IPP-Report

Max-Planck-Institut
für Plasmaphysik

Ralph Dux

STRAHL User Manual

IPP 10/30
September, 2006

STRAHL User Manual

Ralph Dux

Max-Planck-Institut für Plasmaphysik
D–85748 Garching b. München
Assoziation EURATOM–IPP

September 2006

Contents

1 Introduction **1**

1.1 Impurity Transport Equation 1

1.2 Neoclassical Transport 4

1.3 Numerical Solution 5

1.4 Detailed Numerical Scheme 8

1.5 Neutral Impurities 11

2 Input Files **13**

2.1 Plasma Background 13

2.2 Geometry 17

2.3 Atomic Data 22

2.4 Main Parameter File 25

3 Execution and Output **35**

3.1 Executing STRAHL 35

3.2 The Result File 36

3.3 Plotting Routines for IDL 38

Chapter 1

Introduction

STRAHL is a code to calculate the *radial* transport and the emission of impurities in the plasma bulk. The code solves the *radial* continuity equation for each ionisation stage of the impurity in 1-D geometry. For the transport an ansatz of anomalous diffusivities and radial drift velocities is used; a full neoclassical treatment of impurity transport can be switched on if desired. The model focuses on the impurity transport and radiation, while the parameters of the background plasma are taken from the experiment. In this chapter the main aspects of STRAHL are introduced.

The first version of STRAHL was written around 1980. Over the years, the code has gone through several modifications concerning the treatment of atomic data, neoclassical transport and the general scheme of input-output processing. Even though the description of the numerical algorithm [Behringer, 1987] is still valid, a manual has become necessary to introduce the new user to the modifications and to describe the technical details of data input/output in STRAHL.

1.1 Impurity Transport Equation

A short introduction to the *radial* transport equation in the plasma bulk and the definition of the transport parameters will be given first.

The law of particle conservation for the particle density $n_{I,Z}$ of impurity I in ion stage Z may be written as:

$$\frac{\partial n_{I,Z}}{\partial t} = -\nabla\vec{\Gamma}_{I,Z} + Q_{I,Z} \tag{1.1}$$

where $\vec{\Gamma}_{I,Z}$ is the flux density of the impurity and $Q_{I,Z}$ represents the sources and sinks due to ionisation, recombination and charge exchange. The source and sink term connects neighbouring charge states. With an ansatz for the density dependence of the transported flux density $\vec{\Gamma}_{I,Z}$, the particle conservation equation shall be transformed into an impurity transport equation. Here, we are interested in the *radial* transport perpendicular to the magnetic flux surfaces, since the large transport coefficients parallel to the magnetic field cause a practically constant density $n_{I,Z}$ on the magnetic flux surface. Even though $n_{I,Z}$ and also the source/sink term $Q_{I,Z}$ are constant on the flux surface, the radial component of $\vec{\Gamma}_{I,Z}$ shows a variation with the poloidal angle θ. One reason is the poloidally varying distance of the flux surfaces being shorter on the outboard than on the inboard. This leads to a poloidal variation of the gradients of density and temperature, which are the driving terms for the radial impurity flux. Another reason is the $1/R$ dependence of the toroidal magnetic field B_T. In Fig.1.1, the poloidal projection of a few flux surfaces are sketched. For one flux surface, the dependence of the gradient and of B_T on θ are shown in a polar plot. Both quantities vary by about a factor of 2.

The correct flux surface average of Eq.(1.1) is derived by the following steps (see Ref.[Hinton and Hazeltine, 1976]). Inside the last closed flux surface (LCFS), Eq.(1.1) is integrated over

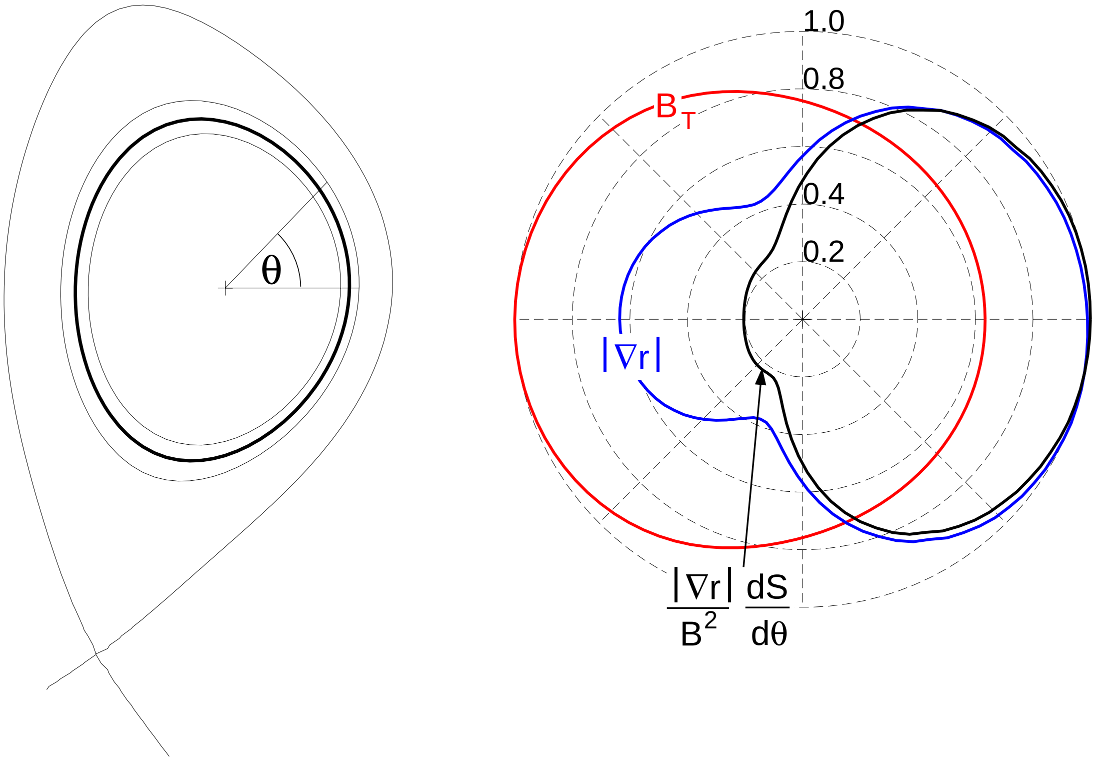


Figure 1.1: *For the magnetic flux surface, whose poloidal projection is drawn on the left with a thick line, the poloidal variation of the toroidal magnetic field(red), the gradient(blue), and the contributions to the classical part of the collisional radial impurity fluxes(black) are shown in the polar plot on the right. All values are normalized to their maximum value.*

the volume V inside a flux surface which leads to.

$$\frac{\partial}{\partial t}\int n_{I,Z}\,dV = -\oint \vec{\Gamma}_{I,Z}\,d\vec{S} + \int Q_{I,Z}\,dV \tag{1.2}$$

The flux surface shall be labeled with ρ, where ρ rises from centre to edge. Thus, $\nabla\rho/|\nabla\rho|$ is the unit vector $\vec{e}_r$ perpendicular to the flux surface pointing towards the outside of the enclosed volume, and the flux density integral becomes

$$\oint \vec{\Gamma}_{I,Z}\,d\vec{S} = \oint \vec{\Gamma}_{I,Z}\frac{\nabla\rho\,dS}{|\nabla\rho|} = \oint \Gamma^{\rho}_{I,Z}\frac{dS}{|\nabla\rho|} \tag{1.3}$$

where $\Gamma^{\rho}_{I,Z}$ is the *radial* contravariant component of the particle flux density $\Gamma^{\rho}_{I,Z} = \vec{\Gamma}_{I,Z}\nabla\rho$. Using the flux surface average of an arbitrary scalar quantity $\mathcal{F}$

$$\langle\mathcal{F}\rangle = \left(\frac{\partial V}{\partial\rho}\right)^{-1}\oint \mathcal{F}\frac{dS}{|\nabla\rho|} \tag{1.4}$$

and

$$\int \mathcal{F}dV = \int_0^{\rho} d\rho\frac{\partial V}{\partial\rho}\langle\mathcal{F}\rangle \tag{1.5}$$

the continuity equation (1.2) becomes

$$\frac{\partial}{\partial t}\int_0^{\rho} d\rho\frac{\partial V}{\partial\rho}n_{I,Z} = -\frac{\partial V}{\partial\rho}\langle\Gamma^{\rho}_{I,Z}\rangle + \int_0^{\rho} d\rho\frac{\partial V}{\partial\rho}Q_{I,Z} \tag{1.6}$$

which can be differentiated with respect to ρ. For a flux surface geometry, which is independent of time, we arrive at.

$$\frac{\partial n_{I,Z}}{\partial t} = -\left(\frac{\partial V}{\partial \rho}\right)^{-1} \frac{\partial}{\partial \rho}\left(\frac{\partial V}{\partial \rho}\langle \Gamma^{\rho}_{I,Z}\rangle\right) + Q_{I,Z} \tag{1.7}$$

We choose the flux surface label r, which is calculated from the enclosed volume V. The volume V depends on r as the volume of a cylinder with length $2\pi R_{axis}$.

$$r = \sqrt{\frac{V}{2\pi^2 R_{axis}}} \tag{1.8}$$

With this choice, the averaged continuity equation is casted in the familiar cylindrical form.

$$\frac{\partial n_{I,Z}}{\partial t} = -\frac{1}{r}\frac{\partial}{\partial r} r\langle \Gamma^{r}_{I,Z}\rangle + Q_{I,Z} \tag{1.9}$$

For the flux density perpendicular to the magnetic surface, an ansatz with a diffusive and a convective part is used. $D(\theta)$ is the radial diffusion coefficient and $v(\theta)$ is the radial drift velocity, where the dependence on the poloidal angle θ is explicitly denoted. The radial covariant component $\Gamma_{I,Z,r}$ of the flux density is.

$$\Gamma_{I,Z,r} = -D(\theta)\nabla n_{I,Z}\vec{e}_r + v(\theta)n_{I,Z} = -D(\theta)\frac{\partial n_{I,Z}}{\partial r}|\nabla r| + v(\theta)n_{I,Z} \tag{1.10}$$

Here, the density gradient has been seperated into a poloidally dependent and independent part using $\nabla n_{I,Z} = (\partial n_{I,Z}/\partial r)\nabla r$. The radial contravariant flux density is obtained by multiplying with $|\nabla r|$.

$$\Gamma^{r}_{I,Z} = \Gamma_{I,Z,r}|\nabla r| = -D(\theta)|\nabla r|^2\frac{\partial n_{I,Z}}{\partial r} + v(\theta)|\nabla r|n_{I,Z} \tag{1.11}$$

Thus, we obtain the *radial* impurity transport equation.

$$\frac{\partial n_{I,Z}}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r} r\left(D^*\frac{\partial n_{I,Z}}{\partial r} - v^* n_{I,Z}\right) + Q_{I,Z} \tag{1.12}$$

Here, D^* and v^* are the flux surface averaged transport coefficients.

$$\begin{aligned} D^* &= \langle D(\theta)|\nabla r|^2\rangle = \frac{1}{4\pi^2 R_{axis} r}\oint D|\nabla r|dS = \frac{1}{4\pi^2 R_{axis} r}\int_0^{2\pi} D(\theta)|\nabla r|\frac{dS}{d\theta}d\theta \\ v^* &= \langle v(\theta)|\nabla r|\rangle = \frac{1}{4\pi^2 R_{axis} r}\oint v dS = \frac{1}{4\pi^2 R_{axis} r}\int_0^{2\pi} v(\theta)\frac{dS}{d\theta}d\theta \end{aligned} \tag{1.13}$$

For the collisional transport parameters, we know the dependence of D and v on the poloidal angle, and the correct surface averages are calculated in the neoclassical transport part of the code. Fig.1.1 depicts the behaviour for the classical part of the collisional impurity flux. The classical diffusion coefficient D_{CL} has a magnetic field dependence $D_{CL} \propto 1/B^2$, and the polar plot of Fig.1.1 depicts the integrand, which appears in Eq.(1.13). All terms, i. e. the magnetic field dependence, the gradient, and the larger surface on the outboard, favour the radial flux contribution from the outboard side being about a factor of 5 larger than the inboard contribution. The fluxes connected to the classical drift velocity v_{CL} have just the same dependence: v_{CL} is proportional to D_{CL} times a combination of gradients of temperature and density, which leads to $v_{CL} \propto |\nabla r|/B^2$. For the turbulent transport, it is also expected, that the main contribution is on the outboard side, however, the exact poloidal dependence is not known. It is important to note that D^* and v^* depend on the choice of the flux surface label. Multiplying r with any constant number k would blow up or shrink our cylinder and yield transport cofficients $D^k = k^2 D^*$ and $v^k = kv^*$. The best choice is a flux surface label with surface averages of $\nabla\rho$ close to 1, which is fulfilled by r. The superscript $*$ will be omitted from now on.

Outside the LCFS in the regime of open field lines, the one dimensional model is not very well suited. Parallel transport to the divertor/limiter can be considered here in the form of a parallel loss time $\tau_{||}$. Thus, in the scrape-off layer (SOL), a term $-n_{I,Z}/\tau_{||}$ is added on the right-hand side of equation (1.12).

The source/sink term $Q_{I,Z}$ couples the transport equation of each ionisation stage with the neighbouring stages.

$$\begin{aligned} Q_{I,Z} = & - & (n_e S_{I,Z} + n_e \alpha_{I,Z} + n_H \alpha^{cx}_{I,Z}) n_{I,Z} \\ & + & n_e S_{I,Z-1} n_{I,Z-1} \\ & + & (n_e \alpha_{I,Z+1} + n_H \alpha^{cx}_{I,Z+1}) n_{I,Z+1} \end{aligned} \tag{1.14}$$

$S_{I,Z}$ is the rate coefficient for ionisation of impurity species I in ionisation stage Z, $\alpha_{I,Z}$ is the recombination coefficient for radiative and di-electronic recombination from ionisation stage Z and $\alpha^{cx}_{I,Z}$ is the respective recombination coefficient due to charge exchange.

The sum of the transport equations for all ions (not the neutral!) of one species is a very useful simplification since all source terms cancel except for the ionisation and recombination between the neutral atom and the single ionised ion.

$$\frac{\partial n_I}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r} r \left(D \frac{\partial n_I}{\partial r} - v n_I \right) + Q_{I,1} \tag{1.15}$$

D and v are average values of the transport coefficients for each ion stage weighted with the fractional abundance of each ion stage at a certain flux surface.

1.2 Neoclassical Transport

Two neoclassical code packages are now coupled with STRAHL. In both toroidal rotation of the plasma is not taken into account and equal temperatures of main ion D and impurity I $T_D = T_I = T$ are assumed.

In the first package the neoclassical transport coefficients are calculated with the numerical code NEOART by Arthur Peeters. It solves the set of linear coupled equations for the parallel velocities in arbitrary toroidally symmetric geometry for all collision regimes. The classical fluxes are given by eq. (5.9) and (5.10) in section 5 of Hirshman and Sigmar [Hirshman and Sigmar, 1981]. The equations for the banana plateau contribution are equal to that used by Houlberg [Houlberg et al., 1997]. The Pfirsch-Schlüter contribution is calculated from the coupled equations (6.1-2) and (6.14-15) of Hirshman and Sigmar [Hirshman and Sigmar, 1981], as described in Ref. [Peeters, 2000]. For all contributions a reduced charge state formalism is applied.

In the second package approximative analytical expressions according to Hirshman and Sigmar [Hirshman and Sigmar, 1981] are used. We follow the description given by Fußmann [Fußmann, 1992].

The classical diffusion coefficient and drift velocity is given by

$$D^{CL}_{I,Z} = \frac{1}{(\partial\Psi/\partial r)^2} \left\langle \frac{R^2 B_p^2}{B^2} \right\rangle \frac{m_I k_B T \nu_{ID}}{e^2 Z^2} \tag{1.16}$$

$$v^{CL}_{I,Z} = D^{CL}_{I,Z} Z \left[\frac{d \ln n_D}{dr} - \frac{1}{2} \frac{d \ln T}{dr} \right] \tag{1.17}$$

Here Ψ is the poloidal magnetic flux, B_p is the poloidal magnetic field and ν_{ID} is the binary collision frequency of impurity and main ion. The binary collision frequency of species a with species b is for equal temperatures

$$\nu_{ab} = \frac{4\sqrt{2\pi} e^4}{3} \frac{1}{m_a} \sqrt{\frac{m_a m_b}{m_a + m_b}} \frac{Z_a^2 Z_b^2 \ln\Lambda}{(k_B T)^{3/2}} n_b \tag{1.18}$$

The brackets $\langle\rangle$ denote a flux surface average as defined in equation (1.4). The product of the first two terms in equation (1.16) reduces to $1/B_0^2$ for the large aspect ratio circular cross section case (B_0 is the vacuum toroidal field on axis). The Pfirsch-Schlüter coefficients are

$$\begin{aligned} D^{PS}_{I,Z} & = & \frac{\langle R B_T \rangle^2}{(\partial\Psi/\partial r)^2} \left(\langle B^{-2} \rangle - \langle B^2 \rangle^{-1} \right) \\ & \times & K^{PS} \frac{m_I k_B T \nu_{ID}}{e^2 Z^2} \end{aligned} \tag{1.19}$$

$$v_{I,Z}^{PS} = D_{I,Z}^{PS} Z \Big[\frac{d\ln n_D}{dr} + \frac{H^{PS}}{K^{PS}}\frac{d\ln T}{dr}\Big] \tag{1.20}$$

where the term in the first row of eq. (1.19) reduces to $2q^2/B_0^2$ in the case of large aspect ratio and circular geometry. The factors K^{PS} and H^{PS} are functions of the impurity strength parameter $\alpha = n_I Z_I^2/n_D$ and the main ion collisionality $\nu_D^* = (\nu_{DD} + \nu_{DI})R_0 q/(v_{th}\varepsilon^{3/2})$.

$$\begin{aligned} K^{PS} &= 1 - \frac{0.52\alpha}{0.59 + \alpha + 1.34(\varepsilon^{3/2}\nu_D^*)^{-2}} \\ H^{PS} &= -\frac{1}{2} + \frac{0.29 + 0.68\alpha}{0.59 + \alpha + 1.34(\varepsilon^{3/2}\nu_D^*)^{-2}} \end{aligned} \tag{1.21}$$

The banana-plateau coefficients are

$$D_{I,Z}^{BP} = \frac{\langle RB_T\rangle^2}{(\partial\Psi/\partial r)^2\langle B^2\rangle}\ \frac{k_B T \mu_{ID}^*}{e^2 Z^2 n_I} \tag{1.22}$$

$$v_{I,Z}^{BP} = D_{I,Z}^{BP} Z \Big[\frac{d\ln n_D}{dr} + H^{BP}\frac{d\ln T}{dr}\Big] \tag{1.23}$$

where the term with the flux surface averages in eq.(1.22) reduces to $q^2/(\varepsilon^2 B_0^2)$ in the case of large aspect ratio and circular geometry (ε is the inverse aspect ratio). μ_{ID}^* is the weighted viscosity coefficient μ_{00} of impurity and main ion.

$$\mu_{ID}^* = \frac{m_D n_D \mu_{D00} \cdot m_I n_I \mu_{I00}}{m_D n_D \mu_{D00} + m_I n_I \mu_{I00}} \tag{1.24}$$

The factor H^{BP} in the drift term is connected to the ratio of the viscosity coefficients μ_{01}/μ_{00} of impurity and main ion.

$$H^{BP} = \big(1 - \frac{\mu_{D01}}{\mu_{D00}}\big) - \frac{1}{Z}\big(1 - \frac{\mu_{I01}}{\mu_{I00}}\big) \tag{1.25}$$

All viscosity coefficients are calculated using the expressions given by Kim [Kim et al., 1991]. For the poloidal plasma cross section of ASDEX Upgrade with elongation $\kappa = 1.6$ and average triangularity $\delta = 0.2$ the ratios of the true geometry factor and the large aspect ratio circular geometry approximation are 1.1/1.3 at ρ_{pol} =0.3/0.9 for the CL term, 0.75/0.42 for the PS term, and 0.75/0.44 for the BP term.

1.3 Numerical Solution

The impurity transport equations (1.12) for the different ion stages of one impurity species can only be solved numerically. Similarly, the solution of the transport equation for the sum of all ion stages (1.15) is only known, if the transport coefficients have simple radial dependences, e.g. $D(r) = const.$ and $v(r) = v_a r/a$ or $D(r) = D_0 + D_a(r/a)^2$ and $v(r) = v_a r/a$. Therefore, computer codes, which provide a numerical solution of Eq. (1.12) and (1.15) are indispensable.

A brief description of the preferred numerical schemes shall be given here. We use a matrix notation for the Z transport equations (1.12) of all ion stages of an impurity with atomic number Z.

$$\begin{aligned} \frac{\partial\vec{n}}{\partial t} &= \frac{1}{r}\frac{\partial}{\partial r} r(\hat{D}(r)\frac{\partial\vec{n}}{\partial r} - \hat{v}(r)\vec{n}) - \hat{S}\vec{n} - \hat{R}\vec{n} + \vec{d} \\ &= \hat{D}\frac{\partial^2\vec{n}}{\partial r^2} + \Big(\frac{\hat{D}}{r} + \frac{d\hat{D}}{dr} - \hat{v}\Big)\frac{\partial\vec{n}}{\partial r} - \Big(\frac{\hat{v}}{r} + \frac{d\hat{v}}{dr}\Big)\vec{n} - \hat{S}\vec{n} - \hat{R}\vec{n} + \vec{d} \end{aligned} \tag{1.26}$$

$\vec{n}$ is a vector in the space of of the ionisation stages containing the number densities of the different ionisation stages. $\hat{D}$ and $\hat{v}$ are diagonal matrices with the transport coefficients for each ion stage. $\hat{S}$ and $\hat{R}$ are matrices with the ionisation and recombination rates, i.e. the elements of $\hat{S}$ are the product of n_e and the rate coefficients for electron impact ionisation, while the elements of $\hat{R}$ are the sum of the products of n_e and the rate coefficients for radiative and di-electronic recombination. $\hat{S}$ has a non-zero diagonal and lower diagonal, while $\hat{R}$ has a non-zero diagonal and upper diagonal. $\vec{d}$ has only one non-zero element at $Z = 1$ and describes the source rate by ionisation of neutral impurity atoms.

An unconditionally stable and effective discretisation method for diffusion-convection equations is the Crank-Nicolson scheme. Here, the change of the density during the time step Δt from time point l to l+1, is calculated by taking the average values of densities and spatial derivatives at the time points l and l+1 for the terms, which appear on the right hand side of Eq.(1.26). With the notation

$$\vec{n}^{l+1/2} = \frac{\vec{n}^{l} + \vec{n}^{l+1}}{2} \tag{1.27}$$

the Crank-Nicolson scheme has the following form.

$$\begin{aligned} \frac{\vec{n}^{l+1} - \vec{n}^{l}}{\Delta t} &= \hat{D}\frac{\partial^2 \vec{n}^{l+1/2}}{\partial r^2} + \Big(\frac{\hat{D}}{r} + \frac{d\hat{D}}{dr} - \hat{v}\Big)\frac{\partial \vec{n}^{l+1/2}}{\partial r} - \Big(\frac{\hat{v}}{r} + \frac{d\hat{v}}{dr}\Big)\vec{n}^{l+1/2} \\ &- \hat{S}\vec{n}^{l+1/2} - \hat{R}\vec{n}^{l+1/2} + \vec{d} \end{aligned} \tag{1.28}$$

An equidistant mesh of N radial grid points with radial step size Δr shall be considered. The spatial derivatives at mesh point k in Eq.(1.28) are replaced by the symmetric three point formulas, which have a truncation error quadratic in Δr.

$$\begin{aligned} \Big[\frac{\partial f}{\partial r}\Big]_k &= \frac{f_{k+1} - f_{k-1}}{2\Delta r} + \frac{\Delta r^2}{6}\Big[\frac{\partial^3 f}{\partial r^3}\Big]_k + O(\Delta r^4) \\ \Big[\frac{\partial^2 f}{\partial r^2}\Big]_k &= \frac{f_{k+1} - 2f_k + f_{k-1}}{\Delta r^2} + \frac{\Delta r^2}{12}\Big[\frac{\partial^4 f}{\partial r^4}\Big]_k + O(\Delta r^4) \end{aligned} \tag{1.29}$$

The truncation error of the finite difference formulation can be calculated by expanding the densities and transport coefficients at the different nodes in a Taylor series and forming the difference with the original transport equation. When expanding around $t + \Delta t/2$ the truncation error reads.

$$\begin{aligned} \vec{E}_{CN} = \frac{\Delta r^2}{12}\Big(-\hat{D}\frac{\partial^4 \vec{n}}{\partial r^4} - 2\big(\frac{\hat{D}}{r} + \frac{d\hat{D}}{dr} - \hat{v}\big)\frac{\partial^3 \vec{n}}{\partial r^3} - 2\frac{d^3\hat{D}}{dr^3}\frac{\partial \vec{n}}{\partial r} + 2\frac{d^3\hat{v}}{dr^3}\vec{n}\Big) \\ + \frac{\Delta t^2}{24}\Big(\frac{\partial^3 \vec{n}}{\partial t^3} + 3\big(\hat{S} + \hat{R}\frac{\hat{v}}{r} + \frac{d\hat{v}}{dr}\big)\frac{\partial^2 \vec{n}}{\partial t^2} - 3\big(\frac{\hat{D}}{r} + \frac{d\hat{D}}{dr} - \hat{v}\big)\frac{\partial^3 \vec{n}}{\partial r \partial t^2} - 3\hat{D}\frac{\partial^4 \vec{n}}{\partial r^2 \partial t^2}\Big) \end{aligned} \tag{1.30}$$

The truncation error is quadratic in the radial and temporal step sizes.

We arrive at a linear system of $N \times Z$ algebraic equations, which yield the densities at the new time point. The equation for the density of an ion stage at mesh point k contains the densities of the neighbouring radial mesh points as well as the neighbouring ion stages. Thus, the difference formulation is rather similar to that of a two-dimensional transport equation. The coefficient matrix of the linear system can be ordered such, that it has tridiagonal form plus additional terms that are displaced from the main diagonal by the number of ion stages Z which result from ionisation and recombination.

An iterative solution of the matrix equation has been used by Behringer [Behringer, 1987] and is one of the two implemented solution methods in STRAHL. To this end the full matrix equation is split into Z matrix equations on the number of radial mesh points which are solved with the fast Thomas-algorithm. These Z matrix equations are solved in ascending order of the ion charge and the densities of the lower charge state at the time point $l + 1$ are known for the calculation of the time-centred ionisation rate into the present charge state. The time-centred recombination rate into the present charge state uses the density of the next higher charge state at $l + 1$ as calculated by the preceeding iteration step.

An alternative method can be used in STRAHL, which has been proposed by Lackner et al [Lackner et al., 1982]. Here, the densities and radial derivatives of the transport part are again treated like in the time-centred Crank-Nicolson scheme, however, for the ionisation and recombination part an alternation between an implicit and explicit method is chosen. Thus, there is a time step which calculates the ionisation rate per volume with the density $\vec{n}^{l+1}$ (implicit) and the recombinations with the densities $\vec{n}^{l}$ (explicit)

$$\begin{aligned}\frac{\vec{n}^{l+1}-\vec{n}^{l}}{\Delta t} &= \hat{D}\frac{\partial^2\vec{n}^{l+1/2}}{\partial r^2}+\Big(\frac{\hat{D}}{r}+\frac{d\hat{D}}{dr}-\hat{v}\Big)\frac{\partial\vec{n}^{l+1/2}}{\partial r}-\Big(\frac{\hat{v}}{r}+\frac{d\hat{v}}{dr}\Big)\vec{n}^{l+1/2} \\ &- \hat{S}\vec{n}^{l+1}-\hat{R}\vec{n}^{l}+\vec{d} \qquad (1.31)\end{aligned}$$

followed by a time step where $\vec{n}^{l+1}$ is used for the recombinations and $\vec{n}^{l}$ for the ionisations.

$$\begin{aligned}\frac{\vec{n}^{l+1}-\vec{n}^{l}}{\Delta t} &= \hat{D}\frac{\partial^2\vec{n}^{l+1/2}}{\partial r^2}+\Big(\frac{\hat{D}}{r}+\frac{d\hat{D}}{dr}-\hat{v}\Big)\frac{\partial\vec{n}^{l+1/2}}{\partial r}-\Big(\frac{\hat{v}}{r}+\frac{d\hat{v}}{dr}\Big)\vec{n}^{l+1/2} \\ &- \hat{S}\vec{n}^{l}-\hat{R}\vec{n}^{l+1}+\vec{d} \qquad (1.32)\end{aligned}$$

The method is again unconditionally stable [Lackner et al., 1982] and has the great advantage to lead to Z tridiagonal matrix equations on the number of mesh points, which can very effectively be solved using the Thomas-algorithm. For the first time step Eq.(1.31), the equations are solved in increasing order of the charge state, so that the densities of the lower charge state at the new time point l+1 are known when needed for the calculation of the ionisation source into the present charge state. For the second time step Eq.(1.32), the equations for the charge states are solved in decreasing order since the recombination source with the densities at the new time point are now needed.

The truncation error of the two steps

$$\begin{aligned}\vec{E}_1 &= \vec{E}_{CN}+\frac{\Delta t}{2}(\hat{S}-\hat{R})\frac{\partial\vec{n}}{\partial t} \\ \vec{E}_2 &= \vec{E}_{CN}-\frac{\Delta t}{2}(\hat{S}-\hat{R})\frac{\partial\vec{n}}{\partial t} \qquad (1.33)\end{aligned}$$

are similar to the time-centred scheme, however, additional terms linear in Δt appear, which change sign from step to step. The numerical solution is found to oscillate from time step to time step and has to be taken after an even number of time steps. A detailed treatment of the discretisation scheme (1.31) and (1.32) can be found in section 1.4.

At the inner boundary, the plasma axis, a vanishing density gradient and zero drift velocity are requested. For $r \to 0$ the transport equations reads.

$$\frac{\partial\vec{n}(r=0)}{\partial t}=2\hat{D}\Big[\frac{\partial^2\vec{n}}{\partial r^2}\Big]_{r=0}-\Big(2\Big[\frac{d\hat{v}}{dr}\Big]_{r=0}+\hat{S}+\hat{R}\Big)\vec{n}(r=0)+\vec{d} \qquad (1.34)$$

Here, the second derivative of the densities is replaced by $2(\vec{n}_1-\vec{n}_0)/\Delta r^2$, while the gradient of the drift velocities at $r=0$ has to be approximated with the asymmetric two-point formula. Dirichlet boundary conditions at the outermost grid point are easily incorporated by setting the density of the last grid point to the given value. More commonly, a decay length λ is used as boundary condition. In order to maintain a second order truncation error in the radial step size, this boundary condition is implemented with the symmetric three point formula using the fictitious grid point $N+1$, i.e.

$$\frac{\partial\vec{n}}{\partial r}=\frac{\vec{n}_{N+1}-\vec{n}_{N-1}}{2\Delta r}=-\frac{\vec{n}_N}{\lambda} \qquad (1.35)$$

The densities $\vec{n}_{N+1}$ are then replaced by $\vec{n}_{N-1}-(2\Delta r/\lambda)\vec{n}_N$ in equations (1.28), (1.31) and (1.32) respectively.

The numerical implementation usually uses a non-uniform mesh of radial grid points. Close to the last closed flux surface, strong gradients of electron temperature and density appear, which can have decay lengths as low as a few mm. The electron temperature might rise from a few eV to 1 keV within 2 cm and strong gradients in the densities of the different ionisation stages can be present. Therefore, the radial step size used in the numerical calculations needs to be around 1 mm in the boundary region, while in the central part of the plasma a grid resolution around 1 cm is usually sufficient. It is advantageous to change the radial grid spacing from centre to edge in order to keep the number of radial mesh points as low as possible. The mesh points shall be equidistant in the coordinate

$\rho = \rho(r)$. STRAHL has two options for the grid. Both choices give a higher density of grid points at the plasma edge. For the first option, $\rho = r^k$ with $k \geq 1$ and the radial distance of two neighbouring grid points is proportional to $1/r^{k-1}$. The second option is.

$$\rho = \frac{r}{\Delta r_{centre}} + \frac{r_{edge}}{k+1}\big(\frac{1}{\Delta r_{edge}} - \frac{1}{\Delta r_{centre}}\big)\big(\frac{r}{r_{edge}}\big)^{k+1} \tag{1.36}$$

This choice produces a radial step size Δr which decreases from Δr_{centre} at the axis to Δr_{edge} at the boundary.

$$\Delta r = \Big[\frac{1}{\Delta r_{centre}} + \big(\frac{1}{\Delta r_{edge}} - \frac{1}{\Delta r_{centre}}\big)\big(\frac{r}{r_{edge}}\big)^{k}\Big]^{-1} \tag{1.37}$$

Transformation of the transport equation to ρ yields for the radial derivatives $\partial/\partial r = \rho'\partial/\partial\rho$ and $\partial^2/\partial r^2 = \rho'^2\partial^2/\partial\rho^2 + \rho''\partial/\partial\rho$. Using the symmetric three point formulas for the ρ derivatives leads to an increase of the truncation error as compared to uniform grids [Thompson et al., 1985, Fletcher, 1997], however, the increase is small as long as the change of Δr for two adjacent grid intervals, i.e. $d^2r/d\rho^2 = -\rho''/\rho'^2$, is low. It is appropriate to keep the change of Δr below ±10%.

1.4 Detailed Numerical Scheme

The discretisation of the coupled impurity transport equations (1.12) and the numerical solution scheme as given in equations(1.31,1.32) is described in detail. We rewrite Eq.(1.12) for the ion stage z.

$$\frac{\partial n_z}{\partial t} = \frac{1}{r}\frac{d}{dr}r(D_z(r)\frac{dn_z}{dr} - v_z(r)n_z) - \frac{n_z}{\tau_{||}} + R_z + Q_z \tag{1.38}$$

The parallel loss in the SOL is approximated by a volume loss term with characteristic decay time $\tau_{||}$. The volume sources due to recombination and ionisation between ions with $z' = z \pm 1$ are contained in the reaction term R_z.

$$R_z = n_e(S_{z-1}n_{z-1} + \alpha_z n_{z+1}) - n_z n_e(S_z + \alpha_{z-1}) \tag{1.39}$$

Q_z describes an external source, e.g. fuelling of singly ionised ions by ionisation of neutrals or generation of He^{++} ions by fusion of D and T.

The mesh points shall be equidistant in the coordinate $\rho = \rho(r)$. Transformation of the transport equation to ρ yields with $d\rho/dr = \rho'$.

$$\frac{\partial n_z}{\partial t} = D_z\rho'^2\frac{d^2n_z}{d\rho^2} + \big((\rho'' + \frac{\rho'}{r})D_z + \rho'^2\frac{dD_z}{d\rho} - \rho' v_z\big)\frac{dn_z}{d\rho} - \big(\frac{v_z}{r} + \rho'\frac{dv_z}{d\rho} + \frac{1}{\tau_{||}}\big)n_z + R_z + Q_z \tag{1.40}$$

The change of the density at the radial mesh point i during a time step Δt, i.e. from t to $t + \Delta t$, is calculated by taking the spatial derivatives of the density at time $t + \Delta t/2$. This approach leads to the Crank-Nicolson difference scheme. Denoting the densities at the old time with superscript 0 and the densities at the new time step with 1 the derivatives at the ith mesh point are given by.

$$\begin{aligned}
\frac{dn_z}{dt} &= \frac{n^1_{z,i} - n^0_{z,i}}{\Delta t} \\
\frac{dn_z}{d\rho} &= \frac{n^0_{z,i+1} + n^1_{z,i+1} - n^0_{z,i-1} - n^1_{z,i-1}}{4\Delta\rho} \\
\frac{d^2n_z}{d\rho^2} &= \frac{n^0_{z,i+1} + n^1_{z,i+1} - 2n^0_{z,i} - 2n^1_{z,i} + n^0_{z,i-1} + n^1_{z,i-1}}{2\Delta\rho^2}
\end{aligned} \tag{1.41}$$

The following notation for the derivatives of diffusion coefficient and drift velocity is used

$$\begin{aligned}
\frac{dD_z}{d\rho} &= \frac{D_{z,i+1} - D_{z,i-1}}{2\Delta\rho} = \frac{\Delta D_z}{2\Delta\rho} \\
\frac{dv_z}{d\rho} &= \frac{v_{z,i+1} - v_{z,i-1}}{2\Delta\rho} = \frac{\Delta v_z}{2\Delta\rho}
\end{aligned} \tag{1.42}$$

This yields the discretisation of the transport equation.

$$\begin{aligned} n^1_{z,i} - n^0_{z,i} &= \frac{\Delta t}{2}\frac{D_z\rho'^2}{\Delta\rho^2}(n^0_{z,i+1}+n^1_{z,i+1}-2n^0_{z,i}-2n^1_{z,i}+n^0_{z,i-1}+n^1_{z,i-1}) \\ &+ \frac{\Delta t}{2}\big((\frac{\rho''}{2\Delta\rho}+\frac{\rho'}{2r\Delta\rho})D_z+\frac{\rho'^2}{4\Delta\rho^2}\Delta D_z-\frac{\rho'}{2\Delta\rho}v_z\big)(n^0_{z,i+1}+n^1_{z,i+1}-n^0_{z,i-1}-n^1_{z,i-1}) \\ &- \frac{\Delta t}{2}(\frac{v_z}{r}+\frac{\rho'}{2\Delta\rho}\Delta v_z+\frac{1}{\tau_{||}})(n^0_{z,i}+n^1_{z,i})+\Delta t R_z+\Delta t Q_z \end{aligned} \tag{1.43}$$

With $p = \rho'/(2\Delta\rho)$ and $q = \rho''/(2\Delta\rho)$ the equation reads.

$$\begin{aligned} n^1_{z,i} - n^0_{z,i} &= \frac{\Delta t}{2}4p^2D_z(n^0_{z,i+1}+n^1_{z,i+1}-2n^0_{z,i}-2n^1_{z,i}+n^0_{z,i-1}+n^1_{z,i-1}) \\ &+ \frac{\Delta t}{2}\big((q+\frac{p}{r})D_z+p^2\Delta D_z-pv_z\big)(n^0_{z,i+1}+n^1_{z,i+1}-n^0_{z,i-1}-n^1_{z,i-1}) \\ &- \frac{\Delta t}{2}(\frac{v_z}{r}+p\Delta v_z+\frac{1}{\tau_{||}})(n^0_{z,i}+n^1_{z,i})+\Delta t R_z+\Delta t Q_z \end{aligned} \tag{1.44}$$

The reaction term changes it form each time step according to equations(1.31,1.32). A time step (I) with implicit treatment of ionisation and explicit treatment of recombination is followed by a time step (II) with explicit treatment of ionisation and implicit treatment of recombination.

$$R_z = \begin{cases} n_e(S_{z-1}n^1_{z-1,i}+\alpha_z n^0_{z+1,i}-n^1_{z,i}S_z-n^0_{z,i}\alpha_{z-1}) & \text{(I)} \\ n_e(S_{z-1}n^0_{z-1,i}+\alpha_z n^1_{z+1,i}-n^0_{z,i}S_z-n^1_{z,i}\alpha_{z-1}) & \text{(II)} \end{cases} \tag{1.45}$$

This difference method leads to the solution of a tridiagonal matrix equation for each ion stage on the number of mesh points.

$$an^1_{z,i-1}+(b+b^R)n^1_{z,i}+cn^1_{z,i+1}=d+d^R \tag{1.46}$$

The coefficients $(b+b^R)$ are on the main diagonal, a stands in the lower, c in the upper diagonal and $(d+d^R)$ on the right hand side. The coefficients with superscript R are due to ionisation/recombination reactions.

$$\begin{aligned} a &= -2p^2D_z\Delta t+\frac{\Delta t}{2}\big((q+\frac{p}{r})D_z+p^2\Delta D_z-pv_z\big) \\ b &= 1+4p^2D_z\Delta t+\frac{\Delta t}{2}(\frac{v_z}{r}+p\Delta v_z+\frac{1}{\tau_{||}}) \\ c &= -4p^2D_z\Delta t-a \\ d &= -an^0_{z,i-1}+(2-b)n^0_{z,i}-cn^0_{z,i+1}+\Delta tQ_z \end{aligned} \tag{1.47}$$

$$\begin{aligned} b^R &= \Delta t\begin{cases} n_eS_z & \text{(I)} \\ n_e\alpha_{z-1} & \text{(II)} \end{cases} \\ d^R &= \Delta t\begin{cases} n_e(S_{z-1}n^1_{z-1,i}+\alpha_z n^0_{z+1,i}-n^0_{z,i}\alpha_{z-1}) & \text{(I)} \\ n_e(S_{z-1}n^0_{z-1,i}+\alpha_z n^1_{z+1,i}-n^0_{z,i}S_z) & \text{(II)} \end{cases} \end{aligned} \tag{1.48}$$

For case (I), the m_z matrix equations are solved in ascending order of z and the density $n^1_{z-1,i}$ of the next lower stage at the new time point is known when needed in the d^R term of the equation for $n^1_{z,i}$. For case (II), $n^1_{z+1,i}$ is needed and the matrix equations are solved in descending order of z.

At the axis $r = 0$, the density gradient and the drift velocity is zero. Expanding density and drift velocity in the vicinity of $r = 0$ yields the transport equation for $r \to 0$.

$$\frac{\partial n_z}{\partial t} = 2D_z(0)[\frac{d^2n}{dr^2}]_{r=0}-2[\frac{dv_z}{dr}]_{r=0}n_z(0)+R_z+Q_z \tag{1.49}$$

The derivative of v_z at $r = 0$ is replaced by $(v_{z,1} - v_{z,0})/\Delta r$ and the discretisation is.

$$\begin{aligned} n^1_{z,0} - n^0_{z,0} &= 2\Delta t \frac{D_z}{\Delta r^2}(n^0_{z,1} + n^1_{z,1} - n^0_{z,0} - n^1_{z,0}) \\ &- \frac{\Delta t}{2}\frac{v_{z,1} - v_{z,0}}{\Delta r}(n^0_{z,0} + n^1_{z,0}) + \Delta t R_z + \Delta t Q_z \end{aligned} \tag{1.50}$$

The coefficients are.

$$\begin{aligned} a &= 0 \\ b &= 1 + 2\Delta t \frac{D_z}{\Delta r^2} + \frac{\Delta t}{2}\frac{v_{z,1} - v_{z,0}}{\Delta r} \\ c &= -2\Delta t \frac{D_z}{\Delta r^2} \\ d &= (2-b)n^0_{z,0} - c n^0_{z,1} + \Delta t Q_z \end{aligned} \tag{1.51}$$

At the outermost radial mesh point k, the density shall decay with decay length λ.

$$\frac{dn_z}{dr} = -\frac{n_{z,k}}{\lambda} \tag{1.52}$$

The boundary condition defines the density at the virtual mesh point k+1 in terms of the densities at k and k-1

$$\frac{dn_z}{dr} = p(n_{z,k+1} - n_{z,k-1}) = -\frac{n_{z,k}}{\lambda} \qquad n_{z,k+1} = n_{z,k-1} - \frac{n_{z,k}}{p\lambda} \tag{1.53}$$

Eq.(1.44) is modified for the outermost grid point k by replacing the densities at k+1 with Eq.(1.53). Furthermore, derivatives of D_z and v_z are neglected.

$$\begin{aligned} n^1_{z,k} - n^0_{z,k} &= \frac{\Delta t}{2} 8p^2 D_z(-(1 + \frac{1}{2p\lambda})(n^0_{z,k} + n^1_{z,k}) + n^0_{z,k-1} + n^1_{z,k-1}) \\ &- \frac{\Delta t}{2}\frac{1}{p\lambda}\big((q + \frac{p}{r})D_z - p v_z\big)(n^0_{z,k} + n^1_{z,k}) \\ &- \frac{\Delta t}{2}(\frac{v_z}{r} + \frac{1}{\tau_{||}})(n^0_{z,k} + n^1_{z,k}) + \Delta t R_z + \Delta t Q_z \end{aligned} \tag{1.54}$$

The coefficients are.

$$\begin{aligned} a &= -4p^2 D_z \Delta t \\ b &= 1 + 4p^2 D_z \Delta t(1 + \frac{1}{2p\lambda}) + \frac{\Delta t}{2}\frac{1}{p\lambda}\big((q + \frac{p}{r})D_z - p v_z\big) + \frac{\Delta t}{2}(\frac{v_z}{r} + \frac{1}{\tau_{||}}) \\ c &= 0 \\ d &= -a n^0_{z,k-1} + (2-b)n^0_{z,k} + \Delta t Q_z \end{aligned} \tag{1.55}$$

For the hard boundary case $n_{z,edge} = 0$, the coefficients are.

$$\begin{aligned} a &= 0 \\ b + b^R &= 1 \\ c &= 0 \\ d + d^R &= 1 \end{aligned} \tag{1.56}$$

1.5 Neutral Impurities

The transport equations (1.12) are only solved for the ions while the neutrals only act as source for the first ionisation stage. The neutrals are assumed to enter the plasma with a given uniform radial speed v_0 and the total number of neutrals entering the plasma per unit time shall be given by Φ.

$$\Phi = 4\pi^2 R r^{edge} v_0 n_0(r^{edge}). \tag{1.57}$$

Due to ionisation, the density of neutral atoms decays with decreasing radius

$$n_0(r) = n_0(r^{edge}) \frac{r^{edge}}{r} \exp\bigl(- \int_r^{r^{edge}} \frac{n_e S_0}{v_0} dr\bigr) \tag{1.58}$$

and the radial distribution of the ionisation source from neutrals to singly ionised ions is

$$Q_{0\rightarrow1}(r) = n_0(r) n_e(r) S_0(r). \tag{1.59}$$

The source distribution $Q_{0\rightarrow1}(r)$ is needed for the solution of the transport equations (1.12) and is calculated from equations (1.57) and (1.58). Volume integration of the source distribution yields the total number of ionised neutrals per unit time

$$\langle Q_{0\rightarrow1} \rangle = 4\pi^2 R \int_0^{r^{edge}} n_0(r) n_e(r) S_0(r) r dr \tag{1.60}$$

It is assumed that the plasma is sufficiently hot and dense to cause complete ionisation of the neutrals. Thus, the total number of ionised neutrals per unit time equals the total number of neutrals entering the plasma per unit time Φ. However, the mesh points at the edge are usually not sufficently dense and the numerically integrated total source $\langle Q_{0\rightarrow1} \rangle$ might be substantially different from Φ. Thus, we multiply the source distribution $Q_{0\rightarrow1}(r)$ with the factor $\Phi / \langle Q_{0\rightarrow1} \rangle$ (which is 1 for correct numerical integration) to insure the correct particle balance even for very strongly radially decaying profiles of the neutral impurity.

Chapter 2

Input Files

Every STRAHL run needs several input files to specify the parameters of the plasma background (e.g. electron density, electron temperature) the geometry of the plasma, the transport parameters, and the atomic data for ionisation, recombination and emission. In this chapter every input file is described.

All path names (directory names) are relative to the directory where STRAHL is executed, which is called the current directory. The needed files are located in the current directory (./) or in sub-directories of the current directory (./sub-directory). All input files are in ASCII format.

2.1 Plasma Background

The parameters for the plasma background are in the directory ./nete. The filename is **./nete/ppnnnnn.i** where **nnnnn** is a 5-digit number (shot number) and **i** is a 1-digit number (index). The program reads all parameters with a free format read statement in Fortran77: read(unit,*). Thus, it is not necessary to obey strict formatting rules. Character constants have to be given within single quotes (e.g. `'text'`).

Each item in the data block is preceded by a line which starts with `cv␣` and the programm scans through the input file to search for the next appearance of `cv␣`. When it finds the next `cv␣` it starts to read the data values from the next line. The advantage of this method is the possibility to include descriptions in the input file which help to understand the structure or memorize changes. Old input data can become comments by replacing `cv␣` by anything else in the preceding line. When replacing `cv␣` by `cv#` the next data block is read from the input file with fixed name **./ext_parameter.dat** in the current directory. This alternative input might simplify scans of different STRAHL runs which are just different by a few parameters.

The pp-file has the following structure:

- data block for the electron density profiles
- data block for the electron temperature profiles
- data block for the ion temperature profiles [optional]
- data block for the neutral hydrogen density [optional]

An example for a data block describing the electron density for the time points t=2.0s and t=4.0s looks like:

```
cv   time-vector
     2
     2.0  4.0

cv   ne function
```

```
     ’interp’

cv   radial coordinate
     ’poloidal rho’

cv   # of interpolation points
     19

cv   radial grid for ne interpolation
     0.0000 0.1000 0.2000 0.3000 0.4000 0.5000 0.6000
     0.7000 0.8000 0.8200 0.8400 0.8600 0.8800 0.9000
     0.9200 0.9400 0.9600 0.9800 1.0000

cv   ne [cm**-3]
     5.000E+13  1.000E+00  9.752E-01  9.076E-01  8.088E-01  6.977E-01
     6.019E-01  5.464E-01  5.240E-01  5.043E-01  4.959E-01  4.826E-01
     4.673E-01  4.477E-01  4.195E-01  3.827E-01  3.430E-01  3.080E-01
     2.747E-01  2.355E-01
     6.000E+13  1.000E+00  9.752E-01  9.076E-01  8.088E-01  6.977E-01
     6.019E-01  5.464E-01  5.240E-01  5.043E-01  4.959E-01  4.826E-01
     4.673E-01  4.477E-01  4.195E-01  3.827E-01  3.430E-01  3.080E-01
     2.747E-01  2.355E-01

cv   ne decay length [cm] in rho_volume
     4.2  4.2
```

Each block contains the following data:

- number of time points followed by the time points
- type of parametrization
- radial coordinate of parametrization
- one profile for every time point

The code uses the profile of the first time point for times which are before the first time point and the profile of the last time point for times which are after the last time point. For the times which are inside the specified time intervall a linear interpolation in time is used. The different types of possible parametrizations are identical for the electron density, the electron temperature, and the ion temperature.

In the radial range outside the last closed flux surface (LCFS) the profile is just described by an exponential decay length λ (unit: cm). The profile outside the LCFS with radius r_{LCFS} is then given by

$$y(r) = y(LCFS) * \exp\,[-\ \frac{r - r_{LCFS}}{\lambda}]$$

For the profiles inside the LCFS four different functions are accepted. Each parametrization can be used with ρ_{pol} or r for the radial coordinate. The formulas are given here with the dummy ρ. Each profile contains a scale factor y_0 which transforms the mathematical function into a physical qunatity. The scale factor for the electron density has the unit [cm^{-3}] while the scale factor for the temperatures has the unit [eV].

- `’interp’`
  causes quadratic interpolation between values at given radial positions. The data block looks like:

```
cv   times
     nt
     t(1) .... t(nt)

cv   function
     ’interp’

cv   radial coordinate
     ’poloidal rho’  or  ’volume rho’

cv   # of interpolation points
     n_rho

cv   radial grid for ne interpolation
     rho(1) .... rho(n_rho)

cv   ne[cm**-3] / Te[eV] / Ti[eV]
     y0(t(1))   y(1,t(1)) ... y(n_rho,t(1))
     ........
     y0(t(nt))   y(1,t(nt)) ... y(n_rho,t(nt))

cv   decay
     lambda(t(1)) .... lambda(t(nt))
```

- `’interpa’`
  also causes quadratic interpolation between values at given radial positions, however, the interpolation is used for the whole radaial grid, i.e. also outside the LCFS. The data block looks the same as for the previous option. The only difference is, that the decay length block is left out.

```
cv   times
     nt
     t(1) .... t(nt)

cv   function
     ’interpa’

cv   radial coordinate
     ’poloidal rho’  or  ’volume rho’

cv   # of interpolation points
     n_rho

cv   radial grid for ne interpolation
     rho(1) .... rho(n_rho)

cv   ne[cm**-3] / Te[eV] / Ti[eV]
     y0(t(1))   y(1,t(1)) ... y(n_rho,t(1))
     ........
     y0(t(nt))   y(1,t(nt)) ... y(n_rho,t(nt))
```

- `’exppol0’`
  exponential of an even polynom

$$y(\rho) = y_0 \exp\left[p_0\rho^2 + p_1\rho^4 + p_2\rho^6 + p_3\rho^8\right]$$

```
cv   times
     nt
     t(1) .... t(nt)

cv   function
     ’exppol0’

cv   radial coordinate
     ’poloidal rho’  or  ’volume rho’

cv   ne[cm**-3] / Te[eV] / Ti[eV]
     y0(t(1))   p0(t(1))    p1(t(1))   p2(t(1))   p3(t(1))
     ......
     y0(t(nt))   p0(t(nt))  p1(t(nt))  p2(t(nt))  p3(t(nt))

cv   decay
     lambda(t(1)) .... lambda(t(nt))
```

- ’exppol1’
  exponential of a polynom with zero slope on the axis

$$y(\rho) = y_0 \exp\left[p_0\rho^2 + p_1\rho^3 + p_2\rho^4 + p_3\rho^5\right]$$

```
cv   times
     nt
     t(1) .... t(nt)

cv   function
     ’exppol1’

cv   radial coordinate
     ’poloidal rho’  or  ’volume rho’

cv   ne[cm**-3] / Te[eV] / Ti[eV]
     y0(t(1))   p0(t(1))    p1(t(1))   p2(t(1))   p3(t(1))
     ......
     y0(t(nt))   p0(t(nt))  p1(t(nt))  p2(t(nt))  p3(t(nt))

cv   decay
     lambda(t(1)) .... lambda(t(nt))
```

- ’ratfun’
  rational function of the form

$$y(\rho) = y_0\big((1-p_0)(1-\rho^{p_1})^{p_2} + p_0\big)$$

```
cv   times
     nt
     t(1) .... t(nt)

cv   function
     ’ratfun’

cv   radial coordinate
```

```
          'poloidal rho'  or  'volume rho'

     cv   ne[cm**-3] / Te[eV] / Ti[eV]
          y0(t(1))   p0(t(1))    p1(t(1))   p2(t(1))   p3(t(1))
          ......
          y0(t(nt))   p0(t(nt))  p1(t(nt))  p2(t(nt))  p3(t(nt))

     cv   decay
          lambda(t(1)) .... lambda(t(nt))
```

The parameter p_3 is not relevant but must be given in the data block.

The ion temperature T_I is just needed when calculations of neo-classical transport are performed. If T_I is not needed or if T_I is equal T_e the number of time points for the ion temperature must be set to zero and the rest of the profile information can be left out. STRAHL assumes equal temperature of electrons and ions in this case.

Neutral hydrogen density is needed for the calculation of charge exchange(CX) with thermal neutral hydrogen. The inclusion of CX is optional. It can be set in **./Xx.atomdat** as discussed below. When CX is switched on, the neutral hydrogen density has to be specified in the pp-file. The only possible parametrization is quadratic interpolation for the *whole* radial grid. The input block looks like:

```
cv   times
     nt
     t(1) .... t(nt)

cv   function
     'interpa'

cv   radial coordinate
     'poloidal rho'  or  'volume rho'

cv   # of interpolation points
     n_rho

cv   radial grid for interpolation
     rho(1) .... rho(n_rho)

cv   neutral hydrogen density [cm**-3]
     y0(t(1))   y(1,t(1)) ... y(n_rho,t(1))
     ........
     y0(t(nt))   y(1,t(nt)) ... y(n_rho,t(nt))
```

2.2 Geometry

The plasma geometry is assumed to be independent of time. The parameters for the plasma geometry are in the directory ./nete. The filename is **./nete/grid_nnnnn.i** where **nnnnn** is a 5-digit number (shot number) and **i** is a 1-digit number (index). The program reads all parameters with a free format read statement in Fortran77: read(unit,*). Each item in the data block is preceded by a line which starts with `cv␣` (see 2.1). The grid file gives some flux functions or flux surface averages on a common mesh of flux surfaces. Most quantites are needed for the calculation of neoclassical transport.

The quantities which have to be specified on the mesh inside and outside the LCFS are the following:

- polidal flux label ρ_{pol} (defined inside and outside LCFS):

$$\rho_{pol} = \sqrt{\frac{\Psi - \Psi_{axis}}{\Psi_{LCFS} - \Psi_{axis}}}$$

- r is the radius of a circular torus with same volume V as the flux surface contour (unit = [cm]). Inside the LCFS r is well defined:

$$r = \sqrt{\frac{V}{2\pi^2 R_{axis}}}$$

  Outside the LCFS r is not defined. For diverted ASDEX Upgrade discharges I blow up the last close flux surface to match for a given ρ_{pol} the the mean value and the difference of the major radii of the true flux surface at the height of the magnetic axis, i.e. for $Z = Z_{mag}$.

- major radius R for z=0 at low field side (unit= [cm])

- major radius R for z=0 at high field side (unit = [cm]

The first two flux surface labels ρ_{pol} and r are used to map from the r to the ρ_{pol} grid which is usualy taken by the other tokamak diagnostics. The ρ_{pol} grid is not essential for the code if you do not use ρ_{pol} in your density/temperature profile definition. The major radius R at z=0 is only used in the plotting routines to calculate line integrals of the radiation at mid plane.

All the following quantities are just needed for the calculation of neo-classical transport and may be set to zero if this feature is not needed. Many of these quantities are flux surface averages. The definition of the flux surface average was given in eq. (1.4). An alternative definition of the identical quantity is the following expression

$$\langle A \rangle = \frac{\oint A dl_p / B_p}{\oint dl_p / B_p}$$

where dl_p is the length along the flux surface in poloidal direction, B_p is the poloidal magnetic field and the integration is over one poloidal turn. It is only defined inside the LCFS.

NEOART uses a fourier expansion of some quantities in the generalized poloidal angle Θ. This angle is defined as

$$\Theta = 2\pi \frac{\int_0^{l_p} B dl'_p / B_p}{\oint B dl'_p / B_p}$$

We need to specify the following quantities:

- the loop voltage

- the safety factor q

- the fraction of circulating particles f_c

- $\oint \frac{dl_p}{B_p}$. This quantity is also related to $d\Psi/dr = 2\pi r R_{axis} / \oint \frac{dl_p}{B_p}$

- the flux surface averages:
  $\langle B \rangle$, $\langle B^2 \rangle$, $\langle 1/B^2 \rangle$, $\langle R B_T \rangle$, $\langle R^2 B_p^2 / B^2 \rangle$, $\langle 1/R^2 \rangle$

- the fourier coefficients:
  $\langle \cos(m\Theta) B^2 \rangle$, $\langle \sin(m\Theta) B^2 \rangle$
  $\langle \cos(m\Theta) B \ln B \rangle$, $\langle \sin(m\Theta) B \ln B \rangle$
  The maximum order of m can be set in the header part of the grid-file.

The fraction of circulating particles f_c can be calculated with:

$$f_c = \frac{3\langle B^2\rangle}{4}\int_0^{1/B_{max}} \frac{\lambda d\lambda}{\langle\sqrt{1-\lambda B}\,\rangle}$$

For circular concentric flux surfaces the above quantities can be expressed analytically as a function of the major radius of the magnetic axis R_0, the toroidal field at the magnetic axis $B_{t,0}$, the safety factor q, and the inverse aspect ratio of the flux surface $\epsilon = r/R_0$. The toroidal, poloidal and total field vary as:

$$B_p = \frac{B_{p,0}}{1+\epsilon\cos\theta}, \quad B_t = \frac{B_{t,0}}{1+\epsilon\cos\theta}, \quad B = \frac{B_0}{1+\epsilon\cos\theta}$$

with

$$B_{p,0} = \frac{B_{t,0}\epsilon}{q\sqrt{1-\epsilon^2}} \quad \text{and} \quad B_0 = B_{t,0}\sqrt{1+\frac{\epsilon^2}{q^2(1-\epsilon^2)}}$$

One finds for the flux surface averages:

$$\oint \frac{dl_p}{B_p} = \frac{2\pi r}{B_{p,0}}, \quad \langle B\rangle = B_0, \quad \langle B^2\rangle = \frac{B_0^2}{\sqrt{1-\epsilon^2}}, \quad \langle 1/B^2\rangle = \frac{1}{B_0^2}(1+\frac{3}{2}\epsilon^2)$$

$$\langle RB_t\rangle = R_0 B_{t,0}, \quad \langle B_p^2 R^2/B^2\rangle = \frac{B_{p,0}^2 R_0^2}{B_0^2}(1+\frac{3}{2}\epsilon^2), \quad \langle 1/R^2\rangle = \frac{1}{R_0^2}\frac{1}{\sqrt{1-\epsilon^2}}$$

$$\langle \sin(m\theta)B^2\rangle = 0, \quad \langle \cos(m\theta)B^2\rangle = \frac{B_0^2}{\sqrt{1-\epsilon^2}}\Big(\frac{\sqrt{1-\epsilon^2}-1}{\epsilon}\Big)^m$$

$$\langle \sin(m\theta)B\ln B\rangle = 0, \quad \langle \cos(m\theta)B\ln B\rangle = \frac{B_0}{m}\Big(\frac{\sqrt{1-\epsilon^2}-1}{\epsilon}\Big)^m$$

The program `grid_circular` prepares a geometry file for circular concentric flux surfaces assuming a parabolic profile for the safety factor q, where the values of q on axis and at the LCFS are specified by the user. For a q-profile which is given by: $q(r) = q_0(1+\alpha\epsilon^2)$ the polidal flux label ρ_{pol} is given by.

$$\rho_{pol} = \sqrt{\frac{\text{atanh}(\gamma) - \text{atanh}(\gamma * \sqrt{1-\epsilon^2})}{\text{atanh}(\gamma) - \text{atanh}(\gamma * \sqrt{1-\epsilon_{LCFS}^2})}} \quad \text{with} \quad \gamma = \sqrt{\frac{\alpha}{1+\alpha}}$$

An example of a geometry input file prepared with `grid_circular` is given on the following pages. It includes small headers for all quantities and shows all the units which have to be used.

```
cv  rho volume(LCFS)[cm]   R_axis[cm]   U_loop[V]    time[s]
            50.0            150.0      0.5000       2.0000


cv  number of grid points  points up to separtrix  fourier coefficients
             28                     21                       3


cv  sqrt( (Psi-Psi_ax) / (Psi_sep - Psi_ax) )
   0.00000   0.06659   0.13269   0.19786   0.26169   0.32384   0.38403
   0.44207   0.49783   0.55125   0.60231   0.65104   0.69752   0.74182
   0.78404   0.82429   0.86269   0.89935   0.93439   0.96790   1.00000
   1.00448   1.00893   1.01335   1.01775   1.02212   1.02646   1.03078


cv     rho volume / rho_volume(LCFS)
   0.00000   0.05000   0.10000   0.15000   0.20000   0.25000   0.30000
   0.35000   0.40000   0.45000   0.50000   0.55000   0.60000   0.65000
   0.70000   0.75000   0.80000   0.85000   0.90000   0.95000   1.00000
   1.00714   1.01429   1.02143   1.02857   1.03571   1.04286   1.05000


cv   major radius low field side / R_axis
   1.00000   1.01667   1.03333   1.05000   1.06667   1.08333   1.10000
   1.11667   1.13333   1.15000   1.16667   1.18333   1.20000   1.21667
   1.23333   1.25000   1.26667   1.28333   1.30000   1.31667   1.33333
   1.33571   1.33810   1.34048   1.34286   1.34524   1.34762   1.35000


cv   major radius high field side / R_axis
   1.00000   0.98333   0.96667   0.95000   0.93333   0.91667   0.90000
   0.88333   0.86667   0.85000   0.83333   0.81667   0.80000   0.78333
   0.76667   0.75000   0.73333   0.71667   0.70000   0.68333   0.66667
   0.66429   0.66190   0.65952   0.65714   0.65476   0.65238   0.65000


cv  safety factor
   1.00000   1.00500   1.02000   1.04500   1.08000   1.12500   1.18000
   1.24500   1.32000   1.40500   1.50000   1.60500   1.72000   1.84500
   1.98000   2.12500   2.28000   2.44500   2.62000   2.80500   3.00000
```

```
cv  fraction of circulating particles
   1.00000   0.81250   0.73624   0.67868   0.63095   0.58960   0.55285
   0.51965   0.48928   0.46127   0.43526   0.41097   0.38821   0.36680
   0.34660   0.32750   0.30940   0.29223   0.27591   0.26038   0.24560

cv  Integral( dl_p / B_p) [m/T]
   4.71238   4.73529   4.80396   4.91828   5.07805   5.28299   5.53274
   5.82685   6.16481   6.54599   6.96971   7.43518   7.94154   8.48782
   9.07297   9.69584  10.35518  11.04964  11.77776  12.53799  13.32864


cv  < B_total > [T]
   2.00000   2.00028   2.00107   2.00229   2.00382   2.00552   2.00724
   2.00888   2.01036   2.01163   2.01266   2.01346   2.01403   2.01442
   2.01463   2.01471   2.01467   2.01455   2.01436   2.01412   2.01384


cv  < B_total**2 > [T**2]
   4.00000   4.00166   4.00650   4.01420   4.02426   4.03614   4.04932
   4.06336   4.07796   4.09295   4.10825   4.12390   4.13998   4.15662
   4.17396   4.19217   4.21141   4.23182   4.25355   4.27676   4.30157

cv  < 1./B_total**2 > [1/T**2]
   0.25000   0.25004   0.25015   0.25036   0.25071   0.25122   0.25192
   0.25285   0.25403   0.25546   0.25715   0.25911   0.26132   0.26379
   0.26650   0.26946   0.27265   0.27607   0.27972   0.28359   0.28767

cv  < R B_T > [m*T]
   3.00000   3.00000   3.00000   3.00000   3.00000   3.00000   3.00000
   3.00000   3.00000   3.00000   3.00000   3.00000   3.00000   3.00000
   3.00000   3.00000   3.00000   3.00000   3.00000   3.00000   3.00000

cv  < R**2 B_p**2/B**2 > [m**2]
  0.00E+00  6.19E-04  2.41E-03  5.17E-03  8.64E-03  1.25E-02  1.64E-02
  2.03E-02  2.37E-02  2.68E-02  2.94E-02  3.15E-02  3.31E-02  3.44E-02
  3.52E-02  3.58E-02  3.61E-02  3.63E-02  3.63E-02  3.62E-02  3.60E-02

cv  < 1/R**2 > [1/m**2]
  4.44E-01  4.45E-01  4.45E-01  4.45E-01  4.45E-01  4.46E-01  4.47E-01
  4.48E-01  4.48E-01  4.50E-01  4.51E-01  4.52E-01  4.54E-01  4.55E-01
  4.57E-01  4.59E-01  4.61E-01  4.63E-01  4.66E-01  4.69E-01  4.71E-01
```

```
cv  <cos (m theta) B_total**2> [T**2]
  0.00E+00 -3.33E-02 -6.68E-02 -1.00E-01 -1.34E-01 -1.68E-01 -2.03E-01
 -2.38E-01 -2.73E-01 -3.09E-01 -3.45E-01 -3.81E-01 -4.18E-01 -4.56E-01
 -4.94E-01 -5.32E-01 -5.72E-01 -6.12E-01 -6.53E-01 -6.95E-01 -7.38E-01
  0.00E+00  2.78E-04  1.11E-03  2.51E-03  4.48E-03  7.03E-03  1.02E-02
  1.39E-02  1.83E-02  2.33E-02  2.89E-02  3.52E-02  4.22E-02  5.00E-02
  5.84E-02  6.76E-02  7.77E-02  8.85E-02  1.00E-01  1.13E-01  1.27E-01
  0.00E+00 -2.32E-06 -1.86E-05 -6.28E-05 -1.50E-04 -2.93E-04 -5.10E-04
 -8.15E-04 -1.22E-03 -1.76E-03 -2.43E-03 -3.26E-03 -4.27E-03 -5.48E-03
 -6.91E-03 -8.59E-03 -1.05E-02 -1.28E-02 -1.54E-02 -1.84E-02 -2.17E-02

cv  <sin (m theta) B_total**2> [T**2]
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00

cv  <cos (m theta) B_total ln(B_total)> [Tln(T)]
  0.00E+00 -1.67E-02 -3.34E-02 -5.01E-02 -6.69E-02 -8.37E-02 -1.01E-01
 -1.18E-01 -1.35E-01 -1.52E-01 -1.69E-01 -1.86E-01 -2.03E-01 -2.21E-01
 -2.38E-01 -2.56E-01 -2.74E-01 -2.91E-01 -3.09E-01 -3.27E-01 -3.46E-01
  0.00E+00  6.95E-05  2.78E-04  6.27E-04  1.12E-03  1.75E-03  2.52E-03
  3.44E-03  4.51E-03  5.72E-03  7.09E-03  8.60E-03  1.03E-02  1.21E-02
  1.41E-02  1.63E-02  1.86E-02  2.11E-02  2.37E-02  2.66E-02  2.96E-02
  0.00E+00 -3.86E-07 -3.09E-06 -1.04E-05 -2.48E-05 -4.86E-05 -8.43E-05
 -1.34E-04 -2.01E-04 -2.88E-04 -3.97E-04 -5.30E-04 -6.92E-04 -8.85E-04
 -1.11E-03 -1.38E-03 -1.68E-03 -2.03E-03 -2.43E-03 -2.88E-03 -3.39E-03

cv  <sin (m theta) B_total ln(B_total)> [Tln(T)]
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00  0.00E+00
```

2.3 Atomic Data

The atomic data are read from files which contain the according rates or emissivity coefficients in tabulated form. Most of these data are stored according to ADAS conventions. Since the ADAS tables are written in fixed format, editing of these files has to be done with great care. The interface between the data tables and STRAHL is a file named **./Xx.atomdat**. It is located in the current directory and **Xx** is the element symbol (e.g. Ar, Ne). For elements with a one character symbol, like C and O, **C_** and **O_** are used.

The first two lines of **./Xx.atomdat** contain some switches:

- Bremsstrahlung due to electron scattering at the main ions is (not) calculated.
- Emission in the spectral range of Soft X-ray diagnostic is (not) calculated.
- Spectral emissivity of bremsstrahlung for one wavelegth is (not) calculated.
- Charge Exchange with thermal neutral hydrogen is (not) calculated.
- Line emission of special lines (diagnostic lines) is (not) calculated.

Then the names of the needed atomic data files are specified. The names are given in a block which starts and ends with `cc␣`. Only the first fifteen characters of the names are read by the code. Each name is given after a three character label followed by a colon (e.g. `acd:`). The label tells the program which data set is given in this line. Data sets which are not needed for the current setting of the switches do not have to be specified. The meaning of the labels are:

- `acd:` rate coefficients for recombination
- `scd:` rate coefficients for ionisation
- `prb:` power loss coefficients for continuum radiation
- `plt:` power loss coefficients for line radiation
- `ccd:` rate coefficients for thermal charge exchange
- `prc:` power loss coefficients for thermal charge exchange contiuum radiation
- `pls:` power loss coefficients for line radiation in SXR range
- `prs:` power loss coefficients for continuum radiation in SXR range
- `fis:` sensitivity of Soft X-ray diagnostic as function of photon energy
- `brs:` spectral power loss coefficients for bremsstrahlung

All these data files are in the sub-directory ./atomdat/newdat.

In the diagnostic line section of **./Xx.atomdat** the data line identifier changes from `cc␣` to `cd␣`. In the first line there are switches which indicate whether emission of diagnostic lines following excitation, recombinaion and/or charge exchange with thermal neutral hydrogen is considered. The number of diagnostic lines is given in the next line. Then follows the block of diagnostic lines which contains the information to choose the emission lines from specific files.

- the charge state of the ion
- the central wavelength λ of the wavelength window in Angstrøm
- the half width of the wavelength window $\Delta\lambda$ in Angstrøm
- a three character file name identifier

Some of the information in this data block is needed to construct the name of the file which contains the photon emissivity coefficients (pec-files). The construction of the file name is somewhat complicated and is explained using an example of a CIII line. The file name is build from the nuclear charge of the element (6), the ion charge (`2`), the file name identifier (e.g. `xyz`) and additional information about the directory path in a file called **./pec_files** in the current directory. In **./pec_files** the code goes to line number 6 (= nuclear charge of C) and reads the main directory (e.g. `/usr/home/strahl/atomdat`) and the path name for the subdirectory of the element (e.g. `pec96#c`). From this information the code looks for a file with one of the following two names:

- `/usr/home/strahl/atomdat/adf15/pec96#c/pec96#c_xyz#c2.dat`
- `/usr/home/strahl/atomdat/adf15/pec96#c/xyz#c2.dat`

The first name follows strictly the ADAS conventions while the second name omits the directory name in the file name. The `c2` after the last # is constructed from the character after # in `pec96#c` and the charge of the ion (`2`). Please note the added `adf15` in the directory path.

Having found the right file, the program looks for all lines which fall into the specified wavelength window $\lambda \pm \Delta\lambda$ and adds up all the selected contributions (excitation, recombination, CX) of all lines. Thus, it is possible to add several lines of a pec-file which are not resolved by the spectrometer. Note that the wavelengths in the pec-files are sometimes not the same as the measured wavelengths.

On the next page **./C_atomdat** is shown as an example.

```
cv   main ion brems       SXR     spectral brems      thermal CX
            1              1            1                 0

cv  diagnostic lines
            1

cc begin atomicData
acd:acd96_c.dat          recombination
scd:scd96_c.dat          ionisation
prb:prb96_c.dat          continuum radiation
plt:plt96_c.dat          line radiation
ccd:ccd89_c.dat          thermal charge exchange
prc:prc89_c.dat          thermal charge exchange contiuum radiation
pls:plsx1_c.dat          line radiation in SXR range
prs:prsx1_c.dat          continuum radiation in SXR range
fis:sxrfil1.dat          sensitivity of SXR
brs:brs10400.dat         spectral bremsstrahlung
cc end atomic Data

********************Diagnostic Lines********************

cd     excitation     recombination   charge exchange
         1                0               0

cd   # of lines
         10

cd   charge of ion  wavelength(A)  half width of window(A)  file extension
         1             903.7                   3.              'pju'
         2             977.0                   3.              'pju'
         2            1175.7                   3.              'pju'
         2            4650.1                   3.              'pju'
         3            1549.1                   3               'pju'
         3             312.4                   3.              'pju'
         3             384.1                   3.              'pju'
         3             419.7                   3.              'pju'
         4              40.3                   0.1             'pju'
         5              33.8                   1.              'pju'
```

2.4 Main Parameter File

The main parameter file is located in the sub-directory ./param_files and has arbitrary name. At run time STRAHL asks for the name of the main parameter file. It contains information about the anomalous transport, the impurity source and recycling and some code specific data, like the time step evolution. We now go through an example file line by line. The complete example is given at the end of this section.

```
E L E M E N T
cv     element   atomic weight(amu)  energy of neutrals(eV)
       'C_'           12.             1.0

cv     main ion:  atomic weight(amu)    charge
```

```
                                  2.                  1.
```

The first line contains the element symbol, the atomic weight [amu] and the energy of the neutrals [eV] when starting at the wall. The element symbol is just used to construct the file name of **./Xx.atomdat** and the same name conventions apply here. Then follows the atomic weight [amu] of the main plasma ion and the charge.

```
G R I D - F I L E
cv    shot      index
     99999        0
```

The next line gives the number and the index of the geometry file. Here the geometry is taken from **./nete/grid_99999.0**.

```
G R I D   P O I N T S  A N D  I T E R A T I O N
cv     K      number of grid points   dr_center(cm)  dr_edge(cm)
      6.0             101                  2.0           0.1

      max. iterations     stop iteration
cv     at fixed time    if change below(%)
            100               0.02
```

Then follows the grid parameters. STRAHL uses a grid which is not equidistant in r. You may choose from the following two options (see chapter 1):

- Set `dr_center` to a negative number, specify K with K$\geq$ 1 and the number of grid points. Here, the grid is equidistant in r^K.
- Specify `dr_center`, `dr_edge` and K. The distance of the grid points is then given by equation (1.37) and the number of grid points is calculated by STRAHL.

In the next line the numerical treatment of the differential equation solver can be influenced. When the equations are solved in ascending order of the charge state Z the density n^1_{z+1} of the next higher stage at the new time point is not known and similar in descending order n^1_{z-1} is unknown. STRAHL iteratively solves the equations in ascending order by calculating the recombination from n^0_{z+1} in the first iteration. In the following iterations n^1_{z+1} is taken from the previous iteration step. The iteration stops when the relative change of the densities at the new time point is below a certain threshold, which can be set here.

When using a value less than 0 and greater than -10 for the threshold parameter (second parameter of the second line), the iterative method is switched off and the Lackner method according to eqautions (1.31) and (1.32) is used. To this end the time step is split into two half time steps. For the first half time step, the respective equations are solved in ascending order of z and for the second half time step, the deschending order is used.

When using a value less or equal -10 and greater than -20 for the threshold parameter, the Lackner method is used without splitting the time step into two half time steps. This method is more for educational purposes to see the oscillation of the solution between even and uneven number of time steps.

```
S T A R T   C O N D I T I O N S
cv    start new=0/from old calc.=1   take distr. from shot   at  time
                 0                         12345                 1.0
```

During a STRAHL run the impurity distribution can be saved and can be used in a later run as start conditions. You can either start from zero impurity density (choose 0) or from the distribution of a previous run (choose 1). Shot number, time and the element symbol is used to find the file with the old impurity distribution. In this example STRAHL would search for the file **./result/C_12345t1.00**.

```
O U T P U T
cv    save all cycles = 1, save final and start distribution = 0
           1
```

The time steps in a STRAHL run have to be defined before run time. A number of equal time steps is called a cycle. After each cycle STRAHL stores the current impurity distribution and related quantities (choose 1). Sometimes it is not necessary to store the complete time evolution and one just wishes to have the result at the end of the run. In this case choose 0 and only the start distribution and the final distribution is saved.

```
T I M E S T E P S
cv    number of changes (start-time+... +stop-time)
                 2
cv    time    dt at start    increase of dt after cycle     steps per cycle
      1.0        1.e-3               1.20                            10
      2.0        1.e-3               1.20                            10
```

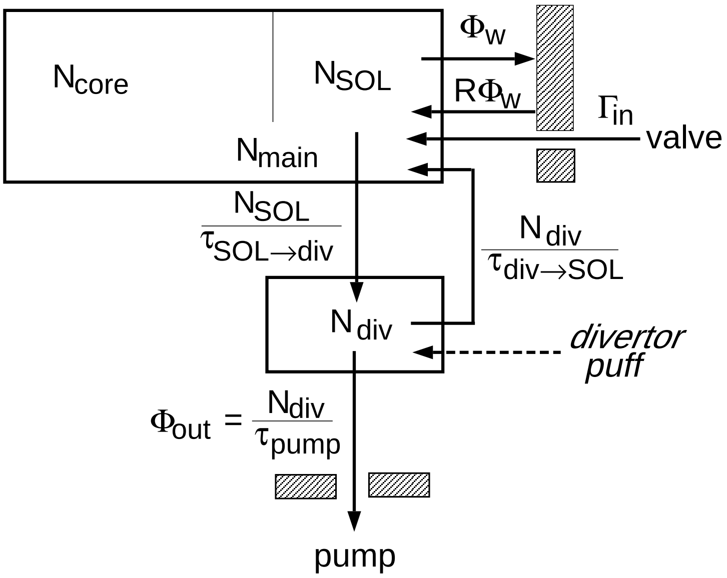
Now we really come to the time step definition. In the first line you give the number of time step changes including the start and stop time. Then you give as many lines as you specified above. The first number is the time at which the time steps shall change (in this example it is just the start and the stop time). Then comes the time increment Δt with which the program shall start after the change. The last number gives the number of time steps n of one cycle. After the completion of a cycle the impurity distribution is stored in a result file. Then the time increment is multiplied with a constant number α and in the new cycle we have $\Delta t_{new} = \Delta t_{old} \times \alpha$. The third parameter in the line defines α. In this example only the stop time from the last input line is relevant and the rest of the line can be filled with dummies. They are only needed when the primitive saw tooth model is switched on (see below).

```
S O  U R C E
cv    position(cm)    constant rate(1/s)   time dependent rate from file
         1000.0             2.5e21                          0

cv    divertor puff   source width in(cm)     source width out(cm)
          0                     0                      0
```

The puff rate of neutral impurities Φ_0 is given in the next line. If Φ_0 is constant in time it is defined by the parameter `constant rate` and the switch under `time dependent ...` has to be set to zero. When setting the switch to 1 a time dependent rate can be specified in the file **./nete/Xxflxnnnnn.dat** where Xx is the element symbol and nnnnn is the same number as is used for the plasma background (the file **./nete/ppnnnnn.i**). In **./nete/Xxflxnnnnn.dat** the first line contains the number of time points n and in the following n lines you give the times and the according rates. The radial position where the neutrals enter the plasma usually is the plasma edge and r_{source} (parameter `position`) has to be set to a value which is greater or equal the maximum r. It is also possible to choose $r_{source} < r_{max}$ to simulate pellet ablation. Another more convenient option for pellet simulation are `source width in` and `source width out`. For `source width in` or `source width out` greater than zero, the code assumes a neutral source distribution with a gaussian shape, which has the maximum at r_{source} and a full width at half maximum (FWHM) of `source width in` to the inside and a FWHM of `source width out` to the outside. For `source width in` and `source width out` less than zero, the code assumes a neutral source being located at the single grid point which is closest to r_{source}. The option `divertor puff` puffs the neutral impurities in a seperate reservoir (the *divertor*). The particles N_{div} in that reservoir enter the plasma at r_{source} at a rate $N_{div}/\tau_{div \rightarrow SOL}$ which is specified by the decay time constant $\tau_{div \rightarrow SOL}$ (see the recycling options below).

```
E D G E ,  R E C Y C L I N G
cv    decay length of impurity outside last grid point(cm)
                          2.0
```

```
cv    Rec.:ON=1/OFF=0   wall-rec.  Tau-div->SOL(ms)   Tau-pump(ms)
            0              1.           55.             240.

cv    SOL-width(cm)
          5.0
```



N_{core}
N_{SOL}
N_{main}
Φ_W
$R\Phi_W$
Γ_{in}
valve
$\frac{N_{SOL}}{\tau_{SOL\rightarrow div}}$
$\frac{N_{div}}{\tau_{div\rightarrow SOL}}$
N_{div}
divertor puff
$\Phi_{out} = \frac{N_{div}}{\tau_{pump}}$
pump

Figure 2.1: *Recycling fluxes in STRAHL.*

Impurity behaviour outside the LCFS in the regime of open field lines is not very well modelled in a one dimensional code (local effects, parallel transport...). The following edge and recycling parameters are needed. The flux of ions from the last mesh point towards the wall is specified by the diffusion constant on the last mesh point and a gradient length λ outside the grid which has to be given in the first line. The radial flux density towards the wall is then $\Gamma_{z,w} = D_{edge} n_{z,edge}/\lambda$. The parallel loss is specified by a time constant $\tau_{SOL\rightarrow div}$ which has to be given in the transport section (see below). A sketch of the recycling of impurities is given in figure 2.1. Wall recycling is specified by the factor R which is in the range from 0(=no wall recycling) to 1(=complete wall recycling). The particles that are lost parallel to the field lines go into the *divertor* reservoir. From there the particles recycle with a time constant $\tau_{div\rightarrow SOL}$ or are pumped by the pumps with a time constant τ_{pump}. The particles N_{div} in the *divertor* reservoir cause a recycling flux rate $N_{div}/\tau_{div\rightarrow SOL}$ and a rate of pumped particles N_{div}/τ_{pump}. All recycling particles enter again as neutrals. The recycling can be switched on and off with the first parameter in that line. The last parameter in that block specifies the radial width (in units of r) of the open field line region (SOL).

```
D E N S I T Y,  T E M P E R A T U R E
A N D   N E U T R A L  H Y D R O G E N  F O R  CX
cv    take from file with:    shot        index
                                1            5
```

This line specifies which file for the plasma background shall be used. In this example it would be the file **./nete/pp00001.5**.

```
N E O -  C L A S S I C A L    T R A N S P O R T
                                    method
     0 = off,  >0 = % of Drift,    1 = approx.
```

```
cv  <0 =figure out, but dont use   2/3 = NEOART   neoclassics for rho_pol <
                0                        2                    .95
```

The rest of the input file deals with transport. The neo-classical transport can be switched on by using a positive number for the first parameter η. In the code the neo-classical drift velocity is multiplied with $\eta/100$. For $\eta < 0$ the code calculates the neo-classical transport but does not use it for the solution of the differential equations. For $\eta = 0$ neo-classical transport is switched off completely. The different methods to figure out the neo-classical values were explained in section 1 and can be chosen next (1 = analytical expressions, 2 = NEOART with one effective impurity species in NEOART with density = total density of all ion stages and z = mean impurity charge, 3 = NEOART with each ion stage as a seperate fluid). The neo-classical transport equations apply only to the region inside the LCFS. The radial region where neo-classical values shall be calculated and used can be set by the last parameter.

```
A N O M A L O U S     T R A N S P O R T
cv    # of changes for transport
                   1

cv    time-vector
              0.0000

cv    parallel loss times (ms)
               2.5

cv    Diffusion  [m^2/s]
      'funct'

cv    D       D_SOL   Dmax    beta    gamma   r_max/r(LCFS)
      0.1     1.0     2.5     15.      6.0       0.8

cv    Drift function       Drift Parameter/Velocity
      'const_c'                 'velocity'
```

The anomalous transport can be given as a function of time. The code uses the settings of the first time point for times which are before the first time point and the settings of the last time point for times which are after the last time point. For the times which are inside the specified time intervall a linear interpolation is used. The first line gives the number of time points and the second line lists all times. It is followed by the parallel loss times in the region of open field lines $\tau_{SOL \rightarrow div}$ [ms].

The diffusion constant D has unit [m^2/s] and the drift velocity v has unit [m/s]. The drift velocity can be specified by using the drift parameter α. In this case the drift velocity is calculated using:

$$v = \alpha \frac{2Dr}{r_{LCFS}^2}$$

The diffusion constant and drift velocity can be parametrized in several ways and the upper example is just one possibilty. The possible options are:

- `'interp'`
  causes quadratic interpolation between values at given radial positions. The data block looks like:

  ```
  cv   function                          only for Drift
       'interp'                    'velocity'  or  'parameter'

  cv   # of interpolation points
  ```

```
     n_rho

cv   rho poloidal grid for interpolation
     rho(1) .... rho(n_rho)

cv   D[m**2/s] / v[m/s] / alpha[1]
     y0(t(1))   y(1,t(1)) ... y(n_rho,t(1))
     ........
     y0(t(nt))   y(1,t(nt)) ... y(n_rho,t(nt))
```

- `'funct'`
  invokes a gauss function with different decay lengths inside and outside some radial position.

$$r_1 = p_2 \times r_{LCFS}$$

For $r \leq r_1$ the function is

$$y(r) = y_0 + (y_1 - y_0) \exp\left[\frac{(r - r_1)^2}{p_0^2}\right]$$

For $r > r_1$ the function is

$$y(r) = y_2 + (y_1 - y_2) \exp\left[\frac{-(r - r_1)^2}{p_1^2}\right]$$

This function is quite flexible to produce various shapes. The data block looks like:

```
cv   function                only for Drift
     'funct'           'velocity'  or  'parameter'

cv   D[m**2/s] / v[m/s] / alpha[1] / pi[1]
     y0(t(1))   y1(t(1))   y2(t(1))  p0(t(1))  p1(t(1))  p2(t(1))
     ........
     y0(t(n))   y1(t(n))   y2(t(n))  p0(t(n))  p1(t(n))  p2(t(n))
```

- `'funct2'`
  This option uses the sum of two functions of the type `'funct'`. Thus, for every time point two lines have to be specified. Each line has the same order and meaning as for the previous option `'funct'`.

- `'const_c'`
  This option just applies to the drift. The code looks up the electron density profile in the middle of the time intervall and adjusts the drift velocity in such a way that the impurity concentration n_{imp}/n_e in the source free region (for $\rho_{pol} < 0.9$) is constant. Thus the drift velocity for $\rho_{pol} < 0.9$ is.

$$v = D \frac{1}{n_e} \frac{dn_e}{dr}$$

For $\rho_{pol} > 0.9$ v is set to zero.

In the case of the drift velocity the line which contains the identifier of the functional form has to be completed with `'velocity'` or `'parameter'` to notify whether v or α shall be used. For the option `'const_c'` the code wants to read this parameter even though it is not needed.

```
cv    # of sawteeth     inversion radius (cm)
            0                 15.

cv    times of sawteeth
            0.0
```

The last two lines give the possibilty to invoke a *very simple* sawtooth model. You just give the inversion radius and the times for sawtooth collapses. At the crash times the code redistributes the impurity density inside $r = \sqrt{2} \times r_{inversion}$ to completely flat profiles keeping the total number of impurity particles constant. The time intervall Δt is reduced to the settings which are in the last line of the time step block. Also the number of time steps and the parameter how to increase Δt are taken from there.

Now you might have a look at the complete input file and ask yourself whether you still remember everything.

```
E L E M E N T
cv     element   atomic weight(amu)  energy of neutrals(eV)
       'C_'           12.             1.0

cv     background ion:  atomic weight    charge
                             2.           1.

G R I D - F I L E
cv    shot      index
     99999        0

G R I D   P O I N T S  A N D    I T E R A T I O N
cv     K      number of grid points   dr_center(cm)  dr_edge(cm)
      6.0             101                 2.0           0.1

      max. iterations     stop iteration
cv     at fixed time    if change below(%)
            100               0.02

S T A R T   C O N D I T I O N S
cv    start new=0/from old calc.=1   take distr. from shot   at  time
                 0                             0      0.0
O U T P U T
cv    save all cycles = 1, save final and start distribution = 0
           1
T I M E S T E P S
cv    number of changes (start-time+... +stop-time)
                 2

cv    time    dt at start    increase of dt after cycle     steps per cycle
      1.0        1.e-3               1.20                        10
      2.0        1.e-3               1.20                        10

S O  U R C E
cv    position(cm)    constant rate(1/s)   time dependent rate from file
         1000.0             2.5e21                          0

cv    divertor puff   source width in(cm)     source width out(cm)
          0                     0                      0

E D G E ,  R E C Y C L I N G
cv    decay length of impurity outside last grid point(cm)
                          2.0

cv    Rec.:ON=1/OFF=0   wall-rec.  Tau-div->SOL(ms)   Tau-pump(ms)
            0              1.            55.             240.

cv    SOL-width(cm)
          5.0

D E N S I T Y,  T E M P E R A T U R E
A N D    N E U T R A L  H Y D R O G E N  F O R CX
```

```
cv    take from file with:    shot        index
                               1            5

N E O C L A S S I C A L    T R A N S P O R T
                                     method
     0 = off,  >0 = % of Drift,    1 = approx.
cv  <0 =figure out, but dont use   2/3 = NEOART   neoclassics for rho_pol <
                0.                      2                    .95

A N O M A L O U S    T R A N S P O R T
cv    # of changes for transport
                   1

cv    time-vector
              0.0000

cv    parallel loss times (ms)
               2.5

cv    Diffusion  [mˆ2/s]
      ’funct’

cv    D       D_SOL   Dmax    beta    gamma   r_max/r(LCFS)
      0.1     1.0     2.5     15.      6.0       0.8

cv    Drift function       Drift Parameter/Velocity
      ’const_c’                 ’velocity’

cv    # of sawteeth     inversion radius (cm)
            0                 15.

cv    times of sawteeth
            0.0
```

Chapter 3

Execution and Output

Once you prepared all the necessary input files you finally can run STRAHL and look at the results with several IDL plotting routines. When using STRAHL and the plotting routines you should know two things about the input on the screen. There are always default values for the control variables which are shown in brackets at the end of the input line. The default values can be accepted by entering a slash (/). Very often there is a list of key-words and one of the characters is displayed in inverse mode (white on black). You activate the option by entering the highlighted character (lower or upper case).

3.1 Executing STRAHL

An interactive STRAHL run starts by just typing `strahl` in the current directory. The code comes back and asks for the name of the main input files. Give the file name without the path specification (i.e. without **./param_files**). Now you are asked whether you want to halt the calculation at some time in the time intervall you were defining in the input file. By default STRAHL runs to the end of the simulation time. After the transport calculations have finished the radiation is calculated and then STRAHL asks you whether you want to save the current distribution (option A) or whether you want to save the time development (option S). If you are not at the end of the simulation time you can check what you got so far by using one of the plotting routines and come back to STRAHL to continue the run.
STRAHL stores the result in two files.

- Every run is saved in **./result/strahl_result.dat**. Thus the next run overwrites the old results.
- If you ask for a backup of the result (option S) STRAHL invokes a system call to copy **./result/strahl_result.dat** to a file with name **./result/XXnnnnntx.xx_y.yy_i**. XX is the element symbol nnnnn is the number of the file for the background plasma, x.xx and y.yy give the time intervall and i is an index you enter at run time. For times larger the 10s the format xx.x is used.

There are several options which can be supplied with the call of STRAHL:

- no option: `strahl`
  All input parameters you enter at run time are stored in the file **./strahl.control**.
- `strahl a`
  All input parameters are read from **./strahl.control**
- `strahl n`
  No radiation results are calculated.

- `strahl d`
  Only the impurity distribution and the diagnostic line radiation is calculated.
- `strahl s`
  Only the total impurity density and the soft X-ray radiation is saved.
- `strahl w`
  Some atomic data are saved in directoy /tmp/.
- `strahl v`
  STRAHL produces a lot of output on the screen. This option might be helpful in the case of runtime errors or to detect errors in the input files. The best way to use this option is to combine it with the a-option and enter: `strahl a v > strahl.trace`. Now you can search the error in **./strahl.trace**.
- `strahl h`
  All options are displayed.

If you want to include STRAHL into other programs, e.g to perform scans or fitting, you might use the following scheme. Setup all the input files and denote the parameter that you want to change by the program with a `cv#` in the appropriate input file. These parameter are then transferred to STRAHL via the file **./ext_parameter.dat** and STRAHL is started with option a.

3.2 The Result File

The result files are written in the Network Common Data Form (netCDF). The netCDF format is self describing and allows direct-access of a small subset of the data. The netCDF software is freely available for C, C++, Fortran and other languges. It is also included in IDL which I used to write readout and plotting routines. Documentation about netCDF can be get from `http://www.unidata.ucar.edu` and the software for various platforms is available via anonymous FTP from the directory
`ftp://ftp.unidata.ucar.edu/pub/netcdf/`.
The fastest way to inspect the output file is the program `ncdump` which comes with the netCDF software. The best is to pipe the result through `more` or `less` by entering `ncdump filename | more`. Part of the `ncdump` output is shown here for a run with carbon:

```
netcdf strahl_result {
dimensions:
grid_points = 101 ;
ionisation_stages = 7 ;
time = UNLIMITED ; // (18 currently)
.........

variables:
float time(time) ;
time:units = "s" ;
time:_FillValue = 0.f ;
time:C_format = "%.4g" ;
        float rho_poloidal_grid(grid_points) ;
rho_poloidal_grid:units = "1" ;
rho_poloidal_grid:_FillValue = 0.f ;
rho_poloidal_grid:C_format = "%.4g" ;
float impurity_density(time, ionisation_stages, grid_points) ;
impurity_density:units = "1/cm**3" ;
impurity_density:_FillValue = 0.f ;
impurity_density:C_format = "%.4g" ;
............

// global attributes:
:mass_of_impurity = 12.f ;
:maximum_charge = 6 ;
.........
data:

 time = 1, 1.009, 1.02, 1.033, 1.05, 1.069, 1.093, 1.121, 1.155, 1.196,
    1.244, 1.303, 1.373, 1.457, 1.558, 1.68, 1.825, 2 ;

........
}
```

The structure of the different data arrays is very easy to understand from the output of `ncdump`. The dimensions of the different variables are shown first, then comes a block with the variables followed by a block with global attributes and finally there is the long data section. The definition of three varibles is shown in this example. The first is `time` with one dimension `time` and unit [s] . The second is `rho_poloidal_grid` with dimension `grid_points` and unit[1] and the third is `impurity_density` with three dimesions `(time, ionisation_stages, grid_points)` and unit [cm^{-3}]. There are also global attributes defined like `mass_of_impurity` and `maximum_charge`. I think the names and the structure of the variables need no long explanations.

An alternative way is a public domain IDL-program which is called `ncbrowse`. With `ncbrowse` you can inspect the structure of the variables and make plots.

When you want to write your own program to read and analyse the data you might use the routine `read_strahl.pro` which runs under IDL. With `read_strahl.pro` you can access all data in a convenient way. The usage of `read_strahl.pro` is very easy when you have some basic knowledge of IDL. The header of the routine hopefully gives enough explanations:

```
pro read_strahl,file_name,var_name,data,$
                time=time,rho=rho,stage=stage,gatr=gatr,all=all,$
                verbose=verbose
;*******************************************************************
;Reads variables and attributes from STRAHL result files:
;
;Input:
;-> file_name: a) Give full name including path
;              b) name without path takes file from
;                 $Home/strahl/result
;                 where $Home is the home directory as stated in the
;                 UNIX environment variable Home
;-> var_name:  Name of the variable/attribute
;              for an attribute set the keyword "gatr"
;Keywords:
;-> time:      =0     -> all times are retrieved
;              =t > 0 -> only result at time=t is retrieved
;-> rho:       =0     -> all radii are retrieved
;              =r > 0 -> only result at normalized poloidal flux label
;                        rho = r is retrieved
;-> stage:     =0     -> all stages/lines are retrieved
;              =n > 0 -> only stage/line  n is retrieved.
```

```
;-> all        = 1 -> get all data of the  variable.
;-> gatr       = 0 -> read a variable
;              = 1 -> read an  attribute
;-> verbose    write information to the screen
;
;Output:
;<- data:      data
;   *****************************************************************
```

Generally, the meaning of the different dimensions of the variables is obvious. However, the radiation stages need some extra explanation.

- `impurity_radiation` and `sxr_radiation`
  index 1: total line radiation of neutral impurity
  index 2: total line radiation of singly ionised impurity
  ....
  index n: total line radiation of hydrogen-like ion
  index n+1: bremsstrahlung due to electron scattering at main ion
  index n+2: total continuum radiation of impurity (bremsstrahlung and recombination continua)
  index n+3: bremsstrahlung due to electron scattering at impurity
  index n+4: total radiation of impurity (and main ion, if set in **Xx.atomdat**)

- `spectral_bremsstrahlung`
  index 1: = 0
  index 2: bremsstrahlung due to electron scattering at singly ionised impurity
  ....
  index n+1: bremsstrahlung due to electron scattering at fully ionized impurity
  index n+2: bremsstrahlung due to electron scattering at main ion
  index n+3: total bremsstrahlung of impurity (and main ion, if set in **Xx.atomdat**)

3.3 Plotting Routines for IDL

There are two plotting routines available which run under IDL. They are called `pstrahl` and `pdiag`. All the routines which are needed to run these two programs are in the directory **./idl**.

The following selection of plots is implemented so far:
pstrahl

- impurity densities of each ion stage and total impurity density (radial distribution or time development for one radius)
- impurity densities neglecting the transport effect, .i.e. in corona ionisation equilibrium (radial distribution or time development for one radius)
- impurity concentrations of each ion stage and and total impurity concentration (radial distribution or time development for one radius)
- ΔZ_{eff} of each ion stage, total ΔZ_{eff} of impurity and Z_{eff} (radial distribution or time development for one radius)
- emissivity due to total line radiation of each ion stage, bremsstrahlung, and recombination radiation. The total emissivity is also compared with the emissivity which would follow from corona ionisation equilibrium. (radial distribution or time development for one radius)
- the same for the spectral range of SXR

- bremsstrahlung due to impurity and main ion (radial distribution or time development for one radius)
- emissivity due to line radiation of diagnostic lines (radial distribution or time development for one radius)
- total power loss density due to radiation/ionisation of impurity (radial distribution or time development for one radius)
- electron density, main ion density, main ion loss due to impurity (radial distribution or time development for one radius)
- electron temperature, main ion temperature (radial distribution or time development for one radius)
- anomalous and neo-classical (total and parts) diffusion coefficent (radial distribution or time development for one radius)
- anomalous and neo-classical (total and parts) drift velocity (radial distribution or time development for one radius)
- anomalous and neo-classical v_{drift}/D (radial distribution or time development for one radius)
- collisionality for main ion and impurity plus Pfirsch-Schlüter and banana-plateau limit (radial distribution or time development for one radius)
- various volume integrals of total power loss density (time development)
- impurity particle rates for flux through valve, recycling fluxes, perpendicular and parallel losses (time development)
- number of impurity particles in the confined plasma, at the *wall*, in the *divertor* and in the *pump* (time development)
- energy content of the plasma (time development)

pdiag

- emissivity due to line radiation of diagnostic lines (radial distribution or time development for one radius)
- emissivity due to line radiation of diagnostic lines following corona ionisation equilibrium (radial distribution or time development for one radius)
- Intensity of diagnostic lines when integrating the emissivity along the major radius at the mid plane

  $$\int_{z=0} \epsilon(R)dR$$

  (time development)
- the same for the total line radiation of each stage (time development)
- the same for the total soft X-ray radiation (time development)
- mean poloidal flux label of emission shells of diagnostic lines:

  $$\frac{\int_{z=0} \epsilon(R)\rho_{pol}(R)dR}{\int_{z=0} \epsilon(R)dR}$$

  (time development)

- mean electron temperature of emission shells of diagnostic lines:

$$\frac{\int_{z=0} T_e(R)\epsilon(R)dR}{\int_{z=0} \epsilon(R)dR}$$

  (time development)

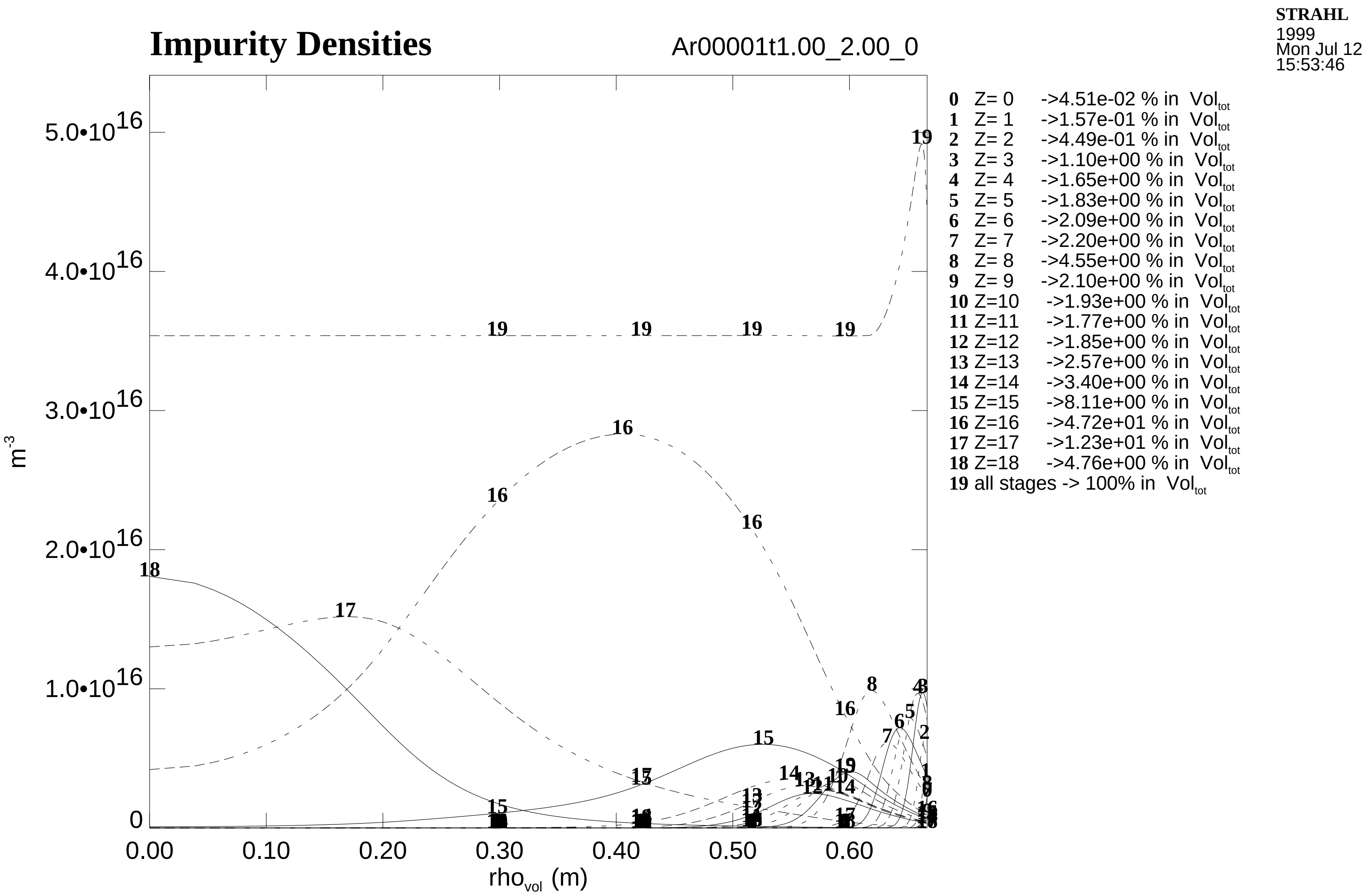
- mean impurity density in emission shells of diagnostic lines:

$$\frac{\int_{z=0} n_{imp}(R)\epsilon(R)dR}{\int_{z=0} \epsilon(R)dR}$$

  (time development)

- spectrum of the diagnostic lines for a given line width

You should be able to find out how the program works. There is also an option to print the data on a PostScript printer. One option should be noted. You might call the programs with the keyword `save_file` and `input_file`. You start by using `pstrahl, save_file='my_plots'`. Now you continue with pstrahl and produce some plots on the printer. After changing something in the input parameter sets you repeat the STRAHL run and want to look at the same type of plots for the new result. Now, you just enter `pstrahl, input_file='my_plots'`. The previos session of pstrahl was saved in the file **./my_plots** and is now repeated by taking the commands from the file. When having a number of plots this might be a little fast on the screen and it is only useful to combine this option with plots that are printed. An example plot of the impurity density distribution of argon as produced by `pstrahl` is shown on the following page.



t= 2.00000s a= 61.6cm Z/A: plasm.=1/2 imp.18/ 40 <ne>=6.90e+19m^{-2} Te(0)= 3.53keV ne(0)=8.58e+19m^{-3} Zeff(0)=1.12
for rho=0.1/0.4/0.9: D=0.10/0.16/2.20 m^2/s v= 0.0/ 0.0/ 0.0 m/s neocl= 0.% CEX=0
influx(s^{-1}):valve=1.90e+20 wall=0.00e+00 div=0.00e+00 div/main=64.7 tau(ms):sol= 2.5 div= Inf pump= Inf
separatrix: Te=1.01e+02eV Ne=3.49e+19m^{-3} falloff:Te=1.5cm ne=1.9cm w(SOl)=2.5cm Ion.Length=0.02cm

Bibliography

[Behringer, 1987] Behringer, K. (1987). Description of the impurity transport code STRAHL. *'Description of the impurity transport code STRAHL', JET-R(87)08, JET Joint Undertaking, Culham.*

[Fletcher, 1997] Fletcher, C. A. J. (1997). *Computational Techniques for Fluid Dynamics*. Springer, Berlin Heidelberg New-York, 2nd edition.

[Fußmann, 1992] Fußmann, G. (1992). Teilchentransport in magnetisch eingeschlossenen plasmen. Technical Report 1/273, IPP, Garching, Germany. Habilitationsschrift.

[Hinton and Hazeltine, 1976] Hinton, F. L. and Hazeltine, R. D. (1976). Theory of plasma transport. *Reviews of Modern Physics*, 48(2):239–308.

[Hirshman and Sigmar, 1981] Hirshman, S. P. and Sigmar, D. J. (1981). Tokamak impurity transport. *Nuclear Fusion*, 21(9):1079–1201.

[Houlberg et al., 1997] Houlberg, W. A., Shaing, K. C., Hirshman, S. P., and Zarnstorff, M. C. (1997). Bootstrap current and neoclassical transport in tokamaks of arbitrary collisionality and aspect ratio. *Physics of Plasmas*, 4(9):3230–3241.

[Kim et al., 1991] Kim, Y. B., Diamond, P. H., and Groebner, R. J. (1991). Neoclassical poloidal and toroidal rotation in tokamaks. *Physics of Fluids B: Plasma Physics*, 3(8):2050–2060.

[Lackner et al., 1982] Lackner, K., Behringer, K., Engelhardt, W., and Wunderlich, R. (1982). An algorithm for the solution of impurity diffusion under finite reaction rates. *Zeitschrift für Naturforschung*, 37a(5):931–938.

[Peeters, 2000] Peeters, A. G. (2000). Reduced charge state equaitons that describe pfirsch schlüter impurity transport in tokamak plasma. *Physics of Plasmas*, 7(1):268–275.

[Thompson et al., 1985] Thompson, J. F., Warsi, Z. U. A., and Mastin, C. W. (1985). *Numerical Grid Generation, Foundations and Applications*. North-Holland, New-York, 1st edition.