EFDA–JET–PR(11)02

M. Schneider, L.-G. Eriksson, I. Jenkins, J.F. Artaud, V. Basiuk, F. Imbeaux,
T. Oikawa, ITM-TF contributors and JET-EFDA contributors

Simulation of the Neutral Beam Deposition within Integrated Tokamak Modelling Frameworks

Simulation of the Neutral Beam Deposition within Integrated Tokamak Modelling Frameworks

M. Schneider[1], L.-G. Eriksson[2], I. Jenkins[3], J.F. Artaud[1], V. Basiuk[1], F. Imbeaux[1], T. Oikawa[4], ITM-TF contributors and JET EFDA contributors*

JET-EFDA, Culham Science Centre, OX14 3DB, Abingdon, UK

[1]CEA, IRFM, F-13108 Saint-Paul-lez-Durance, France
[2]European Commission, Research Directorate General, B-1049 Brussels, Belgium
[3]EURATOM-CCFE Fusion Association, Culham Science Centre, OX14 3DB, Abingdon, OXON, UK
[4]ITER Organization, St Paul lez Durance, France
** See annex of F. Romanelli et al, "Overview of JET Results", (23rd IAEA Fusion Energy Conference, Daejon, Republic of Korea (2010)).*

Preprint of Paper to be submitted for publication in
Nuclear Fusion

Abstract

The NEMO (NEutral beam MOdelling) code for simulating the neutral beam ionization during Neutral Beam Injection (NBI) in tokamak plasmas has been developed for its implementation in integrated modelling frameworks and is presented. Integrated modelling of fusion plasmas is becoming increasingly important, both for preparation and analysis of experiments in large devices. Moreover, it should play a crucial role for the design of future fusion reactors. In a modern context, integrated modelling requires codes that are: (i) flexible in terms of their interfaces, i.e. can be adapted for different simulation environments, (ii) machine independent; i.e. they should not contain hard coded information on a particular device to be simulated; (iii) optimised for speed of execution, (iv) verified and validated. The NEMO code has been specially designed to meet these requirements. The code is based on the physics concept outlined in [1] and is a completely modular program: it works with any input NBI geometry and can be coupled to any external Fokker-Planck calculation for evaluating the distribution function of the injected species, i.e. it can provide source terms for both Monte Carlo codes and codes using finite difference/elements methods. The NEMO code has already been integrated with the CRONOS integrated modelling suite [2] and the European Integrated Tokamak Modelling Task Force (ITM-TF) [3]. The basics of the code are described in this paper along with an illustration of its integration in the ITM-TF simulation platform. A crucial aspect is the verification of the code, the results of benchmarks carried out with other NBI codes for JET and ITER discharges are thereby presented.

1 Introduction

In present day tokamaks as well as in ITER, auxiliary heating is needed to reach fusion relevant plasma temperatures. One of the most important sources of such heating is Neutral Beam Injection (NBI). Consequently, simulation of NBI deposition in fusion plasmas is essential to prepare scenarios for experiments, and to predict the plasma performances for future devices. NBI simulations have to be included in an integrated simulation of tokamak discharges. For instance, the European ITM-TF is in the process of developing a general tokamak simulator in order to provide whole device simulations of tokamaks. The challenge of this project is to create a fully modular environment in order to allow for various types of simulations where different codes solving the same physics problem can easily be interchanged. In this context, new concepts of data structure and workflow organisation have been developed, as described in [4]. The data structure concept implies that physics codes must be machine independent, i.e. they should not contain any hard coded information for a particular device, and that they are fully modular. Furthermore, codes running in integrated simulation frameworks should preferably be fast and have a basic structure that allow for a flexible interfacing with different frameworks. Turning to the simulation of NBI, this generally require a combination of two codes: (i) an

NBI ionization code, which calculates the number of injected particles that are ionized per unit volume and time in the plasma; (ii) a Fokker-Planck code which simulates the slowing down of the resulting ions. In this paper we concentrate on the former. Many NBI ionization codes are already available, with varying levels of physics fidelity and performance in terms of computing time. One can distinguish between two main types: (i) narrow beam models, with either zero or finite width, which are based on a combination of analytic and numeric resolution of the problem and therefore can be relatively fast; (ii) Monte Carlo models, based on a complete numeric resolution of the problem, allowing in principle for very accurate results, but requiring a much longer computing time. A beam model with finite width beam is a good compromise between calculation accuracy and rapidity of execution as long as the approximations associated with the injector geometry are adequate. The model presented in Ref. [1] presents an approach to the device geometry that is quite accurate while still allowing for rapid evaluation of the source of ionized particles. For this reason, the NEMO module is based on this same model. In order to fulfil the integrated tokamak modelling requirements, the model has been generalised to any NBI injector geometry. Furthermore, NEMO code is written in fortran 95 and extensively uses modern computing features allowed by this fortran version (dynamic allocations of arrays, derived types etc). Obviously, modularity is a key feature in the design of NEMO. It is currently integrated into the CRONOS transport suite of codes as well as the European ITM-TF platform. Its modularity allows for a flexible Fokker-Planck calculation to be used in conjunction with NEMO in order to calculate the distribution function of the injected ions. As a result, one has the choice of using it with fast as well as more accurate simulations of the neutral beam macroscopic quantities, like current drive, plasma heating etc. The NEMO model, including beam attenuation cross sections and some other physics features together with code numerics and computing performances are described in section 2. Section 3 describes the implementation of the NEMO code in the ITM-TF platform. Simulations of a JET hybrid discharge and the ITER steady-state reference scenario 4 are presented in section 4, providing benchmarks with currently used NBI codes. Finally, conclusions are drawn in section 5.

2 The narrow beam model

Before describing the model, the terminology used in the present paper has to be clarified: a “source” refers to a neutral beam injector, which should not be confused with a box of

injectors. For example, the JET NBI system [5] consists of 2 injection boxes, each one made of 8 injectors, i.e. 8 sources. Each of them consists of beamlets (holes the injected particles passes through before entering the vacuum chamber of the tokamak).

The model used in NEMO to describe the beam geometry is based on the narrow-beam model described in Ref. [1]. This model relies on the approximation that the dimension of the source is very small compared to the distance between the source and the plasma. This is illustrated in Fig.1 showing the geometry of JET NBI injectors. This model presents a good compromise between accuracy and computation performance. Moreover, it allows for easy upgrades, as well as easy coupling with an external Fokker-Planck calculation for simulating the distribution function of the injected ions.

The model rests on one key simplification: the distances (or more precisely the attenuation) from two different points on an injector surface to a point in the plasma are approximated to be the same. This reduces the calculation of the attenuation of the beam to that along the central line from the injector surface to the plasma. Here it should be reminded that an injector surface normally consists of a number of beamlets. In the model considered here the discretness introduced by the beamlets is neglected, instead the injection is "smeared out" over the whole surface covered by the beamlets. When calculating the beam intensity distribution, the beam divergence and focussing properties of the injection geometry are properly accounted for, as illustrated in Fig. 2. The concept and justification for this model are discussed in detail in Ref. [1] and will not be repeated here.

We now turn to a more detailed description of the salient features of the NEMO code. The first crucial point is to determine the beam intensity distribution in a plane perpendicular to the direction of the central line of the injection direction, see Fig.3. A point on this surface is characterised by the beam path length and the two spherical angles as shown in Fig.3. The beam intensity on the plane is calculated by integrating over contribution from the injector surface taking divergence and focussing properties into account. As remarked earlier this process is greatly simplified by the approximation that the attenuation from different parts of the surface is approximately the same (i.e. they have experienced the approximately same attenuation). In fact, for simple arrangements of beamlets, such as approximately circular and rectangular, analytic expressions for the beam intensity distribution have been obtained [1]. The calculation is carried out for each three energy component of the injected neutrals separately (full, half and third energy for positive ion beams). Furthermore, for one energy component, the neutral beam is

considered as monoenergetic.

Compared to a more precise Monte Carlo method where a large number of test particles have to be injected from each beamlet with a direction probability consistent with the beam divergence and then followed until they have been ionized, only one attenuation calculation along the central line from the injector is necessary once the beam intensity distribution has been determined.

2.1 Beam intensity distribution

Let us now consider the mathematical details of the calculation of the beam intensity distribution, without beam attenuation at first. The normalized rate by which fast neutrals leave the injection surface per unit area is denoted by $g(x_b, y_b)$, where x_b and y_b represent the spatial coordinates on the source face, referenced as the B point in Fig. 3. Furthermore, we introduce the function $f_j(x_b, y_b, \theta_b)$ for each injection energy j (full, half and one third for positive ion beams). It represents the rate of atoms leaving a point on the injector surface in a given direction per unit area and unit solid angle, where θ_b is the angle between this direction and the main axis of beamlets near this point (remember that we "smear out" the discrete beamlets over the surface). The beamlet axis can be parametrised as: $(x, y) = ([1 - \frac{z}{z_h}]x_b, [1 - \frac{z}{z_v}]y_b)$, where z_h and z_v are the horizontal and vertical focal lengths of the injector respectively and z is the position along the main beam line. Assuming a small beam divergence the function $f_j(x_b, y_b, \theta_b)$ is approximately given by:

$$f_j(x_b, y_b, \theta_b) = \frac{C_j \dot{N}_{inj}}{\pi \Delta\theta_j^2} g(x_b, y_b) \exp\left[-\left(\frac{\theta_b}{\Delta\theta_j}\right)^2\right] \tag{1}$$

where C_j is the particle fraction for the energy component j, $\dot{N}_{inj}$ is the total rate of injected particles and $\Delta\theta_j$ is the beam divergence for each energy component.

In the above expression the functions g and f_j are normalized such that $\int_S g(x_b, y_b)\, dx_b\, dy_b = 1$ and $\int_S f_j(x_b, y_b, \theta_b)\, dx_b\, dy_b\, \sin\theta_b\, d\theta_b\, d\varphi = C_j \dot{N}_{inj}$, where S is the source face surface.

Let us now turn to the calculation of the normalized beam intensity distribution, $\bar{I}_j(z, \theta, \varphi)$, in a plane with a normal in the direction of the main beam axis (starting at the central point on the surface). The angles θ and φ are illustrated in Fig. 3.

As discussed above, we evaluate the local normalized intensity distribution $\bar{I}_j(z, \theta, \varphi)$ in different planes along the beam path length in the plasma, i.e. along the z coordinate. In Fig. 3, P is a point (x_p, y_p, z_p) in the plane where $\bar{I}_j$ is evaluated, B is a point (x_b, y_b) on

the injector surface contributing to $\bar{I}_j$ and A is the point where the main axis of an injector representative of point B intersects the plane (i.e. the plane including the point P). Due to the beam divergence, each point on this plane has a contribution from every point of the injector surface, with the contribution being weighted by the divergence. Since we have adopted the approximation to neglect differences in the path length between different points on the injection surface and points in the plane, we can express the beam intensity distribution in the plane as:

$$\bar{I}_j(z,\theta,\varphi) = \iint_S \frac{1}{\dot{N}_{inj} C_j} f_j\left(x_b, y_b, \theta_b(x_b, y_b, z, \theta, \varphi)\right)\ dx_b\ dy_b, \tag{2}$$

It is normalized such that:

$$\int_0^{\pi}\int_0^{2\pi} \bar{I}_j(z,\theta,\varphi)\sin\theta\ d\theta\ d\varphi = 1, \tag{3}$$

Using Fig. 3, one can demonstrate the following expression for the angle θ_b in terms of x_b, y_b, z, θ and φ, see appendix A:

$$\cos\theta_b = \frac{-x_b(x-x_b)\frac{z}{z_h} - y_b(y-y_b)\frac{z}{z_v} + z^2}{\sqrt{\left(\frac{z}{z_h}x_b\right)^2 + \left(\frac{z}{z_v}y_b\right)^2 + z^2}\sqrt{(x-x_b)^2 + (y-y_b)^2 + z^2}} \tag{4}$$

where x and y can be expressed in terms of θ, φ and z as:

$$\begin{cases} x &= z\tan\theta\cos\varphi \\ y &= z\tan\theta\sin\varphi \end{cases} \tag{5}$$

Finally, the expression of the beam intensity distribution is given by:

$$\bar{I}_j(z,\theta,\varphi) = \frac{1}{\pi\Delta\theta_j^2}\iint_S g(x_b, y_b)\exp\left[-\left(\frac{\theta_b(x_b, y_b, z, \theta, \varphi)}{\Delta\theta_j}\right)^2\right] dx_b\ dy_b, \tag{6}$$

Eq. (6) is directly used in NEMO, while further approximations were made in [1] for rectangular and circular source geometries. However, those approximations were found to be incompatible with e.g. the DIII-D neutral beam injector geometry. For this reason the more accurate evaluation is used in NEMO.

2.2 Neutral beam attenuation

We now turn to the calculation of the beam attenuation, which is needed in the evaluation of the local intensity. For the rest of the paper, a new system of coordinates (X, Y, Z) is introduced, where (X, Y) defines the view from above of the torus and (X, Z) its side

view, i.e. Z represents the symmetry axis of the tokamak. Both views are illustrated in Fig. 4. In this system, a new coordinate is introduced for describing the neutral beam attenuation, $s(X, Y, Z)$, which characterizes the neutral beam path length in the plasma; s is defined so that $s = 0$ at the tangency point, i.e. the closest point to the Z-axis along the beam. The relation between the path length s and the z coordinate defined in section 2.1 is a simple translation, i.e. $s = z - d_{XY}/\cos\Theta_s$, where d_{XY} is the projection of the distance between the center of the source and the tangency point in the (X, Y) plane, and Θ_S is the angle between the projection in (X, Z) plane of the injection direction and the X-axis. The normalized local intensity distribution is then expressed as $\bar{I}_j(s, \theta, \varphi)$ instead of $\bar{I}_j(z, \theta, \varphi)$.

Neutral particles are attenuated all along the beam, reducing the beam intensity accordingly. The neutral beam attenuated intensity at the position s along the path length is expressed as:

$$I_j(s, \theta, \varphi) = I_{0j} \exp(-\eta) = C_j \dot{N}_{inj} \bar{I}_j(s, \theta, \varphi) \exp\left(-\int_0^s \frac{ds}{\lambda}\right), \tag{7}$$

where $I_{0j} = C_j \dot{N}_{inj} \bar{I}_j(s, \theta, \varphi)$ is the incident neutral beam particle rate, η is the attenuation factor defined as $\eta = -\int_0^s ds/\lambda$, s is the path length, ds is the path length element, and λ is the mean free path length.

The rate of deposited particles in a point (s,θ,φ), i.e. the rate of new-born fast ions, is defined as the derivative $dI_j(s, \theta, \varphi)$ of the beam intensity $I_j(s, \theta, \varphi)$ with respect to the path length s:

$$d\dot{N}(s, \theta, \varphi) = -dI_j(s, \theta, \varphi) = C_j \dot{N}_{inj} \bar{I}_j(s, \theta, \varphi) \exp\left(-\int_0^s \frac{ds}{\lambda}\right) \frac{ds}{\lambda}, \tag{8}$$

For the evaluation of this deposition rate, the mean free path length λ needs to be evaluated. Three different processes are involved in the attenuation process: ionization (both on electrons and ions), charge exchange and capture by impurities, leading to the following expression of the mean free path length λ:

$$\frac{1}{\lambda} = \frac{< \sigma_e v_e >}{v_{beam}} n_e + \frac{< (\sigma_{cx} + \sigma_{ion}) v_{rel} >}{v_{beam}} n_i + \sum_j \sigma_j n_j, \tag{9}$$

where σ_e and σ_{ion} are the stopping cross sections on electrons and ions respectively, σ_{cx} is the charge exchange cross section, v_{beam} is the beam velocity, v_e is the electron velocity, v_{rel} is the mean thermal velocity defined as $v_{rel} = v_e - v_{beam}$, n_e and n_i are the

electron and ion density respectively, n_j and σ_j are the density and capture cross section for each impurity j, calculated locally along the plasma profile.

In NEMO, two different kind of cross sections can be used: those from the Janev model [6], using the updated calculation described in [7], and those from the ADAS database [8]. In both cases multistep ionization processes are included. Comparisons have shown that results are very similar using the two different cross sections for neutral beam injection systems on current tokamaks. All results shown in the present paper have been simulated using ADAS cross sections.

Calculating the number of deposited particles at each point along the beam trajectory in the plasma, the heating profile and fast ion birth matrix can be deduced. The deposition profile is obtained by dividing the sum of local attenuation elements $d\dot{N}$ at a given toroidal flux coordinate by the local volume element, i.e. the volume between two magnetic flux surfaces in a radial grid neighbouring the local toroidal flux coordinate. This (R, Z) grid is defined so that to cover the whole plasma in the poloidal view, with a discretization dR and dZ sufficiently fine to resolve the characteristic spatial scales of the ionisation profile. The fifth-order birth matrix $B(injector, E, R, Z, \xi)$ is filled with the local attenuation elements $d\dot{N}$ divided by the phase space volume $2\pi R\ dR\ dZ\ d\xi$, as a function of the injector index, the energy component, the R and Z indices, and the pitch angle index, where the pitch angle ξ depends on the beam injection direction:

$$\xi = \frac{|Y|}{R} D, \tag{10}$$

where Y and R are spatial coordinates of the particle at its attenuation position, calculated further in the paper in Eqs. (12) and (13), and D is the beam directivity.

In its present status, NEMO does not account for the interaction of fast injected ions with the fast neutrals, nor the interaction between fast neutrals and cold ones located in the scrape-off layer (SOL) or at the edge of the plasma.

2.3 Neutral beam trajectory

The neutral beam ionization occurs all along the beam trajectory in the plasma. As mentioned previously, the geometry of the system is based on the path length coordinate $s(X, Y, Z)$ defined in the 3D (X, Y, Z) space all along the beam trajectory.

The (X, Y, Z) coordinates of the neutral particle at its ionization point have to be deduced from the $s(X, Y, Z)$ coordinate according to the geometry of the system defined

in Fig. 4. In this figure, two spherical coordinate systems are involved: the first one, (r, Θ, Φ) is defined such as Φ rotates around the X-axis; the second one, (r, θ, φ) is defined such as φ rotates around the principal beam injection direction. Hence, θ is the angle covering the full divergence cone of the beam, with the φ angle defined around the beam direction in the same spherical coordinate system as θ, as previously shown in Fig. 3. The Φ_s and Θ_s angles are the angles between the projection in the (X, Y) plane and in the (X, Z) plane respectively, of the injection direction of the central point of the injector surface and the X-axis. R_T is the tangency radius, defined between the Z-axis and the tangency point, and R_0 and a are the plasma major and minor radii respectively. Using this geometry, the Φ_s angle can be expressed as:

$$\Phi_s = -\arctan\left(\frac{Y_s}{X_s}\right) + D \times \arctan\left(\frac{R_T}{d_{XY}}\right) \times sign(I_P), \tag{11}$$

where d_{XY} is the projection in the (X, Y) plane of the distance between the center of the source and the tangency point, D is the beam directivity, defining the neutral beam co/counter-injection, i.e. $D = 0$ for strictly perpendicular injection (convention: "+1" for co-injection), and $sign(I_P)$ defining the direction of the plasma current (convention: "+1" refers to the anticlockwise direction in the view from above).

In order to evaluate the (X, Y, Z) coordinates of the fast ions at their birth position, one has to consider the travelled distance $s + d$ of the neutral beam from the source in the three-dimensional space, where d is the distance in the three-dimensional space between the center of the source and the tangency point. Using this geometry, the equations for the (X, Y, Z) coordinates of the particle can be expressed as following:

$$\begin{cases} X &= X_s - (s+d) \times \cos(\Phi_s - \theta\cos\varphi) \times \cos(\Theta_s - \theta\sin\varphi) \times sign(X_s) \\ Y &= Y_s + (s+d) \times \sin(\Phi_s - \theta\cos\varphi) \times \cos(\Theta_s - \theta\sin\varphi) \times sign(X_s) \\ Z &= Z_s + (s+d) \times \sin(\Theta_s - \theta\sin\varphi) \end{cases} \tag{12}$$

The radial coordinates R of the particle is then defined as:

$$R = \sqrt{X^2 + Y^2} \tag{13}$$

The three-dimensional geometry of JET neutral beam injectors is displayed in Fig. 1. As already mentioned, each beam trajectory is calculated as coming from the centre of each injector. The correction according to the true dimension of the source is carried out in the calculation of the beam intensity distribution described previously.

2.4 Finite Larmor Radius effects (FLR)

In order to allow codes based on gyro-averaged kinetic equations to directly use the output of NEMO, the birth profile in guiding centre coordinates is also provided as an output. We denote the shift of the guiding centre position with respect to the birth position by $(\delta R, \delta Z) = -\overrightarrow{\rho_{L,B}}$, where $\overrightarrow{\rho_{L,B}}$ is the position of the ion in its Larmor orbit at the birth point with respect to its guiding centre position. With the aid of $\overrightarrow{\rho_{L,B}} \cdot \overrightarrow{B} = 0$ and $\overrightarrow{\rho_{L,B}} \cdot \overrightarrow{v_\perp} = 0$ one can show that (appendix B):

$$\begin{cases} \delta R = -\delta Z \left(\dfrac{B_Z v_\varphi - B_\varphi v_Z}{B_R v_\varphi - B_\varphi v_R} \right) \\ \delta Z = -\dfrac{\rho_L sign(\hat{Z} \cdot (\hat{R} \times \overrightarrow{B}))}{\sqrt{1 + \left(\frac{B_Z v_\varphi - B_\varphi v_Z}{B_R v_\varphi - B_\varphi v_R} \right)^2 + \left(\frac{B_Z}{B_\varphi} + \frac{B_R}{B_\varphi} \left[\frac{B_Z v_\varphi - B_\varphi v_Z}{B_R v_\varphi - B_\varphi v_R} \right] \right)^2}} \end{cases} , \tag{14}$$

where (B_R, B_Z, B_φ) and (v_R, v_Z, v_φ) are respectively the components of the magnetic field and of the particle velocity in the (R, Z, φ) space, the Larmor radius is given by $\rho_L = v_\perp/\omega$, and ω is its cyclotron frequency, defined as $\omega = ZeB/m$, where Ze is the particle charge, B is the local value of the magnetic field and m is the particle mass.

2.5 Wall geometry for shinethrough calculation

In order to quantify the shinethrough in the plasma, i.e. the power lost due to the beam intersecting the tokamak wall, one has to know the exact wall geometry. This geometry is displayed in Fig. 5 for ITER, JET and DIII-D tokamaks. When the beam hits the wall the remaining power is considered lost.

As can be seen in Fig. 5 for JET, the sometimes adopted approximation of assuming that the remaining power in a beam is lost when it reaches a major radius corresponding the wall position in the midplane, can be quite inaccurate, since the major radius of the vacuum vessel is not constant along the Z-axis. For this reason, the full wall geometry (in the 2D axi-symmetric approximation) is provided in NEMO for a better evalulation of the shinethrough losses. For cases where the wall coordinates are not available it is obviously possible to resort to the vessel radius approximation instead.

2.6 Fast ion torque density profile

The fast ion torque density at birth is provided by NEMO. It is useful especially for diagnosing the torques calculated by kinetic codes using NEMO data as input (there must be toroidal momentun conservation), it could also be used directly by transport

codes if finite orbit width effects are neglected and one is close to steady state. The torque provided by the ionized particles is given by Eq. (15):

$$T(\rho) = mRv_{\parallel}\frac{B_{\varphi}}{B}\frac{\dot{N}_{\rho}}{V}, \tag{15}$$

where m is the fast ion mass, R its radial coordinate, $v_{\parallel}$ its parallel velocity, B and B_{φ} are the magnetic field and its toroidal component respectively, $\dot{N}_{\rho}$ is the local ionization rate, and V is the volume element at the toroidal magnetic flux surface ρ. It should be pointed that the correct torque to be used in transport codes is the one calculated by a kinetic code (normally a Fokker-Planck code) using the NEMO fast ion birth profile as input. This can be especially important for the dynamics of the torque since the torques associated with ions born on trapped orbits is transferred to the thermal plasma on time scale of the bounce time whereas that associated with ions born on passing orbits is transferred on the slowing down time scale. As remarked above, finite orbit width effects is also a factor that can be important.

2.7 Numerics and computing performance

NEMO is written in fortran 95, allowing for arrays and matrices to be allocated dynamically. The main NEMO input and output variables are listed in table 1.

The global flow chart of the code, illustrating the algorithm used for calculating the beam deposition and attenuation, is displayed in Fig. 6. One can distinguish six main steps:

1. At the first call to the code a number of initializations are made, especially the normalized beam intensity function, $\bar{I}(s,\theta,\varphi)$ is evaluated since it is unaffected by the plasma parameters (it only depends on the injection geometry).
2. Cross sections for the beam attenuation are calculated at every call to the routine using input density and temperature profiles as well as the beam characteristics (energy, mass, charge) and the plasma 2D equilibrium.
3. For each injector and energy component, the beam attenuation along the central injection path of an injector surface is calculated, i.e. $\eta = -\int_0^s ds/\lambda$ is evaluated.
4. For each injector surface, energy component, position along the beam path and set of angles (θ,φ), the position and pitch angle are determined.
5. If this position is in the plasma, the number of ions born per unit volume and pitch angle at this point, $d\dot{N}_{inj}/ds$, is evaluated and the birth deposition matrix

is updated. If, on the other hand, the point intersects the wall, the shinethrough vector is filled in.

6. Finally, since the calculation of ionization deposition is numerical, a final (normally small) normalization is applied to ensure that the total power and rate of injected particles is exactly correct.

The computing time required by NEMO for one injector is typically between 15 and 20 seconds on a 3 GHz Intel(R) Xeon(R) CPU E5450 processor with 16 Go of RAM (computer Random Access Memory). This is slower than simple pencil models but remains much faster than Monte Carlo models which provides the same level of accuracy.

3 NEMO on the ITM-TF platform

The simulation infrastructure of the European Integrated Tokamak Modelling Task Force is based on standardised interfaces for codes solving a particular physics problem. This allows for an almost effortless exchange of codes running in the ITM-TF code platform. This has many advantages, e.g. new codes solving a certain problem can easily be benchmarked against existing codes using exactly the same input data; it easy to assess whether codes of a greater physics fidelity is needed or if a more simplified code can be used for a certain type of simulation; validation of codes using experimental data can be carried out in a uniform and transparent way. The key concept of the standardised interfaces is Consistent Physical Objects (CPOs), which are described in detail in [4]. The CPOs are objects (data structures) containing all data considered relevant to a certain physics process (e.g. magnetic equilibrium, core plasma properties, wave power deposition etc). Obviously, the data structure of a CPO can evolve as progress is made in the physics description of a certain process. Generally, if motivated, additions to a CPO can be easily accommodated whereas changes to the existing structure of a CPO must be very well justified. Each code integrated into the ITM-TF framework is a subroutine with an interface where all physics data, input and output, are communicated by CPOs, i.e. a routine has CPOs as formal arguments The procedure to make NEMO conform to ITM-TF requirements was straightforward. NEMO requires three CPOs as an input: *equilibrium*, describing the 2D axi-symmetric tokamak equilibrium, *coreprof*, containing the core plasma 1D profiles (temperature, density, etc) which are taken either from the output of a core transport solver or from experimental data, and *nbi* containing the injector geometry and the injection characteristics (energy, power etc). As an output, NEMO fills a CPO, called

distsource describing the source of particles for input to kinetic solvers like Fokker-Planck calculations, which then fills the *distribution* CPO with fast ion distribution functions. Within the ITM-TF platform, physics codes are represented by elementary acting units, or actors, linked together in a global workflow for describing interactions between physics problems. ITM-TF workflows are presently created using the KEPLER software [9], providing a graphic representation of the interactions between the physics modules. Once the NEMO source code is formatted as a subroutine with CPOs as input/output, the NEMO Kepler actor is generated by automated tools making it ready for use in the Kepler workflow. Note that this is done without any modification of the source code. The NEMO actor is illustrated in Fig. 7, displaying its input and output CPOs.

In order to test the implementation of NEMO on the ITM-TF platform, a test workflow has been created as illustrated in Fig. 8. It is a quite generic workflow (it can be adapted to other heating and current drive methods) the main purpose of which is for validation of the NBI modelling against experimental data or for simply analysing the NBI heating in a discharge. The workflow carries out a time loop at fixed time steps. At each time slice, NBI and Fokker-Planck calculations are executed.

Without entering into the details of this workflow, the description of its running procedure can be summarised in seven steps:

1. **Time and database parameters:** the simulation time and output characteristics have to be defined: in the present workflow, the simulation is carried out from $t_{begin} = 0.1$ s to $t_{end} = 3.8$ s with a time step of $dt = 0.1$ s, for the result to be saved in the local database with the run number #75.

2. **Read input from database:** the workflow starts with UALinit, which takes as input a tokamak name, a discharge number, and a run number (used to distinguish different runs or versions of data). This actor output the plasma data needed for the simulation, i.e. the equilibrium and the core profiles. As indicated above, the data could come from an experiment or from a transport simulation (in the illustration presented here the latter is the case). The time and CPOs (data) in the workflow are bundled together so as to have a coherent set "flowing" through the lines connecting the actors in the main part of the workflow. In this case the *equilibrium* and *coreprof* CPOs, filled with data from UALinit, are bundled together with time point of the simulation (initialised to t_{begin}) and the output CPOs to be filled by the actors in the workflow.

3. **Time step loop controller:** it is required to keep track of the time in the simulation and to end it when the maximum time, t_{end}, of the simulation has been reached. It should be noted that the simulation time is increased after each passage of the simulation loop by dt. At the first time step controller shunts in the data bundle of the point above into the main loop of the calculation. When the end time has been reached it directs the simulation data to the actor that writes them to the database.

4. **Physics calculation:** physics modules are called, and contained in "composite actors", which are nested subworkflows used to clarify the graphical display. The first actor of the workflow fills in the *nbi* CPO containing the injector geometry and characteristics like the injection energy and the power (in the case of data taken from a experiment this data should have been obtained via UALinit directly). Then, the NEMO actor is called in order to fill the *distsource* CPO, containing the fast ion initial state required by the Fokker-Planck actor following the fast ion distribution function to fill the *distribution* CPO.

5. **Visualization:** for each time step, the present time and fast ion energy content is extracted to be transferred to a XY plotter showing the energy content time evolution during the simulation.

6. **Update current time:** the time is incremented of dt for the next simulation step.

7. **Write final results to database:** finally, when the end time has been reached the *distsource* and *distribution* simulation data are directed to the actor UALcollector that writes them to a local database (here for shot #5 and run #75).

In order to better understand the implementation of NEMO in the workflow, Fig. 9 displays the details of the NEMO composite actor shown in Fig. 8 for NBI particle source rate.

The NEMO composite actor can be described in four steps:

1. First, the descriptors of the input CPOs, *equilibrium*, *coreprof* and *nbi*, needed by NEMO must be extracted from the bundle entering the composite actor. Furthermore the time is required such that the wrapper to the NEMO module can obtain the CPOs at the relevant time point.

2. The NEMO actor, which contains the original fortran source code, is then called, receiving the CPOs as input. After its main calculation it fills its output CPO, *distsource*, with the result of the calculation.

3. In order to ensure the coherence of the bundled data in the main loop of the workflow, all the data except the CPO outputted by the NEMO actor must bypass it and be joined up with its output CPO to restore the structure of the bundle.

4. Data are bundled together as an output of the actor to be used by following codes in the workflow (typically a Fokker-Planck code in this type of simulation).

4 Application to JET and ITER

The NEMO code has been applied to the simulation of JET and ITER discharges in order to confront it with other simulation codes, and with experimental data in the case of JET simulations. In order to verify the accuracy of the model, the present section presents the benchmark between NEMO, PENCIL [10] and NUBEAM [11] for a JET hybrid discharge, and the benchmark between NEMO, ACCOME [12], OFMC [13] and NUBEAM for the ITER steady-state reference scenario 4. First, a brief description of each NBI code model, with which benchmarks have been carried out, is presented below:

- NUBEAM: this is a Monte Carlo package for time dependent modelling of fast ions. It includes a model for beam injection and attenuation, and a Fokker-Planck calculation for fast ion propagation. Both are simulated via the Monte Carlo method.
- PENCIL: this is a model where the beam is described by a simple pencil trajectory. The pencil-beamlet method is very fast but less accurate than Monte Carlo methods or beam models. The accuracy is limited by the simplified beam geometry and the zero banana width assumption in the Fokker-Planck package.
- OFMC: this is a guiding-center orbit following Monte-Carlo code that combines a parallel beam with an elliptic footprint and a Gaussian intensity profile with a Monte Carlo calculation for the following of guiding centre orbits.
- ACCOME: it uses the same model as OFMC for the beam injector and the stopping cross sections [7]. However, it is combined with a bounce-averaged 2D Fokker-Planck calculation derived for a circular cross-section plasma.

4.1 Simulation of a JET hybrid discharge

NEMO, associated with the SPOT Monte Carlo code for fast ion propagation [14], has been used in the context of an integrated CRONOS transport simulation of the JET discharge #76063. This discharge has been chosen since it is well documented in term of

transport and analysis using the TRANSP transport code [15]. This is a high confinement hybrid discharge [16] with an enhancement factor of $H_{98} \simeq 1.3$. The simulation of this discharge using the CRONOS transport code allows for the comparison of global plasma quantities with experimental measurements and associated fitting procedures. Furthermore, results can be compared to TRANSP allowing for a benchmark between NEMO and NUBEAM. PENCIL results are shown as well, for completeness.

The scenario of discharge #76063 is displayed in Fig. 10 showing the time evolution of the applied NBI power P_{NBI}, the plasma current I_P, the line-integrated density n_{bar} and the central electron temperature T_e^0. As can be seen, the applied NBI power gets strong variations since the total beam power was modulated by the feedback control of the normalized pressure. Those variations impose to apply the NBI code at a frequency of 0.02 second in CRONOS, in order to account for all NBI modulations.

CRONOS and TRANSP have been used to simulate this discharge interpretatively, i.e. using input experimental kinetic profiles with no prediction of the heat transport, but predicting the plasma current evolution. The validity of this kind of simulation is checked by comparing the simulated loop voltage, plasma pressure and internal inductance with measurements. The same input kinetic profiles have been used in both transport codes, with independent internal consistent equilibrium calculations. Fig. 11 shows the evolution of the loop voltage, measured experimentally and simulated by CRONOS and TRANSP. As can be seen, the loop voltage is very well reproduced by both transport codes. Fig. 12 shows the measured and simulated values of $\beta + \ell_i/2$, where β is the plasma toroidal normalized pressure ($\beta = \langle p \rangle / (B_T^2/2\mu_0)$ where $\langle p \rangle$ is the volume averaged kinetic pressure, B_T is the toroidal magnetic field, μ_0 the vacuum permittivity), and ℓ_i the normalized internal inductance. Again, CRONOS and TRANSP simulations show results close to the experiment. They slightly underestimate $\beta + \ell_i/2$, which may be due to the lack of data in the centre, leading to too flat temperature profiles.

Now focussing on NBI-related quantities, Figs. 13 and 14 show the beam deposition profile and the NBCD profile simulated by CRONOS, TRANSP and PENCIL at t=6 sec. The NBCD accounts for the electron shielding back current, using the model described in [17] for PENCIL, and the one described in [18] for both CRONOS (SPOT) and TRANSP (NUBEAM). Fig. 13 shows a good agreement between the three models for the beam deposition. However, in Fig. 14, one can see that PENCIL overestimates the NBCD compared to other models, which may be due to the absence of orbit width effects in the model, while CRONOS and TRANSP show an overall good agreement between each

other. The CRONOS NBCD is a bit larger than the TRANSP one. The reason for this is that, unlike SPOT, NUBEAM accounts for charge exchange and rotation/friction losses.

In order to quantitfy these differences, one has to check the global time evolution of NBI currents and powers integrated over the plasma volume. Those quantities are shown in Figs. 15 and 16 respectively. Fig. 15 shows the evolution of the total plasma current I_P, the bootstrap current I_{BOOT}, the NBCD (I_{NBI}) and the ohmic current (I_{OHM}), simulated by CRONOS and TRANSP. Initial conditions for magnetic fluxes are different for both codes: CRONOS uses the current density profile at $t = 1.3$ sec while TRANSP uses the q-profile at $t = 2.8$ sec. Both codes show an overall good agreement in the prediction of the evolution of the current sources. However, the CRONOS prediction of NBCD is around 20% larger than the TRANSP one. This difference is explained by Fig. 16, showing the time evolution of the NB injected power along with the prediction of the power transferred to bulk electrons (P_{el}) and ions (P_{ion}), the shine-through power losses and the charge exchange and rotation/friction losses (this latter being available only with NUBEAM, not with SPOT). This figure shows a good agreement between SPOT and NUBEAM for the power transferred to the bulk ions and the power lost by shine-through. However, the power transferred to the bulk electrons is around 20% higher with CRONOS than with TRANSP. This difference corresponds to the amount of charge exchange and friction/rotation losses quantified by NUBEAM in TRANSP, representing around 20% of the power transferred to the bulk electrons. These processes are not simulated by SPOT in CRONOS. This explains the 20% difference observed in the prediction of the NBCD using TRANSP and CRONOS. This difference between the two calculations seems to apply to plasmas in various conditions of plasma current and density.

4.2 Simulation of the ITER steady-state reference scenario 4

A simulation of the ITER steady-state reference scenario 4 [19] has been carried out using NEMO associated with the SPOT Monte Carlo code, in order to compare it with other NBI codes, like ACCOME, OFMC and NUBEAM. For this benchmark, the EDA 2001 design of ITER NBI geometry has been used [20].

For the benchmark between NEMO, OFMC and ACCOME, the same equilibrium, kinetic profiles and geometry for neutral beam injectors have been used. Since SPOT and ACCOME do not simulate charge exchange and rotation losses, these processes have been turned off in OFMC. The NEMO, OFMC and ACCOME fast ion deposition profiles are shown in Fig. 17, for the most off-axis injection (3.365 degree downward). As can be

seen, there is a very good agreement between the codes. The residual differences may be due to different beam models (discretization, numeric integrals, etc).

The resulting neutral beam current drive calculated by NEMO/SPOT is 2.19 MA, while it is 2.31 MA for ACCOME and 2.13 MA for NUBEAM [21]: the three values are in a very good agreement, assessing the validity of the NEMO beam model for ITER. These results have been presented in [22] for the benchmark code activity of steady-state operations.

5 Conclusion

In the context of the integrated tokamak modelling frameworks, a new code for neutral beam deposition, NEMO, has been developped on the basis of the narrow-beam model first presented in [1], but generalized to any tokamak and NBI geometry. The NEMO strong asset is its modularity: not only working with any tokamak equilibrium and injector geometry, it can also be coupled to any Fokker-Planck calculation for fast ion trajectories. This modularity is essential in the present context of integrated simulations, allowing its coupling to any global transport code or integrated modelling platform. In this context, NEMO has been implemented within CRONOS and the ITM-TF platform.

Benchmarks have been carried out in order to verify the model generalization and the code accuracy. A first benchmark has been done using NEMO in the CRONOS transport code for simulating a JET experimental hybrid discharge, leading to comparisons with other NBI codes. Results have shown a good agreement between NEMO and NUBEAM for the prediction of the deposition profiles. The only significant difference between CRONOS and TRANSP NBCD predictions is at the level of the Fokker-Planck calculation, accounting for charge exchange and rotation/friction losses in the case of NUBEAM, but not in the case of the SPOT orbit following Monte Carlo code. Another benchmark has been carried out simulating the ITER steady-state reference scenario 4, where the exact same kinetic profiles and equilibrium geometry have been used for all NBI codes. Again, a very good agreement has been observed between NEMO, OFMC, ACCOME and NUBEAM. These results have been used in [22] for the benchmark of NBI codes in steady-state operations.

The next step will be to repeat benchmarks in the ITM-TF environment in order to: (i) verify the implementation of each module within the platform, (ii) better compare codes accuracy and performances. These benchmarks will be much more accurate than

present benchmarks since all modules will automatically receive the exact same set of data from equilibrium.

The views and opinions expressed herein do not necessarily reflect those of the ITER Organization.

References

[1] Y. Feng et *al*, Computer Physics Comm. 88 (1995) 161-172

[2] J.F. Artaud et *al.* Nuclear Fusion, Vol 50, 4, 043001 (2010)

[3] http://www.efda-itm.eu/index.php

[4] F. Imbeaux et al., Comp. Phys. Comm. **181** (2010) 987-998

[5] G. Duesing et al., Fusion Technology, Vol 11 (1987)

[6] R.K. Janev et al., Nuclear Fusion **12**, 2125 (1989)

[7] S. Suzuki et al., Plasma Phys. Control. Fusion **40**, 2097-2111 (1998).

[8] Summers, H. P. (2004) The ADAS User Manual, version 2.6 http://adas.phys.strath.ac.uk

[9] http://kepler-project.org

[10] P.M. Stubberfield and M.L. Watkins, JET note DPA(06)/87, and D.G. Muir, 'JET note DPA(11)/87

[11] A. Pankin et al., Comp. Phys. Comm **159** (2004) 157-184

[12] K. Tani, M. Azumi and R.S. Devote, Journal of Comp. Physics **98** (1992) 332

[13] K. Tani et al., Journal of Phys. Soc. Jpn. **50** (1981) 1726.

[14] M. Schneider, L.-G. Eriksson, V. Basiuk and F. Imbeaux Plasma Phys. Control. Fusion **47**, 2087-2106 (2005)

[15] R.J. Goldston et al., J. Comp. Phys. **43** (1981) 61

[16] E. Joffrin et al., 23 IAEA Fusion Energy conference, Daejeon, Rep. of Korea (2010)

[17] D.F.H Start and JG Cordey, Phys. Fluids **23** (1980) 1477

[18] Y.R. Lin-Liu, F.L. Hinton, Phys. of Plasmas, **4**, (1997) 4179

[19] A.R. Polevoi et al., Proc. 19th IAEA Fusion Energy Conf. (2002) IAEA-CN-94/CT/P-08.

[20] ITER EDA Documentation Series No.24 ITER Technical Basis, IAEA, Vienna, 2002.

[21] T. Oikawa et al., 22nd IAEA Fusion Energy conference, Geneva, Switzerland (2008)

[22] T. Oikawa et al., 5th ITPA IOS, Seoul (2010)

A Demonstration of the θ_b expression from Eq. (4)

Using points B $[x_b, y_b, 0]$, P $[x_p, y_p, z_p]$, and A $[(1-\frac{z_p}{z_h})x_b, (1-\frac{z_p}{z_v})y_b, z_p]$. from Fig. 3, let us define the three distances a, b and c as: $a = \overline{AB}$, $b = \overline{PB}$ and $c = \overline{AP}$. Replacing coordinates of A, B and P points, one gets:

$$\begin{aligned}
\bullet\ a^2 &= \left[\left(1-\frac{z_p}{z_h}\right)x_b - x_b\right]^2 + \left[\left(1-\frac{z_p}{z_v}\right)y_b - y_b\right]^2 + z_p^2 \\
&= \left(\frac{z_p}{z_h}x_b\right)^2 + \left(\frac{z_p}{z_v}y_b\right)^2 + z_p^2 \\
\bullet\ b^2 &= (x_p - x_b)^2 + (y_p - y_b)^2 + z_p^2 \\
\bullet\ c^2 &= \left[x_p - \left(1-\frac{z_p}{z_h}\right)x_b\right]^2 + \left[y_p - \left(1-\frac{z_p}{z_v}\right)y_b\right]^2 \\
&= (x_p - x_b)^2 + 2(x_p - x_b)\frac{z_p}{z_h}x_b + \left(\frac{z_p}{z_h}x_b\right)^2 \\
&\quad + (y_p - y_b)^2 + 2(y_p - y_b)\frac{z_p}{z_v}y_b + \left(\frac{z_p}{z_v}y_b\right)^2
\end{aligned} \tag{16}$$

Using the relation $c^2 = a^2 + b^2 - 2ac\cos\theta_b$, this leads to:

$$\begin{aligned}
\cos\theta_b &= \frac{a^2 + b^2 - c^2}{2ac} \\
&= \frac{1}{2ac}\left\{\left(\frac{z_p}{z_h}x_b\right)^2 + \left(\frac{z_p}{z_v}y_b\right)^2 + z_p^2 + (x_p - x_b)^2 + (y_p - y_b)^2 + z_p^2 - (y_p - y_b)^2\right. \\
&\quad \left. -2(y_p - y_b)\frac{z_p}{z_v}y_b - \left(\frac{z_p}{z_v}y_b\right)^2 - (x_p - x_b)^2 - 2(x_p - x_b)\frac{z_p}{z_h}x_b - \left(\frac{z_p}{z_h}x_b\right)^2\right\} \\
&= \frac{-x_b(x_p - x_b)\frac{z_p}{z_h} - y_b(y_p - y_b)\frac{z_p}{z_v} + z_p^2}{\sqrt{\left(\frac{z_p}{z_h}x_b\right)^2 + \left(\frac{z_p}{z_v}y_b\right)^2 + z_p^2}\sqrt{(x_p - x_b)^2 + (y_p - y_b)^2 + z_p^2}}
\end{aligned} \tag{17}$$

B Demonstration of δR and δZ FLR variations of Eq. (14)

The Larmor radius of a particle is defined as $\rho_L = v_\perp/\omega$, where $v_\perp$ is the perpendicular velocity of the particle and ω is its cyclotron frequency. In the three-dimensional (R, Z, φ) space, the larmor radius is defined as:

$$\rho_L = \sqrt{\delta R^2 + \delta Z^2 + (R\delta\varphi)^2}, \tag{18}$$

where δR, δZ and $\delta\varphi$ are the variations corresponding to the distance between the particle and its guiding centre. Reminding that the three-dimension Larmor radius is perpendicular to the magnetic field and to the particle perpendicular velocity, one can write $\overrightarrow{\rho_L} \cdot \overrightarrow{B} = 0$ and $\overrightarrow{\rho_L} \cdot \overrightarrow{v_\perp} = 0$:

$$\overrightarrow{\rho_L} \cdot \overrightarrow{B} = 0 \;\; \Rightarrow \;\; \delta R B_R + \delta Z B_Z + R\delta\varphi B_\varphi = 0 \tag{19}$$

$$\begin{aligned} \overrightarrow{\rho_L} \cdot \overrightarrow{v_\perp} = 0 \;\; &\Rightarrow \;\; \overrightarrow{\rho_L} \cdot \left(\overrightarrow{v} - v\frac{\overrightarrow{B}}{B} \right) = 0 \\ &\Rightarrow \;\; \overrightarrow{\rho_L} \cdot \overrightarrow{v} = 0 \\ &\Rightarrow \;\; \delta R v_R + \delta Z v_Z + R\delta\varphi v_\varphi = 0 \end{aligned} \tag{20}$$

Multiplying Eq. (19) with v_φ and Eq. (20) with B_φ, one gets:

$$\left\{ \begin{array}{rcl} \delta R B_R v_\varphi + \delta Z B_Z v_\varphi + R\delta\varphi B_\varphi v_\varphi &=& 0 \\ \delta R B_\varphi v_R + \delta Z B_\varphi v_Z + R\delta\varphi B_\varphi v_\varphi &=& 0 \end{array} \right. \tag{21}$$

Taking the difference yields:

$$\delta R(B_R v_\varphi - B_\varphi v_R) + \delta Z(B_Z v_\varphi - B_\varphi v_Z) = 0 \tag{22}$$

Leading to:

$$\delta R = -\delta Z \left(\frac{B_Z v_\varphi - B_\varphi v_Z}{B_R v_\varphi - B_\varphi v_R} \right) \tag{23}$$

Replacing Eq. (23) in Eq. (19):

$$R\delta\varphi = -\frac{\delta Z}{B_\varphi} \left(B_Z + \frac{B_Z v_\varphi - B_\varphi v_z}{B_R v_\varphi - B_\varphi v_R} B_R \right) \tag{24}$$

Replacing Eqs. (23) and (24) in Eq. (18):

$$\begin{aligned} \delta Z^2 &= \rho_L^2 - \delta Z^2 \left(\frac{B_Z v_\varphi - B_\varphi v_z}{B_R v_\varphi - B_\varphi v_R} \right)^2 - \frac{\delta Z^2}{B_\varphi^2} \left(B_Z - \frac{B_Z v_\varphi - B_\varphi v_Z}{B_R v_\varphi - B_\varphi v_R} B_R \right)^2 \\ \delta Z &= -\frac{\rho_L}{\sqrt{1 + \left(\frac{B_Z v_\varphi - B_\varphi v_Z}{B_R v_\varphi - B_\varphi v_R} \right)^2 + \left(\frac{B_Z}{B_\varphi} + \frac{B_R}{B_\varphi} \left[\frac{B_Z v_\varphi - B_\varphi v_Z}{B_R v_\varphi - B_\varphi v_R} \right] \right)^2}} \end{aligned} \tag{25}$$

C List of NEMO fortran routines

The complete list of NEMO fortran routines is listed in table 2.

	Plasma geometry	**Unit**
INPUT VARIABLES	• R_0 and a plasma major and minor radius	m
	• (R, Z) grids of toroidal flux coordinate ρ	m
	• (R, Z) grids of magnetic field components B_R, B_Z and B_φ	T
	• (R, Z) coordinates of the tokamak wall and/or SOL radius	m
	• B_φ and I_P signs	-
	Kinetic plasma quantities	**Unit**
	• Volume profile	m^3
	• Electron temperature profile	eV
	• Electron density profile	m^{-3}
	• A and Z mass and charge number of plasma other species	-
	• Temperature profiles of plasma other species	eV
	• Density profiles of plasma other species	m^{-3}
	Beam geometry and parameters	**Unit**
	• Tangency radius of each source	m
	• X, Y, Z, R coordinates of each source	m
	• Angle between each source and the X-axis in the (X, Z) plane	rad
	• Horizontal and vertical focal length distance of the beam	m
	• Beam divergence of each source	rad
	• Beam directivity of each source	-
	• Height and width of each source	m
	• Injected power for each source	W
	• A and Z mass and charge number of the injected atom	-
	• Energy of the injected atom	eV
	• Particle fraction per energy component	-
OUTPUT VARIABLES	• Particle deposition profile	m^{-3}
	• Power deposition profile	W/m^3
	• Pitch-angle profile	-
	• Torque density profile	$N{\cdot}m^{-2}$
	• Particle shine-through	-
	• Power shine-through	W
	• 5D birth matrix$(injector, E, R, Z, \xi)$	-
	• Optional 5D birth matrix$(injector, E, X, Y, Z)$	-
	• Optional 5D birth matrix(E, X, Y, Z, R)	-
	• Associated X, Y, Z and R spatial vectors	m
	• Associated ξ pitch angle vector	-

Table 1: NEMO input and output variables.

Main routines	nemo.f90	Main program
	beamdp.f90	Full algorithm of beam deposition and attenuation
Modules	mod_io_management.f90	Functions for input/output management
	mod_variables.f90	Module containing input/output variables
Cross section calculation	janev_cross_section.f90	Cross section calculation using the Janev model
	adas_cross_section.f90	Cross section calculation using ADAS tables
	adas_read.f90	Interface between fortran and ADAS tables
	zeffective.f90	Calculation of the effective charge
Shinethrough	wall_limit.f90	Test if the beam hits the tokamak wall
Data reading and writing	read_geoplasma.f90	Read plasma geometry and equilibrium
	read_kinplasma.f90	Read plasma kinetics
	read_geoparbeam.f90	Read beam geometry and parameters
	read_results.f90	Read variables associated with output format
	write_results.f90	Write results
Interpolation routines	interp1d.f90	1D interpolation from a given profile
	interp2d.f90	2D interpolation from a (R, Z) grid
	yhfit2.f90	Profile interpolation on a different radial coordinate

Table 2: List of NEMO fortran routines.

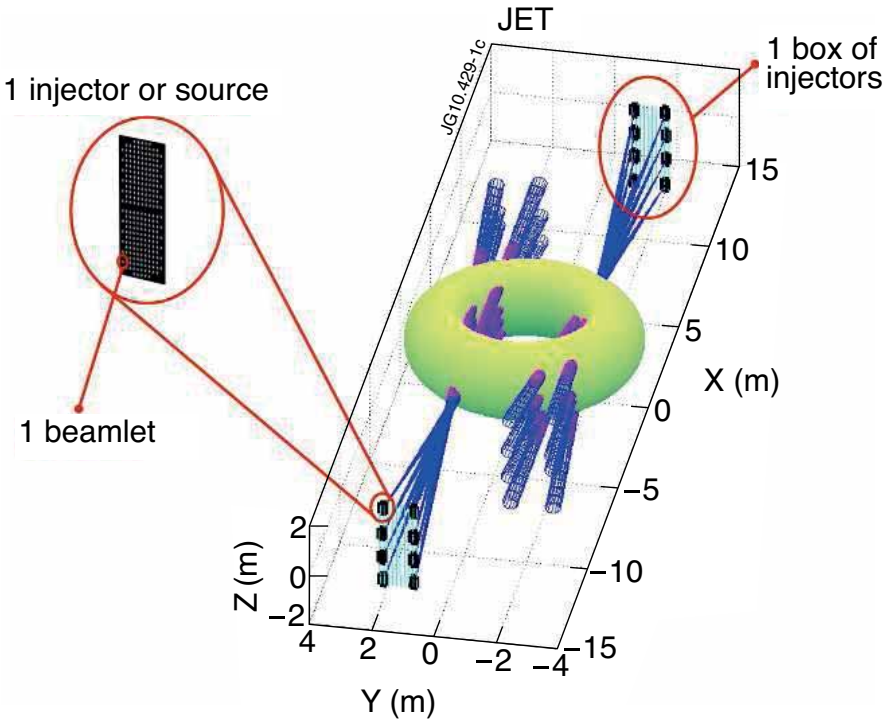

Figure 1: Geometry of JET NBI injectors in the 3D (X, Y, Z) space according to NEMO trajectory equations. Each injector is represented by a black rectangle (JET is equipped with two boxes of 8 injectors). The plasma is represented by the green contour. The divergence cones are the blue contours. Finally, the magenta region represents the points where the deposition is calculated, i.e. between $s = -(R_0 + a)$ and $(R_0 + a)$ where R_0 and a are respectively the plasma major and minor radii (the beam deposits only inside the plasma though) Here, $R_0 = 2.9$ m and $a = 0.9$ m.

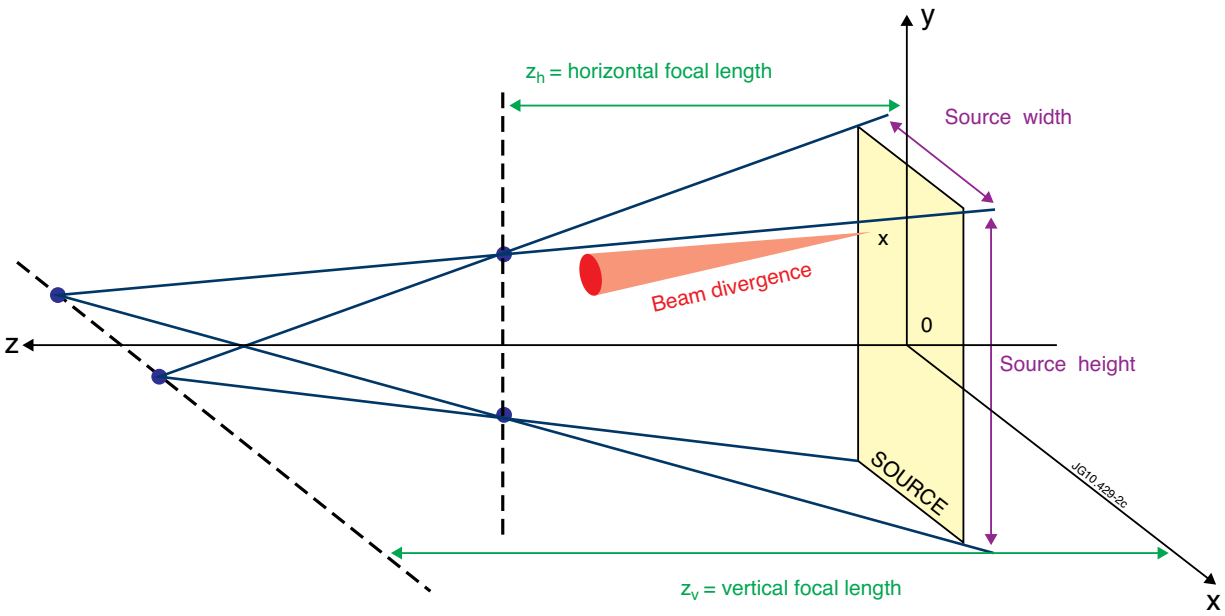

Figure 2: Full injector geometry.

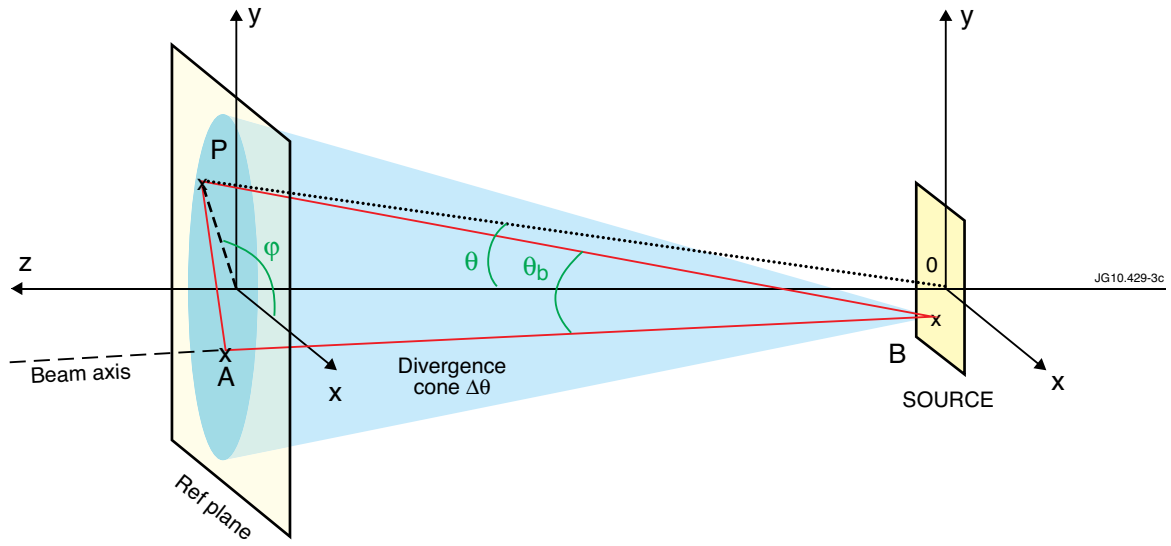

Figure 3: Simplified geometry of the injector after the narrow-beam approximation. The three points B, P and A are used for the intensity calculation.

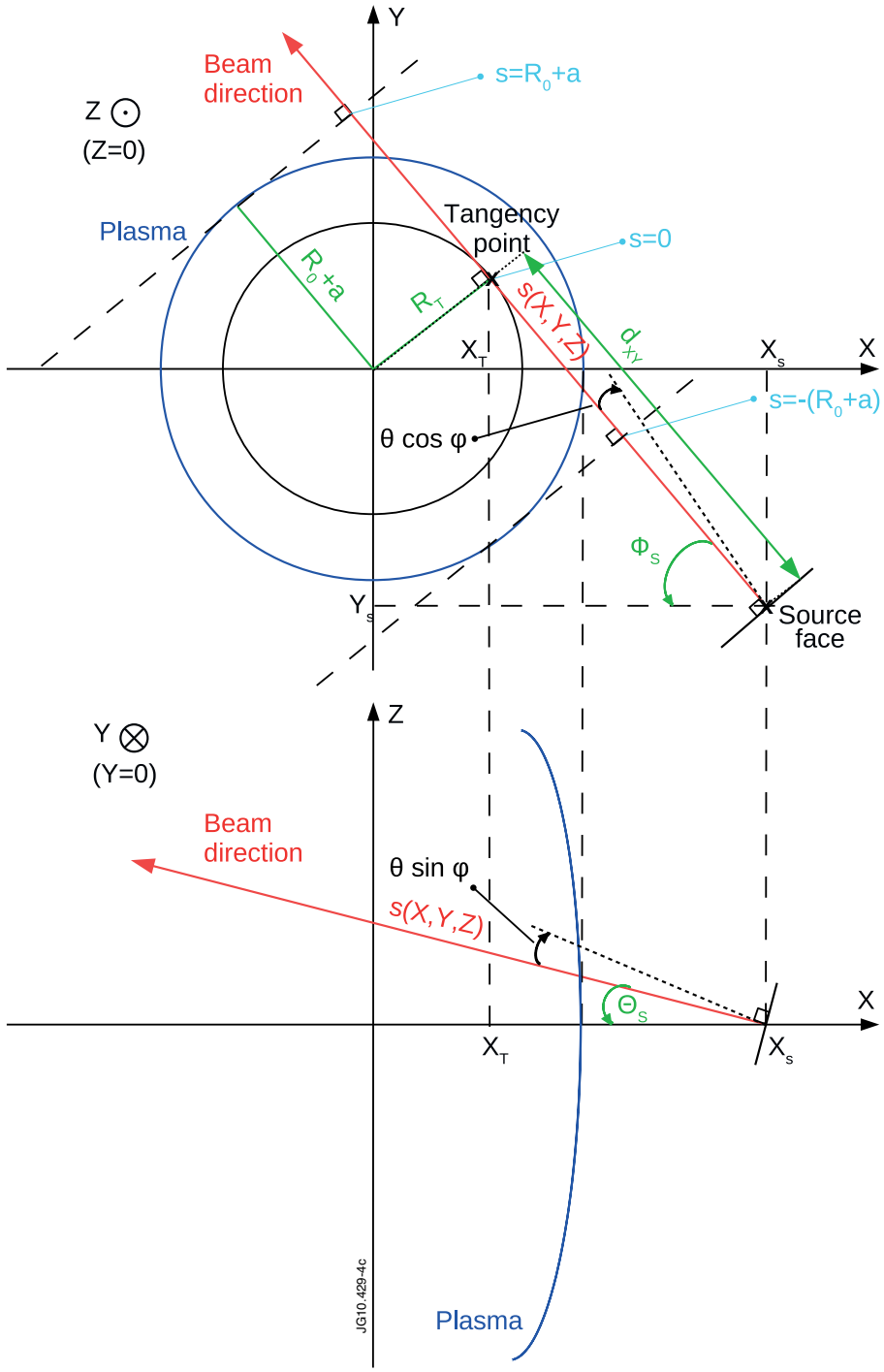

Figure 4: Birth coordinates of fast ions coming from NBI in the (X, Y) and (X, Z) planes. s(X, Y, Z) is the tri-dimensional coordinate along the neutral beam trajectory. R_T is the tangency radius; φ is the angle covering the divergence cone of the beam; θ is the angle defined around the beam direction; Φ_S and Θ_S are the angles between the projection in the (X, Y) and (X, Z) planes respectively, of the injection direction of the central point of the injector surface and the X-axis; d_{XY} is the projection in the (X, Y) plane of the distance between the central point of the injector surface and the tangency point.

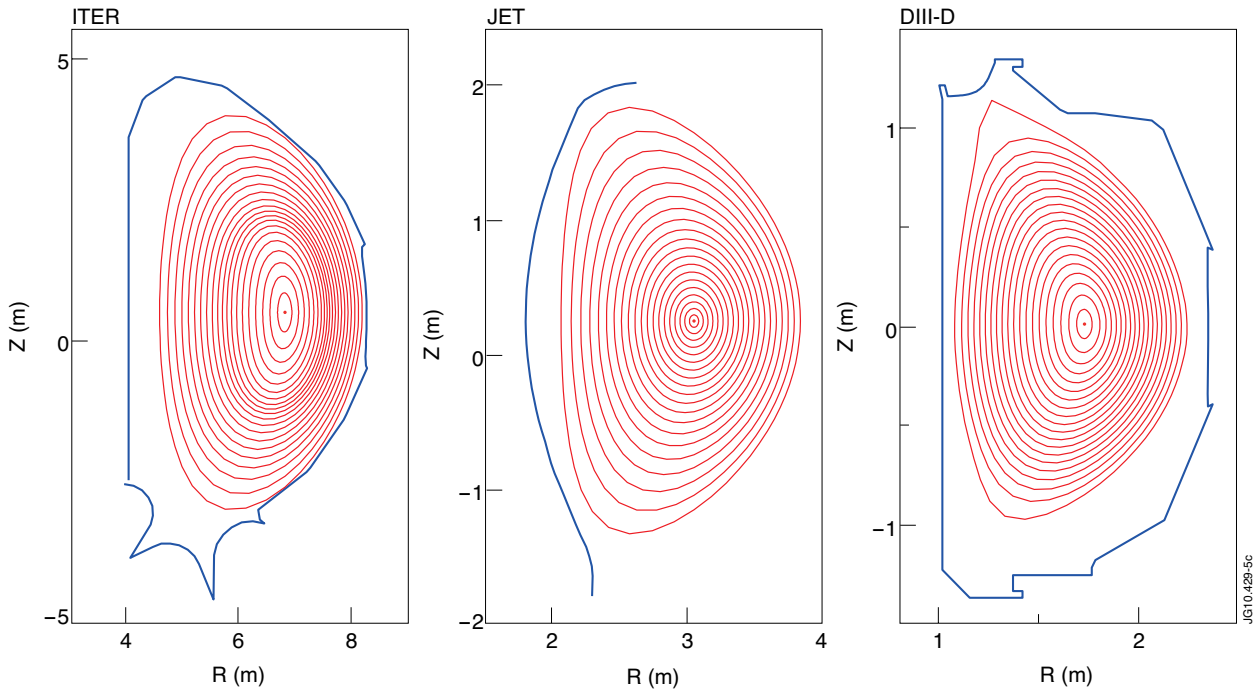

Figure 5: Wall contour (blue lines) for ITER, JET and DIII-D tokamaks, in the (R, Z) poloidal view. Red lines represent the plasma magnetic surfaces.

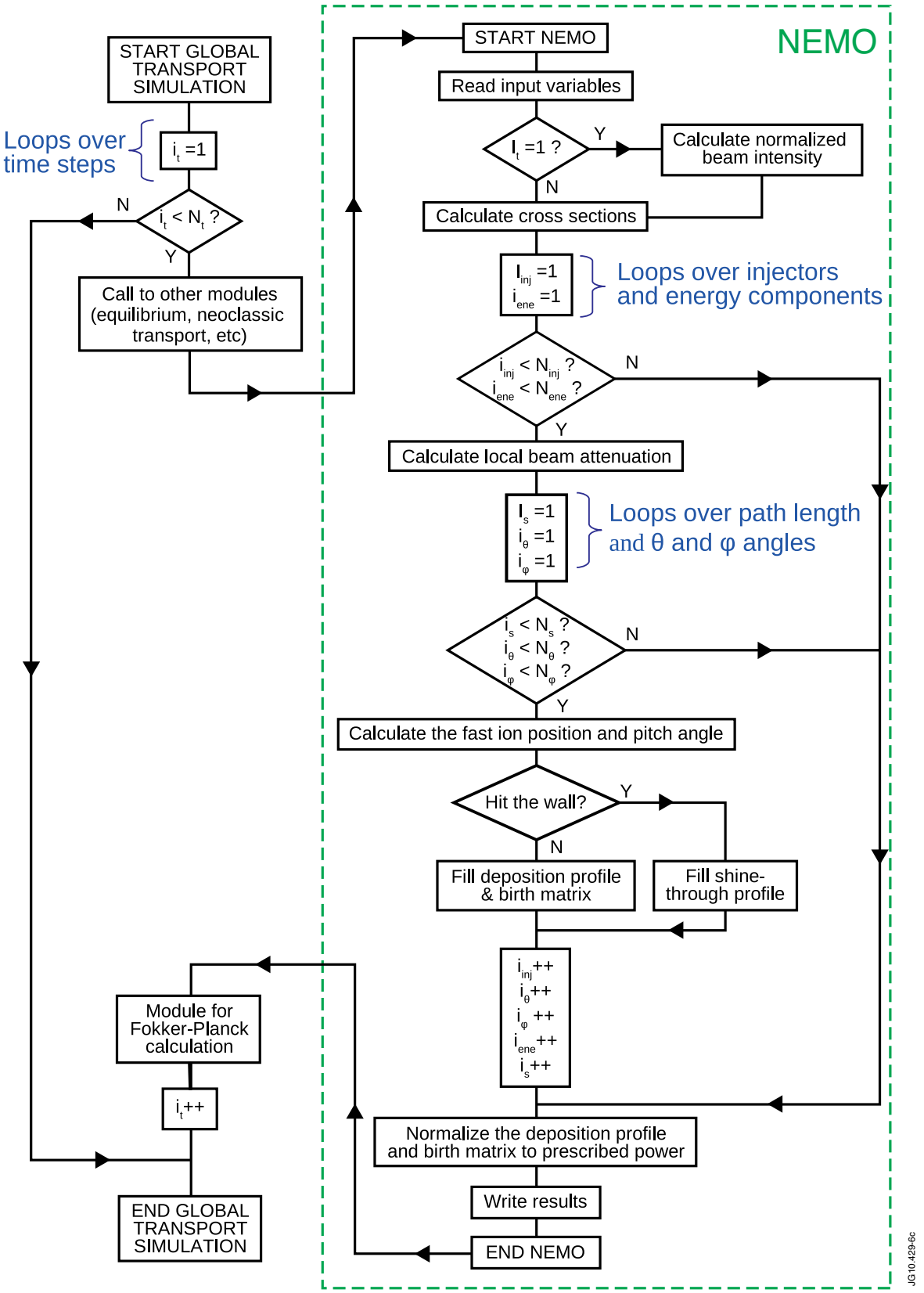

Figure 6: NEMO flow chart.

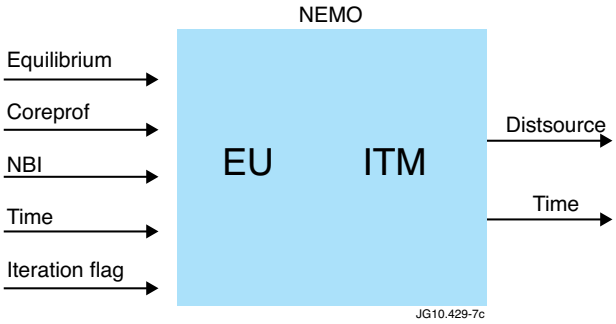

Figure 7: The NEMO Kepler actor.

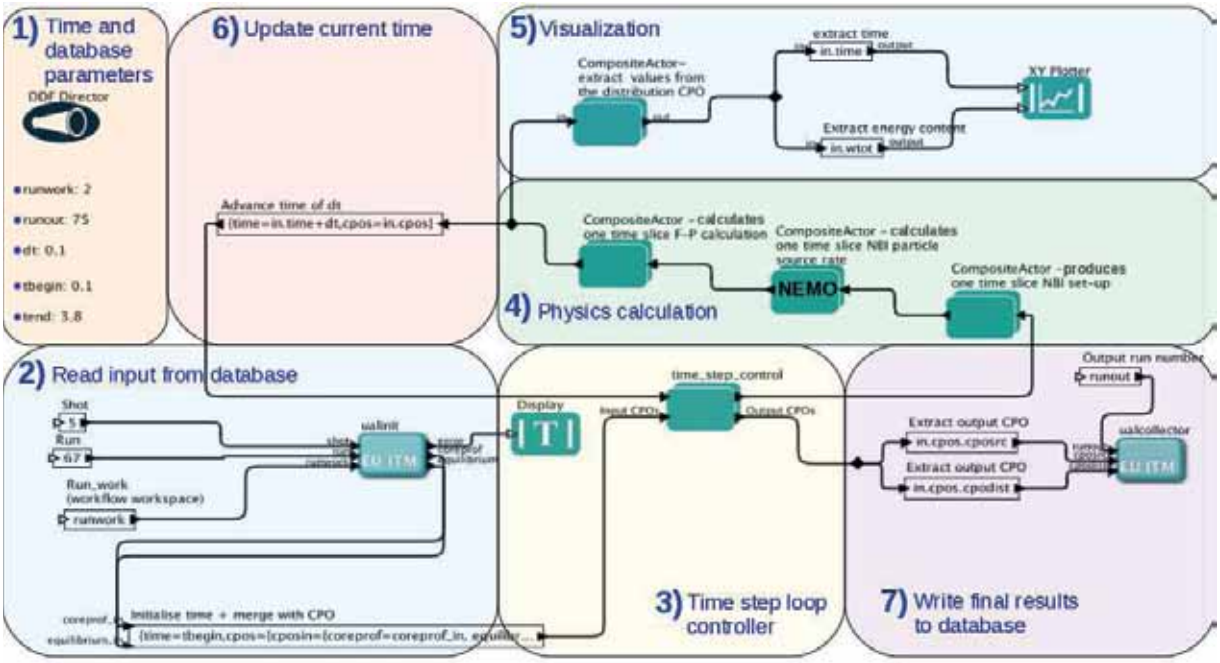

Figure 8: Workflow to run the NEMO code in the ITM-TF platform environment.

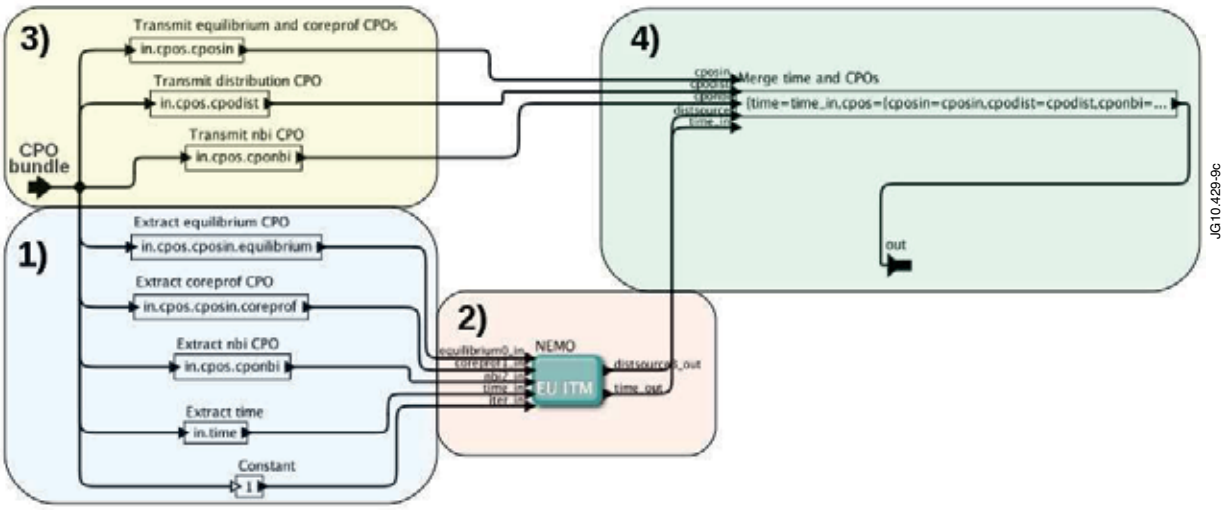

Figure 9: The NEMO composite actor.

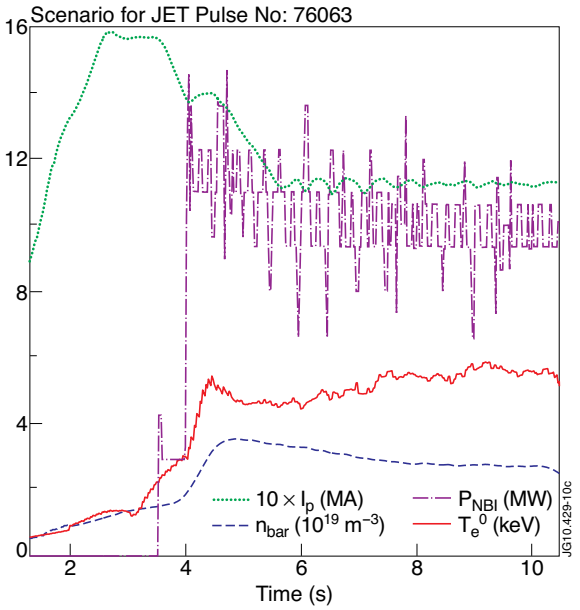

Figure 10: Time evolution of the plasma current I_P, the electron line-integrated density nbar, the applied NBI power P_{NBI} and the central electron temperature T_e^0, for Pulse No: 76063.

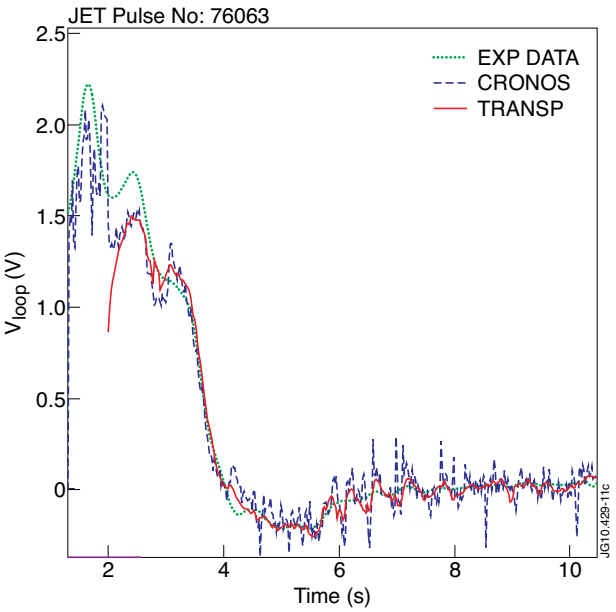

Figure 11: Time evolution of the measured and simulated loop voltage for Pulse No: 76063.

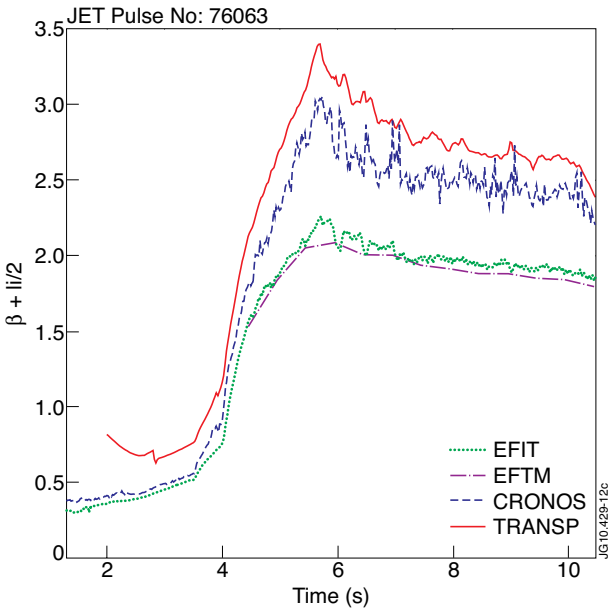

Figure 12: Time evolution of the measured and simulated $\beta + l_i/2$ for Pulse No: 76063.

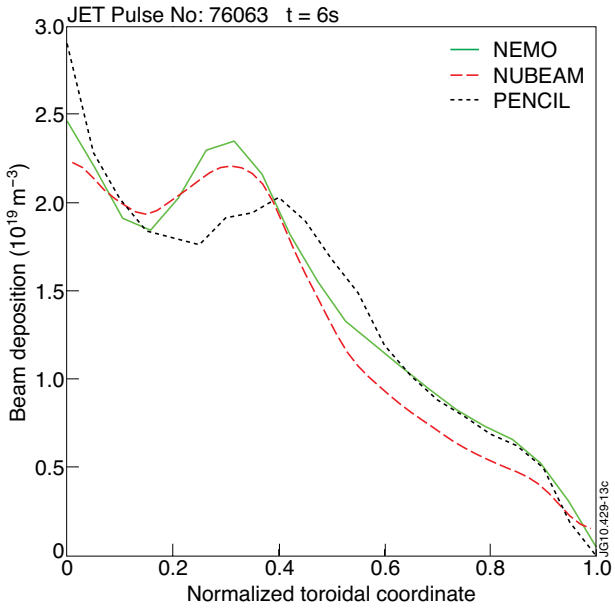

Figure 13: Beam deposition profile for Pulse No: 76063 at t=6 sec, as simulated by NEMO, NUBEAM and PENCIL.

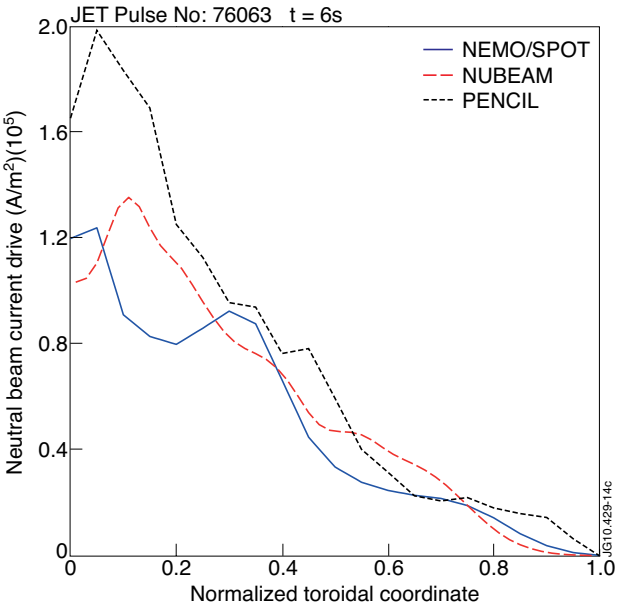

Figure 14: Neutral beam current drive profile for Pulse No: 76063 at t=6 sec, as simulated by NEMO/SPOT, NUBEAM and PENCIL.

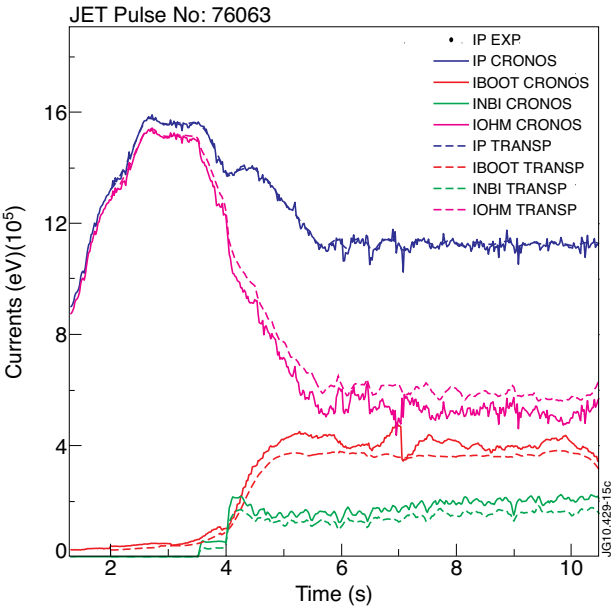

Figure 15: Time evolution of the plasma currents simulated by CRONOS and TRANSP, for Pulse No: 76063.

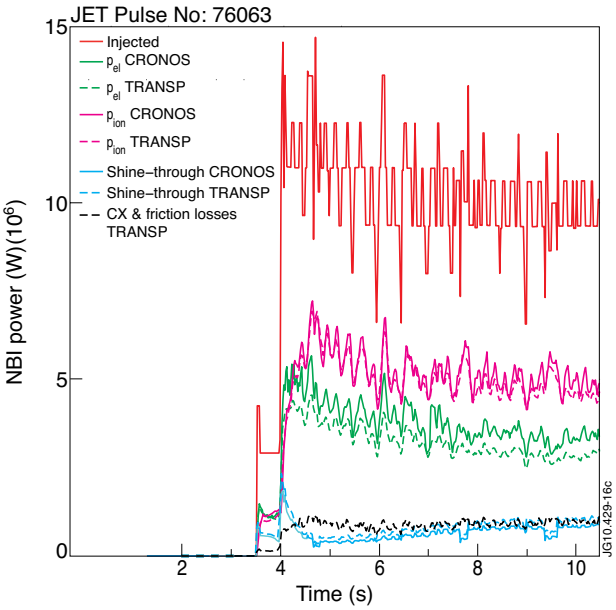

Figure 16: Time evolution of the NBI-related powers simulated by CRONOS and TRANSP, for Pulse No: 76063.

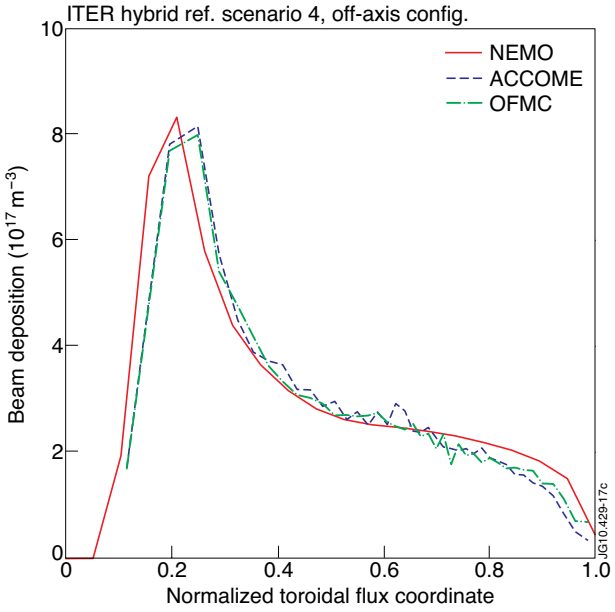

Figure 17: Fast ion deposition profile obtained using NEMO (solid red line), ACCOME (dashed blue line) and OFMC (dashed green line) for the ITER steady-state reference scenario 4 in the off-axis configuration.

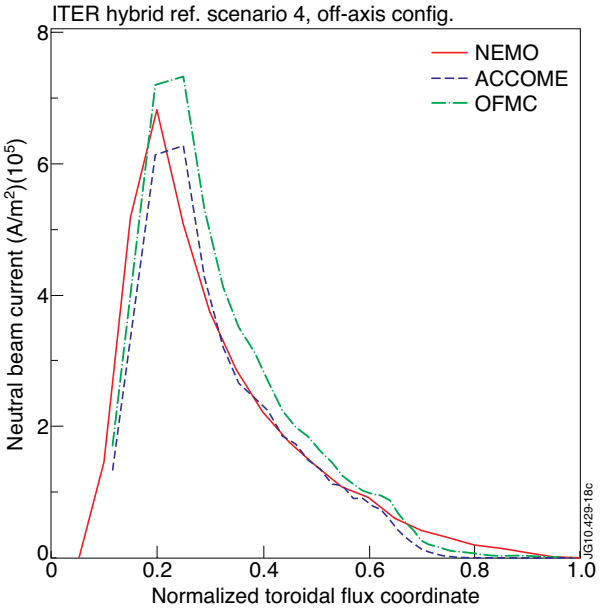

Figure 18: Neutral beam current profile obtained using NEMO (solid red line), ACCOME (dashed blue line) and OFMC (dashed green line) for the ITER steady- state reference scenario 4 in the off-axis configuration.