

OPEN ACCESS

Recent developments on *PLASMAKIN* – a software package to model the kinetics in gas discharges

To cite this article: N R Pinhão 2009 *J. Phys.: Conf. Ser.* **162** 012006

View the [article online](#) for updates and enhancements.

Related content

- [Diagnosis diagrams for passing signals on an automatic block signaling railway section](#)
E Spunei, I Piroi, C P Chioncel et al.
- [Proximity Effects Correction for Advanced Optical Lithography Processes](#)
Alexander Tritchkov, Jo Finders, John Randall et al.
- [Analysis of seismic stability of large-sized tank VST-20000 with software package ANSYS](#)
A A Tarasenko, P V Chepur and A A Gruchenkova

Recent citations

- [PumpKin: A tool to find principal pathways in plasma chemical models](#)
A.H. Markosyan *et al*



IOP | ebooks™

Bringing together innovative digital publishing with leading authors from the global scientific community.

Start exploring the collection—download the first chapter of every title for free.

Recent developments on *PLASMAKIN*— a software package to model the kinetics in gas discharges

N R Pinhão

ITN - Nuclear and Technological Institute, Estrada Nacional 10, 2685 Sacavém Portugal

E-mail: npinhao@itn.pt

Abstract. *PLASMAKIN* is a user-friendly software package to handle physical and chemical data used in plasma physics modeling and to compute the production and destruction terms in fluid models equations. These terms account for the particle or energy production and loss rates due to gas-phase and gas-surface reactions. The package has been re-structured and expanded to (a) allow the simulation of atomic emission spectra taking into account line broadening processes and radiation trapping; (b) include a library to compute the electron kinetics; (c) include a database of species properties and reactions and, (d) include a Python interface to allow access from scripts and integration with other scientific software tools.

1. Introduction

The modeling of non-equilibrium plasmas through fluid equations requires the computation of terms expressing, for the several species involved in the discharge, the changes in density, momentum or energy¹ due to collisional or radiative processes. These terms represent the local 'chemistry' and are, in most of the cases, independent of the transport model or algorithm chosen for the solution of the fluid model equations. In most cases the evaluation of these terms is straightforward. However as the number of species and reactions involved can be very high, a good data management facility is required.

One common approach for the computation of these terms is to write down the equations for each species directly in the program. However this has the disadvantage of linking the program to a specific discharge model, gas composition and set of reactions. The risk of making trivial but probably time consuming errors is also high. With this approach, the 'chemistry modeller' and the 'programmer' have to be the same person.

A better alternative, followed in several commercial [1] and non-commercial packages [2, 3] for chemical kinetics or plasma modeling, is to keep the information on species and reaction in an external file. This file can then be pre-processed by a parser to automatically generate a program code, to be further compiled with a user's main program. With this approach the pre-processing and compilation can be done in an automatic way and the task of selecting the reactions and writing the datafile is separated from coding. However the final program is still specific for a given model and conditions. Any change in reaction values requires a new pre-processing/compiling cycle.

¹ usually the energy conservation equation is only solved for electrons.

A further improvement can be obtained if all processing of species and reactions is handled by a library called from a user's program. This library can provide the information needed by the main program, e.g. species production and destruction terms for continuity equations without requiring any customization or compilation. The properties and processing of species and reactions are hidden from the calling program and the user's program only has to be compiled and linked with the library once. Any change on gas composition or reactions is only reflected on the data input file. Besides the same program can be used for a different gas mixture provided the type of discharge is the same.

This approach has driven the development of the first version of *PLASMAKIN* [4] as a chemical kinetics library. However the study of gas discharge kinetics is not limited to the study of the chemical kinetics.

Another frequent requirement is to obtain electron rate and transport coefficients from the solution of the electron Boltzmann equation (eBE) for a given set of conditions and cross-section data. A common practice is to obtain the required data in tabulated form as a function of the reduced electric field, E/N , or mean electron energy, ε_e . However the solution of the electron Boltzmann equation is a demanding problem requiring high specialization. Fortunately there are also several eBE solvers available either commercially [5] or non-commercially [6]. Both of those codes rely on the classical two-term expansion for the angular dependency of the electron distribution function. The results obtained with this approximation are adequate for most discharges. However it is well known that this approximation is insufficient in high E/N or when the cross sections for inelastic processes become comparable with those for elastic processes [7] and incapable of predicting the values of transport coefficients with the same precision as that achieved in electron swarm experiments.

Finally, a model of a plasma discharge needs to be validated against experimental results. The usual output of a discharge model are the values of species density in space and in time, a voltage or current signal and other diagnostics such as the electron temperature or the relative contribution of each reaction for each species. These are very useful information to understand what happens in the discharge and how it happens. However except for the voltage or current signal, it is not always easy to get experimental access to the quantities obtainable by a model. Often experimental information on values predicted by a model is obtainable indirectly through, e.g. the analysis of the UV-VIS emission of the plasma. The optical spectrum can give information on the density of species, the plasma and electron temperatures or the pressure. It seems reasonable to expect that discharge models would, in turn, also compute the emission spectrum. Besides, a good amount of the information needed to simulate the spectrum is already present in the models: species densities, temperature profiles and emission coefficients. However very few models offer this possibility and among the reasons are the complexity of computing line profiles when radiation trapping is present, the need to consider a large number of levels to obtain realistic spectra, and the treatment of molecular spectra. However a few programs are available through the CPC program library to compute the effect of radiation trapping [8, 9, 10] likewise, a program to simulate the emission spectra of some diatomic molecules is freely available [11].

The present communication reports the recent developments on the OpenSource package *PLASMAKIN* in order to address a few of these concerns.

PLASMAKIN has grown from a chemical kinetics library to a collection of libraries and tools for plasma modeling. It presently includes a database of species and reactions, libraries to compute the chemical kinetics or the electron kinetics, the simulation of the emission spectra of atomic gases and a binding module to allow access of the libraries' routines from Python scripts or command line.

The next chapter introduces the overall structure of the package. The new capacities added to *PLASMAKIN* are detailed in chapters 3 through 6. Finally, we draw some conclusions.

2. Package Structure

The package has been split into several Fortran modules allowing a better organization of the code and facilitating future development and introduction of new facilities. These modules perform the following tasks:

- (i) Set data structures and common routines for all *PLASMAKIN* modules (**pkCommon**);
- (ii) read and process data either from text datafiles or a database (**pkDatasql**);
- (iii) define methods for inquiring and setting species and reaction properties (**pkBase**);
- (iv) compute the chemical kinetics, the power losses in collisions and the atomic emission spectrum (**pkChemicalKin**); and
- (v) compute the electron kinetics: the electron velocity distribution function (evdf), transport parameters and electron collision rate coefficients (**pkElectronKin**).

The relationship between these modules is shown in figure 1.

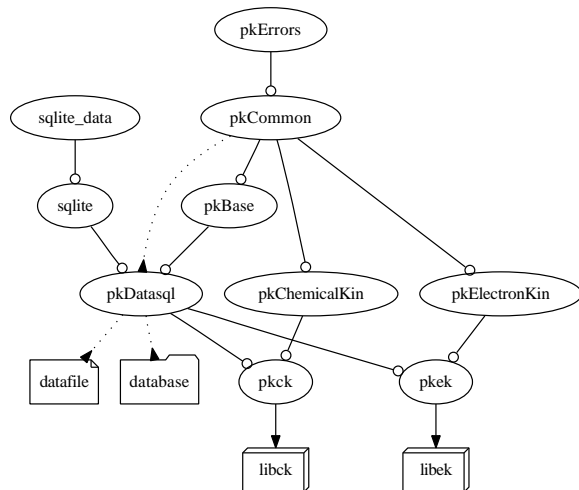


Figure 1. Structure of modules, data sources and libraries in the *PLASMAKIN* package. (Not shown in the figure, the package also includes a Python module to allow access of routines of both libraries from Python scripts or command line.)

The compilation of these modules produces two libraries (**libck** and **libek**) responsible for dealing with the chemical kinetics and the electron kinetics, respectively.

The compilation of the libraries is managed by the GNU **autotools** system that takes care of checking the necessary requirements and adapt the compilation options to the capacities of the local operating system. To obtain a working installation, only three instructions are required: *configure*, *make* and *make install* (as root). To check if everything is well installed the instruction *make check* runs several tests and compares the output with preset values.

The package also includes a sample database of species and reactions and a Python module providing bindings for the libraries. The installation of these tools is optional and requires support to *sqlite*² and a Python interpreter, respectively.

3. Data types and data input

3.1. Species in Local Thermodynamic Equilibrium

The type of species supported in *PLASMAKIN* has been extended to include levels in local thermodynamic equilibrium (LTE) or partial LTE. The population of these levels, n_k , is defined relative to a reference level n_o by a Boltzmann factor:

$$n_m = n_o \frac{g_m}{g_o} \exp \left(-\frac{\varepsilon_m - \varepsilon_o}{k_B T_e} \right) \quad (1)$$

² *sqlite* is a public domain software library that implements a SQL engine that does not require a server and accesses database files on a local disk. It is widely used and can be found in all operating systems.

where g and ε are the statistical weight and energy of the levels, and respectively, T_e the electron temperature.

The LTE levels don't need to be tracked in a program using *PLASMAKIN* but contribute to other levels through radiative decay and are taken into account in the simulation of emission spectra.

The population of LTE levels is computed by the library from T_e and the density of the reference level. If the later is a cascade level (e.g. produced by electron excitation and rapidly decaying to a lower excited level), the density n_o in equation (1) is replaced by

$$n_o = \frac{\sum_{ij} k_{ij}^o n_i n_j}{\sum_m A_{om}} \quad (2)$$

where the sums include all reactions producing level o and all radiative transitions from level o , respectively. If any of the species involved in (2) is also a cascade level, its density is determined again from (2).

3.2. Database

The way data is input in *PLASMAKIN* has been extended in order to reduce the work in preparing the datafile.

Until now all the information on species and reactions was written in an text file. This method of input it is still valid but from now on is also possible to keep most of the data in a database and limit the use of a text datafile to the following: the indication of the database, the database tables used, to impose restrictions on data and to include additional species or reactions missing in the database. With this approach, it is possible for example to keep well established results in the database and use the text datafile only to test the influence of new data on a discharge model.

4. Simulation of atomic emission spectra

The spectral radiation intensity, I_ν , emitted by a plasma and observed at a boundary x_0 , is given by

$$I_\nu(x_0) = \int_0^{x_0} \epsilon_\nu(x) \exp[-\tau(\nu, x, x_0)] dx \quad (3)$$

where $(0, x_0)$ are the boundaries of the plasma along the direction of observation, $\epsilon_\nu(x)$ is the spectral emission coefficient of the plasma along the direction of observation and $\tau(\nu, x, x_0)$ is the optical path between x and x_0 ,

$$\tau(\nu, x, x_0) = \int_x^{x_0} k(\nu, x') dx' \quad (4)$$

and $k(\nu, x)$ the spectral absorption coefficient. In (3) we neglect any contribution from radiation outside the plasma.

In order to solve equation (3) and simulate the emission spectra we have considered the following assumptions:

- (i) We neglected the stimulated emission in the plasma;
- (ii) we assume that the emission and absorption line profiles are the same, $P_e(\nu) = P_a(\nu) \equiv P(\nu)$;
- (iii) we assume complete frequency redistribution to compute the line profiles;

- (iv) we limit ourself to two discharge geometries: plane parallel and cylindrical. In the last case the direction of observation is perpendicular to the cylinder axis and crosses the cylinder at a height Y from the axis. Additionally, in this case we assume that the discharge has radial symmetry; and
- (v) we presently limit the simulation of spectra to atomic emission lines.

4.1. Optically thin lines

With the above assumptions, and considering for now that $\tau \ll 1$, equation (3) can be written

$$I_\nu(R) = \zeta \frac{h\nu}{4\pi} \sum A_{ij} \int_Y^R n_i(r) P_{ij}(\nu, r) \eta_Y(r) dr \quad (5)$$

where the sum is on all radiative transitions between levels i and j , R is the cylinder radius or the outer boundary and $\eta_Y(r)$ a geometric factor. For plane parallel geometry $\zeta = 1$, $Y = 0$, $\eta_0(r) = 1$ while for cylindrical geometry $\zeta = 2$, $\eta_Y(r) = r/\sqrt{r^2 - Y^2}$. The factor $\zeta = 2$ for cylindrical geometry accounts for photons emitted along the line of sight from points with the same value of r but symmetrically positioned regarding the plane perpendicular to the line of sight and passing by the axis of the cylinder.

The normalized line profile, $P(\nu, r)$ depends, apart from the gas temperature and the atom and electron concentrations, upon the constants that characterize the interaction between the radiating atom and the neighboring particles, and upon the broadening mechanism.

Three line profiles are of importance:

- (i) A Gaussian distribution due to Doppler broadening of the lines and given by

$$P_D(x) = \frac{1}{\pi^{1/2}} \exp(-x^2), \quad x = \frac{\nu - \nu_0}{W_D} \quad (6)$$

with ν_0 the line central frequency and $W_D = \frac{\nu_0}{c} \sqrt{\frac{2k_B T}{m}}$, the Doppler width.

- (ii) A Lorentz distribution from natural broadening or pressure broadening in the impact approximation,

$$P_L(x) = \frac{1}{\pi} \frac{1}{1 + x^2}, \quad x = \frac{\nu - \nu_0 - \Delta}{W_L} \quad (7)$$

with Δ a line shift and W_L a Lorentz width characteristic for each type of pressure broadening process. Three types of broadening processes need to be considered:

atom-impact resulting in a van der Waals interaction from a Lennard-Jones potential with a characteristic constant C_6 ;

electron- and ion-impact resulting in Stark broadening, calculated from the Sahal-Brechot theory [12];

resonance broadening resulting from resonance transfer of excitation from collisions of two identical atoms, one of which is excited.

Expressions for the line width and line shift for each type of broadening can be found in [13, 14] and are summarized in Table 1.

- (iii) A line profile from pressure broadening computed in the quasi-static approximation and valid for the far wings,

$$P_I(\nu) = \frac{3}{k\Delta\nu} \left(\frac{\Delta\nu}{\nu - \nu_0} \right)^{\frac{3+k}{k}} \exp \left[- \left(\frac{\Delta\nu}{\nu - \nu_0} \right)^{\frac{3}{k}} \right] \quad (8)$$

with $\Delta\nu = C_k r_0^{-k}$ and $r_0 = \left(\frac{4}{3} \pi n_0 \right)^{-1/3}$ equal to the mean distance between particles. For Stark effect $k = 4$ and for Van der Waals interaction $k = 6$.

Table 1. Line width and line shift for different types of pressure broadening in the impact approximation.

		<i>Type of broadening</i>		
		e- and ion-impact	Atom-impact	Resonance
$2W_L$	$(C_e T^{\gamma_e} + C_i T^{\gamma_i}) n_e$		$8.08 C_6^{2/5} \bar{v}^{3/5} n_0$	$K_r \frac{\pi e^2}{m_e \nu_0} f n_0$
Δ	$(B_e T^{\beta_e} + B_i T^{\beta_i}) n_e$		$0.7 W_L$	0

The actual line profile results from the convolution of the line profiles for all broadening mechanism. Stormberg [15] and Damelincoirt *et. al.* [14] have shown that the convolution of the different profiles can be approximated by the sum of a Voigt profile and a correction term from the quasi-static profile, $P(x) = P_V(x) + C(x)$ with,

$$P_V(x) = \frac{1}{\pi^{1/2}} H(a, x), \quad H(a, x) = \frac{a}{\pi} \int_{-\infty}^{+\infty} \frac{\exp(-y^2)}{a^2 + (x - y)^2} dy \quad (9)$$

where x is the normalized Doppler frequency in equation (6) and $a = W_L/W_D$. Here W_L is the sum of the Lorentz widths for all broadening processes. The Voigt integral is integrated using the polynomial approximation of Humlicek [16] which gives an accuracy of 10^{-4} .

In actual measurements the spectrum obtained depends on the instrument resolution, δ . To take this into account, the spectrum is computed in intervals of width δ where equation (3) is integrated with a sufficiently small resolution.

$$\bar{I}_k = \frac{1}{\delta} \int_{\nu_k}^{\nu_k + \delta} I_R(\nu) d\nu, \quad k = 1, \dots, N_\lambda \quad (10)$$

To obtain the spectrum the user must call a routine supplying a wavelength range, the instrument resolution, δ , the grid points, \vec{X} , along the line of sight or the radius, respectively for plane parallel or cylindrical geometries, the density of species along \vec{X} , and as options, the constants listed in table 1 for pressure broadening processes for each relevant species, gas and electron temperature profiles, $T_g(\vec{X})$, $T_e(\vec{X})$, respectively, and Y .

Figures 2 – 4 show the result of three tests on the routine.

Figures 2 and 3 show the simulation of an emission spectra from neon in the range 550 – 900 nm, with the population of levels in LTE with an electron temperature of 5 eV and obtained with two different instrument resolutions. All transitions from the $2p^2 2p^5 3s$, $2p^2 2p^5 3p$, $2p^2 2p^5 4s$, and $2p^2 2p^5 3d$ configurations have been included. A comparison of figure 2 with a Saha-LTE spectrum from the NIST Atomic Spectra Database [17], obtained in the same conditions, shows a very good agreement with just a few lines from transitions involving higher configurations missing in figure 2.

In another example using the same neon data in LTE, we have considered an electron temperature radial profile given by a Bessel function, $T_e(r) = T_e(0) J_0(j_{0,1} \cdot r/R)$ where $j_{0,1}$ is the first root of J_0 . Figure 4 shows the ratio between the normalized spectra obtained with $T_e(0) = 10$ eV and $T_e(0) = 5$ eV, respectively.

4.2. Optically thick lines

In this case, for a plane-parallel geometry, equation (3) is written

$$I_\nu(x_0) = \frac{h\nu}{4\pi} \sum A_{ij} \int_0^{x_0} n_i(x) P_{ij}(\nu, x) \Phi_{j,0}(\nu; x, x_0) dx \quad (11)$$

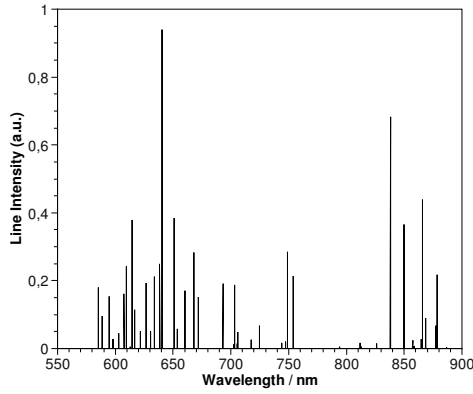


Figure 2. Simulated neon emission spectrum in LTE conditions with $T_e = 5$ eV and an instrument resolution of $\delta = 0.025$ nm.

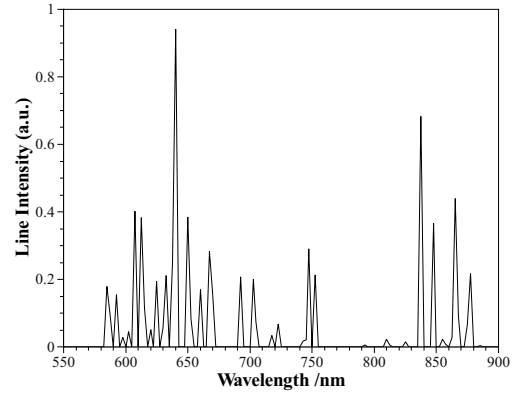


Figure 3. Same conditions as in Figure 2, with an instrument resolution of $\delta = 2.5$ nm.

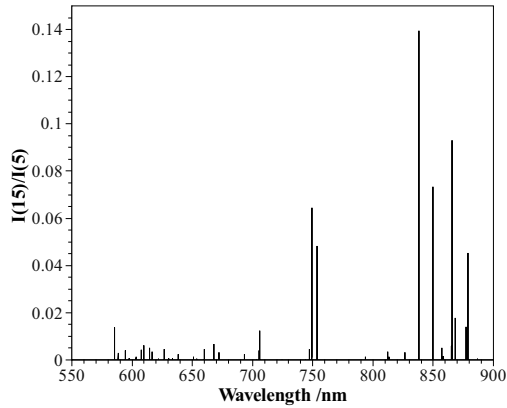


Figure 4. Ratio between the normalized line intensities of neon emission spectra in LTE conditions with a Bessel radial electron temperature profile with, $T_e(0) = 10$, eV and $T_e(0) = 5$ eV, respectively. $\delta = 0.025$ nm.

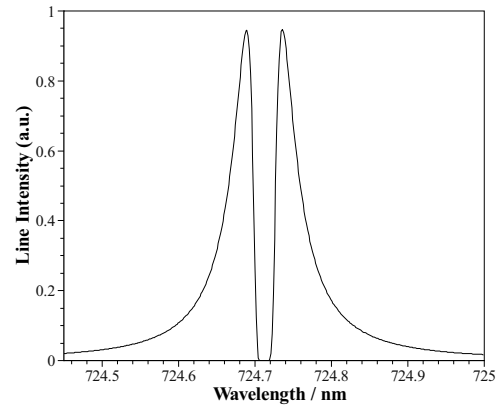


Figure 5. Simulation of a self-reversed line at $p = 1$ atm and a Voigt parameter of $a_V = 0.2$.

with $\Phi_{j,Y}$ the function,

$$\Phi_{j,Y}(\nu; a, b) = \exp \left[-k_{j,0} \int_a^b n_j(x') P_{ji}(\nu, x') \eta_Y(x') dx' \right] \quad (12)$$

where $k_{j,0} = (c^2/8\pi\nu_0^2)(g_j/g_i)A_{ij}$ and the remaining symbols defined as in equation (5).

For cylindrical geometry, we obtain

$$I_\nu(R) = \frac{h\nu}{4\pi} \sum A_{ij} \int_Y^R n_i(r) P_{ij}(\nu, r) \Psi_{j,Y}(\nu; Y, r, R) \eta_Y(r) dr \quad (13)$$

with

$$\Psi_{j,Y}(\nu; Y, r, R) = \Phi_{j,Y}(\nu; Y, r) \Phi_{j,Y}(\nu; Y, R) + \Phi_{j,Y}(\nu; r, R) \quad (14)$$

The function Ψ is equivalent to ζ in equation (5) but takes into account the absorption along the line of sight. The first term accounts for absorption of photons emitted beyond the plane defined by the axis of the cylinder and perpendicular to the line of sight and the last term for photons emitted in front of this plane.

Taking into account that $\Phi_{j,Y}(\nu; Y, R) = \Phi_{j,Y}(\nu; Y, r)\Phi_{j,Y}(\nu; r, R)$, we can write

$$\Psi_{j,Y}(\nu; Y, r, R) = \frac{\Phi_{j,Y}(\nu; Y, R)^2}{\Phi_{j,Y}(\nu; r, R)} + \Phi_{j,Y}(\nu; r, R). \quad (15)$$

As $\Phi_{j,Y}(\nu; r, R) = \Phi_{j,Y}(\nu; r, r + \delta r)\Phi_{j,Y}(\nu; r + \delta r, R)$ and $\Phi_{j,Y}(\nu; R, R) = 1$, the evaluation of Ψ requires the computation of $\Phi_{j,Y}(\nu; r, r + \delta r)$ in each grid interval, starting from $r = R$ and moving backwards to $r = Y$.

Figure 5 shows a line profile obtained for an optically thick line. In this case the broadening values were chosen to evidence self-reversal.

5. The electron kinetics library

A collection of routines to study the electron kinetics under well defined assumptions and compute the electron velocity distribution function, *evdf*, and the transport and rate coefficients for the electrons are now included in the package. The assumptions used apply to a discharge in a constant electric field with or without grown of the electron density.

The main purpose of this library is to obtain values of the transport and rate coefficients for the electrons as a function of E/N or the electron mean energy that can be used on fluid models.

The behaviour of the electrons in a discharge is described by the electron Boltzmann equation

$$\frac{\partial f}{\partial t} + \vec{v} \nabla_r f + \vec{a} \nabla_v f = C(f), \quad (16)$$

for the *evdf*, $f(\vec{r}, \vec{v}, t)$, where \vec{r} , \vec{v} and t are the position, velocity and time, respectively, \vec{a} denotes the acceleration produced by the external field and $C(f)$ is the rate of exchange due to collisions.

The above equation is far too complex to have a general solution and for the time been the algorithm implemented in the library is limited to spatially uniform and constant electric fields. In this case we have $\vec{a} = -(e/m) \vec{E}$.

We further consider that the electrons are moving in the hydrodynamic regime, meaning that although the electron density grows exponentially, the distribution in velocity space and the transport parameters have reached equilibrium values, independent of their initial value. Under these conditions the time derivative term in equation (16) can be replaced by a rate term,

$$\frac{\partial f}{\partial t} = \Omega f \quad (17)$$

where Ω is the macroscopic effective ionization frequency.

The theory of the hydrodynamic regime is well established [18, 19] and only a very brief outline will be given here.

To allow for the influence of electron density gradients in the *evdf*, $f(\vec{r}, \vec{v}, t)$ is represented as,

$$f(\vec{r}, \vec{v}, t) = \sum_{k=0}^{\infty} F^{[k]}(\vec{v}) \otimes^k (-\nabla)^k n(\vec{r}, t), \quad (18)$$

where $F^{[k]}(\vec{v})$ are velocity-dependent tensors of rank k , \otimes^k is the k -fold inner-product operation and $(-\nabla)^k$ is the k -fold outer product of the gradient operator with itself. Obviously, for a

discharge with a spatially homogeneous electron density, equation (18) reduces to the usual representation, $f(\vec{v}, t) = F(\vec{v})n(t)$.

Inserting (17) and (18) into Boltzmann equation (16), a hierarchy of equations is obtained allowing to solve for the functions $F^{[k]}$ and transport coefficients $\Omega^{(k)}$. The first three of these coefficients are the macroscopic effective ionization frequency, Ω , the drift velocity, \vec{W} and the diffusion tensor, \hat{D} .

The transport parameters measured in time-resolved swarm experiments with \vec{E} along the z-axis, can be written as

$$\begin{aligned} W &= \int v_z F^{[0]} d\vec{v} + \int \nu_{\text{eff}} F_z^{[1]} d\vec{v} \\ D_L &= \int \frac{v_z - W}{\nu + \Omega} v_z F^{[0]} d\vec{v} - \vec{a} \cdot \int \frac{v_z}{\nu + \Omega} \nabla_v F_z^{[1]} d\vec{v} + \int \nu_{\text{eff}} F_{zz}^{[2]} d\vec{v} \\ D_T &= \frac{1}{2} \left\{ \int \frac{v_T^2}{\nu + \Omega} F^{[0]} d\vec{v} - \vec{a} \cdot \int \frac{v_z}{\nu + \Omega} \nabla_v F_T^{[1]} d\vec{v} + \int \nu_{\text{eff}} F_T^{[2]} d\vec{v} \right\} \end{aligned} \quad (19)$$

where D_L and D_T are the longitudinal and transverse components of the diffusion tensor, $\nu_{\text{eff}}(v) = \nu_{\text{ion}}(v) - \nu_{\text{att}}(v)$ is the net ionization frequency, $\nu(v) = \nu_m(v) + \nu_{\text{ex}}(v) + \nu_{\text{eff}}(v)$ is the total collision frequency, and the indices z and T respectively, indicate the longitudinal and transverse components of the velocity \vec{v} or of functions $F^{[k]}$.

The drift velocity W is the center of mass velocity for the electron swarm and, if non-conservative reactions are included, is different from the mean velocity, $\langle v \rangle$, given by the first term on the right side.

The method used to obtain the distribution functions $F^{[k]}$ is an half-range method applied to the density gradients representation developed to explain the results of electron swarm experiments and described in [20]. This method solves the hierarchy of equations for the functions $F^{[k]}$ directly, using a finite elements algorithm on a (v, θ) grid. A comparison between the results obtained using this algorithm and other Boltzmann and Monte Carlo techniques can be found in [7].

The results obtained for $F_n^{[0]}(v)$ can be linked with the usual spherical harmonic expansion terms using the relation

$$F_n^{[0]}(v) = \frac{2n+1}{2} \int F^{[0]}(v, \theta) P_n(\cos \theta) \sin \theta d\theta \quad (20)$$

where $P_n(\cos \theta)$ are Legendre polynomials.

Figure (6), taken from [7], shows a comparison between the first six (6) Legendre expansion coefficients obtained with (20) and a multi-term Legendre expansion using eight (8) terms [21]. The agreement improves if the number of intervals in θ is increased.

6. The Python interface

PLASMAKIN now includes a Python interface allowing the use of the library functions from Python scripts.

The development of a Python module serves several purposes:

- (i) It allows a broader audience to use the library since it does not require knowledge of Fortran or C;
- (ii) it integrates the library with the rich set of Python modules. Of special importance to plasma modeling is the integration with other modules for scientific computing as **Scipy** [22] or **Matplotlib** [23];

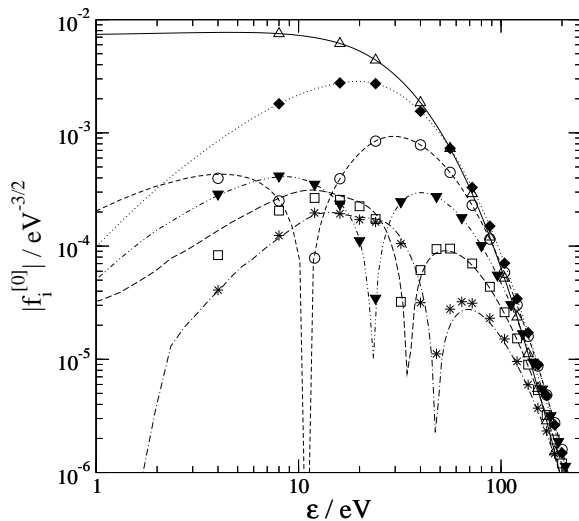


Figure 6. Comparison between the Legendre expansion coefficients $F_n^{[0]}$ for $0 \leq n \leq 5$ calculated using a multi-term Legendre expansion [21] (lines) and the present method at 500 Td in neon: Δ , $—$ $n = 0$; \diamond , \cdots $n = 1$; \circ , $- -$ $n = 2$; ∇ , $- \cdot -$ $n = 3$; \square , $- - -$ $n = 4$; \star , $- \cdot -$ $n = 5$. For $n > 1$ the initial values of the expansion coefficients are negative.

- (iii) although the execution of a script is slower than a compiled program, for 0- or 1-dimension problems the time penalty of running a Python script is not significant. On the other hand development in Python is usually much faster and simpler than in conventional programming languages;
- (iv) it allows the use of Python development tools as `doctest` or `unittest` that have proved useful to track errors in the library itself; and
- (v) speeds up the pre-processing or post-processing of data in a comparable way as what is found in commercial products.

Further details on the Python interface can be found in [24].

7. Conclusions

PLASMAKIN is an OpenSource package for plasma physics modeling open to contributions from the plasma physics community.

Presently is able to handle the chemical kinetics, the electron kinetics for constant E/N and provides useful information as the relative contribution of each process, the power losses in the discharge or the emission spectrum of atomic gases.

It has been extended with a database of species properties and reactions and a Python interface.

Further development of the package has been simplified with a modular organization and the use of the autotools compilation system. Possible directions of development include the simulation of molecular spectra, the treatment of high frequency fields on the electron kinetics, the computation of ion transport properties or the always unfinished task of documentation.

References

- [1] Kee R J, Miller J, Rupley F and Meeks E 1996 Chemkin - iii URL <http://www.reactiondesign.org>
- [2] Carver G D, Brown P and Wild O 1997 *Comp. Phys. Commun.* **105** 197
- [3] Pancheshnyi S, Eismann B, Hagelaar G and Pitchford L 2008 computer code *zdpaskin* URL <http://www.zdpaskin.univ-tlse.fr/>
- [4] Pinhão N R 2001 *Comp. Phys. Commun.* **135** 105–131
- [5] Morgan W L and Penetrante B M 1990 *Comp. Phys. Commun.* **58** 127–152
- [6] Hagelaar G J M and Pitchford L C 2005 *Plasma Sources Sci. Technol.* **14** 722–733
- [7] Pinhão N R, Donkó Z, Loffhagen D, Pinheiro M J and Richley E A 2004 *Plasma Sources Science and Technology* **13** 719–728 URL <http://stacks.iop.org/0963-0252/13/719>

- [8] Molisch A F, Oehry B P, Schupita W and Magerl G 1993 *Comp. Phys. Commun.* **74** 81–90
- [9] Molisch A F, Oehry B P, Schupita W and Magerl G 1993 *Comp. Phys. Commun.* **77** 255–62
- [10] Molisch A F, Oehry B P, Schupita W and Magerl G 1996 *Comp. Phys. Commun.* **93** 127–35
- [11] Luque J and Crosley D R 1999 Lifbase: Database and spectral simulation Tech. Rep. MP 99-009 SRI International URL <http://www.sri.com/psd/lifbase/>
- [12] Sahal-Brechot S 1974 *Astr. Astrophys.* **35** 319
- [13] Karabourniotis D 1986 *Radiative Processes in Discharge Plasmas* (NATO ASI Series vol 149) (Plenum Press) chap Self-Reversed Emission Lines in Inhomogeneous Plasmas, pp 171–247
- [14] Damelincoirt J J, Aubes M, Fragnac P and Karabourniotis D 1983 *J. Appl. Phys.* **54** 3087–3097
- [15] Stormberg H P 1980 *J. Appl. Phys.* **51** 1963–69
- [16] Humlicek J 1982 *JQSRT* **27** 437–44
- [17] Ralchenko Y, Kramida A, Reader J, Saloman E, Sansonetti J, Curry J, Kelleher D, Fuhr J, Podobedova L, Wiese W, Olsen K *et al.* 1979-2008 NIST atomic spectra database URL <http://physics.nist.gov/PhysRefData/ASD/index.html>
- [18] Kumar K, Skullerud H R and E R R 1980 *Australian Journal of Physics* **33** 343–448
- [19] Sakai Y, Tagashira H and Sakamoto S 1977 *J. Phys. D: Appl. Phys.* 1035
- [20] Ségur P, Yousfi P M and Bordage M C 1984 *J. Phys. D: Appl. Phys.* **17** 2199–2214
- [21] Loffhagen D and Winkler R 1996 *J. Phys. D: Appl. Phys.* **29** 618–27
- [22] Jones E, Oliphant T, Peterson P *et al.* 2001- Scipy: Open source scientific tools for python URL <http://www.scipy.org>
- [23] Dale D, Droettboom M, Firing E, Hunter J *et al.* 2008 Matplotlib URL <http://matplotlib.sf.net>
- [24] Pinhão N R 2007 *The Python Papers* **2** 35–47 URL <http://pythonpapers.org/>