Introduction to QPAD (for developers, updated on 05/20/2022)

- Glance of input file
- Numerical workflow and parallelization of QPAD
- Code hierarchy
- Source code organization
- Major routines for each procedure in the workflow
- Future work

Glance of input file: simulation

```
"simulation" :
{
    "nodes" : [2, 2],
    "grid" : [250, 500],
    "box" : {
        "r" : [0.0, 5.0],
        "z" : [-5.0, 5.0]},
    "field_boundary" : "open",
    "max_mode" : 1,
    "interpolation" : "linear",
    "n0" : 1.0e16,
    "time" : 100.1,
    "dt" : 10.0,
    "nbeams" : 1,
    "nspecies" : 1,
    "nneutrals" : 0,
    "nlasers" : 0,
    "dump_restart" : false,
    "ndump_restart" : 1,
    "read_restart" : false,
    "restart_timestep" : 1,
    "iter" : 1,
    "smooth_type" : "none",
    "smooth_order" : 0,
    "verbose" : 0,
    "if_timing" : true,
    "random_seed" : 10,
    "algorithm" : "standard"
},
```

More nodes are preferred to deploy in ξ-direction

Δξ should be smaller than Δr in principle.

There is only open BC for EM fields, which is different from QuickPIC where absorbing BCs are used.

Useless unless the ionization is involved.

No CFL limit. It should be set to well resolve the motion of drive beams/lasers.

Number of predictor-corrector iteration. Usually, iter=1 is enough for most cases, but it is recommended to do convergence test to determine the proper value.

Smooth does not function normally, should be turned off.

Verbose of the debug information and it should be set as 0 unless for debugging purpose.

Glance of input file: beam

```
"beam" :
[
    {
    "profile_type" : "standard",
    "geometry" : "cartesian",
    "profile" : ["gaussian", "gaussian", "gaussian"],
    "evolution" : true,
    "push_type" : "reduced",
    "has_spin" : false,
    "ppc" : [2, 2, 2],
    "num_theta" : 16,
    "npmax" : 1000000,
    "q" : -1.0,
    "m" : 1.0,
    "gamma" : 20000,
    "density" : 4.0,
    "quiet_start" : true,
    "gauss_center" : [0.0, 0.0, -2.5],
    "gauss_sigma" : [0.25, 0.25, 0.5],
    "range1" : [-1.25, 1.25],
    "range2" : [-1.25, 1.25],
    "range3" : [-5.0, 0.0],
    "uth" : [5.0, 5.0, 0.0],
    "den_min" : 1e-10,
    "diag" :
    [
        {
        "name" : ["charge_cyl_m"],
        "ndump" : 1
        },
        {
        "name" : ["raw"],
        "ndump" : 1,
        "psample" : 10
        }
    ]
    }
],
```

There are three types of beam initialization “standard” (Osiris-like), “random” (QuickPIC-like) and importing from a file.

1-, 2- and 3-directions of input parameters correspond to x-, y- and z-directions and r-, θ- and z-directions for “cartesian” and “cylindrical” respectively.

“standard”=Boris pusher and “reduced”=QuickPIC-like pusher which assumes particles move at c.

Maximum particle number per MPI partition. It is dynamically adjusted in the runtime.

Symmetrizing the phase space to mitigate seeding noise. Setting it as “true” will double the particle number.

Density profile parameters in 1-, 2-, and 3-directions.

Minimum density threshold for particle injection.

Sampling frequency, e.g., 10 means dump 1 particle for every 10 particles.

Glance of input file: species and field

```
"species" :
[
    {
    "profile" : ["uniform", "uniform"],
    "ppc" : [2,2],
    "num_theta" : 16,
    "q" : -1.0,
    "m" : 1.0,
    "density" : 1.0,
    "push_type" : "robust",
    "diag" :
    [
        {
        "name" : ["charge_cyl_m"],
        "ndump" : 1
        },
        {
        "name" : ["raw"],
        "ndump" : 1,
        "psample" : 10
        }
    ]
    }
],

"field" :
{
    "diag" :
    [
        {
            "name" : ["psi_cyl_m","er_cyl_m","bphi_cyl_m","ez_cyl_m"],
            "ndump" : 1
        }
    ]
}
```

The “species” only initialize in cylindrical geometry.

The “num_theta” is the number of “virtual” cells in θ, “ppc(2)” is the particle # per cell in each “virtual” cell in θ. The total particle numbers distributed in θ is thus num_theta * ppc(2).

“robust” is a Boris-like pusher

Raw data diagnostic is not in use, and it will be modified as the 2D particle tracking in the future.

It only contains diagnosis for various type of fields.

Numerical workflow

Quasi-static approximation

Cartesian coordinates $(x, y, z; t)$

Co-moving coordinates

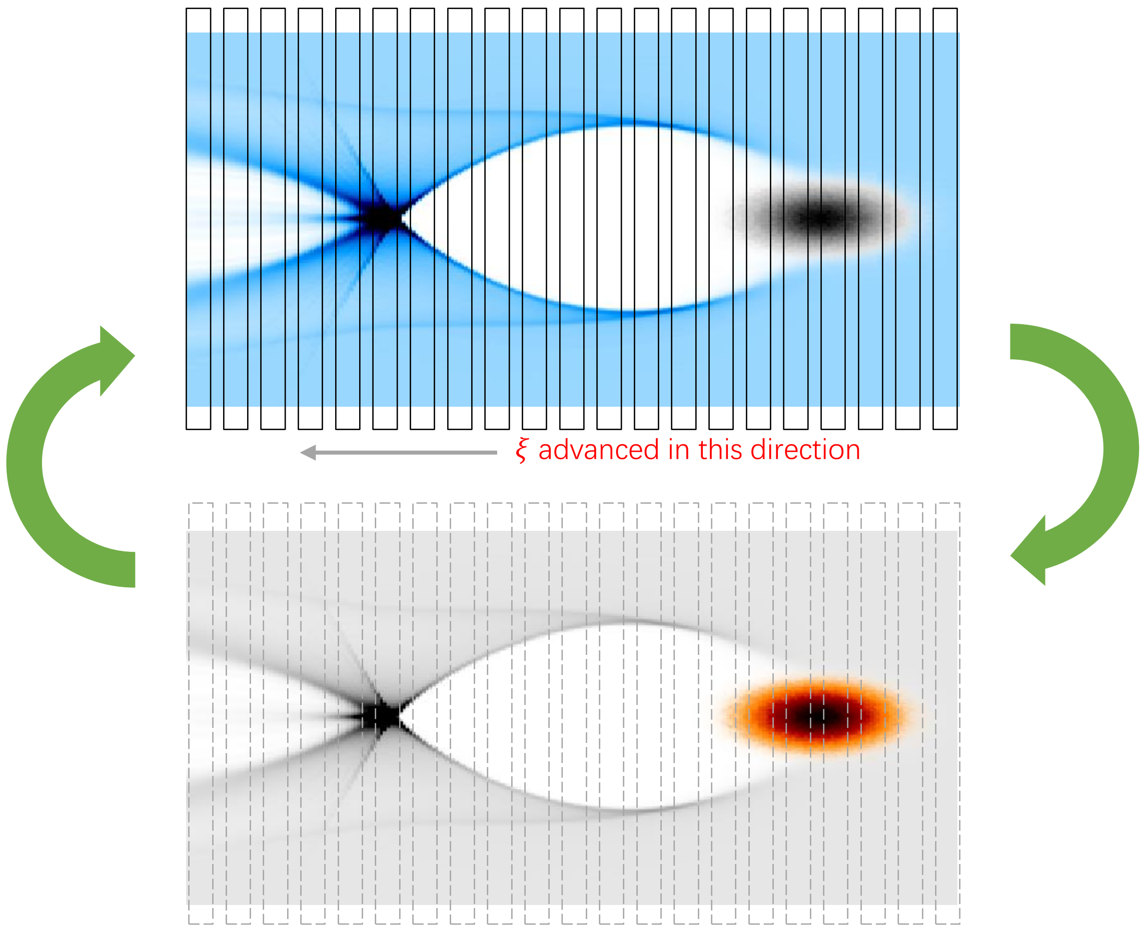
$(x, y, \xi = ct - z; s = z)$

$\partial_s \ll \partial_\xi$

Plasma: $(x, y; \xi)$

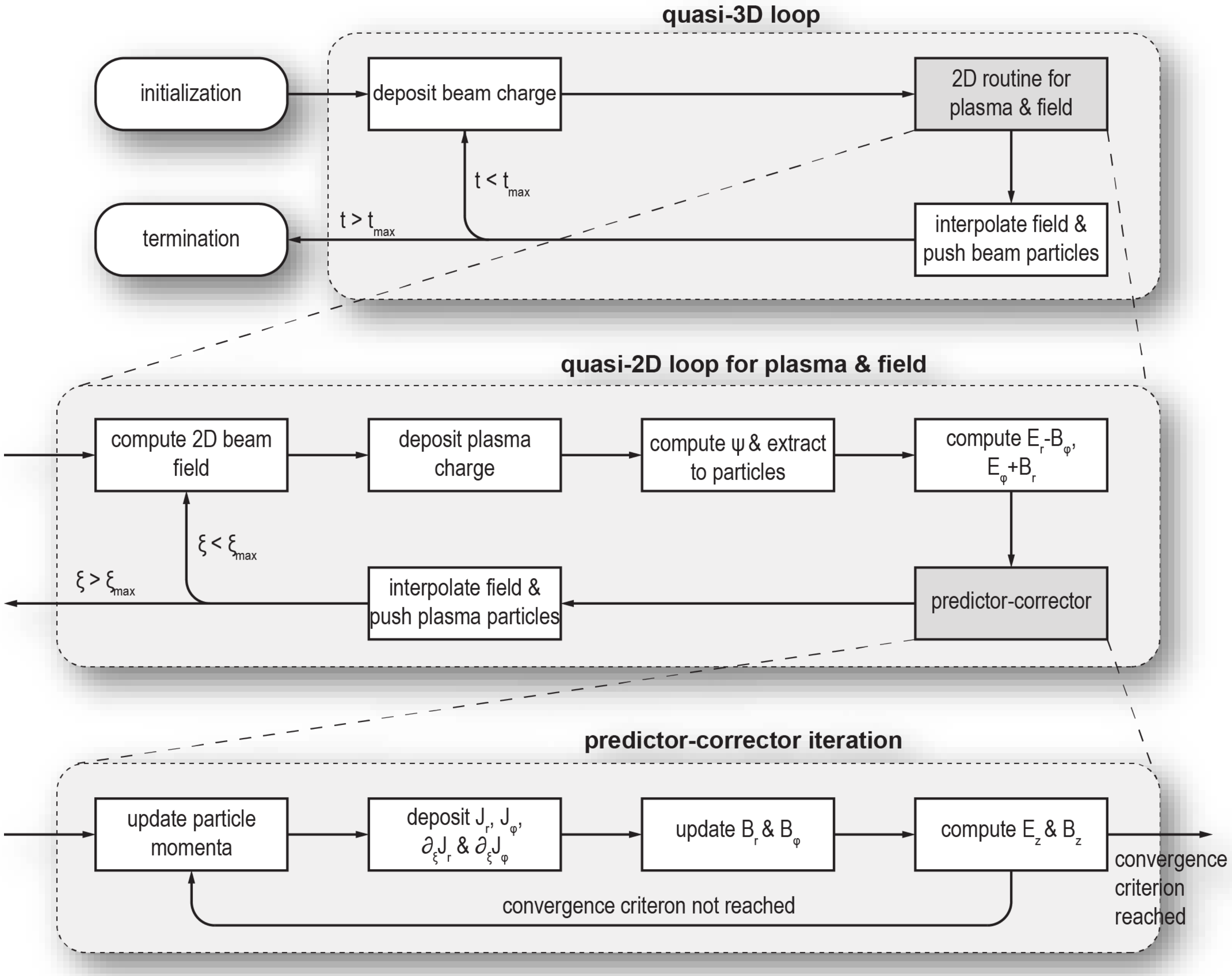
Beam: $(x, y, \xi; s)$

Quasi-2D loop: beam is frozen while plasma and field is evolved



Quasi-3D loop: field and plasma particles are frozen while beam is advanced.

Numerical workflow



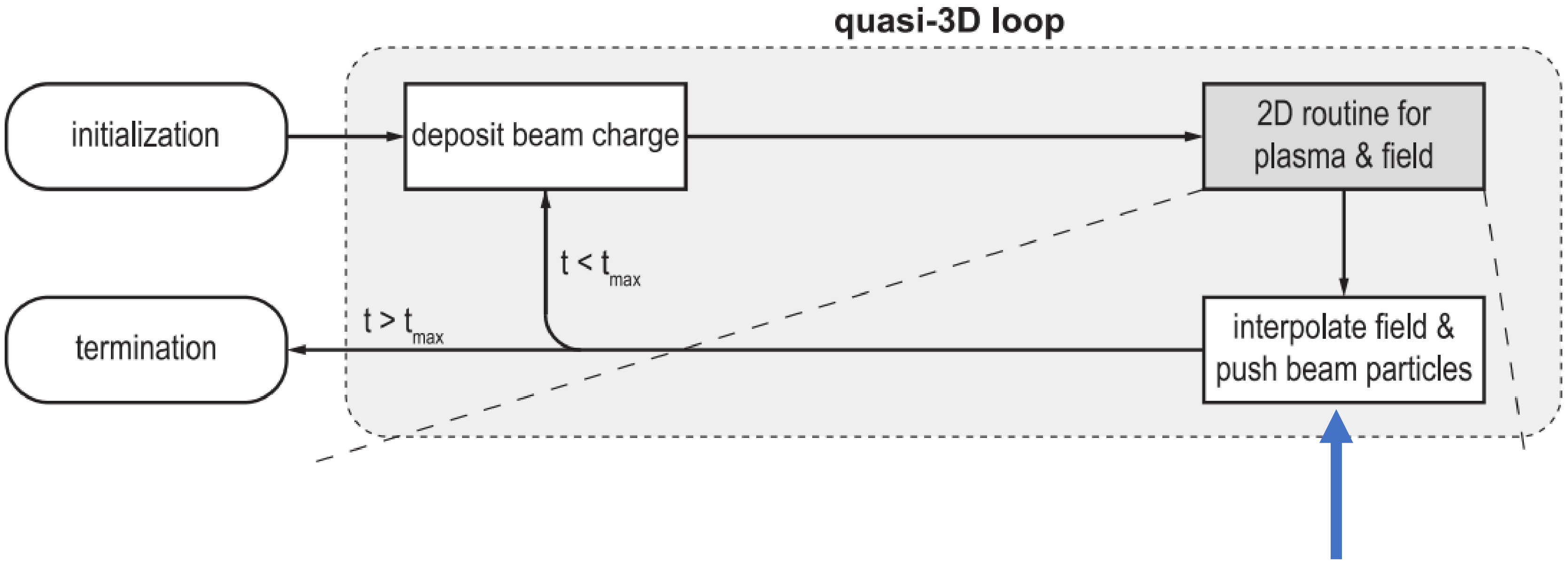
Top layer is the quasi-3D loop used for advancing charged beam particles

Intermediate layer is the quasi-2D loop used for advancing plasma particles and EM fields.

An implicit algorithm used for solving transverse components of B field iteratively.

Numerical workflow: Quasi-3D loop

The quasi-3D loop is similar to that of a standard PIC code.



$$\frac{d\boldsymbol{p}}{dt} = q\left[\boldsymbol{E} + \frac{\boldsymbol{p}}{\gamma} \times \boldsymbol{B}\right]$$

Numerical workflow: Quasi-2D loop

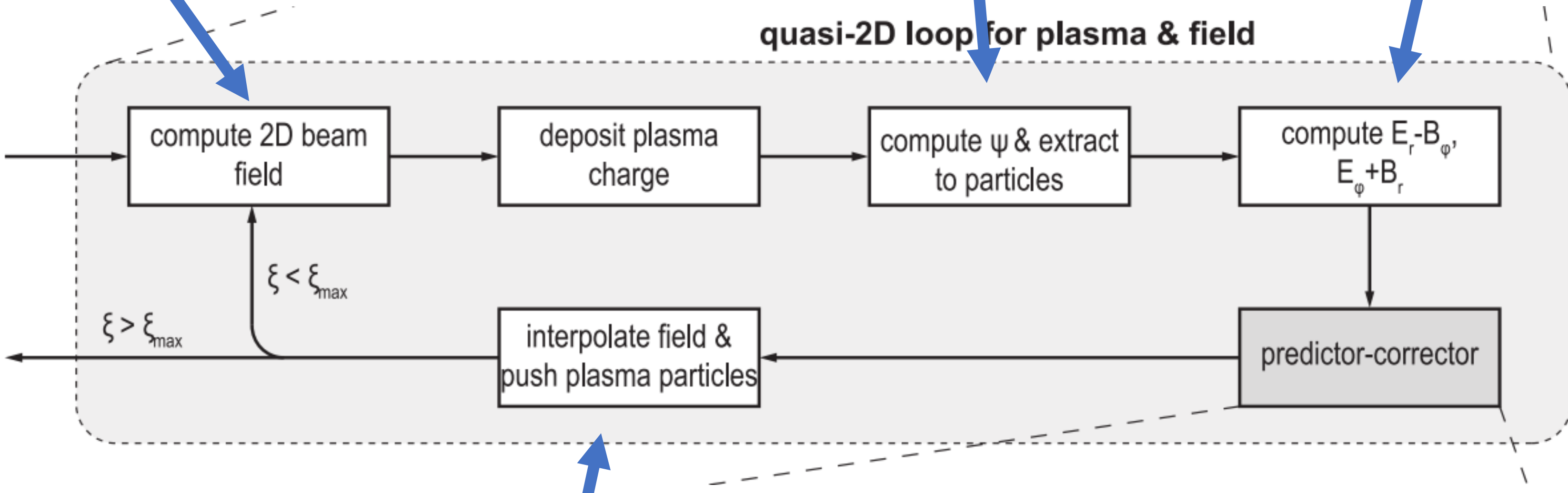
$$E^m_{r,\text{beam}} = B^m_{\phi,\text{beam}} \qquad E^m_{\phi,\text{beam}} = -B^m_{r,\text{beam}}$$

$$\boldsymbol{B}^m_{\perp,\text{beam}} = \boldsymbol{e}_r \frac{\text{i}m}{r} A^m_z + \boldsymbol{e}_\phi \frac{\partial A^m_z}{\partial r}$$

$$-\Delta_m A^m_z = J^m_{z,\text{beam}} = \rho^m_{\text{beam}}$$

$$\boldsymbol{e}_r \frac{\partial \psi^m}{\partial r} + \boldsymbol{e}_\phi \frac{\text{i}m}{r} \psi^m = (-E^m_r + B^m_\phi)\boldsymbol{e}_r - (E^m_\phi + B^m_r)\boldsymbol{e}_\phi$$

$$\Delta_m \psi^m = -(\rho^m - J^m_z)$$



$$\frac{d\boldsymbol{p}}{d\xi} = \frac{q\gamma}{1 - \frac{q\psi}{m}} \left[\boldsymbol{E} + \frac{\boldsymbol{p}}{\gamma} \times \boldsymbol{B}\right]$$

Numerical workflow: predictor-corrector iteration

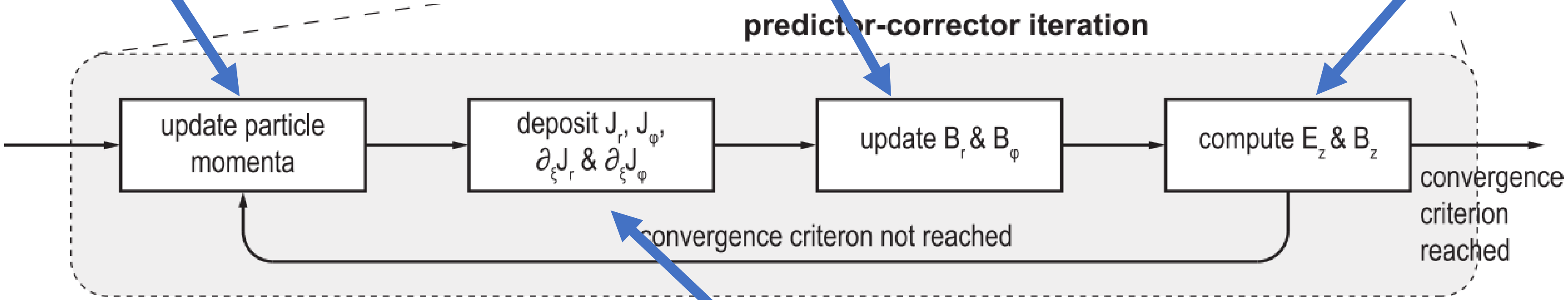
$$\frac{d\boldsymbol{p}}{d\xi} = \frac{q\gamma}{1-\frac{q\psi}{m}}\left[\boldsymbol{E}+\frac{\boldsymbol{p}}{\gamma}\times\boldsymbol{B}\right]$$

$$\triangle_m B_r^{m,l+1} - \left(1+\frac{1}{r^2}\right)B_r^{m,l+1} - \frac{2\mathrm{i}m}{r^2}B_\phi^{m,l+1} = -\left(\frac{\partial J_\phi^m}{\partial\xi}\right)^l - \frac{\mathrm{i}m}{r}J_z^{m,l} - B_r^{m,l}$$

$$\triangle_m B_\phi^{m,l+1} - \left(1+\frac{1}{r^2}\right)B_\phi^{m,l+1} + \frac{2\mathrm{i}m}{r^2}B_r^{m,l+1} = \left(\frac{\partial J_r^m}{\partial\xi}\right)^l + \frac{\partial J_z^{m,l}}{\partial r} - B_\phi^{m,l}$$

$$\triangle_m E_z^m = \frac{1}{r}\frac{\partial}{\partial r}(rJ_r^m) + \frac{\mathrm{i}m}{r}J_\phi^m$$

$$\triangle_m B_z^m = -\frac{1}{r}\frac{\partial}{\partial r}(rJ_\phi^m) + \frac{\mathrm{i}m}{r}J_r^m$$



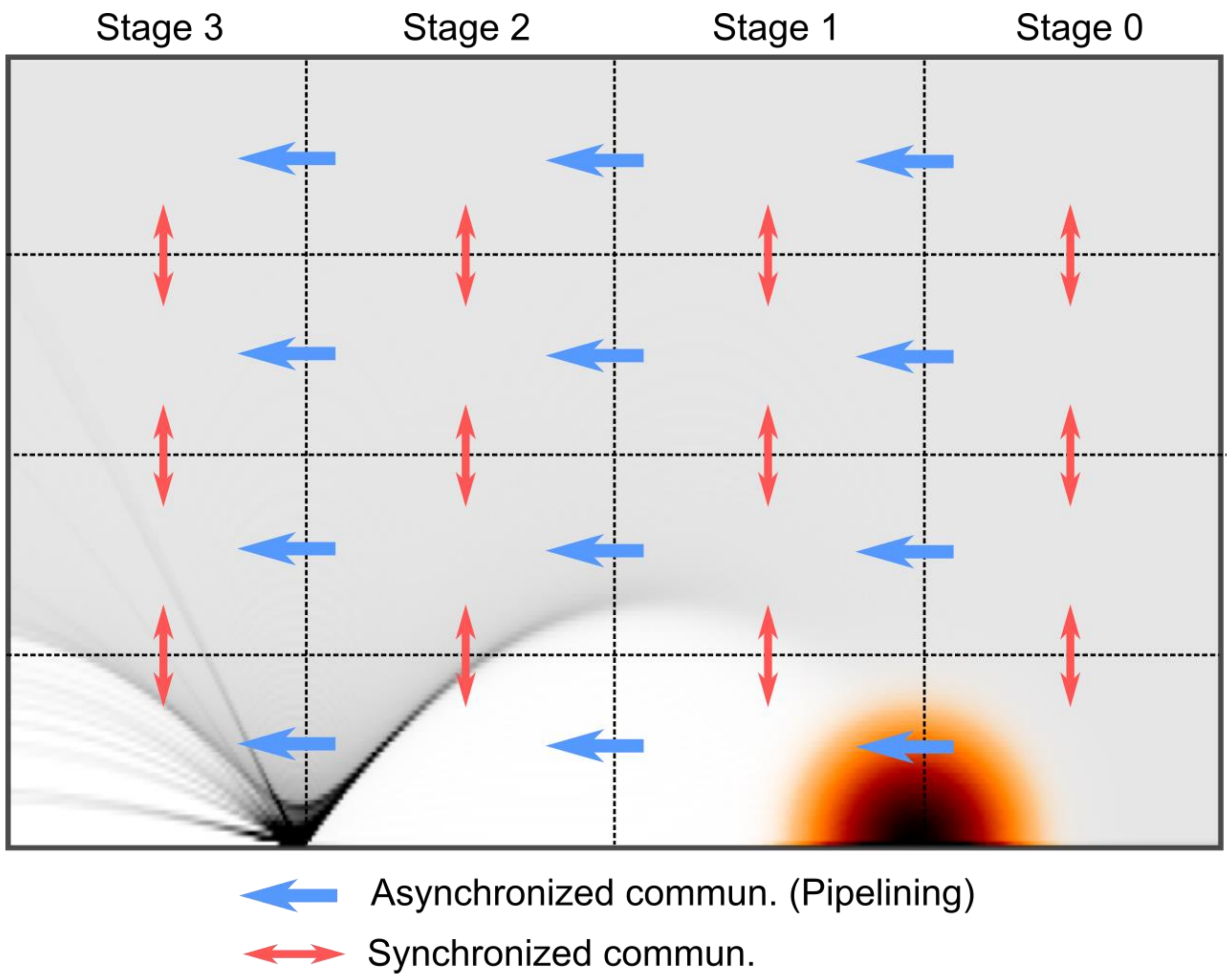
$$\boldsymbol{J}^m = \frac{1}{\mathrm{Vol.}}\sum_i \frac{q_i\boldsymbol{p}_i}{1-\frac{q_i}{m_i}\psi_i}\frac{1}{r}S_r(r-r_i)S_\phi^m(\phi_i)$$

$$\frac{\partial J_r^m}{\partial\xi} = \frac{1}{2\pi\mathrm{Vol.}}\left\{\sum_i q_i e^{-\mathrm{i}m\phi_i}\left(\frac{\mathrm{d}_\xi p_{r,i}}{1-\frac{q_i}{m_i}\psi_i} + \frac{p_{r,i}\mathrm{d}_\xi(\frac{q}{m}\psi_i)}{(1-\frac{q_i}{m_i}\psi_i)^2} - \frac{p_{r,i}p_{\phi,i}}{(1-\frac{q_i}{m_i}\psi_i)^2}\frac{\mathrm{i}m}{r_i} - \frac{p_{r,i}^2}{(1-\frac{q_i}{m_i}\psi_i)^2}\frac{1}{r}\right)\frac{S_r}{r} - \frac{\partial}{\partial r}\left(\sum_i q_i e^{-\mathrm{i}m\phi_i}\frac{p_{r,i}^2}{(1-\frac{q_i}{m_i}\psi_i)^2}\frac{S_r}{r}\right)\right\}$$

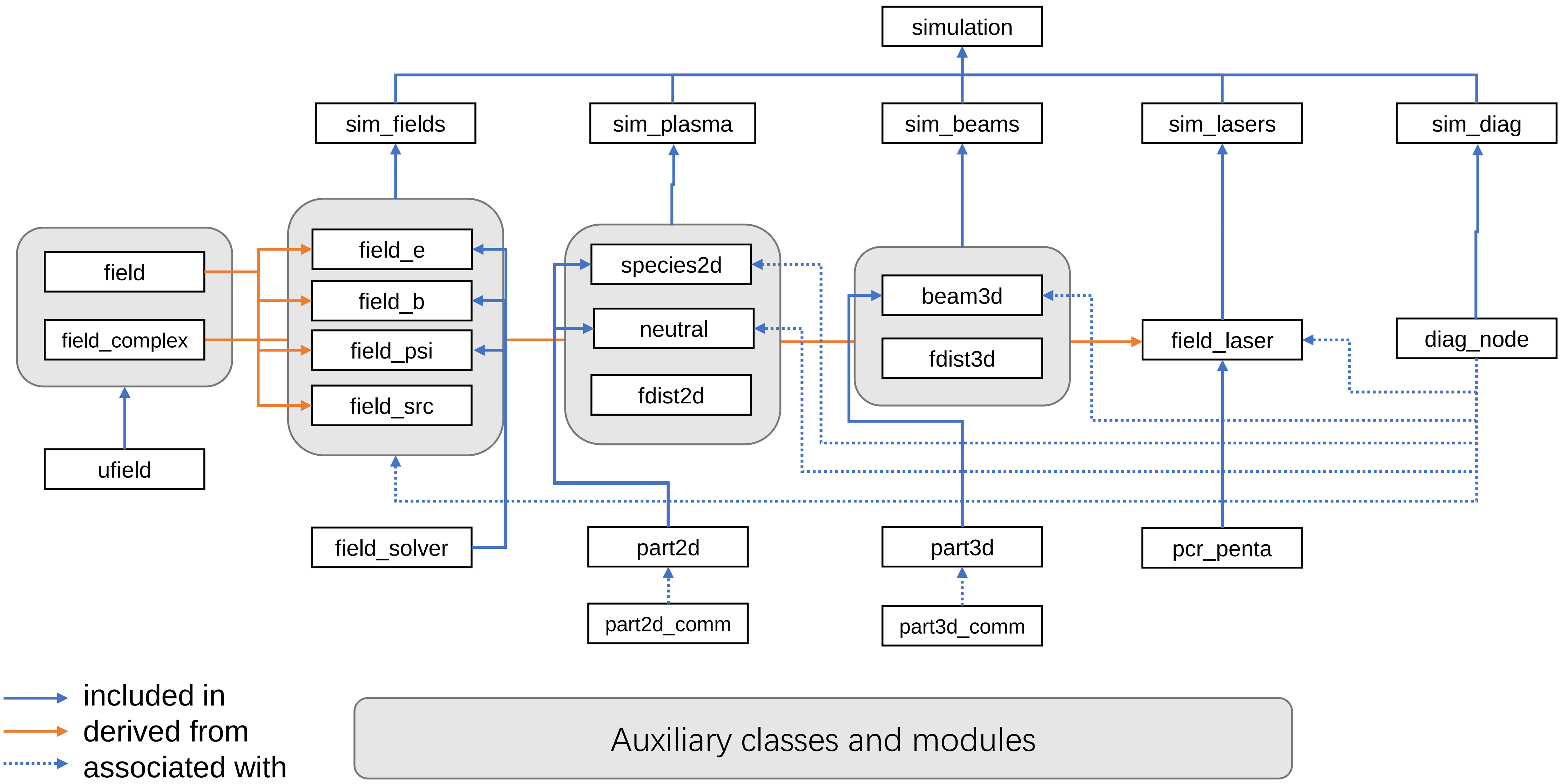
$$\frac{\partial J_\phi^m}{\partial\xi} = \frac{1}{2\pi\mathrm{Vol.}}\left\{\sum_i q_i e^{-\mathrm{i}m\phi_i}\left(\frac{\mathrm{d}_\xi p_{\phi,i}}{1-\frac{q_i}{m_i}\psi_i} + \frac{p_{\phi,i}\mathrm{d}_\xi(\frac{q}{m}\psi_i)}{(1-\frac{q_i}{m_i}\psi_i)^2} - \frac{p_{\phi,i}^2}{(1-\frac{q_i}{m_i}\psi_i)^2}\frac{\mathrm{i}m}{r_i} - \frac{p_{r,i}p_{\phi,i}}{(1-\frac{q_i}{m_i}\psi_i)^2}\frac{1}{r}\right)\frac{S_r}{r} - \frac{\partial}{\partial r}\left(\sum_i q_i e^{-\mathrm{i}m\phi_i}\frac{p_{r,i}p_{\phi,i}}{(1-\frac{q_i}{m_i}\psi_i)^2}\frac{S_r}{r}\right)\right\}$$

Parallelization

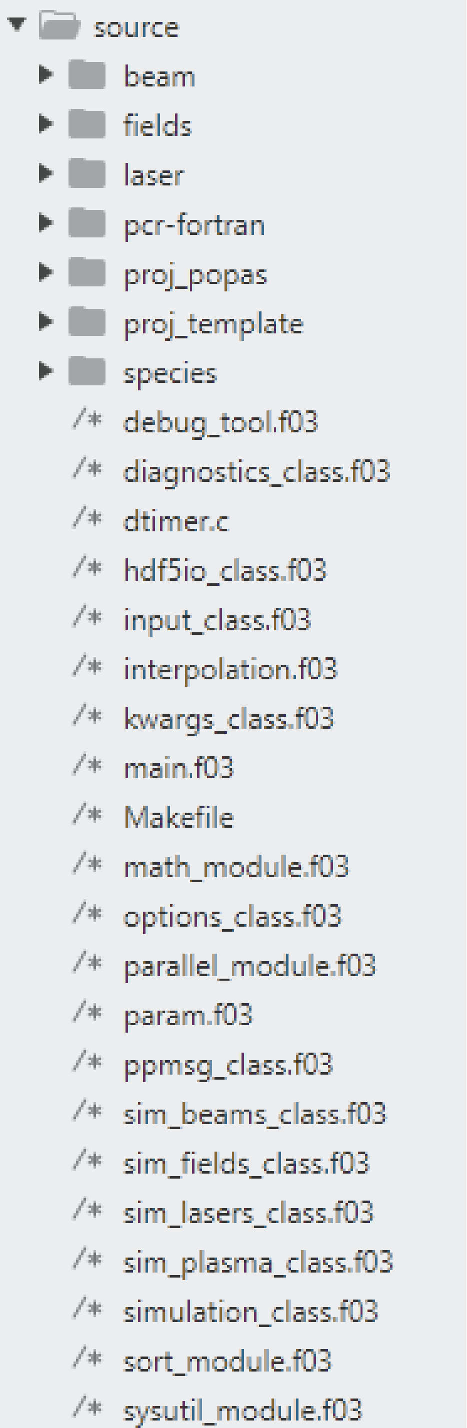
- QPAD is currently only MPI-based.
- QPAD uses hybrid parallelization.
  - Synchronized parallelization in r-direction.
  - Asynchronized parallelization in ξ-direction (pipelining).



Code hierarchy



Source code organization

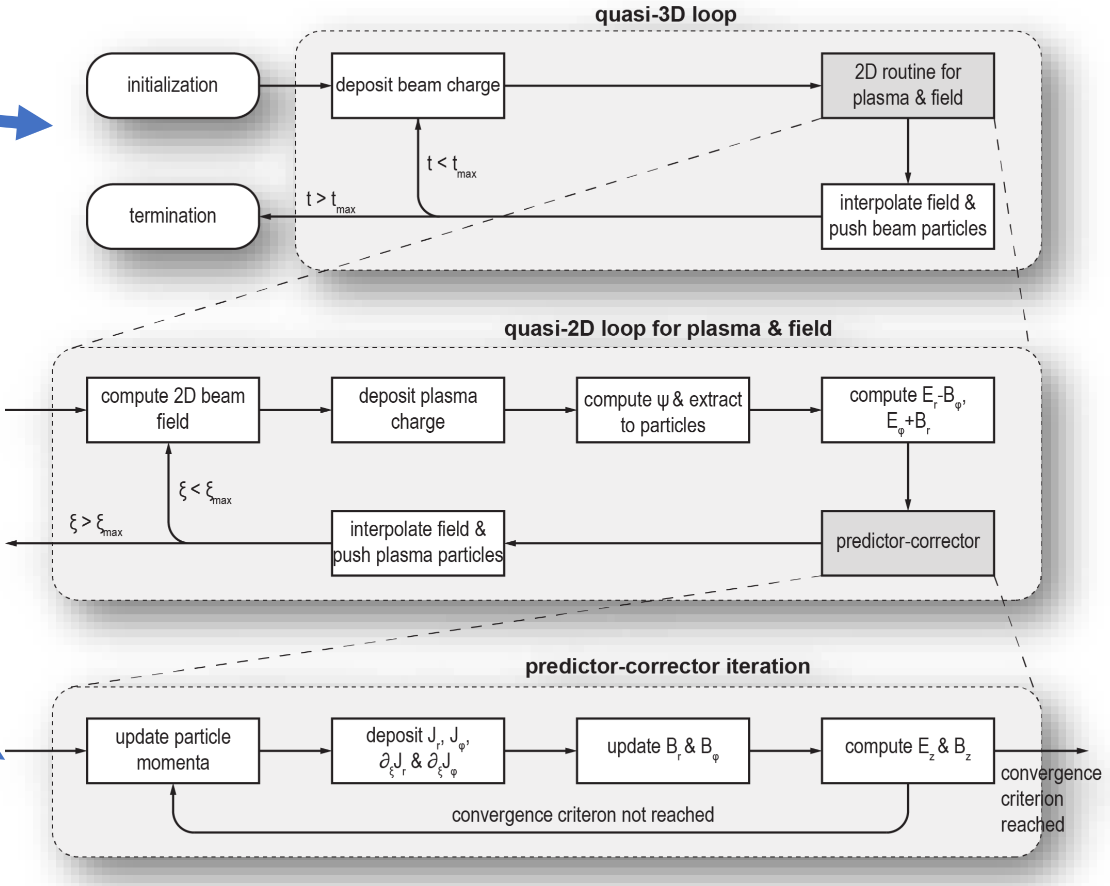


- The source code is organized in a similar way to OSIRIS.
- The contents of most files can be known from the filenames.
  - *_class.f03, *_module.f03
- Each subfolder contains a makefile.

Main loop

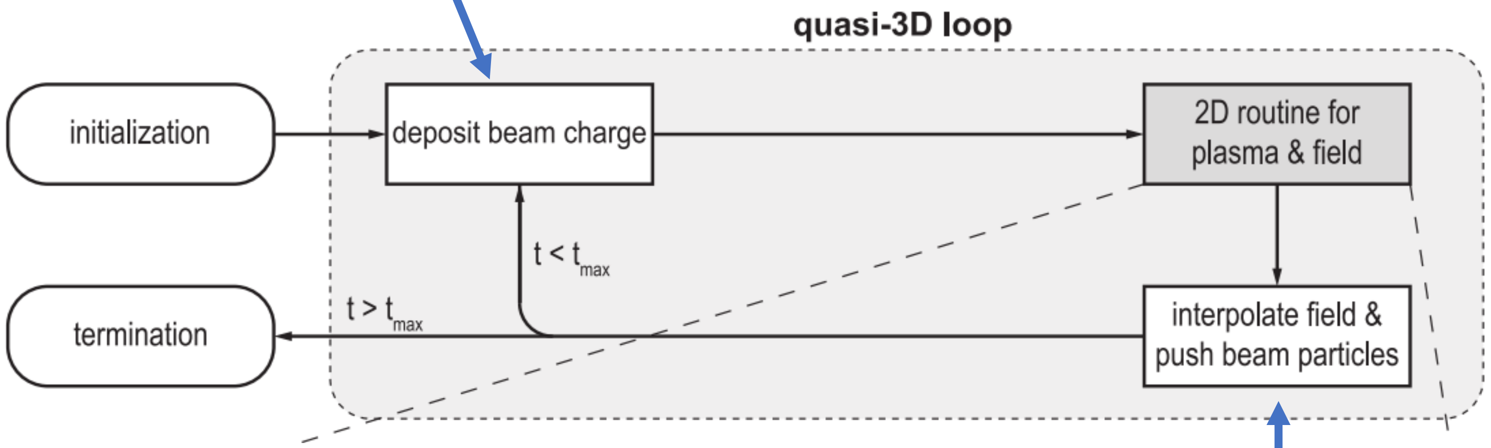
- The main structure of QPAD can be found in the subroutine "run_simulation()" in "simulation_class.f03".
- The main structure is a triple nested loop corresponding to the quasi-3D, quasi-2D and predictor-corrector loops respectively.

```
do i = this%start3d, this%nstep3d
  ! some code for 3D loop
  do j = 1, this%nstep2d
    ! some code for 2D loop
    do l = 1, this%iter
      ! some code for predictor-corrector loop
    enddo
    ! some code for 2D loop
  enddo
  ! some code for 3D loop
enddo
```



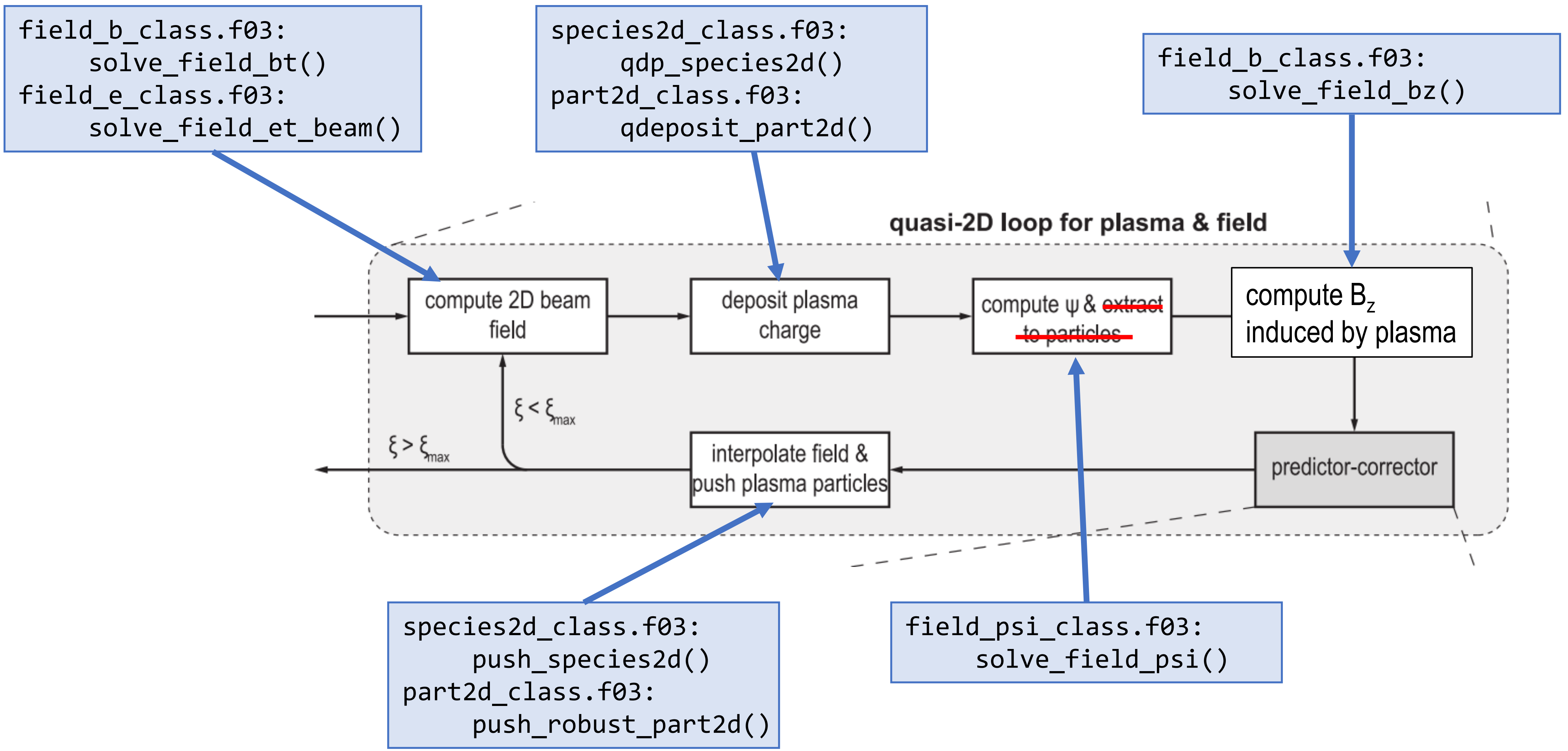
Main code of Quasi-3D loop

```
beam3d_class.f03:
     qdeposit_beam3d()
part3d_class.f03:
     qdeposit_part3d()
```

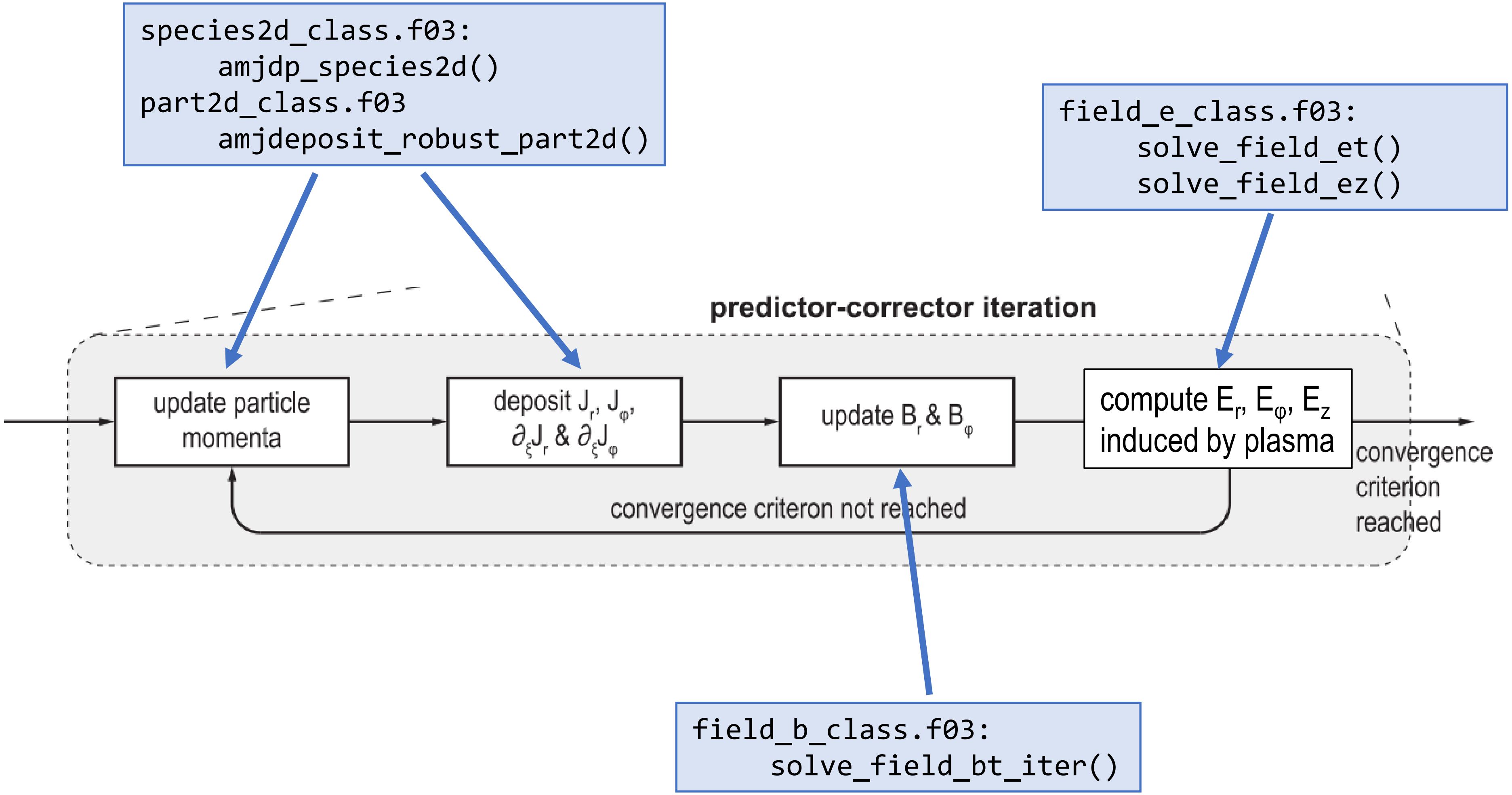



```
beam3d_class.f03:
     push_beam3d()
part3d_class.f03:
     push_boris_part3d()
     push_reduced_part3d()
     interp_emf_part3d()
     push_spin_part3d()
```

Main code of Quasi-2D loop



Main code of predictor-corrector loop



Future work

- Improve the code readability by adding more comments and instructions.
- Rewrite/redesign some interface and rename some variables.
- Optimize the code structure (especially the pipeline).
- Some classes need to be re-encapsulated.
- The current programming style is a mixture of QuickPIC 1.0, QuickPIC 2.0 and OSIRIS, which need to be unified.