Chapter 2
ISDEP

2.1 Introduction

As we said in previous chapters, Integrator of Stochastic Differential Equations for Plasmas (ISDEP) is a code devoted to solve the dynamics of a minority population of particles in a complex 3D fusion device. ISDEP is becoming a rather complex code, with more than 10^4 lines. It is adapted to four different fusion device geometries (two stellarators and two tokamaks). In this Chapter we discuss the basic structure of the code and the tools used to analyze the data in Sect. 2.2 and *benchmark* the code in Sect. 2.3. With *benchmark* we mean the comparison of the ISDEP results with another similar code, in order to assure that ISDEP is free of programming errors. We end this Chapter with an overview of the previously published results in Sect. 2.4.

The main improvements of the code performed during the elaboration of this thesis are related with the measurements and analysis of the particle distribution function (Sects. 2.2.5 and 2.2.6) and its adaptation to three new fusion devices (in Sect. 2.3 for the *benchmark* and in Chaps. 3 and 4).

We start with a description of the code.

2.2 Description of the Code

ISDEP was created under the CIEMAT[1]-BIFI[2]-UCM[3] collaboration in 2007 and is in continuous development and improvement. From a physical point of view, ISDEP solves the Neoclassical (NC) transport avoiding several common approximations of the standard NC theory implemented in existing transport codes.

[1] Centro de Investigaciones Energéticas, Medio Ambientales y Tecnológicas, Madrid, Spain.

[2] Instituto de Biocomputación y Física de los Sistemas Complejos, Zaragoza, Spain.

[3] Universidad Complutense de Madrid, Madrid, Spain.

A. de Bustos Molina, *Kinetic Simulations of Ion Transport in Fusion Devices*,
Springer Theses, DOI: 10.1007/978-3-319-00422-8_2,

As an example, it makes use of the Cartesian coordinates instead of the Boozer coordinates [1], generally used in this code class. Boozer coordinates are specific coordinates for magnetically confined plasmas, but they do not allow the representation of magnetic islands, ergodic zones in the magnetic field or points in space outside the plasma boundary, where the field lines are open. Boozer and magnetic coordinates are only well defined for nested magnetic surfaces but not for those topologies. Therefore, Cartesian coordinates are better suited for our goal. Other common approximations that we can avoid with ISDEP are related to the typical radial width of the particle orbits in the device and the diffusive nature of the transport processes. The particle orbit width is usually assumed to be small compared with the typical distances of the problem, but in many real situations this is not actually the case. Finally, the kinetic energy of the studied particles does not need to be conserved in ISDEP, oppositely to the neoclassical approximation. This allows us the inclusion of strong electric fields and study their effects on ion dynamics. In addition, this code was designed to run on grid architectures, propelling the development of this computing platforms.

As we previously said, ISDEP considers a minority population of test particles, for which we may choose among several options. This minority population can be thermal particles, obtaining then specific information for the plasma bulk that is not given by the plasma equilibrium. The test particle can also be fast particles coming from heating systems, studying then their interaction with thermal particles. Furthermore, although we have not considered the case yet, ISDEP has the potential to handle impurity dynamics.

There exist many computer codes devoted to solve the Neoclassical transport. For example, the codes DKES, NEO-MC and MOCA study similar physics, but with some peculiarities and different approximations. Some of these neoclassical codes have been *benchmarked* and compared in Ref. [2].

- Drift Kinetic Equation Solver (DKES) [3] is a well established code that solves the linearized Drift Kinetic Equation using a functional minimization method. It takes the effective radius and the particle energy as input parameters and then solves the transport equations in the remaining three dimensions.
  DKES solves a FP type equation and computes the whole transport matrix [2] using Boozer coordinates, calculating also the *Bootstrap* current and the parallel conductivity. Unfortunately it presents some drawbacks that ISDEP avoids. It assumes diffusive nature in the transport, infinite fast parallel transport, conservation of kinetic energy and narrow radial excursions of the particle. Moreover, it neglects the poloidal component of the ∇B drift and approximates

$$\frac{\mathbf{E}\times\mathbf{B}}{B^2} \sim \frac{\mathbf{E}\times\mathbf{B}}{\langle B^2\rangle}. \tag{2.1}$$

  The use of Boozer coordinates means that the code can be used only when one has nested magnetic surfaces and cannot be used in the scrape-off-layer.

Computationally, it scales unfavorably with the number of magnetic field Fourier modes, but it is very fast in high collisionality regimes. This code gives large errorbars for the transport coefficients in the long mean free path regime for complex magnetic configurations. Indeed it is not adequate to study complex 3D devices in a low collisionality regime.

- **NEO-MC** [4] solves the same equations as DKES but using a Monte Carlo method instead. NEO-MC has been designed to specifically calculate the *Bootstrap* current in 3D fusion devices. The main advantage of NEO-MC is that it reduces strongly the errorbars of the transport coefficients for any collisional regime.
  In order to improve the accuracy of the code to estimate the *Bootstrap* current, the effect of trapped and barely trapped particles is considered specifically. The velocities of two particles that are moving in opposite directions are subtracted, thus creating a quasi-particle and the number of test particles is increased in the barely trapping regions. Most of the computing time is devoted to follow particles. This improves the scalability, although this code is more expensive in computing resources than DKES. Like DKES, NEO-MC is subjected to the neoclassical ordering. Thus, NEO-MC cannot avoid such approximations that are not present in ISDEP. On the other hand, NEO-MC calculates also for the electrons and, hence, allows one to estimate the self-consistent radial electric field from the ambipolar condition. NEO-MC, like DKES, also assumes nested magnetic surfaces.
- **MOCA** [5] is another Monte Carlo code developed at CIEMAT ten years ago. MOCA is an evolution of the MCT code [6] and it calculates the radial diffusion coefficients (diagonal part of the transport matrix), using Boozer coordinates for the spatial position and Coulomb collisions for the interaction between particles. It usually scales better than DKES, but it does not allow for the bootstrap current calculation in its first version. Opposite to ISDEP, MOCA works in Boozer coordinates and is subjected to the neoclassical ordering, but calculates for both ions and electrons.
- **MOHR** [7] is another guiding center orbit code that solves the Fokker-Planck equation for ions. MOHR is a very similar code to ISDEP indeed. The main differences rely on the statistical error calculation.

Once we have the mathematical model of the particle dynamics from Chap. 1, we describe the Monte Carlo method used in ISDEP, the architecture of the code and the statistical techniques needed to obtain global results from a set of independent trajectories. Then we present an overview of previous ISDEP results and, finally, the ISDEP code is *benchmarked* with MORH in Sect. 2.3.

2.2.1 The Monte Carlo Method

The basis of the Monte Carlo method used in ISDEP relies on the equivalence between the Fokker-Planck and Langevin equations [8]. The Fokker-Planck equation is a linear partial differential equation for a distribution function of a minority population of

particles (called test particles) that interact with a static background. The Langevin approach is equivalent to this description, but providing Stochastic Differential Equations (SDE) [8] for a single test particle motion (see Sect. 1.4). Integrating many test particle trajectories and analyzing the results is mathematically equivalent to obtain the solution to the original Fokker-Planck equation.

ISDEP integrates the trajectories taking into account collisions with ions and electrons from the background, the electrostatic potential and the confining magnetic field. The statistical analysis of many test particles allows the measurements of different plasma parameters, like average energy, confinement time or even the marginal distribution function of the test particle population.

In order to reduce computational requirements, the Guiding Center (GC) approximation, described in Sect. 1.3, is used in the code. The GC coordinates chosen are (x, y, z, v^2, λ), where (x, y, z) are the guiding center space coordinates, v^2 is the normalized particle kinetic energy and

$$\lambda = \mathbf{v} \cdot \mathbf{B}/(B\, v) \tag{2.2}$$

is the pitch. In the Fokker-Planck description, the time evolution of the distribution function $f(x,t)$ is given by the convective $(F^i(x,t))$ and the diffusive transport $(G^i_j(x,t))$ in the 5D phase space:

$$\frac{\partial f(x,t)}{\partial t} = \frac{\partial}{\partial x^i}\left(-F^i(x,t) + \frac{1}{2}\frac{\partial}{\partial x^j} G^i_k(x,t) G^{kj}(x,t)\right) f(x,t)\,. \tag{2.3}$$

The equivalent set of Stochastic Differential Equations (SDE) in Itô's sense [8] (i.e. Langevin equations) is:

$$\mathrm{d}x^i = F^i(x,t)\ \mathrm{d}t + G^i_j(x,t)\ \mathrm{d}W^j. \tag{2.4}$$

The explicit form of F_i and G_{ij} has been discussed in Eqs. (1.34), (1.35) and (1.36). Now the coordinates in phase space x^i refer to the movement of a single particle, whose trajectory is determined by the background via F^i and G^i_j. The Wiener process, $\mathrm{d}W^j(t)$ (see Sect. 1.4) represents the random part of the interaction with the plasma.

In the case of interest, the problem consists of a SDE system of five equations with two Wiener processes. The SDEs can be transformed to the Stratonovich convention because it is more suitable for several numerical methods. In Sect. 1.4 the reader can find a short review of probability theory and stochastic calculus, which provide the necessary tools for the calculations of this thesis.

Once N trajectories are integrated and stored, we can reconstruct the distribution function accumulating the particle path in phase space:

$$f(\mathbf{x}, t) \propto \frac{1}{N}\sum_i^N \delta(\mathbf{x} - \mathbf{x}(t)). \tag{2.5}$$

Since ISDEP calculates f according to the time that the particles spend in a given point of the phase space, the Jacobian of the coordinates $\mathbf{x}$ is included in $f(\mathbf{x}, t)$. In addition, due to the linear nature of ISDEP, $f(\mathbf{x}, t)$ is not exactly a distribution function because all its results have implicit a normalization constant. This means that ISDEP can calculate the intensive properties of the test particles (average energy, average lifetime, etc), but needs some extra information to compute the extensive properties (total energy contained, total electric current, etc).

With the proper normalization, $f(\mathbf{x}, t)$ can be taken as a probability density of the test particle ensemble in phase space.

2.2.2 ISDEP Architecture

ISDEP is programmed in C to maximize its performance and portability and was designed to scale perfectly in distributed computing platforms such as grid or volunteer computing architectures. It does not require external libraries other than the standard C libraries. Consequently, the scaling in massive parallel computers is almost linear. The operation of the code is briefly summarized in the following steps:

Initialization

After compiling, a copy of the executable and the input files are sent to each computing node, or copied into a file-system common to all nodes. The input files contain the plasma background data, the confining magnetic field and trajectory details (time step, numerical algorithm chosen, etc). The first stages of the execution of the code are invested in initializing the random number generator, the magnetic field array and the trajectory itself. The magnetic field array contains all the information related to the magnetic configuration of the device. Part of this array is read from a file (e.g., $\mathbf{B}$) and the remaining is calculated (e.g., ∇B) in order to save CPU time in the next steps. The interpolations in this array are linear, provided that the spatial grid is dense enough. The typical distance between two nodes in the magnetic grid is $<1\,\%$ of the size of the device so the magnetic field is smooth enough. The size of the magnetic array may be ~ 400 MB, representing most of the memory that ISDEP uses.

The trajectories are initialized according to a given distribution. If one deals with bulk ions, the spatial distribution is given by the plasma density. In velocity space the distribution is locally Gaussian in v^2 and uniform in λ. Alternatively, when dealing with suprathermal ions, ISDEP can read the output of a neutral beam injection code, like FAFNER2 [9], to calculate the initial test particle distribution (see Chap. 4).

Orbit Iintegration

After the initialization routines, every node starts to integrate trajectories independently of each other. The statistical independence is guaranteed reading the random

seed locally in the node. Then the orbits are integrated and the data written in a file. A description of the numerical methods used in ISDEP can be found in Sect. 1.4.4. This is the most CPU time consuming stage.

There are two main output files in ISDEP: trajectory files (OUT.DAT) and histogram files (OUT.HIS). In the former the 5D position in phase space is stored for each trajectory at selected times. Since we are interested in the plasma evolution time scales, the measurement times are chosen to be approximately equidistant in logarithmic scale. The latter contains histograms of different particle quantities (energy, distribution function, rotation velocity, radial flux, ...). In order to increase statistics the following technique is applied in the histograms: assuming that the evolution of the system is slow, one may take all the measurements at times $t \in (0.9\,t_0,\ 1.1\,t_0)$ belonging to t_0. In this way the statistical errors are significantly reduced.

Analysis

The output of each node is stored in a particular node and is analyzed with the ISDEP analysis tools. Many physical quantities are calculated in this stage, like average energy, velocity profiles, steady state distribution function and escape points. ISDEP uses the jack-knife method for all statistical error estimation [10], described in Sect. 2.2.3. Some output analysis, related to Sects. 2.2.5 and 2.2.6 is done using the Python programming language.

Table 2.1 summarizes the profiles measured with ISDEP, as functions of the effective radius and time. In addition, ISDEP calculates the global average of all these magnitudes as a function of time. Finally, the distribution function of the test particles is obtained, but averaging in the magnetic surfaces: $f(t, \rho, v_{||}, v_{\perp})$.

The particle escape distribution is presented as a list of points in phase space: $(t_i, x_i, y_i, z_i, v_i^2, \lambda_i)$, being t_i the escape time of the ith particle. A lot of information can be extracted from this list with little effort. For example, accumulation of losses in a region of the device can produce severe damage to the device, and ISDEP can help to prevent this effect.

It is essential to mention that ISDEP requires some feedback to determine the time discretization parameter Δt. The usual procedure to determine Δt requires at least two simulations with ISDEP. First one must decide what statistical accuracy in the output is needed, usually around 5 %. This errorbars can be diminished knowing that they scale with $N^{-1/2}$, being N the total number of trajectories integrated. Then, starting with some reasonable value of Δt, ISDEP is run for $\Delta t/2,\ \Delta t/5,\ \Delta t/10\ \ldots$ until the results are the same within the statistical errorbars. This procedure must be done for each simulation to ensure that the statistical errors are always larger than the discretization errors.

Figure 2.1 shows the workflow of ISDEP in a distributed computing architecture like the grid.

Table 2.1 ρ-dependent profiles calculated with ISDEP. Each one is presented as a function of time

1D profile observable	meaning
ρ	Average effective radius
$(\rho - \rho_0)^2$	Deviation from the initial position
θ	Poloidal angle
E	Total energy [units of $mc^2/2$]
v^2	Normalized kinetic energy
κ_v	Binder cumulant of v: $\kappa_v = \langle v^4 \rangle / \langle v^2 \rangle^2$
λ	Pitch angle
v_b	Parallel velocity, in units of c
v_b^2	Parallel kinetic energy
κ_{v_b}	Binder cumulant of v_b
v_φ	Toroidal velocity
v_φ^2	Normalized toroidal kinetic energy
κ_{v_φ}	Binder cumulant of v_φ
v_θ	Poloidal velocity
v_θ^2	Normalized poloidal kinetic energy
κ_{v_θ}	Binder cumulant of v_b
v_r	Radial velocity
v_r^2	Normalized radial kinetic energy
κ_{v_r}	Binder cumulant of v_r
Γ	Radial particle flux
Q	Radial energy flux
z	Average z coordinate

2.2.3 Output Analysis: Jack-Knife Method

ISDEP incorporates the Jack-Knife method [10] for output analysis. This method is a robust and simple algorithm for the statistical error calculation.

Let us consider a set of N independent, identically distributed vector random variables, $\mathbf{X}_i, i = 1, \ldots, N$, and a nonlinear function f of the expectation values $\langle \mathbf{X} \rangle$. By a vector random variable, we intend a set of M physical quantities that are measured on the same experiment (or numerical simulation) $\mathbf{X}_i = (X_i^{(1)}, X_i^{(2)}, \ldots, X_i^{(M)})$. Of course, the components of $\mathbf{X}_i$ can be statistically correlated, but they are independent in the subscript i.

The problem that the Jack-Knife method solves is that of computing the statistical error for our estimator of $f(\langle \mathbf{X} \rangle)$. The procedure takes care at once of two problems: (i) it treats correctly the statistical correlations among the components of $\mathbf{X}_i$ and (ii) it avoids the instabilities caused by the non-linear nature of the function f.

As an example, we may think of $X_i^{(m)}$ as the energy of the ith particle at time t_m. Even thought there will be no correlation between particles, obviously the energy of a given particle is correlated in time with itself. Thus, the calculation of the time

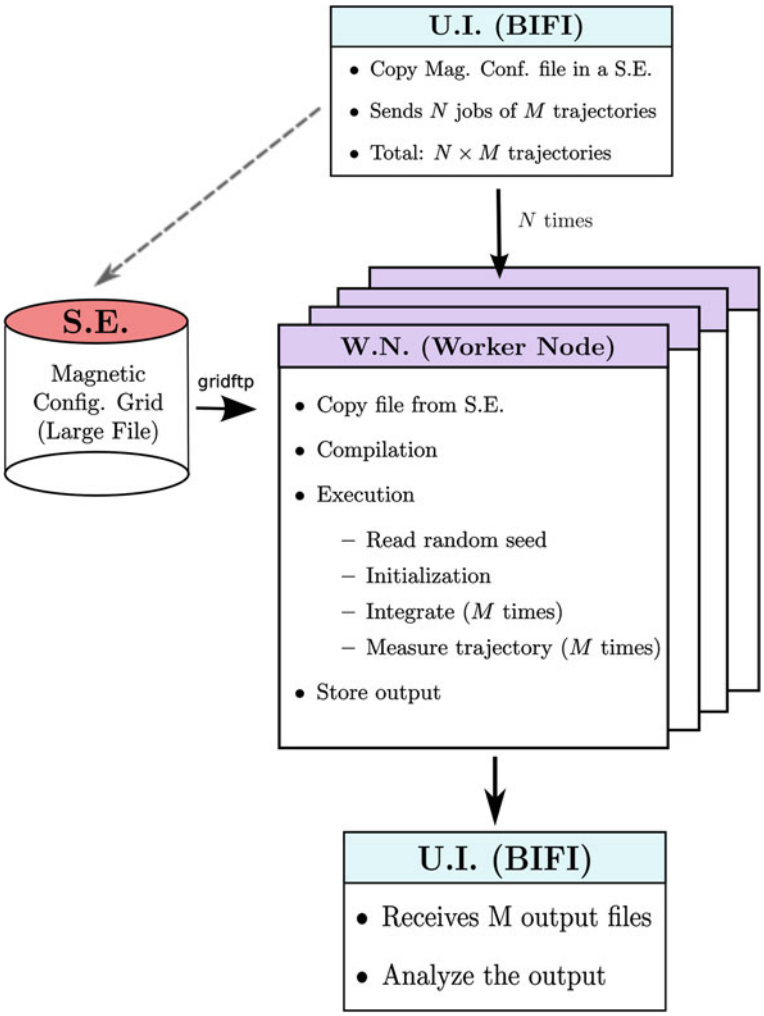

Fig. 2.1 ISDEP workflow on the grid. First, the magnetic field file is copied to a Storage Element (SE) and the jobs submitted to the Worker Nodes (WN). The WN retrieve copies of the magnetic field file from the SE, integrate a certain number of trajectories specified by the user and compress the result. When finished, all output files are copied back to the User Interface (UI) and then locally analyzed

differences in the kinetic energy requires a method that includes correlations between measurements.

The Jack-Knife procedure is as follows:

We first compute the average of the random variables as

$$\overline{\mathbf{X}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{X}_i, \tag{2.6}$$

and construct the estimator for $f(\langle\mathbf{X}\rangle)$

$$\overline{f} = f\left(\overline{\mathbf{X}}\right). \tag{2.7}$$

A direct computation of the statistical error using the random variables $f_i = f(\mathbf{X}_i)$ would be impractical (unless f is nearly a linear function). On the other hand, an error propagation computation requires to take into account the statistical correlations of the different components of $\mathbf{X}_i$. A simple alternative procedure consist in the following. First define the (non independent) random variables

$$\mathbf{X}_i^{\mathrm{JK}} = \frac{1}{N-1}\sum_{j=1;\, j\neq i}^{N}\mathbf{X}_j, \tag{2.8}$$

and

$$f_i^{\mathrm{JK}} = f(\mathbf{X}_i^{\mathrm{JK}}), \tag{2.9}$$

then compute the Jack-Knife estimate for the statistical error of $\overline{f}$

$$\Delta_{\overline{f}} = \sqrt{(N-1)\left[\sum_{i=1}^{N}\frac{(f_i^{\mathrm{JK}})^2}{N} - \left(\sum_{i=1}^{N}\frac{f_i^{\mathrm{JK}}}{N}\right)^2\right]}. \tag{2.10}$$

Note that the error is proportional to the square root of the number of blocks, rather than to $\frac{1}{\sqrt{N}}$. The number of blocks should be large enough, say 50, so this technique works properly. It is straightforward to show that the Jack-Knife method gives the same results as Eq. (2.11) for linear functions.

$$\overline{f} = \frac{1}{N}\sum_{1}^{N} f_i, \qquad \Delta_{\overline{f}\ \mathrm{Linear-only}} = \sqrt{\frac{\overline{f^2} - \overline{f}^2}{N-1}}. \tag{2.11}$$

2.2.4 Computing Platforms

Since the communication between nodes is zero, ISDEP is able to run in several computing platforms: high performance computing (HPC) and distributed platforms. The scaling with the number of nodes is, in all cases, almost linear as we mentioned above.

In distributed platforms there is no fast communication between the nodes. On the contrary, a huge number of nodes are available. These nodes are very inhomogeneous

in performance and characteristics, so ISDEP should be as stable as possible to minimize the problems caused by this fact. Grid and volunteer computing are the main resources used. Grid is provided by the Fusion Virtual Organization (EGEE[4] [11] and EGI-InSPIRE[5] projects). The volunteer computing projects Ibercivis [12] and its precursor Zivis [13] have provided hundreds of thousands of CPU hours. Moreover, they had an important role in the divulgation of fusion science in Spain and Portugal. ISDEP was designed from the early steps to run on Grid architectures, but it had to be adapted to volunteer computing.

High Performance Computing (HPC) consists of a set of nodes (cluster) located in the same facility, characterized by fast communication between nodes. In this work, HPC time is provided by the EULER cluster at CIEMAT. EULER is formed by 1,152 Xeon cores (13.8 Tflops), connected with Infiniband.

2.2.5 Steady State Calculations

The steady state of a system is a time-invariant state in which the particle and heat sources and sinks are in equilibrium with each other. The sinks in ISDEP are caused by the lost particles that escape from the plasma and hit the vacuum vessel. In ISDEP we calculate the steady state of the test particle distribution, using the Green function's formalism, following Ref. [14]. Let $f(x,t)$ be the distribution function of our system, t the time, x the coordinates in phase space, $\mathcal{L}$ a differential operator over f and $S(x,t)$ the source term. With this notation, the problem is expressed as:

$$\mathcal{L}\left(f(x,t)\right) = S(x,t). \tag{2.12}$$

In the case of interest $f(x,t)$ is the minority particle distribution function, $\mathcal{L}$ is the Fokker Planck operator for the guiding center and Boozer-Kuo-Petravic collision operator and the source is the continuous injection of particles into the plasma, computed with other MC codes. The Green function $G(x,t;x_0)$ is defined such that

$$\mathcal{L}\left(G(x,t;x_0)\right) = \delta(x-x_0)\,\delta(t), \tag{2.13}$$

with x_0 playing the role of initial position. Then:

$$f(x,t) = \int \mathrm{d}t_0\,\mathrm{d}x_0\,G(x,t-t_0;x_0)\,S(x_0,t_0), \tag{2.14}$$

because

$$\mathcal{L}\left(f(x,t)\right) = \int \mathrm{d}t_0\,\mathrm{d}x_0\,\mathcal{L}\left(G(x,t-t_0;x_0)\right)\,S(x_0,t_0) = S(x,t). \tag{2.15}$$

[4] Project number EGEE-III INFSO-RI-222667, http://public.eu-egee.org/.

[5] Project number EGI-InSPIRE RI-261323, www.egi.eu.

Note that the only contribution to this integral comes when $t = t_0$. In the systems studied here the source is assumed to be constant in time. This is in agreement with the linear description of the problem because the background plasma is kept constant. Thus, this technique should not be used in combination with the inclusion of nonlinear terms (see Sect. 2.2.7) neither for time varying plasmas. Then $S(x, t) = S(x)$ and:

$$f(x,t) = \int \mathrm{d}t_0 \, \mathrm{d}x_0 \, G(x, t - t_0; x_0) \, S(x_0) \tag{2.16}$$

$$= \int \mathrm{d}t_0 \int \mathrm{d}x_0 \, G(x, t - t_0; x_0) \, S(x_0). \tag{2.17}$$

Defining

$$H(x, t - t_0) = \int \mathrm{d}x_0 \, G(x, t - t_0; x_0) \, S(x_0), \tag{2.18}$$

the distribution function becomes a time integral:

$$f(x,t) = \int_0^t \mathrm{d}t_0 \, H(x, t - t_0) = \int_0^t \mathrm{d}t_0 \, H(x, t_0). \tag{2.19}$$

Except for a multiplicative constant, the function $H(x, t)$ is calculated by ISDEP after integrating $10^5 - 10^6$ test particle trajectories and analyzing the results. Furthermore, $H(x, t)$ is the solution to Eq. 2.15 using $S(x, t) = S(x)\,\delta(t)$ as a source term. Finally, with a 1D numerical integration, $f(x, t)$ can be easily found. In fact, for sufficient large times, it is expected that $f(x, t)$ is constant in time, becoming $f(x)$, because of the balance between continuous injection and particle losses (the number of the test particles always goes to zero if the source is a delta in time). Using the Jack-Knife method [10], one can estimate the average and statistical error of any plasma magnitude.

Due to its linear nature, ISDEP cannot provide absolute values of f, so the results are usually presented normalized. Nevertheless, real values can be calculated multiplying f times the incoming flux of particles.

2.2.6 NBI-Blip Calculations

NBI-Blip experiments are plasma discharges in which the NBI heating system is switched on for a small period of time in the discharge duration [15]. This injector pulse, with length $t_B > 0$, is represented mathematically with the Heaviside function in the source term:

$$S(x, t) = S(x)\,(\Theta(t) - \Theta(t - t_B)). \tag{2.20}$$

Then, using the formalism introduced in the previous section:

$$f(x,t) = \int \mathrm{d}t_0 \int \mathrm{d}x_0 \, G(x, t - t_0, x_0) S(x)\,(\Theta(t) - \Theta(t - t_B)) = f_1(x,t) + f_2(x,t). \tag{2.21}$$

The first term in Eq. 2.21 is:

$$f_1(x,t) = \int_{-\infty}^{t} \mathrm{d}t_0 \int \mathrm{d}x_0\, G(x, t-t_0, x_0)\, S(x)\Theta(t) \tag{2.22}$$

$$= \int_{0}^{t} \mathrm{d}t_0 \int \mathrm{d}x_0\, G(x, t-t_0, x_0)\, S(x) \tag{2.23}$$

$$= \int_{0}^{t} \mathrm{d}t_0\, H(x, t-t_0). \tag{2.24}$$

This is the usual procedure to calculate the steady state of $f_1(x,t)$. When t is large, f_1 becomes independent of t. The second term is then:

$$f_2(x,t) = -\int_{-\infty}^{t} \mathrm{d}t_0 \int \mathrm{d}x_0\, G(x, t-t_0, x_0)\, S(x)\, \Theta(t_0 - t_B) \tag{2.25}$$

$$= -\int_{t_B}^{t} \mathrm{d}t_0 \int \mathrm{d}x_0\, S(x, t-t_0, x_0)\, S(x) \tag{2.26}$$

$$= -\int_{t_B}^{t} \mathrm{d}t_0\, H(x, t-t_0). \tag{2.27}$$

Adding both expressions together:

$$f(x,t) = f_1(x,t) + f_2(x,t) \tag{2.28}$$

$$= \int_{0}^{t} \mathrm{d}t_0\, H(x, t-t_0) - \int_{t_B}^{t} \mathrm{d}t_0\, H(x, t-t_0) \tag{2.29}$$

$$= \int_{0}^{t_B} \mathrm{d}t_0\, H(x, t-t_0). \tag{2.30}$$

Notice that it is implicit in the equations that $t_0 < t$. Keeping this in mind, two extreme cases are:

- $t_B = 0 \Rightarrow f(x,t) = 0$.
- $t_B = t \Rightarrow f(x,t) \to f_1(x,t)$, the very same one from previous section. When t_B is very large, then $f(x,t) = f(x)$.

2.2.7 Introduction of Non Linear Terms

The physical description of the plasma implemented in ISDEP is a linear theory so, in principle, the background plasma is not modified. But here we propose a method to modify the background profiles, i.e. including some non-linearities. As of yet, only modifications in the background temperature are considered through

an iterative process. The developed procedure can be applied to the density or any quantity estimated as a moment of the distribution function.

For the test particles ensemble, the temperature profile is taken to be the average kinetic energy in an interval of $\Delta\rho = 0.1$ centered in ρ at a time t: $v^2(\rho, t)$. Let q_i be the quotient of the average kinetic energy in the ith iteration (v_i^2) and the original energy profile (v_0^2):

$$q_i(\rho, t) = \frac{v_i^2(\rho, t)}{v_0^2(\rho)}. \tag{2.31}$$

Then, in the iteration $i + 1$ we take as temperature the initial profile, multiplied by q_i:

$$T_{i+1}(\rho, t) = T_0(\rho, t)\, q_i(\rho, t). \tag{2.32}$$

Since the coordinates (ρ, t) are discretized, a linear interpolation is done to obtain $T(\rho, t)$ at arbitrary position and time. We stop iterating when $T_{i+1}(\rho, t) = T_i(\rho, t)$ within error bars, which is the final self-consistent profile. This method has been used in [16] and will be shown in Sect. 2.4. An example of this procedure can be found in Fig. 2.2, where the test particle energy profile is plotted for a simple tokamak with ICRH heating assuming a Gaussian power deposition profile. In the figure, the energy profile increases due to the external energy input and converges to a stable value after 6 iterations.

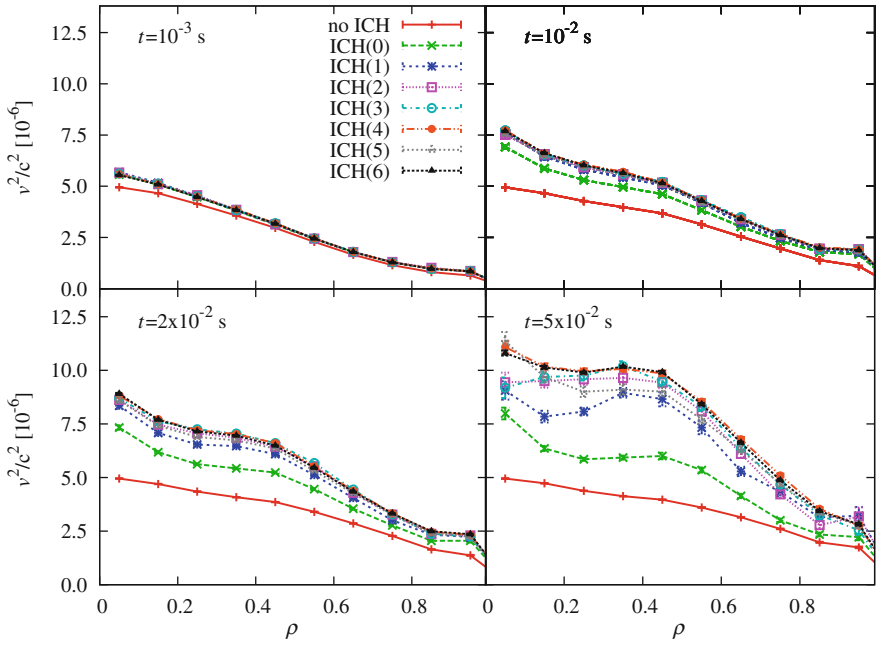

Fig. 2.2 Example of the iterative procedure. We modify the background temperature of a test tokamak with ICRH. The profile converges after 6 iterations

2.3 Benchmark of the Code

The *benchmark* of ISDEP is performed in different tests to check the validity of the results presented in this thesis and related works. First, the guiding center motion without collisions is compared with another orbit code [17] with a nice coincidence of the results. Then the collision operator is tested by estimating the energy slowing down time and comparing it with the standard theory. Finally, the particle diffusion in a circular tokamak geometry is compared with the one estimated by the code MORH (Monte-Carlo code based on Orbit following in the Real coordinates for Helical devices) [7].

In the first step a proton trajectory in the Stellarator TJ-II (see Chap. 4 for more details on the device) is compared with the calculated by means of the code used in [17]. Both trajectories start at the same initial point and for the first times the agreement is good, as can be seen in Fig. 2.3. After some toroidal turns around TJ-II the numerical errors in the interpolation of **B** accumulate and the trajectories start to differ. These results show that this module of ISDEP is validated.

The collision operator is validated in a small circular tokamak with characteristics $R_0 = 1\,\mathrm{m}$, $a = 0.2\,\mathrm{m}$, $B \sim 1\,\mathrm{T}$. We consider flat profiles to avoid the influence of the transport, and only one background species ($T_i = 100\,\mathrm{eV}$, $n_i = 10^{20}\,\mathrm{cm}^{-3}$). A population of test particles with $T(t=0) \neq T_i$ is evolved with ISDEP in velocity

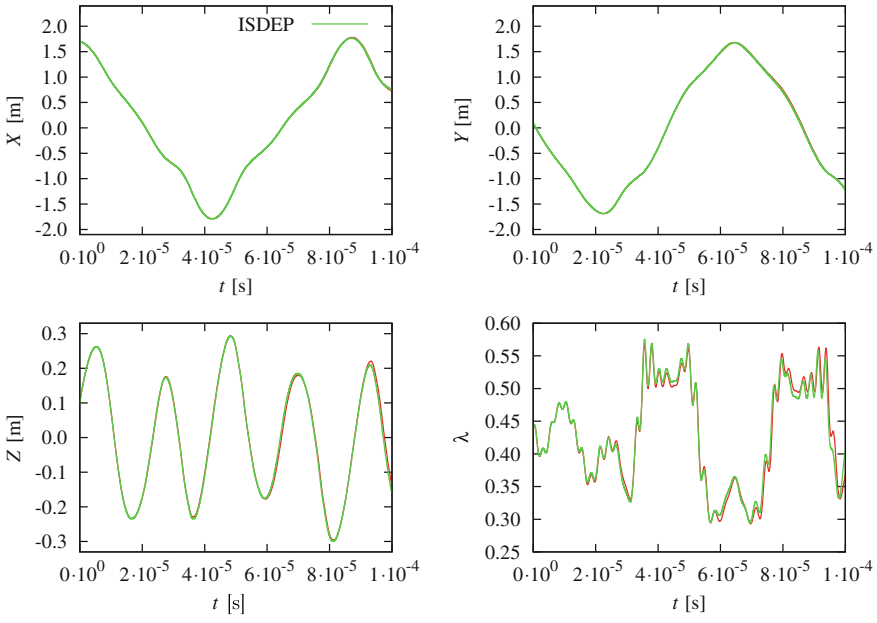

Fig. 2.3 Same trajectory in TJ-II calculated with ISDEP (*green line*) and the code in [17] (*red line*). The initial point is $(x, y, z, v^2, \lambda) = (1.7,\ 0.1,\ 0.1,\ 10^{-6}\mathrm{c}^2,\ 0.44)$. The energy is conserved in all the trajectory

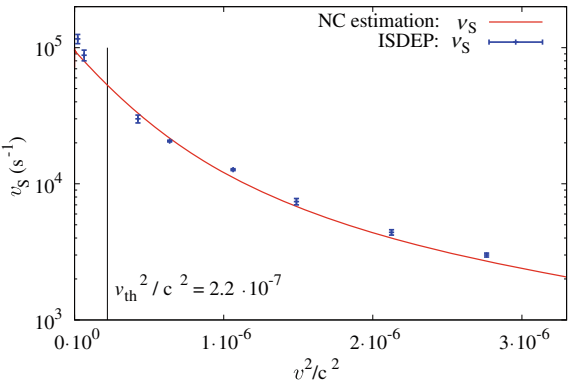

Fig. 2.4 Comparison of the energy slowing down frequency ν_S calculated with ISDEP and the NC prediction. The vertical line corresponds to the ion thermal velocity

space but not in position. In this way the effect of the collision operator is isolated. These test particles tend to thermalize with certain frequency depending on the initial temperature. Fitting the test particle temperature with $T(t) = T_i + Ae^{-t/\nu_S}$ we can calculate the slowing down frequency and compare with the theory [18]. The theoretical expression of ν_S is:

$$\nu_S = \frac{8\pi e^4 n \ln \Lambda \Psi(x)}{m^2 v^3}, \tag{2.33}$$

with the notation of Sect. 1.3.2. A good agreement between our calculations and the theory is found, as can be seen in Fig. 2.4, so we also consider the collision operator validated.

As a final test, both ISDEP and the Monte Carlo code MORH [7] are adapted to the small tokamak used here, but with more realistic profiles and including a radial electric field (see Fig. 2.5). In order to include 3D features a small ripple (1 %) is considered, following [19] (this will also be used in Chap. 3). In addition, test particles collide with ions and electrons of the background plasma, taking $T_i = T_e$, $n_i = n_e$.

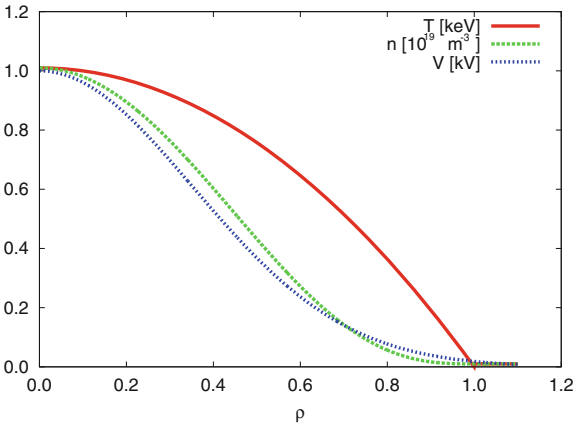

Fig. 2.5 Plasma profiles of the tokamak used in the *benchmark*

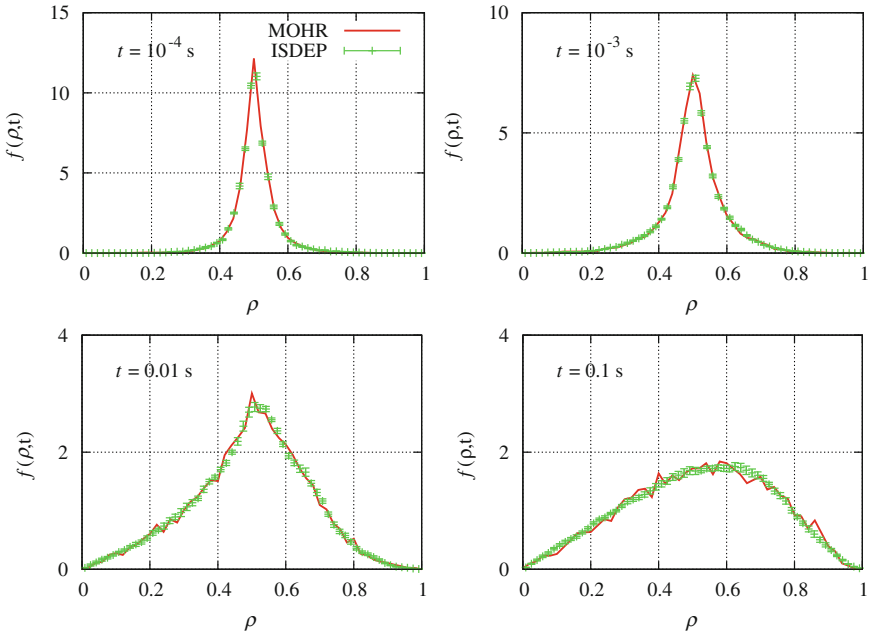

Fig. 2.6 Diffusion of particles in a circular tokamak geometry calculated with ISDEP (*green*) and MORH (*red*). The initial state in both cases is $f(\rho, t = 0\,\mathrm{s}) = \delta(\rho - 0.5)$

In both simulations a population of test particles is launched from $\rho = 0.5$ in position and with a Maxwellian distribution in velocity space. The 1D test particle distribution function $f(\rho, t)$ is plotted in Fig. 2.6 for several times, comparing the results of both codes. Also the average radial velocity $\langle v_\rho \rangle$ and the persistence $P(t)$ are plotted in Fig. 2.7. The persistence is defined as the fraction of surviving particles. The two codes present a general good agreement. In Fig. 2.6 ISDEP and MORH reproduce the same dynamics of $f(\rho, t)$, both in the width and in the asymmetry. The differences in v_ρ are due to the different integrators used in the two codes, but they are not statistically significant. The accumulation of numerical errors cause a discrepancy in the persistence for $t > 0.01$ s (see Fig. 2.7). The particle loss conditions are much more sensitive to numerical errors than other quantities of the plasma. The average radial velocity calculated with ISDEP is compatible with MOHR, although it is a very noisy quantity (Fig. 2.7).

As a conclusion, we consider that the ISDEP code is *benchmarked*.

2.4 Overview of Previous Physical Results

Previously to the elaboration of this thesis, ISDEP has been already used in fusion science for the following purposes.

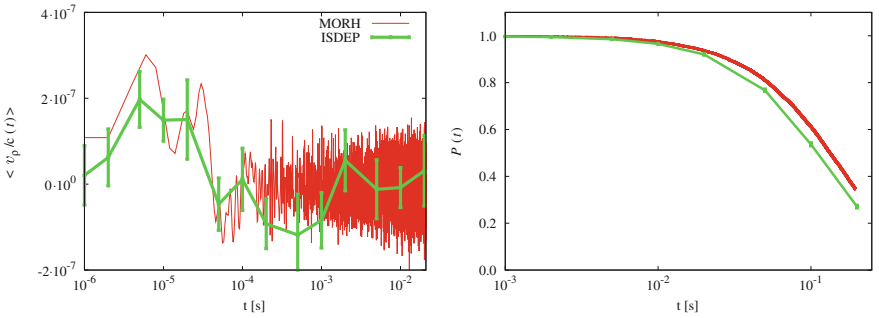

Fig. 2.7 Average radial velocity and test particle persistence calculated with ISDEP (*green*) and MORH (*red*)

2.4.1 Thermal Ion Transport in TJ-II

ISDEP has been mainly applied to the TJ-II stellarator [20]. A first work calculating the transport of thermal ions in ECRH plasmas was published in [21]. The thermal ion transport in absence of ion-electron collisions was calculated in that paper, where the violation of the Neoclassical local ansatz was explicitly shown. Additionally, this work estimated the poloidal accumulation of particles and deviation of the test-particle distribution function from de Maxwellian. From these data, a first estimate of the ion contribution to the bootstrap current was provided.

2.4.2 CERC and Ion Confinement

A particular effect, known as the Core Electron Root Confinement (CERC) was simulated in [16], exploring its influence on ion confinement. CERC means the enhancement of the electron heat confinement with the onset of a strong positive radial electric field. Collisions with electrons and the self consistent scheme for plasma temperature modification was implemented here with a complex workflow on the Fusion Virtual Organization of the EGEE Grid. The conditions of the plasma before and after the transition to CERC were simulated. The variation of the radial electric field and the rising of the electron temperature were the ingredients required to reproduce the experimentally observed rise of the ion temperature.

2.4.3 Violation of Neoclassical Ordering in TJ-II

Non-diffusive features of the radial transport in TJ-II can be found in [22], showing that even this linear collisional model is enough to find non-diffusive transport in plas-

mas. The radial transport was estimated for ions at different radial locations and different plasma regimes. A rough estimate of the Hurst exponent (which quantifies the diffusivity of transport) was extracted from the simulations. The local ansatz was shown to be approximately fulfilled for plasmas heated by Neutral Beam Injection (NBI).

2.4.4 Flux Expansion Divertor Studies

The last previous paper on TJ-II included a detailed study of escape particles and divertor effects [23]. The 3D fluxes on the plasma wall were calculated in several configurations in order to exploit their potential as flux expansion divertor. The toroidal and poloidal resolution available in ISDEP was a key factor in the proposal of a new divertor at TJ-II.

References

1. Boozer AH (2005) Rev Mod Phys 76:1071
2. Beidler C et al (2011) Nucl Fusion 51:076001
3. Hirshman S et al (1986) Phys Fluids 29:2951
4. Allmaier K, Kasilov S, Kernbichler W, Leitold GO (2008) Phys Plasmas 15:072512
5. Tribaldos V (2001) Phys Plasmas 8:1229
6. Lotz W, Nührenberg J (1988) Phys Fluids T31:2984
7. Seki R et al (2010) Plasma Fusion Res 5:014
8. Kloeden PE, Platen E (1992) Numerical solution of stochastic differential equations. Springer-Verlag, Berlin
9. Teubel J (1994) Monte carlo simulations of NBI into the TJ-II helical axis stellarator. Max planck institute fur plasmaphisik. 4/268, Germany
10. Amit D, Martin-Mayor V (2005) Field theory the renormalization group and critical phenomena, 3rd edn. World Scientific Publishing, Singapore
11. Castejón F et al (2008) Comput Inf 27:261
12. Benito D et al (2008) In: Proceedings for 2008 ibergrid meeting, Porto, Portugal, p 273
13. Antolí B et al (2007) In: Proceedings of the CEDI, II congreso espanol de informática conference, Zaragoza, p 523
14. Murakami S et al (2006) Nucl Fusion 46:S425
15. Osakabe M et al (2010) Plasma Fusion Res 5:014
16. Velasco J et al (2008) Nucl Fusion 48:065008
17. Castejón F et al (2006) Fusion Sci Technol 50:412
18. Huba J (2009) NRL plasma formulary. Taylor and Francis, USA
19. Boozer A, Kuo-Petravic G (1981) Phys Fluids 24(5):851
20. Alejaldre C et al (1990) Fusion Technol 17:131
21. Castejón F et al (2007) Plasma Phys Controlled Fusion 49:753
22. Velasco J, Castejón F, Tarancón A (2009) Phys Plasmas 16:052303
23. Castejón F et al (2009) Nucl Fusion 49:085019

Springer

http://www.springer.com/978-3-319-00421-1

Kinetic Simulations of Ion Transport in Fusion Devices

de Bustos Molina, A.

2013, XI, 128 p. 70 illus. in color., Hardcover
ISBN: 978-3-319-00421-1