nuclear fusion
IAEA

PAPER • OPEN ACCESS

Verification and validation of the high-performance Lorentz-orbit code for use in stellarators and tokamaks (LOCUST)

To cite this article: S.H. Ward *et al* 2021 *Nucl. Fusion* **61** 086029

OPEN ACCESS
IOP Publishing | International Atomic Energy Agency Nuclear Fusion
Nucl. Fusion **61** (2021) 086029 (14pp) https://doi.org/10.1088/1741-4326/ac108c

Verification and validation of the high-performance Lorentz-orbit code for use in stellarators and tokamaks (LOCUST)

S.H. Ward[1,2,3,*], R. Akers[2], A.S. Jacobsen[4], P. Ollus[5], S.D. Pinches[3], E. Tholerus[2], R.G.L. Vann[1] and M.A. Van Zeeland[6]

[1] York Plasma Institute, Department of Physics, University of York, York YO10 5DD, United Kingdom of Great Britain and Northern Ireland
[2] Culham Centre for Fusion Energy, Culham Science Centre, Abingdon, OX14 3DB, United Kingdom of Great Britain and Northern Ireland
[3] ITER Organization, Route de Vinon-sur-Verdon, CS 90 046, 13067 St. Paul Lez Durance Cedex, France
[4] Technical University of Denmark, Department of Physics, 2800 Kgs Lyngby, Denmark
[5] Department of Applied Physics, Aalto University, FI-00076 Aalto, Finland
[6] General Atomics, PO Box 85608, San Diego, CA 92186-5608, United States of America

E-mail: samuel.ward@york.ac.uk

Received 3 March 2021, revised 26 May 2021
Accepted for publication 1 July 2021
Published 19 July 2021

Abstract

A novel high-performance computing algorithm, developed in response to the next generation of computational challenges associated with burning plasma regimes in ITER-scale tokamak devices, has been tested and is described herein. The Lorentz-orbit code for use in stellarators and tokamaks (LOCUST) is designed for computationally scalable modelling of fast-ion dynamics, in the presence of detailed first wall geometries and fine 3D magnetic field structures. It achieves this through multiple levels of single instruction, multiple thread parallelism and by leveraging general-purpose graphics processing units. This enables LOCUST to rapidly track the full-orbit trajectories of kinetic Monte Carlo markers to deliver high-resolution fast-ion distribution functions and plasma-facing component power loads. LOCUST has been tested against the prominent NUBEAM and ASCOT fast-ion codes. All codes were compared for collisional plasmas in both high and low-aspect ratio toroidal geometries, with full-orbit and guiding-centre tracking. LOCUST produces statistically consistent results in line with acceptable theoretical and Monte Carlo uncertainties. Synthetic fast-ion D-α diagnostics produced by LOCUST are also compared to experiment using FIDASIM and show good agreement.

Keywords: fusion, tokamak, high performance computing, energetic particles, LOCUST, verification, validation

(Some figures may appear in colour only in the online journal)

* Author to whom any correspondence should be addressed.

1. Introduction

The size, power and performance of the ITER tokamak represents a paradigm shift in both experimental and computational fusion science. ITER will be one of the first tokamaks to generate a burning plasma, where a significant population of energetic particles must be sufficiently confined to simultaneously sustain the reaction and protect the plasma-facing components (PFC). Hence the transport of energetic particles in ITER remains a vital area of study [1] (see the activities of the ITPA Energetic Particle Physics Topical Group). Nevertheless, the spatiotemporal scales involved make detailed systematic analyses challenging with traditional computational methods. Resolving heat loads over the greater PFC surface areas of larger tokamaks requires evaluating additional gyro-resolved trajectories. Similarly, the extra compute time required to track ions over the increased ($\sim$1 s) slowing-down time is compounded by the need for smaller timesteps ($\sim$ns) and finer spatial resolution to minimise numerical drift. With similar large-scale devices on the horizon, such as STEP, this type of metaproblem has recently begun to attract more urgent attention [2–4].

There are multiple ways to advance computational capabilities, and adapting to novel or specialised hardware is one. This approach is advantageous for a number of reasons other than an immediate speed boost: the lower cost, energy consumption and space required can make specialised hardware more efficient for specific tasks; specialised hardware can be more accessible at the hardware level, for example interfacing with workstation devices directly via PCIe buses to avoid the need for remote data centres; minimal adaptation is required for modular or encapsulated code; and future hardware generations bring passive performance improvements more rapidly, depending on the type of hardware market [5]. This costs time tuning and adapting codes and algorithms. Additionally, sometimes hardware may be inaccessible if particularly new or prohibitively expensive for individual users. But high-performance computing platforms, including the cloud, now routinely offer heterogeneous hardware, often featuring combinations of central processing units (CPU), GPGPUs and even field-programmable gate arrays. Likewise, user-level accessibility is constantly improved by high-level APIs [6, 7]. For embarrassingly parallel tasks, which describe most linear energetic particle physics, GPGPUs in particular offer scalable hardware acceleration for little deployment cost; GPGPUs are energy-efficient, have a low capital cost per computational thread and can be hosted by basic desktop computers. Despite this, the ratio of computational plasma physics codes utilising GPGPUs does not match that of the codes found on top supercomputers.

LOCUST is an algorithm designed specifically to use off-the-shelf GPGPU hardware to reach a computational performance that enables routine simulations of fast-ions in ITER-scale devices. For an ITER burning plasma scenario (electron density $\sim 10^{20}$ m^{-3}, electron temperature $\sim$25 keV, twelve separate 3D field components at 1 cm precision (which add significant burden upon the code due to the need to reconstruct the 3D field at each timestep) and a wall mesh comprising 6×10^7 tetrahedra/3×10^6 surface triangles), LOCUST can track 250 000 markers over 1 s to thermalisation in 1 ns time steps in 15 h on a node with eight Nvidia P100 GPGPUs controlled by one Intel Xeon E5-2689.

As part of the software life cycle, it is vital to continually verify that new tools like LOCUST are correctly implemented and to validate the accuracy of any underlying models by comparing with experiment. To this end, LOCUST has been benchmarked over a range of test scenarios, with the aim that its performance matches multiple well-established tools. Emphasis has subsequently been placed on accurately verifying the fundamental implementation of the code, while reducing possible interference from high-order physics models, before validating this against experiment.

In this paper, LOCUST is both described and rigorously tested. Section 2 describes the physical model, including its assumptions, their computational implications and the resulting information calculated by LOCUST. A description of the algorithm and its execution, along with the required inputs and outputs, is given in section 3. Results from cross-code benchmarks are presented in section 4. Finally a summary of findings is made in section 5.

2. Model overview

The primary goal of LOCUST is to calculate the steady-state distribution function of fast-ion species as efficiently as possible, while also resolving individual ion trajectories in a realistic 3D geometry to calculate PFC power loads. To complement massively parallel memory-bound hardware, we opt for a mathematically simple but computationally intensive approach: solving the Lorentz equation of motion (1) for individual kinetic markers i representing real-space position $\mathbf{r}_i(t)$ and velocity $\mathbf{v}_i(t)$ Monte Carlo samples of the fast-ion distribution function:

$$\frac{\mathrm{d}^2\mathbf{r}_i}{\mathrm{d}t^2} = \frac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t} = \frac{q_i}{m_i}(\mathbf{v}_i \times \mathbf{B}(\mathbf{r}_i) + \mathbf{E}(\mathbf{r}_i)) \tag{1}$$

with $\mathbf{B}(\mathbf{r}_i)$ and $\mathbf{E}(\mathbf{r}_i)$ the magnetic and electric fields evaluated at the position $\mathbf{r}_i$ of the ith marker. In LOCUST, these trajectories are evaluated using fixed-step numerical schemes to minimise thread divergence. These schemes track either the particle's position in real space throughout a full gyro-orbit (FO) or track its guiding centre (GC) and gyro-phase by solving the modified equations of motion in [8] (see appendix A). Multiple such numerical integration schemes are included: Strang-splitting [9]; BGSDC [10]; Runge–Kutta-type integration methods such as McClements–Thyagaraja–Hamilton, Fehlberg [11], Cash–Karp [12], Dormand–Prince [13], and Goeken–Johnson [14]; and Euler methods such as the popular Leapfrog/Boris [15].

LOCUST relies on approximations to enhance the computational tractability of the model. Firstly, the electromagnetic field experienced by each fast particle is evaluated without the relatively weak contributions from other fast-ion species. If we further assume that the background plasma is in static

equilibrium and self-consistent with the fast-ion distribution, we can simplify the computation in three ways:

(a) We ignore the influence of fast ions on the thermal plasma. If and only if we chose to do so then it becomes valid to track independent sub-samples of the fast-ion distribution (see item (b)). Without this approximation, these sub-samples of the fast-ion population would be self-coupled via their exchanges with the bulk plasma, but they can be treated independently in the case where the background equilibrium is assumed to be static and self-consistent with the final steady-state fast-ion distribution. In practice, equilibria are taken from pre-converged time-dependent transport simulations, which use simplified fast-ion models. In a given LOCUST simulation, the equilibrium is held constant by fixing the background plasma temperature, density and rotation profiles as well as the magnetic field.

(b) We utilise the entire trajectory history of each marker when calculating the distribution function such that a marker ensemble of constant size need only be tracked from source to sink once to estimate the steady-state distribution function. By extending the logic from item (a), one can treat individual points along a marker trajectory as independent subsamples of a common distribution function. This effectively parallelises the calculation across time similar to other fast-ion codes such as ASCOT [16].

(c) Each fast ion is tracked independently in parallel by non-blocking processes. This is again enabled by item (a).

Without interactions between fast-ion species, the equilibrium $\mathbf{E}(\mathbf{r}_i)$ and $\mathbf{B}(\mathbf{r}_i)$ terms are dominated by background sources, such as the fields due to external coils and the resulting plasma response, and short-range Coulomb interactions with the thermal species. While the background equilibrium is prescribed numerically, the short-range interactions with bulk species are replicated using a stochastic perturbation to the marker velocity vector term in equation (1). This is implemented with a Monte Carlo Fokker–Planck collision operator that includes terms for diffusion and drift of pitch angle (λ) and energy (ϵ). For the computational convenience of evaluating uniformly distributed random numbers, using the Nvdia CURAND XOR shift, these collision operators are based upon the binomial operators derived in [17]:

$$\Delta\lambda_i = -\lambda_i \nu_\mathrm{d} \Delta t \pm \left[\left(1-\lambda_i^2\right)\nu_\mathrm{d}\Delta t\right]^{1/2} \tag{2}$$

$$\Delta\epsilon_i = -2\nu_\epsilon \Delta t \left[\epsilon_i - \left(\frac{3}{2} + \frac{\epsilon_i}{\nu_\epsilon}\frac{\mathrm{d}\nu_\epsilon}{\mathrm{d}\epsilon}\right) k_\mathrm{B} T_\mathrm{th}\right] \pm 2(k_\mathrm{B} T_\mathrm{th} \epsilon_i \nu_\epsilon \Delta t)^{1/2} \tag{3}$$

,where $\pm$ is a random sign with equal probability and the ν_ϵ and ν_d terms denote collision frequencies as functions of the Coulomb logarithm $\ln(\Lambda)$ and error function [18–20]. For a given fast-ion population, these equations are solved for each bulk species. An option to accelerate the effects of these equations is also included—similar to the goosing algorithm in NUBEAM [21].

By solving these equations, LOCUST calculates and outputs a range of physics results in many formats. 3D power loads are derived from the intersections of orbits and PFCs. Similarly, Hamiltonian field line trajectories can be efficiently evaluated to create various types of Poincaré maps, each designed to illustrate particular magnetic field structures. The distribution function is generated by binning markers typically every 100 ns, and it can be generated in $[R, Z, v, \lambda]$ and constants-of-motion $[\epsilon, P_\phi, \sigma, \mu]$ spaces. Here P_ϕ is the fast-ion canonical angular momentum, σ the sign of the first-order guiding-centre pitch and μ the instantaneous magnetic moment, which is expanded to first order [22] to improve accuracy when binning markers in devices with steep field gradients. LOCUST also calculates one-dimensional poloidal flux profiles of fast-ion-driven current, torque ($\mathbf{J}\times\mathbf{B}$ and collisional), pressure and heating to bulk species channels.

3. Code design and execution

The overall data flow to and from LOCUST is illustrated in figure 1, along with preprocessing stages and related external physics solvers. The background equilibrium fields describing the 2D axisymmetric and 3D perturbative components are passed to LOCUST as separate numerical representations. These can be in the form of IMAS interface data structures (IDS), GEQDSKs, 3D rectilinear grids, or Fourier-decomposed data. The bulk species temperature and density are supplied to the collision operator numerically as interpolated functions of poloidal magnetic flux. The initial fast-ion ensemble is read from individual marker phase-space positions, often generated by external plasma heating codes. Finally, the 3D PFC power flux can be calculated from an axisymmetric limiter outline or an unstructured volumetric tetrahedral mesh, avoiding the need for runtime octrees. 3D meshes are generated by defeaturing and repairing elements of computer-aided design (CAD) engineering models, typically using CADfix or SpaceClaim, before re-meshing volumetrically in Attila [23]. Equally, this geometry can be represented by the IMAS generic grid description in the wall IDS.

Once runtime settings and input data are specified, execution proceeds through stages 1–7 in figure 1.

First, the X-point, magnetic axis and last-closed flux surface (LCFS) are precisely located. Since LOCUST is not storage bound, the rectangular 2D field is then effectively cached by storing pre-computed bicubic spline coefficients for each knot—either just the required derivatives and cross-derivatives or the entire set. While this method is also offered in LOCUST for storing 3D fields as tricubic splines on a rectilinear grid, a Fourier-decomposed format is also available. This latter option is preferred if on-board memory begins to limit grid resolution or when resolution is only needed in particular dimensions; the freed space may be used to redistribute resolution, reducing magnetic field divergence enough to enable linear interpolation—which is faster. The 3D mesh is then cached by labelling vacuum-facing triangles as PFC surfaces. For rapid and synchronous tracking of tetrahedra

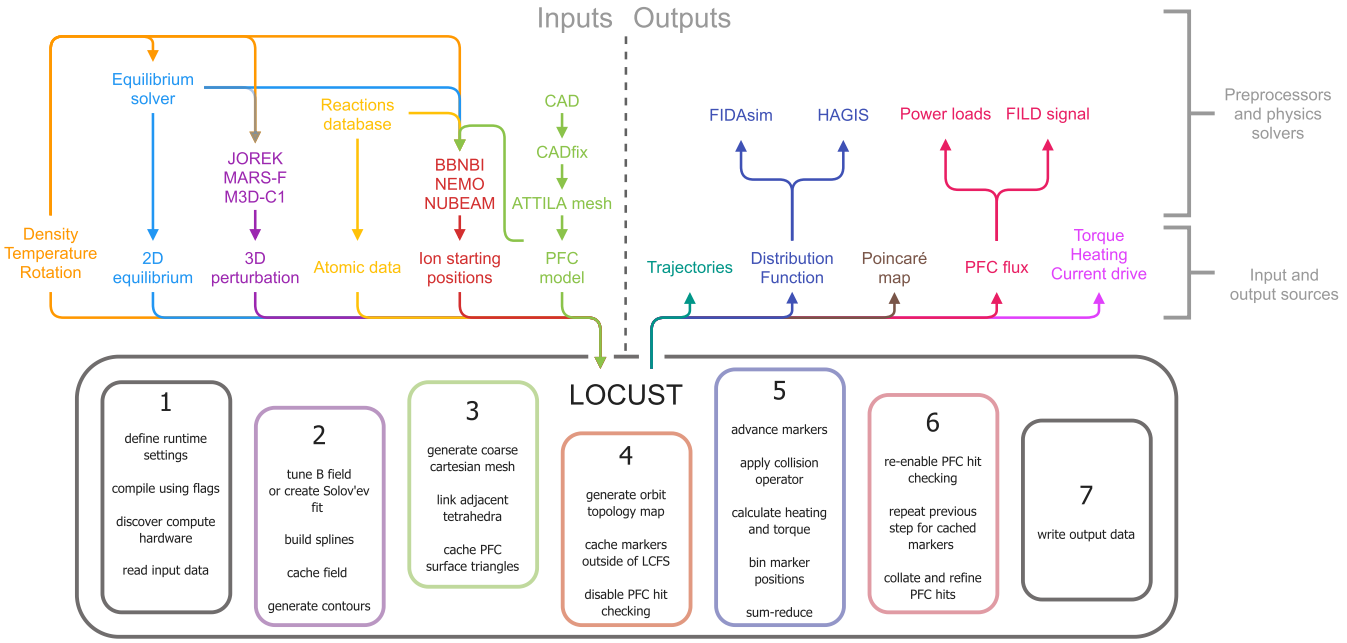

Figure 1. Information flow (arrows) to and from LOCUST, itself shown in the solid grey box with main preprocessing, runtime and postprocessing stages. These stages execute in ascending order one to seven and are colour-coded to match relevant input/output data. Included are examples of external physics codes which have been used to pre-process or post-process data [21, 23–32].

traversal, nodes comprising the unstructured wall mesh are also mapped to a coarse Cartesian grid and adjacent tetrahedra are linked [33].

After runtime preprocessing, marker tracking is performed in two stages: markers which start and remain inside the LCFS are first tracked without PFC interception checks before tracking all remaining markers with PFC checks. Considering the constraints outlined in section 2, optimal performance is achieved if the number of occupied threads is maximised, indicating GPGPUs are currently the most suitable hardware—with each on-board streaming multiprocessor running thousands of threads, each able to track an individual marker in parallel. The Nvidia CUDA application-programming interface (API) is used to interface with this hardware, and the core tracking algorithm is written in PGI FORTRAN. CUDA also offers libraries for efficient random number generation, including CURAND and a Mersenne Twister [34] implementation which are both optional in LOCUST.

Implementing SIMT algorithms of this type requires some hardware-specific design considerations. The marker tracking algorithm, which comprises the stages in box 5 of figure 1, is illustrated in figure 2. The required data is first transferred from the host CPU to the GPGPU device before a non-blocking device kernel is triggered to advance markers in phase space. Simultaneously, fast-ion positions from the previous timestep are sum-reduced on the host. Care is taken to overlap these processes; upon their completion, all processes are synchronised by a transfer of the next fast-ion positions to the host. These positions take the integer form of the corresponding distribution function bin indices to lower data throughput. This process is scaled across multiple devices within a node using OpenMP.

As marker trajectories are evaluated, the latest positions are cumulatively binned to build up the steady-state distribution function. This process, illustrated in figure 3, occurs until markers either strike a PFC, reach a tracking time limit or slow to a prescribed velocity cut-off—typically $3T_{\mathrm{bulk}}/2$. Upon completion, incident PFC power is collated and adaptively refined across the surface mesh.

4. Testing

4.1. Orbit tracking

The most fundamental test is to examine collisionless, full-orbit trajectories followed in LOCUST. Here, these trajectories are calculated using the Boris integrator in the presence of a 2D and 3D wall model of DIII-D, and they are compared to equivalents calculated by MPI_ORBF (deployed in [35]), which uses a slightly different 3D wall. Figure 4 shows that, for markers initially distributed uniformly in $[R, \lambda]$ space along the outer horizontal mid-plane and at an injection energy of 80 keV, each simulation measures the same prompt loss boundary to within ($\mathrm{d}\lambda = 0.04$, $\mathrm{d}R = 1.0$ cm). Variations of this magnitude amount to absolute changes of ∼0.1–0.4 percentage points in total loss fraction: 6.8% (MPI_ORBF), 6.9% (LOCUST with 2D wall) and 6.5% (LOCUST with 3D wall). Therefore, the differences between the trajectories calculated by LOCUST and MPI_ORBF are within the variations caused by the wall model.

4.2. Collisional transport

To test collisional transport in the presence of a toroidally symmetric background plasma, comparisons were made against

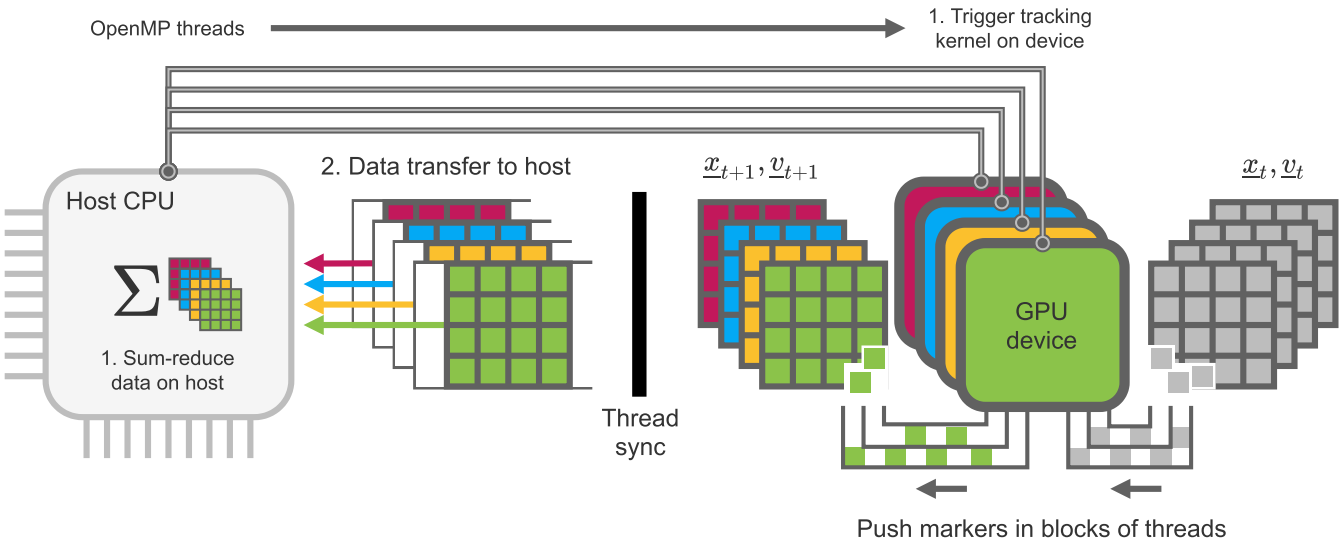

Figure 2. Execution model for the kernel-level marker tracking algorithm. Steps 1 and 2 execute in serial, but operations therein are performed asynchronously. In step 1 the GPGPU tracking kernel is triggered while the previous marker positions are sum-reduced on the CPU to form the distribution function. These processes are synchronised by a memory transfer of the new marker positions from the device to host in step 2.

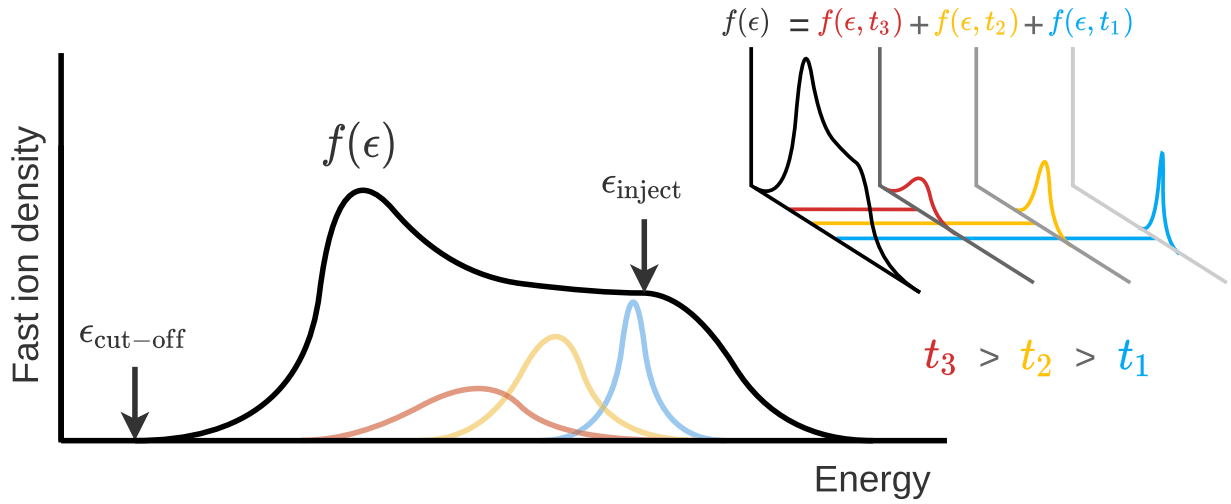

Figure 3. Numerical algorithm for generating the steady-state distribution function f, shown here in black as a single function of marker energy $f(\epsilon)$. This is formed by cumulatively summing $f(\epsilon, t_i)$, which are calculated independently at discrete time intervals t_i.

ASCOT4 [16] (version 5 has since been shown to produce similar results [36]) and the NUBEAM [21] module of TRANSP [37] via OMFIT [38]. ASCOT and NUBEAM each employ different definitions of the Coulomb logarithm, as described in appendix C, and truncate their collision operators to varying degrees.

Data from a single time slice at 3 s was taken from DIII-D shot #157418 [39]. This discharge featured an ITER-similar shape designed for ELM suppression studies. Relevant 0D parameters for this discharge are given in table 1 below. The full-energy component of the deposition of a counter-current-injected 80 keV deuterium neutral beam into a static plasma was calculated by NUBEAM and used in all subsequent simulations among all codes. Co-injection was also explored, but it was concluded that the larger volume of phase-space explored by counter-injected markers is vital for creating a test rigorous enough to expose discrepancies between the codes. Because all codes were forced to use an identical starting marker list from NUBEAM, which only provides the weight, real-space position, pitch and energy of markers, the study was limited to guiding-centre tracking. Consequently, when calculating collisional effects and wall interceptions, there is an unavoidable systematic spatial error $\sim r_{\text{Larmor}}$ in the marker position introduced by differences in the finite Larmor radius (FLR) model implemented by each code:

- NUBEAM assigns markers a random gyrophase, assuming a circular orbit.
- LOCUST tracks gyrophase from birth.
- ASCOT ignores FLR corrections unless near the PFC wall, where a random gyrophase is chosen.

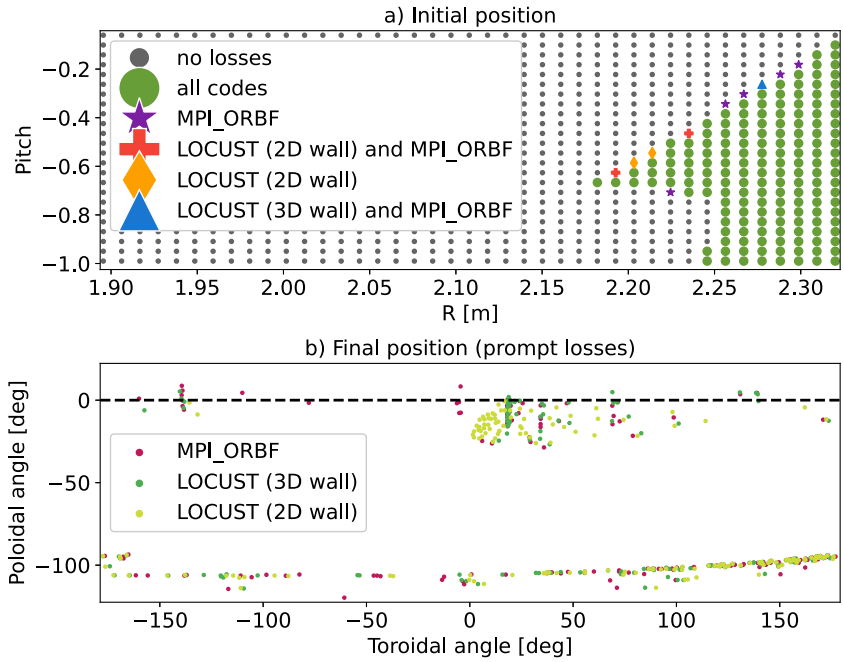

Figure 4. (*a*) Phase-space locations for markers initially distributed on a uniform pitch-radius grid along the outer horizontal mid-plane. Pitch is measured relative to the plasma current. Marker type denotes which simulations measure losses. Discrepancies are localised to the loss boundary within $\mathrm{d}\lambda\,\mathrm{d}R$. (*b*) Final real-space positions of all promptly lost markers. All three runs used a slightly different wall model but predict most losses at the outboard mid-plane ($\theta_{\mathrm{pol}} = 0°$) and divertor ($\theta_{\mathrm{pol}} = -100°$). The structure of the mid-plane port box cut-outs can be observed in the loss patterns produced by 3D simulations.

Table 1. DIII-D and MAST discharge data.

Quantity	DIII-D	MAST
I_p (MA)	1.6	0.8
R/a	$\sim$2.5	$\sim$1.3
$\lvert B_\phi \rvert$ (T)	1.9	0.4
Core T_i (keV)	9.4	1.5
Core T_e (keV)	4.1	1.1
Core n_e (10^{19} m^{-3})	5.9	3.7

GCs were followed for 100 milliseconds and collisions disregarded for markers outside the LCFS, as is required by NUBEAM. Furthermore, impurities, neutral species, bulk rotation, electric fields and neutral particle interactions were removed. To effectively remove any beam–beam interaction, the neutral beam density was artificially lowered by reducing the injected NBI power to 1 W in OMFIT, before scaling the deposited marker ensemble power to 1 W across all simulations. All codes employed a lower energy cut-off at $3T_{\mathrm{bulk}}/2$. It is worth noting that, to achieve the required fidelity without resorting to Monte Carlo smoothing, which may mask possible discrepancies, NUBEAM simulations for DIII-D were conducted with the untypical settings shown in table B1 in appendix B. For the following studies, LOCUST used the Cash–Karp integrator scheme to track marker guiding-centres and the Strang-splitting scheme to track full-orbit positions.

All codes produce similar results when fast-ion dynamics are isolated to within the plasma. To achieve this, an artificial axisymmetric PFC surface was created concentric to the LCFS but 5% further from the magnetic axis ($R_{\mathrm{fac}} = 1.05$)—the closest permitted by NUBEAM. Figures 5 and 6 show the $[R,Z]$ and $[\epsilon,\lambda]$ projections respectively of the calculated distribution functions collected at the marker GCs. Subfigures (a) show the same density contours as produced by each code while subfigures (b) and (c) both show the absolute element-wise differences, $\delta f(R,Z)$ and $\delta f'(R,Z)$, between the ASCOT and LOCUST distribution functions. A similar element-wise comparison against NUBEAM could not be performed reliably, since NUBEAM collates the fast-ion density onto a flux-aligned grid. In subfigures (a) and (b), LOCUST aims to match the ASCOT collision operator, and in figures (c) LOCUST aims to match NUBEAM. This was achieved by using a different Coulomb logarithm and by varying the degree to which its collision operator is truncated. In the former case, the collision operator was fully expanded. In the latter, collisions against plasma species p were truncated to $\sim\mathcal{O}([v_i/v_i]^5)$ while thermal ion accumulation was disabled. This was done without assuming prior knowledge of the ASCOT and NUBEAM collision operators.

In most regions, δf $\sim$3% between both codes, which is within the fundamental uncertainties in the theoretical formulation of the Coulomb logarithm. Switching to an NUBEAM-like collision operator, as in $\delta f'$, creates a lower density in the core and outboard edge but higher density towards the X-point. Within the LCFS, this leads to the average δf increasing from 3.6% to 4.7%, a change still in line with theoretical variations. Nevertheless, when the collision operator is matched, δf shows only regions near the X-point and wall retain any distinguishable difference—caused by the influence of FLR model on wall interceptions. The remaining noise in

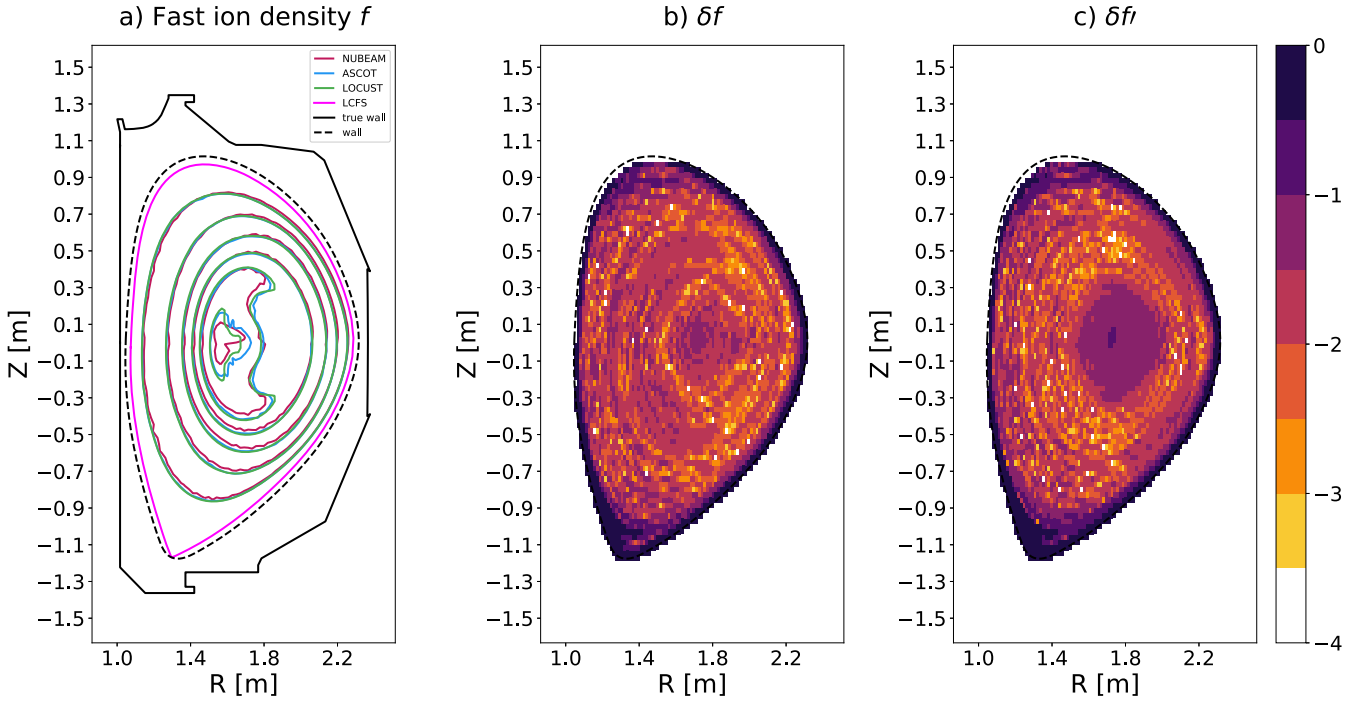

Figure 5. (*a*) Contours of fast-ion density integrated over velocity space. Contours are at equal levels across codes. The real and artificial limiter profiles are shown in black solid and dashed respectively. NUBEAM bins according to a flux-aligned spiralised grid whereas ASCOT and LOCUST use rectilinear, hence some variation near the magnetic axis is expected due to numerics. (*b*) Absolute element-wise difference between LOCUST and ASCOT distribution functions $\delta f \equiv \log_{10}(|f_{\mathrm{LOCUST}} - f_{\mathrm{ASCOT}}|/\max(f_{\mathrm{LOCUST}}, f_{\mathrm{ASCOT}}))$ where max() is the local, element-wise maximum. A high-order collision operator was used in LOCUST, as well as the ASCOT definition of $\ln(\Lambda)$. (*c*) The same as plot (*b*) except with LOCUST using a truncated collision operator and the NUBEAM definition of $\ln(\Lambda)$. This choice primarily affects the core region, though some differences on the outboard side are noted. The loss region near the X-point remains.

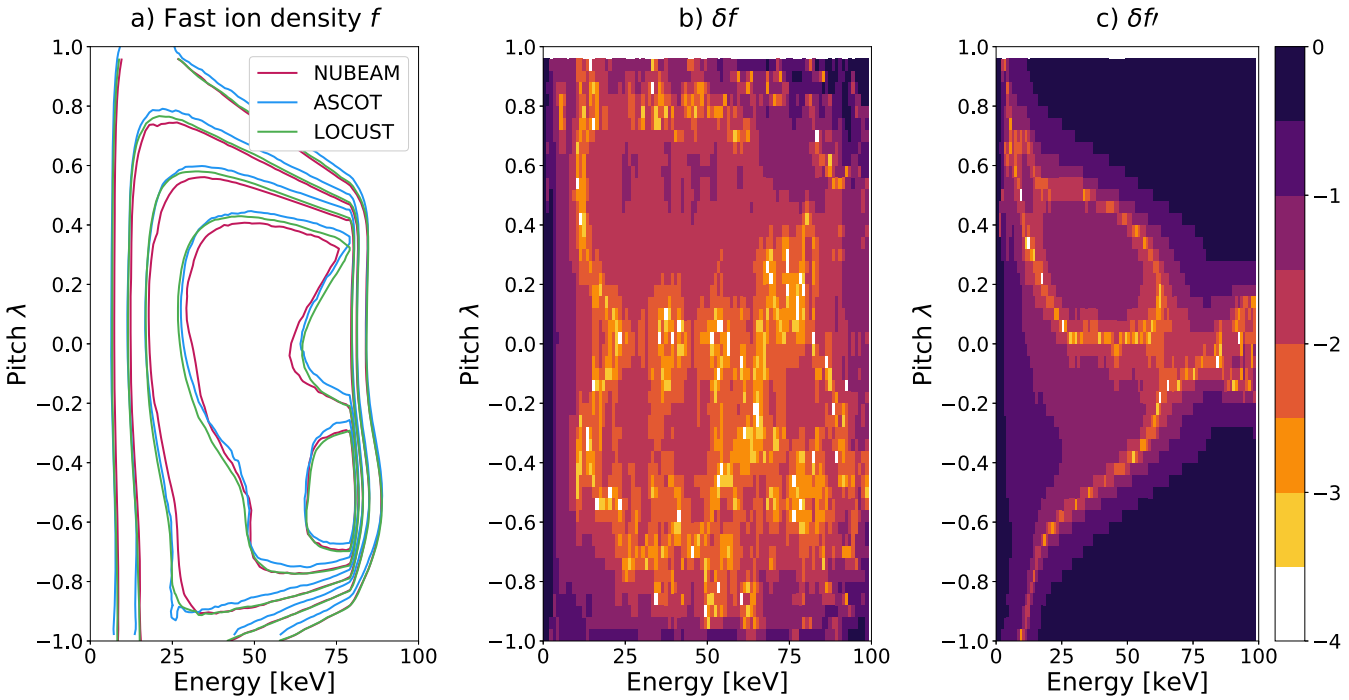

Figure 6. (*a*) Contours in pitch-energy space of fast-ion density integrated over real space, where λ is defined with respect to the direction of toroidal current flow as is convention in NUBEAM. Contours are at equal levels across codes. (*b*) The absolute element-wise difference between the LOCUST and ASCOT distribution functions $\delta f \equiv \log_{10}(|f_{\mathrm{LOCUST}} - f_{\mathrm{ASCOT}}|/\max(f_{\mathrm{LOCUST}}, f_{\mathrm{ASCOT}}))$ where max() is the local, element-wise maximum. A high-order collision operator was used in LOCUST, as well as the ASCOT definition of $\ln(\Lambda)$. (*c*) The same as plot (*b*) except with LOCUST using a truncated collision operator and the NUBEAM definition of $\ln(\Lambda)$. The injection energy is 80 keV, so diffusive noise can be expected above this energy level.

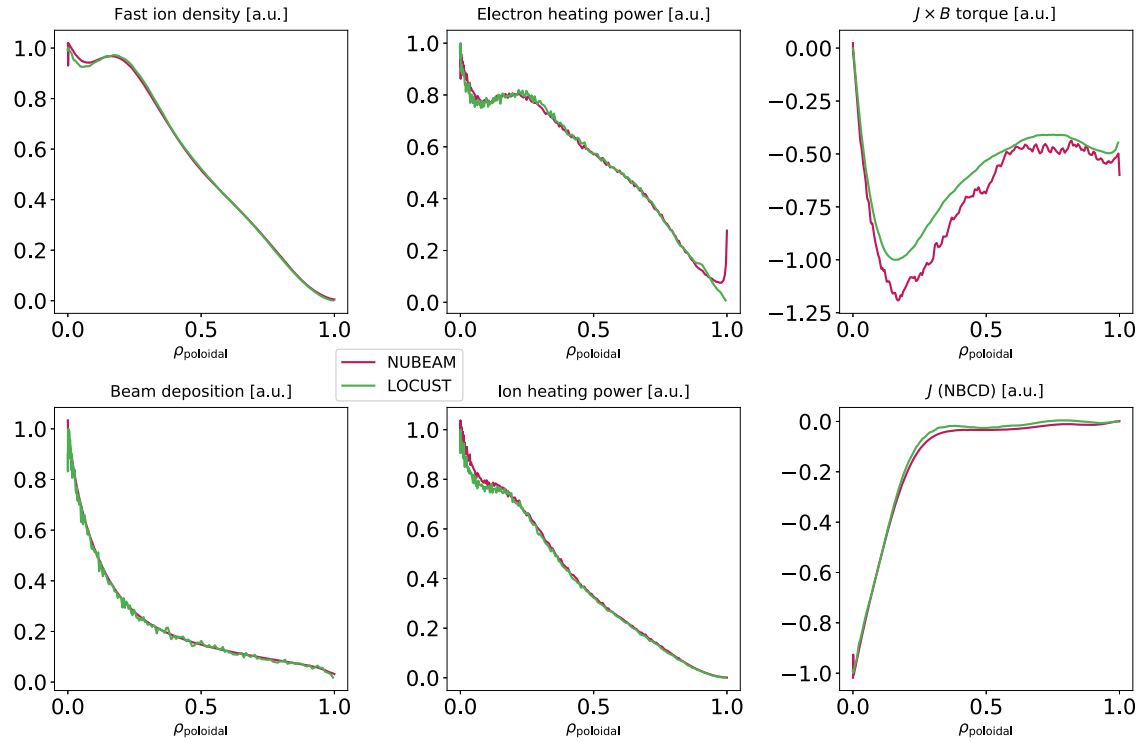

Figure 7. Normalised quantities measured against normalised poloidal flux ρ_{poloidal}, including the neutral beam current drive. Some discrepancy around $\rho = 0$ is expected due to binning width. Most important is the discrepancy in the $\mathbf{J} \times \mathbf{B}$ torque, which suggests a difference in the measured orbit width.

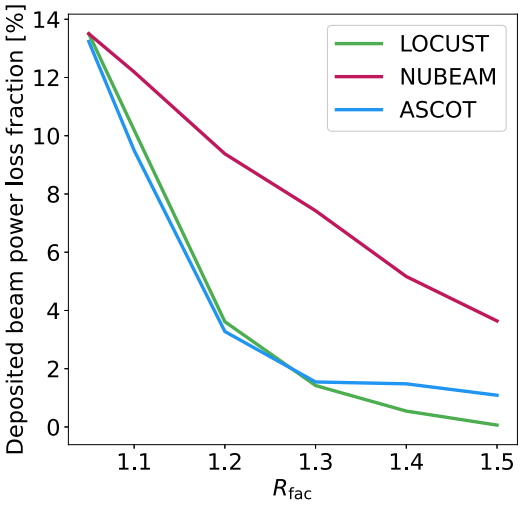

Figure 8. Fraction of deposited power lost as a function of limiter wall radius where $R_{\text{fac}} \equiv r_{\text{limiter}}/r_{\text{LCFS}}$ with r representing minor radius.

the core plasma is likely caused by differences in any tuning applied to the equilibrium field by each code, which may perturb the position of the flux surfaces and cause a flux-aligned noise pattern. Furthermore, the slight mismatch in the core is possibly due to the adaptive time step in ASCOT—as described later.

The calculated flux profiles in figure 7 also show good agreement but highlight the importance of the plasma boundary. The uptick in edge electron heating is caused by the FLR corrections in NUBEAM spreading deposited power over a gyro-orbit width; the orbits of markers with GCs located just outside the LCFS concentrate their deposited power into a thin shell where the orbit overlaps the plasma. In LOCUST, this power is instead collected at the GC—outside the plasma for these markers—and is thus ignored. This effect is artificial, and simulations can avoid this by imposing ion sinks outside the plasma boundary, such as a neutral density for charge-exchange or an extrapolated plasma density.

The measured $\mathbf{J} \times \mathbf{B}$ torque suggests some discrepancy in orbit topology, especially at the edge [40], but this information cannot be directly extracted from NUBEAM. To explore this further, additional simulations were performed with similar artificial limiters up to $R_{\text{fac}} = 1.5$. This allows orbits to populate the vacuum region between the plasma and first wall. The XBMBND setting in TRANSP, which, in all previous DIII-D simulations, registered any fast ion located at $\sqrt{\psi_{\text{toroidal}}} >$ XBMBND as hitting a PFC, was permanently increased to avoid artificial termination of markers. Figure 8 shows the measured steady-state PFC power flux as the limiter distance is increased. As first-wall losses are sensitive to the wall model, some disagreement between the codes is expected, especially at high R_{fac} when orbits between the plasma and outboard PFCs may have a significant r_{Larmor}. Hence the agreement between ASCOT and LOCUST is satisfactory, as it is mostly within the variations expected from differences in wall model. However, it is unclear why the NUBEAM power fluxes diverge so quickly, though it is encouraging that the resulting

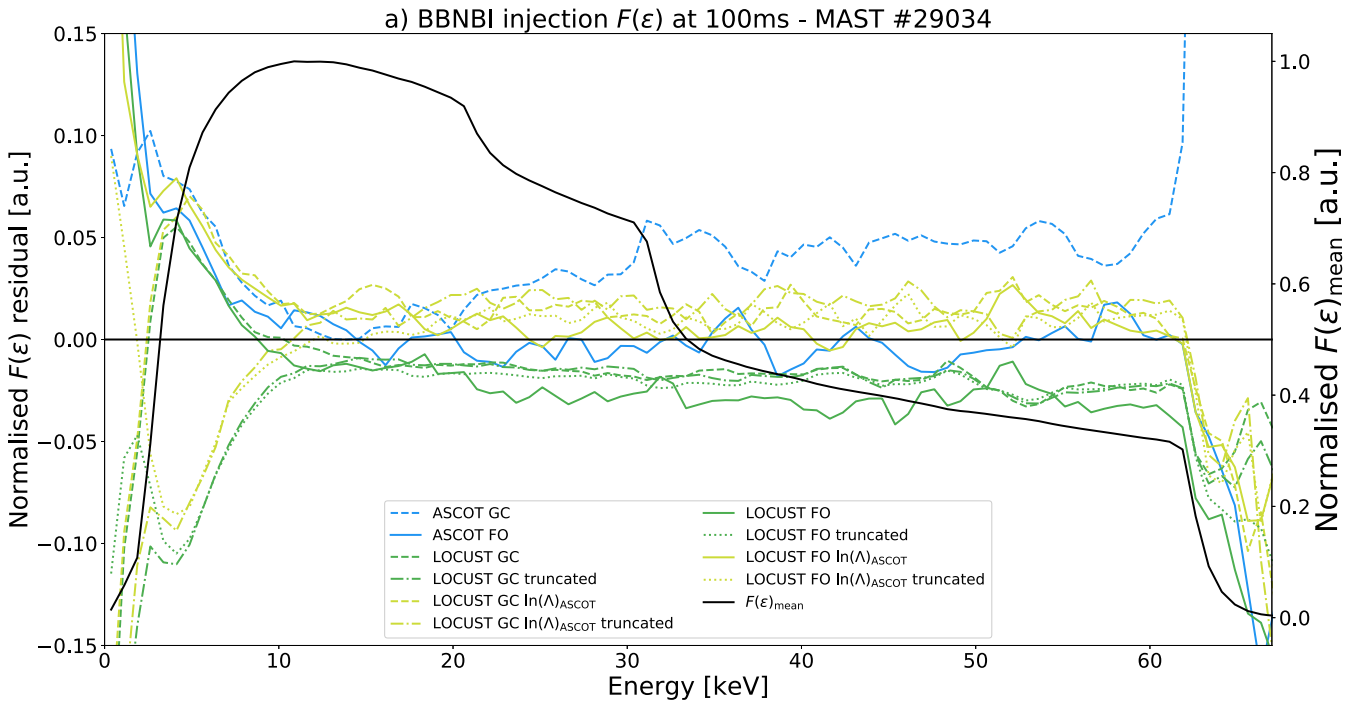

Figure 9. The fast-ion density $f(\epsilon)$ after 100 milliseconds, integrated over all dimensions except energy, ϵ, is averaged across all simulations, normalised and shown in black as $f(\epsilon)_{\text{mean}}$. The residual differences $(f(\epsilon) - f_{\text{mean}})/f_{\text{mean}}$ for each simulation are shown on the density residual axis. Simulations using GC and FO tracking (solid and dashed respectively); truncated and high-order collision operators (truncated and non-truncated labels respectively); and LOCUST $\ln(\Lambda)$ and ASCOT $\ln(\Lambda)$ (dark green and light green respectively), are all shown, with combinations of these linestyles representing the corresponding combinations of options. The near-symmetrical splitting of predictions at lower energies is caused by the collision operator truncation while systematic differences can be mainly attributed to the $\ln(\Lambda)$ used.

discrepancies in the distribution function remain solely at the plasma edge.

The previous methodology was repeated for a spherical tokamak topology. Such devices tend to have steeper gradients [41], meaning any inaccuracies in fast-ion models will be exacerbated; in MAST for example, r_{Larmor} can approach the order of the minor radius, and the impact on the validity of the guiding-centre approximation has been questioned [42, 43]. MAST shot #29034 was selected to allow for comparison with measured fast-ion D-α (FIDA) emission [44]. Relevant 0D parameters for this discharge are given in table 1 above. The NBI deposition code BBNBI was included to enable comparison of full-orbit simulations. Like before, a single time slice of data describing the background plasma was extracted at 360 ms. A quiescent period was chosen during the flat-top phase when core electron temperature and density were relatively constant. To create a realistic deposition, time-resolved NBI data from OMFIT were used to generate the NUBEAM deposition for the south-south neutral beam, while settings as similar as possible were chosen for BBNBI: a 62 keV co-current beam with 62% full, 27% half and 11% third energy fractions. Hence it is technically inappropriate, and out of the scope here, to cross-compare the results attached to each beam code in this case. GC and FO trajectories were then calculated by each code over 100 ms—enough to reach steady state.

The co-current NBI confines the fast ions to the plasma core, where discrepancies are hard to distinguish and there is a systematic shift in spatial density due to the FLR displacement, so instead we examine $f(\epsilon)$, the distribution function integrated over all dimensions except energy ϵ, which still encodes some real-space information through the effects of the steep temperature and density gradients on the fast-ion diffusion rate. The only unexpected discrepancy in real space is caused by noise in the NUBEAM distribution function, since this case decreased NPTCLS to 10^5.

The average $f(\epsilon)$ across all simulations is shown normalised in black in figures 9 and 10. The residuals—the difference between each simulation and the un-normalised average—are also shown normalised against the average. Most simulations are within $\pm 3\%$ of the mean, except at lower energies due to the collision operator truncation.

As a figure of merit, for simulations using the deposition calculated by BBNBI in figure 9, the maximum difference in total fast ions between any two simulations is 5%. This falls to $\leqslant 2\%$ for pairs of simulations which follow similar assumptions, even including the ASCOT GC simulation, which differs from the LOCUST equivalent by 1.7% (the FO equivalents differ by 0.9%). The reason for the increased density at high energies measured by ASCOT GC is possibly due to the characteristics of the adaptive time step near the magnetic axis. NUBEAM, which also uses orbit acceleration, shows a similar feature in figure 10. For comparison, in comparing both FO and GC simulations of a homogeneous plasma by both LOCUST and ASCOT, the only discrepancy is a higher fast-ion density of $\sim 20\%$ in the high-energy diffusive tail measured

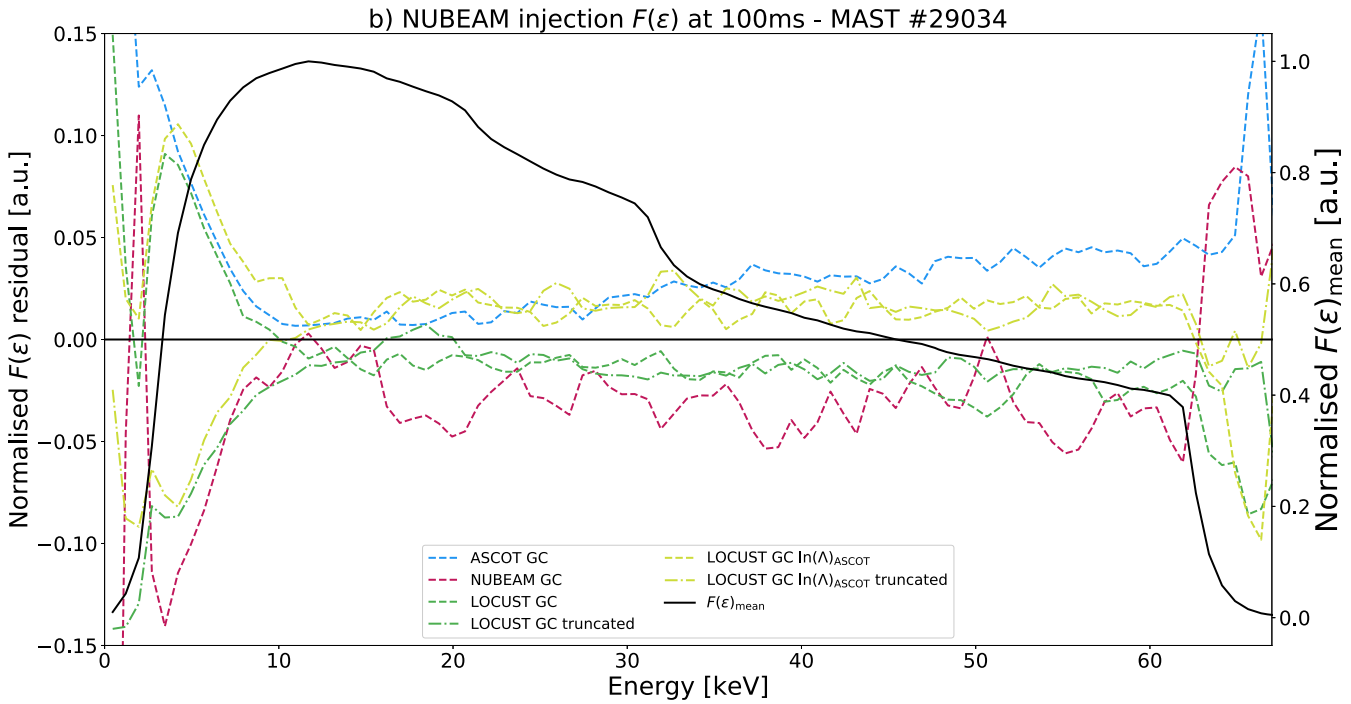

Figure 10. Equivalent to figure 9 but using deposition from NUBEAM, meaning only GC simulations can be performed. The overall trend is similar to figure 9 but the ASCOT tail is reduced.

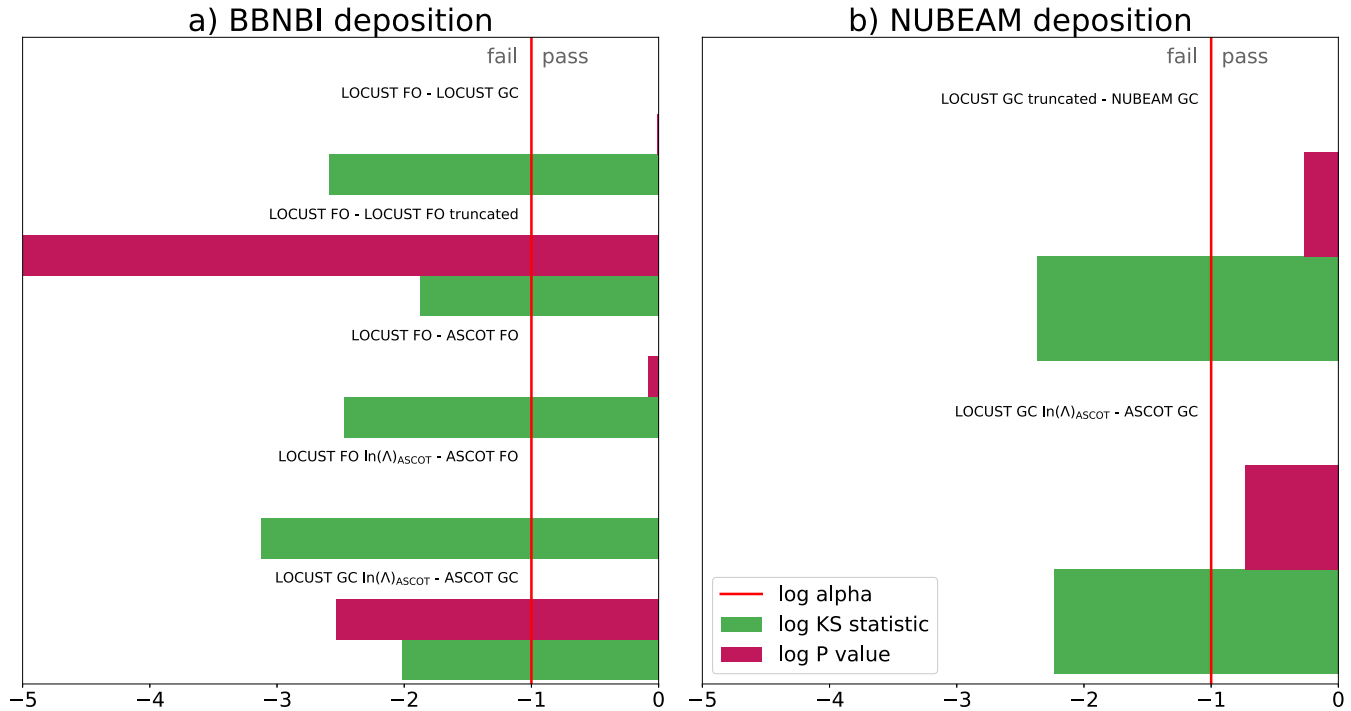

Figure 11. Kolmogorov–Smirnov test statistics (in green) for combinations of distribution functions shown in figures 9 and 10. The associated $P = P(D_{\text{measured}} < D)$ values are shown in red, with the arbitrary test pass-fail boundary α shown vertically in bright red. To pass the test, P must be lower than α, as demonstrated by comparing GC and FO LOCUST simulations.

by ASCOT GC. While this effect is much more pronounced in MAST than DIII-D, the extra $\sim$20% in the homogeneous case still only leads to a $\sim$0.2% difference in total fast ions—small enough to be affected by slight variations in beam deposition, as shown in figure 10 where the effect is lessened when the beam deposition is varied.

To compare these predictions more quantitatively, calculated Kolmogorov–Smirnov (KS) statistics D [45], and their

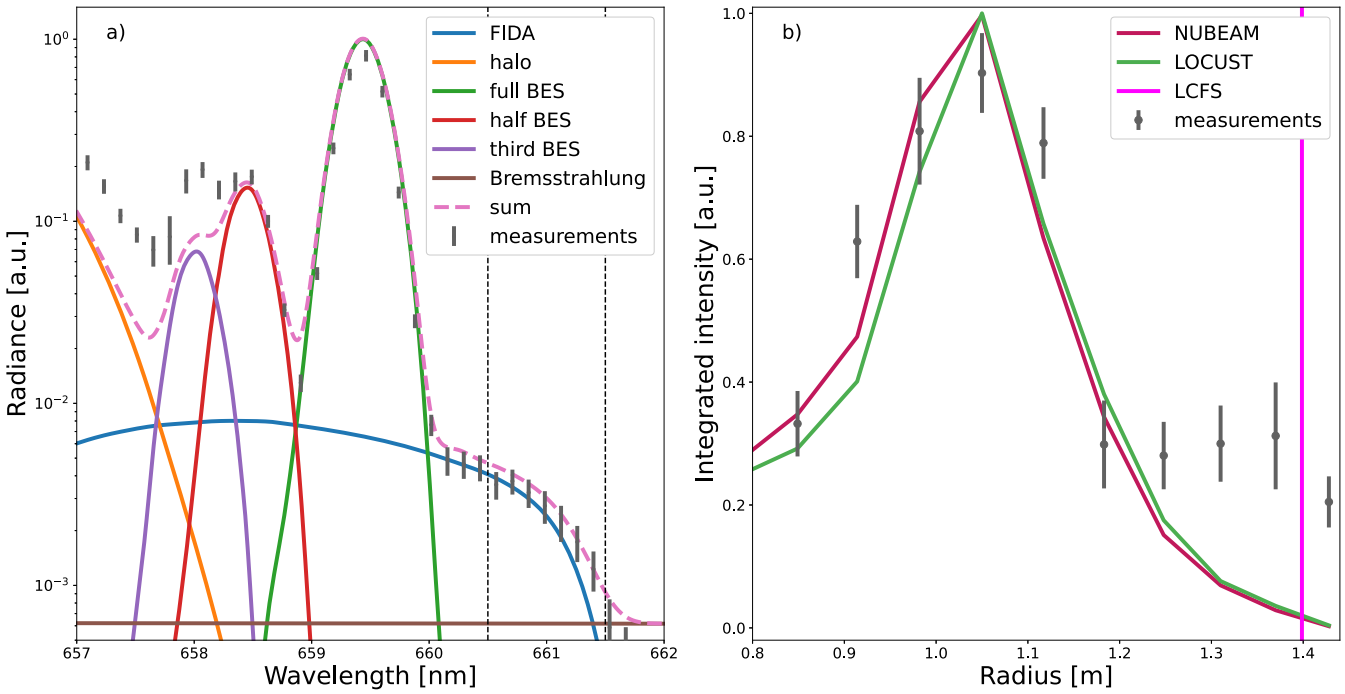

Figure 12. (*a*) Theoretical and measured signal intensities in MAST shot #29034 at 300 ms for one channel. The wavelength filter denoted by vertical dashed lines encompasses 660.5–661.5 nm and is used to integrate all signals across all channels to produce plot (*b*). Some signals are still left out near 660 nm to avoid integration of beam emission in other channels. (*b*) Radially resolved FIDA signal measured in MAST and as produced by `FIDASIM` for `NUBEAM` and `LOCUST`. An FLR correction was applied post-simulation.

corresponding probabilities $P(D_{\text{measured}} < D)$, are shown in figure 11 for matching pairs of $f(\epsilon)$. Typically, the null hypothesis of the KS test, that the two empirical distribution functions to be compared are drawn from the same distribution function, is *rejected* if the measured KS statistic D_{measured} satisfies $P(D_{\text{measured}} < D) \leqslant \alpha$, with α typically chosen to be $\leqslant 0.05$—here we increase α to 0.1 for added rigour. For calibration, figure 11(*a*) first shows the KS statistic for the GC and FO `LOCUST` simulations. In this case $P \approx 1$ ($D_{\text{measured}} \approx 0$), meaning these distributions comfortably agree as expected. Contrast this to the next KS statistic, where the collision operator has been truncated, and the test clearly fails. Next we see that the `LOCUST` FO simulation easily agrees with the `ASCOT` equivalent, with $P \approx 1$ after using the same Coulomb logarithm. The equivalent GC comparison fails solely due to the high-energy tail effect, since the equivalent measurement for the `NUBEAM` deposition in figure 11(*b*) passes. Finally, we also observe that `NUBEAM` decisively matches `LOCUST`.

4.3. Synthetic diagnostics

To validate that these comparisons for MAST are realistic, the distribution functions calculated from the `NUBEAM` deposition by `LOCUST` (GC) and `NUBEAM` were fed into `FIDASIM` to generate synthetic FIDA measurements. The predicted and total measured signals for this spectral range are shown in figure 12(*a*). Signals within a 660.5–661.5 nm gate are shown radially resolved in figure 12(*b*), along with each predicted signal from `FIDASIM`.

Within the core plasma, signals from `LOCUST` and `NUBEAM` are within the smallest error bars of each other so as to be indistinguishable by the FIDA diagnostic. Despite the gate, the presence of background bremsstrahlung emission is still observed outside of $R_{\text{major}} = 1.25$ m and the LCFS in figure 12(*b*). Current error estimates do not take background light into account, or even the lack of a time-dependent background plasma, but if errors are increased globally by $\sim 70\%$ then the reduced χ^2 approaches unity for data points within $R_{\text{major}} < 1.25$ m.

5. Summary

The `LOCUST` code's ability to calculate fast-ion distribution functions in realistic 3D geometries has been described. This includes discussions on the topics of the underlying kinetic physics model, software and hardware implementation, and algorithm design—as well as their mutual influence. In short, assumptions which allow for the independent tracking of fast-ion markers enable the adoption of massively parallel SIMT hardware—for example GPGPUs.

`LOCUST` has been shown to compare well with popular fast-ion codes at a fundamental level over a range of realistic test environments. In the correct circumstances, it was shown that the predictions of all codes converge to within the assumptions of their respective physics models: the accuracy of the collision operator, choice of Coulomb logarithm and FLR model. These comparisons have been validated against experiment. In achieving this, credibility has been added to the

conclusions of other parallel benchmarking activities, which may consider physics not included here [43].

Most importantly, these results demonstrate a readiness for the routine use of LOCUST in physics exploration, virtual engineering and plant design, and sophisticated integrated modelling workflows via the IMAS platform.

Moving forwards, IMAS itself will become increasingly important for the standardisation of verification and validation activities, as codes adapt to studying high-performance devices whereby problems grow increasingly sensitive. Indeed, the prediction [46] that computational tools will need to continually evolve into more integrated workflows is still actively being realised [47]. As such, verification capabilities should evolve accordingly via continuous testing. Fortunately, the modelling community currently has an opportunity to enable this type of software life cycle framework in the form of IMAS. With this capability, the modelling community will be empowered to quickly and confidently adapt to computationally challenging studies in the future.

Acknowledgments

This project was undertaken on the Viking Cluster, which is a high performance compute facility provided by the University of York. We are grateful for computational support from the University of York High Performance Computing service, Viking and the Research Computing team. Additional thanks are also owed to Stuart Henderson, David Keeling, Marina Gorelenkova, James Buchanan, Clive Michael and the team at Aalto University. This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014–2018 and 2019–2020 under Grant Agreement No. 633053. The project was partly undertaken on the ITER Organization's Titan GPU server within its Scientific Data & Computing Centre (SDCC). The work presented in this publication has been carried out under a joint PhD project among the ITER Organization, UKAEA and the University of York and has received financial support from the ITER Organization under Collaboration Agreement LEGAL#6411249v1. The views and opinions expressed herein do not necessarily reflect those of the European Commission. The views and opinions expressed herein do not necessarily reflect those of the ITER Organization.

Appendix A. Guiding-centre equations of motion

The guiding-centre equations of motion as derived in [8] that are solved by LOCUST are included here for reference:

$$\dot{\chi} = \frac{qB}{m}, \tag{A.1}$$

where χ represents gyro-phase,

$$\dot{\mu} = 0 \tag{A.2}$$

$$\dot{v_\parallel} = \frac{q}{mB^\star_\parallel}\mathbf{B}^\star \cdot \mathbf{E}^\star \tag{A.3}$$

$$\dot{\mathbf{R}} = v_\parallel \frac{\mathbf{B}^\star}{B^\star_\parallel} + \mathbf{E}^\star \times \frac{\widehat{\mathbf{b}}}{B^\star_\parallel} \tag{A.4}$$

and where parallel terms are vector components in the direction of the magnetic field $\widehat{\mathbf{b}} = \mathbf{B}/B$. The modified potentials $\mathbf{E}^\star$ and $\mathbf{B}^\star$ are given by

$$\mathbf{E}^\star \equiv -\frac{\partial \mathbf{A}}{\partial t} - \nabla \Phi^\star \tag{A.5}$$

$$\mathbf{B}^\star \equiv \nabla \times \mathbf{A}^\star, \tag{A.6}$$

where

$$\Phi^\star \equiv \Phi + \frac{\mu B}{q} \tag{A.7}$$

$$\mathbf{A}^\star \equiv \mathbf{A} + \frac{m v_\parallel}{q}\widehat{\mathbf{b}}. \tag{A.8}$$

Appendix B. TRANSP settings

A noteworthy conclusion of this study is the environment required to accurately compare the codes on scales that did not disguise discrepancies. For reference, included in table B1 below are the namelist settings required by TRANSP to achieve this fidelity.

Table B1. TRANSP namelist settings.

Setting	Value	Comment
GOOCON	20	Low orbit acceleration
AVGTIM	0.0	No statistical smoothing
DXBSMOO	0.0	No statistical smoothing
NZONE_FB	40	—
NZONE_FP	40	—
NZONE_NB	200	Typically 20–60 *with* smoothing
NZONES	200	—
XBMBND	1.5	Previously locked to 1.3
NPTCLS	10^7	1.7×10^5 in ASCOT, 1.3×10^5 in LOCUST
WGHTA	20.0	—

Appendix C. Coulomb logarithm

The two definitions of the Coulomb logarithm, $\ln(\Lambda)$, used by LOCUST in this study are included here for reference. The complete expressions below describe the Coulomb logarithm in NUBEAM, while removing the terms labelled in colour yields the definition used in ASCOT.

The Coulomb logarithm, for a test particle species i colliding with background plasma species p, is defined as

$$\ln(\Lambda_{ip}) \equiv \ln\left(\frac{\mathrm{b}_{\max,i}}{b_{\min,ip}}\right), \tag{C.1}$$

where the b terms denote the maximum and minimum impact parameters. $b_{\max,i}$ is defined as

$$b_{\max,i} = \left[\sum_p \frac{\omega_p^2 + \Omega_p^2}{\frac{T_p}{m_p} + \frac{2E_i}{m_i}}\right]^{-\frac{1}{2}}, \tag{C.2}$$

with ω_p the plasma frequency,

$$\omega_p^2 = \frac{n_p q_p^2}{m_p \epsilon_0}, \tag{C.3}$$

and Ω_p the cyclotron frequency,

$$\Omega_p = \frac{q_p}{m_p} B. \tag{C.4}$$

$b_{\min,ip}$ is defined as

$$b_{\min,ip} = \max\left(b_{\min,ip}^{\text{quantum}}, b_{\min,ip}^{\text{classical}}\right), \tag{C.5}$$

the largest minimum impact parameter out of the classical expression and the expression with quantum corrections:

$$b_{\min,ip}^{\text{classical}} = \frac{q_i q_p}{4\pi\epsilon_0 \mu_{ip} u_{ip}^2} \tag{C.6}$$

$$b_{\min,ip}^{\text{quantum}} = \frac{\hbar}{2\mu_{ip} u_{ip}} \mathrm{e}^{-\frac{1}{2}}, \tag{C.7}$$

where both define the effective relative velocity u_{ip} as

$$u_{ip}^2 = \frac{3T_p}{m_p} + \frac{2E_i}{m_i} \tag{C.8}$$

and μ_{ip} the reduced mass

$$\mu_{ip} = \frac{m_i m_p}{m_i + m_p}. \tag{C.9}$$

ORCID iDs

S.H. Ward https://orcid.org/0000-0002-0641-2589
A.S. Jacobsen https://orcid.org/0000-0002-9033-1094
E. Tholerus https://orcid.org/0000-0002-3262-1958
R.G.L. Vann https://orcid.org/0000-0002-3105-2546
M.A. Van Zeeland https://orcid.org/0000-0002-7911-2739

References

[1] Pinches S.D., Chapman I.T., Lauber P.W., Oliver H.J.C., Sharapov S.E., Shinohara K. and Tani K. 2015 Energetic ions in ITER plasmas *Phys. Plasmas* **22** 021807
[2] Konsta S. 2020 Efficient and rigorous evaluation of fast particle losses in non-axisymmetric tokamak plasmas *Nucl. Fusion* **60** 036002
[3] Boyer M.D., Kaye S. and Erickson K. 2019 Real-time capable modeling of neutral beam injection on NSTX-U using neural networks *Nucl. Fusion* **59** 056008
[4] Weiland M. *et al* 2018 Rabbit: real-time simulation of the NBI fast-ion distribution *Nucl. Fusion* **58** 082032
[5] Hooker S. 2020 The hardware lottery (arXiv:2009.06489)
[6] Klöckner A., Pinto N., Lee Y., Catanzaro B., Ivanov P. and Fasih A. 2012 PyCUDA and PyOpenCL: a scripting-based approach to GPU run-time code generation *Parallel Comput.* **38** 157–74
[7] Lam S.K., Antoine P. and Seibert S. 2015 Numba: a LLVM-based Python JIT compiler *Proc. 2nd Workshop on the LLVM Compiler Infrastructure in HPC* St. Louis, MO , USA pp 1–6
[8] Littlejohn R.G. 1983 Variational principles of guiding centre motion *J. Plasma Phys.* **29** 111–25
[9] Delzanno G.L. and Camporeale E. 2013 On particle movers in cylindrical geometry for particle-in-cell simulations *J. Comput. Phys.* **253** 259–77
[10] Tretiak K. and Ruprecht D. 2019 An arbitrary order time-stepping algorithm for tracking particles in inhomogeneous magnetic fields *J. Comput. Phys.:* X **4** 100036
[11] Fehlberg E. 1969 *Low-Order Classical Runge–Kutta Formulas With Stepsize Control and Their Application to Some Heat Transfer Problems* (Washington, DC: National Aeronautics and Space Administration)
[12] Cash J.R. and Karp A.H. 1990 A variable order Runge–Kutta method for initial value problems with rapidly varying right-hand sides *ACM Trans. Math. Softw.* **16** 201–22
[13] Dormand J.R. and Prince P.J. 1980 A family of embedded Runge–Kutta formulae *J. Comput. Appl. Math.* **6** 19–26
[14] Goeken D. and Johnson O. 1999 Fifth-order Runge–Kutta with higher order derivative approximations *Electron. J. Differ. Equ.* **2** 1–9
[15] Boris J.P. and Shanny R.A. 1972 *Proc. 4th Conf. Numerical Simulation of Plasmas* vol 3 (2 November 1970) (Washington, DC: Naval Research Laboratory)
[16] Hirvijoki E., Asunta O., Koskela T., Kurki-Suonio T., Miettunen J., Sipilä S., Snicker A. and Äkäslompolo S. 2014 ASCOT: solving the kinetic equation of minority particle species in tokamak plasmas *Comput. Phys. Commun.* **185** 1310–21
[17] Boozer A.H. *et al* 1981 Monte Carlo evaluation of transport coefficients *Phys. Fluids* **24** 851
[18] Hinton F.L. 1983 Collisional transport in plasma *Handbook of Plasma Physics* vol 1 Amsterdam : North-Holland Pub p 147
[19] Galeev A. and Sudan R.N. 1983 Basic plasma physics *Handbook of Plasma Physics* (Amsterdam: North-Holland)
[20] Chandrasekhar S. 1943 Stochastic problems in physics and astronomy *Rev. Mod. Phys.* **15** 1
[21] Pankin A., McCune D., Andre R., Bateman G. and Kritz A. 2004 The tokamak Monte Carlo fast ion module NUBEAM in the national transport code collaboration library *Comput. Phys. Commun.* **159** 157–84
[22] McGuire K. *et al* 1983 Study of high-beta magnetohydrodynamic modes and fast-ion losses in PDX *Phys. Rev. Lett.* **50** 891–5
[23] Wareing T.A., McGhee J.M. and Morel J.E. 1996 ATTILA: a three-dimensional, unstructured tetrahedral mesh discrete ordinates transport code *Trans. Am. Nucl. Soc.* **75** 12
[24] Lütjens H., Bondeson A. and Sauter O. 1996 The chease code for toroidal MHD equilibria *Comput. Phys. Commun.* **97** 219–60
[25] Lao L.L., John H. St., Stambaugh R.D., Kellman A.G. and Pfeiffer W. 1985 Reconstruction of current profile parameters and plasma shapes in tokamaks *Nucl. Fusion* **25** 1611–22
[26] Huysmans G.T.A. and Czarny O. 2007 MHD stability in x-point geometry: simulation of ELMs *Nucl. Fusion* **47** 659–66
[27] Liu Y.Q., Bondeson A., Fransson C.M., Lennartson B. and Breitholtz C. 2000 Feedback stabilization of nonaxisymmetric resistive wall modes in tokamaks. i. electromagnetic model *Phys. Plasmas* **7** 3681–90
[28] Fransson C.M., Lennartson B., Breitholtz C., Bondeson A. and Liu Y.Q. 2000 Feedback stabilization of nonaxisymmetric resistive wall modes in tokamaks. ii. control analysis *Phys. Plasmas* **7** 4143–51
[29] Jardin S.C., Ferraro N., Breslau J. and Chen J. 2012 Multiple timescale calculations of sawteeth and other global macroscopic dynamics of tokamak plasmas *Comput. Sci. Discovery* **5** 014002
[30] Asunta O., Govenius J., Budny R., Gorelenkova M., Tardini G., Kurki-Suonio T., Salmi A. and Sipilä S. 2015 Modelling neutral beams in fusion devices: beamlet-based model

for fast particle simulations *Comput. Phys. Commun.* **188** 33–46
[31] Schneider M., Eriksson L.-G., Jenkins I., Artaud J.F., Basiuk V., Imbeaux F. and Oikawa T. 2011 Simulation of the neutral beam deposition within integrated tokamak modelling frameworks *Nucl. Fusion* **51** 063019
[32] Geiger B. *et al* 2020 Progress in modelling fast-ion d-alpha spectra and neutral particle analyzer fluxes using FIDASIM *Plasma Phys. Control. Fusion* **62** 105008
[33] Assous F., Degond P. and Segre J. 1992 A particle-tracking method for 3D electromagnetic pic codes on unstructured meshes *Comput. Phys. Commun.* **72** 105–14
[34] Matsumoto M. and Nishimura T. 1998 Mersenne twister *ACM Trans. Model. Comput. Simul.* **8** 3–30
[35] Van Zeeland M.A. *et al* 2013 Modulation of prompt fast-ion loss by applied $n = 2$ fields in the DIII-D tokamak *Plasma Phys. Control. Fusion* **56** 015009
[36] Varje J., Konsta S., Kontula J., Ollus P., Kurki-Suonio T., Snicker A., Hirvijoki E. and Äkäslompolo S. 2019 High-performance orbit-following code ASCOT5 for Monte Carlo simulations in fusion plasmas (arXiv:1908.02482)
[37] Breslau J., Gorelenkova M., Poli F., Sachdev J. and Yuan X. 2018 *TRANSP Computer Software* (https://doi.org/10.11578/dc.20180627.4)
[38] Meneghini O. *et al* 2015 Integrated modeling applications for tokamak experiments with OMFIT *Nucl. Fusion* **55** 083008
[39] Van Zeeland M.A. *et al* 2015 Fast ion transport during applied 3D magnetic perturbations on DIII-D *Nucl. Fusion* **55** 073028
[40] Helander P., Akers R.J. and Eriksson L.-G. 2005 On neutral-beam injection counter to the plasma current *Phys. Plasmas* **12** 112503
[41] Harrison J.R. *et al* 2019 Overview of new MAST physics in anticipation of first results from MAST upgrade *Nucl. Fusion* **59** 112011
[42] Michael C.A. *et al* 2013 Dual view FIDA measurements on MAST *Plasma Phys. Control. Fusion* **55** 095007
[43] Sperduti A., Cecconello M., Conroy S. and Snicker A. 2020 Neutron rate estimates in MAST based on gyro-orbit modelling of fast ions *Nucl. Fusion* **61** 016028
[44] Owen J. 2015 Experimental fast-ion transport studies on the mega-amp spherical tokamak *PhD Thesis* Durham University
[45] Kolmogorov A. 1933 Sulla determinazione empirica di una lgge di distribuzione *Inst. Ital. Attuari, Giorn.* **4** 83–91
[46] Becoulet A., Strand P., Wilson H, Romanelli M and Eriksson L.-G. (The Contributors to the European Task Force on Integrated Modelling Activity) 2007 The way towards thermonuclear fusion simulators *Comp. Phys. Commun.* **177** 55–9
[47] Rodriguez-Fernandez P., Howard N.T., Greenwald M.J., Creely A.J., Hughes J.W., Wright J.C., Holland C., Lin Y. and Sciortino F. 2020 Predictions of core plasma performance for the SPARC tokamak *J. Plasma Phys.* **86** 865860503