2047355-TN-01-3: Co-design of PIC methods and other elements of NEPTUNE.

Ben F McMillan

April 21, 2021

This report is the deliverable associated with Milestone 6.

The NEPTUNE project scope includes simulating the dynamics of the plasma and neutrals in the edge region of the plasma. Depending on the fidelity required, and the situations of interest, this requires solving both for fluid-like motion in collisional regions, as well as essentially free-streaming kinetic behaviour for less collisional plasma constituents and neutrals.

Various computational methods to solve these kinetic problems exist, and we will here be discussing particle-based Monte-Carlo methods, where the computational markers are used to follow the trajectory of a small subsample of the physical particles. These are in more general settings referred to as Lagrangian methods and distinct from Eulerian methods, in which distributions are evolved on a fixed grid. Particle-In-Cell methods, and Monte-Carlo methods designed to evolve the neutral population, are quite similar in concept, and we will refer to them as PIC methods for the purposes of this document. We restrict the discussion to using particle methods to solve kinetic (rather than fluid) problems.

Integrating these methods with other components of the NEPTUNE project may require careful planning. Interaction of the proposed particle method with two other work packages was highlighted in the bidding process as needing explicit consideration, and a discussion of these interactions as well as a plan for going forward the principal purpose of this document.

Firstly, because the gyrokinetics work package will determine the appropriate Fokker-Planck-Maxwell equations that well-magnetised particles follow, it is critical to determine whether the particle methods can be made to support these equations. The gyrokinetics work package is also investigating a moment-based numerical scheme to discretise these equations in a way suitable for implicit solution; we will explain the relationship between this scheme and the moment-based particle discretisation that is designed to reduce noise, but also can be leveraged as part of an implicit method.

Secondly, the NEPTUNE project has selected a spatial representation (for various plasma quantities and electromagnetic fields) based on finite elements, on an unstructured grid, with a view to using curved element boundaries in order to conform to the magnetic field topology and the physical first wall of the tokamak. This is intended to be provided through the Nektar++ library. The advanced particles methods implemented in this exploratory stage are however based around simple uniformly spaced regular Cartesian meshes. We outline design considerations of a particle solver interfaced with Spectral/hp curvilinear finite element methods. We also explain what features of the representation of the electromagnetic fields are desirable for accurate particle tracing.

1 What are PIC methods useful for?

The relative computational advantages of various numerical schemes for solving kinetic problems will determine where each one is used, and for which class of particles, or where kinetic solution is feasible at all.

Particle based methods are intuitively attractive for solving kinetic problems, but Eulerian methods converge faster as a result of the Monte-Carlo sampling used in PIC methods. PIC methods are often favored over Eulerian methods when the following conditions apply

- Three velocity space directions need to be resolved.
- Collision operators are easily written in the form of a stochastic differential equation, but not as a PDE.
- It is difficult to find an appropriate velocity-space grid to represent the particle distribution function on (eg. multiple-beam distributions).
- It is difficult to find an optimal spatial grid (as PIC codes are less severely impacted by running on a non-optimal spatial grid).

In general PIC methods tend to be relatively easy to implement and parallelise compared to Eulerian methods. This, and a willingness to tolerate some statistical noise, mean that early examples of complex plasma kinetic codes tend to be PIC-based (e.g. witness the evolution of core and edge gyrokinetic codes). PIC codes are dominant for neutral particle pushing and for 6D Vlasov-Maxwell (i.e. non gyrokinetic PIC) because except in special cases 6D Eulerian codes are computationally uncompetitive. For example, if full fast particle orbits need to be tracked to determine when and where their large orbits hit the wall, PIC is a natural choice for this problem. Some general discussion on particle methods is given in ref. [1].

In edge plasmas, fluid modelling of at least some of the species and some of the regions is essential. As a result, Eulerian and Lagrangian kinetic models need to be coupled to fluid models. To provide maximum predictive and numerical performance, it is highly desirable for the Neptune project to be able to choose where and when to use each kind of model in a flexible and correctly coupled way. The remainder of the document described interfaces between the numerical discretisations of these models.

2 Edge-relevant Gyrokinetic formalism

We briefly introduce the baseline mathematical formulation (independent of discretisation) of the Vlasov equations for the initial stage of the NEPTUNE project.

3 Reformulation of kinetic equations in terms of closure to fluid equations

The Fokker-Planck equation for the evolution of a species distribution function $f(\mathbf{Z})$, with $\mathbf{Z}$ the phase space coordinate (representing position and velocity coordinates) may be written in the form

$$\frac{\partial f}{\partial t} + \nabla_{\mathbf{Z}}.\left[\dot{\mathbf{Z}} f\right] = S(f) \tag{1}$$

with $S(f)$ a generalised source term (including self and non-self collisions). For phase-space conserving equations of motion ($\nabla.\dot{\mathbf{Z}} = 0$), this may be written in the conservative form

$$\frac{\partial f}{\partial t} + \dot{\mathbf{Z}}.\nabla_{\mathbf{Z}} f = \frac{df}{dt} = S(f). \tag{2}$$

As in the Eulerian representation, it is often useful to separately represent the evolution of a background distribution parameterised using fluid moments, and the remaining kinetic response. That is, we introduce a splitting $f(\mathbf{Z},t) = f_0(\mathbf{Z},t) + g(\mathbf{Z},t)$.

In standard (non-transformed) coordinates the phase space vector may be decomposed into velocity and position as $\mathbf{Z} = (\mathbf{R},\mathbf{v})$. We then choose the parameterisation

$$f_0(\mathbf{Z},t) = \frac{n(\mathbf{R},t)}{[2\pi kT(\mathbf{R},t)]^{3/2}} \exp\left[\frac{m\{\mathbf{v} - \mathbf{v}_0(\mathbf{R},t)\}^2/2}{kT(\mathbf{R},t)}\right] \tag{3}$$

which is, again, a shifted Maxwellian, where the time and space varying parameters $\mathbf{v}_0, T$ and n, which are the mean velocity, temperature and density associated with the distribution f_0, are as yet not specified.

We require that the moments 1, $\mathbf{v}$ and v^2 of g vanish; this is both a condition on the initial condition, as well as the evolution of g. By taking moments of the Fokker-Planck equation, one then obtains three fluid equations for the time evolution of f_0.

For the moment, we restrict the analysis to considering a Lorentz-like force $\mathbf{F}$ (with the property that energy transfer to the fluid is only through acceleration). In this case the moment equations may be written in a conservative form (c.f. eqs. 57-59 of Brunner et. al., which are essentially identical but for specific forms of force/collisions, and with momentum equation in convective form) as

$$\frac{\partial n}{\partial t} + \nabla.[n\mathbf{v}_0] = \int d^3v\, S, \tag{4}$$

$$\frac{\partial mn\mathbf{v_0}}{\partial t} + \nabla.[mn\mathbf{v}_0\mathbf{v}_0 + \mathbf{P}_1] + \nabla[nkT] - n\mathbf{F}_0 = \int d^3v\, m\mathbf{v}S, \tag{5}$$

$$\frac{\partial[3nkT + mnv_0^2]}{\partial t} + \nabla.\left[\mathbf{v_0}\{5nkT + mnv_0^2\} + 2\mathbf{P_1}.\mathbf{v}_0 + \int d^3v\, m(\mathbf{v} - \mathbf{v_0})(v - v_0)^2 g\right]$$
$$-2n\mathbf{F}_0.\mathbf{v_0} = \int d^3v\, mv^2 S. \tag{6}$$

where $\mathbf{F}_0$ is the mean per-particle force, $q[\mathbf{E} + \mathbf{v}_0 \times \mathbf{B}]$ for a Lorentz force and

$$\mathbf{P}_1 = \int d^3v(\mathbf{v} - \mathbf{v_0})g(\mathbf{v} - \mathbf{v}_0) \tag{7}$$

is the anisotropic part of the pressure tensor (i.e. the Stress tensor). The fluid viscosity and heat conduction which would be found through an ordering argument in the collisional limit are here expressed in terms of the kinetic correction g.

In the last two moment equations, the quantities in the time derivative are momentum and kinetic energy, but these may be expanded and combined to find equations in the 'convective form' where the only time-derivatives that appear are the velocity and temperature instead.

Note that the kinetic correction (closure) term in the energy equation would normally be written in terms of a collisional closure in a fluid scheme, as diffusive heat fluxes, so switching between collisional and fully kinetic solutions to these moment equations can be achieved in a natural way.

The Maxwell equation source term is a function of current, which may be written in terms of the fluid quantities as $qn\mathbf{v}_0$, thus the fluctuation g only enters as a closure term in the energy equation. In the highly collisional limit, where the heat fluxes are negligible, we recover a conventional (multi-species) set of plasma fluid equations simply by setting $g \to 0$.

For time evolution of g, we require for consistency

$$\frac{dg}{dt} = -\frac{df_0}{dt} + S(g + f_0). \tag{8}$$

We have considered the Vlasov-Maxwell system here, but conceptually gyro-Vlasov-Maxwell follows the same derivation path, and although the fluid equations differ somewhat, the equation for g in the symbolic form above is equivalent for Vlasov and gyroVlasov equations. In 1D, with the simulation aligned along a constant background field, which is the initial configuration to be considered, these systems are identical. Note that the above equation is also valid for neutral particles.

It is beyond this point that the proposed methods in NEPTUNE differ. Actually, the fluid equations, and the coupling to the gyro-Maxwell or gyro-Poisson equations is identical, so the same fluid and field solvers may be used. However, the representation of g differs. Firstly, and obviously, in the Eulerian approach, g is represented using coefficients on a fixed spatial grid, whereas in the particle-based approach, g is represented using markers.

Secondly, the Eulerian method described in Ref. [2] performs a transformation on velocity space, shifting and rescaling the local coordinate $\mathbf{V} = M(\mathbf{v}, \mathbf{v}_0, T)$ such than $F_0(V) = f_0(M^{-1}V)/n$ is an unshifted Maxwellian of temperature and density unity. The grid-based or spectral Eulerian scheme is then written in terms of $\mathbf{V}$; this allows the grid to adapt to regions of low temperature by refining the grid resolution so the spacing is constant in terms of local thermal velocity.

Although it would also be possible to perform this transformation in the Particle-based scheme, it is not helpful, because the effective velocity resolution depends on where the markers are in velocity space, rather than their coordinate labelling. Particle schemes adapt to velocity space automatically because the marker phase-space density remains (in a statistical sense) proportional to the particle phase-space density, so markers in cold regions naturally have low typical velocities.

4 Particle-specific implementation of the moment-based scheme

We briefly summarise the method to be implemented; for Vlasov-Maxwell systems, this was implemented in Ref. [3].

We consider N markers (computational macroparticles) loaded in phase space with positions $Z_i(t)$ with a phase space density

$$K(\mathbf{Z}) = \lim_{N\to\infty} \int d\mathbf{V} \sum_N \delta^6(\mathbf{Z} - \mathbf{Z_i(t)}) \tag{9}$$

that represents how likely we are to find a marker in a small region of phase space near $\mathbf{Z}$. Due to Liouville's theorm, we find that $dK/dt = 0$. We store the initial value $K_i = K[\mathbf{Z}_i(0)]$ at each marker position and identify the effective volume of phase space associated with each marker as $1/K(\mathbf{Z})$.

To represent the fluctuating distribution function, we define a weight g_i at each particle location

$$g(\mathbf{Z}, t) = \sum_N \frac{1}{K_i} g_i(t) \delta^6(\mathbf{Z} - \mathbf{Z}_i(t)) \tag{10}$$

with $g_i(0) = g(\mathbf{Z}_i(0), 0)$ and

$$\frac{d\mathbf{Z}_i(t)}{dt} = \dot{\mathbf{Z}}(\mathbf{Z_i}, t) \tag{11}$$

and if we evolve $g_i(t)$ via

$$\frac{dg_i}{dt} = - \left.\frac{df_0}{dt}\right|_{\mathbf{Z}=\mathbf{Z_i}} + S(\mathbf{Z_i}) \tag{12}$$

then by integrating over any volume K, we can show that in the large-marker limit, this will satisfy

$$\frac{dg}{dt} + \frac{df_0}{dt} - S(g + f_0) \equiv Q = 0. \tag{13}$$

in the weak form, that is $\lim_{N\to\infty} \int_K d^3vQ = 0$. That is, unlike standard PIC, which trivially satisfies $df/dt = 0$ exactly for the numerical distribution and numerical orbits, the numerical Vlasov equation is satisfied only in the weak form and large particle number limit when using control-variates[4].

Note that this the standard control variates[5, 6] method. Confusingly, this is often referred to as the 'δf' method, but 'δf' is also used to denote methodologies that are valid only when the fluctuation is small. There is no such approximation used here; there is no formal restriction on the size of g, although when f and g are the same size, the numerical advantages may be lost.

The principal difference between this control-variates method and that implemented in codes such as ORB5 and EPOCH is time-evolution of the background distribution function f_0 (however, a project is underway to implement background density variation in ORB5).

In a Galerkin method, moments of the particle distribution appear multiplied by basis functions. For example, consider a representation of density

$$n(x) = \sum_k n_k \Lambda_k(x) \tag{14}$$

where n_k are the nodal density values and Λ the basis function. In the weak form, this may be equated to the velocity-space integral of the particle representation of the distribution function f. Although this may straightforwardly be extended to control-variates, to simplify the presentation, we will use a standard-PIC representation, without background extraction, and represent the full distribution on markers using f_i (for $f_0 = 0$ we have $f_i = g_i$). We then have

$$\int dx\Phi(x) \int dvf = \int dx\Phi(x) \int dv \sum_i \frac{f_i}{K_i} \delta^6(\mathbf{Z} - \mathbf{Z_i}) = \int dx\Phi(x) \sum_k n_k \Lambda_k(x) \tag{15}$$

for all trial functions $\Phi(x)$ in the finite element representation: since the basis functions span this space, it is sufficient if this equality is satisfied for $\Phi(x) = \Lambda_h \forall h$. Because, if we choose trial functions in the same space as the density representation, we have $\Phi(x) = \sum_h n_h \Lambda_h(x)$, this gives us

$$\forall h, \int dx \lambda_h \sum_i \frac{f_i}{K_i} \delta(\mathbf{x} - \mathbf{x_i}) = \int dx \Lambda_h(x) \sum_k n_k \Lambda_k(x) \tag{16}$$

or

$$\forall h, \sum_i \lambda_h(x_i) \frac{f_i}{K_i} = \sum_k M_{hk} n_k \tag{17}$$

where we have defined a mass matric M_{hk} representing the integral on the RHS. We may then find a FEM representation of the density by performing the sum on the LHS and inverting the mass matrix.

In control-variates form, if the background density may be represented spatially using the FEM basis functions, this equation (eq 17) holds with f_i replaced by g_i and n_k replaced with the density fluctuation.

5 Computational advantages of moment-based schemes.

This moment-based scheme (whether in Eulerian or Lagrangian form) allows a close connection to the fluid equations, and allows the fluid and field equations to be partially decoupled from the kinetic equations.

One way this can be exploited is that in regions where kinetic effects are minimal, the kinetic equations no longer need to be solved at all.

An additional advantage is that the coupled Vlasov-Maxwell system can be quite *stiff* in the sense that there are oscillations and relaxations on much faster timescales than the dynamical timescales of interest. For example, Alfven waves propagate at close to the speed of light in certain edge regions, whereas even the highest energy particles typically propagate at one percent of this speed. In the usual cases, these rapid oscillations are associated with fluid processes (i.e. waves) and the kinetic effects, such as nonlocal thermal conduction, appear on longer timescales. It is computationally advantageous to evolve or relax these waves by solving low-dimensional (3D) implicit or explicit fluid equations rather than (5-6D) kinetic equations. Even if the fluid motion needs to be solved explicitly, this simplification can allow a massive computational speedup.

For the particle method, the moment-based scheme also allows for noise reduction by extracting out the background distribution and integrating it analytically. That is, consider the moment

$$n = \int dV f = \int dV f_0 + \int dV g. \tag{18}$$

Because f_0 may be evaluated analytically, only the integral over g needs to be evaluated using a Monte-Carlo method; and the RMS error of the Monte-Carlo integration scales like the amplitude of g, so if g is small, then this can lead to a radical decrease in the number of markers needed to attain a specific accuracy. For core plasma simulations $< g^2 > / < f^2 > \sim 10^{-4}$ this allows 10,000 times fewer markers to be used. In the edge, fluctuations are large, but relatively high collisionality means that the departures from Maxwellian are not necessarily large (e.g. low density high velocity tails are present). Maintaining noise reduction over long simulation timespans requires additional machinery to maintain good sampling of phase space and avoid excessive *weight spreading.*

6 Proof of principle implementation

A 1D, two-moment model is implemented in the *minepoch* code that is on the github repository.

In the test case setup (an input file 'singlestream.deck' is given for this case) is that a low temperature single-species collisionless neutral species is given an initial density and velocity perturbation of the form

$$n = n_0[1 + 0.3\sin(4\pi x/L_x)], \quad v = v_0[1 + \sin(6\pi x/L_x)]. \tag{19}$$

The details of parameters for the test case setup are described in detail in the testing code. The purpose of this test case is a proof of principle demonstration that the control variates method is able to reduce the statistical sampling error in a case with order one density fluctuations (as one might find in the edge): this is an advancement over control variates schemes used in the core of tokamaks, which cannot handle large variations.

The low temperature means the energy equation is not needed, and as the evolution is force free, a simplified momentum evolution equation is sufficient.

For this setup, because the evolution is force-free, the method of characteristics may be used to find the solution analytically (in an implicit form).

We show the time-evolution of the density in the simulation in figures 1, 2 and 3.

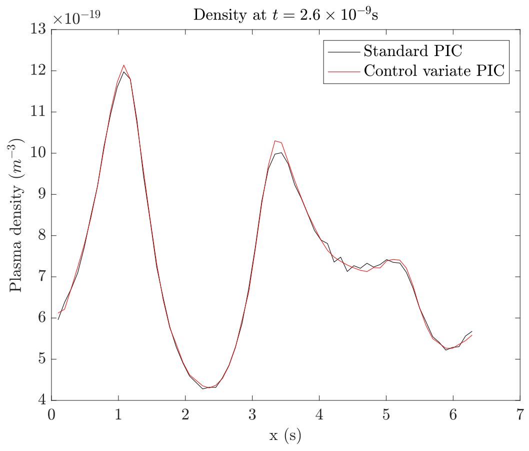

Figure 1: Density versus position at an intermediate simulation time.

At $t = 2.6 \times 10^{-9}$ s, with $f = f_0 + g$ in the control-variates simulation, we have $< g^2 > / < f^2 >= 1.1 \times 10^{-3}$. That is, despite the variation of density by a factor of 2 in the x domain, the kinetic correction to the particle distribution g is ~ 30 times smaller than the overall distribution function f. The squared noise amplitude scales like $< g^2 > /N$, so either one may use 1000 times fewer markers in the control-variates simulation for the same noise, or 1000 times lower squared noise amplitude with the same number of markers.

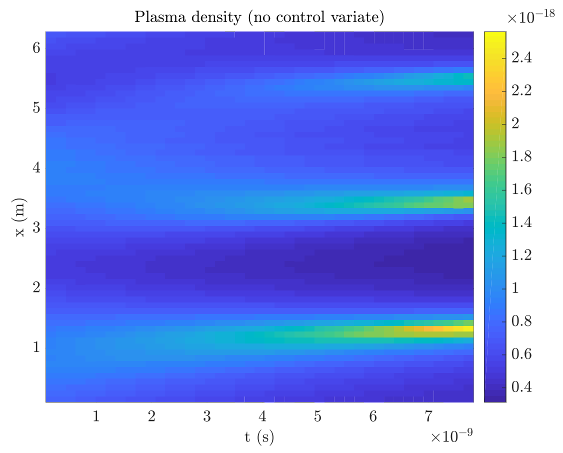

Figure 2: Colour plot of density versus position and time in a standard (no control Variates) PIC simulation.

7 Finite elements and particle methods, gyrokinetic examples.

Finite element method meshes are fairly regularly used as a basis for particle-in-cell codes. Two examples of gyrokinetic particle-in-cell codes are the XGC[7] code and ORB5[8]; the author of this document is a developer of the ORB5 code.

The ORB5 code used a logically Cartesian tensor-product B-spline representation in magnetic coordinates. The elements are curved in laboratory Cartesian space, and this allows a simple structured grid that nonetheless conforms to the curved magnetic field topology of the tokamak core. This approach is not able to handle the X-point of scrape-off layer region directly.

We have magnetic coordinates (s, θ, ζ), which are toroidal coordinates with $s \in [0, 1]$ representing the radial distance from the magnetic axis, χ the straight-field line poloidal angle, and ζ the (geometrical) toroidal angle.

Taking the equal spaced case for simplicity, fields are discretely represented as:

$$\phi(s,\theta,\zeta) \sum_{i=[0,N_i]} \sum_{j=[0,N_j]} \sum_{k=[0,N_k]} \phi_{i,j,k} \Lambda \left[\frac{s-s_i}{\delta s}\right] \Lambda \left[\frac{\theta-\theta_j}{\delta\theta}\right] \Lambda \left[\frac{\zeta-\zeta_k}{\delta\zeta}\right] \quad (20)$$

Here, $\Lambda(x)$ are (quadratic or cubic) B-spline functions (Fig. 4), which serve as compact-support basis functions for the finite-element scheme. For quadratic basis functions, the representation has continuous values and first derivatives (i.e. it is C^1) and for cubic, the second derivatives are also continuous (it is C^2).

XGC uses a block-structured regular mesh, also based on low-order finite elements. The blockwise decomposition allows the code to represent, using one

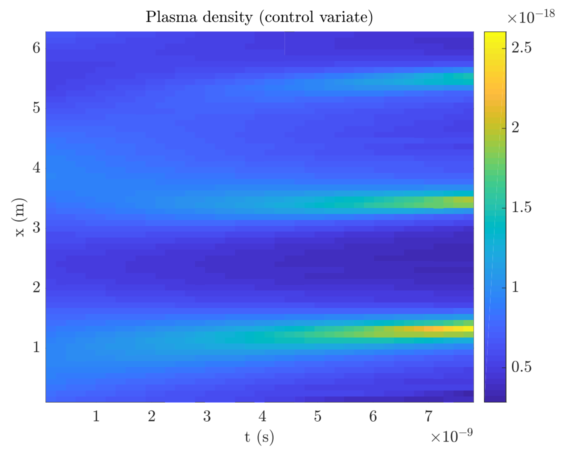

Figure 3: Colour plot of density versus position and time in a control-Variates PIC simulation.

block the core plasma using elements that conform to magnetic field lines to capture the strong anisotropy. The block region around the X-point is not required to conform to the magnetic geometry, but the low poloidal field in this region simplifies the handling of anisotropy. Additional blocks in the scape off layer allow the full geometry to be handled.

Particle-field operations are seen in a radically different way in these finite-element based PIC codes to traditional PIC codes, which use finite difference or volume concepts. Traditional PIC (e.g. EPOCH) considers the markers to have a smooth 'cloud of charge' associated with all the physical particles the marker is representing, leading to a 'shape function' which may be used to define a continuous particle density throughout the simulation domain. This smooth field may then be evaluated at grid points in a finite difference scheme, or integrated over grid volumes in a finite volume scheme (e.g. EPOCH). Finite-element based PIC schemes, on the other hand, consider the markers as point-particles without an explicit spatial extent. However, functionally the smooth field representation in a finite element code plays a very similar role, and the effective charge density as represented on the FEM grid is smooth.

All these codes involve direct evaluation of the fields at the particle positions, which is logically straightforward. First, one determines which grid cell the particle is in: in ORB5 the particles coordinates are evolved in magnetic coordinates, and these coordinates can be individually mapped to grid indexes in each direction. The compact support then allows an evaluation using a relatively small number of basis functions. For tensor-product based schemes, about two multiplications are needed per contributing basis function, so even though in the cubic case 64 basis functions contribute in 3D (cubic spline elements span 4 cells in each direction, see fig 4), the per-evaluation cost is not extreme.

For a fully unstructured, curvilinear grid, one can in principle evaluate the

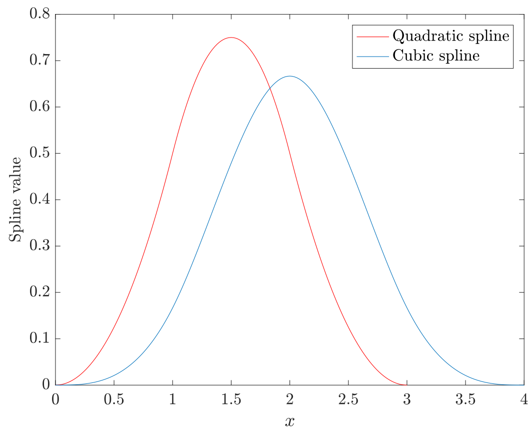

Figure 4: B-Splines on a unit-interval grid.

field at arbitrary locations, but this may not be straightforward or efficient. It may in general be complicated or inefficient to determine which grid cell a particle is in based on its global coordinate position. The mapping used to allow distorted element positions is the forward mapping from element-wise local coordinates to global Cartesian coordinates and the inverse mapping may be difficult to compute, so even if the grid cell can be determined, the local coordinate position in the cell may be annoying to compute. In this case, it may be desirable to keep track of the cell-local coordinates of the particle, and, when it exits the cell, to update which cell it is in based on its crossing of boundaries.

Using a cell-local coordinate scheme complicates somewhat the particle timestepping in the sense that:

- The numerical equations may have to be rewritten in a general curvilinear form.
- The coordinates are continuous between cells, but not necessarily smooth, which will lead to low order of time-convergence of timestepping schemes without any additional measures being taken.
- One must keep track of which cell the particle is in and hand it between cells during the timestep, possibly across multiple cell boundaries.

Overall, the use of cell coordinates rather than global coordinates creates a much tighter coupling and more complicated interface between the particle and FEM code elements. But tight coupling may be required in any case to handle practical numerical issues such as parallelisation.

8 Timestepping accuracy and continuity

A particle timestepping equation symbolically of the form

$$\frac{d\mathbf{Z}}{dt} = F(\mathbf{Z}(t), t) \tag{21}$$

usually involves a dependence on certain fields (density fields leading to particle drag, electromagnetic fields for charged particles) that are represented, in the case of Nektar++, using the FEM. Consider a trajectory $\mathbf{Z}'(t)$ that is locally analytic (i.e. smooth to arbitrary order) and the reduced equation

$$\frac{d\mathbf{Z}}{dt} = F(\mathbf{Z}'(t), t) \tag{22}$$

If $F(\mathbf{Z}, t)$ is a piecewise polynomial function in $\mathbf{Z}$, then as $\mathbf{Z}'(t)$ crosses element boundaries, F will be non-smooth in time. If derivatives up to the Nth are continuous (a C^N representation) then a simple time-stepping scheme will have a maximum local convergence rate of order $N + 2$ on timesteps where this boundary is crossed, and $N + 2$ globally, since most intervals will not cross a cell boundary.

This means, for example, that a standard fourth-order Runge-Kutta method, with 4th order global accuracy, will only achieve that order of convergence tracing particles in fields with at least C^2 smoothness. Although order of convergence is not a complete diagnostic of the usefulness of a scheme, this indicates that high-order timestepping may not be appropriate. Whether high-order timestepping accuracy is required is quite problem dependent.

Special purpose integrators may be built to break the time domain up into regions so that in each time sub-interval the particle does not cross any cell boundaries, but this clearly creates significant extra complexity.

Particle tracing in magnetised plasmas is a somewhat special-purpose problem, where the evolution equations may often be split into two portions $F_0 + F_1$, with F_0 representing (normally) the long-wavelength background magnetic field, and F_1 associated with fine-scale low-amplitude turbulent fluctuation. Because of the extreme anisotropy of transport, it is often important to very accurately solve for the particle motion along the background field F_0, in order that small errors in tracing unperturbed particle orbits do not overwhelm the small transport effect due to F_1.

Note that in a tokamak the particle drifts (in gyrokinetic theory) depend on derivatives of the magnetic field ($\nabla\mathbf{B}$); this means that even defining these drifts requires a high order of continuity of the magnetic field representation. This issue will presumably also appear in Eulerian gyrokinetics. Due to the long wavelength smoothness of the background fields, even if these derivatives formally lack continuity, the discontinuities may not be very large.

The standard solution in tokamak codes is to represent the background fields in a way that allows high-order derivatives to be taken, perhaps on a simple Cartesian grid, where high-order continuous representations are straightforward, and then evaluate them in a setup phase on the grid points on which the dynamics will be solved, which may be a semi-structured or unstructured mesh.

Note that, depending on the timestepping scheme chosen, the particle evolution may need field evaluation at intermediate time points.

9 Particles in Nektar++, existing implementation, and proof of principle.

Particles have been implemented as a *filter* within the Nektar++ framework, and exploited for the purposes of considering erosion wear on surfaces[9]. This framework (currently) allows for certain kinds of particle motion relevant to solving the Navier-Stokes equations, whose motion is subject to local fluid forces, but not for particles to impact on the fluid motion.

Several kinds of particle motion are possible, and are schematically either fluid particles, where the particles have the local fluid velocity $\mathbf{v}$ and satisfy the equation

$$\frac{d\mathbf{r}}{dt} = \mathbf{v}(\mathbf{r}, t) \tag{23}$$

or finite-sized particles, which follow equations of motion

$$\frac{d\mathbf{r}}{dt} = \mathbf{V}, m\frac{d\mathbf{V}}{dt} = F(\mathbf{V}, \mathbf{v}, t) \tag{24}$$

of particles subject to forces due to the fluid. The filter implements linear multistep methods to solve these equations; these are quite adaptable to plasma problems, but adaption to use other kinds of schemes would not be excessively complicated.

The existence of the particles filter in nektar++, as well as a Hasegawa-Wakatani solver, has allowed us to set up a basic coupled plasma-particles test case, as a preliminary to exploring the impact of e.g. curved simulation elements on the particle tracing problem. As a demonstration, we initialise particles on a uniform grid and allow them to be advected by the fluid motion (fig. 5).

10 Co-design recommendations

- The similarity of the Particle and Eulerian moment-based-schemes mean that a common codebase should be possible and this will avoid duplication of effort. Until that point, only very simple fluid solvers are required.
- Particle methods for gyrokinetic systems are generally easy to adaptable in terms of trajectory equations of the chosen gyrokinetic equations, so it is not crucial to fix these at an early stage of the project.
- The representation of background and perturbed electromagnetic field needs special care if particle orbits are to be accurately solved. Some

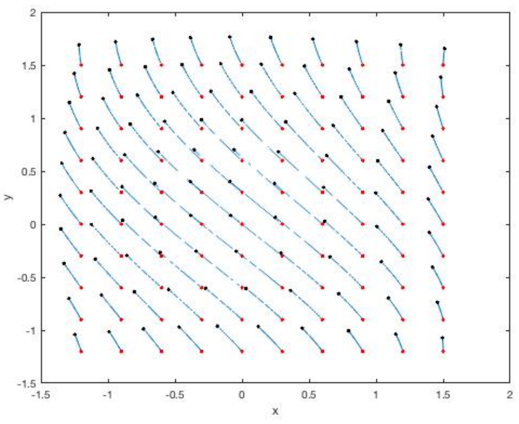

Figure 5: Particle trajectories on the (x, y) plane. Red markers show initial positions, black shows positions at $t = 4$.

precalculation may be necessary for derivatives of magnetic fields. Especially if perturbed magnetic fields may be large careful thought needs to be given to creating a smooth representation of fields or indirectly evaluating derivatives.

- Thought needs to be given to co-designing timestepping schemes for combined fluid-particle codes; particle methods can use e.g. linear multistep methods, or special purpose integrators (Boris algorithm) but the choice depends somewhat on the difficulty of evaluating fields at intermediate timepoints.

- A dedicated study would be needed to examine particle-tracking in piecewise curvilinear grids if the Neptune software is to use particle methods. There is infrastructure already existing to perform this work with Nektar++, both for conventional fluid problems or coupled to plasma-specific problems.

References

[1] NEPTUNE authors. M2.3.1 options for particle algorithms. Technical report, CCFE, 2020.

[2] Michael Barnes Felix I. Parra and Michael Hardman. 2047357-TN-01-2, 1d drift kinetic models with periodic boundary conditions. Technical report, Oxford University, 2021.

[3] S. Brunner, E. Valeo, and J. A. Krommes. Collisional delta-f scheme with evolving background for transport time scale simulations. *Physics of Plasmas*, 6(12):4504–4521, 1999.

[4] B. F. McMillan and L. Villard. Accuracy of momentum and gyrodensity transport in global gyrokinetic particle-in-cell simulations. *Physics of Plasmas*, 21(5):052501, 2014.

[5] A.Y. Aydemir. A unified Monte Carlo interpretation of particle simulations and applications to non-neutral plasmas. *Physics of Plasmas*, 1:822–831, 1994.

[6] Roman Hatzky, Trach Minh Tran, Axel Knies, Ralf Kleiber, and Simon J. Allfrey. Energy conservation in a nonlinear gyrokinetic particle-in-cell code for ion-temperature-gradient-driven modes in theta-pinch geometry. *Physics of Plasmas*, 9(3):898–912, 2002.

[7] S. Ku, C. S. Chang, R. Hager, R. M. Churchill, G. R. Tynan, I. Cziegler, M. Greenwald, J. Hughes, S. E. Parker, M. F. Adams, E. D'Azevedo, and P. Worley. A fast low-to-high confinement mode bifurcation dynamics in the boundary-plasma gyrokinetic code XGC1. *Physics of Plasmas*, 25(5):056107, 2018.

[8] E. Lanti, N. Ohana, N. Tronko, T. Hayward-Schneider, A. Bottino, B.F. McMillan, A. Mishchenko, A. Scheinberg, A. Biancalani, P. Angelino, S. Brunner, J. Dominski, P. Donnel, C. Gheller, R. Hatzky, A. Jocksch, S. Jolliet, Z.X. Lu, J.P. Martin Collar, I. Novikau, E. Sonnendrücker, T. Vernay, and L. Villard. ORB5: A global electromagnetic gyrokinetic code using the PIC approach in toroidal geometry. *Computer Physics Communications*, page 107072, 2019.

[9] Manuel F. Mejía, Douglas Serson, Rodrigo C. Moura, Bruno S. Carmo, Jorge Escobar-Vargas, and Andrés González-Mancera. Erosion wear evaluation using Nektar++. In Spencer J. Sherwin, David Moxey, Joaquim Peiró, Peter E. Vincent, and Christoph Schwab, editors, *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2018*, pages 419–428, Cham, 2020. Springer International Publishing.