Notes on WarpX Fluids

Grant R Johnson

October 12, 2023

1 Introduction

[1] The fluid timeloop is embedded inside the standard PIC timeloop. Before the loop begins, it is assumed that the program is in the state where fields **E** and **B** are available at the half timestep of the fluids. The fluids themselves are described by arrays on our nodal grid of the density and momentum density, $\mathbf{Q} \equiv \{N, NU_x, NU_y, NU_z\}$. These quantities are then pushed one time-step at a time via the fluid loop, see figure 1. Implementation details of each step are described below, but to outline the fluid loop procedure we compute: 1. H&C push the momentum, 2. Non-inertial (momentum source) terms Communications 3. Apply BC and and exchange boundary data. 4. MUSCL scheme for advection terms 5. Current and Charge Deposition.

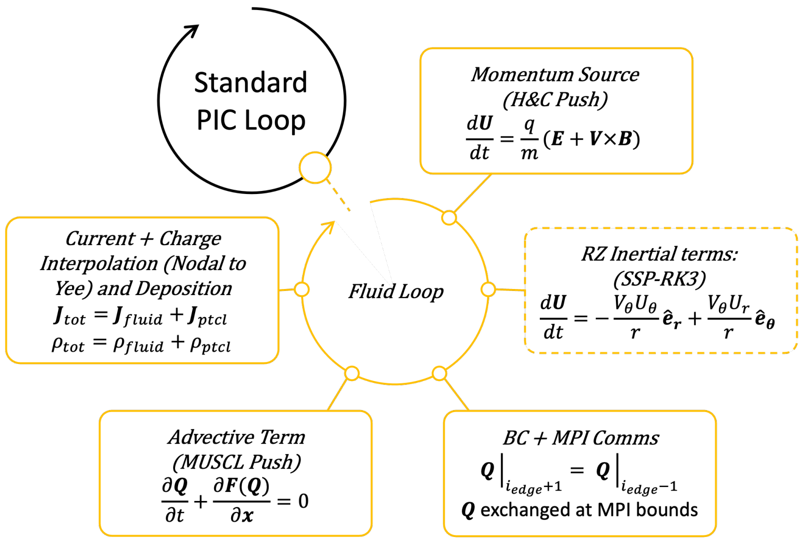
Standard PIC Loop
Momentum Source (H&C Push)
$\frac{d\boldsymbol{U}}{dt} = \frac{q}{m}(\boldsymbol{E} + \boldsymbol{V}\times\boldsymbol{B})$
Current + Charge Interpolation (Nodal to Yee) and Deposition
$\boldsymbol{J}_{tot} = \boldsymbol{J}_{fluid} + \boldsymbol{J}_{ptcl}$
$\rho_{tot} = \rho_{fluid} + \rho_{ptcl}$
Fluid Loop
RZ Inertial terms: (SSP-RK3)
$\frac{d\boldsymbol{U}}{dt} = -\frac{V_\theta U_\theta}{r}\hat{\boldsymbol{e}}_r + \frac{V_\theta U_r}{r}\hat{\boldsymbol{e}}_\theta$
Advective Term (MUSCL Push)
$\frac{\partial \boldsymbol{Q}}{\partial t} + \frac{\partial \boldsymbol{F}(\boldsymbol{Q})}{\partial \boldsymbol{x}} = 0$
BC + MPI Comms
$\boldsymbol{Q}\big|_{i_{edge}+1} = \boldsymbol{Q}\big|_{i_{edge}-1}$
$\boldsymbol{Q}$ exchanged at MPI bounds

Figure 1: The fluid evolution loop. Fields at the beginning of this loop are pushed to time $t_n + dt/2$ and interpolated to nodal locations to match the fluid locations.

1.1 Governing equations

In the cold limit (zero internal pressure) of a relativistic plasma, our system of equations governing the plasma evolution is governed by the Maxwell-Fluid system. The fluid equations are given by,

$$\begin{aligned}
\frac{\partial N_s}{\partial t} + \nabla\cdot(N_s\mathbf{V}_s) &= 0\\
\frac{\partial (N\mathbf{U})_s}{\partial t} + \nabla\cdot((N\mathbf{U})_s\mathbf{V}_s) &= \frac{q_s N_s}{m_s}(\mathbf{E}_s + \mathbf{V}_s\times\mathbf{B}_s).
\end{aligned} \tag{1}$$

[1]The fluid algorithm in WarpX can be found in the github PR: Johnson452/WarpX.

Where the fields are updated via SI versions of Maxwell's equations [2]

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0} \qquad \nabla \cdot \mathbf{B} = 0$$
$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \qquad \nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t}$$

The fluids are coupled to the fields via,

$$\rho = \rho_{ptcl} + \sum_s q_s N_s \qquad \mathbf{J} = \mathbf{J}_{ptcl} + \sum_s q_s N_s \mathbf{V}_s$$
$$\mathbf{V}_s = \frac{(N\mathbf{U})_s}{\sqrt{N_s^2 + (N\mathbf{U})_s^2/c^2}} \qquad (N\mathbf{U})_s = N_s \mathbf{U}_s$$

where the particle quantities are calculated by the PIC algorithm. Therefore, the fluids and particles only couple through Maxwell's equations.[3]

1.2 Space Discretization

The Yee grid stores quantities for the current density, electric field, and magnetic field (and charge at the nodal points, but it is only used for diagnostics). All fluid quantities, $\mathbf{Q}$, exist at the nodal points. This choice is made for the momentum source terms which requires all the fluid elements to exist at the same location. This co-location also allows us to transform without interpolation between $\mathbf{Q}$ variables and $\{N, \mathbf{V}, \mathbf{U}, \mathbf{J}_{fluid}\}$.

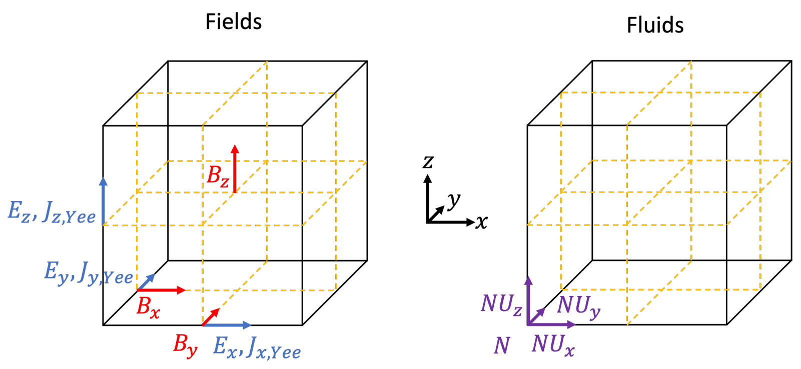

Figure 2: Yee and fluid grid quantity locations.

1.3 Operator Splitting

Discretely, Maxwell's Equations are updated via the Yee (FDTD) algorithm. $\mathbf{J}_{ptcl}$ and ρ_{ptcl} are update and interpolated within the WarpX PIC loop. (For our test-cases so far, we do not have any particles, just fluids.) For discretely solving the fluid equations, we employ operator splitting on equations 1. In general, we use first-order Strang splitting to separate these operations. More explicitly, we take a full time step for the quantity $\mathbf{X}$ we wish to update with a single operator, then compute the next operation using the result of the first. Once we have completed all the operations which make up our system, have have completed our fluid timeloop. We split the fluid equations into operators $\mathcal{L}_1$ and $\mathcal{L}_2$,

$$\frac{\partial N_s}{\partial t} + \underbrace{\nabla \cdot (N_s \mathbf{V}_s)}_{\mathcal{L}_1} = 0$$
$$\frac{\partial (N\mathbf{U})_s}{\partial t} + \underbrace{\nabla \cdot ((N\mathbf{U})_s \mathbf{V}_s)}_{\mathcal{L}_1} = \underbrace{\frac{q_s N_s}{m_s}(\mathbf{E}_s + \mathbf{V}_s \times \mathbf{B}_s)}_{\mathcal{L}_2}.$$

[2]In WarpX all quantities are in SI units, including under-the-hood.

[3]We have no fluid self-injection model for the fluids in WFA, explicitly, we do not exchange fluid elements with particles or vis versa.

Updating $\mathcal{L}_2$ from $t_n \rightarrow t_n + dt$ via,

$$\begin{aligned}\frac{\partial N_s}{\partial t} &= 0\\ \frac{\partial (N\mathbf{U})_s}{\partial t} &= \frac{q_s N_s}{m_s}(\mathbf{E}_s + \mathbf{V}_s \times \mathbf{B}_s).\end{aligned}$$

Which can be simplified using the time-independence of the density too Lorentz's equation,

$$\frac{\partial \mathbf{U}_s}{\partial t} = \frac{q_s}{m_s}(\mathbf{E}_s + \mathbf{V}_s \times \mathbf{B}_s). \tag{2}$$

Next, updating $\mathcal{L}_1$, we employ the MUSCL scheme described in the subsection *MUSCL Advection.*

$$\begin{aligned}\frac{\partial N_s}{\partial t} + \nabla \cdot (N_s \mathbf{V}_s) &= 0\\ \frac{\partial (N\mathbf{U})_s}{\partial t} + \nabla \cdot ((N\mathbf{U})_s \mathbf{V}_s) &= 0.\end{aligned} \tag{3}$$

In Cartesian cases we only have two operators $\mathcal{L}_1$ and $\mathcal{L}_2$ but we need to split $\mathcal{L}_1$ further in RZ for non-inertial terms; see *Momentum Source, Non-Inertial Terms in RZ* section for the third operator.

1.4 Momentum Source, H&C Push

First we split our saved variables $\mathbf{Q}$ into $\{N, \mathbf{U}\}$. This splitting is exact since all our fluid quantities exist at the same location. Next we interpolate our fields (using the WarpX default which is linear interpolation) to convert $\mathbf{E}_{Yee}$ and $\mathbf{B}_{Yee}$ to $\mathbf{E}_{Nodal}$ and $\mathbf{B}_{Nodal}$. Using the Higeura and Cary algorithm [HC17], we push our fluid momentum $\mathbf{U}$, Equation 2, by one timestep. We then recombine the updated $\mathbf{U}$ with the unchanged N back into $\mathbf{Q}$.

1.5 Momentum Source, Non-Inertial Terms in RZ

Not used at the moment...

1.6 Boundary Conditions and Communications

Non-periodic boundaries with perfect electric conductor boundary conditions use copy boundaries for the fluids. With some index i defining a boundary cell in the same direction, the ghost cell $i-1$ is updated using the quantity $i+1$ adjacent to the boundary inside the valid domain. In equation form we copy our vector $\mathbf{Q}$ to the ghost cell via (dropping the species subscript),

$$\mathbf{Q}|_{i_{bounds}-1} = \mathbf{Q}|_{i_{bounds}+1}. \tag{4}$$

Interior MPI boundaries and periodic directions are then copied for the field and fluid elements (handled by *AMReX* using *FillBoundary()*) to their ghost cell counterparts.

1.7 MUSCL Advection

We update equation 3 by one timestep with the MUSCL scheme. We begin by rewriting the equation 3 in terms of $\mathbf{Q}$ and $\mathbf{F}(\mathbf{Q}) \equiv \mathbf{QV}$,

$$\frac{\partial \mathbf{Q}}{\partial t} + \nabla \cdot (\mathbf{F}(\mathbf{Q})) = 0$$

which we can re-write in a Quasi-linear form (in Cartesian coordinates) as,

$$\frac{\partial \mathbf{Q}}{\partial t} + \sum_{m=1}^{N_{dim}} \frac{\partial \mathbf{F}_m(\mathbf{Q})}{\partial \mathbf{Q}} \frac{\partial \mathbf{Q}}{\partial x_m} = 0. \tag{5}$$

where we define the Flux-Jacobian in each direction as, $\overleftrightarrow{A}_m \equiv \frac{\partial \mathbf{F}_m(\mathbf{Q})}{\partial \mathbf{Q}}$ where $\mathbf{F}_m \equiv \mathbf{Q}V_m$.

Before we continue, let me introduce some helpful operators and shorthand. $\Delta_{\{L,I,R\}}, m$ is an operator which takes the index $m \in \{1, 2, 3\}$ mapping to $\{i, j, k\}$ which are the discrete x, y, z

dimension indices, and either subtracts 1 (Δ_L), returns the same value (Δ_I), or adds on to the index (Δ_R). An example calculation would be, $\Delta_{L,2}\mathbf{Q}_{i,j,k} = \mathbf{Q}_{i,j,k-1}$. Another shorthand to note is the use of $(...)_{i,j,k}$ which means all discrete quantities in the parenthesis inherit the subscript. For example, $(NV)_{i,j,k} = N_{i,j,k}V_{i,j,k}$. Lastly, the flux surface areas an volumes have been defined as $dS_{\{R,L\},m,i,j,k}$, and $dV_{i,j,k}$ respectively. For Cartesian cases these are simple, for instance in the x-direction ($m = 1$) for 3D, then we have for all R, L, i, j, k, $dS = dydz$ and $dV = dxdydz$. The R, L are referring to which flux surface we are discussing. In general, R means the surface to the right on a positively oriented direction, e.g. $i+1$ rather than i. This is then sufficiently general to apply to the 1D, 2D, 3D, and RZ configurations we simulate.

Our MUSCL scheme and flux limit then occur in the following order starting with $\mathbf{Q}$ at timestep t_n [4]:

1. Compute the flux Jacobians $\overleftrightarrow{A}_m$ at each cell $\{i, j, k\}$ (see Appendix A) via,
$$\overleftrightarrow{A}_{m,i,j,k} = \left(\frac{\partial \mathbf{F}_m(\mathbf{Q})}{\partial \mathbf{Q}}\right)_{i,j,k}.$$

2. Compute the cell-slopes using shifting operators (see subsection 1.9),
$$d\mathbf{Q}_{m,i,j,k} = ave(\Delta_{I,m}\mathbf{Q}_{i,j,k} - \Delta_{L,m}\mathbf{Q}_{i,j,k}, \Delta_{R,m}\mathbf{Q}_{i,j,k} - \Delta_{I,m}\mathbf{Q}_{i,j,k}).$$

3. Predict $\tilde{\mathbf{Q}}$, which is $\mathbf{Q}$ at $t_n + dt/2$ using the grid spacing in each dimension, dx_m, (note this is a matrix multiplication between $\overleftrightarrow{A}_m$ and $d\mathbf{Q}_m$),
$$\tilde{\mathbf{Q}}_{i,j,k} = \mathbf{Q}_{i,j,k} - \left(\frac{dt}{2}\sum_{m=1}^{N_{dim}} \frac{\overleftrightarrow{A}_m d\mathbf{Q}_m}{dx_m}\right)_{i,j,k}.$$

4. Compute the values at the cell edges and half timestep,
$$\tilde{\mathbf{Q}}^{\pm}_{edges,m,i,j,k} = \tilde{\mathbf{Q}}_{i,j,k} \mp d\mathbf{Q}_{m,i,j,k}/2.$$

5. Positivity limiter, if $\tilde{\mathbf{N}}^{\pm}_{edges,m,i,j,k} < 0$ then set $d\mathbf{Q}_{m,i,j,k} = \mathbf{0}$ (hence only the direction which positivity is violated) and recomputes steps 3-4 with this updated slope.

6. Compute the fluxes at the boundaries (see subsection 1.10),
$$\mathbf{F}_{R,m,i,j,k} = Flux_m(\Delta_{I,m}\tilde{\mathbf{Q}}^{-}_{edges,m,i,j,k}, \Delta_{R,m}\tilde{\mathbf{Q}}^{+}_{edges,m,i,j,k}).$$
$$\mathbf{F}_{L,m,i,j,k} = Flux_m(\Delta_{L,m}\tilde{\mathbf{Q}}^{-}_{edges,m,i,j,k}, \Delta_{I,m}\tilde{\mathbf{Q}}^{+}_{edges,m,i,j,k}).$$

7. We update $\mathbf{Q}$ to the next timestep,
$$\mathbf{Q}^{n+1}_{i,j,k} = \mathbf{Q}^{n}_{i,j,k} - \left(\frac{dt}{dV}\sum_{m=1}^{N_{dim}} (\mathbf{F}_{R,m} \times dS_{R,m} - \mathbf{F}_{L,m} \times dS_{L,m})\right)_{i,j,k}.$$

Steps 1-5 happen in a single loop over the entire grid, updating each quantity cell by cell and saving the result, $\tilde{\mathbf{Q}}^{\pm}_{edges,m,i,j,k}$, on the grid, including the ghost cells. Including the ghost cells means we have some redundant computations where overlaps occur, but this saves us from an expensive MPI communication at this point. We then preform a second iteration over just the valid indices of our domain, completing steps 6-7. Details on these specific implementations such as *ave()* and *Flux()* are presented in subsequent sections.

[4]The time index is a suppressed superscript in steps 1-6 except in step 7.

1.8 Primitive Variable Half-Step

Using the conserved variables $\mathbf{Q}$ to reconstruct our solution at the half step (see 1.7) introduced unacceptable noise into the solutions. We found the better choice was to reconstruct these solutions using the primitive variables, $\mathbf{U} = \{N, U_x, U_y, U_z\}$. Rewriting the fluid-advective equations we get,

$$\frac{\partial \mathbf{U}}{\partial t} + \sum_{m=1}^{N_{dim}} \overleftrightarrow{\mathcal{J}}_m \frac{\partial \mathbf{U}}{\partial x_m} = 0. \tag{6}$$

where the matrices, $\overleftrightarrow{\mathcal{J}}_m$, are written in the appendix A. These are simpler than their conservative-form counter parts to compute. Using $\mathbf{U}$ and equation 6 we replace steps 1-5 with:

1. Compute the matrices $\overleftrightarrow{\mathcal{J}}_m$ at each cell $\{i, j, k\}$ (see Appendix A).
2. Compute the cell-slopes using shifting operators (see subsection 1.9),
$$d\mathbf{U}_{m,i,j,k} = ave(\Delta_{I,m}\mathbf{U}_{i,j,k} - \Delta_{L,m}\mathbf{U}_{i,j,k}, \Delta_{R,m}\mathbf{U}_{i,j,k} - \Delta_{I,m}\mathbf{U}_{i,j,k}).$$
3. Predict $\tilde{\mathbf{U}}$, which is $\mathbf{U}$ at $t_n + dt/2$ using the grid spacing in each dimension, dx_m, (note this is a matrix multiplication between $\overleftrightarrow{\mathcal{J}}_m$ and $d\mathbf{U}_m$),
$$\tilde{\mathbf{U}}_{i,j,k} = \mathbf{U}_{i,j,k} - \left(\frac{dt}{2} \sum_{m=1}^{N_{dim}} \frac{\overleftrightarrow{\mathcal{J}}_m d\mathbf{U}_m}{dx_m} \right)_{i,j,k}.$$
4. Compute the values at the cell edges and half timestep,
$$\tilde{\mathbf{U}}^{\pm}_{edges,m,i,j,k} = \tilde{\mathbf{U}}_{i,j,k} \mp d\mathbf{U}_{m,i,j,k}/2.$$
5. Positivity limiter, if $\tilde{\mathbf{N}}^{\pm}_{edges,m,i,j,k} < 0$ then set $d\mathbf{U}_{m,i,j,k} = \mathbf{0}$ (hence only the direction which positivity is violated) and recomputes steps 3-4 with this updated slope.
6. Reconstruct $\tilde{\mathbf{Q}}^{\pm}_{edges,m,i,j,k}$ using $\mathbf{U}$ in each cell. Then proceede with section 1.7 steps 6 & 7.

1.9 The *ave()* function

The *ave()* function is interchangeable with a several limiters available. These can further be applied to specific components of the slope. For instance, we can compute $ave_{superbee}$*()* for momentum density slopes and ave_{minmod}*()* for the density slope. The default choice for all slopes is the low-diffusion minmod:

$$ave_{low-diff,minmod}(a, b) = \begin{cases} minmod((a + b)/2, 2a, 2b) & \text{ab} > 0 \\ 0 & \text{ab} \leq 0 \end{cases} \tag{7}$$

where our other choices are

$$ave_{superbee}(a, b) = \begin{cases} minmod(maxmod(a, b), minmod(2a, 2b)) & \text{ab} > 0 \\ 0 & \text{ab} \leq 0 \end{cases} \tag{8}$$

$$ave_{high-diff,minmod}(a, b) = \begin{cases} minmod((a + b)/2, a, b) & \text{ab} > 0 \\ 0 & \text{ab} \leq 0 \end{cases} \tag{9}$$

$$ave_{first-order}(a, b) = 0 \tag{10}$$

which can be used interchangeably by changing the function associated with computing each quantity. Superbee, and minmod are all total variation diminishing (TVD) to reduce spurious oscillations. As a note, the total variation of a discretized quantity, u, is defined as,

$$TV(u^n) = \sum_j |u^n_{j+1} - u^n_j|. \tag{11}$$

where TVD is defined as $TV(u^{n+1}) \leq TV(u^n)$.

1.10 The *Flux()* function

We implemented the Rusanov Flux,

$$Flux_m(\mathbf{Q}_L, \mathbf{Q}_R) = \frac{1}{2}(V_m(\mathbf{Q}_R)\mathbf{Q_R} + V_m(\mathbf{Q}_L)\mathbf{Q_L}) - \frac{c_m(\mathbf{Q}_R, \mathbf{Q}_L)}{2}(\mathbf{Q}_R - \mathbf{Q}_L) \tag{12}$$

where the function $c(\mathbf{Q}_R, \mathbf{Q}_L)$ is defined by the maximum of the absolute value of the set of all the eigenvalues from both the Flux Jacobians computed using $\mathbf{Q}_R$, and $\mathbf{Q}_L$,

$$c_m(\mathbf{Q}_R, \mathbf{Q}_L) \equiv max\{|V_m(\mathbf{Q}_R)|, |V_m(\mathbf{Q}_L)|\}. \tag{13}$$

Using Maxima, it can be shown that there are four repeated eigenvalues for our equations which are the velocity in the direction we compute the Flux Jacobian. Finally, he velocity is reconstructed from $\mathbf{Q}$ via,

$$V_m(\mathbf{Q}) = \frac{Q_m}{\sqrt{Q_0^2 + (Q_1^2 + Q_2^2 + Q_3^2)/c^2}}. \tag{14}$$

1.11 Current and Charge Deposition

We do linear interpolation to deposit current and rho to the total values. This is a change between Nodal and cell-centered values.

2 Laser Envelope

In order to avoid high-frequency noise introduced by the laser, we approximate the laser with the pondermotive force term in our fluid momentum equation.

$$\frac{d\mathbf{p}}{dt} = q(\mathbf{E}_s + \mathbf{v} \times \mathbf{B}_s) - \frac{q^2}{2\gamma m}\nabla \left\langle A_{env}^2 \right\rangle \tag{15}$$

This replaces out the high-frequency laser into a slow moving envelope, and can be seamlessly integrated into our Higuera and Cary [HC17] push by rewriting this term as an externally applied electric field. Equating $A_{env} = (mc/q)a_{env}$, $\left\langle A_{env}^2 \right\rangle = 0.5 \left| A_{env}^2 \right|$, and from the gaussian laser pulse $a_{env} = a_0 \exp^{\frac{-(z-ct-z_0)^2}{c^2\tau^2}}$, we can come up with an expression for our effective electric field,

$$\begin{aligned}
\mathbf{E}_{ext} &= -\frac{q}{4\gamma m}\nabla \left| A_{env}^2 \right| \\
E_{z,ext} &= -\frac{mc^2}{4\gamma q}\partial_z \left(a_0^2 \exp^{\frac{-2(z-ct-z_0)^2}{c^2\tau^2}} \right) \\
&= -\frac{mc^2}{4\gamma q}\frac{(-4(z-ct-z_0))}{c^2\tau^2}\left(a_0^2 \exp^{\frac{-2(z-ct-z_0)^2}{c^2\tau^2}} \right) \\
&= \frac{m}{\gamma q}\frac{((z-ct-z_0))}{\tau^2}a_{env}^2.
\end{aligned}$$

The x and y-components of the external fields are zero. We also note that γ is defined as $\gamma = \sqrt{1 + (\mathbf{p}^2 + q^2 \left\langle A_{env}^2 \right\rangle)/(mc)^2}$ and simplify by assuming $\mathbf{p}^2 = 0$ to $\gamma \approx \sqrt{1 + 0.5a_{env}^2}$. So our final result of the effective electric field which approximates our pondermotive force due to the laser is,

$$E_{z,ext} = \frac{m((z-ct-z_0))}{q\tau^2\sqrt{1 + 0.5a_{env}^2}}a_{env}^2. \tag{16}$$

A Appendix A: The Flux Jacobians

In the x-direction, the Flux Jacobian computed with Maxima is,

$$\overleftrightarrow{A}_1 = \begin{bmatrix} \frac{(U_x U_z^2 + U_x U_y^2 + U_x^3)\gamma}{a} & \frac{(U_z^2+U_y^2+1)\gamma}{a} & -\frac{U_x U_y}{\gamma^3} & -\frac{U_x U_z}{\gamma^3} \\ -\frac{U_x^2}{\gamma^3} & \frac{(2U_x U_z^2 + 2U_x U_y^2 + U_x^3 + 2U_x)\gamma}{a} & -\frac{U_x^2 U_y}{\gamma^3} & -\frac{U_x^2 U_z}{\gamma^3} \\ -\frac{U_x U_y}{\gamma^3} & \frac{(U_y U_z^2 + U_y^3 + U_y)\gamma}{a} & \frac{(U_x U_z^2 + U_x^3 + U_x)\gamma}{a} & -\frac{U_x U_y U_z}{\gamma^3} \\ -\frac{U_x U_z}{\gamma^3} & \frac{(U_z^3 + (U_y^2+1)U_z)\gamma}{a} & -\frac{U_x U_y U_z}{\gamma^3} & \frac{(U_x U_y^2 + U_x^3 + U_x)\gamma}{a} \end{bmatrix} \tag{17}$$

$$\overleftrightarrow{A}_2 = \begin{bmatrix} \frac{(U_y U_z^2 + U_y^3 + U_x^2 U_y)\gamma}{a} & -\frac{U_x U_y}{\gamma^3} & \frac{(U_z^2+U_x^2+1)\gamma}{a} & -\frac{U_y U_z}{\gamma^3} \\ -\frac{U_x U_y}{\gamma^3} & \frac{(U_y U_z^2 + U_y^3 + U_y)\gamma}{a} & \frac{(U_x U_z^2 + U_x^3 + U_x)\gamma}{a} & -\frac{U_x U_y U_z}{\gamma^3} \\ -\frac{U_y^2}{\gamma^3} & -\frac{U_x U_y^2}{\gamma^3} & \frac{(2U_y U_z^2 + U_y^3 + (2U_x^2+2)U_y)\gamma}{a} & -\frac{U_y^2 U_z}{\gamma^3} \\ -\frac{U_y U_z}{\gamma^3} & -\frac{U_x U_y U_z}{\gamma^3} & \frac{(U_z^3 + (U_x^2+1)U_z)\gamma}{a} & \frac{(U_y^3 + (U_x^2+1)U_y)\gamma}{a} \end{bmatrix} \tag{18}$$

$$\overleftrightarrow{A}_3 = \begin{bmatrix} \frac{(U_z^3 + (U_y^2+U_x^2)U_z)\gamma}{a} & -\frac{U_x U_z}{\gamma^3} & -\frac{U_y U_z}{\gamma^3} & \frac{(U_y^2+U_x^2+1)\gamma}{a} \\ -\frac{U_x U_z}{\gamma^3} & \frac{(U_z^3 + (U_y^2+1)U_z)\gamma}{a} & -\frac{U_x U_y U_z}{\gamma^3} & \frac{(U_x U_y^2 + U_x^3 + U_x)\gamma}{a} \\ -\frac{U_y U_z}{\gamma^3} & -\frac{U_x U_y U_z}{\gamma^3} & \frac{(U_z^3 + (U_x^2+1)U_z)\gamma}{a} & \frac{(U_y^3 + (U_x^2+1)U_y)\gamma}{a} \\ -\frac{U_z^2}{\gamma^3} & -\frac{U_x U_z^2}{\gamma^3} & -\frac{U_y U_z^2}{\gamma^3} & \frac{(U_z^3 + (2U_y^2+2U_x^2+2)U_z)\gamma}{a} \end{bmatrix} \tag{19}$$

where $a \equiv c^2\gamma^3$, and $\gamma = \sqrt{1 + \mathbf{U}^2/c^2}$. The eigenvalues of $\lambda_{A_1} = V_x$ with multiplicity four, and similarly for $\lambda_{A_2} = V_y$, $\lambda_{A_3} = V_z$ also both with multiplicity four.

In terms of primative variables, rather than conserved variables

$$\overleftrightarrow{J}_1 = \begin{bmatrix} V_x & \frac{N}{\gamma}\left(1 - \frac{Vx^2}{c^2}\right) & \frac{-NU_xU_y}{a} & \frac{-NU_xU_z}{a} \\ 0 & V_x & 0 & 0 \\ 0 & 0 & V_x & 0 \\ 0 & 0 & 0 & V_x \end{bmatrix} \tag{20}$$

$$\overleftrightarrow{J}_2 = \begin{bmatrix} V_y & \frac{-NU_xU_y}{a} & \frac{N}{\gamma}\left(1 - \frac{Vy^2}{c^2}\right) & \frac{-NU_yU_z}{a} \\ 0 & V_y & 0 & 0 \\ 0 & 0 & V_y & 0 \\ 0 & 0 & 0 & V_y \end{bmatrix} \tag{21}$$

$$\overleftrightarrow{J}_3 = \begin{bmatrix} V_z & \frac{-NU_xU_z}{a} & \frac{-NU_yU_z}{a} & \frac{N}{\gamma}\left(1 - \frac{Vz^2}{c^2}\right) \\ 0 & V_z & 0 & 0 \\ 0 & 0 & V_z & 0 \\ 0 & 0 & 0 & V_z \end{bmatrix} \tag{22}$$

References

[HC17] Adam V Higuera and John R Cary. Structure-preserving second-order integration of relativistic charged particle trajectories in electromagnetic fields. *Physics of Plasmas*, 24(5), 2017.