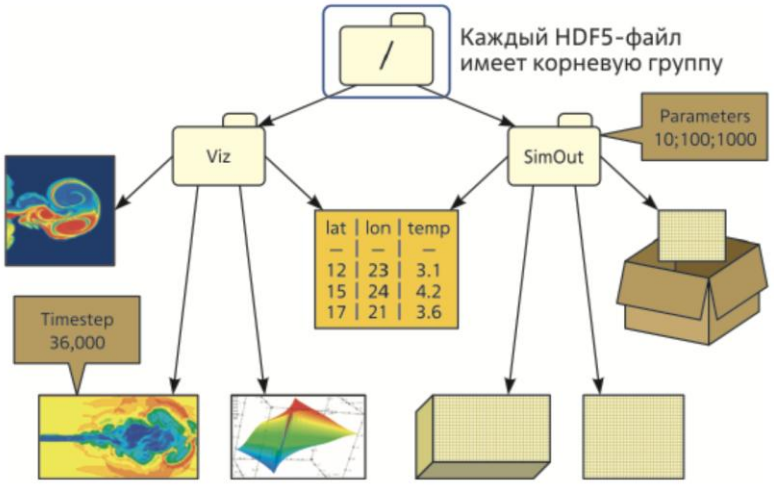
Storage – хранение массива структур с динамическим набором полей

Обеспечивает универсальное представление данных, совместимость с Python, параллелизацию и сопряжение численных методов

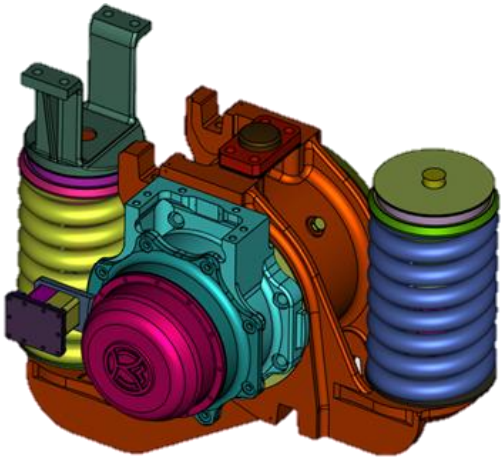


Интерфейс языка **Python** и библиотеки **NumPy** позволяет использовать широкий набор средств для ввода и вывода данных.
Пользователь может ограничиться имеющимися в открытом доступе библиотеками для ввода и вывода данных либо воспользоваться некоторыми форматами, реализованными в рамках комплекса PyPHIA.

HDF (Hierarchical Data Format)

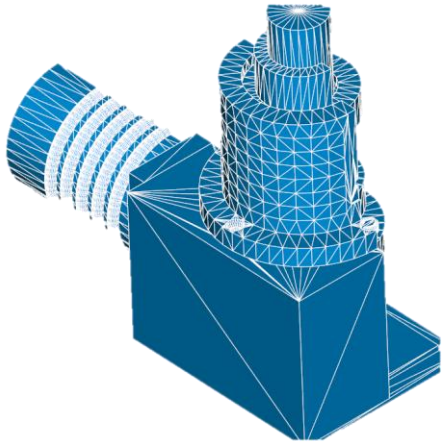


https://www.hdfgroup.org



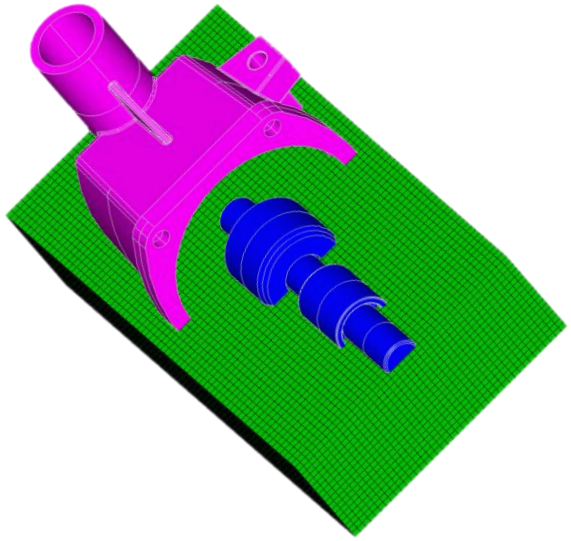
1. Геометрическая модель

Перед выполнением численного моделирования построение геометрии элементов конструкции в расчетной области с помощью интерактивного CAD-редактора.



2. Триангуляция

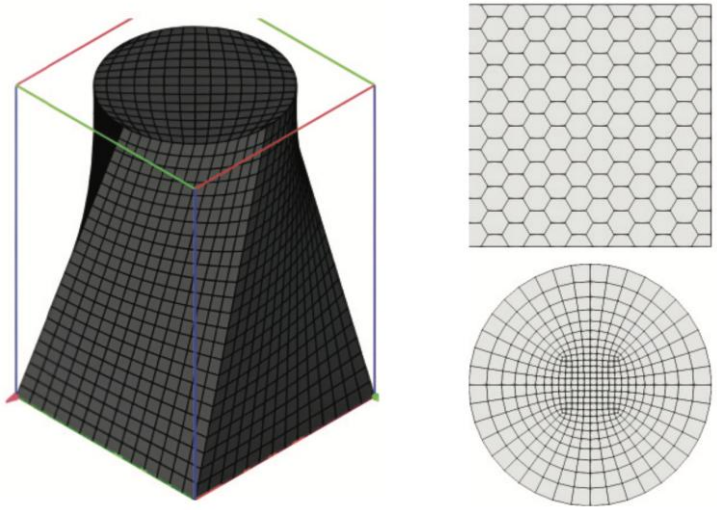
После задания геометрических моделей выполняется триангуляция поверхностей. Полученная полигональная модель используется для вычисления объемной доли пересечения с расчетной сеткой, которая строится на следующем этапе.



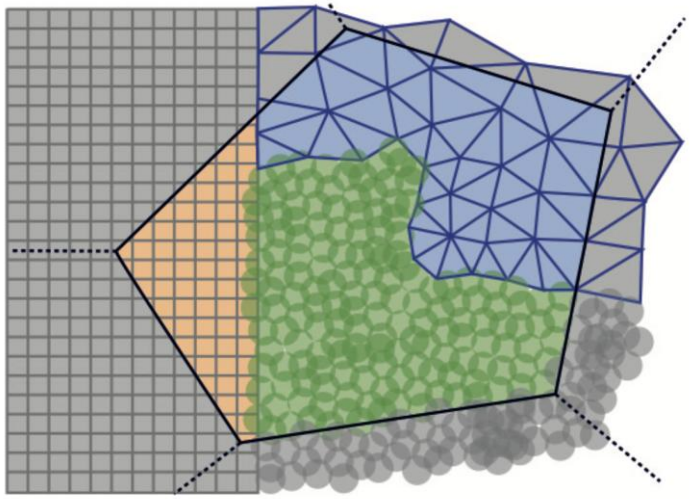
3. Построение сетки

Полигональная модель проецируется на сетку, так что каждой ячейке приписываются идентификаторы, соответствующие геометрической модели, что позволяет задать корректные начальные данные на сетке перед началом расчета.

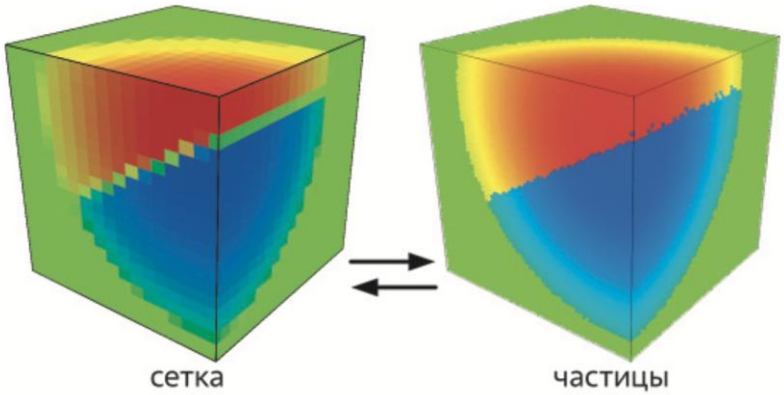
generator



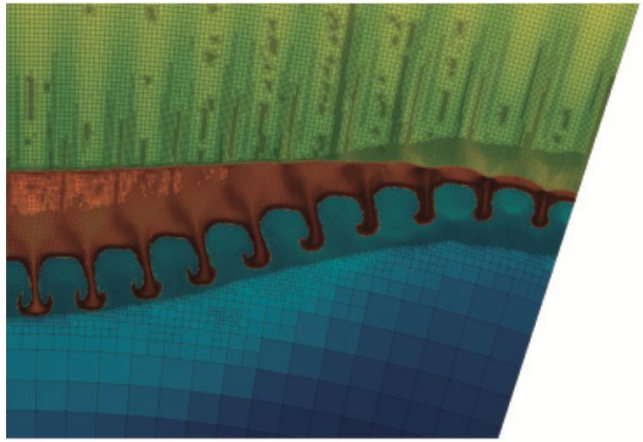
coupling



convert



refiner

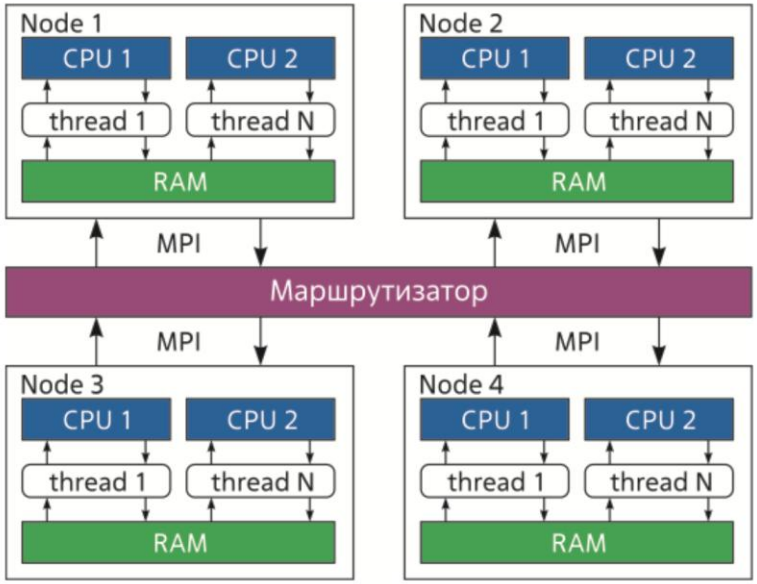


pyphia.decomposition

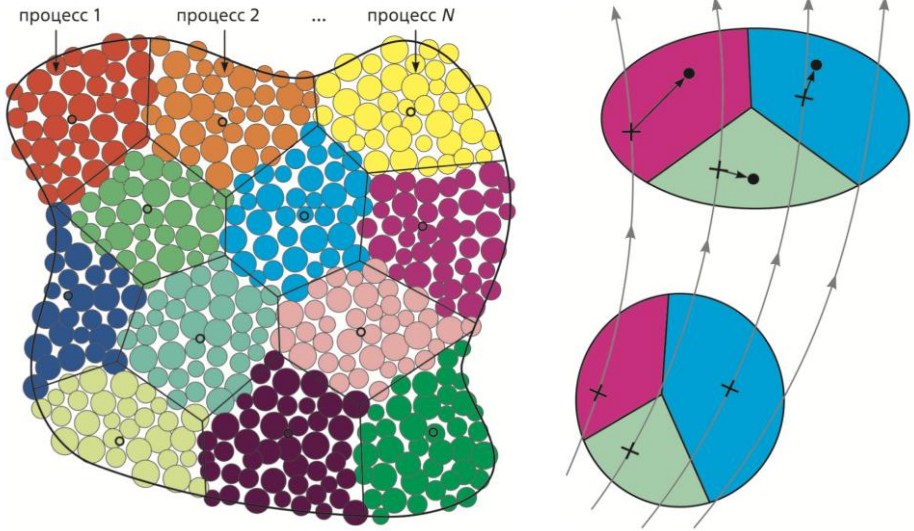
Модуль распределённых вычислений

ВНИИА РОСАТОМ

Гибридная параллелизация

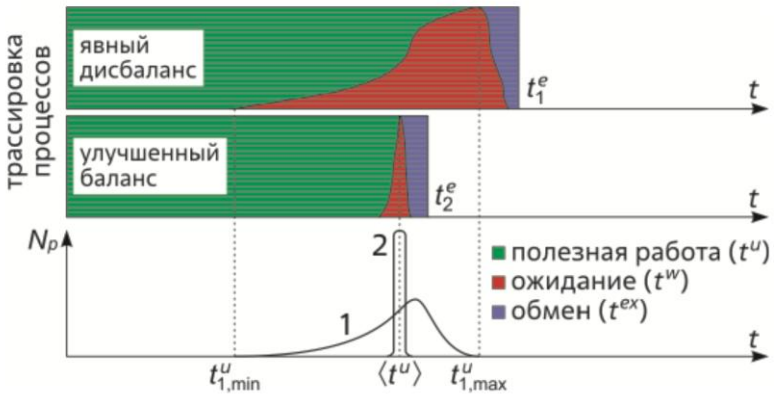


Декомпозиция по диаграмме Вороного

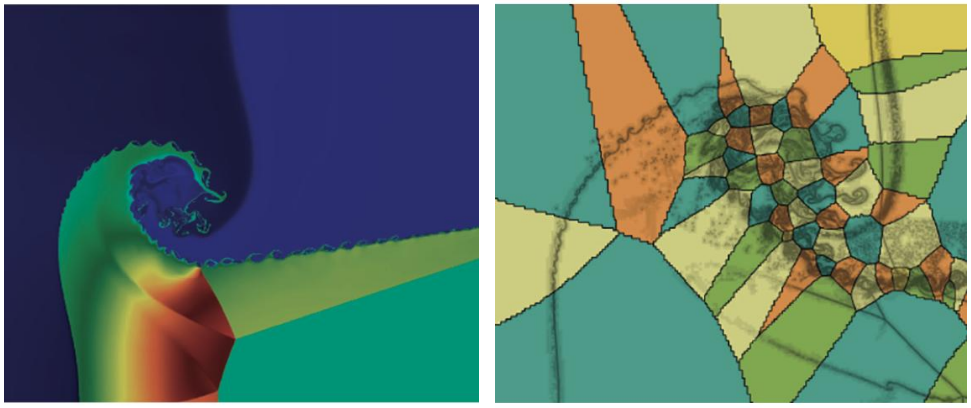


M.S. Egorova et al. Comput. Phys. Commun. (2019) DOI: 10.1016/j.cpc.2018.07.019

Балансировка нагрузки

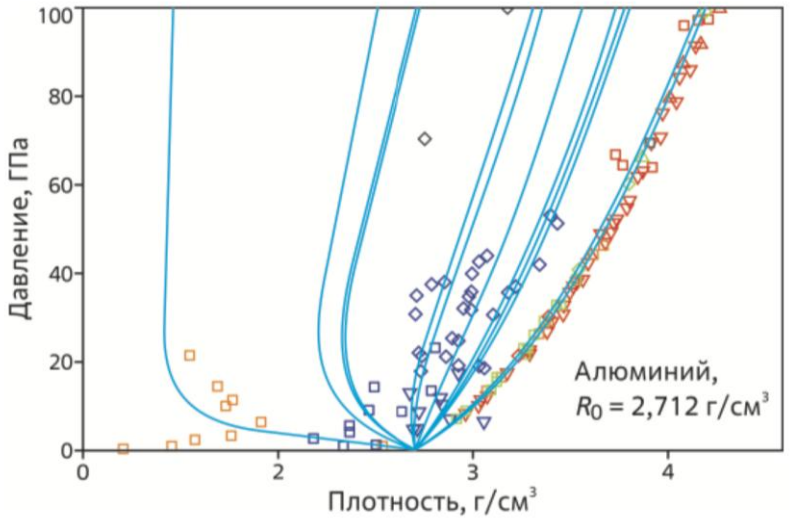


Применение к адаптивным сеткам

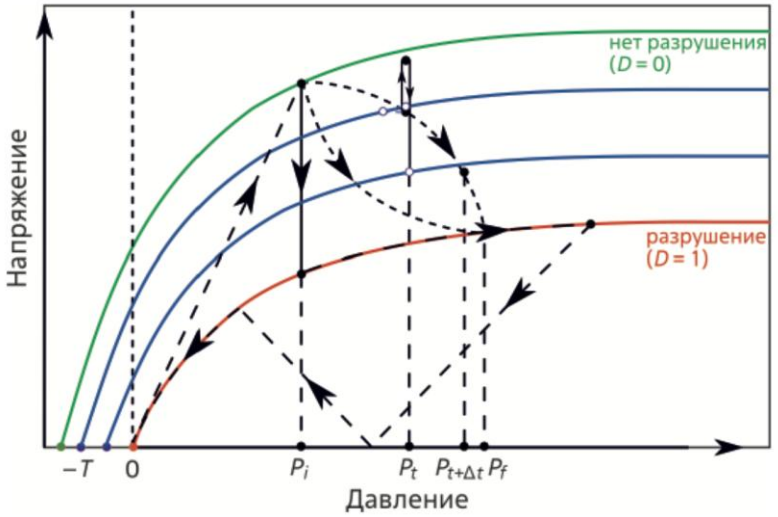


R.V. Muratov et al. Comput. Phys. Commun. (2023) DOI: 10.1016/j.cpc.2023.108790

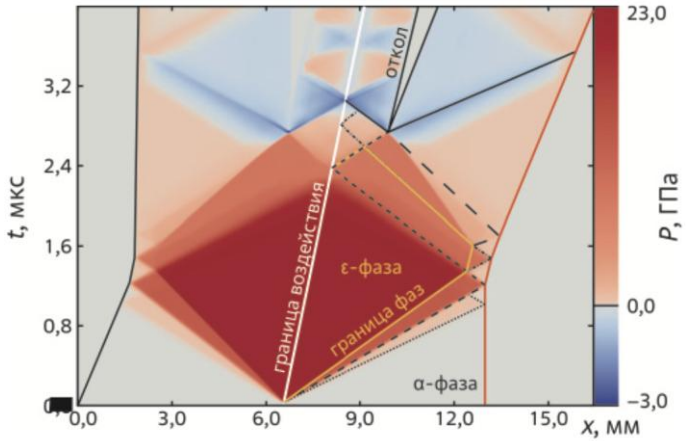
Уравнения состояния



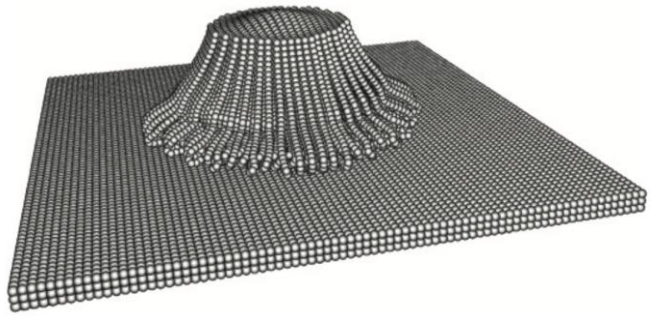
Прочность и упругопластика



Разрушение и откол

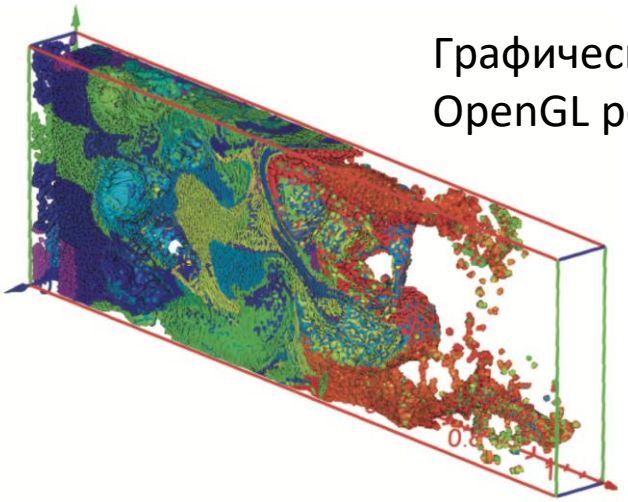


Вязкость и капиллярность



pyphia.render | Модуль интерактивной визуализации

ВНИИА РОСАТОМ

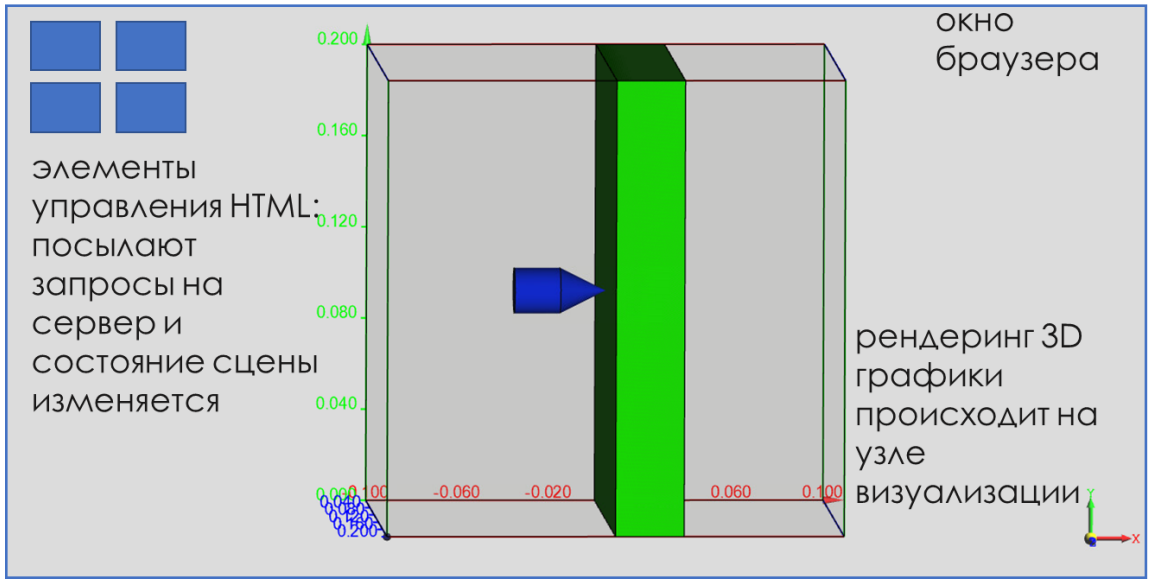


Графическое ядро:
OpenGL рендеринг больших объемов данных

Возможности ядра:

- Выбор цветовой палитры;
- Взятие сечений;
- Пороговая фильтрация;
- Построение распределений;
- Построение изоповерхностей;
- Построение 1D и 2D графиков;
- и другие.

Клиент-серверный механизм удаленного рендеринга



- Видеоизображение передается с сервера в браузер
- Элементы управления передают API-запросы по HTTP
- Сервер графики работает как в скриптовом режиме, так и через веб-интерфейс

pyphia.solver.FVM | Решатель метода конечного объёма

ВНИИА РОСАТОМ

Модуль обеспечивает решение системы гиперболических уравнений:

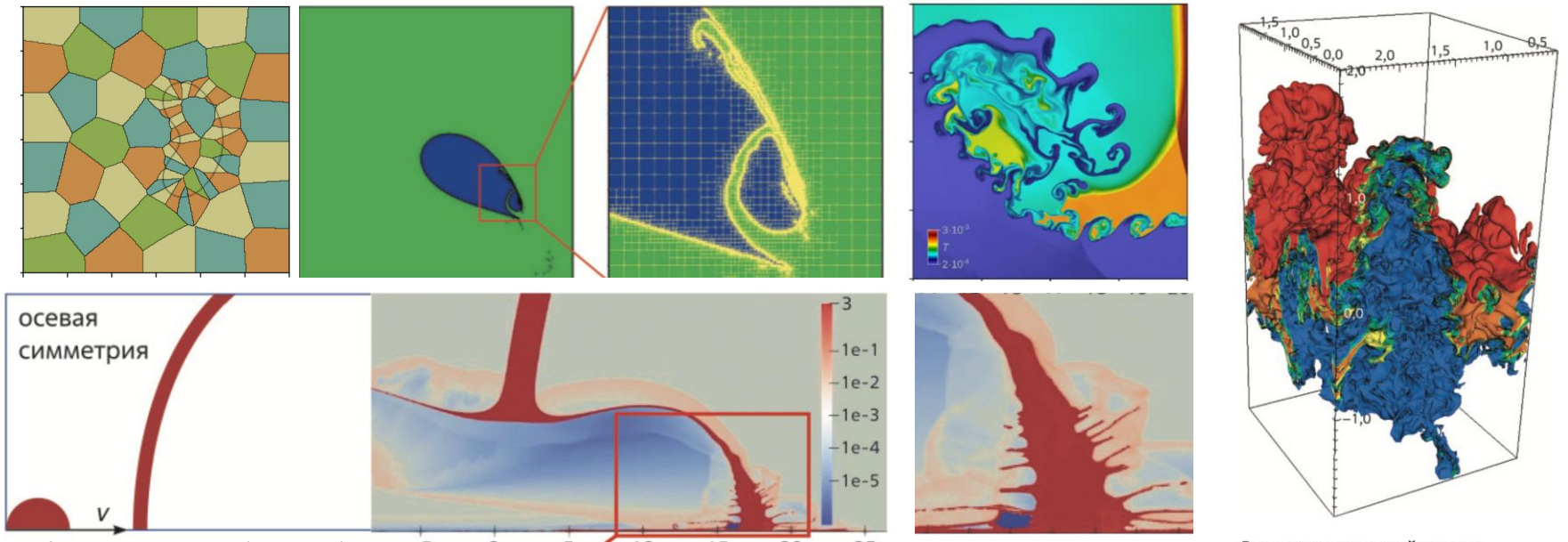
$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{f}_x}{\partial x} + \frac{\partial \mathbf{f}_y}{\partial y} + \frac{\partial \mathbf{f}_z}{\partial z} = \mathbf{S};$$

$$\mathbf{q} = \left(\rho, \rho v, \rho u, \rho w, \rho E, \rho\beta_1, \ldots, \rho\beta_n\right)^T.$$

$$V_i \frac{d\mathbf{q}_i}{dt} = -\sum_{\sigma} (\mathbf{f}_k n_k) s_\sigma .$$

Для расчёта потока в области контактных границ применяется аппроксимация численного потока, основанная на решении **составной задачи Римана**: вводится подсеточная реконструкция контактных границ материалов и там, где это возможно, потоки вычисляются между чистыми компонентами.

Menshov I. Zakharov P. Journal for Numerical Methods in Fluids V. 76, No 2. P. 109–127 (2014).



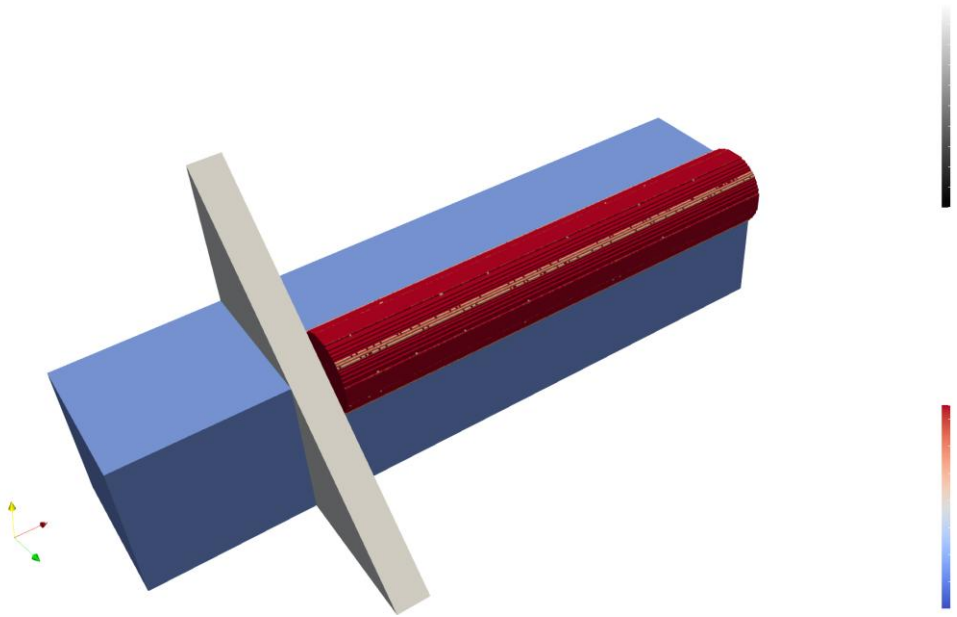
Моделирование пробития обшивки космического летательного аппарата. Разрушение обшивки с разным уровнем сеточной адаптации

Развитие неустойчивого процесса горения

В 2025 году

1. Реализована возможность моделирования упругопластических течений

$$\frac{\partial \varphi^\alpha \rho^\alpha}{\partial t} + \frac{\partial \varphi^\alpha \rho^\alpha u}{\partial x} = 0$$
$$\frac{\partial \varphi^\beta \rho^\beta}{\partial t} + \frac{\partial \varphi^\beta \rho^\beta u}{\partial x} = 0$$
$$\frac{\partial (\varphi^\alpha \rho^\alpha + \varphi^\beta \rho^\beta) u}{\partial t} + \frac{\partial (\varphi^\alpha \rho^\alpha + \varphi^\beta \rho^\beta) u^2}{\partial x} - \frac{\partial \sigma_{11}}{\partial x} = 0$$
$$\frac{\partial (\varphi^\alpha \rho^\alpha + \varphi^\beta \rho^\beta) E}{\partial t} + \frac{\partial (\varphi^\alpha \rho^\alpha + \varphi^\beta \rho^\beta) E u}{\partial t} - \frac{\partial \sigma_{11} u}{\partial x} = 0$$
$$\frac{\partial (\varphi^\alpha \rho^\alpha S_{11}^\alpha + 4/3 \mu^\alpha \varphi^\alpha \rho^\alpha ln \rho^\alpha)}{\partial t} + \frac{(\partial (\varphi^\alpha \rho^\alpha S_{11}^\alpha + 4/3 \mu^\alpha \varphi^\alpha \rho^\alpha ln \rho^\alpha) u)}{\partial x} = 0$$
$$\frac{\partial (\varphi^\beta \rho^\beta S_{11}^\beta + 4/3 \mu^\beta \varphi^\beta \rho^\beta ln \rho^\beta)}{\partial t} + \frac{(\partial (\varphi^\beta \rho^\beta S_{11}^\beta + 4/3 \mu^\beta \varphi^\beta \rho^\beta ln \rho^\beta) u)}{\partial x} = 0$$
$$\frac{\partial \varphi^\alpha}{\partial t} + u \frac{\partial \varphi^\alpha}{\partial x} - (\varphi^\beta \Lambda^\beta - \varphi^\alpha \Lambda^\alpha) \frac{\partial u}{\partial x} = 0$$



2. Реализована возможность моделирования капиллярных явлений

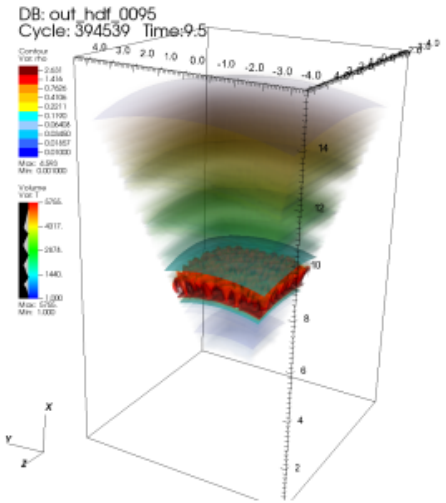
pyphia.solver.FVM

Решатель для магнитной гидродинамики

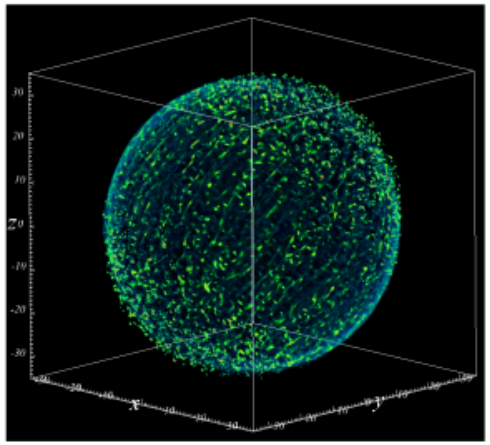
ВНИИА
РОСАТОМ

Создан для расчётов физики плазмы:

- 1T, 2T, многожидкостные модели
- Магнитная гидродинамика
- Модели теплопереноса
- Радиационный перенос (серое приближение, многогрупповой)
- Модели лазерного поглощения
- Ионизационные процессы (равновесные модели, кинетика)
- Транспорт нетепловых частиц
- Модули для расчётов свойств веществ (УРС, пробеги)
- Релятивистская гидродинамика
- Химические и ядерные реакции



Мишени ЛТС



Сверхновые

$$\partial_t \rho_c + \partial_i(\rho_c v_i^{(c)}) = 0$$

$$\partial_t(\rho_c v_i^{(c)}) + \partial_j(\rho_c v_i^{(c)} v_j^{(c)} + \delta_{ij} p_c) = \frac{1}{c}[\mathbf{j} \times \mathbf{B}]_i$$

$$\partial_t(\rho_c e_c) + \partial_i(\rho_c v_i^{(c)} e_c) + p_c \partial_i v_i^{(c)} = \mathbf{jE}$$

$$\partial_t \mathbf{B} = \mathrm{rot}[\mathbf{v} \times \mathbf{B}]$$

pyphia.solver.SPH

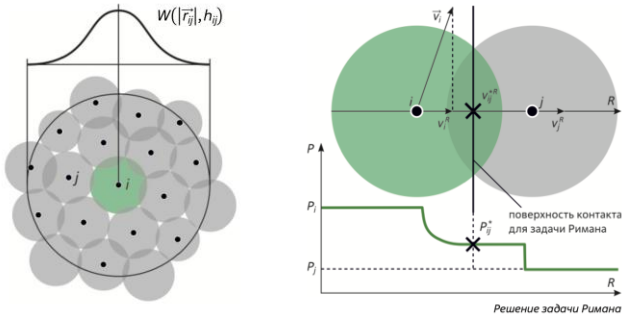
Решатель метода сглаженных частиц

ВНИИА
РОСАТОМ

Схема бессеточного метода SPH [8]

$$f(\mathbf{r}_i) = \sum_{i=1}^{N} \frac{m_i}{\rho_i} f(\mathbf{r}_i) W_{ij} + O(h^2)$$

$$\nabla f(\mathbf{r}_i) = \sum_{i=1}^{N} \frac{m_i}{\rho_i} f(\mathbf{r}_i) \nabla_i W_{ij} + O(h^2)$$



Уравнения контактного метода SPH:

$$\frac{d\rho_i}{dt} = 2\rho_i \sum_j \frac{m_j}{\rho_j} \left(\vec{v}_i - \vec{v}_{ij}^*\right) \cdot \nabla_i W_{ij};$$

$$\frac{d\vec{v}_i}{dt} = -\frac{2}{\rho_i} \sum_j \frac{m_j}{\rho_j} P_{ij}^* \cdot \nabla_i W_{ij};$$

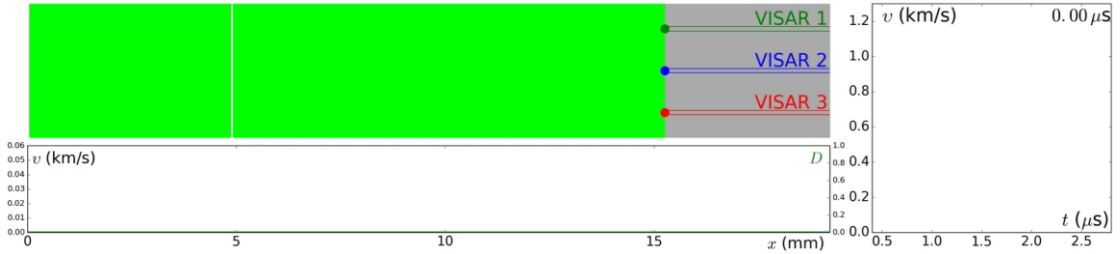
$$\frac{d(e_i + \frac{v_i^2}{2})}{dt} = \frac{1}{\rho_i} \sum_j \frac{m_j}{\rho_j} P_{ij}^* \vec{v}_{ij}^* \cdot \nabla_i W_{ij}.$$

Система замыкается уравнением состояния.

Моделирование кумулятивной струи

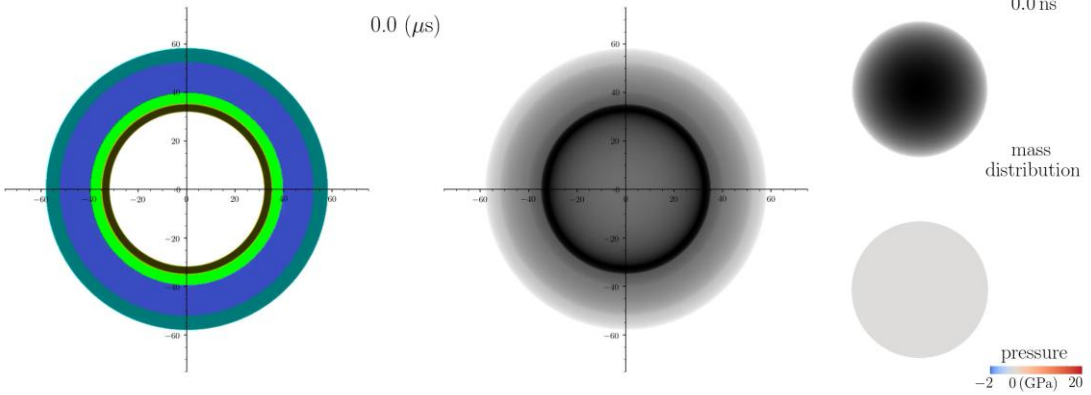
M.S. Egorova et al. Comput. Phys. Commun. (2019) DOI: 10.1016/j.cpc.2018.07.019

Моделирование ударных волн, откола и разрушения

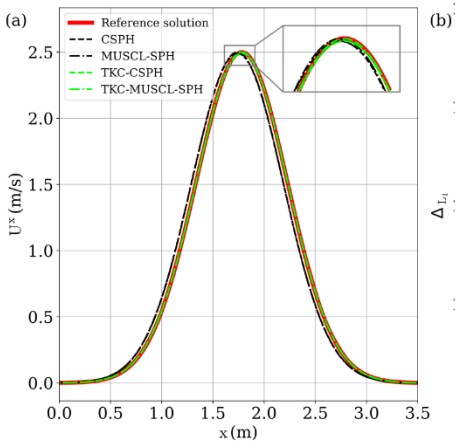


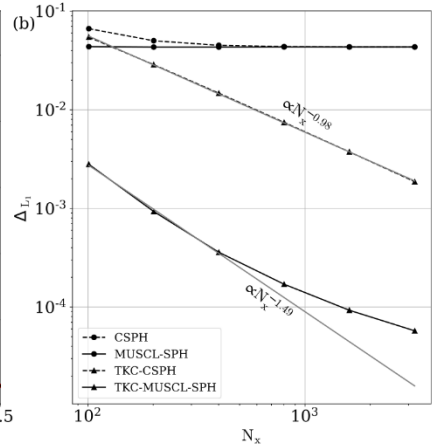
S.A. Dyachkov et al. Journal of Applied Physics 124, 085902 (2018); DOI: 10.1063/1.5043418.

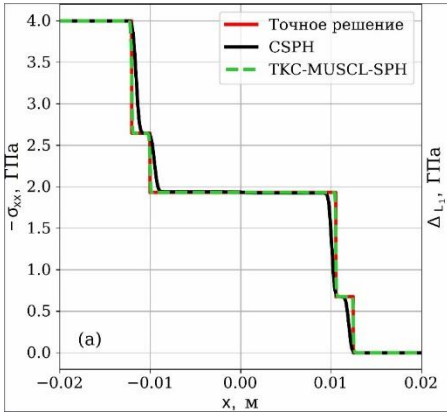
Взрывное и лазерное воздействие

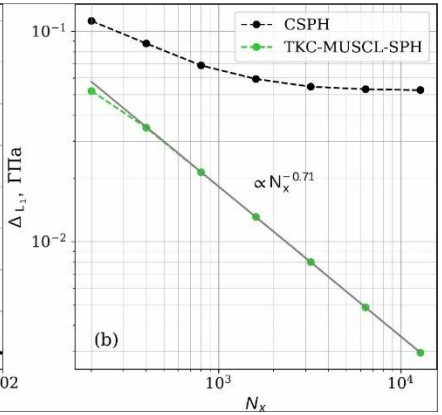


Введена корректировка градиента сглаживающего ядра.
Повышена точность моделирования упругопластических сред.
Реализован осесимметричный решатель.

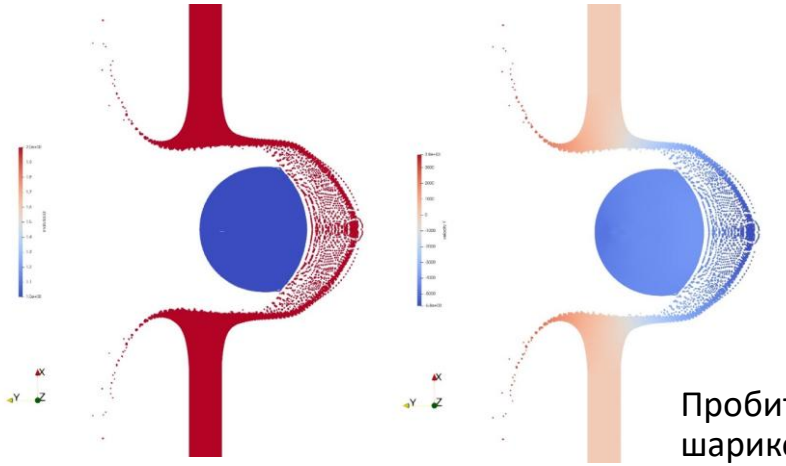


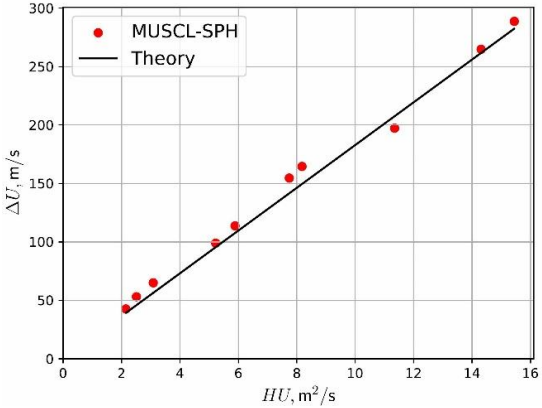






G.D. Rublev, A.N. Parshikov, S.A. Dyachkov. Applied Mathematics and Computation. V. 488, pp. 129128 (2025) DOI: 10.1016/j.amc.2024.129128 (Q1)





Пробитие преграды из ПММА толщиной H=1,74 мм стальным шариком диаметром 6,35 мм со скоростью U = 4,45 км/с

pyphia.solver.FEM

Решатель метода конечных элементов

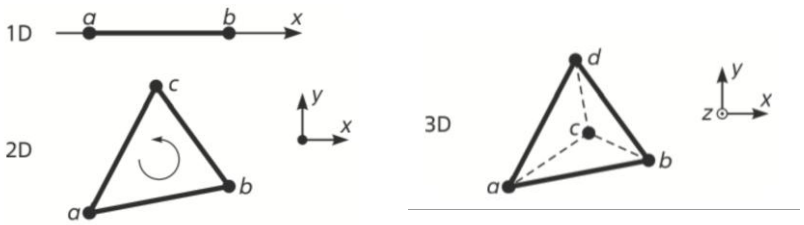
ВНИИА
РОСАТОМ

Метод конечных элементов для многоматериальных течений обеспечивает решение системы уравнений в лагранжевых координатах:

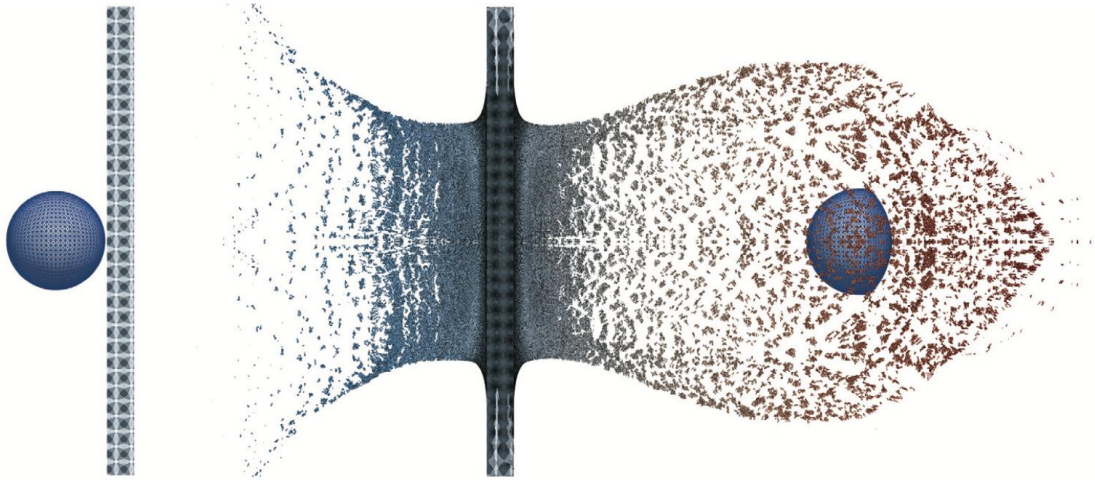
$$\frac{dM}{dt} = 0; \qquad \frac{dv}{dt} = \frac{1}{\rho}(\nabla \cdot \hat{\sigma});$$

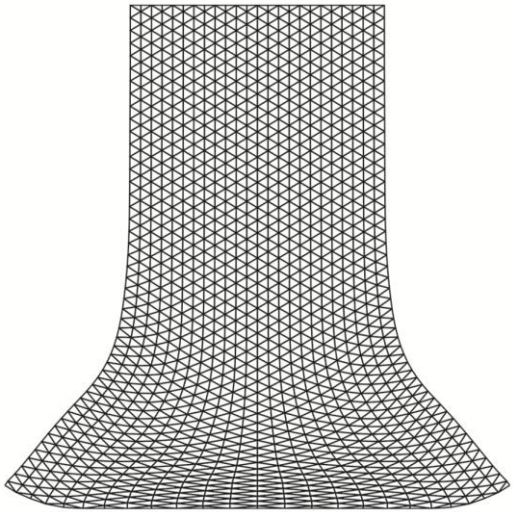
$$\frac{dE}{dt} = -P\dot{V} + V(\hat{s} \circ \hat{\epsilon});$$

$$\frac{d\hat{s}}{dt} = 2G\left(\hat{\epsilon} - \frac{\hat{I}}{3}\dot{\epsilon}_v\right) - \hat{s}\hat{\Omega} + \hat{\Omega}\hat{s}.$$



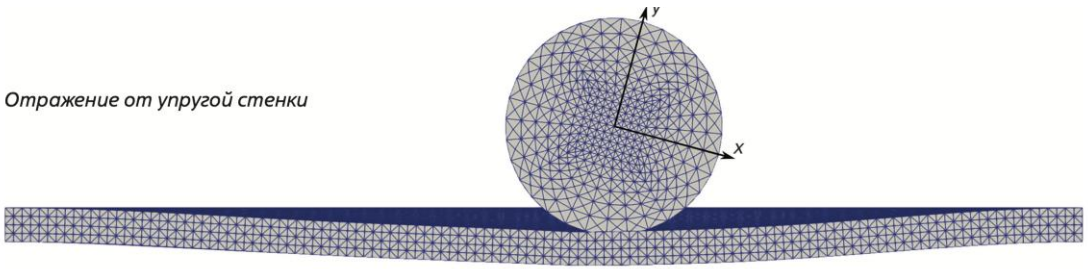
Пробитие SPH-преграды сеточным ударником



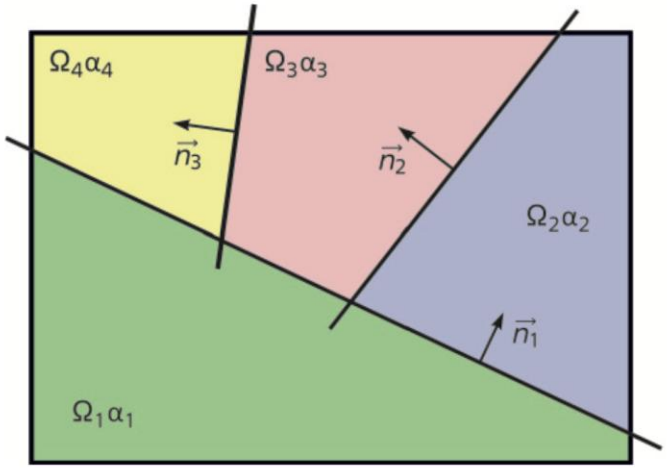


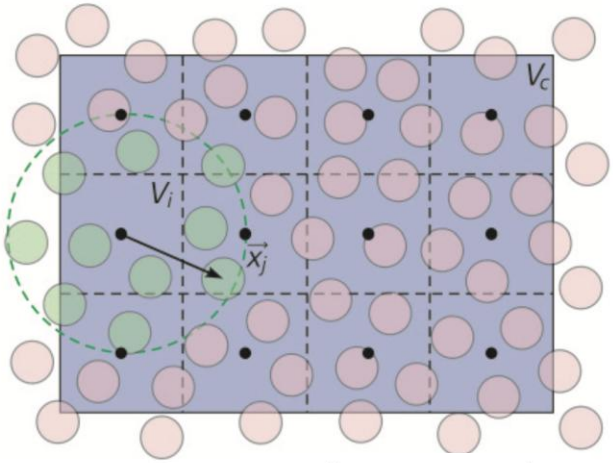
Моделирование теста Тейлора

Отражение от упругой стенки



Сопряжение метода конечного объема и SPH





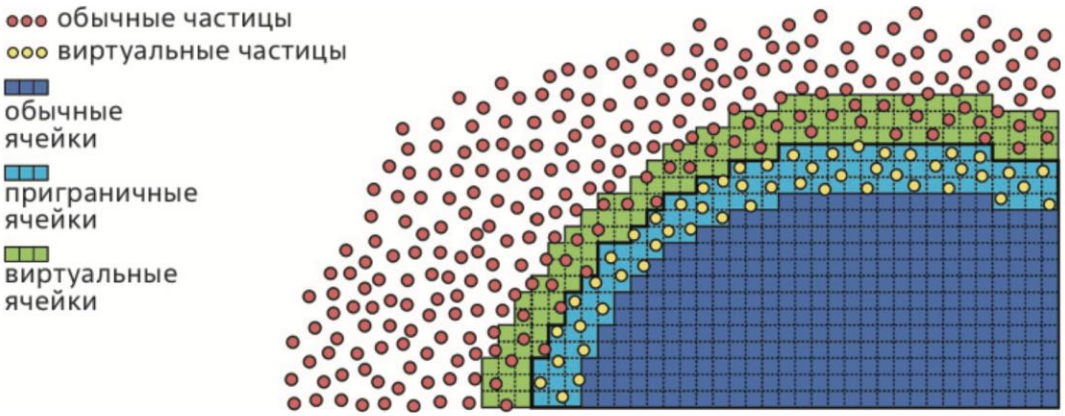
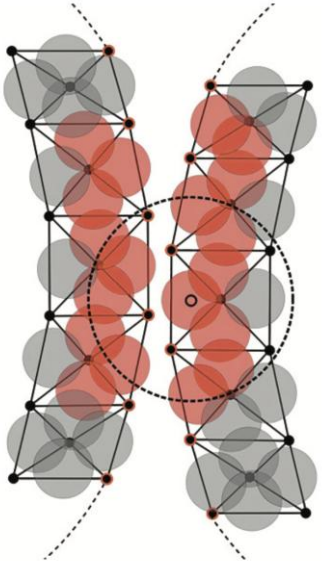
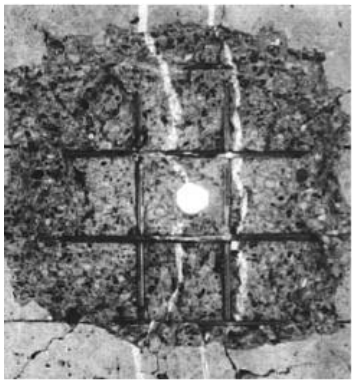
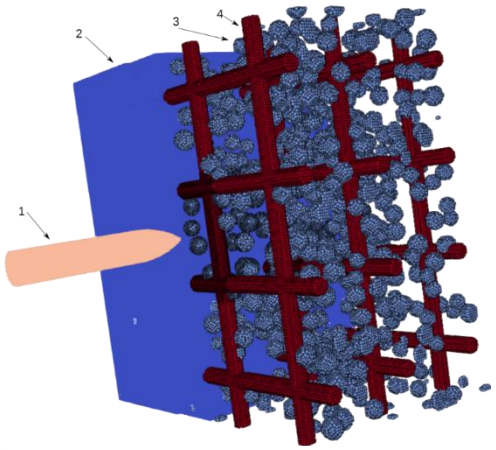


Схема интерфейса сопряжения

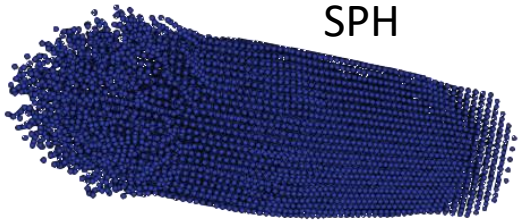


Сопряжение FEM и SPH

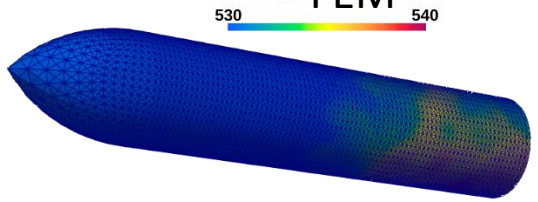


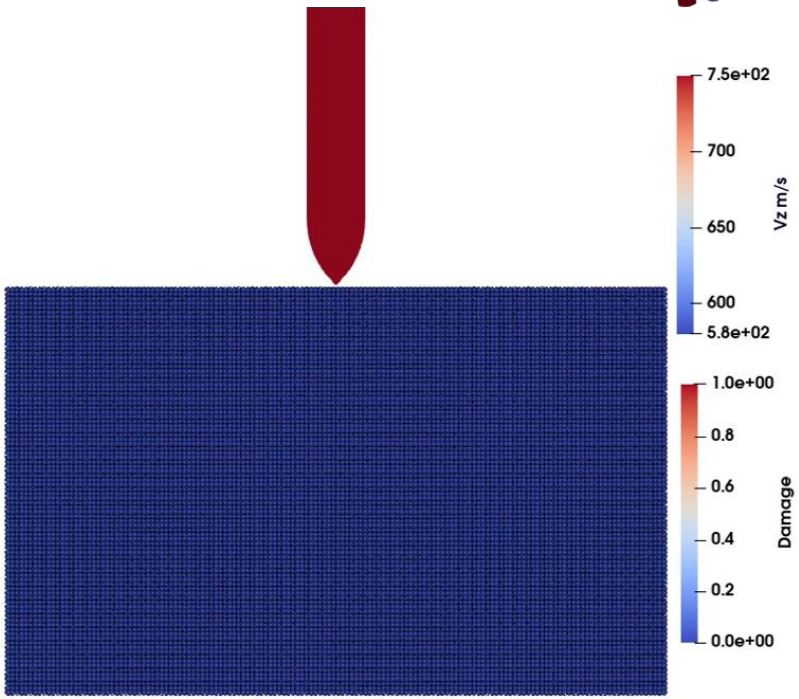


SPH

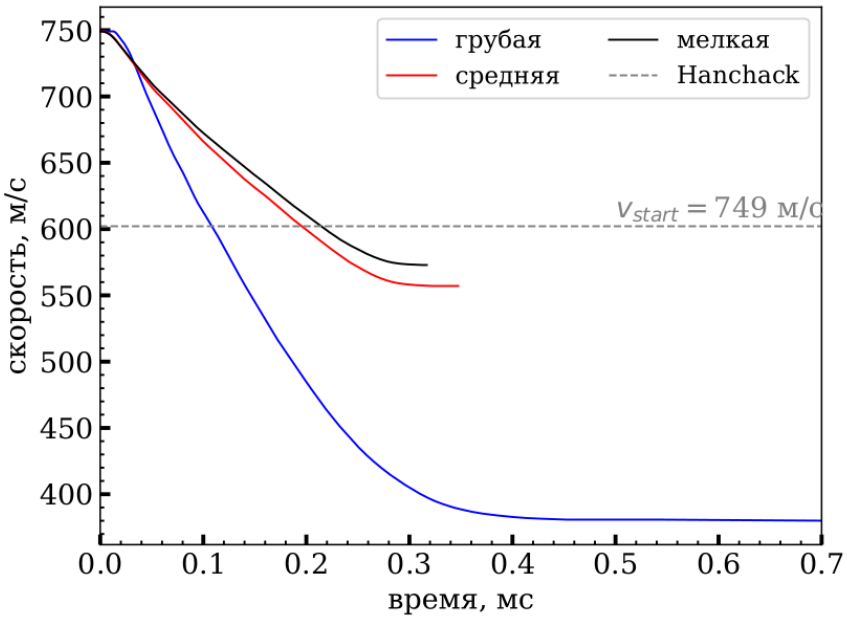


FEM





Пробитие преграды из железобетона



pyphia.solver.MD

Решатель метода молекулярной динамики

ВНИИА
РОСАТОМ

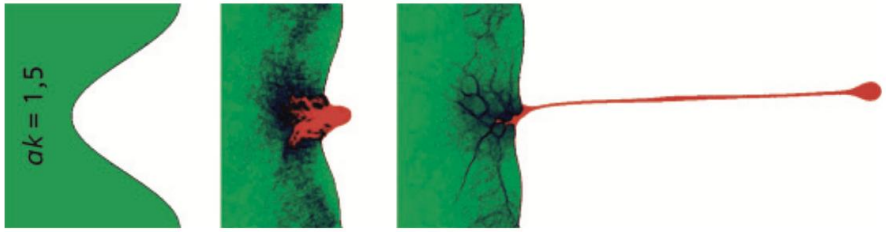
Модуль обеспечивает решение системы уравнений Ньютона для N взаимодействующих частиц:

$$m_i \frac{d^2\mathbf{r}_i(t)}{dt^2} = \mathbf{f}_i = -\frac{\partial U}{\partial r_i}; \quad i = 1, \ldots, N.$$

Как правило, для интегрирования по времени используется схема Верле:

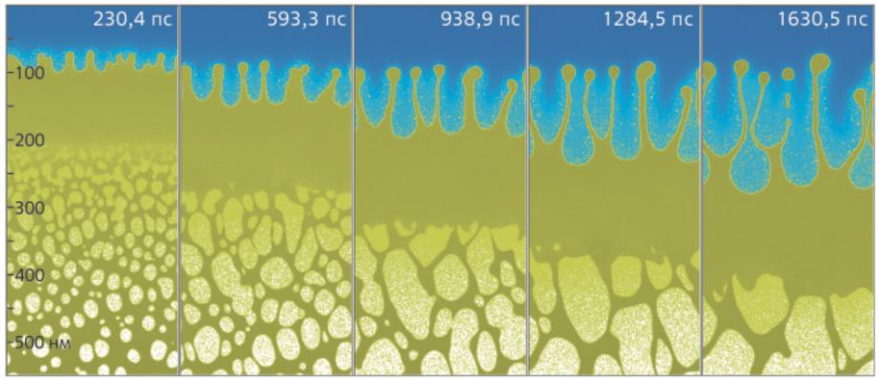
$$\mathbf{r}_i(t+\Delta t) = \mathbf{r}_i(t) + \mathbf{v}(t)\Delta t + \frac{1}{2}\left(\frac{\mathbf{f}_i(t)}{m_i}\right)\Delta t^2;$$

$$\mathbf{v}_i(t+\Delta t) = \mathbf{v}_i(t) + \frac{1}{2}\left[\frac{\mathbf{f}_i(t)}{m_i} + \frac{\mathbf{f}_i(t+\Delta t)}{m_i}\right]\Delta t.$$

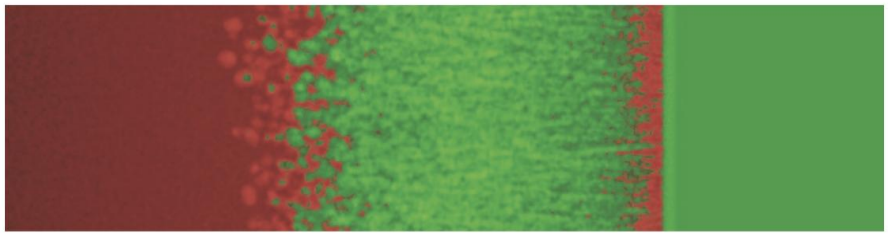


Выброс кумулятивной струи расплавленной меди при выходе ударной волны на гофрированную поверхность

S. A. Dyachkov et al. // AIP Conf. Proc. 1793, 100024 (2017) DOI: 10.1063/1.4971649



Формирование наночастиц золота при абляции в воду под действием ультракоротких лазерных импульсов



Кристаллизация за фронтом ударной волны от воздействия пикосекундного лазерного импульса в титане при лазерной ковке

V.V. Zhakhovsky et al. Phys. Fluids 35, 096104 (2023) DOI: 10.1063/5.0165622

Моделирование работы полупроводников

$$\Delta\varphi = -\frac{q}{\varepsilon\varepsilon_0}\left(p - n + N_d - N_a\right)$$

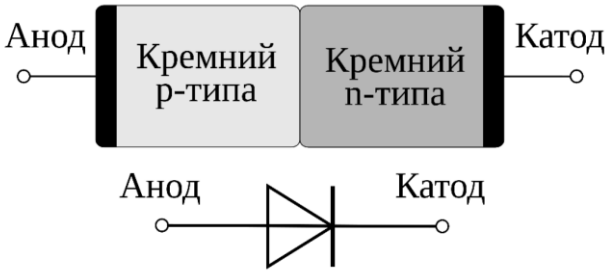
$$\mathrm{div}\boldsymbol{J}_n - qR - q\frac{\partial n}{\partial t} = 0$$

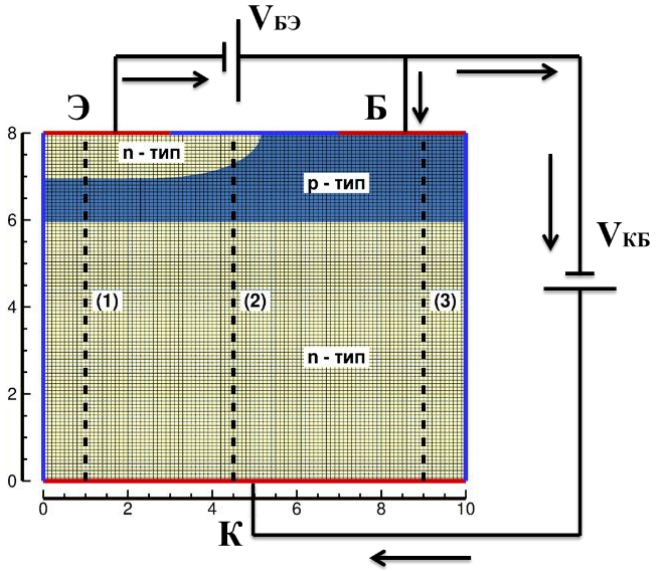
$$\mathrm{div}\boldsymbol{J}_p + qR + q\frac{\partial p}{\partial t} = 0$$

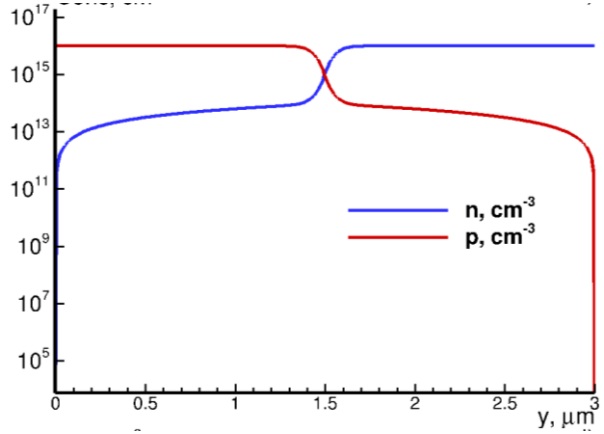
$$\boldsymbol{J}_n = qD_n\nabla n - q\mu_n\nabla\varphi$$

$$\boldsymbol{J}_p = -qD_p\nabla p - q\mu_p\nabla\varphi$$

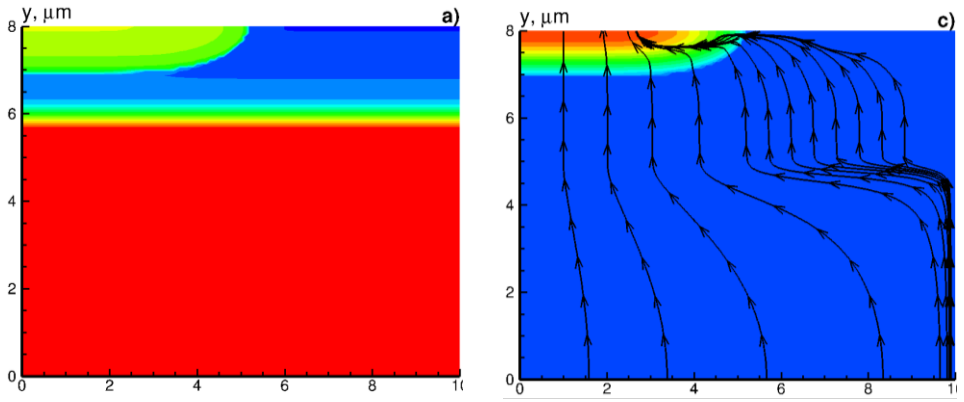
$$D_n = \frac{\mu_n E_n}{q},\ D_p = \frac{\mu_p E_p}{q}$$







Распределение концентрации носителей заряда и напряженность электрического поля в диоде (V=0.6 В)



Распределение потенциала, концентрации электронов в биполярном транзисторе при $V_{БЭ}$= 0.4 В.

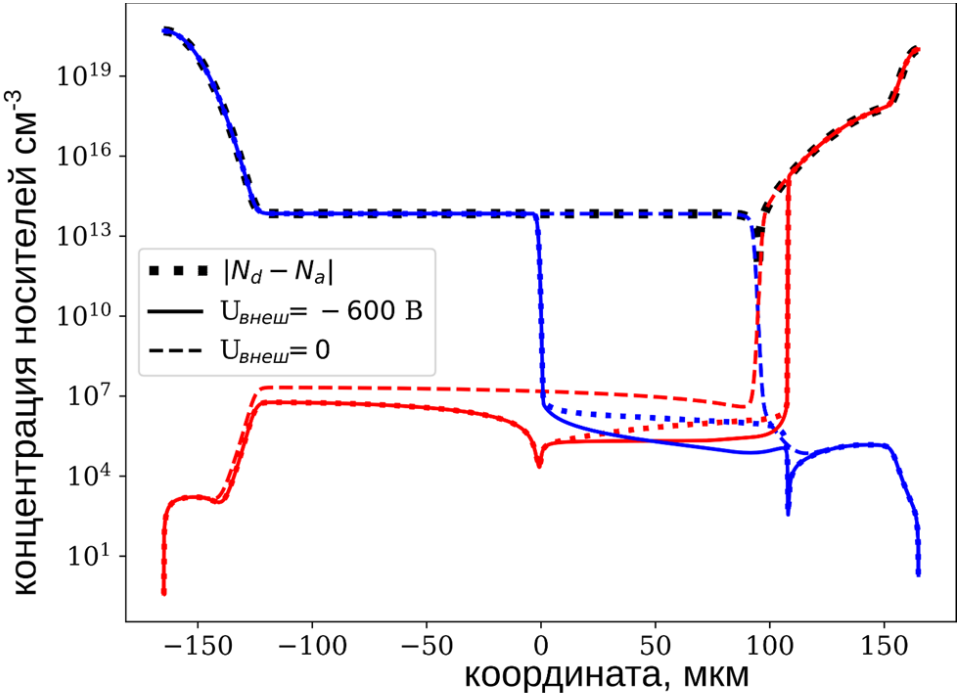
Моделирование работы полупроводников

Аппроксимация системы уравнений методом сглаженных частиц

1. Аппроксимация системы уравнений в приближении Шарфеттера – Гуммеля на основе метода SPH для теплопроводности

$$< \nabla \cdot (\varepsilon \nabla \varphi) >_i = 2 \sum_j \frac{\varepsilon_{ij} V_j W'_{ij} \mathbf{G}_{ij} \vec{e}_{ij} \cdot \vec{e}_{ij}}{r_{ij}} (\varphi_i - \varphi_j)$$

2. Полностью бессеточный подход
3. Предложенный метод является аппроксимацией при использования матрицы ***G*** корректировки операторов
4. В граничных частицах с условием Дирихле заданы значения, для смешанных условий использована оценка производной методом воспроизводящих ядер сглаженных частиц.



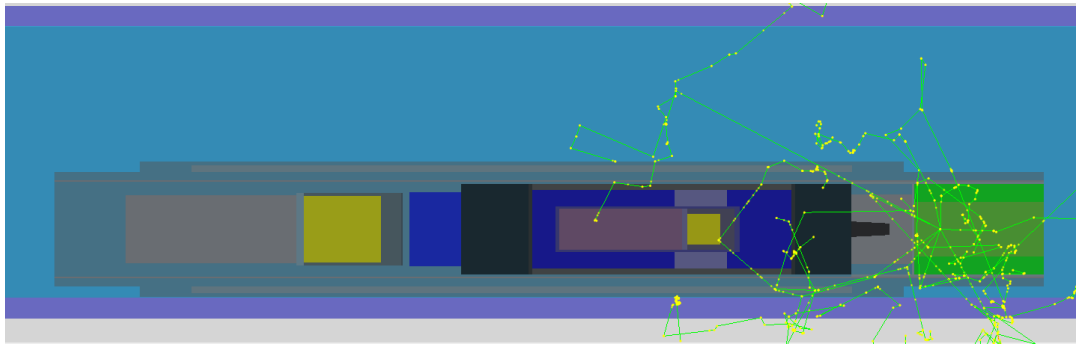
Пример расчета pin диода сильно легированного по краям с обратным напряжением 600 В

$$\begin{bmatrix} 1 \dots 0 \\ \dots A_{11}^{(\varphi)} \dots A_{1j}^{(\varphi)} \\ \vdots \\ \dots A_{Mj}^{(\varphi)} \dots A_{MM}^{(\varphi)} \dots \end{bmatrix} \begin{bmatrix} \varphi_0^{[k+1]} \\ \varphi_1^{[k+1]} \\ \vdots \\ \varphi_M^{[k+1]} \end{bmatrix} = \begin{bmatrix} \varphi_0^{(G)} \\ b_1^{(\varphi)} \\ \vdots \\ b_M^{(\varphi)} \end{bmatrix}$$

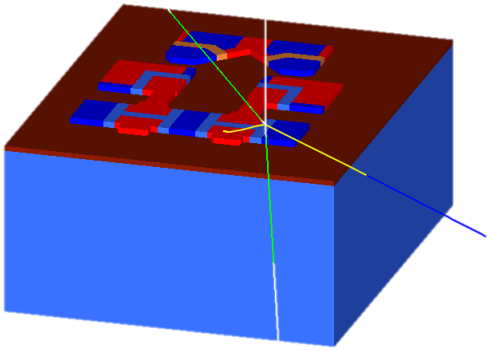
Условие Дирихле

Внутренние и граничные точки

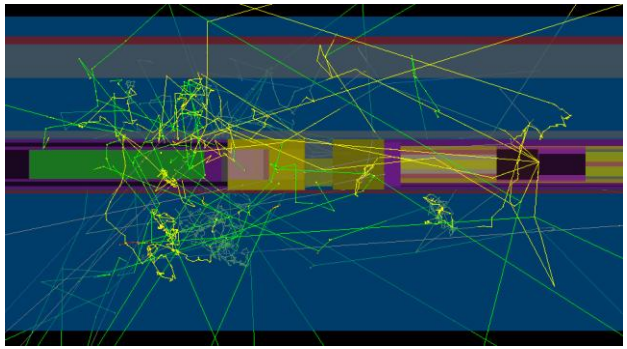
обеспечивает Монте-Карло моделирование транспорта частиц в среде с учетом процессов упругого и неупругого рассеяния, радиационного торможения и др. При расчетах используется обширная база данных ядерных реакций и сечений взаимодействий частиц.

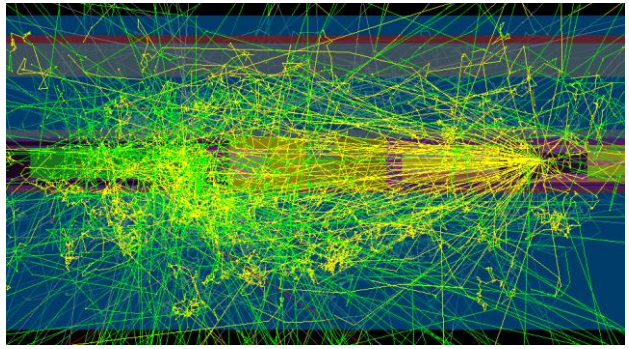


Моделирование каротажной установки



Моделирование радиационной стойкости электроники

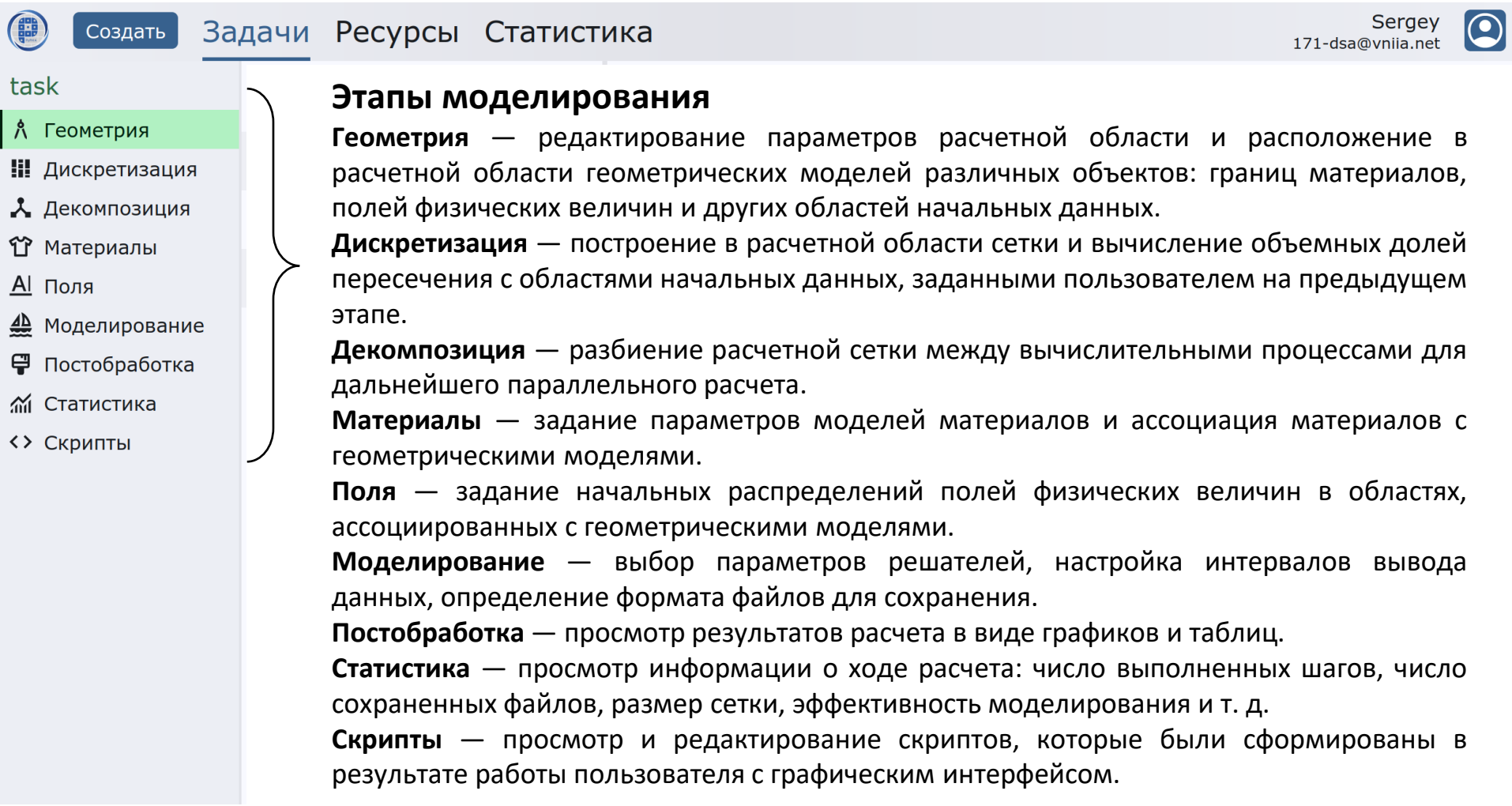




16 и 150 событий, в которых рождённые нейтронами γ-кванты попали в γ-детектор.

Интерфейс пользователя
Веб-интерфейс

ВНИИА
РОСАТОМ



Этапы моделирования

Геометрия — редактирование параметров расчетной области и расположение в расчетной области геометрических моделей различных объектов: границ материалов, полей физических величин и других областей начальных данных.

Дискретизация — построение в расчетной области сетки и вычисление объемных долей пересечения с областями начальных данных, заданными пользователем на предыдущем этапе.

Декомпозиция — разбиение расчетной сетки между вычислительными процессами для дальнейшего параллельного расчета.

Материалы — задание параметров моделей материалов и ассоциация материалов с геометрическими моделями.

Поля — задание начальных распределений полей физических величин в областях, ассоциированных с геометрическими моделями.

Моделирование — выбор параметров решателей, настройка интервалов вывода данных, определение формата файлов для сохранения.

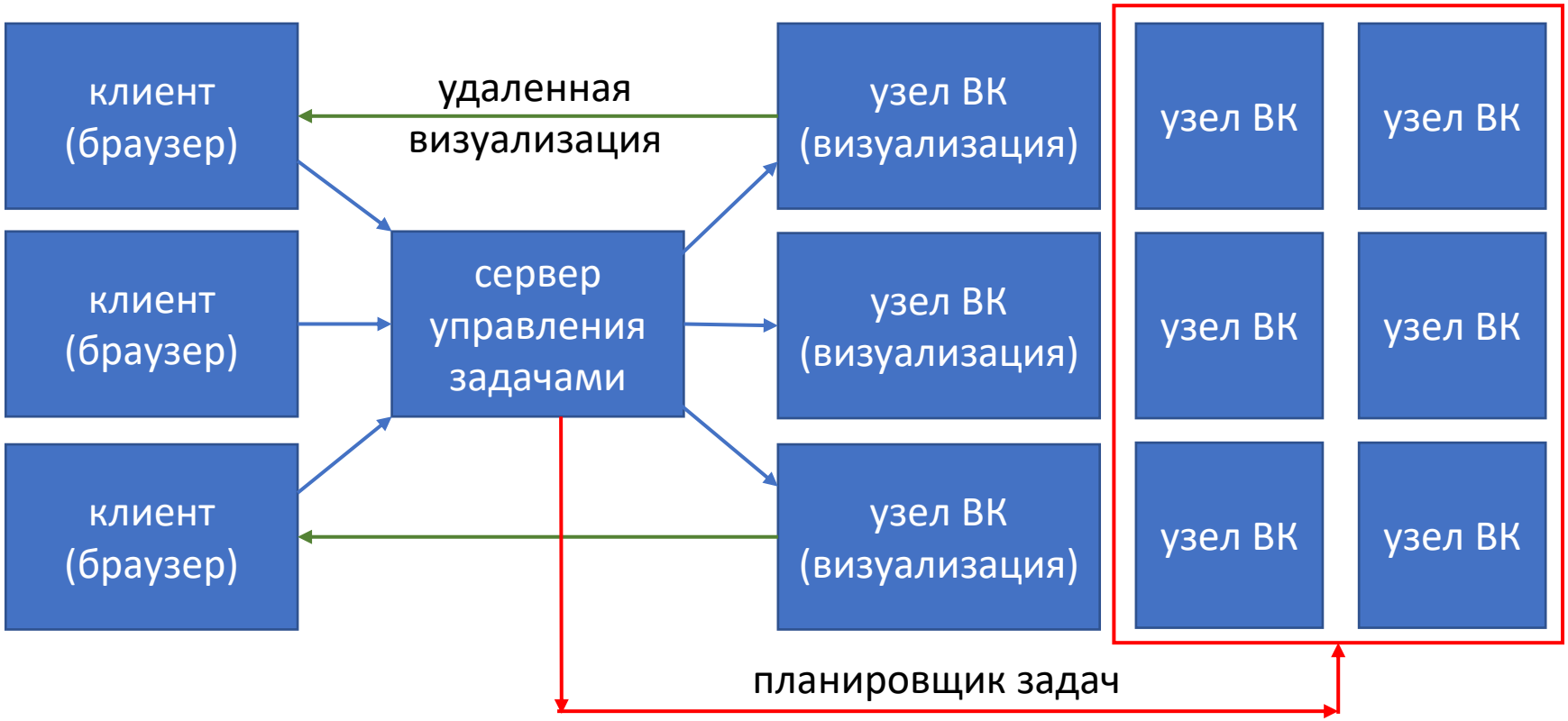
Постобработка — просмотр результатов расчета в виде графиков и таблиц.

Статистика — просмотр информации о ходе расчета: число выполненных шагов, число сохраненных файлов, размер сетки, эффективность моделирования и т. д.

Скрипты — просмотр и редактирование скриптов, которые были сформированы в результате работы пользователя с графическим интерфейсом.

Интерфейс пользователя
Архитектура

ВНИИА
РОСАТОМ



- Пользователь открывает сайт в браузере, открывается интерфейс управления задачами
- Выбирается задача, отправляется запрос на старт сервера визуализации
- Осуществляется интерактивное задание геометрии, работа с сеткой, в соответствующих формах задаются параметры задачи
- Последовательность команд транслируется в Python, скрипт ставится в очередь на счет
- Результаты просматриваются также на сервере визуализации

Заключение

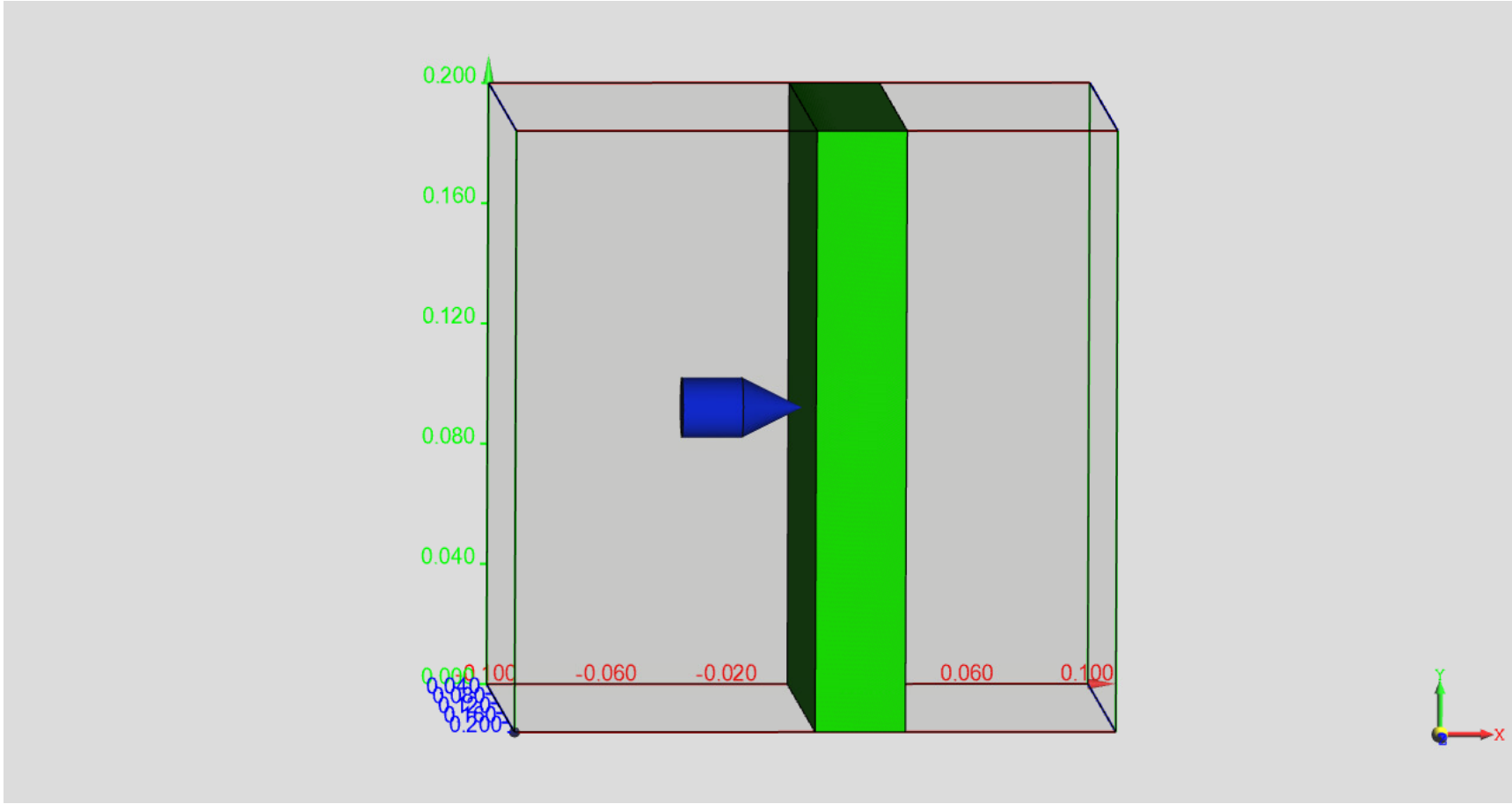
ВНИИА
РОСАТОМ

- Разработана программная платформа PyPHIA, в рамках которой обеспечен единый формат представления данных и общий алгоритм параллелизации вычислений
- В рамках платформы успешно реализованы сеточные и бессеточные методы решения задач механики сплошной среды, а также ведется работа по развитию интерфейсов сопряжения различных методик
- Для выскопроизводительных модулей реализованы соответствующие интерфейсы на языке Python, поддерживающие NumPy, что позволяет проводить интеграцию модулей комплекса с большим количеством Python-библиотек
- Ведутся работы по созданию полноценного пользовательского интерфейса в виде интерактивного веб-приложения с возможностью проведения высокопроизводительных вычислений

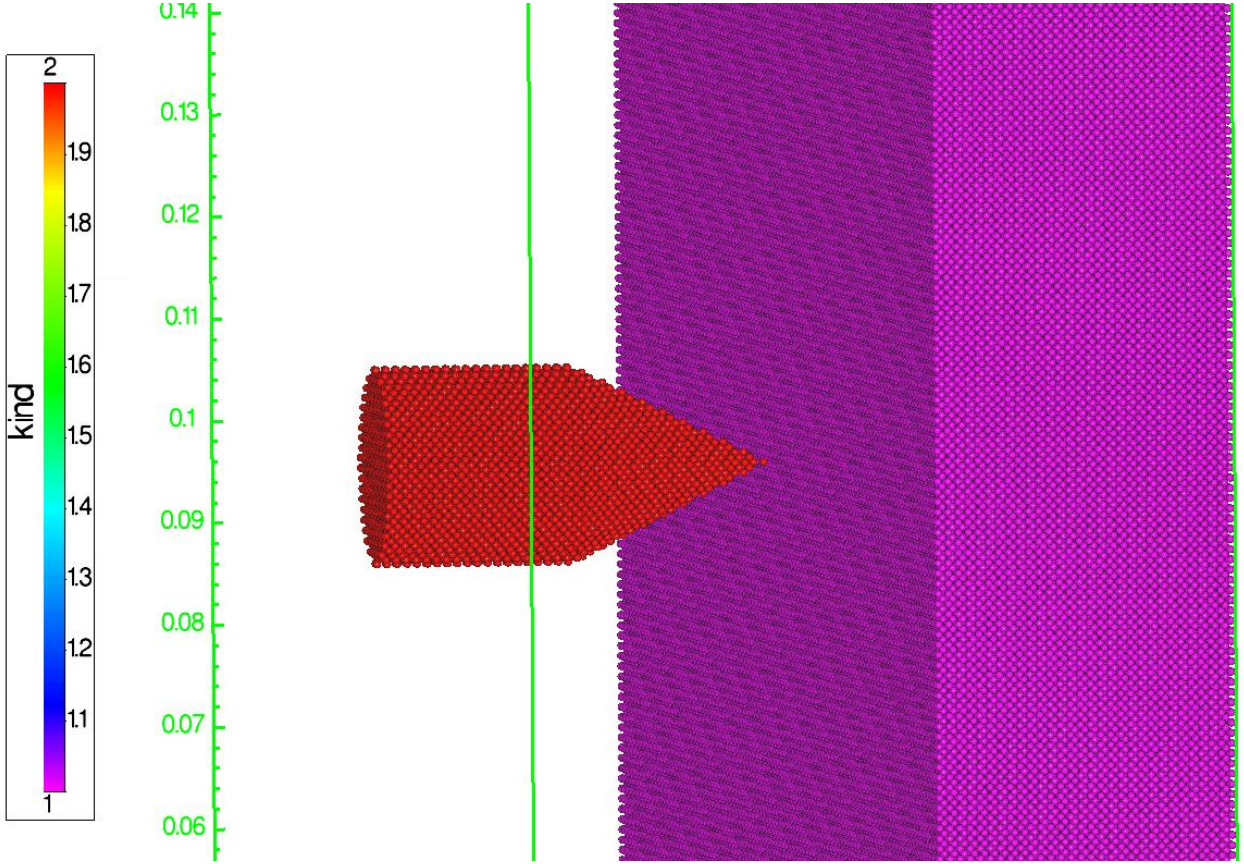
ВНИИА
РОСАТОМ

Проведение расчетов

Задание начальных данных (препроцессинг)



Расчет сетки частиц (результат)



Скрипт для расчета
задание моделей материалов

ВНИИА
РОСАТОМ

```
# ======= setup models ============================
models = ModelList(threads=threads)

B4C = CeramicModel("B4C")

fe_eos     = eos(dens0=7900, c0=4570, a=1.4, gamma=2.0)
fe_elastic = elastic(yieldStrength=0.5e+9, poissonRatio=0.25)
fe_tensile = tensile(T=2.0e+9)

iron = CompoundModel()

iron.add(fe_eos)
iron.add(fe_elastic)
iron.add(fe_tensile)

models.add(B4C)
models.add(iron)
```

- Железо – составная модель: УрС + упругопластика + откол
- Карбид бора – одна модель

Скрипт для расчета
задание вектора состояния и загрузка данных

ВНИИА
РОСАТОМ

```
# ========= define field quantities ===========
dtype = np.dtype([
    coords,
    velocity,
    force,
    mass,
    density,
    pressure,
    energy,
    energyDerivative,
    soundSpeed,
    strainRate,
    tensileStrength,
    yieldStrength,
    shearModulus,
    neibsSearchRadius,
    strainRateDeviator,
    stressDeviator,
    soundSpeedLongitudinal,
    soundSpeedTransversal,
    angularVelocity,
    equivalentStress,
    Dfrag,
    dPres,
    material,
    element,
    size
])
```

```
# ========= load initial geometry =============
infile = hdf.file(name="bullet_result.h5", access="r", driver="mpio")

nVD = network.size

Npart = infile["particles"]["set0"].shape[0]
Nparts_from = [0]
Nparts_to   = [Npart//nVD]
for i in range(1, nVD):
    Nparts_from.append(Nparts_from[i - 1] + Npart//nVD)
    Nparts_to.append(Nparts_to[i - 1] + Npart//nVD)
Nparts_to[-1] = Npart
my_rank = network.self.rank
partgeom = infile["particles"]["set0"][Nparts_from[my_rank]:Nparts_to[my_rank]]

elements = Storage(dtype, partgeom.size)

elements.data["coords"] = partgeom["r"]
elements.data["material"]["id"] = partgeom["kind"]
elements.data["size"]["value"] = partgeom["size"]
elements.data["element"]["kind"] = kind.SPH
elements.data["element"]["dimension"] = 3
```

Скрипт для расчета
организация вывода частиц на диск

ВНИИА
РОСАТОМ

```
# ====== setup converter to particles =========
wsize = np.array([0.1, 0.1, 0.1])
wcenter = np.array([0.0, 0.1, 0.1])
worientation = np.array([
    1.0, 0.0, 0.0,
    0.0, 1.0, 0.0,
    0.0, 0.0, 1.0
])
window3D = Window(wsize, wcenter, worientation)
to_out_particles = Particles(
    window3D,
    np.dtype([
        coords,
        velocity,
        density,
        pressure,
        Dfrag,
        energy,
        size
    ])
)
# all dtypes to save particles
to_save_particles = Particles(window3D, dtype)
# ============================================
```

```
def output_particles(fname, elements, converter):
    converter.convert(elements, network)
    outparticles = converter.get()
    outfile = hdf.file(name=fname, access="w", driver="mpio")
    my_rank = network.self.rank
    counts = []
    for rank in range(network.size):
        outfile["window"]["1"]["particles"]["set" + str(rank)] =
            hdf.dataset(
                outparticles.data.dtype,
                converter.count(rank)
            )
        counts.append(converter.count(rank))
    outfile["window"]["1"]["particles"]["set" + str(my_rank)] = outparticles.data
```

- При параллельной записи HDF-файл должен быть подготовлен независимо каждым MPI-процессом, после чего производится запись в конкретную часть файла
- Информация о числе частиц в каждом MPI-процессе передается при запуске конвертера

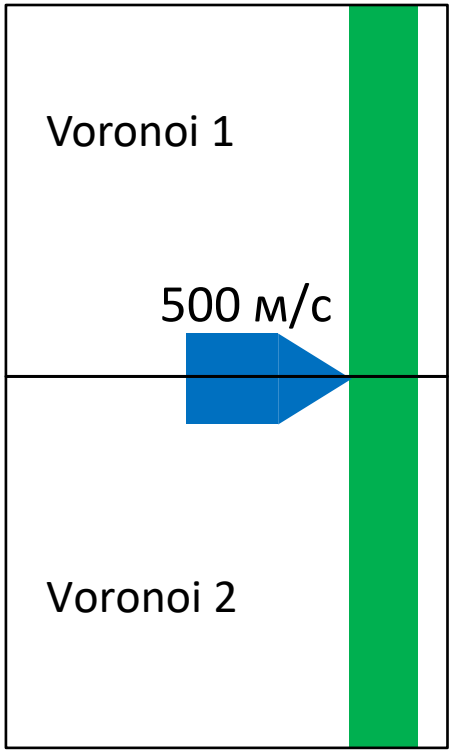
Скрипт для расчета
декомпозиция расчетной области и начальные условия

ВНИИА
РОСАТОМ

```
# ===== domain decomposition =================
bounds = Bounds(-0.1, 0.3,
                -0.1, 0.3,
                -0.1, 0.3)

VDcenters = Storage(np.dtype([coords]), nVD)
VDcenters.data["coords"]["x"] = 0.0
VDcenters.data["coords"]["y"] = np.array([0.07, 0.014])
VDcenters.data["coords"]["z"] = 0.0
VD = Decomposer(
    network=network,    # required
    elements=elements,  # required
    bounds=bounds,      # required
    dimension=3,        # optional
    threads=threads,    # optional
    centroidal=0.25,    # optional
    centers=VDcenters,  # optional
    stopwatch=useful    # optional
)
VD.update()
VD.exchange()
```

```
nlist = Nlist(
    decomposition=VD,
    dimension=3,
    threads=threads,
    periodic_x = False,
    periodic_y = False,
    periodic_z = False
)
# init SPH solver and stepper
solver = Solver(
    kernels["WendlandC2"],
    threads=threads
)
stepper = Stepper(
    models=models,
    solver=solver,
    threads=threads
)
```




```
# ========= setup initial conditions ==========
iFe  = np.nonzero(np.equal(elements.data["material"]["id"], models.id(iron)))
iB4C = np.nonzero(np.equal(elements.data["material"]["id"], models.id(B4C)))

elements.data["density"]["value"][iFe]  = iron.density0
elements.data["density"]["value"][iB4C] = B4C.density0
elements.data["velocity"] = 0.0
elements.data["velocity"]["x"][iFe] = 500.0
elements.data["energy"]["value"] = 0.0
elements.data["mass"]["value"] = elements.data["density"]["value"]*
                                 elements.data["size"]["value"]**3
elements.data["neibsSearchRadius"]["value"] = 4.0*elements.data["size"]["value"]
```

Скрипт для расчета
основной цикл

ВНИИА
РОСАТОМ

```
while(nSteps <= 10000):
    if (nSteps % 100 == 0):
        output_particles(
            "out/bullet-" + str(nSteps) + ".h5",
            elements, to_out_particles
        )
    if (nSteps % 1000 == 0):
        save_particles(
            "save/bullet-" + str(nSteps) + ".h5",
            elements, to_save_particles
        )
    if (nSteps % 10 == 0):
        evaluate_search_radius(
            kBuf=1.5,
            natives=VD.inner_elements,
            aliens=VD.outer_elements,
            neibs=nlist,
            threads=threads
        )
        load.find(useful, elapsed)
        VD.update(load.all(network))
        VD.exchange()
        nlist.update()
    dt = stepper.get_step(VD, CFL=0.5)
    stepper.make_step(VD, nlist, dt)
    time += dt
    nSteps += 1
```

- Каждые 100 шагов записываем выходной файл
- Каждые 1000 шагов записываем полный набор данных для перезапуска
- Каждые 10 шагов обновляем декомпозицию в соответствии с нагрузкой и перестраиваем список соседей
- Вычисляем шаг по времени из критерия Куранта и интегрируем

Постпроцессинг