МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ» (НОВОСИБИРСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ, НГУ)

Факультет: **ФИЗИЧЕСКИЙ**

Кафедра: **ФИЗИКИ ПЛАЗМЫ**

Направление подготовки: **03.03.02 ФИЗИКА**

Образовательная программа: **БАКАЛАВРИАТ**

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Охотникова Никиты Викторовича

Тема работы: Разработка версии квазистатического кода для моделирования кильватерного ускорения LCODE с произвольной геометрией.

«К защите допущена»

И.О. заведующего кафедрой
к.ф.-м.н.
с.н.с., ИЯФ СО РАН

Сковородин Д.И./_____
(подпись, МП)

«_____»_____ 2022 г.

Научный руководитель
инженер-исследователь, ИЯФ СО РАН

Горн А. А./_____
(подпись, МП)

«_____»_____ 2022 г.

Дата защиты: «_____»_____ 2022 г.

Новосибирск — 2022 г.

Содержание

Введение

Для получения больших энергий ускоренных частиц в ускорителях требуются все большие масштабы, однако постройка ускорителей подобных размеров связана с рядом трудностей. Огромные временные и финансовые затраты, а так же проблема размещения гигантской установки заставляют задуматься об альтернативах. В последнее время стали развиваться новые методы ускорения, один из которых — плазменное кильватерное ускорение.

Кильватерное ускорение основано на возбуждении плазменной кильватерной волны. В данном методе пучок заряженных частиц или лазерный импульс, называемый драйвером, расталкивает лёгкие электроны плазмы, почти не оказывая влияния на тяжёлые ионы остаются, неподвижные из-за своей большой массы. Таким образом, заряды разделяются, генерируя в плазме электромагнитные поля, в том числе продольное электрическое поле, ускоряющее сгусток электронов, или витнесс [1]. При этом максимальный ускоряющий градиент намного больше достигаемого в других типах ускорителей сравнимых размеров. При той же энергии ускоренных электронов плазменные ускорители имеют на два порядка меньший размер по сравнению с ускорителями традиционного типа (Рис. 1). Это связано с тем, что в традиционных ускорителях максимальное ускоряющее поле ограничено величиной пробоя в резонаторе, а плазма выдерживает гораздо большие поля, порядка поля опрокидывания плазменной волны, при котором происходит полное разделение зарядов, $E_0 = mc\omega_p/e$, где $\omega_p = \sqrt{4\pi n_0 e^2/m}$ — плазменная частота, m — масса электрона, c — скорость света, e — элементарный заряд, n_0 — плотность невозмущённой плазмы. Данный способ позволяет получить градиенты ускорения порядка ГэВ/м при плотности плазмы $n_0 = 10^{15}$ см$^{-3}$. В эксперименте FACET был получено ускоряющее поле 4.4 ГВ/м [2], в то время как проектируемый мульти-ТэВный ускоритель CLIC [3] будет оперировать с ускоряющими полями до 100 МВ/м.

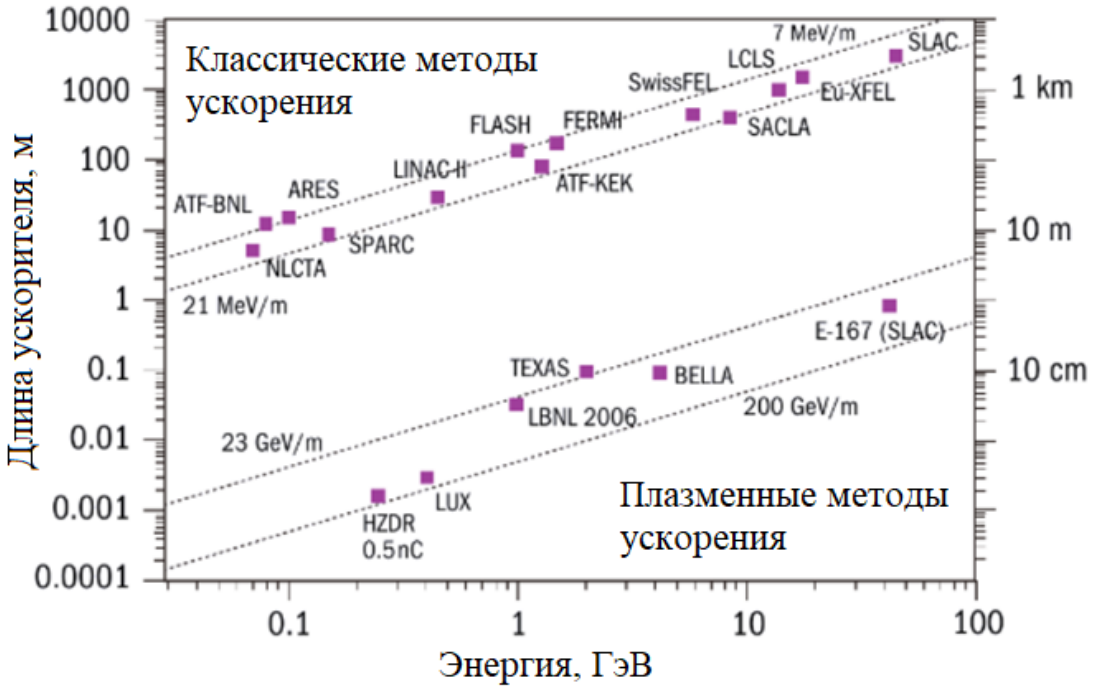

Рисунок 1: Сравнение классических и плазменных методов ускорения

Расчёт полей кильватерной волны крайне сложен, решить систему уравнений Максвелла аналитически возможно только при наличии различных упрощающих предположений. Это связано с нелинейностью процессов. В связи с этим для успешного проектирования эксперимента необходимо прибегать в численному моделированию.Для моделирования кильватерного ускорения в ИЯФ СО РАН был разработан программный комплекс LCODE [4], позволяющий вычислять распределение зарядов, полей и другие необходимые эксперименту характеристики посредством решения уравнений Максвелла методом частиц в ячейках. Код был написан на языке C, разрабатывался долгие годы большим количеством людей, каждый из которых имел свой стиль написания кода. Это затрудняет понимание, а следовательно дальнейшую разработку и поддержку программы. Код имеет ограниченный функционал, что затрудняет обработку данных моделирования и проведение параметрических сканирований. Параметры LCODE задаются текстовым документом и передаются исполняемому файлу, что не предполагает их изменение в процессе моделирования. В то же время

сложные эксперименты по кильватерному ускорению, одним из которых является AWAKE [5], могут предполагать наличие нескольких плазменных секций, областей неоднородности плазмы и инжекции электронного витнесса, что предполагает их моделирование с разным временным шагом. С другой стороны, цилиндрическая геометрия позволяет уменьшить ресурсоёмкость моделирования, хотя её применимость и ограничена. Это делает невозможным моделирование некоторых процессов. Например, для эксперимента AWAKE была показана необходимость внеосевой инжекции пучка электронов [6], что выходит за рамки применимости цилиндрической геометрии. Решением этой проблемы является моделирование разных этапов ускорения с помощью разных инструментов. Стандартизация входных и выходных данных в этом случае значительно упростила бы этот процесс.

Целью данной работы является написание Python версии LCODE для расширения его функционала и области применимости, а также будущей интеграции с трёхмерной версией. Возможность изменения параметров во время моделирования упрощает исследование комплексных систем, позволяет выполнять параметрические сканирования и сложные многомерные оптимизации параметров средствами кода, а также — динамически менять геометрию задачи. Это сделает возможным проведение в рамках одного запуска подробных, но медленных расчётов сложных процессов, и менее подробных, но быстрых моделирований процессов в однородной плазме. Применение стандартных форматов файлов упрощает взаимодействие с другими инструментами. Использование абстрактных конструкций языка, а также методов ООП повышает гибкость и читаемость кода, что играет ключевую роль в его дальнейшем улучшении и взаимодействии с другими научными командами. При этом JIT-компиляции средствами Numba [7] помогает не сильно проиграть в скорости выполнения моделирования.

1 Описание LCODE

1.1 Математическая модель

Программа LCODE использует ряд приближений для упрощения решаемой системы уравнений Максвелла и уравнений движения макрочастиц. Код использует предположение об осевой симметрии, не учитывает столкновения частиц и использует квазистатическое приближение [8, 9].

В квазистатическом приближении считается, что ультрарелятивистский пучок эволюционирует в сопутствующем ему окне намного медленнее плазмы, то есть остаётся статичным при вычислении эволюции плазмы. Положение вдоль такого окна задаётся координатой $\xi = z - ct$. В рамках квазистатического приближения счётное окно движется со скоростью света, поэтому информация может передаваться только в сторону уменьшения ξ. Принцип расчёта одного окна моделирования в рамках квазистатического приближения показан на Рис. 1.1. Пучок летит направо, цветом выделен уже посчитанный участок. В цилиндрической системе координат система

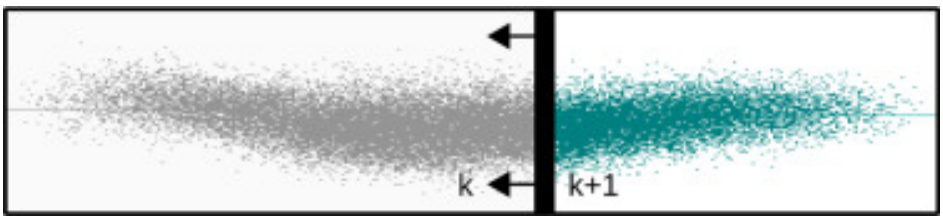

Рисунок 1.1: Счётное окно моделирования отклика плазмы, индекс k отвечает за номер вычисляемого слоя вдоль ξ, стрелочками показано направление вычисления плазменного отклика

уравнений Максвелла принимает вид

$$\begin{cases} \nabla \times \vec{B} = \frac{4\pi}{c}(\vec{j} + \vec{j_b}) + \frac{1}{c}\frac{\partial \vec{E}}{\partial t}, \\ \nabla \times \vec{E} = -\frac{1}{c}\frac{\partial \vec{B}}{\partial t}, \\ \nabla \vec{E} = 4\pi(\rho + \rho_b), \\ \nabla \vec{B} = 0. \end{cases} \Leftrightarrow \begin{cases} \frac{1}{r}\frac{\partial}{\partial r} r E_r = 4\pi(\rho + \rho_b) - \frac{\partial E_z}{\partial \xi}, \\ \frac{1}{r}\frac{\partial}{\partial r} r B_r = -\frac{\partial B_z}{\partial \xi}, \\ \frac{1}{r}\frac{\partial}{\partial r} r\left(E_r - B_\phi\right) = 4\pi(\rho - \frac{1}{c} j_z), \\ \frac{\partial E_z}{\partial r} = \frac{4\pi}{c} j_r, \\ \frac{\partial B_z}{\partial r} = -\frac{4\pi}{c} j_\phi, \\ E_\phi = -B_r \end{cases} \tag{1.1}$$

Переход к сопутствующей координате здесь осуществляется следующим образом:

$$\frac{\partial}{\partial z} = -\frac{1}{c}\frac{\partial}{\partial t} = \frac{\partial}{\partial \xi}. \tag{1.2}$$

Поля E_z, B_z находятся численным интегрированием, остальные — методом прогонки.

1.2 Описание циклов

Разобьём наше моделирование на отдельные шаги по времени, на которых будет выполняться движение части плазмы и пучка. Квазистатическое приближение позволяет считать пучок стационарным во время вычисле-

ния движения частиц плазмы. Разобьём вычислительное окно на равные промежутки по ξ. Так как пучок является ультрарелятивистским, то информация передаётся в сторону уменьшения ξ.

Появляются два основных цикла:внешнего по времени t и внутреннего по продольной координате ξ в сторону её уменьшения.

Внешний цикл отвечает за движение частиц пучка и состоит из: разложения частиц пучка на сеточную плотность, цикла по ξ, в котором считается отклик плазмы, и движения частиц пучка плазменными полями.

Цикл по ξ состоит из: разложения частиц плазмы на сетку, расчёт полей, движение частиц плазмы. При этом пучок считаем стационарным.

1.3 Ввод и вывод данных

Для взаимодействия с пользователем LCODE использует конфигурационный файл в качестве входных данных и файлы диагностик в качестве выходных. Конфигурационный файл это текстовый файл, состоящий из набора параметров и их значений. С его помощью можно задать начальную форму пучка с помощью нескольких шаблонов, но это не всегда применимо. Для более сложных форм, необходимо генерировать пучок самостоятельно в файл со специальной структурой. Хотя и возможность генерировать пучок сторонним скриптом является мощным инструментом, но это не всегда необходимо, а в некоторых ситуациях неудобно и замедляет проектирование эксперимента.

1.4 Быстродействие

Благодаря принятым упрощениям: квазистатическое приближение, ультрарелятивистский пучок, цилиндрическая геометрия — LCODE работает очень быстро. В то же время особое внимание к частицам с низкой энергией

и тонкой структуре поля мер позволяет коду работать стабильно [9].

Результаты моделирования сходятся с экспериментом и другими кодами. Благодаря своему быстродействию и стабильности код активно используется для проектирования многих экспериментов по кильватерному ускорению.

2 Python-версия

Разработке модуля, вычисляющего отклик плазмы на пучок — плазменного решателя, а также модуля, отвечающего за движение частиц — пучкового толкателя выполнил Николай Марчук. Передо мной стояла цель выполнить оставшуюся работу для создания работоспособного кода: объединить данные модули под цельным интерфейсом, реализовать диагностики и по возможности реализовать оптимизацию программы для её ускорения.

2.1 Оболочка

Главным объектом для пользователя является объект класса Моделирование и его метода step(n). Внутри он скрывает временной цикл, где на каждом шаге создаётся новая плазма и обрабатывается данный временной шаг. Обработка временного шага подразумевает выделение памяти под диагностику, цикл по ξ, сброс диагностик на диск и передача пучка на следующий шаг. Внутри цикла по ξ последовательно происходит получение сеточной плотности частиц, вызов плазменного решателя, пучкового толкателя, а также обработку диагностик.

2.2 Запуск

Код оформлен в виде библиотеки языка Python, предоставляющую доступ к классу моделирования и его методу step(n), который выполняет n временных шагов с данными параметрами. Данный интерфейс даёт воз-

можность между вызовами метода step менять параметры моделирования, а также обрабатывать промежуточные данные, вызывать любые сторонние программы и продолжать счёт.

Это позволяет моделировать комплексные эксперименты, требующие разных подходов на разных стадиях ускорения. Например, в эксперименте AWAKE это позволит в одном скрипте посчитать сначала самомодуляцию с помощью разрабатываемого кода, затем инжекцию электронов с помощью трёхмерного года без предположения о цилиндрической симметрии, и затем ускорение снова с помощью разрабатываемого кода.

2.3 Генератор пучка

Для удобства работы был создан гибкий модуль генерации пучка. Раньше приходилось пользоваться небольшим количеством шаблонов или генерировать пучок с помощью отдельной программы. Данный модуль позволяет сгенерировать произвольный пучок прямо внутри скрипта запуска. Также данный модуль совместим с LCODE 3d, что упростит взаимодействие в моделированиях с непостоянной геометрией.

2.4 Диагностики

Для анализа результатов моделирования был написан диагностический модуль с набором первичных диагностик: электрические и магнитные поля, поведение частиц пучка и плазмы в окне моделирования, а также возможностью добавить свою диагностику.

Предыдущая версия кода для каждой диагностики на каждом временном шаге создавала отдельный бинарный или текстовый файл, из-за чего после выполнения моделирования в рабочей директории могли появиться сотни файлов. Это может вызвать путаницу, а также замедляет время

обработки файлов ввиду их большого количества и относительно медленного переключения между файлами в операционных системах. Также при передаче файлов другой организации возникала необходимость отдельно разъяснять их структуру из-за отсутствия соглашения о ней.

Для нового диагностического модуля был выбран иерархический бинарный формат данных HDF5 ввиду его распространённости в научном сообществе. Данный формат является самоописываемым, то есть хранит свою структуру внутри файла. Он позволяет хранить все диагностики в одном файле и работать с ними с помощью маршрутов аналогичных файловой системе. Формат позволяет работать с одним файлом в парралельном режиме, что даёт возможность хранить все шаги по времени в одном файле.

Так как все диагностики представляют собой список частиц или данные на сетке окна моделирования, то был выбран стандарт openPMD (open standard for particle-mesh data files). Данный стандарт описывает, как должна выглядеть общая структура HDF5 файла, а также некоторые имена метаданных, таких как размерность величины, коэффициент для перевода в СИ и другие. Данный стандарт получил распространение среди PIC кодов, моделирующих кильватерное ускорение, таких как FBPIC, HiPACE++, Osiris. Это упрощает сравнение результатов моделирования между ними.

Как итог, результатом моделирования является HDF5 файл, который легко обрабатывать и делиться с другими командами благодаря стандарту openPMD. Данные сохраняются в безразмерных единицах [10].

На данный момент в коде реализованы только базовые диагностики: электромагнитные поля, частицы плазмы и частицы пучка в окне моделирования. Однако на их основе можно посчитать более комплексные величины, такие как плотность электронов или кильватерный потенциал.

На рис. 2.1 изображён портрет пучка, полученный с помощью диагностики.

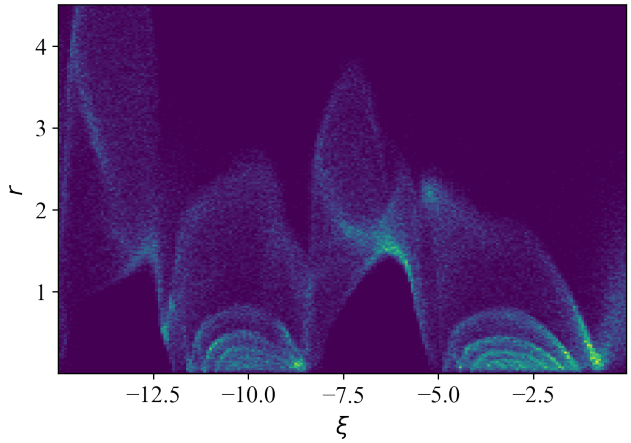

Рисунок 2.1: Портрет пучка, полученный с помощью диагностики

На рис. 2.2 изображено продольное поле на оси, полученное с помощью диагностики.

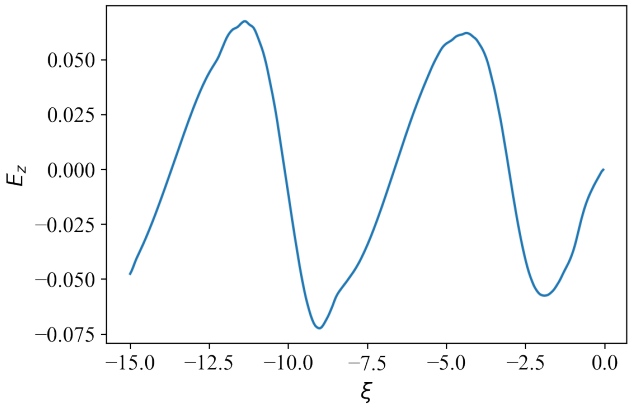

Рисунок 2.2: Продольное электрическое поле на оси при $r = 0$

На рис. 2.3 изображено движение плазменных частиц вдоль окна, полученное с помощью диагностики.

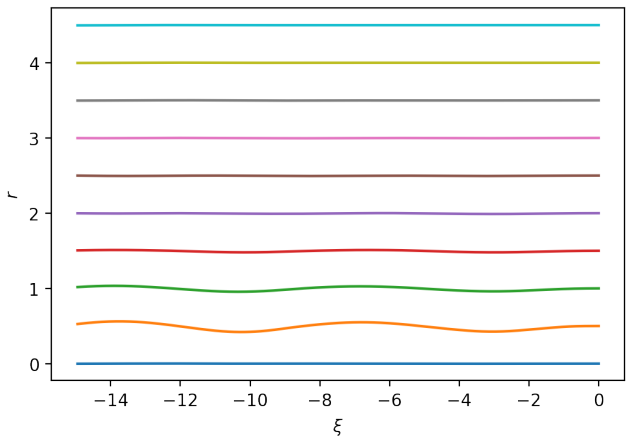

Рисунок 2.3: Движение плазменных частиц

2.5 Оптимизация вычислений

В ходе работы вычислительному модулю приходиться обрабатывать огромные потоки данных. Поэтому оптимизация вычислений для ускорения их обработки играет одну из ключевых ролей при использовании кода.

Python является интерпретируемым языком программирования. Это очень удобно при написании и отладке программы, но сильно увеличивает время выполнения по сравнению с компилируемыми языками.

Библиотека NumPy частично решает проблему медлительности, предоставляя многомерные массивы, а также функции и операторы, эффективно работающие с массивами. Также для решения этой проблемы были использованы методы Just-in-time(JIT) компиляции средствами библиотеки Numba. Данная библиотека позволяет переводить отдельные функции при первом их вызове в быстрый машинный код, что заметно ускоряет их при последующих вызовах. При этом использование NumPy и Numba позволяет сохранить лаконичность кода.

В квазистатическом приближении обработка вычислетельного окна идёт послойно от головы пучка к его хвосту. Слой не будет изменяться после его вычисления на данном временном шаге, и он готов к следующему шагу по времени. Следовательно, при вычислении слоя k в данном шаге по времени

мы можем работать со слоями после $k+1$ на следующем шаге по времени (Рис. 2.4). [9]

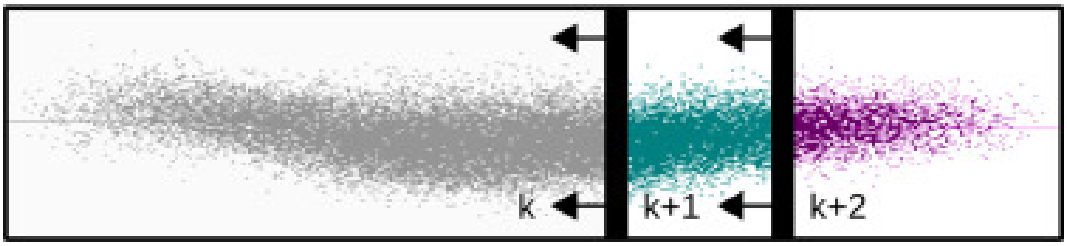

Рисунок 2.4: Принцип параллельного вычисления

Для передачи слоя был использован интерфейс MPI и библиотека mpi4py. Интерфейс MPI де-факто стал стандартом для систем с распределённой памятью. Он позволяет обмениваться сообщениями с произвольной информацией между процессами выполняющими одну задачу.

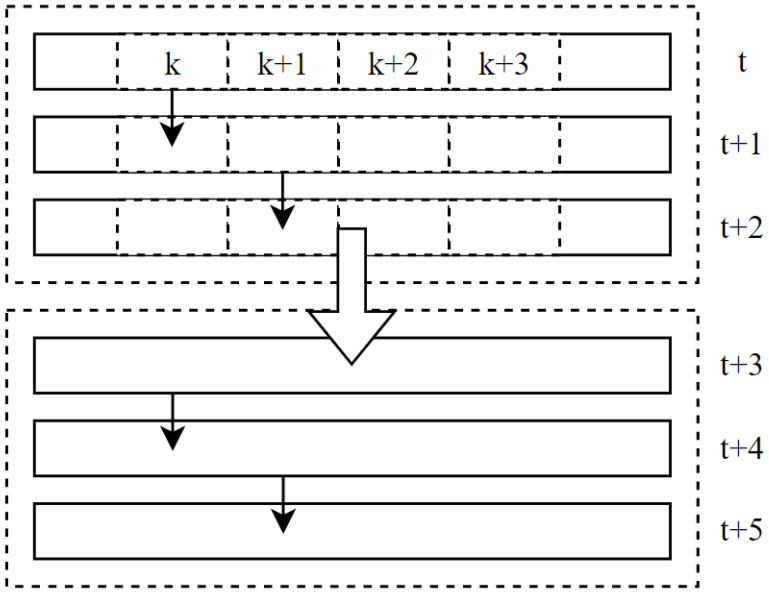

Рисунок 2.5: Пример работы параллельного вычисления для $n = 3$.

В начале моделирования запускается n процессов с кодом, каждый из которых ждёт свой слой. После вычисления своего слоя он передаёт его следующему процессу. После того, как каждый из n процессов вычислил все слои. Весь пучок передаётся от процесса n к процессу 1 и данный алгоритм повторяется, пока не закончатся все шаги по времени. Такой подход позволяет уменьшить время моделирования в n раз.

На Рис. 2.5 прдставлен пример работы алгоритма для $n = 3$. Вычисление идёт справо налево. На слое t посчитаны слои от правого края до k включительно, на $t+1$ — до $k+1$, на $t+2$ — до $k+2$. После того, как слой $t+2$ закончит вычисление всех слоёв, он передаст всё окно на процессу 1.

3 Тестирование

3.1 Скорость работы

Одно из главных опасений при переходе на Python было связано со временем работы программы. Были предприняты шаги для его уменьшения описанные в главе 2.5. Для проверки времени была взята конфигурация Test 1 из работы [11], но с разными длинами окна моделирования. Сравнение проводилось для старого кода на си и разрабатываемого кода. Отношение времени моделирования кодом на Python ко времени моделирования кодом на C представлено на Рис. 3.1.

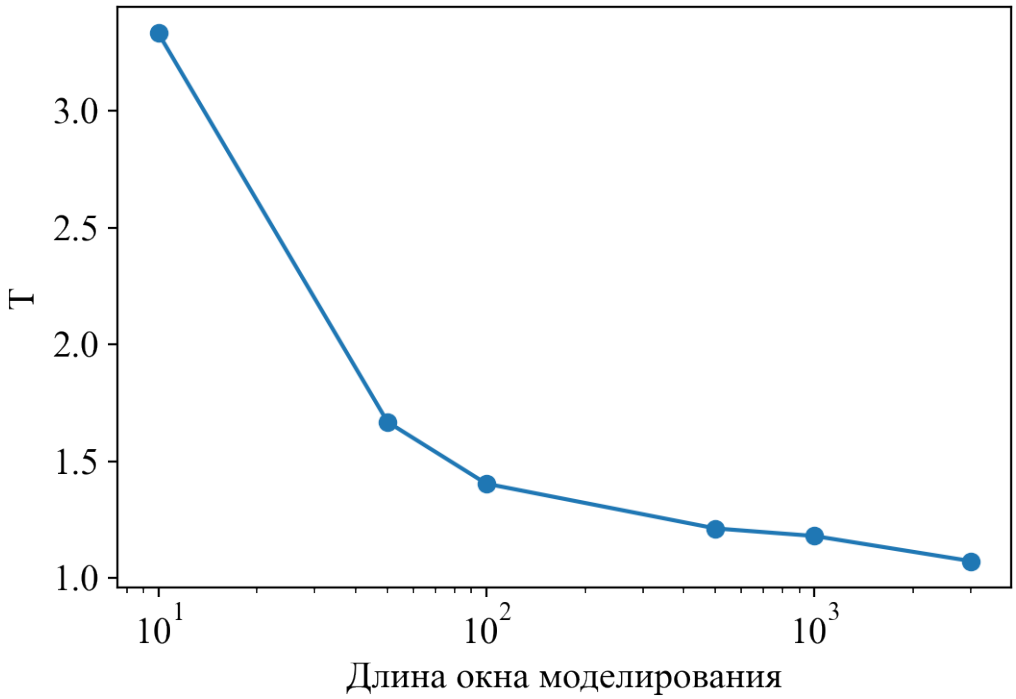

Рисунок 3.1: Сравнение времени работы для разных длин окон
T — отношение времён моделирования

3.2 Тест на стабильность

Для проверки кодов моделирующих кильватерное ускорение был разработан Test 1. [11–13] В данном тесте запускается очень короткий пучок в плазму. При этом вдоль ξ возникает кильватерное поле постоянной амплитуды и частоты. Величина амплитуды поля составляет $0.725E_0$. Чем дольше код удерживает данное поле, тем ожидается стабильнее его работа.

Результат проведённого теста показан на Рис. 3.2. Оранжевая линия это теоретический результат, синяя — огибающая поля для кода. Как видно, на протяжении всего окна разница не превышает 10%, что считается хорошим результатом.

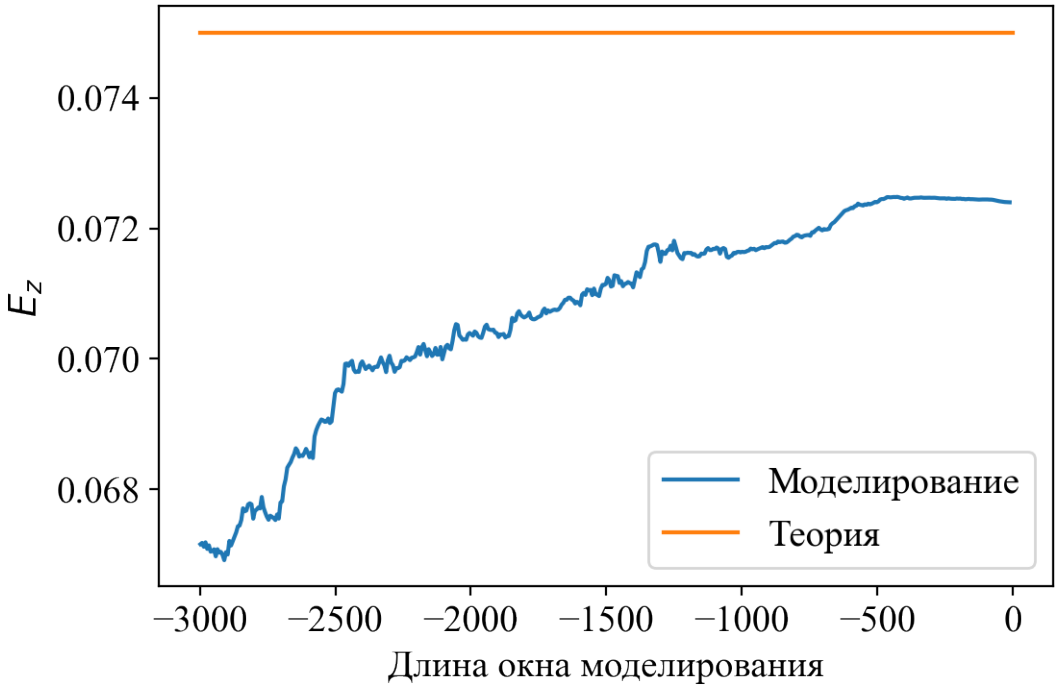

Рисунок 3.2: Результат Test 1 для кода

3.3 Самомодуляция

При попадании в плазму пучок создаёт в ней поперечную кильватерную волну, которая разрушает его на микросгустки. Данный эффект называет-

ся самомодуляцией пучка [14].

На рис 3.3 изображён пучок до столкновения с плазмой (сверху) и через некоторое время после столкновения (снизу). Хорошо видно, что он разделился на микросгутски.

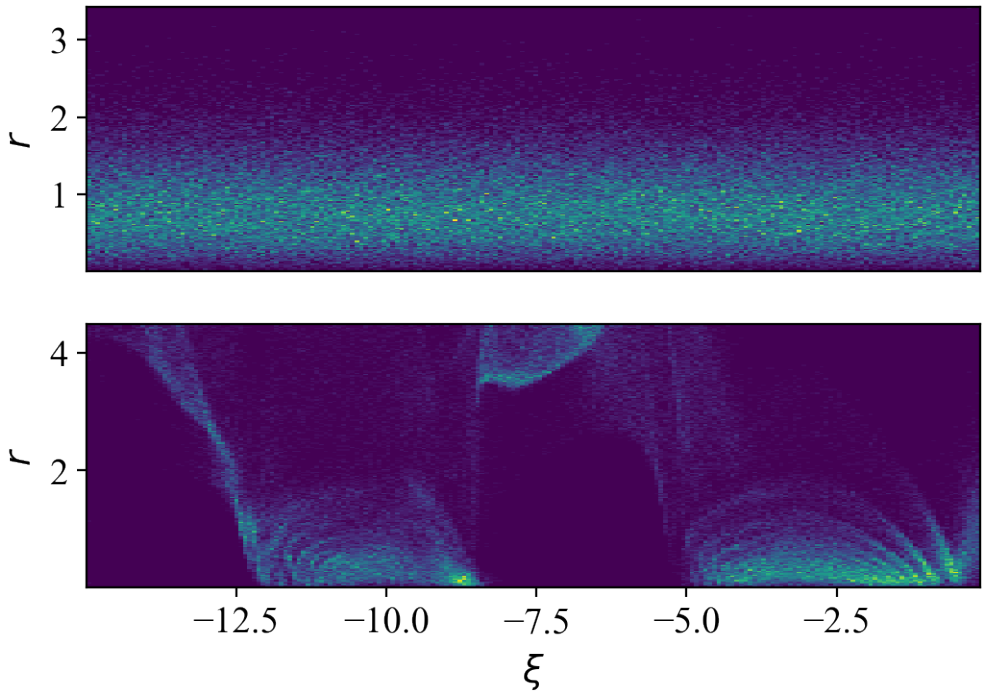

Рисунок 3.3: Самомодуляция пучка в плазме

Так как код смог промоделировать достаточно сложный физический процесс, то он готов к использованию.

Заключение

Разработан рабочий параллельный двухмерный квазистатический код для моделирования кильватерного ускорения. Пользовательский интерфейс позволяет менять параметры моделирования во время выполнения. Генератор пучка даёт возможность создавать сложные конфигурации прямо из скрипта запуска. Диагностические файлы эффективно сохраняются по общепринятому стандарту для PIC кодов openPMD. Параллельность по времени осуществляется за счёт взаимодействия между процессами через интерфейс MPI.

Параллельные вычисления в совокупности с методами Numba и NumPy позволили приблизиться к скорости кода на си для длинных окон. Время, затраченное на JIT-компиляцию, незначительно по сравнению со временем счёта реальных задач.

Код проверен на стабильность на задаче, аналитическое решение которой известно. Выполнено сравнение времени выполнения. Также посчитан эффект самомодуляции — один из сложных, но хорошо наблюдаемых эффектов в задачах кильватерного ускорения.

Список использованных источников

[1] Joshi Chandrashekhar. Plasma accelerators // Proc. of the XIIIth Int'l. Conf. on High Energy Accelerators, Novosibirsk, USSR, Augusi. — Vol. 6. — 1986.

[2] High-efficiency acceleration of an electron beam in a plasma wakefield accelerator / M Litos, E Adli, W An et al. // Nature. — 2014. — Vol. 515, no. 7525. — P. 92–95.

[3] A Multi-TeV linear collider based on CLIC technology: CLIC Conceptual Design Report : Rep. / SLAC National Accelerator Lab., Menlo Park, CA (United States) ; Executor: Markus Aicheler, P Burrows, M Draper et al. : 2014.

[4] Sosedkin A.P., Lotov K.V. LCODE: A parallel quasistatic code for computationally heavy problems of plasma wakefield acceleration // Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. — 2016. — Vol. 829. — P. 350–352. — 2nd European Advanced Accelerator Concepts Workshop - EAAC 2015. URL: https://www.sciencedirect.com/science/article/pii/S0168900215016034.

[5] AWAKE, the advanced proton driven plasma wakefield acceleration experiment at CERN / Edda Gschwendtner, Erik Adli, L Amorim et al. // Nuclear Instruments and Methods in Physics Research Section

A: Accelerators, Spectrometers, Detectors and Associated Equipment. — 2016. — Vol. 829. — P. 76–82.

[6] Simulations of Electron-Proton Beam Interaction before Plasma in the AWAKE Experiment : Rep. ; Executor: Ulrich Dorda, Ralph Aßmann, Alexey Petrenko et al. : 2015.

[7] Lam Siu Kwan, Pitrou Antoine, Seibert Stanley. Numba: A llvm-based python jit compiler // Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC. — 2015. — P. 1–6.

[8] Lotov K. V. Simulation of ultrarelativistic beam dynamics in plasma wake-field accelerator // Physics of Plasmas. — 1998. — Vol. 5, no. 3. — P. 785–791. — https://doi.org/10.1063/1.872765.

[9] Sosedkin AP, Lotov KV. LCODE: a parallel quasistatic code for computationally heavy problems of plasma wakefield acceleration // Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. — 2016. — Vol. 829. — P. 350–352.

[10] K.V. Lotov A.P. Sosedkin. — LCODE user manual, 2020. — URL: https://lcode.info/.

[11] Lotov K.V. AWAKE-related benchmarking tests for simulation codes // Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. — 2018. — Vol. 909. — P. 446–449. — 3rd European Advanced Accelerator Concepts workshop (EAAC2017). URL: https://www.sciencedirect.com/science/article/pii/S0168900217314572.

[12] Chen Pisin. Grand disruption: A possible final focusing mechanism for

linear colliders // Part. Accel. — 1986. — Vol. 20, no. SLAC-PUB-3823. — P. 171–182.

[13] Path to AWAKE: Evolution of the concept / A. Caldwell, E. Adli, L. Amorim et al. // Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. — 2016. — Vol. 829. — P. 3–16. — 2nd European Advanced Accelerator Concepts Workshop - EAAC 2015. URL: https://www.sciencedirect.com/science/article/pii/S0168900215016307.

[14] Kumar Naveen, Pukhov Alexander, Lotov Konstantin. Self-modulation instability of a long proton bunch in plasmas // Physical review letters. — 2010. — Vol. 104, no. 25. — P. 255003.