**Метод частиц в ячейках (Particle-in-Cell, PIC)**

**Замечание 1**. Решено сделать описание для сетки, в которой все компоненты поля и токов расположены в одних точках. Потом ее можно будет легко адаптировать для более широко используемой сетки со сдвигами (сетка Йи, метод FDTD).

**Замечание 2**. Рассматривается двумерный случай. Это значит, что координаты частиц являются двумерными векторами. Но скорости частиц и компоненты поля все равно должны быть 3-мерными.

**Замечание 3**. Далее скалярные величины обозначаются обычным шрифтом, а векторные в трехмерном пространстве полужирным. Например, радиус-вектор   
 выделяется полужирным, а его евклидова норма, т.е. длина, обычным шрифтом: .

Рассматривается трехмерная область в виде прямоугольного параллелепипеда , которую в дальнейшем будем называть ***расчетной областью*** (значения , и др. заданы и не меняются).

В каждой точке расчетной области определены 3-компонентное ***электрическое поле***, которое будем обозначать , и 3-компонентное ***магнитное поле*** , . Пара векторных полей  называется ***электромагнитным полем***.

Для представления значений поля расчетная область покрывается равномерной сеткой, содержащей узлов ( ячеек) по соответствующим размерностям. Шаги сетки равны , . Для индексирования узлов сетки используются естественные трехмерные индексы . В каждом узле сетки определены ***сеточные значения поля***  и и плотности тока . Узел для всех компонент поля и плотности тока соответствует точке физического пространства (см. замечание в начале документа).

Аналитическое решение возможно лишь для малого числа частных случаев. Численное моделирование происходит с заданным шагом по времени , т.е. в дискретные моменты 0, , , , … , последовательность завершается при превышении . При этом все данные о полях и частицах хранятся только на текущий момент времени и обновляются на соответствующих этапах метода.

***Интерполяция поля***

***Задача интерполяции поля*** (*field interpolation*) состоит в определении значений поля в произвольной точке расчетной области по заданному набору сеточных значений. Интерполяцию можно выполнять различными способами, которые отличаются количеством используемых при интерполяции сеточных значений и способом выбора коэффициентов, а также порядком (характером зависимости погрешности интерполяции от величины шагов сетки). Для более точной интерполяции нужно использовать большее количество сеточных значений, и данная процедура является более затратной с вычислительной точки зрения.

В качестве первого метода интерполяции будем рассматривать ***линейную интерполяцию***. В данном методе интерполяция осуществляется по 4 ближайшим сеточным значениям, лежащим в вершинах одной ячейки сетки (которая является прямым параллелепипедом).

По заданным координатам , в которых необходимо вычислить значение поля, вычисляется индекс ячейки сетки, в которой лежит данная точка физического пространства: , , (под операцией понимается нижняя целая часть). Интерполяция производится с использованием сеточных значений с индексами .

Интерполированное значение является линейной комбинацией указанных сеточных значений с неотрицательными коэффициентами, в сумме дающими 1 (такая линейная комбинация называется выпуклой комбинацией). Для вычисления коэффициентов перейдем к внутренней системе координат, связанной с используемой ячейкой. Переход от исходных координат в новую систему осуществляется по формулам: , . Таким образом, в новой системе координат точка, соответствующая узлу , имеет координаты , а точка, соответствующая узлу имеет координаты . Соответственно, все точки ячейки в новой системе имеют координаты от до .

Интерполированное значение поля вычисляется по формуле:

Аналогично для других компонент поля.

Для проверки корректности реализации можно выбрать линейную функцию и заполнить сеточные значения поля в соответствии с выбранной функцией: и т.д. В этом случае метод линейной интерполяции должен давать точные значения, с точностью до погрешности компьютерной арифметики с плавающей запятой. Таким образом, нужно проверить, что для случайно выбранных точек расчетной области интерполированные значения равны с учетом погрешности компьютерной арифметики.

***Интегрирование уравнений движения частиц***

Рассматривается задача моделирования динамики заряженных частиц в электромагнитном поле. Заряженная частица определяется следующими параметрами: переменные положение (2-мерный вектор) и скорость (3-мерный вектор), постоянные масса и заряд (скаляры). В электромагнитном поле (, ) (3-мерные векторы) частица движется под действием силы Лоренца согласно второму закону Ньютона в релятивистской форме (уравнения приведены в системе СГС, которую обычно используют в физике плазмы):

( – скорость света в вакууме; – импульс частицы, в релятивистском случае скорость и импульс однозначно связаны соотношением ). Здесь и далее в выражениях типа , где левая часть является 2-мерным вектором, а правая часть – 3-мерным, подразумевается, что компоненты **,** используются для обновления **,** , а хранится и вычисляется, но просто не используется для изменения координат (она нужна будет при вычислении токов).

На каждом шаге метода по положению и скорости частицы в текущий момент времени определяются положение и скорость в следующий момент времени. Методы интегрирования уравнений движения отличаются способом пересчета новых положения и скорости по текущим. *Метод Бориса* (Boris) является одним из наиболее широко используемых при моделировании плазмы.

Значения поля (, ) и шаг по времени заданы (получаются в результате интерполяции). Введем обозначения ,, перепишем уравнения движения в виде:

Для удобства записи будем использовать верхние индексы: (old) относится к текущим значениям, (new) – к новым, вычисляемым при выполнении метода Бориса. Обозначим –известное значение импульса и – неизвестное новое значение, соответствующим образом обозначим , , , . Запишем разностное соотношение для первого из уравнений движения, используя:

Перейдем к новым переменным , , сделав замену , . Таким образом, известной является переменная (она вычисляется только по текущим данным), значение может быть найдено из разностного соотношения, принимающего вид:

Это соотношение можно записать как линейную систему из 3 уравнений с 3 неизвестными, так будет невырожденная матрица и поэтому существует единственное решение . Так делать в принципе можно и это будет работать, но на практике обычно вместо этого пользуются приближенной схемой, основанной на физике движения частиц в электромагнитном поле – схеме Бориса (объяснение причин весьма сложное, не будем пока в них вдаваться).В этом случае после вычисления вычисляются векторы:

(Т.е. вместо того, чтобы честно решить систему 3х3 выше, мы нашли приближенное решение ).

По найденному вычисляются , .Новое значение скорости . Новое положение частицы определяется из соотношения:

Входными данными задачи являются: данные о частице (положение, скорость, масса, заряд), значения поля (, ), шаг по времени. Выходными данными являются обновленные данные о частице. Проверка корректности осуществляется путем численного решения задачи с известным аналитическим решением (движение электрона в статическом поле) и сравнением результатов моделирования с теоретически ожидаемыми.

В PIC-кодах не всегда возможно промоделировать все частицы, это требует огромных вычислительных затрат. Поэтому частицы объединяются в макрочастицы. Макрочастица представляет собой одну большую частицу, которая «объединяет» несколько частиц одного типа. Если макрочастица объединяет электронов, то ее масса и заряд в раз больше массы и заряда электрона. При этом , называемое весом или фактором частицы, не обязательно должно быть целым.

Для проверки корректности предлагаются 2 задачи с известным аналитическим решением. Описания тестов приводятся для оси и положительного направления. Для начала можно сделать только в том варианте, который дан в описании. После того, как тесты в исходной постановке будет отлажены, их нужно проводить вдоль обеих осей и обоих направлений. Во всех тестах необходимо проверять не точное совпадение с аналитическим решением, а то, что относительная погрешность достаточно мала. При этом погрешность складывается из аппроксимационной ошибки и погрешности вычислений. Пока шаг по времени не становится достаточно малым, аппроксимационная ошибка вносит вклад больше, чем ошибка округления.

Тест «Релятивистское ускорение в статическом поле». Взять одну частицу, выбрать массу и заряд (например, взять значения для электрона), начальное положение и скорость нулевые. Произвольно, но «с умом» выбрать значение , в качестве значений поля на всех шагах моделирования использовать , . Выбрать число шагов по времени (например, 100 или больше) и взять . Произвести итераций моделирования. Аналитический результат: по окончании моделирования положение частицы , импульс .

Тест «Осцилляция в статическом магнитном поле». Взять одну частицу, выбрать массу и заряд , начальное положение нулевое. Произвольно выбрать константу , взять начальный импульс . Произвольно выбрать значение , в качестве значений поля на всех шагах моделирования использовать , . Выбрать число шагов по времени (например, 100) и взять . Произвести итераций моделирования. Аналитический результат: по окончании моделирования положение частицы , импульс .

***Вычисление (взвешивание) токов***

***Задача взвешивания токов*** *(current deposition)* состоит в определении сеточных значений плотности тока, создаваемого заряженной частицей с заданным положением , скоростью и зарядом . Название «взвешивание» обусловлено тем, что аналитически ток создается лишь в точке , но при численном решении он распределяется по ближайшим сеточным значениям с подобранными в зависимости от координат весами. Взвешивание можно выполнять различными способами, которые отличаются количеством сеточных значений плотности тока, в которые вносит вклад каждая частица, способом выбора весов, а также порядком (характером зависимости погрешности взвешивания от величины шагов сетки). Взвешивание во многом является обратной операцией к интерполяции.

В качестве первого метода взвешивания будем рассматривать ***взвешивание первого порядка***. В данном методе взвешивания частица вносит вклад в плотность тока в 4 ближайших узлах сетки, лежащих в вершинах одной ячейки сетки (которая является прямоугольником).

Перед началом выполнения процедуры взвешивания все сеточные значения плотности тока инициализируются нулем. Затем происходит последовательное вычисление вкладов всех частиц с накоплением (суммированием) результатов. В дальнейшем рассматривается процедура для одной частицы.

По заданным координатам частицы вычисляется индекс ячейки сетки, в которой лежит данная точка физического пространства: , (под операцией понимается нижняя целая часть). Во время взвешивания вносится вклад в сеточные значения с индексами , остальные сеточные значения плотности тока при обработке данной частицы не меняются.

Аналитически движение частицы со скоростью и зарядом создает ток . Соответствующая плотность тока . Во время взвешивания данное значение распределяется по сеточным значениям . Вклад в каждое из указанных сеточных значений равен , умноженному на коэффициент от 0 до 1, сумма всех коэффициентов равна 1. Таким образом, суммарный вклад во все сеточные значения равен .

Для вычисления коэффициентов перейдем к внутренней системе координат, связанной с используемой ячейкой. Переход от исходных координат в новую систему осуществляется по формулам: , . Таким образом, в новой системе координат точка, соответствующая узлу , имеет координаты , а точка, соответствующая узлу имеет координаты . Соответственно, все точки ячейки в новой системе имеют координаты от до .

Вклад в сеточные значения плотности тока определяется следующим образом:

***Интегрирование уравнений поля***

Динамика электромагнитного поля подчиняется системе уравнений Максвелла (запись в системе единиц СГС):

***FDTD*** (Finite-Difference Time-Domain) является одним из наиболее широко используемых методов численного решения уравнений Максвелла. Метод относится к типу конечно-разностных и основан на замене производных на их конечно-разностные сеточные аппроксимации. В литературе методы численного интегрирования уравнений Максвелла могут называться решателями поля (field solver).

В самом простом варианте производная может быть заменена на выражение, стоящее под знаком предела в определении производной: . Доказывается, что это дает первый порядок аппроксимации метода, т.е. при уменьшении шага на порядок точность численного решения также увеличивается на порядок. Мы будем использовать такой разностный оператор для аппроксимации производной по времени: . Существуют другие разностные операторы, дающие более высокий порядок аппроксимации. Для аппроксимации производных по пространственным координатам (эти производные «заключены» в символе ) мы будем использовать центральный разностный оператор второго порядка . Второй порядок означает, что при уменьшении шага сетки на порядок точность численного решения увеличивается на 2 порядка. Итого в нашем методе первый порядок аппроксимации по времени и второй порядок аппроксимации по пространству.

Если подставить вместо производных их аппроксимации, то получим формулы ниже. На каждом шаге по времени хранится набор сеточных значений поля на текущем шаге. Начальные значения поля определяются из заданных начальных условий. Итерация метода состоит из двух этапов: использование текущих значений ***E***, ***B*** и ***J*** для вычисления новых значений ***E*** (***обновление E***), использование текущих значений ***B*** и новых значений ***E*** для вычисления новых значений ***B*** (***обновление B***). В приводимых далее формулах подразумевается, что вычисления производятся для всех узлов сетки , для которых все члены правой части определены (индексы не выходят за пределы сетки), для остальных узлов значения определяются через граничные условия. Операцию присваивания := нужно трактовать следующим образом: текущее сеточное значение используется для вычисления правой части и затем перезаписывается новым значением (аналогично выполнению строки кода на С++ вида a = a + b;).

Обновление выполняется по следующей схеме:

Обновление выполняется по следующей схеме:

Эти формулы применяются для внутренних узлов сетки. Значения поля в узлах на границе расчетной области не могут быть вычислены по формулам выше, так как в этих выражениях будет хотя бы одно значение, лежащее за пределами сетки. Можно использовать периодические условия: считать, что и так далее (визуальная аналогия: расчетная область это прямоугольный лист, мы его сворачиваем в цилиндр, совмещая противоположные стороны – только здесь мы это делаем по обеим переменным). Замечание: стоит граничные условия реализовать отдельной функцией или классом, чтобы потом можно было легко заменить периодические граничные условия, например, на отражающие.

В предлагаемом для реализации варианте поля и хранятся без смещений. В оригинальном варианте метода FDTD поля хранятся со смещениями относительно друг друга на полшага по пространству и времени. При смещении на полшага по времени вычисляется в точках и т.д., а в точках . Это дает возможность использовать центральный разностный оператор по времени и получить второй порядок аппроксимации по времени: . При смещении на полшага по пространству вычисляется в точках и т.д., а в точках ; тогда производная заменяется центральным разностным оператором с шириной шаблона , а не : . По координатам и аналогично. Хотя это и не увеличивает второй порядок аппроксимации схемы по пространственным координатам, но дает качественно лучшее численное решение. В литературе встречается именно такое описание метода FDTD. С точки зрения программирования все остается прежним, только немного видоизменяются формулы и нужно аккуратно задавать начальные и граничные условия с учетом смещений.

Для трехмерных задач получается 6 формул, аналогичных формулам для и выше. Эти формулы получаются заменой каждой производной в исходных уравнениях центральным разностным оператором второго порядка вида (для производной по ) для пространственных переменных и разностным оператором первого порядка для времени .

Для проверки корректности реализации можно реализовать следующий тест, описывающий распространение плоской монохроматической электромагнитной волны. Даны начальные условия в момент времени (далее – границы расчетной области по координате ):

Аналитическое решение уравнений Максвелла при данных начальных условиях в момент времени следующее:

Таким образом, будет происходить с течением времени сдвиг поля вдоль оси .

***Тест «колебания холодной плазмы»***

В качестве несложного теста для проверки корректности работы PIC-кода модно рассмотреть Ленгмюровские колебания холодной плазмы. Приведенный тест близок к тем конфигурациям, которые моделируются «взрослыми» PIC-кодами. Задаются следующие константы (СГС):

ElectronMass = -4.80320427e-10 # масса электрона

ElectronCharge = 9.10938215e-28 # заряд электрона

LightVelocity = 29979245800 # скорость света

MatrixSizeX = 64 # размер сетки по оси x, варьируется

MatrixSizeY = 8 # размер сетки по оси y

L = 1.0 # длина области по оси x

NumPerL\_Debay = 0.5 # число ячеек сетки на дебаевскую длину

NumPerPlasmaPeriod = 256 # число операций на плазменный период

NumPerCell = 30 # среднее число частиц на ячейку, варьируется

NumPeriods = MatrixSize / (2 \* sqrt(2) \* Pi \* NumPerL\_Debay) # число периодов

SpaceStep = L / MatrixSize # шаг сетки по оси x и y

L\_Debay = SpaceStep \* NumPerL\_Debay # дебаевская длина

Temp = 1e-2 \* ElectronMass \* LightVelocity \* LightVelocity # начальная температура частиц, erg

Density = Temp / (8 \* Pi \* ElectronCharge \* L\_Debay \* ElectronCharge \* L\_Debay) # определяет начальную плотность частиц

w\_p = sqrt(4 \* Pi \* ElectronCharge \* ElectronCharge \* Density / ElectronMass) # плазменная частота

TimeStep = 2 \* (Pi / w\_p) / NumPerPlasmaPeriod # шаг по времени

IterationsNumber = NumPeriods \* NumPerPlasmaPeriod # число шагов по времени

IterationsBetweenDumps = NumPerPlasmaPeriod / 16 # раз в сколько итераций делать вывод результата

ParticlesFactor = Density \* SpaceStep \* SpaceStep \* SpaceStep / NumPerCell # «вес» макрочастицы

A = 0.05

Amp = 2 \* L \* Density \* ElectronCharge \* A # амплитуда колебаний

На основе данных констант задается начальное распределение частиц и поле в начальный момент времени. Граничные условия периодические как для поля, так и для частиц.

Ex0(x, y, t) = -Amp \* cos(2 \* Pi \* x / L) # начальное поле, остальные компоненты нулевые

distribution(x, y) = Density \* (1 + A \* sin(2\*Pi\*x / L)) # функция плотности начального распределения частиц

Частицы генерируются в рамках каждой ячейки случайным образом с использованием равномерного распределения. Количество частиц в ячейке соответствует плотности распределения частиц и определяется по формуле distribution(x, y) \* SpaceStep \* SpaceStep / ParticlesFactor.

Скорость каждой частицы определяется случайным образом с использованием нормального распределения, близкого к распределению Максвелла. Математическое ожидание, равное средней скорости частиц, нулевое. Среднеквадратичное отклонение определяется температурой: sqrt(Temp/ElectronMass). Стандартные средства C++ позволяют генерировать подобного рода распределения (std::normal\_distribution).

Результат выглядит наглядно, если каждые несколько итераций (например, IterationsBetweenDumps) вычислять текущую плотность частиц и строить график плотности в зависимости от в сечении . На полученных графиках будут видны колебания плотности частиц (рис. 1-5). На рисунках слева изображено поле , справа – плотность частиц. Результаты на рисунках получены программным комплексом PICADOR в аналогичной трехмерной конфигурации.

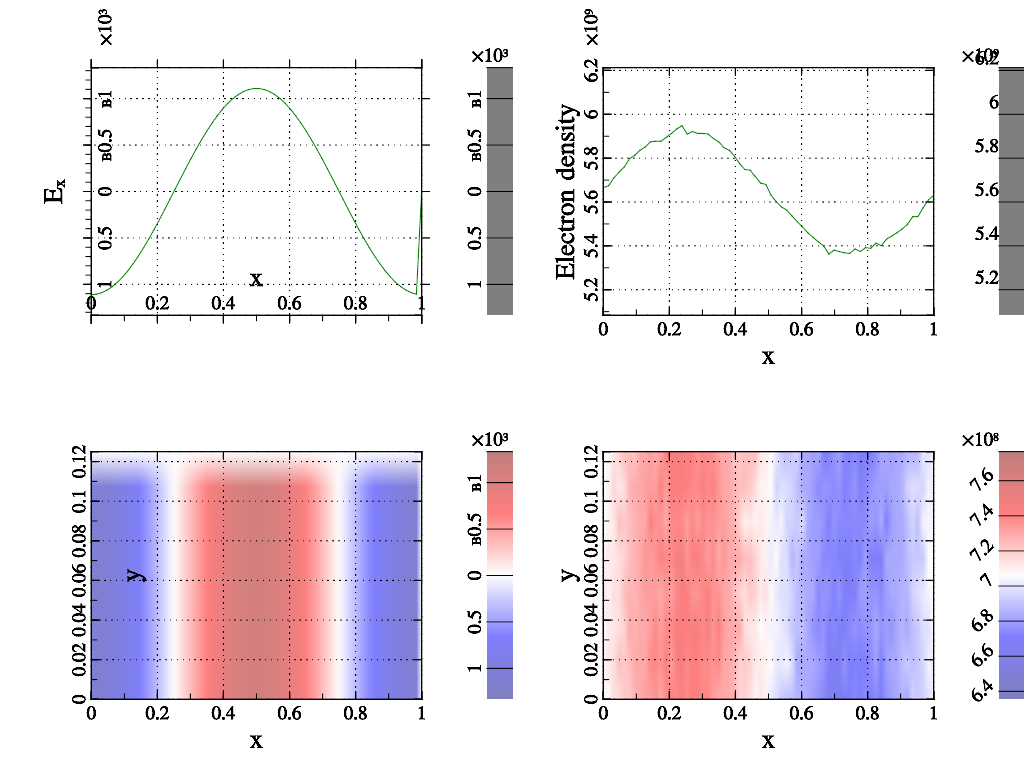


Рисунок 1. Начальное распределение

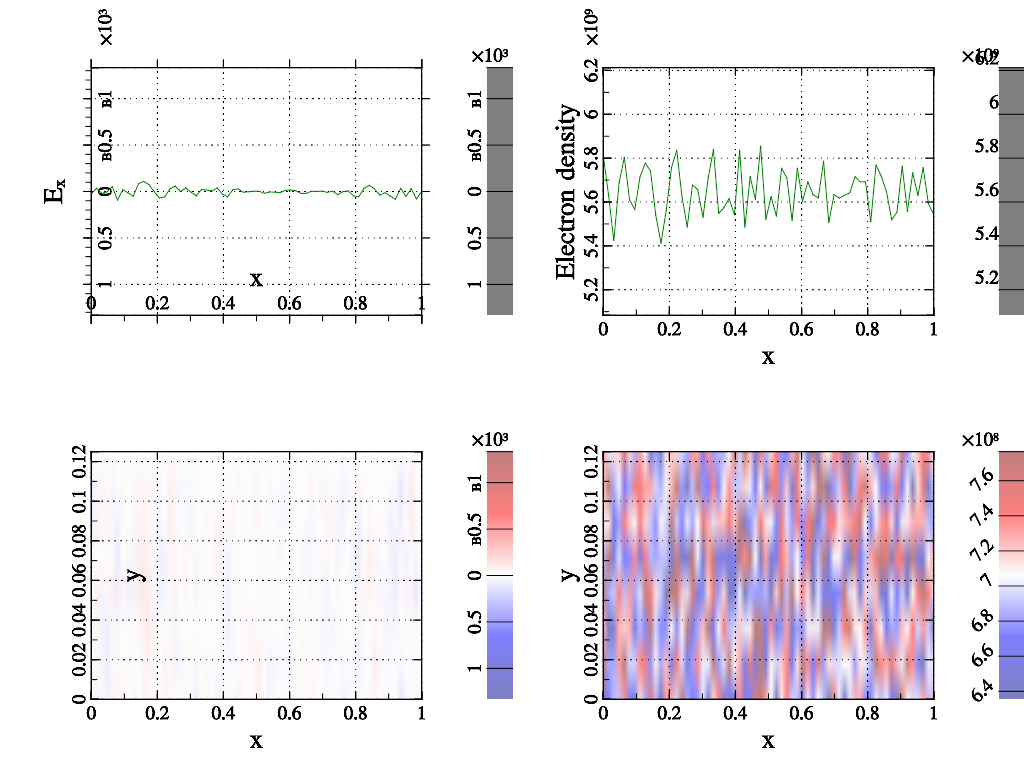


Рисунок 2. Итерация NumPerPlasmaPeriod/4

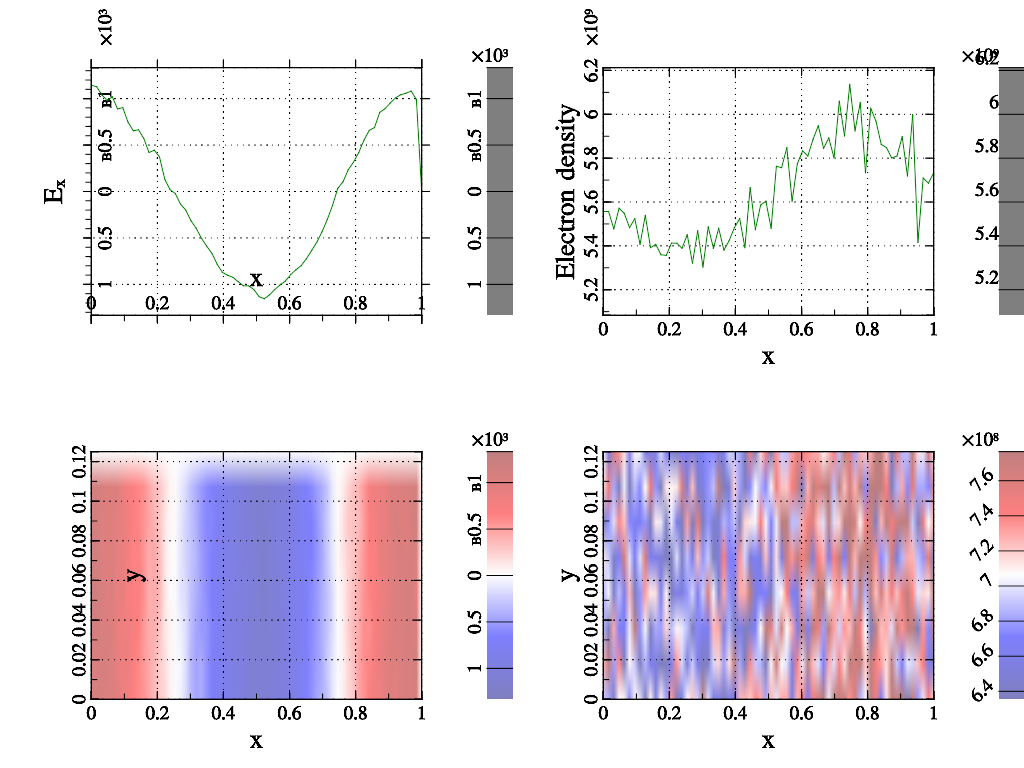


Рисунок 3. Итерация NumPerPlasmaPeriod/2

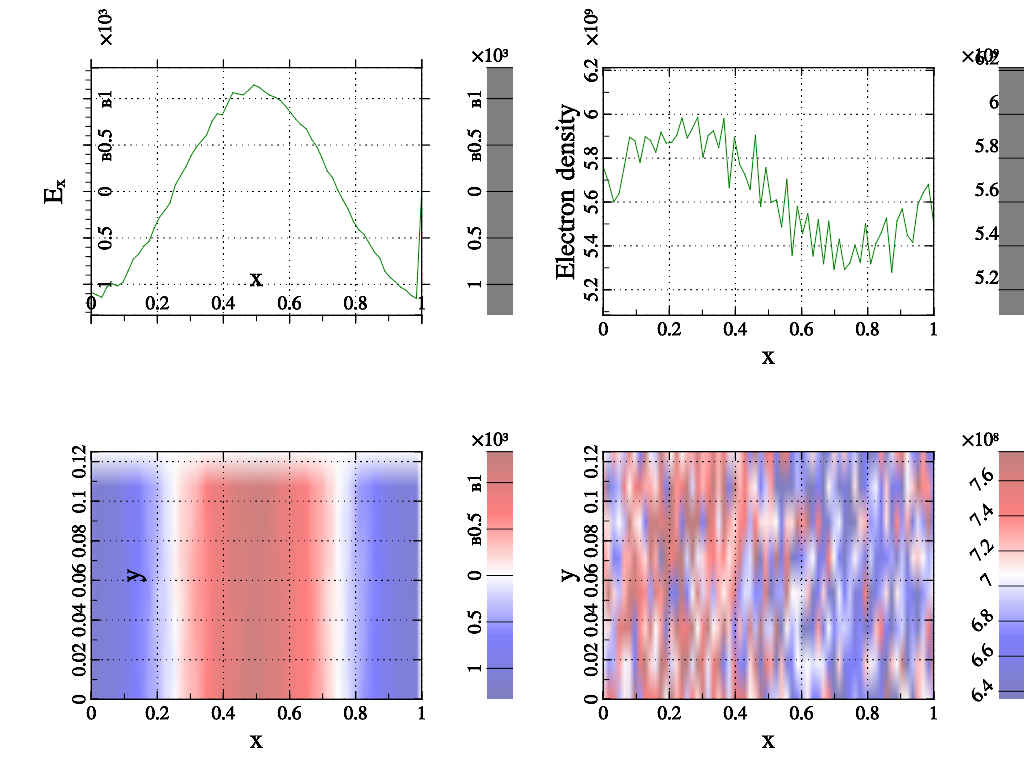


Рисунок 5. Итерация NumPerPlasmaPeriod