intro

June 23, 2024

1 Intro to Hybrid-VPIC

1.1 优点

1.1.1 一般的混合模拟

- 省事儿，不用考虑电子的短暂/小尺度，省计算负担

1.1.2 Hybrid-VPIC

- 计算速度特别快

1.2 模型

1.2.1 方程组

1. Vlasov, 带碰撞项 $C_i\{f_s\}$
2. Ohm: $\mathbf{E} = -(en_e)^{-1}\nabla p_e - \mathbf{u}_i \times \mathbf{B} + (en_e)^{-1}\mathbf{j} \times \mathbf{B} + \mathbf{R}_{ei}$
3. Bulk current $\mathbf{j} = n_e e \cdot (\bar{\mathbf{u}}_i - \mathbf{u}_e)$
4. Resistive $\mathbf{R}_{ei} = \eta\mathbf{j} - \eta_H \nabla^2 \mathbf{j}$
5. Ampere's law $\mu_0 \mathbf{j} = \nabla \times \mathbf{B}$
6. Faraday's law $\mathbf{B}_{,t} = -\nabla \times \mathbf{E}$
7. 本构方程，二选一
    1. polytropic $p_e = p_0 \cdot (n_e/n_0)^\gamma$
    2. heat flux $p_{e,t} = -\gamma\nabla \cdot (p_e \mathbf{u}_e) + (\gamma - 1)\mathbf{u}_e \cdot \nabla p_e + (\gamma - 1)(-\nabla \cdot \mathbf{Q}_e + H_{ei})$ 其中引入 $\mathbf{u}_e = -(en_e)^{-1}(\mu_0^{-1}\nabla \times \mathbf{B} - \mathbf{J}_i)$ 热流 $\mathbf{Q}_e = -\kappa\nabla T_e$.

1.2.2 计算格式

1. 粒子步进：蛙跳格式
2. 半步推整步：Eq. (11)
3. 场量：都在 cell 中心
4. 磁场分开：$\mathbf{B}_0 + \mathbf{B}_1$

1.2.3 边界条件

- 可指定为全周期的

1.3 安装方法

1.3.1 源

github: `lanl/vpic-kokkos` 其中 `hybridVPIC` 分支（不是目录，是分支名!）

1.3.2 编译与运行

VPIC 本体似乎是比较标准的 cmake 工作流。至少 `cmake ..; make` 能编译。出来的 VPIC 是一个编译器/代码生成器，需要套一个 cxx.

1.4 运行方法

1.4.1 Input Deck

先来分析一个: shock 文件包复杂的话，虽然只有一句生成语句，包一个 Makefile （或者更现代化的东西），是值得学习的。* 位置：`examples/shock` * 内容：1. 主程序 `shock-hyb.cxx`, 里面包含了 `injection.cxx` 2. 数据分析、转化程序（两个 f90 的）3. `conf.dat`, `translate_shock.f90` 所用；摘取部分参数（更改时不要忘了更新）4. IDL 和 python 的画图程序 * 那么 shock 是怎么进去的？在 VAC 中，场的设置方法，是 `set_region_field`. 这里是 346 看样子都是常数，并没有怼一条 shock tube 进去。* 左边的 bounce off 粒子在 242, 场在 237. * 右边的 inject open 粒子在 243, 场在 239. * field injection 在 899 行。给右边界注的。

1.4.2 编译与运行

把里面的 Makefile 和 cxx 都拷到目录里。把生成的 vpic 也拷进来。

然后对 Makefile 做两件现代化的事。

1. 指定第 2 行 `SHELL = bash`
2. 指定第 6 行 `PROJECTDIR = ./`

注意他的 case 需要 16 个核，用 mpirun 跑。

1.4.3 后处理

1.5 边界条件的指定方法

1.5.1 场

`pec_fields`

- 定义在 `src/grid/grid.h`
- 是 `anti_symmetric_fields` 的缩写
- 也可写作 `metal_fields`

其他：* 有 `symmetric_fields`, `pmc_fields`, `absorb_fields` 等可用

注入场 在边界处直接更改 `field(x, y, z)` 的值。注意 `x` 等不是浮点数的坐标，而是下标。场的元素，在 `src/field_advance/field_advance.h` 里 * cbx, cby, cbz 是磁场 * ex, ey, ez 是电场 * jfx, jfy, jfz 离子电流；rhof 离子电荷

1.5.2 粒子

暂可指定作 `absorb_particles` 和 `reflect_particles`

1.6 尝试注入 AW

1.6.1 重要物理量，以及对模拟本身的变更

1. 为了方便计算极化关系，把背景速度设为零，磁场设为 $B_0\hat{\mathbf{z}}$.
    1. (L. 346) 原设 $\mathbf{v} = (0, -V_d b_0 \sin\theta)$, theta 的定义在 L. 77, 定义为 10 度
    2. (L. 347) 设 $\mathbf{B} = (b_0 \cos\theta, 0, b_0 \sin\theta)$
    3. 应该将 theta 改为 90 度，或者直接令 cs = 0, sn = 1
    4. 将 V_d 设为 0
2. 主要的物理量已经设好。在模拟所用的单位系下，保存了
    1. `b0` 磁场强度
    2. `v_A` Alfven speed
3. 时间怎么取?
    1. 查 `grid/grid.h`, 有 `C++ t = g->t0 + (double)g->dt*(double)g->step` 但不能引用 `t` 变量；`g` 在程序段中，应作 `grid`.

1.6.2 对 AW 的描述

注意本物理模型中，y 方向很细，导致 y 轴是不变轴，只能在 xoz 面内传播波。扰动是沿 y 面的。因而，设置 kx 与 kz.

用 `f_coord_x` 和 `phase_factor` 等简化表达式的书写。

放入磁场是容易的，但是放入电场，使用 $\mathbf{E} + \mathbf{v} \times \mathbf{B} = 0$, 从而 $\delta\mathbf{E} = -\delta\mathbf{v} * \mathbf{B}_0$. +v 沿 +y, B0 沿 z, 所以 +E 沿 +x.

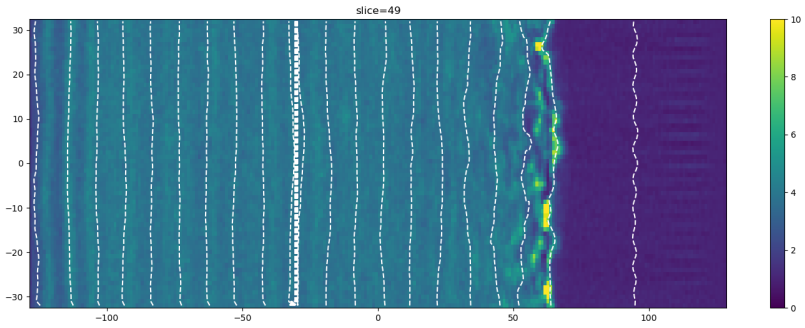
1.7 后处理

看起来那个 py 文件不是直接运行的。

拷来 translate_shock, 但只吃 mpiifort. 所以

1. 先 `mpiifort -o translate_shock translate_shock.f90`
2. 运行它。运行之前自己 `mkdir data`
3. `mpiifort -o ay_gda_integrate ay_gda_integrate.f90`
4. 运行它。
5. 运行 `plotfigs.py`.

1.7.1 结果



1.7.2 Troubleshoot

ay_gda 跑不动 提示记录问题。查磁场文件长度为 0x330000, 网格格点数为 0x4000, 所以每个格点的数不 “整”，有因数 17 (或者说 0x11)，应该是个包含很多信息的字段，而不是矩阵。基于这种猜测，打开 `ay_gda_integrate.f90` 53 行，关掉 56 行。

然后是记录数的问题。既然跑的是第 52 个记录出错，干脆让 19 行的 `it2 = 51`. 跑了 50 个 ω_{ci}^{-1}，那么，有 51 条记录也是合理的。