MASTER

Numerical model for divertor detachment control

de Kinkelder, E.M.

Award date:
2019

Eindhoven University of Technology, Industrial and Applied Mathematics

Masters thesis:

Numerical Model for Divertor Detachment Control

Supervisors:

prof. dr. ir. Barry Koren	ir. Timo Ravensbergen	dr. ir. Matthijs van Berkel
Eindhoven University of Technology	DIFFER	DIFFER

Author: Eloy de Kinkelder
Studentnumber: 0858115

Date of Submission: February 12, 2019

Abstract

Power production via nuclear fusion promises reliable and localised sustainable energy production for future generations to come. One of the hurdles that still has to be overcome to realise nuclear fusion is reducing the heat and ion flux towards the divertor targets. Future high energy tokamaks will be required to operate in a detached regime to reduce the heat and ion flux towards the targets. To stabilise the detachment front a feedback control method will have to be implemented. Testing and designing this requires a numerical model describing the dynamics of the plasma and neutral particles in the divertor.

The model that is considered in this thesis is a 1-dimensional fluid dynamics model of the plasma and neutral particles along a magnetic field line consisting of four coupled partial differential equations. Since it is needed for feedback control, the computational cost is an important feature. It also needs to be robust against the subsonic and supersonic flow which can both occur in the divertor plasma.

Three numerical schemes are constructed and compared to each other and to a benchmark model. Only one, a flux vector-splitting scheme with implicit time integration, proves to be robust enough to model both the subsonic and supersonic flow in both directions. This numerical scheme is able to simulate both an attached and a detached plasma. The implicit time integration and the implementation of the analytic Jacobian are the main factors that improve the simulation time. The latter also provides linearizations around detached operation points, which are necessary for controller synthesis.

Acknowledgements

First I would like to express my gratitude to my supervisors prof. dr. ir. Barry Koren, ir. Timo Ravensbergen and dr. ir. Matthijs van Berkel for their advice during this graduation project. I would also like to thank dr. ir. Jim Portegies for being a member of the committee.

Furthermore I would like to thank dr. Egbert Westerhof for helping me understand the physical aspects of divertor detachment. I would also like to thank dr. Ben Dudson for making the code for the SD1D model available and answering my emails to help me understand it.

Finally I would like to thank my family, friends and housemates for mentally supporting me through this graduation project.

Contents

1 Introduction

Nuclear fusion is a process where two small particles are fused together to form a heavier particle. This reaction has energy as by-product and produces no greenhouse gases nor long lasting radioactive waste. Therefore it is seen as a future optional long term solution to the greenhouse effect. The temperatures required to reach sufficiently high reaction rates imply the fusion fuel is in the plasma state. The plasma is magnetically confined in devices with toroidal symmetry, called tokamaks. The "exhaust" of the nuclear fusion device is the divertor, which is the part that is considered in this thesis. Here relatively cold plasma flows towards a surface called the target. When the plasma interacts with a solid surface it can damage it. Therefore it is needed to minimise the heat and ion fluxes towards the target. One way to do so is by pumping a neutral gas into the plasma causing extensive cooling via volumetric processes like radiation and ionisation, causing the plasma to recombine close to the target and reducing the heat and ion flux. The amount of gas pumped into the system must be controlled by feedback control, which requires a numerical model to describe the dynamics of the plasma and neutral particles. In this masters thesis several numerical schemes are constructed to solve the system of partial differential equations describing the flow dynamics numerically. The model is needed for control and predictive capabilities so its computational cost has priority over its physical correctness.

Although the dynamics of the neutral particles require a 2-dimensional simulation [1, Chapter 2] and advanced 2-dimensional models like SOLPS-ITER, EDGE2D and SOLDIV already exist. They can not be used for feedback control because of their computational complexity. A zero dimensional model, like the modified 2-point model is not accurate enough [1] and can by definition not capture the dynamics of the detachment front. A one dimensional model will be implemented. The model must be time dependent to be usable in control. So a model with only the stationary solution as in [2] can not be used. Completely dynamic models already exist [3], [4] and will be used as a starting point. Three numerical schemes are constructed and tested for their suitability to model divertor detachment. To increase the computation speed it is also attempted to use the stationary solutions for the plasma equations with the dynamic solutions for the neutral equations.

In Chapter 2 some background information will be given about nuclear fusion devices to explain divertor detachment. Then a 1-dimensional model describing the dynamics of divertor detachment will be introduced and analysed. In Chapter 3 three numerical schemes will be described: the Implicit scheme (Section 3.1), the Semi-explicit scheme (Section 3.2) and the Flux-splitting scheme (Section 3.3). These will be validated, analysed and compared to each other and a benchmark model in Section 3.4. Using the stationary plasma equations instead of the time dependent equations to improve performance will be investigated in Section 3.5. In Chapter 4 some applications of the Flux-splitting scheme will be displayed to demonstrate that it can simulate a detached plasma. The conclusion and outlook are given in Chapter 5.

2 Modelling divertor detachment

2.1 What is divertor detachment control?

2.1.1 Tokamaks

Nuclear fusion is a process where two small atom nuclei are fused together to form a heavier nucleus with energy as by-product. For nuclear fusion there are several reactions possible, an example is (2.1.1) [5]:

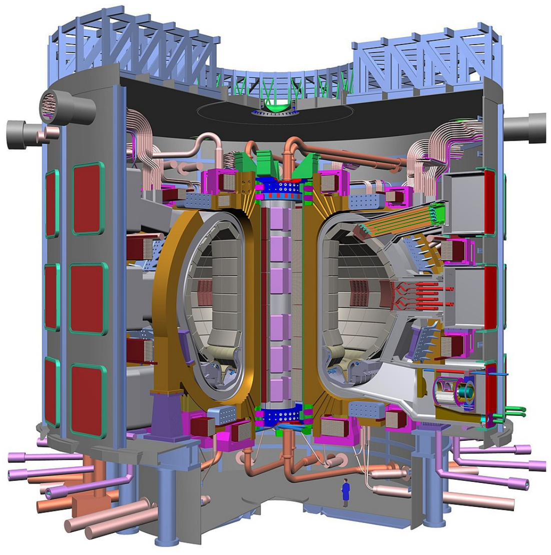
$$^2H + {}^3H = {}^4He + n + 17.6\text{MeV}. \tag{2.1.1}$$

The nuclei of atoms are positively charged, so they repulse each other. To make the particles fuse together, we need to overcome this repelling force (Coulomb barrier) [6]. When the nuclei get close enough, the strong interaction becomes dominant and fuses the particles. To overcome the Coulomb barrier the particles require a lot of kinetic energy, so the fusion fuel will have high temperatures. These temperatures cause the Hydrogen particles to become fully ionised, bringing them in the fourth state of matter called plasma. Plasma is a state in which the electrons are separated from the atoms. Hence all the fuel particles are charged, making their trajectories controllable by magnetic forces in the ideal limit. The plasma should not interact with solid surfaces. This will cool down the plasma, damage the surface and mix solid particles with the plasma, making it impure. Therefore a magnetic field is used to control the trajectory of the plasma. The device used for the magnetic confinement is called a tokamak, two pictures of tokamaks are given in Figures 2.1a and 2.1b.

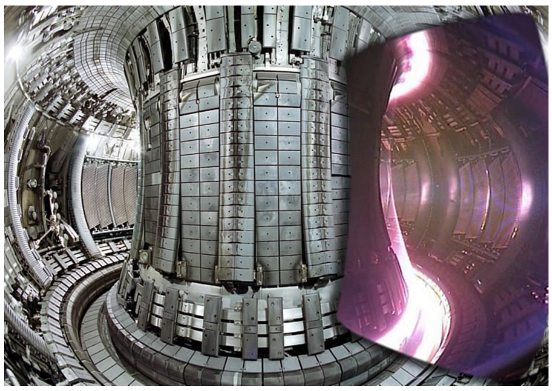
The name tokamak comes from the Russian translation of toroidal chamber with magnetic coils. Toroidal refers to the shape of the chamber, which is shaped like a torus. The chamber is where the main plasma flows and nuclear fusion takes place. Magnetic coils induce the magnetic field that is used to control the flow of the plasma to minimise its interaction with the walls. (Creating a perfectly closed magnetic field is not possible, the field lines will eventually hit a wall.)
A schematic picture of the magnetic field is given in Figure 2.2. The field line B_ϕ is induced by the coils that are around the torus alongside the walls of the chamber. The plasma particles are charged. Hence when moving they induce a magnetic field as well. It is similar to an electrical current in a wire that induces a magnetic field. The field induced by the plasma is called B_θ. The total magnetic field becomes $B = B_\phi + B_\theta$ which is indicated with the green arrow in Figure 2.2. In the ideal limit the plasma particles follow these magnetic field lines.

2.1.2 Divertors

In principle, the plasma follows the magnetic field lines, but due to various drift effects there is also transport perpendicular to the field lines. The high temperature and velocity will cause



(a) Schematic picture of a tokamak, the magnetic coils that control the trajectory of the plasma are coloured orange, the chamber is within these coils. [7]



(b) Photograph of a tokamak, on the right they added colour for the plasma. The plasma in a tokamak is too hot to emit any visible light, but at the boundaries it is colder and does emit light. [8]

Figure 2.1: Illustrations of tokamaks.

the plasma to damage the wall. Moreover the solid particles that come from the wall will mix with the plasma, cooling it down, making fusion impossible. So the plasma-wall interaction has to be minimised. To do this there are two options, limiters and divertors, both are shown in Figure 2.3. Limiters are easiest to explain, but to do so, some definitions have to be explained first. Particles mostly follow the magnetic field lines (ϕ in Figure 2.2) with a velocity component $v_{||}$. The component of the velocity that is perpendicular to these field lines is called $v_{\perp}$. The flow along the field lines is very fast compared to the perpendicular flow that is caused by drift effects, so $|v_{||}| \gg |v_{\perp}|$. In the cross-cut section of the plasma several areas can be defined.

- A closed field line is a line that is not interrupted by anything and makes a closed loop.
- The separatrix is the most outer magnetic field line that is closed. In Figure 2.3 the separatrix has been indicated.

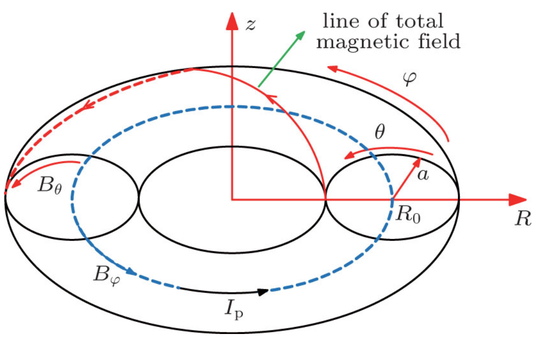


Figure 2.2: Schematic view of magnetic field used in tokamak to confine the plasma [9].

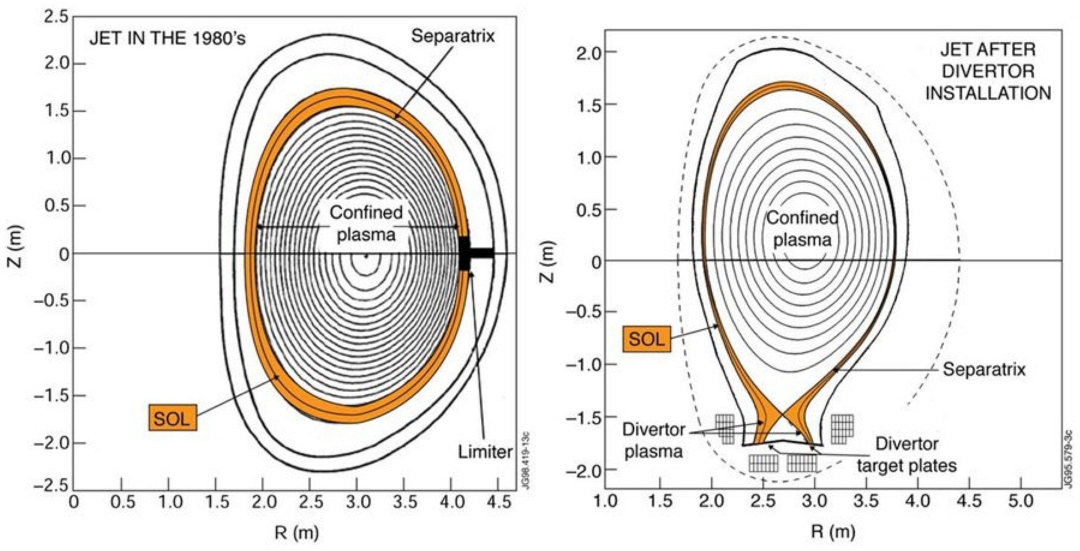


Figure 2.3: Schematic view of a limiter and a divertor [6].

- The layer outside the separatrix is called the Scrape Off Layer (SOL) which has been indicated in Figure 2.3.

With these definitions the concept of a limiter can be described. A limiter interrupts the magnetic field lines in the SOL. Most particles that drift past the separatrix will collide with the limiter quickly because $|v_{||}| >> |v_{\perp}|$. Since $|v_{\perp}|$ is relatively small the particles can only travel a short distance towards the wall. So the limiter takes the impact of most particles that travel past the separatrix prohibiting them from colliding with the wall.

Divertors have the same purpose, preventing plasma particles to interact with the solid surface wall, but has some advantages over the limiter. The divertor uses a different group of coils that carry a current in the opposite direction with respect to the plasma current. Thereby creating a saddle point in the magnetic flux, this saddle point is called the x-point. The plasma particles in the SOL keep following the magnetic field lines and collide with the divertor target plates. A big advantage of this method is that the plasma-solid surface interaction is isolated from the main plasma. Hence unwanted particles are generated away from the main plasma. Additionally, divertors allow physical baffling, which can further reduce the transport of neutral particles to the core. For this reason all modern tokamaks use divertors instead of limiters, hence this thesis only considers the divertor.

2.1.3 Detachment

For high power machines the power and particle fluxes towards the divertor targets are too much. So measures have to be taken to stop the particle flux from damaging the targets. One way to do this is by creating a detached plasma, which will be explained in this section.

The divertor plasma is the plasma between the x-point and the target plates, see Figure 2.3. In this region several operating regimes are defined, of which only detachment is discussed in this report. According to Matthews [10] detachment is defined by: "*state in which large pressure* *gradients (static plus dynamic) are observed parallel to the magnetic field with consequently low plasma power and ion fluxes to the material surfaces bounding the system*". The low plasma power and ion fluxes to the material surfaces (the divertor target plates) make this an appealing operating regime, because there is less plasma-surface interaction. It is experimentally characterised by the following observations [5].

1. When increasing the upstream ion flux there is an increase in the ion flux at the divertor targets, this is called a density ramp. The target ion flux will keep increasing with the upstream ion flux up until a given point, at which it will start decreasing. This is called the roll-over.
2. D_α radiation (Helium nuclei) continues to increase even after the roll-over, when the target ion flux is decreasing.

Hsu et al. [11] have shown that a detached divertor plasma can be achieved by making the plasma "collide" with a neutral gas. Ion-neutral collisions and impurity radiation decrease the plasma temperature until it is low enough for the plasma to recombine. This will change the plasma particles to neutrals reducing the plasma flux to the targets and with this the heat-flux to the target also decreases. The problem with this method however is that if too much neutral gas is inserted into the system the detachment front will move away from the target towards the x-point. When it reaches the x-point the main plasma will start to cool, stopping nuclear fusion. On the other hand, if too little gas is inserted into the system, the divertor plasma will not detach and the target will be damaged. Therefore to keep the detachment front in the right place the insertion of gas needs to be controlled.

2.1.4 Model based controller design

A brief introduction into model based controller design is required as background information for this thesis to understand the need for and requirements of the numerical model. Consider the following system of linear differential equations:

$$\dot{x} = Ax + Bu, \tag{2.1.2}$$

$$y = Cx + Du, \tag{2.1.3}$$

where x contains the states of the system, y the output, u the input that can be controlled and A, B, C and D are matrices. For example y is the location of the detachment front, u is the amount of neutral gas being pumped into the system per second and x contains all the variables that define the plasma and neutral dynamics in the divertor region. (these dynamics are actually not linear, so this can not be done directly, in Section 4 the linearized system is given.) The goal is to choose u such that y remains as close to its desired value as possible.

To do this we first need to define this system of equations and make them into an ODE, this is done by the discretization of the spatial derivatives. Then, controllers can be mathematically synthesised based on the linearization [12]. Possible non-linearities can often be described as perturbations or uncertainties in a linear model, for which also synthesis techniques exist [12].

2.2 Modelling detachment

To better understand the underlying dynamics and design a controller a priori to an experiment, a model for the divertor detachment is needed. For its control purposes the model needs to be computationally inexpensive. Even though detachment is a two or even three dimensional process [1, Chapter 2], using a multi-dimensional model for the entire divertor region is too expensive. Because of the toroidal symmetry of the tokamak only a cross section of the tokamak has to be modelled, reducing the problem to two dimensions. This is still too computationally intensive. Another dimension can still be removed by considering that the plasma moves along the magnetic field lines. So the position of a particle can be expressed by its position along the field line. This results in a one dimensional model. Overall the model is identical to Nakazawa's model [4] thus far. The geometry of the field lines is simplified to a straight line, so the variables can be expressed by their coordinates on this line. A schematic view of the domain is given in Figure 2.4. As a model for the dynamics the partial differential equations used in [4] are also used here.

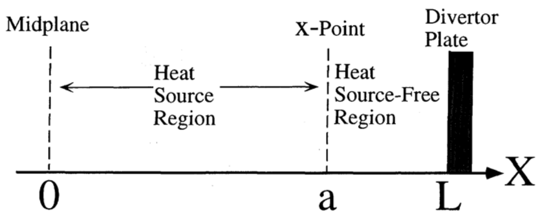


Figure 2.4: Schematic picture of the 1-dimensional domain used in the model [4]. The midplane is a point upstream in the SOL where by symmetry it is assumed that there are as many particles going left as right. The SOL acts as a particle and heat source.

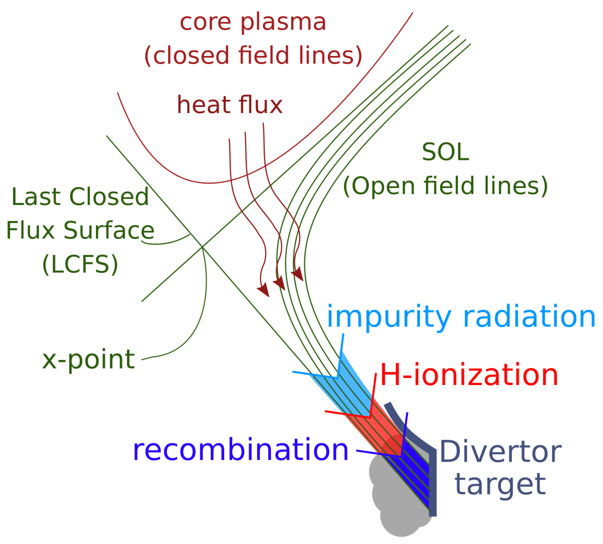


Figure 2.5: Schematic picture of the divertor plasma region. The domains for the main radiative sources are coloured, these will be explained later [13].

2.2.1 Mathematical model

To get a better understanding of this system of partial differential equations (PDEs) each equation will be considered separately, for a more in depth explanation and the derivation [14, Chapter 9] is recommended. We start with the particle conservation equation:

$$\frac{\partial n}{\partial t}+\frac{\partial(nv)}{\partial x}=s_n, \tag{2.2.1}$$

in which n is the particle density in m^{-3}, t the time in seconds, v the average particle velocity parallel to the magnetic field line in m/s and x the distance in m. The first term is the change in particle density. The second term is the particle density flux. The term on the r.h.s. is the particle source which like all source terms, will be explained in section 2.2.5. The equation

$$\frac{\partial(mnv)}{\partial t}+\frac{\partial(mnv^2+2q_enT)}{\partial x}=s_v \tag{2.2.2}$$

expresses the conservation of momentum. Here m is the mass of a particle in kg, T is the energy in eV (electron volt) and q_e is the elementary charge in Coulombs. (q_e is not originally in [4] because they use T for the energy in Joule, but we prefer this notation.) The first term is the time derivative of the momentum. The second term consists of a spatial derivative of two terms. The first is the momentum flux and the second represents the pressure. The factor 2 is because two species are considered in this model, ions and electrons. It is assumed that their particle densities, velocities and temperatures are the same. So $n_e = n_i = n$. The same holds for the velocity and temperature. Because $m_i \gg m_e$ all the terms containing the electron mass m_e are neglected and m_i is denoted as m. The term on the r.h.s. is the momentum source. The energy equation reads

$$\frac{\partial}{\partial t}\left(2\cdot\frac{3}{2}nq_eT+\frac{1}{2}mnv^2\right)+\frac{\partial}{\partial x}\left(2\cdot\frac{5}{2}nq_eTv+\frac{1}{2}mnv^3-\kappa_{\parallel}^{e}\frac{\partial T}{\partial x}\right)=Q. \tag{2.2.3}$$

The first term is the change in energy, which consists of the kinetic energy $\frac{1}{2}mnv^2$ and the internal energy $2\frac{3}{2}nq_eT$. The factors 2 are again because a system with two species is considered. The second term is a spatial derivative, containing three separate terms. The first term is the transport of internal energy, the second term $\frac{1}{2}mnv^3$ is the flux of kinetic energy and the last term is the transport of heat by conduction. Its coefficient $\kappa_{\parallel}^{e}$ is a nonlinear function of T. The term on the r.h.s. is the energy source.

To model the dynamics of the neutral particles it is assumed that these move randomly in the domain as they are not confined to the magnetic field lines. So a diffusion equation is used instead of a set of fluid dynamics equations. This also reduces the computational cost, making it more suited for our purpose, control. The equation reads:

$$\frac{\partial n_n}{\partial t}-\frac{\partial}{\partial z}\left(D\frac{\partial n_n}{\partial z}\right)=s_{n_n}, \tag{2.2.4}$$

where the diffusion coefficient D is a non-linear function of T and n. Different spatial coordinates are used because the plasma particles collide with the target at an angle θ. It is assumed that the neutral particles are emitted from the target perpendicular to it. So dz is defined as $|dz|=\sin\theta|dx|$. The origin of the z-coordinate is at the target instead of upstream, Figure 2.6 illustrates this.

In the numerical models the scaled dimensionless version of these equations will be used, which

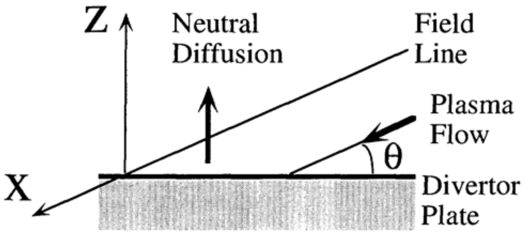


Figure 2.6: z and x axis at the target plates. [4]

are given by:

$$\frac{\partial n}{\partial t} + \frac{\partial (nv)}{\partial x} = s_n, \tag{2.2.5}$$

$$\frac{\partial (nv)}{\partial t} + \frac{\partial}{\partial x}\left(nv^2 + 2nT\right) = s_v, \tag{2.2.6}$$

$$\frac{\partial}{\partial t}\left(\frac{1}{2}nv^2 + 3nT\right) + \frac{\partial}{\partial x}\left(\frac{1}{2}nv^3 + 5nvT - \frac{\hat{t}}{q_e \hat{n} \hat{x}^2}\kappa_{\parallel}^{e}\frac{\partial T}{\partial x}\right) = s_Q, \tag{2.2.7}$$

$$\frac{\partial n_n}{\partial t} - \frac{\hat{t}}{(\hat{x}\sin\theta)^2}\frac{\partial}{\partial x}\left(D\frac{\partial n_n}{\partial x}\right) = s_{n_n}. \tag{2.2.8}$$

All variables and source terms are scaled. The parameters with a hat are constants that have a dimension and are chosen as:

- $\hat{n} = 10^{20}$ m $^{-3}$, which is a typical particle density in the divertor plasma.
- $\hat{T} = 100$ [eV], which is a typical temperature in the SOL.
- $\hat{x} = L$ m, L is the length of the domain.
- $\hat{v} = \sqrt{\frac{q_e \hat{T}}{m}}$ m/s, which is the thermal velocity in a system containing particles of mass m and temperature $\hat{T}$ [eV].
- $\hat{t} = \hat{x}/\hat{v}$ [s], which is the time it would take a particle to travel the entire length of the domain with velocity $\hat{v}$.

The derivation of Equations (2.2.5)-(2.2.8) is given in Appendix B.1.

2.2.2 Analysis of the system of PDEs

In this section the dimensionless PDE (Equations (2.2.5)-(2.2.8)) will be analysed such that it can be classified and the number of boundary conditions can be derived. First ignore the heat diffusion term, the source terms and the neutral particles equation, so only the first order derivatives of the plasma equations are considered. Notate the remaining part as $\frac{\partial F(y)}{\partial t} + \frac{\partial G(y)}{\partial x} = 0$, where $y = (n; v; T)$ and F and G are given by:

$$F(y) = \begin{pmatrix} F^p(y) \\ F^m(y) \\ F^E(y) \end{pmatrix} = \begin{pmatrix} n \\ nv \\ \frac{1}{2}nv^2 + 3nT \end{pmatrix}, \tag{2.2.9}$$

$$G(y) = \begin{pmatrix} nv \\ nv^2 + 2nT \\ \frac{1}{2}nv^3 + 5nvT \end{pmatrix}. \tag{2.2.10}$$

Here F^p, F^m and F^E represent the particle density, momentum and energy respectively. Now it can be seen that this part of the system of PDEs is quasi-linear since it can be written in the form

$$\sum_\alpha A_\alpha \frac{\partial y}{\partial \alpha} = 0, \tag{2.2.11}$$

where A_α does not contain any derivatives w.r.t. t or x. In this case $\alpha \in \{t, x\}$, $A_t = \frac{\partial F}{\partial y}$ and $A_x = \frac{\partial G}{\partial y}$. The classification of this PDE is done according to [15, Chapter 3]. Substitute $y = \hat{Y}e^{i(x-\omega t)}$ into equation (2.2.11). Then to calculate ω the following equation has to be solved:

$$\det(-\omega A_t + A_x) = 0 \Leftrightarrow \det\left(\omega \frac{\partial F}{\partial y} - \frac{\partial G}{\partial y}\right) = 0.$$

Using the property of the determinant $\det(AB) = \det(A)\det(B)$ and the inverse function theorem (Theorem 1) the following calculations can be done.

Theorem 1 *Inverse function theorem:*
If a function $F : \mathbb{R}^N \to \mathbb{R}^N$, with $N \in \mathbb{N}$ is continuously differentiable at a point p and the determinant of the Jacobian $J_F(p) \in \mathbb{R}^{N\times N}$ is nonzero, then an inverse function $F^{-1} : \mathbb{R}^N \to \mathbb{R}^N$ exists near a point $q = F(p)$. Also the derivative of the inverse F^{-1} at q: $J_{F^{-1}}(q) \in \mathbb{R}^{N\times N}$ exists and is equal to the inverse of $J_F(p)$. So $J_{F^{-1}}(q) = (J_F(p))^{-1}$.

Start with the original equation for ω:

$$0 = \det\left(\omega \frac{\partial F}{\partial y} - \frac{\partial G}{\partial y}\right).$$

Multiply both sides with $\det\left(\frac{\partial y}{\partial F}\right)$ to get:

$$\Leftrightarrow 0 = \det\left(\frac{\partial y}{\partial F}\right)\det\left(\omega \frac{\partial F}{\partial y} - \frac{\partial G}{\partial y}\right).$$

Using that $\det(AB) = \det(A)\det(B)$ we can state:

$$\Leftrightarrow 0 = \det\left(\left(\omega \frac{\partial F}{\partial y} - \frac{\partial G}{\partial y}\right)\frac{\partial y}{\partial F}\right).$$

The Inverse function theorem states $\frac{\partial y}{\partial F} = \left(\frac{\partial F}{\partial y}\right)^{-1}$, this implies:

$$\Leftrightarrow 0 = \det\left(\omega I - \frac{\partial G}{\partial F}\right).$$

So the solutions for ω are the eigenvalues of the Jacobian matrix $\frac{\partial G}{\partial F}$, which is the product of $\frac{\partial G}{\partial y}$ and $\frac{\partial y}{\partial F}$. To find $\frac{\partial y}{\partial F}$ use that $y = F^{-1}$ which is given by:

$$\begin{pmatrix} n \\ v \\ T \end{pmatrix} = \begin{pmatrix} F^p \\ F^m/F^p \\ \left(F^E - \frac{1}{2}(F^m)^2/F^p\right)/(3F^p) \end{pmatrix}.$$

Then its Jacobian is:

$$\frac{\partial y}{\partial F} = \begin{pmatrix} 1 & 0 & 0 \\ \frac{-F^m}{(F^p)^2} & \frac{1}{F^p} & 0 \\ \frac{-F^E}{3(F^p)^2} + \frac{(F^m)^2}{3(F^p)^3} & \frac{-F^m}{(F^p)^2} & \frac{1}{3F^p} \end{pmatrix} = \frac{1}{n} \begin{pmatrix} n & 0 & 0 \\ -v & 1 & 0 \\ \frac{v^2}{6} - T & \frac{-v}{3} & \frac{1}{3} \end{pmatrix}.$$

The eigenvalues of $\frac{\partial G}{\partial F}$ are:

$$\omega \in \left\{ v, v + \sqrt{\frac{10}{3}T}, v - \sqrt{\frac{10}{3}T} \right\}. \tag{2.2.12}$$

Since all variables are real and T is positive, the eigenvalues are real, hence the first order derivative part of the system of PDEs is hyperbolic.

To derive the required number of conditions for each boundary consider the convective terms and the diffusion terms separately. This is done because if either part of the system of PDEs goes to 0, the number of conditions imposed on the other part must not be influenced. The diffusion terms both require a condition on the left and right boundary. The number of conditions for the convective part is decided by the transport of "information" over the boundary. For flow equations, if information goes from the exterior to the interior then a condition is required. If information goes from the interior to the exterior then we can not set a condition on this boundary. Consider again the general solution $y = \hat{Y} e^{i(x-\omega t)}$, and assume $\hat{Y}$ to be constant. Then the solution is constant for $x - \omega t = c$ which is equivalent to $x = c + \omega t$. This implies that information travels in the direction of ω in space. Hence for the left boundary the transport of information is towards the interior if $\omega > 0$, on the right boundary if $\omega < 0$. So assuming that $v > 0$ and $v < \sqrt{\frac{10}{3}T}$ there should be 2 conditions at $x = 0$ and one at $x = 1$. (this is for the scaled equations, for the unscaled equations it would be at $x = L$) This assumption can be made because the conditions that are set on the boundaries will make it hold.

2.2.3 Benchmark model SD1D

The numerical model of Ben Dudson [3] is used to benchmark the numerical models of this thesis. His model is made using the numerical PDE solvers `Bout++` and `Sundials` which both allow for a variety of numerical schemes and simulations. The settings, parameters and rate-coefficients used are in Appendix B.3. It uses a different set of plasma equations and a domain parallel along the magnetic field lines, which were simplified for this thesis by taking the field lines as straight lines. The equations then become:

$$\frac{\partial n}{\partial t} = -\frac{\partial (nv)}{\partial x} - S, \tag{2.2.13}$$

$$\frac{\partial}{\partial t}\left(\frac{3}{2}p\right) = -\frac{\partial q}{\partial x} + v\frac{\partial p}{\partial x} - R - E, \tag{2.2.14}$$

$$\frac{\partial}{\partial t}(mnv) = -\frac{\partial}{\partial x}\left(mnv^2 + p\right) - F, \tag{2.2.15}$$

$$q = \frac{5}{2}pv - \kappa_{\parallel}^{e}\frac{\partial T}{\partial x}, \tag{2.2.16}$$

$$p = 2q_e nT. \tag{2.2.17}$$

The energy equation (2.2.14) only uses the internal energy instead of both the internal and kinetic energy. These equations are equivalent to those of Nakazawa, which is proven in Appendix B.2.

2.2.4 Boundary Conditions

The boundary conditions that are used in our models are given in this Section. Other conditions have been implemented as well in earlier models, these are discussed in Appendix C.2. First the neutral equation will be considered. This is a diffusion equation so on both boundaries a condition is needed.

- $\frac{\partial n_n}{\partial x} = 0$ for $x = 0$. So there is no neutral particle flux across the midplane, which is the origin in our domain. The midplane is indicated in Figure 2.4. This condition has little influence because the neutral particle density is negligible in the SOL.

- $D\frac{\partial n_n}{\partial z} = -nv\sin\theta$ for $x = L$. The plasma particles hit the target at an angle θ. So $nv\sin\theta$ is the particle flux into the target. At the target the ions receive an electron, changing them into neutral particles and then return to the system. So the neutral flux out of the target is equal to the ion flux into the target. The neutral flux out of the target is $-D\frac{\partial n_n}{\partial z}$, resulting in the condition $D\frac{\partial n_n}{\partial z} = -nv\sin\theta$. Expressing this condition in x-coordinates results in $D\frac{\partial n_n}{\partial x} = nv\sin^2\theta$. For later models a gas puff which injects neutral particles into the system at the target is included as well. The boundary condition then becomes $D\frac{\partial n_n}{\partial x} = nv\sin^2\theta + c_{\text{puff}}$.

According to Section 2.2.2 there should be two upstream conditions and one downstream for a subsonic inflow. It is assumed that by symmetry there is no particle and heat flow across the midpoint, because half of the particles goes to the divertor at the left and the other half to the divertor at the right. (In Figure 2.3 it can be seen that there are two divertors. Tokamaks can have more divertors, but that has no influence when simplifying the model to transport along magnetic field lines.) Hence the conditions for the first order derivatives at $x = 0$ are $v = 0$ and $\frac{\partial n}{\partial x} = 0$. The condition $v = 0$ will guarantee that the inflow is subsonic. Hence the condition $\frac{\partial T}{\partial x} = 0$ is only added to the heat diffusion term because three conditions at the left boundary on the first order derivatives will result in an over determined system according to Section 2.2.2.

At the target there are plasma surface interactions resulting in a plasma sheath. The results of this are implemented as two boundary conditions, the Bohm-criterion and the heat transmission to the target.
The Bohm-criterion [14, Chapter 2] is caused by the fact that atom nuclei are much heavier than electrons. So after the start-up of the tokamak the electrons hit the target before the nuclei do, giving the target a negatively charged sheath. This attracts the positively charged nuclei. If the ions hit the target with a velocity less than the ion thermal velocity $c_s = \sqrt{\frac{2Tq_e}{m}}$ the sheath will expand into the plasma, resulting in a larger force on the ions. If the ions reach the target with a velocity $v > c_s$ the sheath is pushed back, decreasing the negative potential. So at the target $v = c_s$.
The sheath heat transmission states [16, Chapter 9] that the power flux into the target is equal to $\gamma q_e c_s T n$ resulting in,

$$5q_e nvT + \frac{1}{2}mnv^3 - \kappa_\parallel^e \frac{\partial T}{\partial x} = \gamma q_e c_s T n.$$

Because of the earlier mentioned condition $v = c_s$ at $x = L$ the second condition for the heat diffusion becomes

$$(5-\gamma)q_e nvT + \frac{1}{2}mnv^3 = \kappa_\parallel^e \frac{\partial T}{\partial x}, \tag{2.2.18}$$

where γ is approximately 6.5 [4]. In Table 2.1 an overview is given of all the boundary conditions and their scaled versions that are implemented. In Appendix C.2 conditions that were implemented in earlier models are given.

Condition	Imposed on	Location	Scaled
$\frac{\partial n}{\partial x} = 0$	First derivatives	$x = 0$	$\frac{\partial n}{\partial x} = 0$
$v = 0$	First derivatives	$x = 0$	$v = 0$
$\frac{\partial T}{\partial x} = 0$	Heat diffusion	$x = 0$	$\frac{\partial T}{\partial x} = 0$
$\frac{\partial n_n}{\partial x} = 0$	Neutral particles	$x = 0$	$\frac{\partial n_n}{\partial x} = 0$
$v = \sqrt{\frac{2q_e T}{m}}$	First derivatives	$x = L$	$v = \sqrt{2T}$
$(5-\gamma)q_e nvT + \frac{1}{2}mnv^3 = \kappa_\parallel^e \frac{\partial T}{\partial x}$	Heat diffusion	$x = L$	$(5-\gamma)nvT + \frac{1}{2}nv^3 = \frac{\kappa_\parallel^e}{q_e \hat{x}\hat{v}\hat{n}} \frac{\partial T}{\partial x}$
$D\frac{\partial n_n}{\partial x} = nv\sin^2\theta + c_{puff}$	Neutral particles	$x = L$	$D\frac{\partial n_n}{\partial x} = \hat{x}\hat{v}nv\sin^2\theta + c_{puff}$

Table 2.1: Overview of the boundary conditions, in the last condition $c_{\text{puff}} \equiv c_{\text{puff}}/\hat{n}$.

2.2.5 Source terms and Coefficients

Nakazawa's source terms

First the source terms used by Nakazawa [4] are considered:
The particle, momentum, energy and neutral particle source terms are given below. The particle source reads:

$$s_n = nn_n\langle\sigma v\rangle_{\text{ion}} - n^2(\langle\sigma v\rangle_{\text{rre}} + \langle\sigma v\rangle_{3\text{re}}) + S_n, \tag{2.2.19}$$

where $\langle\sigma v\rangle_{\text{ion}}$, $\langle\sigma v\rangle_{\text{rre}}$ and $\langle\sigma v\rangle_{3\text{re}}$ are the ionisation, radiative recombination and three-body recombination rate coefficients. The rate coefficients are usually dependent on n and T. The term S_n is not used in [4]. This is a particle source representing the particles coming from the SOL. It is only non-zero before the x-point. The momentum source reads:

$$s_v = -mnv\left(\langle\sigma v\rangle_{\text{cx}} + n(\langle\sigma v\rangle_{\text{rre}} + \langle\sigma v\rangle_{3\text{re}})\right), \tag{2.2.20}$$

where $\langle\sigma v\rangle_{\text{cx}}$ is the charge exchange rate coefficient. The energy source

$$\begin{aligned} Q = &-\left(\frac{1}{2}mv^2 + 3q_eT\right)n^2\left(\langle\sigma v\rangle_{\text{rre}} + \langle\sigma v\rangle_{3\text{re}}\right) \\ &-\left(\frac{1}{2}mv^2 + \frac{3}{2}q_eT\right)nn_n\langle\sigma v\rangle_{\text{cx}} - \varepsilon nn_n\langle\sigma v\rangle_{\text{ion}} - W_{\text{imp}} + S_Q \end{aligned} \tag{2.2.21}$$

has a sink W_{imp}, which is the energy loss by impurity radiation. ε is the energy loss due to ionisation and excitation and is taken as 30 eV. Again a source term S_Q is added to represent the upstream influx of energy, which is only non-zero before the x-point. The neutral particle source is:

$$s_{n_n} = -\left(nn_n\langle\sigma v\rangle_{\text{ion}} - n^2(\langle\sigma v\rangle_{\text{rre}} + \langle\sigma v\rangle_{3\text{re}})\right) - n_{n,\text{loss}}n_n. \tag{2.2.22}$$

Ignoring the upstream ion source of equation (2.2.19) and the $n_{n,\text{loss}}n_n$ term, the source term for the neutral equations is equal to -1 times the source of the ions. Hence all the ions that are lost from the system return as neutral particles and vice-versa. The parameter $n_{n,\text{loss}}$ is the fraction of neutral particles that is lost per second. This is to stop the solution from exploding if there is a particle source upstream. If there is no particle source then $n_{n,\text{loss}} = 0$.

SD1Ds source terms

The source terms used by the SD1D model are almost identical. But the three-body recombination is combined with the recombination rate coefficient. In that model only the internal energy is used in the energy equation, instead of both the internal and kinetic energy. So to get the correct source terms for our model consider that

$$\frac{\partial}{\partial t}\left(\frac{1}{2}mnv^2\right) = v\frac{\partial}{\partial t}(mnv) - \frac{1}{2}mv^2\frac{\partial n}{\partial t}.$$

This is equal to v times the derivative w.r.t. the time of the momentum and $-\frac{1}{2}mv^2$ times the derivative w.r.t. the time of the particle density. So v times the momentum source and $-\frac{1}{2}mv^2$ times the particle source are added to the energy source term. Then the correct source term for the energy equation using equations (2.2.14), (2.2.15) and (2.2.13) is,

$$-R - E - vF + \frac{1}{2}mv^2 S.$$

For the values of R, E, F and S there are several options given in the SD1D model, the ones that are used are given in Appendix B.3. The difference in solution between using Nakazawa's source terms or those of SD1D were negligible.

Rate coefficients, impurity radiation and diffusion coefficients

Because we use the SD1D model as a benchmark model, the source terms and most of the coefficients are the same as those in SD1D. The rate coefficients and cooling rate of carbon are plotted in Figure 2.7.

For convenience we chose to use heuristic functions. There exist more accurate functions which are dependent on both n and T and fitted to experimental data. But the profiles of the heuristic functions are similar to these fitted functions for the particle densities used in our simulations. Moreover implementing the fitted functions would require copying many coefficients and also calculating the Jacobian of the function which will be needed for our numerical method. This would require a lot of time, whereas the main focus of this thesis is on the numerical scheme and not the source terms.

- The ionisation rate $\langle\sigma v\rangle_{\text{ion}}$ is the rate at which neutral particles ionise and become charged particles. It is given by the following function of T [3].

$$\langle\sigma v\rangle_{\text{ion}} = \begin{cases} 5.875\times 10^{-12}\cdot T^{-0.5151}\cdot 10^{-2.563/\log_{10}T} & \text{if } T \geq 20\text{eV} \\ 10^{-6}\cdot T^{-3.054}\cdot 10^{-15.72\exp(-\log_{10}T)+1.603\exp\left(-\log_{10}^2 T\right)} & \text{if } 1\text{eV} < T < 20\text{eV} \\ 7.638\times 10^{-21} & \text{if } T \leq 1\text{eV} \end{cases}$$

- Charge exchange is a reaction in which the charge of the ion is transferred to a neutral particle during a collision. Because the ions are much faster than the neutral particles this works as a drain on the momentum of the plasma. The rate coefficient is [3]:

$$\langle\sigma v\rangle_{\text{cx}} = \begin{cases} 10^{-14} & \text{if } T < 1\text{ eV} \\ 10^{-14}\cdot T^{1/3} & \text{if } T \geq 1\text{ eV} \end{cases}.$$

- The three body recombination rate $\langle\sigma v\rangle_{3\text{re}}$ is not used in SD1D, so only the recombination rate is defined. This is the rate at which protons and electrons recombine to create neutral particles, which is given by $\langle\sigma v\rangle_{\text{rec}} = 0.7 \cdot 10^{-19}\sqrt{\frac{13.6}{T}}$, [17].

- The impurity radiation is $W_{\text{imp}} = n^2 \xi L(T)$. It is caused by particles that are not part of the fuel, which is mainly carbon because of the carbon wall machines considered here. Therefore a function for the carbon cooling rate is used. ξ is the fraction of impure particles per ion, so there are ξn impurities in total. The carbon cooling rate is [3, Source code],

$$L(T) = 2 \cdot 10^{-31}\left(\frac{T}{10}\right)^3 \Big/ \left(1 + \left(\frac{T}{10}\right)^{4.5}\right).$$

- The heat diffusion coefficient is $\kappa_{\parallel}^e = 3.1 \times 10^4 \frac{T^{5/2}}{\log\Lambda}$, [3]. The Coulomb logarithm Λ is a constant such that its logarithm is approximately between 10 and 20, [18]. Our choice is $\log\Lambda = 10$.

- The diffusion coefficient for the neutral particles is ([3])

$$D = \frac{v_{th,n}^2}{n\langle\sigma v\rangle_{\text{cx}} + v_{th,n} n_n a_0}, \tag{2.2.23}$$

where T_n and $v_{th,n} = \sqrt{\frac{q_e T_n}{m}}$ are the temperature and thermal velocity of the neutral particles. a_0 is the area of the cross section of a neutral hydrogen atom. Assuming that $T_n = T_e$ and $v_{th,n} n_n a_0 \ll n\langle\sigma v\rangle_{cx}$ the diffusion coefficient can be taken as

$$D = \frac{q_e T}{mn\langle\sigma v\rangle_{\text{cx}}}, \tag{2.2.24}$$

which is also used in [4].

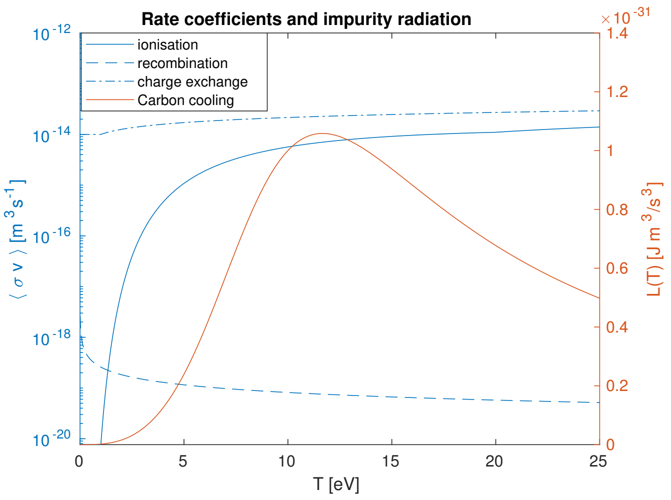


Figure 2.7: Plot of the rate coefficients and the cooling rate of carbon as functions of the plasma temperature.

With Figure 2.7 and 2.5 the cause of the plasma cooling down can also be explained. At $T \approx 13$ [eV] the carbon cooling rate has a peak, so the impurity radiation will be the main heat sink. Then the temperature decreases and ionisation will be the main heat sink followed by recombination. When recombination is dominant the neutral particles are created. The charge exchange acts mainly as a sink on the momentum of the flow, decreasing the ion velocity.

3 Numerical schemes

Three numerical schemes are constructed for this thesis, they are named: the Implicit scheme (Section 3.1), the Semi-Explicit scheme (Section 3.2) and the Flux-splitting scheme (Section 3.3). The first two are named for their time integration methods, the third is an abbreviation of ”flux-vector splitting scheme”.

3.1 The Implicit scheme

3.1.1 Time integration

The PDE contains double derivatives so for an explicit scheme this will result in a stability condition of the form $\left|c_1\frac{\Delta t}{\Delta x} + c_2\frac{\Delta t}{\Delta x^2}\right| < 1$, where $c_1, c_2 \in \mathbb{C}$ are not dependent on Δt and Δx. This implies that a small Δx requires a very small Δt to compensate for the dominant $\frac{1}{\Delta x^2}$ term. Therefore an implicit scheme is chosen for the time integration.

The time integration is implemented in two ways.

1. A self implemented Implicit Euler method. For each time step a nonlinear equation has to be solved, this is done using Matlabs `fsolve()`. The time step Δt is predefined but is halved if `fsolve()` does not converge. This is repeated until it does converge or reaches a minimal time step. If the minimal time step is reached the simulation is stopped because the time steps can not become smaller than the computing precision. The main reason for using Implicit Euler instead of a solver given by Matlab is its simplicity. This makes it more convenient to look at errors for each iteration separately and makes analysis of the discretization less complicated.

2. An ordinary differential equation solver of Matlab. The only fully implicit solver is `ode15i()`. It is based on the Backward Differentiation Formulas and uses a variable step size based on the relative and absolute error [19]. This proved to be faster than the self implemented Implicit Euler, so it is used for most computations. More common and higher order solvers like `ode45()` and `ode15s()` were not able to compute a solution. Our conjecture is that this is because these are not fully implicit methods [20].

3.1.2 Spatial discretization

First the grid that is used to discretize the spatial domain is defined, the same grid is used for all schemes in this report. Then the spatial derivatives and their discretizations are considered for several terms separately, depending on the discretization that is needed for the term.

The grid is defined as follows. Let N be the number of points in the domain. And define the grid precision $\Delta x = L/N$ where L is the scaled length of the domain. The first grid point x_1 is at $\Delta x/2$ and the last x_N at $L - \Delta x/2$. This grid is chosen because it makes the implementation of the boundary conditions for the diffusion term more convenient. The diffusion discretization (3.1.7) can be interpreted as

$$\frac{1}{\Delta x}\left(\alpha_{j+1/2}\frac{\partial u_{j+1/2}}{\partial x} - \alpha_{j-1/2}\frac{\partial u_{j-1/2}}{\partial x}\right).$$

At a boundary, for example for $j = 1$ this becomes

$$\frac{1}{\Delta x}\left(\alpha_{3/2}\frac{\partial u_{3/2}}{\partial x} - \alpha_{1/2}\frac{\partial u_{1/2}}{\partial x}\right),$$

where $\frac{\partial u_{1/2}}{\partial x} = \left.\frac{\partial u}{\partial x}\right|_{x=0}$. So setting a Neumann condition at this bound can be done by substitution.

Upwind scheme for the first order derivatives

A lot of terms are derivatives of a product of variables. For the particle and momentum conservation equations ((2.2.5) and (2.2.6)) the chain rule is used to get only one derivative per term. This made implementing the boundary conditions more straightforward. Each derivative is then approximated numerically. The energy flux in the energy equation (2.2.7) is discretized as a whole. This made an earlier implementation of the sheath heat transmission (2.2.18) more convenient. This implementation is explained in Section C.2.

An upwind scheme is required for most derivatives. Not doing so results in numerical oscillations. A toy problem is constructed to analyse this in more detail. The equations read:

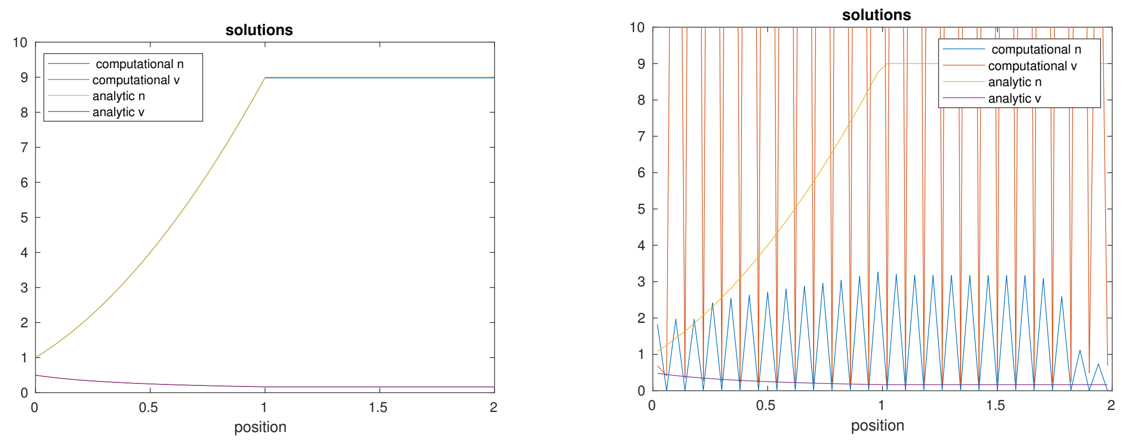
$$\begin{cases} \frac{\partial(nv)}{\partial x} = s_n(x) \\ \frac{\partial(nv^2)}{\partial x} = s_v(x) \\ v(0) = v_0 \\ n(0) = n_0 \end{cases}. \tag{3.1.1}$$

In this case both central difference and the upwind scheme were applied to approximate the spatial derivatives. The results are given in Figure 3.1.

Downwind scheme for the pressure term

For the first order derivatives there is one exception that does not use the upwind scheme. This is the pressure term $2nT$ in Equation (2.2.6) which requires the downwind scheme. Two toy problems are considered to demonstrate the cause of this problem, first for the upwind scheme, then for the central difference scheme, both schemes result in numerical oscillations. Problem 1 is Equation (3.1.2):

$$\begin{cases} \frac{\partial(nv)}{\partial x} & = c \\ \frac{\partial(nv^2)}{\partial x} + T\frac{\partial n}{\partial x} & = 0 \end{cases}, \tag{3.1.2}$$



(a) Upwind (b) Central difference

Figure 3.1: Numerical and analytic solutions to Equation (3.1.1). The vertical axes contain both n and v, which are both scaled so have no dimension.

where T and c are constants. Using the upwind scheme to discretize all the derivatives results in equations (3.1.3) and (3.1.4):

$$\frac{n_i - n_{i-1}}{\Delta x} v_i + \frac{v_i - v_{i-1}}{\Delta x} n_i = c, \tag{3.1.3}$$

$$T\frac{n_i - n_{i-1}}{\Delta x} + \frac{n_i - n_{i-1}}{\Delta x} v_i^2 + 2n_i v_i \frac{v_i - v_{i-1}}{\Delta x} = 0. \tag{3.1.4}$$

Subtracting $2v_i$ times (3.1.3) from (3.1.4) results in

$$T\frac{n_i - n_{i-1}}{\Delta x} - v_i^2 \frac{n_i - n_{i-1}}{\Delta x} = -2cv_i.$$

Then the following function for n_i can be derived

$$n_i = n_{i-1} + \frac{2v_i c \Delta x}{v_i^2 - T}.$$

If $v_i^2 < T$ (a subsonic flow as is usually the case for the solutions of Equations (2.2.5)-(2.2.8) in this report) then the particle flow would be in the opposite direction of the velocity, which makes the solution unfeasible.

For the central difference scheme consider the same toy problem but now with $c = 0$. This results in the following discretization:

$$\frac{n_i - n_{i-1}}{\Delta x} v_i + \frac{v_i - v_{i-1}}{\Delta x} n_i = 0, \tag{3.1.5}$$

$$T\frac{n_{i+1} - n_{i-1}}{2\Delta x} + \frac{n_i - n_{i-1}}{\Delta x} v_i^2 + 2n_i v_i \frac{v_i - v_{i-1}}{\Delta x} = 0. \tag{3.1.6}$$

The same way the following expression can be derived,

$$T\frac{n_{i+1} - n_{i-1}}{2\Delta x} - v_i^2 \frac{n_i - n_{i-1}}{\Delta x} = 0.$$

Substituting λ^i as a solution for n_i and dividing by λ^{i-1} results in a second order polynomial for λ which has two solutions, namely

$$\lambda_{1,2} = \frac{v_i^2 \pm \sqrt{v_i^4 - T(v_i^2 - T)}}{T}.$$

The solution is $n_i = c_1\lambda_1^i + c_2\lambda_2^i$. If $T > v_i^2$ then $\lambda_1 > 0 > \lambda_2$. This will make the λ_2 term switch sign for each iteration, creating numerical oscillations.

For the downwind scheme the solutions for λ would be:

$$\lambda_{1,2} = \frac{v_i^2 + T \pm \sqrt{(v_i^2 + T)^2 - 4Tv_i^2}}{2T} = \frac{v_i^2 + T \pm \sqrt{(v_i^2 - T)^2}}{2T},$$

which has no negative values for λ and therefore no numerical oscillations.

Direction of the upwind/downwind schemes

The direction of the upwind/downwind schemes should be dependent on the sign of the velocity v. But in this case it results in numerical errors such as in Figure 3.2. To investigate this error a toy problem has been constructed which is explained in Appendix C.6. The results of the toy problem also proved applicable for this system of PDEs. So the method did not appear to be unstable for an upwind/downwind scheme that is independent of the sign of v. Using this scheme also does not result in numerical errors, the result is seen in Figure 3.2. However when the velocity becomes too small the method will become unstable. A possible explanation why the method might me stable for $v < 0$ if $|v|$ is small is given in Appendix C.7.

Discretization of the diffusion terms

Both diffusion terms contain a diffusion coefficient that is dependent on the solution variables. To discretize this the following scheme is used because it is a second order scheme that only uses 3 grid points:

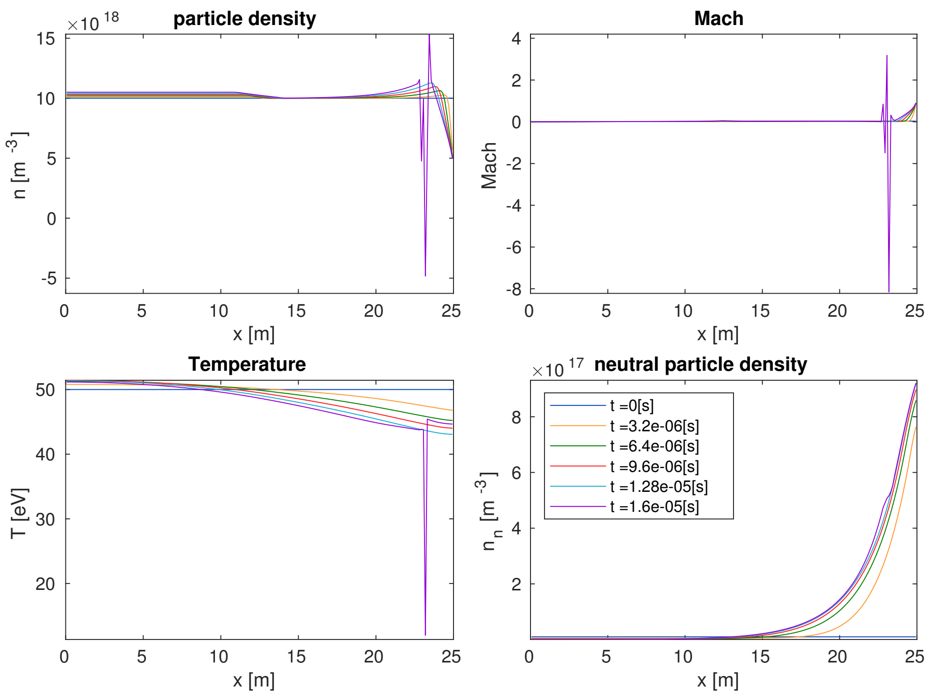
$$\frac{\partial}{\partial x}\left(\alpha \frac{\partial f}{\partial x}\right) = \frac{1}{\Delta x^2}\left(\alpha_{j+1/2}(f_{j+1} - f_j) - \alpha_{j-1/2}(f_j - f_{j-1})\right) + O(\Delta x^2), \tag{3.1.7}$$

where $\alpha_{j+1/2} = \frac{1}{2}(\alpha_{j+1} + \alpha_j)$ idem for $\alpha_{j-1/2}$. For all the different methods that have been implemented for the time integration and the first order spatial derivatives, this scheme was always able to compute a solution. In Appendix C.1 the second order convergence and well-posedness of the discretization are proven.

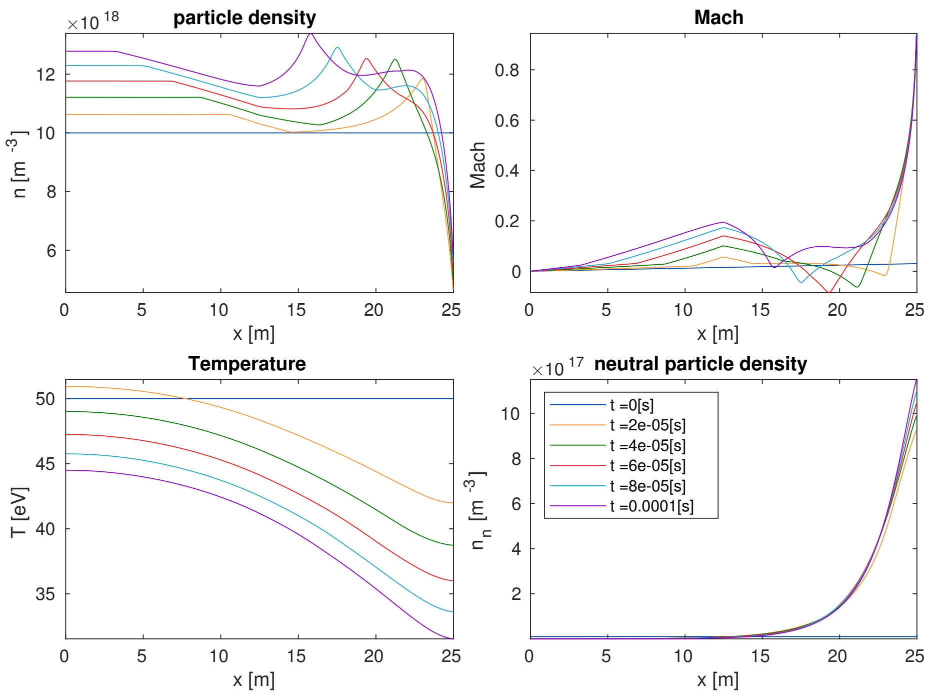
3.1.3 Boundary conditions

For this numerical scheme the same boundary conditions are used as given in Section 2.2.2. The conditions are implemented using a ghost point outside the domain. There are three types of physical boundary conditions: Dirichlet, Neumann and Robin conditions and there are numerical boundary conditions. To show how they are implemented consider the left side of the domain. The left boundary $x = 0$ is halfway between x_0 (x_0 is outside the domain) and x_1. So a Dirichlet condition $u(0) = \hat{u}$ is implemented as,

$$\frac{1}{2}(u_0 + u_1) = \hat{u} \qquad \Rightarrow \qquad u_0 = 2\hat{u} - u_1.$$



(a) Upwind/downwind direction dependent on sign of v.



(b) Upwind/downwind direction independent of sign of v.

Figure 3.2: Numerical solutions calculated with Implicit scheme. Comparing two versions of the Implicit scheme.

Similarly the Neumann condition $\frac{\partial u}{\partial x} = \hat{u}$ at $x = 0$ is implemented as,

$$\frac{u_1 - u_0}{\Delta x} = \hat{u} \qquad \Rightarrow \qquad u_0 = \Delta x \hat{u} + u_1.$$

The Bohm criterion is implemented using extrapolation at the boundary. First it has to be rewritten to $T(L) = v^2(L)/2$ because only upwind schemes are used for the terms containing the velocity, so there is no ghost point for v. Therefore $v^2(L)$ is approximated by $v^2(L) = 1.5v_N^2 - 0.5v_{N-1}^2 + O(\Delta x^2)$. The boundary condition then becomes

$$T_{N+1} = 2\frac{v^2(L)}{2} - T_N = (1.5v_N^2 - 0.5v_{N-1}^2) - T_N.$$

To demonstrate the implementation of the boundary conditions on the diffusion terms we take $D\frac{\partial n_n}{\partial x} = \hat{x}\hat{v}nv\sin^2\theta + c_{\text{puff}}$ as an example. This is approximated by

$$\frac{1}{2\Delta x}(D_N + D_{N+1})((n_n)_{N+1} - (n_n)_N) = \hat{x}\hat{v}(1.5n_Nv_N - 0.5n_{N-1}v_{N-1})\sin^2\theta + c_{\text{puff}},$$

which is substituted into Equation (3.1.7) at grid point N. The Robin condition at $x = L$ on the heat diffusion is implemented similarly.

Numerical boundary conditions are also implemented using extrapolation, this is done as follows:

$$u_{N+1} = 2u_N - u_{N-1} + O(\Delta x^2),$$

For the extrapolation of a nested function $f(g(x))$ we chose to use $1.5f(g(x-0.5\Delta x))-0.5f(g(x-1.5\Delta x))$ instead of $f(1.5g(x-0.5\Delta x)-0.5g(x-1.5\Delta x))$ because both are second order convergent and the first is easier to implement. The second order convergence for the first option is proven by substituting the Taylor expansion of $f \circ g$ around x, resulting in:

$$\begin{aligned}1.5&\left(f(g(x)) - \frac{\Delta x}{2}f'(g(x))g(x)' + \frac{\Delta x^2}{8}\left(f''(g(x))(g')^2 + g''(x)f(x)\right)\right)\\ &- 0.5\left(f(g(x)) - \frac{3\Delta x}{2}f'(g(x))g(x)' + \frac{9\Delta x^2}{8}\left(f''(g(x))(g(x)')^2 + g''(x)f(x)\right)\right) + O(\Delta x^3).\end{aligned}$$

This is equal to

$$f(g(x)) - \frac{3\Delta x^2}{8}\left(f''(g(x))(g'(x))^2 + g(x)''f'(x)\right) + O(\Delta x^3).$$

So the error is of order 2. The same can be done for the second option by substituting the Taylor expansion of g around x, resulting in:

$$\begin{aligned}f&\left(1.5\left(g(x) - \frac{\Delta x}{2}g'(x) + \frac{\Delta x^2}{8}g''(x) + O(\Delta x^3)\right)\right.\\ &\left.- 0.5\left(g(x) - \frac{3\Delta x}{2}g'(x) + \frac{9\Delta x^2}{8}g''(x) + O(\Delta x^3)\right)\right).\end{aligned}$$

This is equal to

$$f\left(g(x) - \frac{3\Delta x^2}{8}g''(x) + O(\Delta x^3)\right).$$

Taking the Taylor expansion of f around $g(x)$ results in

$$f(g(x)) - \frac{3\Delta x^2}{8}g''(x)f'(x) + O(\Delta x^3).$$

Hence both options are second order convergent. The boundary conditions for the other schemes are implemented in a similar fashion.

3.1.4 Results and discussion

The method is validated using the Method of Manufactured Solutions in Section 3.4.1. Before the plasma detaches the solution is smooth and the boundary conditions are met as well.

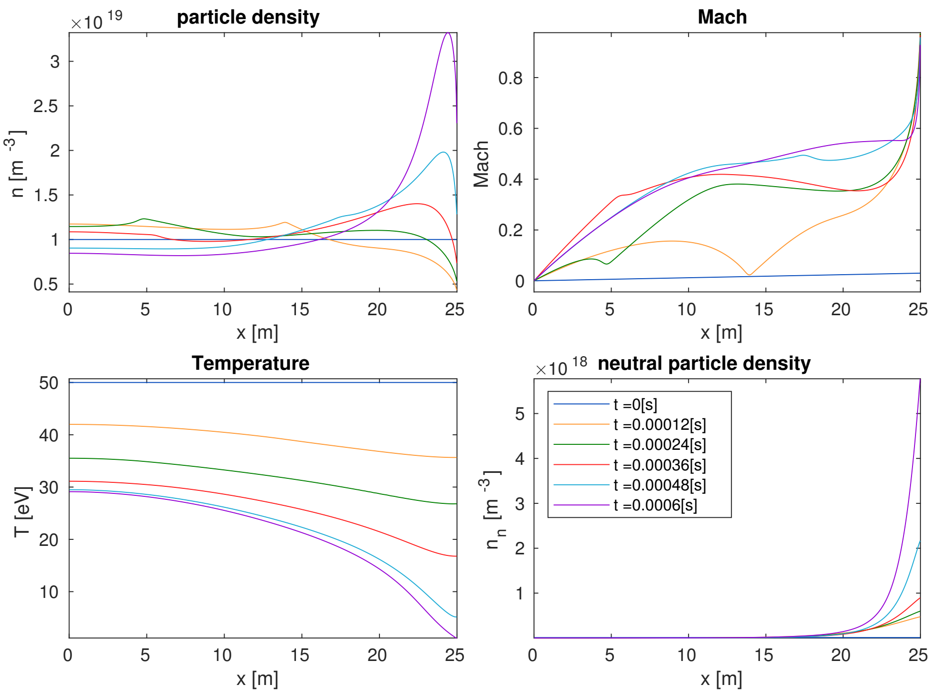
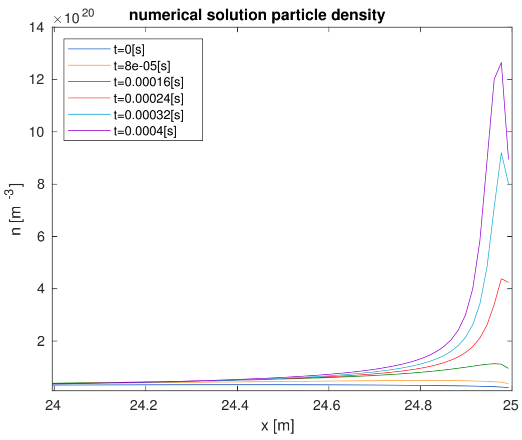


Figure 3.3: Numerical solution calculated with the Implicit scheme.

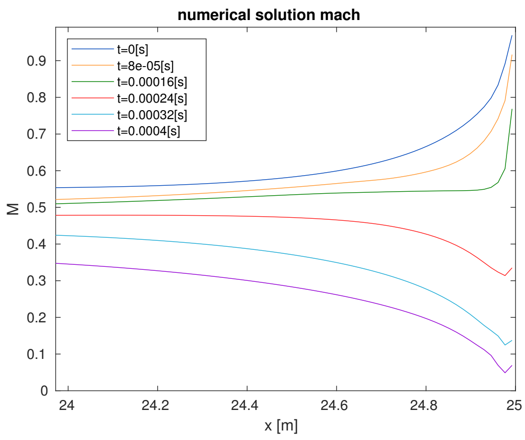
When the plasma detaches it becomes less smooth (Figure 3.4 and 3.5). At the right boundary v becomes less than $\sqrt{2T}$ so this boundary condition is not met anymore either. This is also the case in Nakazawa's paper [4], but not for the simulations with the SD1D model. The solutions for all 4 variables are more similar to those of Nakazawa than those of SD1D as well. A more elaborate comparison with the SD1D model will be done in Section 3.4.2.

When simulating detachment with a large number of grid points the numerical diffusion will be reduced, resulting in a sharper drop in v making it too small to meet the stability condition, as is explained in Appendix C.7. Hence for some times in the simulation the direction of the upwind/downwind scheme should be dependent on the sign of v and for others it should not. This makes using a flux-vector splitting scheme necessary for this PDE, because for this scheme the direction of the upwind/downwind schemes are based on the eigenvalues of the Jacobian of the system of PDEs.

The Implicit scheme could be improved by first rewriting the plasma equations to Equations (3.1.8) - (3.1.10). Then instead of calculating n, v and T from n, nv and $\frac{1}{2}nv^2 + 3nT$ for each time iteration and then solving Equations (2.2.5) - (2.2.7), the time integration can be done directly. This saves a lot of computational steps for each time iteration. The equations to be



(a) Numerical solution of the particle density.



(b) Numerical solution of the Mach number.

Figure 3.4: Solutions of the Implicit scheme when detachment occurs zoomed in at the target.

solved then become:

$$\frac{\partial n}{\partial t} + \frac{\partial (nv)}{\partial x} = s_n, \tag{3.1.8}$$

$$\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} + \frac{2}{n}\frac{\partial (nT)}{\partial x} = \frac{1}{n}(s_v - vs_n), \tag{3.1.9}$$

$$\frac{\partial T}{\partial t} + \frac{3}{2}T\frac{\partial v}{\partial x} + v\frac{\partial T}{\partial x} - \frac{1}{3n}\frac{\partial}{\partial x}\left(\kappa_\| \frac{\partial T}{\partial x}\right) = \frac{1}{3n}\left(s_Q - vs_v + \left(\frac{1}{2}v^2 - 3T\right)s_n\right). \tag{3.1.10}$$

A derivation similar to the derivation of these equations is in Section 3.3.

3.2 Semi-Explicit numerical scheme

The Implicit scheme requires solving a set of non-linear equations for each time step. To solve a non-linear equation an iterative method is used that does require solving a system of the form $Ay = b$ for each iteration. Where A is a matrix, y the solution variable and b a vector. If the time integration is not completely implicit it can be reduced to solving equations of the form $y_{n+1} = f(y_n)$ or $y_{n+1} = A_n y_{n+1} + b_n$. So a substitution in a formula or only one matrix vector equation to calculate the next solution for each time step, so less computations are needed. Because of the diffusion terms with large diffusion coefficients, meeting the stability condition for an explicit time integration requires a time step smaller than the computing precision, according to the `ode45()` function, so a completely explicit time integration is not possible. Thus a semi-explicit model is considered. In this model it is also attempted to implement the Bohm-criterion as a condition on the velocity instead of implementing it as condition on the temperature or a source term (Appendix C.2).

3.2.1 Spatial derivatives

Several schemes have been investigated for the derivatives. First the schemes for which only one attempt was needed are considered.

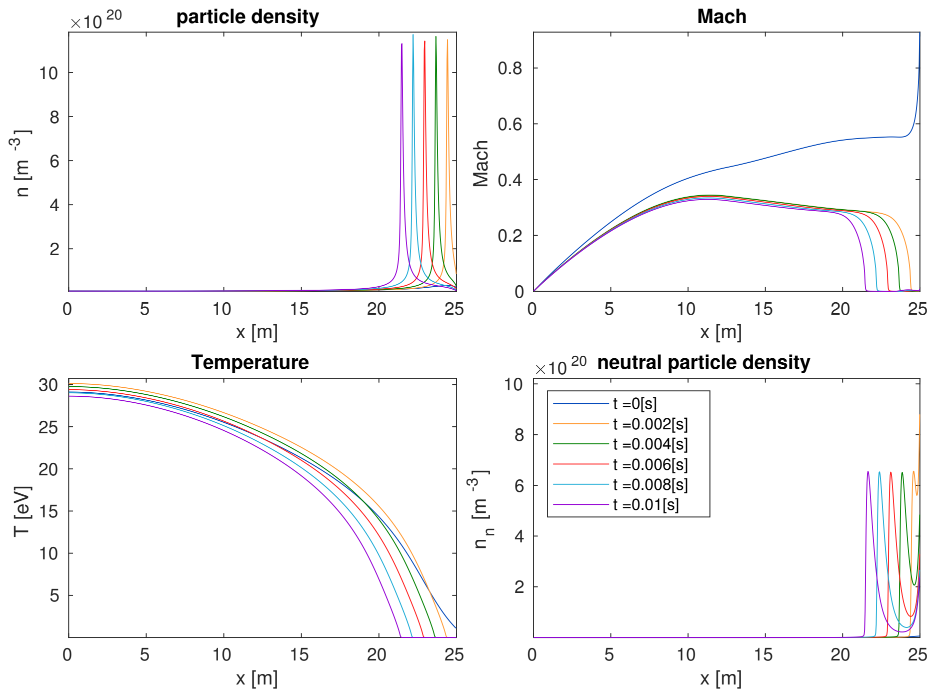


Figure 3.5: Travelling detachment front.

Discretization of the diffusion terms

These are discretized in the same manner as in the Implicit scheme (3.1.7). The coefficients however are from time t^k instead of t^{k+1}, making it a matrix vector product instead of a nonlinear equation. So the scheme becomes

$$\left(\frac{\partial}{\partial x}\left(\alpha\frac{\partial f}{\partial x}\right)\right)_j^{k+1} = \frac{1}{\Delta x^2}\Big(\alpha_{j+1/2}^k(f_{j+1}^{k+1}-f_j^{k+1}) - \alpha_{j-1/2}^k(f_j^{k+1}-f_{j-1}^{k+1})\Big) + O(\Delta x^2). \tag{3.2.1}$$

Discretization of the energy flux

To make it more convenient to implement the sheath heat transmission for an earlier model (Appendix C.2) the derivative of the entire energy flux is discretized with central difference without using the product rule first, resulting in:

$$\frac{\partial}{\partial x}\left(\frac{1}{2}nv^3+5nvT\right)_j = \frac{1}{2\Delta x}\left(\left(\frac{1}{2}nv^3+5nvT\right)_{j+1} - \left(\frac{1}{2}nv^3+5nvT\right)_{j-1}\right). \tag{3.2.2}$$

Discretization of the particle and momentum conservation equations

The initial goal of the Semi-explicit scheme was to implement the Bohm-criterion $v(L) = \sqrt{2T(L)}$ as a condition on the velocity instead of the temperature. Thus using an upwind scheme for the entire domain was not an option. Several schemes have been investigated to find one that allows setting a condition on the right boundary without resulting in numerical oscillations. For each method the results are in Appendix C.5.

- The first method uses central difference for all derivatives. This gives second order convergence and ghost cells at both boundaries. Making it possible to implement the Bohm-criterion on $v(L)$. However the method gives numerical oscillations, which can be seen in Figure C.6 in Appendix C.5.
- For the second method a mix of upwind, central difference and downwind is used.
  - Upwind is used for $v\frac{\partial n}{\partial x}$ from Equation (2.2.5) and $2nv\frac{\partial v}{\partial x}$ from Equation (2.2.6). The four PDEs are solved separately each time step. To compute the next solution of n the particle conservation equation is integrated and to compute v the momentum equation. Upwind is used for these terms because $v\frac{\partial n}{\partial x}$ is a term in the equation to calculate n and $2nv\frac{\partial v}{\partial x}$ in the equation to calculate v, so these are the convective terms of the equation.
  - Central difference is used for $n\frac{\partial v}{\partial x}$ from Equation (2.2.5) and $v^2\frac{\partial n}{\partial x}$ from Equation (2.2.6).
  - Downwind is used for the pressure term $2\left(T\frac{\partial n}{\partial x} + n\frac{\partial T}{\partial x}\right)$ because this worked for the Implicit scheme and it is not needed to set a condition at the left boundary.

  In Appendix C.5.2 it is shown that the mix of central difference and upwind results in numerical diffusion which is most likely the reason that the numerical oscillations were damped and occurred less soon. However they are not completely gone as can be seen in Figure C.7 in Appendix C.5. So this scheme is also unfeasible.
- For the third method the biased-upwind scheme is implemented for every term but the pressure term, which is discretized using a downwind scheme. The biased upwind scheme is:
$$\left(\frac{\partial f}{\partial x}\right)_j = \frac{2f_{j+1} + 3f_j - 6f_{j-1} + f_{j-2}}{6\Delta x} + O(\Delta x^3). \tag{3.2.3}$$
  It is a combination of the second order upwind scheme and central difference resulting in a third order scheme. We chose to implement it because we presume it to be less prone to numerical oscillations than the central difference scheme as is explained more elaborately in Appendix C.5. The implementation of the boundary condition is different from those for the other schemes in this thesis because at the first grid point there are two ghost points f_0 and f_{-1}. The Dirichlet condition $f(0) = 0$ is implemented by setting $f_{-1} - 6f_0 = -(-6f_1 + f_2)$ for the derivative at $j = 1$ and $f_0 = -f_1$ for $j = 2$. The Neumann condition $\frac{\partial f}{\partial x} = 0$ at $x = 0$ is implemented setting $f_{-1} - 6f_0 = -6f_1 + f_2$ for $j = 1$ and $f_0 = f_1$ for $j = 2$. This method however also gives numerical oscillations as can be seen in Figure C.8 in Appendix C.5.
- The only option that appears to be stable is to use an upwind scheme for the convective terms and downwind for the pressure term. To still implement the Bohm-criterion as a condition on the velocity, the derivative at the last grid point is calculated by using both the upwind and central difference combined using a weight $\zeta \in (0, 1)$,
$$\left(\frac{\partial v}{\partial x}\right)_N = \zeta\frac{v_N - v_{N-1}}{\Delta x} + (1-\zeta)\left(\frac{-v_N - v_{N-1}}{2\Delta x} + \frac{c_s}{\Delta x}\right).$$
  If $\zeta = 1$ then the upwind scheme is used, if $\zeta = 0$ the central difference is used and the Bohm-criterion is implemented. For $\zeta = 0.9$ there is still a numerical error at the last grid

point. However this did not influence the rest of the domain. The Bohm-criterion is not satisfied though, thus this method is considered unfeasible as well.

- The same discretization is used as in the Implicit scheme. The complete set of difference Equations is given by (3.2.4)-(3.2.7). The condition $v(L) = \sqrt{2T(L)}$ is implemented as a condition on T in the pressure term which uses a downwind scheme.

3.2.2 Time Integration

For the time integration both an implicit and an explicit method is used. The heat diffusion term has a large diffusion coefficient which would require a really small time step to make the method stable. So the time integration of the heat diffusion term is implicit. The neutral diffusion coefficient is smaller so this does not require a fully implicit scheme. The time integration for all first order derivatives is done explicitly. For each iteration all solution variables are calculated using those from the previous step. The methods used per solution variable are given below:

- n^{k+1} is calculated by solving Equation (3.2.4) for the interval $[t_k, t_{k+1}]$ with `ode45()`.
- v^{k+1} is calculated by solving Equation (3.2.5) for the interval $[t_k, t_{k+1}]$ with `ode45()`. The derivative $\frac{\partial n}{\partial t}$ is approximated by $\frac{n^{k+1}-n^k}{\Delta t}$.
- T^{k+1} is calculated by solving Equation (3.2.6) for the interval $[t_k, t_{k+1}]$ with `ode15i()`. For the convective part the solution at time t^k is used and the derivatives $\frac{\partial n}{\partial t}$ and $\frac{\partial (nv^2)}{\partial t}$ are approximated using the solutions from t^k and t^{k+1}.
- $(n_n)^{k+1}$ is calculated by solving Equation (3.2.7) with `ode15s()` instead of `ode15i()`. This is not completely implicit, but the diffusion coefficient was small enough to use `ode15s()` which has better precision [20].

For all source terms the values of the previous time step are used. The complete set of difference equations is given by Equations (3.2.4) to (3.2.7).
Particle conservation:

$$\left(\frac{\partial n_j}{\partial t}\right) = -\left(v_j^k \frac{n_j - n_{j-1}}{\Delta x} + n_j \frac{v_j^k - v_{j-1}^k}{\Delta x}\right) + (s_n)_j^k . \tag{3.2.4}$$

Momentum conservation:

$$\begin{aligned} n_j^{k+1}\left(\frac{\partial v_j}{\partial t}\right) + v_j \frac{n_j^{k+1} - n_j^k}{\Delta t} = -\Bigg(& v_j^2 \frac{n_j^k - n_{j-1}^k}{\Delta x} + 2n_j^k v_j \frac{v_j - v_{j-1}}{\Delta x} + 2n_j^k \frac{T_j^k - T_{j-1}^k}{\Delta x} \\ & + 2T_j^k \frac{n_j^k - n_{j-1}^k}{\Delta x}\Bigg) + (s_v)_j^k . \end{aligned} \tag{3.2.5}$$

Energy conservation:

$$\begin{aligned} & 3n_j^{k+1}\left(\frac{\partial T_j}{\partial t}\right) + 3\frac{n_j^{k+1} - n_j^k}{\Delta t} T_j + \frac{\left(\frac{1}{2}nv^2\right)_j^{k+1} - \left(\frac{1}{2}nv^2\right)_j^k}{\Delta t} \\ & = -\frac{\left(\frac{1}{2}nv^3 + 5nvT\right)_{j+1}^k - \left(\frac{1}{2}nv^3 + 5nvT\right)_{j-1}^k}{2\Delta x} \\ & \quad + \frac{1}{\Delta x^2}\Big(\kappa_{j+1/2}^k (T_{j+1} - T_j) - \kappa_{j-1/2}^k (T_j - T_{j-1})\Big) + Q_j^k . \end{aligned} \tag{3.2.6}$$

Neutral particle diffusion:

$$\left(\frac{\partial (n_n)_j}{\partial t}\right) = \frac{1}{\Delta x^2}\left(D^k_{j+1/2}((n_n)_{j+1} - (n_n)_j) - D^k_{j-1/2}((n_n)_j - (n_n)_{j-1})\right) + (s_{n_n})^k_j. \tag{3.2.7}$$

Superscript k means that the solution at time t^k is used, superscript $k+1$ is for the solution at time t^{k+1}. If no superscript is given then that variable is updated by the ode solvers for each iteration on the interval $(t^k, t^{k+1}]$. The time derivatives that remain in the schemes are integrated by solvers from Matlab.

3.2.3 Results

The Semi-Explicit scheme is validated by the Method of Manufactured Solutions in Section 3.4.1. For small enough Δt the solution is smooth, for bigger time steps it becomes unstable. An example of a smooth solution is given in Figure 3.6. It appears as if the condition $T(L) = v(L)^2/2$ is not satisfied. But this is caused by the low number of grid points and the steep slope of v at $x = L$.

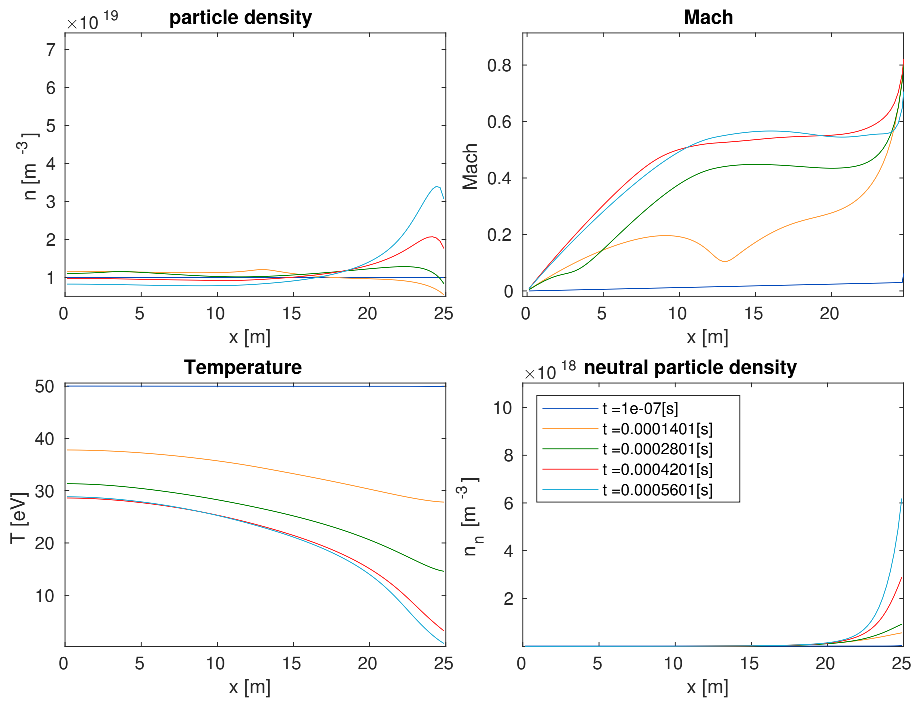


Figure 3.6: Solution for the Semi-Explicit scheme.

When the plasma detaches (Figure 3.7), the same phenomenon as with the Implicit scheme occurs. The detachment front keeps moving towards the midpoint and the transition from an attached to a detached plasma also shows the same peak in n as is the case with the Implicit scheme (Figure 3.8).
The advantage of the Semi-Explicit time integration is that there are no non-linear equations that have to be solved. The calculation of the next time step is faster as can be seen in Figure 3.15. However the stability condition requires the time steps to be very small. Because $\frac{\partial n}{\partial t}$ and $\frac{\partial}{\partial t}\left(\frac{1}{2}nv^2\right)$ are approximated using Forward Euler the stability condition is expected to be similar to that of Forward Euler. This also implies that increasing the number of grid points will

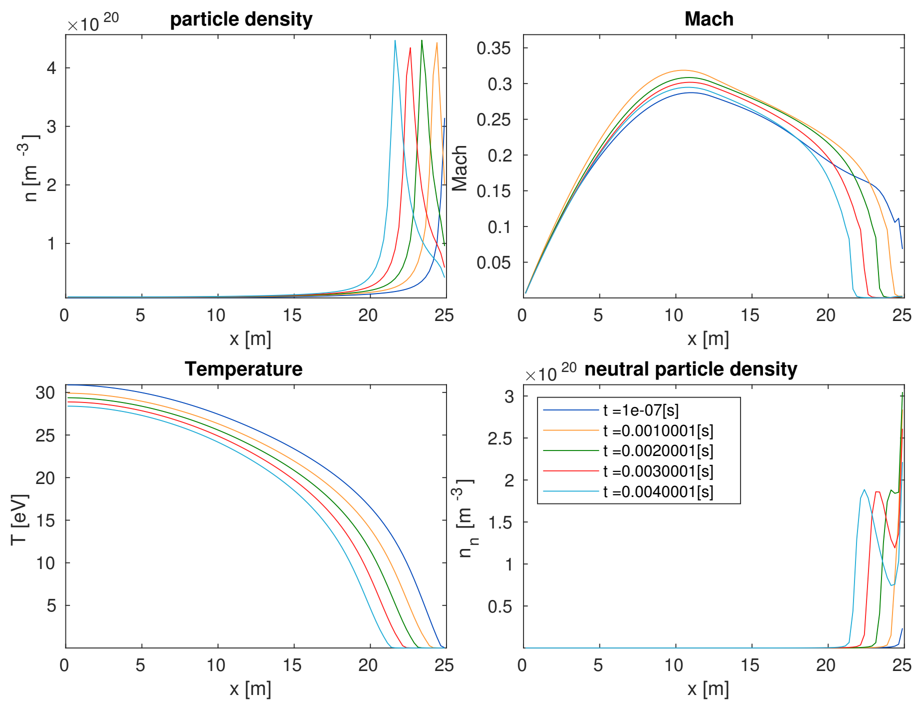


Figure 3.7: Solution for the Semi-Explicit scheme.

make the time integration even slower. Although there are several improvements possible for this scheme, we do not expect it to become as fast or achieve the same precision as the Implicit scheme.

The Semi-Explicit scheme can be improved in the same way as the Implicit scheme. By using Equations (3.1.8) - (3.1.10) instead of the original Equations (2.2.5)-(2.2.7) we remove the need to approximate $\frac{\partial n}{\partial t}$ and $\frac{\partial v}{\partial t}$ with Explicit Euler. Doing so will remove these first order approximations, possibly giving better precision. Moreover the stability condition of Explicit Euler is also very strict so without it the stability condition will be better, possibly allowing larger time steps. But most importantly it would have made it possible to do the time integration for all four equations simultaneously instead of using four subroutines.

We did not calculate the stability condition for the time integration. With a stability condition a maximal time step size can be found. Using the maximal time step will increase the computational speed without making the method unstable. The current time step is chosen based on experiences with the scheme.

We did not implement these improvements because we do not expect the Semi-Explicit scheme to become as fast as the Implicit scheme for the same precision. So we find it unnecessary to invest more time in improving this method.

3.3 Flux-splitting scheme

The sign of the velocity and the Mach number define the direction of the upwind and downwind schemes. So to rule out numerical errors caused by using the wrong upwind and downwind

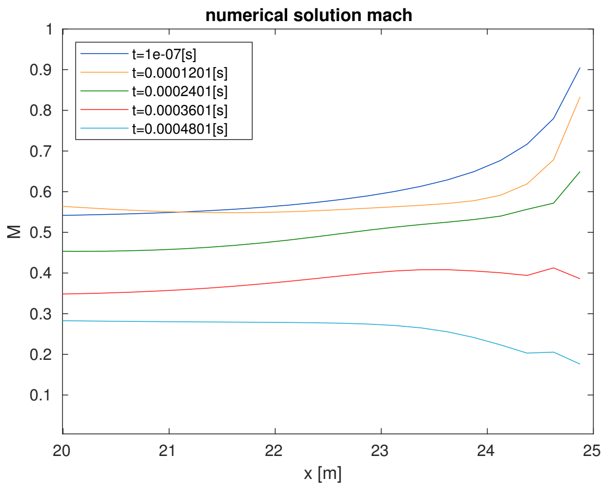


Figure 3.8: Solution for the Semi-Explicit scheme, zoomed in at the target.

direction in the Implicit and the Semi-Explicit scheme, the Flux-splitting scheme is implemented. If the propagation of information is in the positive direction then upwind should be used, if it is in the negative direction, downwind should be used. The direction of the propagation of information is defined by the sign of the eigenvalues of the Jacobian of the system. The Flux-splitting scheme uses the eigenvalues to set the upwind/downwind directions. It is based on the flux-vector splitting scheme of the Euler equations in [21, Chapter 20].

3.3.1 Flux-vector splitting of the first order spatial derivatives

Only consider the first order spatial derivatives because diffusion is symmetric and does not require an upwind or downwind scheme. Use the same definitions for F, G and y as in Section 2.2.2 and write the system of equations as,

$$\frac{\partial F(y)}{\partial t} + \frac{\partial G(y)}{\partial x} = 0.$$

This is equivalent to

$$\frac{\partial F}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial G}{\partial y}\frac{\partial y}{\partial x} = 0, \tag{3.3.1}$$

where $\frac{\partial F}{\partial y}$ and $\frac{\partial G}{\partial y}$ are the following Jacobian matrices:

$$\frac{\partial F}{\partial y} = \begin{pmatrix} 1 & 0 & 0 \\ v & n & 0 \\ v^2/2 + 3T & vn & 3n \end{pmatrix}, \tag{3.3.2}$$

$$\frac{\partial G}{\partial y} = \begin{pmatrix} v & n & 0 \\ v^2 + 2T & 2nv & 2n \\ v^3/2 + 5vT & 3/2v^2 + 5nT & 5nv \end{pmatrix}. \tag{3.3.3}$$

Multiply (3.3.1) with the inverse of $\frac{\partial F}{\partial y}$ to remove the matrix in front of $\frac{\partial y}{\partial t}$. Define A as the matrix product $\left(\frac{\partial F}{\partial y}\right)^{-1}\frac{\partial G}{\partial y}$. Then A is equal to

$$A = \frac{1}{n}\begin{pmatrix} n & 0 & 0 \\ -v & 1 & 0 \\ v^2/6 - T & -v/3 & 1/3 \end{pmatrix}\begin{pmatrix} v & n & 0 \\ v^2+2T & 2nv & 2n \\ v^3/2+5vT & 3/2v^2+5nT & 5nv \end{pmatrix} = \begin{pmatrix} v & n & 0 \\ 2T/n & v & 2 \\ 0 & \frac{2}{3}T & v \end{pmatrix}.$$

A can be written as a product of a left eigenvector matrix V^L, an eigenvalue matrix Λ and a right eigenvector matrix V^R:

$$\Lambda = \begin{pmatrix} v & 0 & 0 \\ 0 & v+\sqrt{\frac{10}{3}T} & 0 \\ 0 & 0 & v-\sqrt{\frac{10}{3}T} \end{pmatrix},$$

$$V^R = \begin{pmatrix} -n & 3n & 3n \\ 0 & 3\sqrt{\frac{10}{3}T} & -3\sqrt{\frac{10}{3}T} \\ T & 2T & 2T \end{pmatrix},$$

$$V^L = \begin{pmatrix} \frac{-2}{5n} & 0 & \frac{3}{5T} \\ \frac{1}{10n} & \frac{1}{6\sqrt{10/3T}} & \frac{1}{10T} \\ \frac{1}{10n} & \frac{-1}{6\sqrt{10/3T}} & \frac{1}{10T} \end{pmatrix}.$$

Two sets of eigenvalues are defined, positive and negative eigenvalues. The diagonal matrix Λ^+ contains all the positive eigenvalues and Λ^- the negative ones, such that $\Lambda^+ + \Lambda^- = \Lambda$. A^+ and A^- are defined as $V^R\Lambda^+V^L$ and $V^R\Lambda^-V^L$ respectively. Then Equation (3.3.1) becomes

$$\frac{\partial y}{\partial t} + A^+\frac{\partial y}{\partial x} + A^-\frac{\partial y}{\partial x} = 0. \tag{3.3.4}$$

To discretize $\frac{\partial y}{\partial x}$ both the upwind and downwind schemes are used. Upwind for $A^+\frac{\partial y}{\partial x}$ and downwind for $A^-\frac{\partial y}{\partial x}$.

3.3.2 Rewriting the complete set of equations

The full set of equations also has a source term and two diffusion terms. Using the same notation as before, Equations (2.2.5)-(2.2.7) can be written as

$$\frac{\partial F}{\partial y}\frac{\partial}{\partial t}\begin{pmatrix} n \\ v \\ T \end{pmatrix} + \frac{\partial G}{\partial y}\frac{\partial}{\partial x}\begin{pmatrix} n \\ v \\ T \end{pmatrix} - \frac{\partial}{\partial x}\begin{pmatrix} 0 \\ 0 \\ \kappa_{\parallel}^e\frac{\partial T}{\partial x} \end{pmatrix} = \begin{pmatrix} s_n \\ s_v \\ s_Q \end{pmatrix}.$$

Multiplying with $\left(\frac{\partial F}{\partial y}\right)^{-1}$ results in

$$\frac{\partial}{\partial t}\begin{pmatrix} n \\ v \\ T \end{pmatrix} + A\frac{\partial}{\partial x}\begin{pmatrix} n \\ v \\ T \end{pmatrix} - \left(\frac{\partial F}{\partial y}\right)^{-1}\frac{\partial}{\partial x}\begin{pmatrix} 0 \\ 0 \\ \kappa_{\parallel}^e\frac{\partial T}{\partial x} \end{pmatrix} = \left(\frac{\partial F}{\partial y}\right)^{-1}\begin{pmatrix} s_n \\ s_v \\ s_Q \end{pmatrix}.$$

The neutral equation is not affected by the Flux-splitting scheme because it does not contain any first order derivatives. So the entire set of equations that has to be solved is,

$$\frac{\partial}{\partial t}\begin{pmatrix} n \\ v \\ T \\ n_n \end{pmatrix} + \begin{pmatrix} A\frac{\partial}{\partial x}\begin{pmatrix} n \\ v \\ T \end{pmatrix} \\ 0 \end{pmatrix} - \begin{pmatrix} \left(\frac{\partial F}{\partial y}\right)^{-1}\frac{\partial}{\partial x}\begin{pmatrix} 0 \\ 0 \\ \kappa_{\parallel}^{e}\frac{\partial T}{\partial x} \end{pmatrix} \\ \frac{\partial}{\partial x}\left(D\frac{\partial n_n}{\partial x}\right) \end{pmatrix} = \begin{pmatrix} \left(\frac{\partial F}{\partial y}\right)^{-1}\begin{pmatrix} s_n \\ s_v \\ s_Q \end{pmatrix} \\ s_{n_n} \end{pmatrix}. \tag{3.3.5}$$

3.3.3 Implementation of the numerical scheme

To implement Equation (3.3.4) for every grid point the following notation is introduced:

$$A^{+} = \begin{pmatrix} A_1^{+} & & \\ & \ddots & \\ & & A_N^{+} \end{pmatrix},$$

where A_i^{+} is a 3×3 matrix for $i \in \{1, ..., N\}$ and

$$\underline{y} = \begin{pmatrix} \underline{n} \\ \underline{v} \\ \underline{T} \end{pmatrix}, \text{ where } \underline{n} = \begin{pmatrix} n_1 \\ \vdots \\ n_N \end{pmatrix} = \begin{pmatrix} n(x_1) \\ \vdots \\ n(x_N) \end{pmatrix}.$$

Similar notations hold for A^{-}, V^{R}, Λ^{+}, V^{L}, Λ^{-}, $\underline{v}$, $\underline{T}$ and $\underline{n}_n$. For each grid point $i \in \{1, ..., N\}$, A_i^{+} is the product $V_i^{R}\Lambda_i^{+}V_i^{L}$. So A^{+} is the product $V^{R}\Lambda^{+}V^{L}$ because these are all block diagonal matrices. A permutation matrix P is needed to reorder $\underline{y}$ to $(n_1; v_1; T_1; ...; y_N; v_N; T_N)$. The inverse of P is again needed to reorder the matrix products back for the source and diffusion terms to be added. So the semi discrete version of Equation (3.3.4) becomes

$$\frac{\partial \underline{y}}{\partial t} + P^{-1}\left(A^{+}PD_{\text{upw}}\underline{y} + A^{-}PD_{\text{dw}}\underline{y}\right) = 0, \tag{3.3.6}$$

where D_{upw} and D_{dw} are the upwind and downwind discretization matrices. The diffusion and source terms are simply added and do not require any permutations. So the discrete version of the complete set of equations (3.3.5) becomes

$$\begin{gathered} \frac{\partial}{\partial t}\begin{pmatrix} \underline{n} \\ \underline{v} \\ \underline{T} \\ \underline{n}_n \end{pmatrix} + \begin{pmatrix} P^{-1}\left(A^{+}PD_{\text{upw}}\begin{pmatrix} \underline{n} \\ \underline{v} \\ \underline{T} \end{pmatrix} + A^{-}PD_{\text{dw}}\begin{pmatrix} \underline{n} \\ \underline{v} \\ \underline{T} \end{pmatrix}\right) \\ 0 \end{pmatrix} - \\ \begin{pmatrix} \left(\frac{\partial \underline{F}}{\partial \underline{y}}\right)^{-1}\begin{pmatrix} 0 \\ 0 \\ D_{\kappa}\underline{T} \end{pmatrix} \\ D_{D}\underline{n}_n \end{pmatrix} = \begin{pmatrix} \left(\frac{\partial \underline{F}}{\partial \underline{y}}\right)^{-1}\begin{pmatrix} \underline{s}_n \\ \underline{s}_v \\ \underline{s}_Q \end{pmatrix} \\ \underline{s}_{n_n} \end{pmatrix}, \end{gathered} \tag{3.3.7}$$

where D_{κ} and D_D are the discretizations of the diffusion terms. $\frac{\partial \underline{F}}{\partial \underline{y}}$ is the discrete Jacobian of $\underline{F}$ w.r.t. $\underline{y}$. The analytic definition of $\left(\frac{\partial F}{\partial y}\right)^{-1}$ is,

$$\left(\frac{\partial F}{\partial y}\right)^{-1} = \frac{1}{n}\begin{pmatrix} n & 0 & 0 \\ -v & 1 & 0 \\ v^2/6 - T & -v/3 & 1/3 \end{pmatrix}.$$

So $\left(\frac{\partial F}{\partial \underline{y}}\right)^{-1}$ is implemented as a block matrix consisting of nine $N \times N$ diagonal matrices:

$$\begin{pmatrix}
1 & & & & & & & & \\
 & \ddots & & & & & & & \\
 & & 1 & & & & & & \\
-v_1/n_1 & & & 1/n_1 & & & & & \\
 & \ddots & & & \ddots & & & & \\
 & & -v_N/n_N & & & 1/n_N & & & \\
\frac{v_1^2/6-T_1}{n_1} & & & -v_1/3/n_1 & & & 1/3/n_1 & & \\
 & \ddots & & & \ddots & & & \ddots & \\
 & & \frac{v_N^2/6-T_N}{n_N} & & & -v_N/3/n_N & & & 1/3/n_N
\end{pmatrix}. \tag{3.3.8}$$

The time integration is done by Matlabs `ode15i()`. So to reduce the running time and improve the convergence to a solution for each time step the analytic Jacobian of Equation (3.3.7) is implemented. Its calculation and implementation are explained in Appendix C.3.

3.3.4 Results and discussion

The scheme is validated in Section 3.4.1 with the Method of Manufactured Solutions. The solutions are similar to those of the Implicit scheme and the Semi-Explicit scheme (Figure 3.9), a more in depth comparison is in Section 3.4.2. The most noticeable difference is the peak in the velocity right of the detachment front. This peak is also visible in the detached plasma in Nakazawa's paper. Our conjecture is that it is either caused by a strong gradient in the pressure which accelerates the ions, followed by a peak in the charge exchange. The charge exchange acts as friction and decelerates the particles. For the other models the pressure gradient is not as sharp, so these do not have this peak in v. Or it is caused by v and T being close enough to 0 for numerical errors to change the sign of the eigenvalues, resulting in the wrong choice for upwind and downwind. The velocity v is very small though so our conjecture is that this error is negligible. In Figure 3.10 an example is plotted, the peak in v is between the drop in pressure and the peak in charge exchange.

3.4 Validation and verification

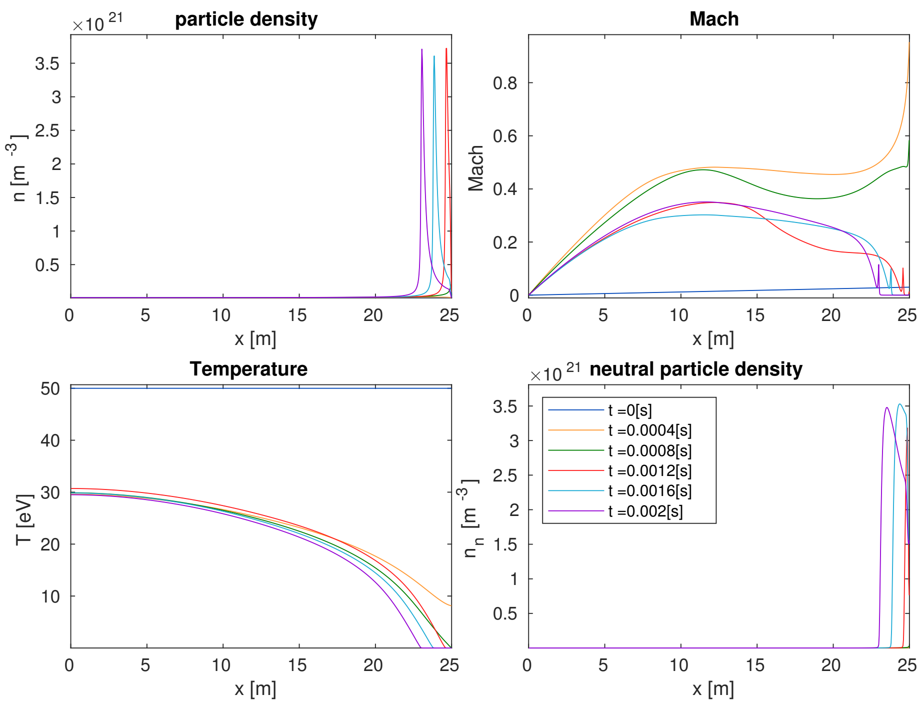
3.4.1 Method of Manufactured Solutions

To validate the numerical schemes and to find their orders of convergence the Method of Manufactured Solutions is used. The method is as follows.

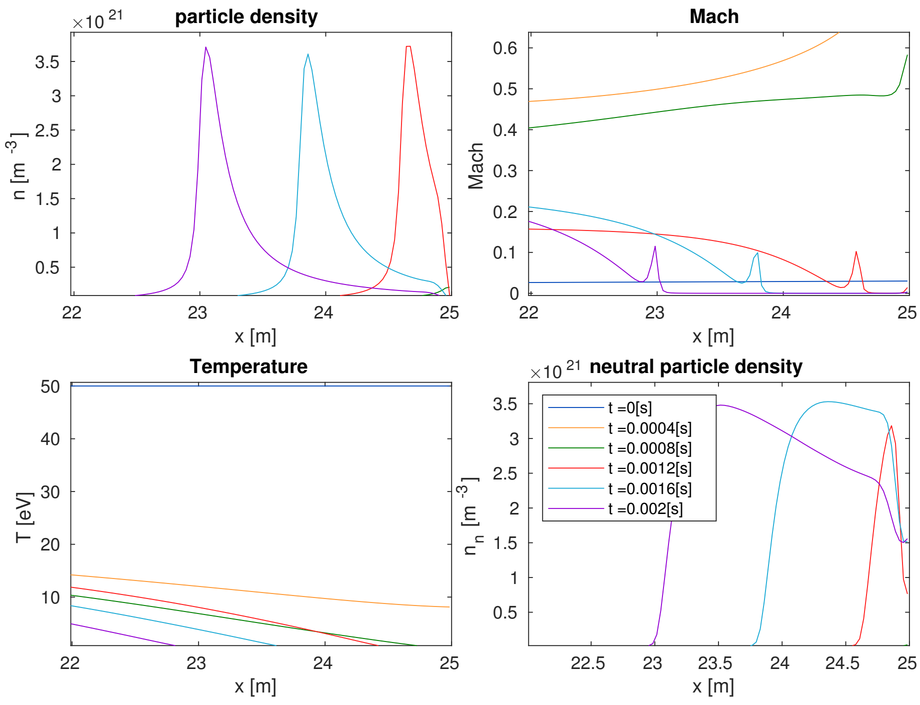
1. Consider a general homogeneous PDE $L(u) = 0$, with solution u and differential operator L.

2. Choose an analytic solution $\hat{u}$ that is sufficiently smooth and meets the boundary conditions.

3. Because $\hat{u}$ is not necessarily the analytic solution of the PDE it will result in a source term $S_{\hat{u}}$. This source term can be found by substituting it into L, so $S_{\hat{u}} = L(\hat{u})$.



(a) Solutions using the Flux-splitting scheme.



(b) Solutions using the Flux-splitting scheme, zoomed in at the target.

Figure 3.9

4. Solve the PDE $L(u) = S_{\hat{u}}$ numerically. If the numerical method is valid then it should approximate the manufactured solution.

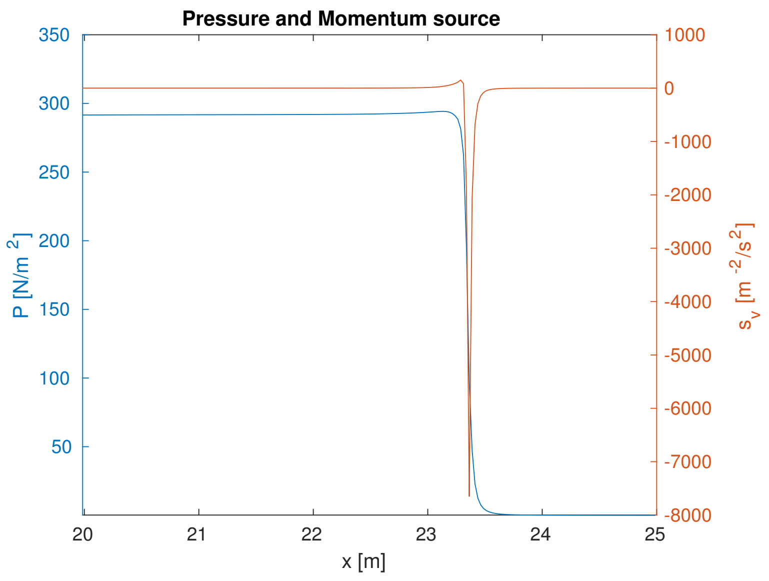


Figure 3.10: Pressure and charge exchange profiles to explain the peak in v near the detachment front.

5. To calculate the error and order of convergence, define u as the numerical solution. Then the global error ε is given by the 2-norm, $\varepsilon = \sqrt{\frac{1}{N}\sum_j (u_j - \hat{u}_j)^2}$, where $\hat{u}_j = \hat{u}(x_j)$.

6. To find the order of convergence $p_{\Delta t}$ for the time step Δt, assume that $\varepsilon_{\Delta t} = C\Delta t^p + O(\Delta t^q)$, where $q > p$. This also implies that $\varepsilon_{\Delta t/2} = C(\Delta t/2)^p + O(\Delta t^q)$. Ignoring the higher order terms and dividing $\varepsilon_{\Delta t}$ by $\varepsilon_{\Delta t/2}$ expression (3.4.1) can be derived for $p_{\Delta t}$,

$$p_{\Delta t} = \log_2\left(\frac{\varepsilon_{\Delta t}}{\varepsilon_{\Delta t/2}}\right). \tag{3.4.1}$$

The order of convergence for the spatial discretization is calculated similarly.

Implementation of the Method of Manufactured Solutions

Equations (3.4.2) - (3.4.5) are the manufactured solutions, their derivation is in Appendix C.4. These solutions are chosen because the exponential power and trigonometric functions are infinitely differentiable and it is still relatively easy to calculate their derivatives. Particle densities and temperatures (when using [eV]) must be positive. So only the velocity can become negative. Thus to implement the same restrictions as are set on the actual solutions, all the manufactured solutions are positive, except for the velocity which is both positive and negative on the domain. The diffusion coefficients for the heat and neutral particles are taken as constant in space because the original functions made calculating the manufactured solutions too hard. So $\kappa_{\parallel}^{e}(T) \equiv \kappa_0 e^{-2t}$ and $D(n,T) \equiv D$. Also, this discretization for a diffusion term has already been validated analytically in Appendix C.1. The manufactured solutions are:

$$n(x,t) = e^{-t}\left(2+\cos\left(\frac{\pi x}{L}\right)\right), \tag{3.4.2}$$

$$v(x,t) = e^{-t}\sin\left(\frac{2.5\pi x}{L}\right), \tag{3.4.3}$$

$$T(x,t) = e^{-2t}\left(\frac{3-\frac{\gamma}{2}}{-\kappa_0\frac{\pi}{2L}}\cos\left(\frac{\pi x}{2L}\right)+\frac{1}{2}\right), \tag{3.4.4}$$

$$n_n(x,t) = \frac{2L\hat{x}v\sin^2\theta}{2\pi D}e^{-2t}\left(2+\cos\left(\frac{3\pi}{2L}\right)\right). \tag{3.4.5}$$

Because there are a numerical schemes for both spatial and time derivatives, two different rates of convergence are defined. All the variables have their own rates of convergence as well. It is not possible to solve the stationary equations directly so to validate the spatial discretization consider the following example: $\frac{\partial u}{\partial t}+\frac{\partial u}{\partial x}=0$. Define manufactured solution $y(x,t)$ then the manufactured problem becomes $\frac{\partial u}{\partial t}+\frac{\partial u}{\partial x}=\frac{\partial y}{\partial t}+\frac{\partial y}{\partial x}$. Use some method A_t for the time integration and assume that $A_t(u)$ converges to $\frac{\partial u}{\partial t}$ for $\Delta t \to 0$. Use some discretization A_x for the spatial derivative, then the scheme becomes

$$A_t(u)+A_x(u)=\frac{\partial y}{\partial t}+\frac{\partial y}{\partial x}.$$

For $\Delta t \to 0$ this problem converges to $A_x(u)=\frac{\partial y}{\partial x}$. So for a small Δt the manufactured problem becomes similar to the manufactured stationary problem. Also, the Jacobian will be diagonally dominant for small Δt, making calculating the solution easier. The results are given in the next three subsubsections.

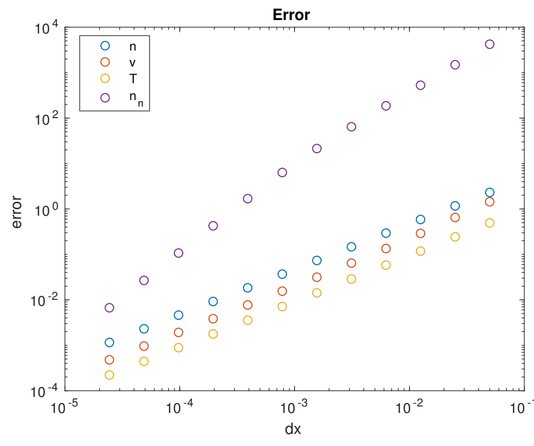
Validation of the Implicit scheme

Using the previously described method, the error and order of convergence for the Implicit scheme are calculated and shown in Figure 3.11. The errors of all variables converge to 0, which validates this spatial discretization. The time integration is done by `ode15i()` so this is validated by Matlab. The order of convergence goes to 1 for n, v and T and towards 2 for n_n. This is as expected because the upwind and downwind schemes are first order convergent and the scheme for the neutral particle diffusion (Equation (3.1.7)) is second order convergent. For lower N the order of convergence for n_n is less than 2. Our conjecture is that this is caused by the error at the right boundary which depends on n and v, which have first order convergence. For larger N the number of grid points that are influenced by the boundary conditions stays the same so they have relatively less influence, making the order of convergence go to 2.

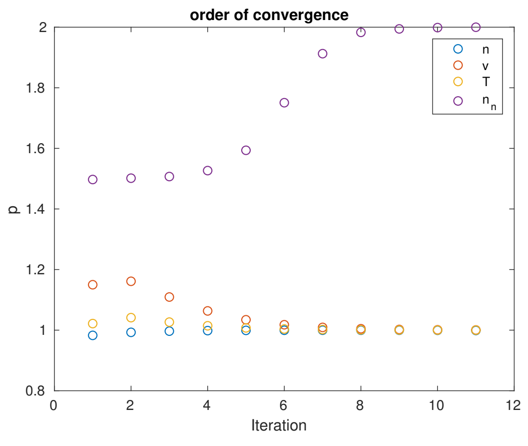
Validation of the Semi-Explicit scheme

The errors converge towards 0 for the Semi-Explicit scheme as well (Figure 3.12). The order of convergence for the plasma variables converges to 1 as is expected when using the upwind and downwind schemes. The order of convergence for n_n goes towards 2 at first, as is expected, but then it decreases again. Our conjecture is that it is caused by the machine precision. The errors are divided by the time step $\Delta t = 10^{-9}$ before they are plotted so they are actually between 10^{-11} and 10^{-13}. It could also be caused by the errors in n and v impeding the order 2 convergence of n_n.

The Method of Manufactured Solutions is also applied to the time integration to calculate



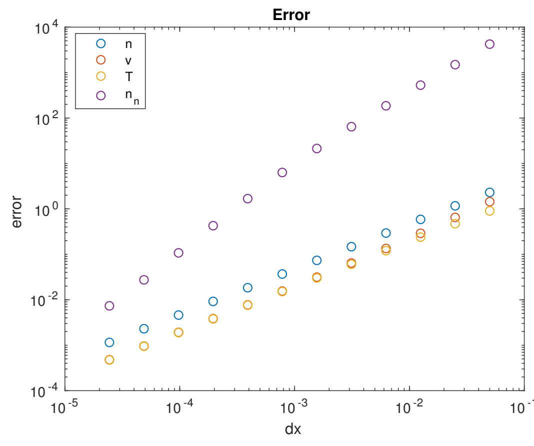
(a) Error



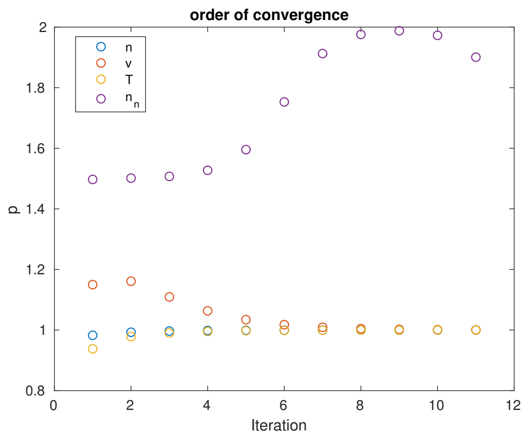
(b) Rate of Convergence

Figure 3.11: Implicit scheme spatial discretization. The time step that is used is 10^{-9} and the error depicted here is multiplied by it. One iteration consists out of calculating p using the current and previous number of grid points.

the order of convergence (Figure 3.13). As expected this converges towards 1 because of the use of Forward Euler to approximate derivatives.



(a) Error

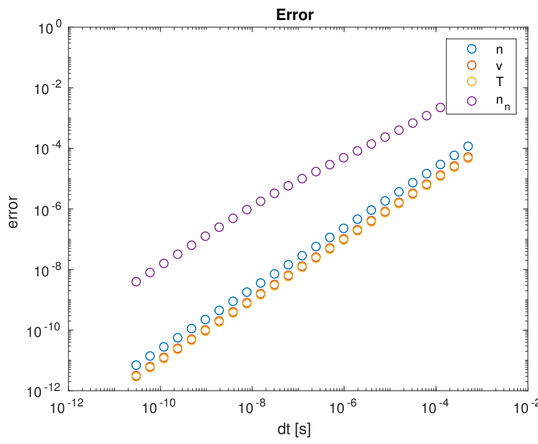


(b) Rate of Convergence

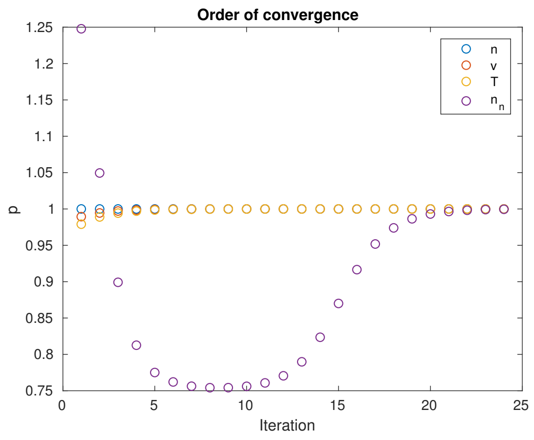
Figure 3.12: Semi-explicit scheme spatial discretization. The time step that is used is 10^{-9} and the error depicted here is multiplied by it. One iteration consists out of calculating p using the current and previous number of grid points.

Validation of the Flux-splitting scheme

The errors of all variables converge to 0, which validates this spatial discretization (Figure 3.14). The time integration is again done by `ode15i()`. The order of convergence goes to 1 for n, v and T and towards 2 for n_n. This is as expected because the upwind and downwind schemes are first order convergent and the scheme for the neutral particle diffusion is second order convergent.



(a) Error



(b) Rate of Convergence

Figure 3.13: Semi-explicit scheme time discretization. One iteration consists out of calculating p using the current and previous time step.

3.4.2 Comparison of the numerical schemes

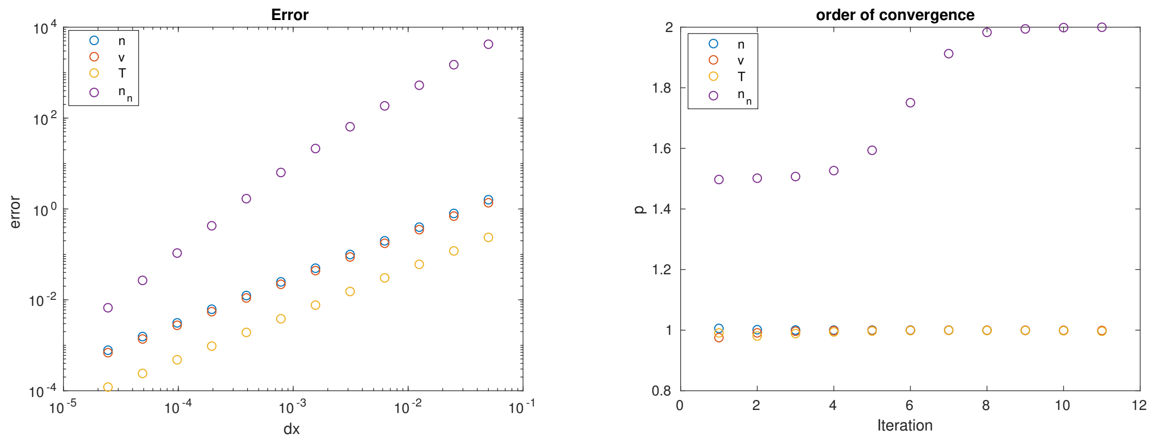
First the 3 methods will be compared to each other. In Figure 3.15 the running times that are needed to compute the solution for the time interval $(0, 10^{-9})$[s] are given. This is short enough to meet the stability conditions for most numbers of grid points and also to prohibit unstable methods from diverging. The Semi-Explicit method is the fastest in this case as is expected, because it does not have to solve a non-linear equation. But due to its stability conditions which requires taking smaller time steps this method is by far the slowest for larger time intervals. We presume that the disorderly behaviour for low values of N is caused by lower order processes in the initialisation of the simulation and/or Matlabs optimisation of its library for each method. For the Implicit and Flux-splitting schemes the differences in solution for several numbers of grid points is tested. The time and number of time steps that is needed to compute the solution at $t = 10^{-3}$ [s] is also plotted. The Semi-Explicit scheme is not considered here because its execution time is significantly longer, making a more detailed comparison unnecessary.

Numerical properties of the Implicit scheme

The results are in Figure 3.16. For $N \geq 800$ the solutions are almost identical, below this value the differences are substantial which will influence the outcome of the simulation. Hence $N = 800$ will be used for simulations. The run time and the required number of iterations are approximately linear to N.

Numerical properties of the Flux-splitting scheme

The results are in Figure 3.17. Higher values for N are used because the run time and number of function evaluations proved to be significantly lower than those for the Implicit scheme, hence we wanted to study it for larger N as well. For $N \geq 1000$ the solutions are almost identical, below this value the differences are substantial, which will influence the outcome of the simulation. Hence $N = 1000$ will be used for most simulations. However when detachment occurs, $N > 2000$ is preferred because the detachment front has larger gradients requiring higher precision. The run time and the required number of iterations are approximately linear to N and significantly



(a) Error (b) Rate of Convergence

Figure 3.14: Flux-splitting scheme spatial discretization. The time step that is used is 10^{-9} and the error is multiplied by it before it is plotted. One iteration consists out of calculating p using the current and previous number of grid points.

lower than those of the Implicit scheme for large N. We can not say why the number of function evaluations is not strictly increasing with N because the algorithm of `ode15i()` is unknown to us.

Comparison of the Implicit scheme and the Flux-splitting scheme

To demonstrate the differences in solution for both methods two example cases are considered, a stationary solution and a detached plasma. First we compare a solution where the plasma does not detach. At time $t = 0.005$ [s] the solution has reached its steady state. The solutions are in Figure 3.18. The Implicit and Flux-splitting schemes are practically similar. The steady state solution is not calculated with the Semi-Explicit scheme because this scheme is too slow. Not only does it take long to simulate 0.005 [s], but its long execution time also makes is unsuitable for control system design.

To simulate a detached plasma the steady state solution is used as initial value, then a gas puff is introduced into the model, $c_{\text{puff}} = 60$. Unscaled this would be a continuous gas puff of $c_{\text{puff}} = 6 \cdot 10^{21}$ [m^{-3}s^{-1}]. The solutions are in Figure 3.19. The detachment front in the Flux-splitting scheme travels much faster. So to make the profiles more similar to each other the simulation for the Flux-splitting scheme is for a shorter period of time than the Implicit scheme. The velocity in the Flux-splitting scheme also becomes negative as a result of the rapid decrease in v when detachment occurs. The velocity in the Implicit scheme however, does not become negative. For the Implicit scheme only 400 grid points are used. For more grid points the simulation crashes because the velocity becomes too small, as is explained in Appendix C.7. For $N = 400$ the Implicit scheme has not converged yet so we consider this solution to be invalid. We presume that this is also the reason why the velocities of the detachment fronts in both solutions are different. The cause of this has already been discussed in Section 3.1.4. Therefore we consider the Implicit scheme not robust enough to model a detached plasma, making the Flux-splitting scheme the only feasible option discussed in this report.

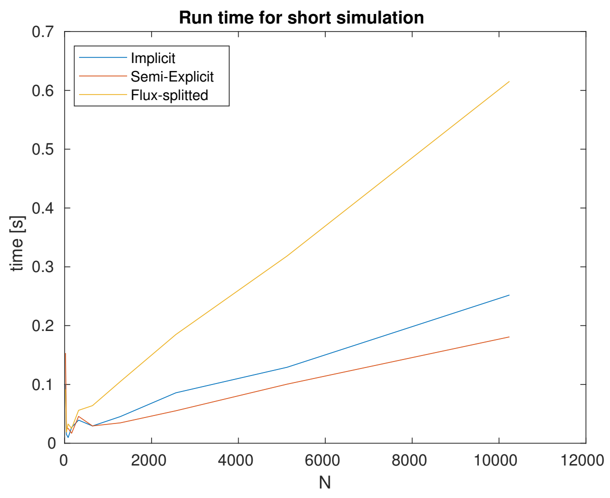


Figure 3.15: Comparison between run-time for the time interval $(0, 10^{-9})$[s] for different values of N.

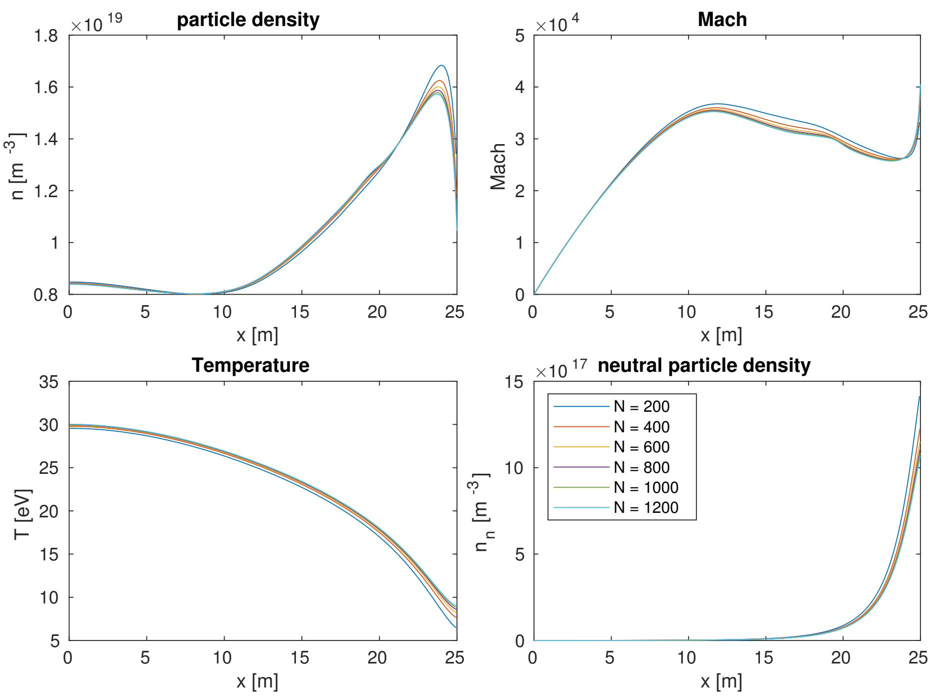
Comparison to the benchmark model, SD1D

To compare the Flux-splitting scheme to the SD1D model again a stationary solution and a detached plasma are used. These are in Figure 3.20. Both are very different from the solutions calculated with the Flux-splitting scheme, even though the parameters and functions are chosen to make the solutions as similar as possible. Using the same parameters for both models results in completely different solutions.

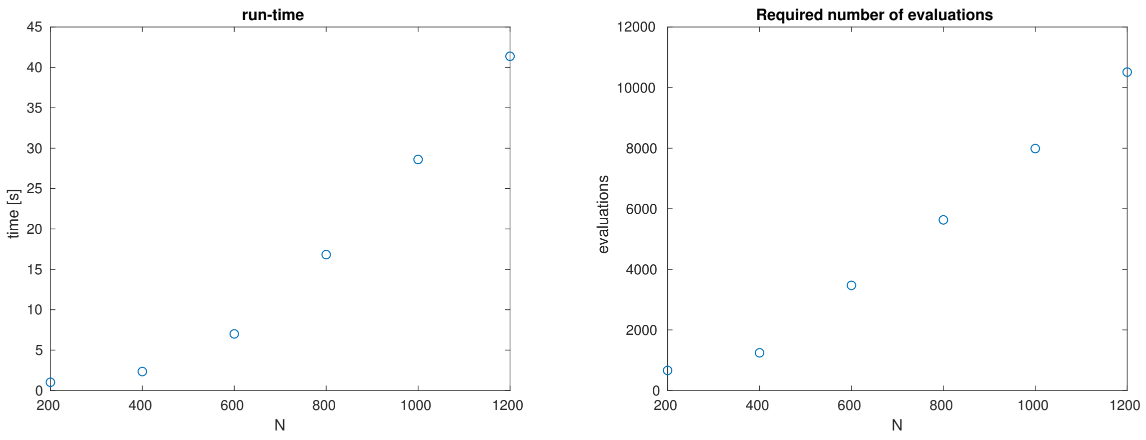
It appears the neutral diffusion coefficient of the SD1D model is much smaller, resulting in a much higher neutral particle density at the target. For the stationary solution the peak in n is also closer to the target. Away from the target the velocity is lower. The profile for the temperature is similar to that of the Flux-splitting scheme. We have not been able to find or implement the right parameters and rate coefficients to make both simulations similar. The solution of the temperature however is qualitatively similar to the solution of the Flux-splitting scheme.

For the detached solutions the most notable difference is the particle density between the detachment front and the target. This could be caused by less recombination in the SD1D solution because the temperature does not drop below 2.5 [eV]. We have not been able to explain why the energy does not decrease below 2.5 [eV].

The SD1D method uses the CVODE/PVODE solver created by Sundials for the time integration, these use either Backward Differentiation Formula (BDF) or Adam-Moulton [22]. Adam-Moulton is not an implicit method, BDF is and is also the method that is used by SD1D. The spatial integration is done by a flux-vector splitting scheme as well. We tested the required number of time steps for several inputs, these are given in table 3.1. The number of iterations that is required increases approximately linearly by N. It is much bigger than those needed for the Flux-splitting scheme. Our conjecture is that this is caused partly by the complexity of the SD1D model but mainly by the lack of an analytic Jacobian in the SD1D model. The run time is not a reliable measure of its performance since it can be influenced by programs running in the background. It is only given to give a rough estimate of the influence of N on the run time.



(a) Solutions at $t = 0.001$ [s] for several values of N.

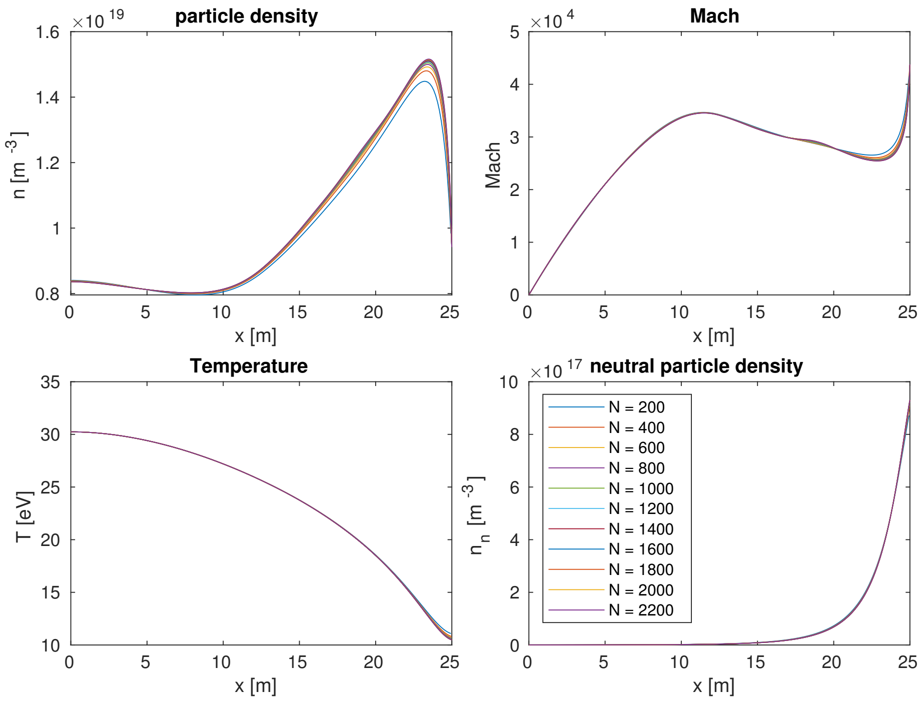


(b) Run time to calculate the solution at $t = 10^{-3}$[s].

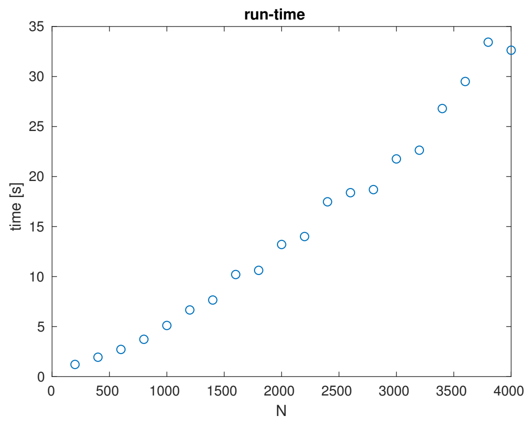
(c) Number of function evaluations required on the interval $(0, 10^{-3})$[s].

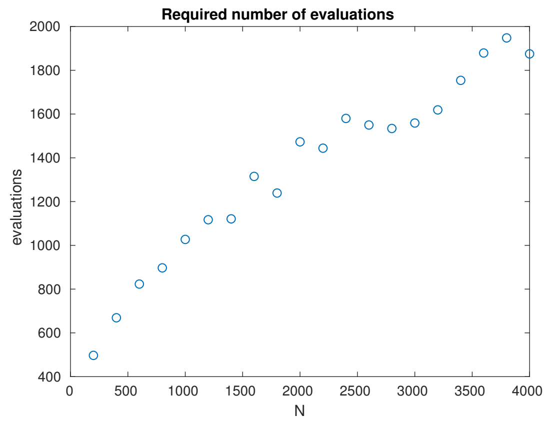
Figure 3.16: Run time and used number of time steps to calculate the solution at $t = 10^{-3}$[s] for the Implicit scheme.

In conclusion, the solutions of the Flux-splitting scheme are qualitatively similar to those of the SD1D model in the sense that both can simulate detached plasmas. The quantitative differences however are large. For similar parameters the outcomes are very different both in value and behaviour. We do not understand the source code well enough to explain these differences. The adjustments we made to it are in Appendix B.3. The Flux-splitting method is much faster than the SD1D method, making it better suited for feedback control or when solutions are needed quickly.



(a) Solutions at $t = 0.001$ [s] for several values of N.





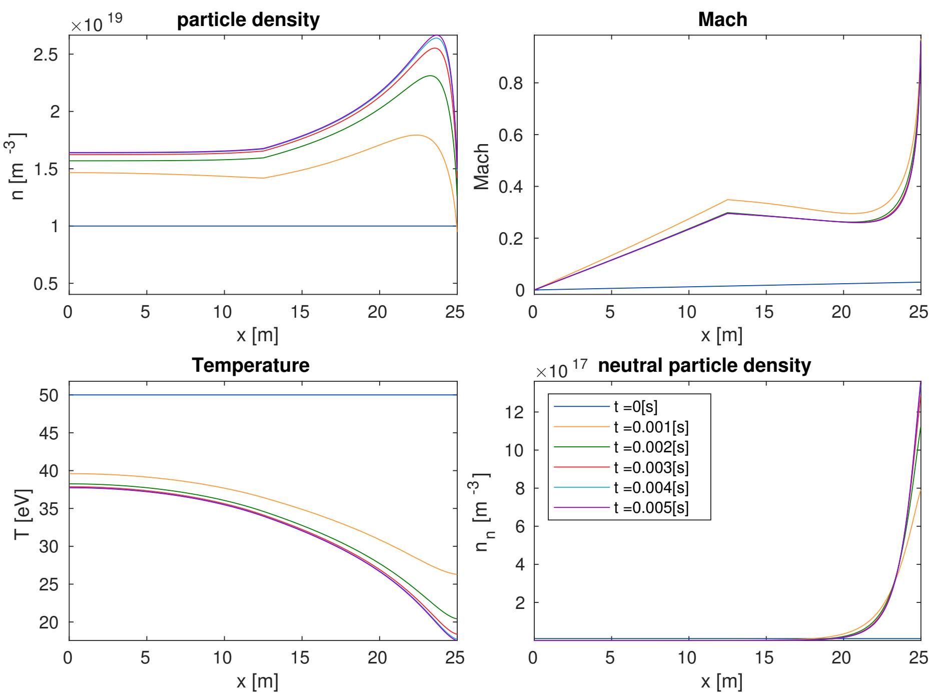
(b) Run time to calculate the solution at $t = 10^{-3}$[s].

(c) Number of function evaluations required on the interval $(0, 10^{-3})$[s].

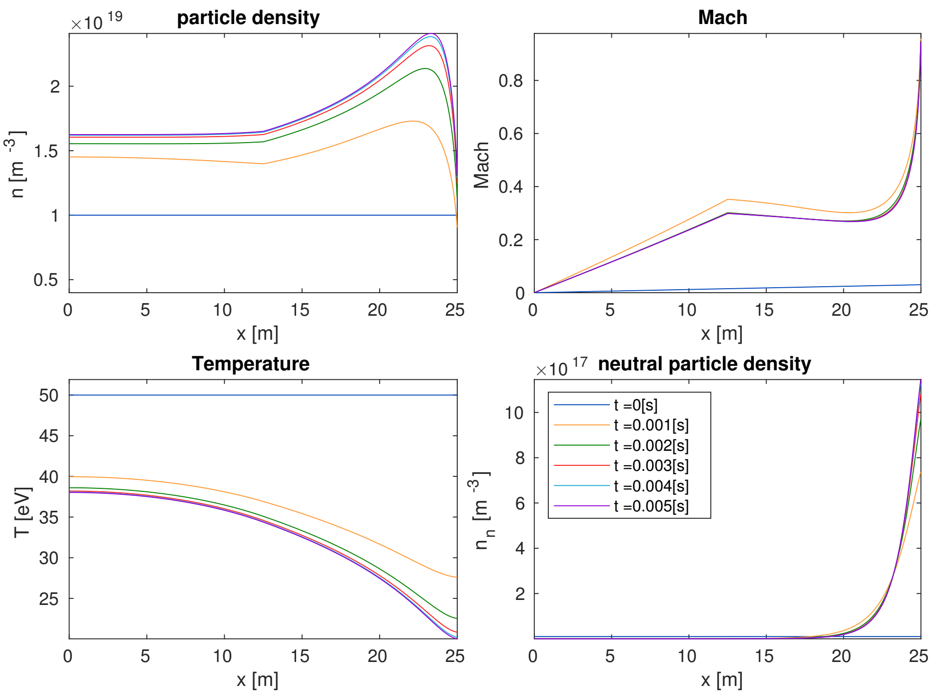
Figure 3.17: Run time and used number of time steps to calculate the solution at $t = 10^{-3}$[s].

3.5 Stationary plasma equations

In an upstream density ramp experiment in [23] it is observed that the movement of the detachment front is approximately 0.4 [m] in poloidial distance in 1.5 seconds. For the exact velocity we need to know the geometry of the magnetic field lines and have more precise data. But we know it will be less than 100 meter, so we can state that the velocity of the front is a lot slower than the ion thermal velocity, which is a typical velocity for the plasma particles [14]. Therefore, it is assumed that the plasma dynamics have a much smaller timescale than the neutral dynamics. If this is the case then the plasma variables can be approximated by their steady state solution instead of solving a time dependent PDE for them. Our conjecture is that this would allow the



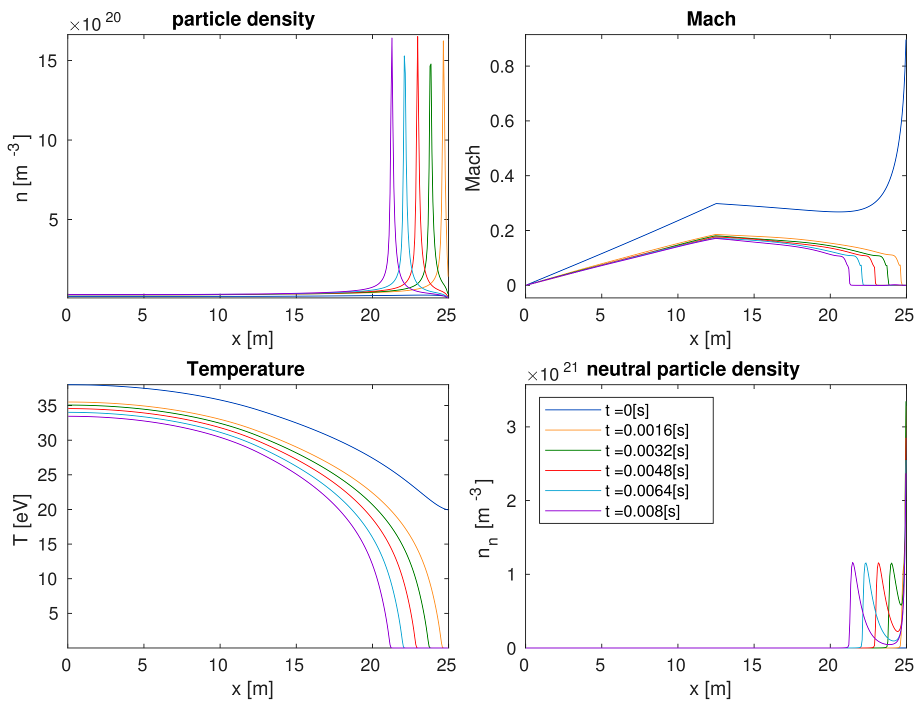
(a) Implicit scheme.



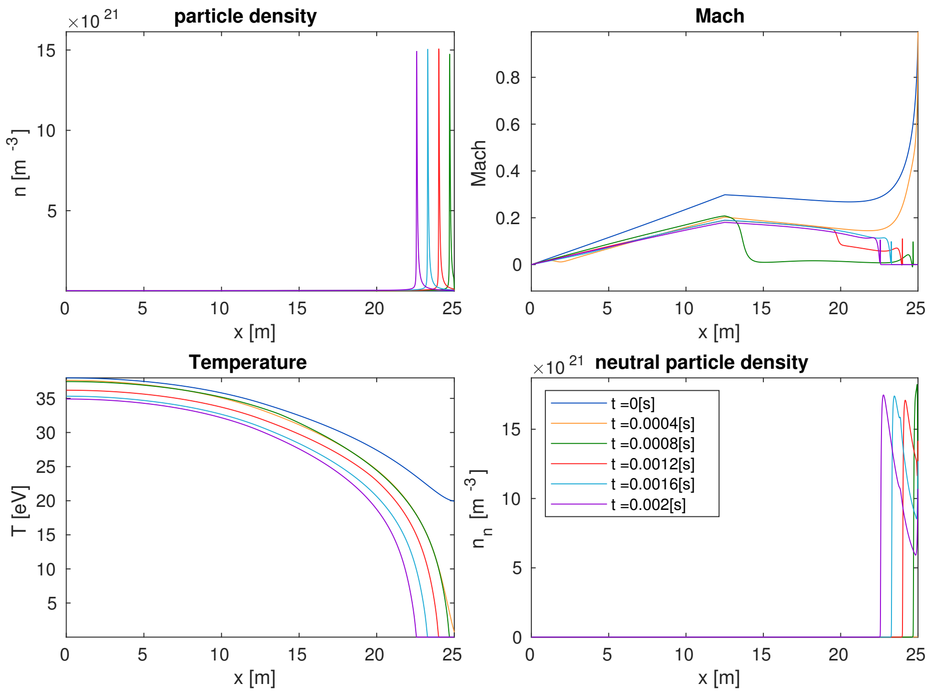
(b) Flux-splitting scheme.

Figure 3.18: Steady state solution comparisons.

model to take time steps based on the neutral dynamics, these steps would be bigger so the computation speed would be increased.



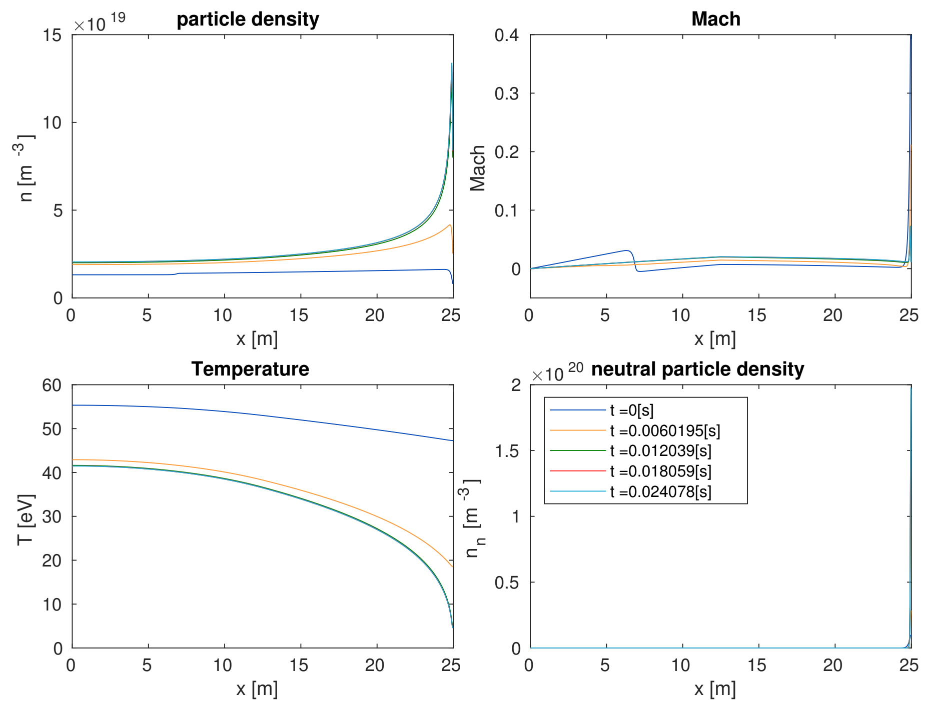
(a) Implicit scheme, number of grid points $N = 400$.



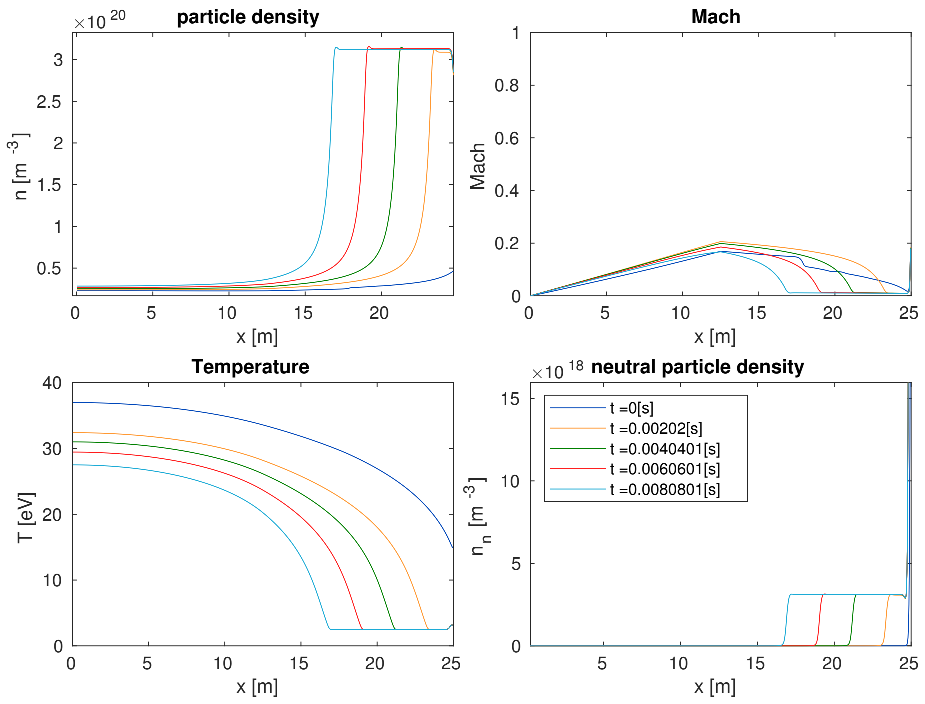
(b) Flux-splitting scheme, number of grid points $N = 4000$.

Figure 3.19: Comparison of detached solution.

We attempted to solve the system of differential algebraic equations (DAEs) with two Matlab solvers that are programmed to solve systems of DAEs. Doing this with `ode15i()` is not possible. This only results in an error notification that the time step has to be reduced below



(a) Stationary solution.



(b) Detached solution.

Figure 3.20: Solutions calculated with the SD1D model.

machine precision to meet the toleration standards. Using `ode15s()` to solve the system of DAEs makes calculating a stationary solution remarkably fast. However the solution for the plasma variables are different from those of the original PDE, so the dynamics of the system are wrong.

N	Time iterations	run time
800	44368	11 m and 52 s
600	33375	6 m 39 s
400	22683	3 m 2 s
200	12124	49 s

Table 3.1: Time steps and running times for several inputs in the SD1D model for calculating the solution at $t = 10^{-3}$ [s].

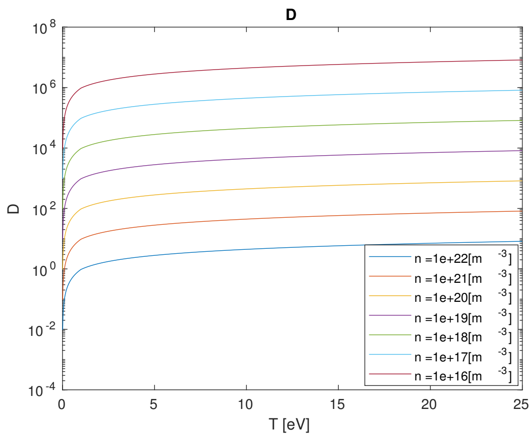
When calculating a detached plasma `ode15s()` gives the same error as `ode15i()`. In Figure 3.22 an example is shown of a simulation using `ode15s()` and the stationary plasma equations to calculate a stationary solution. The steady state is already reached after approximately $3 \cdot 10^{-5}$ [s], whereas the regular Flux-splitting scheme needs approximately 0.005 [s]. The computation time and number of function evaluations is also significantly lower. But because the dynamics have changed we deem it unfit to be used to model divertor plasma.

Using definition (2.2.24) for the diffusion coefficient also does not result in a large difference in time scales between the plasma and neutral dynamics. In Figure 3.21a the scaled diffusion coefficient is shown as a function of the temperature for ion particle densities that occur in divertor plasmas. For the scaled equations most coefficients for the terms in the plasma equations are 1, hence only for small temperatures and large ion particle densities there is a large relative difference. This is the case close to the detachment front. However right of the detachment front the particle densities become much lower, increasing the diffusion coefficient and making the neutral dynamics faster than the plasma dynamics. In Figure 3.21b the scaled neutral diffusion coefficient is plotted as well, but Function (2.2.23) is used. The neutral particle density is not negligible anymore once the temperatures become low enough for recombination. So the reasoning in Section 2.2.5 that (2.2.24) can be used instead of (2.2.23) is not valid anymore once the plasma detaches.

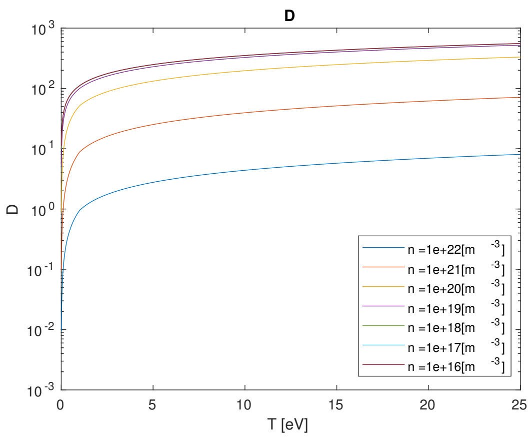
Computing the stationary solutions of the plasma equations requires solving a non-linear equation for each time step. When using Matlab this is usually done using a method similar to or based on Newton-Raphson iteration. To converge to the solution these methods require an initial guess that is close enough to the solution. The initial guess that is used is the previous solution, hence increasing the time step will increase the distance between the initial guess and the solution. This makes the likeliness of convergence to a solution smaller. Moreover the analytic Jacobian of the nonlinear equation will become more diagonally dominant for smaller Δt, making convergence with Newton-Raphson faster. Our conjecture is that these two factors and the fact that the difference in time scales is small, cause the errors when trying to solve the system of DAEs with Matlab.

3.6 Overview of all numerical schemes

In this section all the numerical methods that have been discussed in this report are summarised in Tables 3.2-3.6. The tables also contain some relevant parameters for the schemes that were used to compare the methods/schemes.



(a) Diffusion coefficient using Function (2.2.24).



(b) Diffusion coefficient using Function (2.2.23) with $n_n = 10^{22}[\mathrm{m}^{-3}]$.

Figure 3.21: Scaled diffusion coefficient as a function of the temperature. For several ion particle densities.

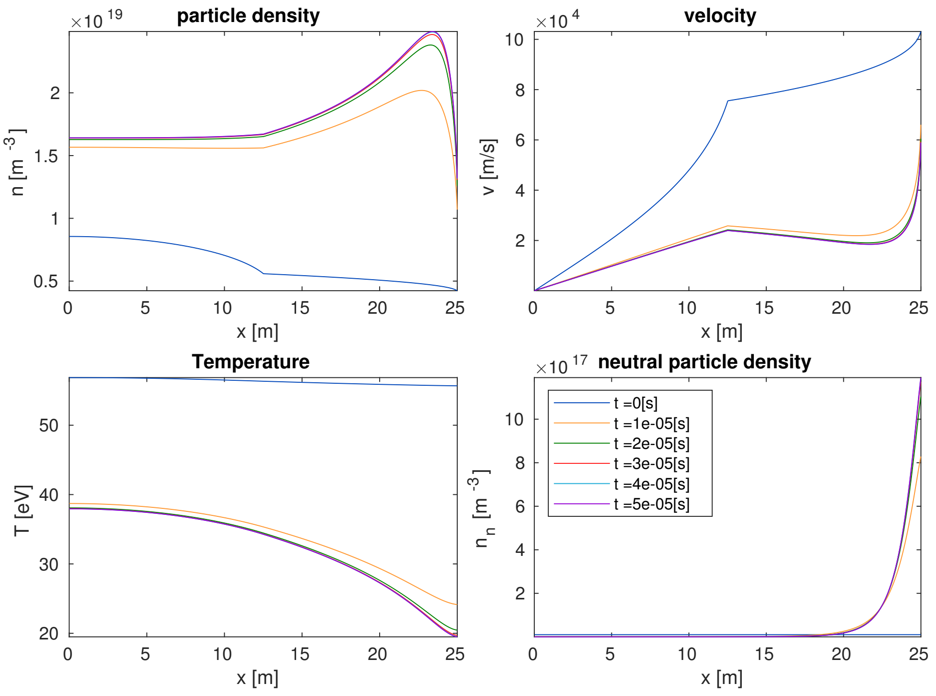


Figure 3.22: Calculation of a stationary solution using the stationary plasma equations.

Implicit scheme

Method	Stability	Order of Convergence	computation time
Implicit Euler with variable step size	Stable	1 [24, Chapter 6]	49.7 [s]
`ode45()`	Unstable	-	-
`ode15s()`	Unstable	-	-
`ode15i()`	Stable	1-5 [19]	45.5 [s]

Table 3.2: Time integration methods for the Implicit scheme. The computation time is only for one simulation and serves merely as an example. For this case $N = 1000$ and the simulated time is 10^{-5} [s].

The number of function evaluations for the self implemented Implicit Euler method is dependent on the default time step, which has to be chosen by the user. This also influences the computation time, so the right choice for Δt has to be made for short computation times. This makes it less convenient to use than `ode15i()`.

Terms	Method	Numerical performance
Diffusion terms	Equation (3.1.7)	Second order convergent
First order derivatives	Downwind for pressure upwind for other terms	First order convergent
	All upwind	Numerical oscillations
	Central difference for all [1]	Numerical oscillations

Table 3.3: Spatial discretization for the Implicit scheme.

Semi-Explicit scheme

For the Semi-Explicit scheme relatively many methods have been implemented. Partly because the four PDEs were solved separately each time step and partly because many discretizations were implemented for the spatial derivatives.

Equation	Method	Stability	Implementation	Order of convergence
Particle	`ode15s()`	Stable	Requires Jacobian	1-5 [25]
	`ode45()`	Stable	No Jacobian	4 [26]
Momentum	`ode15s()`	Stable	Requires Jacobian	1-5 [25]
	`ode45()`	Stable	No Jacobian	4 [26]
Energy	`ode15s()`	Unstable	Requires Jacobian	[25]
	`ode15i()`	Stable	Requires Jacobian	1-5 [19]
Neutral	`ode15s()`	Stable	Requires Jacobian	[25]
	`ode15i()`	Stable	Requires Jacobian	1-5 [19]

Table 3.4: Time integration methods for the Semi-Explicit scheme.

Term	Method	Result
Diffusion terms	Equation (3.2.1)	2nd order convergent
Convection of Energy	Central difference	2nd order convergent [2]
Remaining terms	Central difference for all	Numerical oscillations
	Mix of central difference upwind and downwind	Numerical oscillations
	Downwind for pressure biased upwind for others	Numerical oscillations
	Downwind for pressure upwind for others	First order convergent

Table 3.5: Spatial discretization for the Semi-Explicit scheme.

[1]Only considered in a toy problem.
[2]Numerical errors might damped by the dominant diffusion term.

Flux-splitting scheme

For the Flux-splitting scheme only first order upwind and downwind has been used to discretize the spatial derivatives. The methods that were implemented for the time integration are in Table 3.6. It is again demonstrated that the DAE approach only serves for quickly finding a stationary solution.

Method	Number of function evaluations	Time	Detached plasma
`ode15i()`	1933	0.014 [s]	yes
DAE `ode15i()`	No solution	-	no
DAE `ode15s()`	200	$6.5 \cdot 10^{-5}$ [s]	no

Table 3.6: Time integration methods for the Flux-splitting scheme. The number of function evaluations and the time till a stationary solution is reached are given as well (This is real time, not computation time). These were only computed for one specific set of parameters and only serve as an example. The last column contains whether or not a method could simulate a detached plasma.

4 Application

To demonstrate the application of the Flux-splitting scheme, its application in control will be explained and its solution to two important phenomena for divertor plasmas will be simulated.

4.1 Implementation in system control

To use system control on this system of PDEs it is required to linearize it. This is done with the same method as in [12, Section 1.5]. Let w be the internal states of the system, which are the minimum set of variables required to fully characterise the state of the system at time t. u is the controllable input and z the measurable output. Then the system of PDEs must be transformed to the form,

$$\begin{aligned}\dot{w} &= Aw + Bu,\\ z &= Cx + Du.\end{aligned}$$

The solution variables can be scalars or vectors and A, B, C and D are scalars or matrices.

The system of PDEs in Equations (2.2.5) to (2.2.8) has to be made into a system of ODEs. This is done by discretizing the domain and approximating all the spatial derivatives numerically, which is done by the numerical scheme. This will result in a set of ODEs of the form $\dot{\underline{w}} = H(\underline{w})$. Where $\underline{w} \in \mathbb{R}^{4N}$ is the $4N$ dimensional vector, $\underline{w} = (\underline{n}; \underline{v}; \underline{T}; \underline{n}_n)$ and $H : \mathbb{R}^{4N} \to \mathbb{R}^{4N}$ is a nonlinear function.

The only thing that can be controlled in this model is the gas valve which is located at the target. So the control variable is $u = c_{\text{puff}}$ which is in the boundary condition table 2.1. By discretizing the set of spatial derivatives the boundary condition of the gas puff becomes $\hat{x}\hat{v}c_{\text{puff}}\underline{e}_{4N}$. This also defines B as a zero vector except for the $4N$th entry which is equal to $\hat{x}\hat{v}$.

The output is the peak of the impurity radiation front which is used to define the detachment front as is done in [23] and [27]. (In these papers it is referred to as the CIII emissivity front. In [23, Figure 4] an example is shown.) The radiation front is experimentally determined via radiation (bolometry) measurements, spectroscopy, or spectral imaging. So $z = \arg\max_{x\in(x_{\text{pt}},L)} \xi n^2 L(T)$. Using a discrete domain this becomes $z = x_j$ where

$$j = \underset{i\in\{\lceil N/2\rceil, N\}}{\arg\max}\ \xi n_i^2 L(T_i).$$

This function can not be linearised with the same method. It is not a continuous function so the derivative is not defined.

The equation for the time derivative of $\underline{w}$ is not linear, so we linearize this equation around $\underline{w}^*$ and $\underline{u}^*$. The nonlinear function H is approximated by a first order Taylor expansion:

$$H(\underline{w}) = H(\underline{w}^*) + \mathrm{Jac}_H(\underline{w}^*)(\underline{w} - \underline{w}^*) + O\left((\underline{w} - \underline{w}^*)^2\right).$$

Define the deviation variables $\delta\underline{w} = \underline{w} - \underline{w}^*$ and $\delta u = u - u^*$. Then the expression for $\delta\dot{\underline{w}}$ becomes,

$$\delta\dot{\underline{w}} + \dot{\underline{w}}^* \approx H(\underline{w}^*) + \mathrm{Jac}_H(\underline{w}^*)\delta\underline{w} + B(\delta u + u^*).$$

Under the assumption that $\dot{\underline{w}}^* = H(\underline{w}^*) + Bu^*$ the equation becomes,

$$\delta\dot{\underline{w}} \approx \mathrm{Jac}_H(\underline{w}^*)\delta\underline{w} + B\delta u.$$

The assumption can be made if $\underline{w}$ and u^* are the solutions from the previous time step. The resulting equation is linear. The Jacobian matrix Jac_H is the same Jacobian as is used to speed up the convergence in both implicit time integration models. So these are already calculated. If the control variable and the output are also taken into consideration then the complete set of equations becomes:

$$\begin{aligned} \delta\dot{\underline{w}} &= \mathrm{Jac}_H(\underline{w}^*)\delta\underline{w} + B\delta u, \\ z &= x_j \text{ such that } j = \underset{i\in\{\lceil N/2\rceil, N\}}{\arg\max}\ \xi n_i^2 L(T_i). \end{aligned}$$

4.2 Simulations with the Flux-splitting scheme

4.2.1 Simulation of a roll-over

The start of the detachment of the plasma is the roll-over as is described in Section 2.1.3. Before detachment the target particle density n_t is proportional to the square of the upstream particle density n_u. If the solution becomes detached this relation will not hold any more. After detachment n_t will be much lower than the expected $n_\mathrm{t} \propto n_\mathrm{u}^2$. To simulate a roll-over the solution is computed for several upstream particle sources. The upstream particle source terms are

$$S_n \in \left\{ s \cdot \frac{10^{23}}{L - x_\mathrm{point}} \middle| s = 1, 1.5, ..., 10 \right\}.$$

The simulated time is taken long enough for the solution to either reach its steady state or for the plasma to become detached. If the plasma detaches then the simulation does not last long enough to reach a steady state. The results are in Figure 4.1. It can be seen that at first n_t is approximately proportional to n_u^2. Between $n_\mathrm{u} = 2 \times 10^{19}[\mathrm{m}^{-3}]$ and $n_\mathrm{u} = 2.9 \times 10^{19}[\mathrm{m}^{-3}]$ the roll-over occurs and n_t drops. There are no values in this interval because we simulated it by changing S_n instead of controlling n_u. The D_α radiation is not computed because it is caused by the electrons of a hydrogen atom changing their radial quantum number, which is not implemented in our model.

4.2.2 Simulation of a detached plasma

As a demonstration of using gas puffing to detach the plasma we simulated a detached plasma and measured the location of the detachment front and the heat flux towards the target. Given

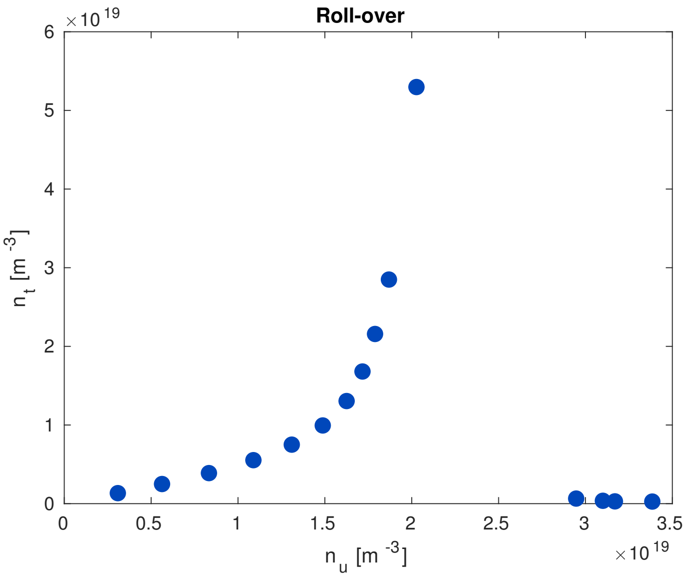
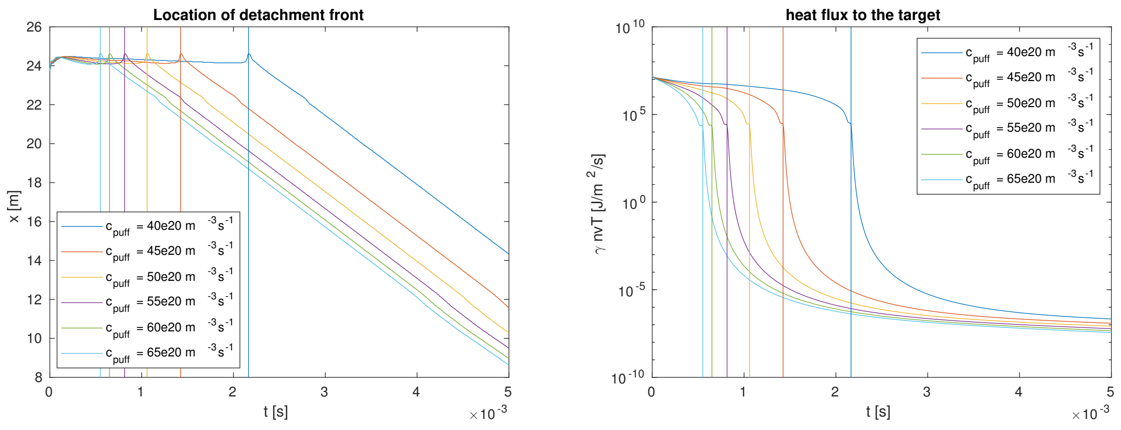


Figure 4.1: Example of a roll-over.

that the solution is detached, the detachment front is defined as the location of the peak in the impurity radiation, else it is not defined. The heat flux towards the target is γnvT because of the heat transmission boundary condition in table 2.1. The initial value in the simulation is the same stationary solution as is depicted in Figure 3.18. For the simulation the Flux-splitting scheme is used. To detach the plasma, gas is puffed into the system at the target. The results are plotted in Figure 4.2. When the plasma detaches the impurity radiation peak moves towards the x-point. To indicate the moment in time when the front detaches the time when the n_t is maximal is used. After detachment the heat flux to the target becomes negligibly small which demonstrates the necessity of operating in a detached regime for tokamaks. The plasma detaches faster for a larger gas puff, the front however moves with the same velocity regardless of the different gas puffs.



(a) Location of the detachment front.

(b) Heat flux to the target.

Figure 4.2: The vertical lines are when the plasma detaches, before that the location of the detachment front is not defined.

5 Conclusion and Outlook

Conclusion

To construct a numerical model for divertor detachment the system of PDEs of [4] is studied and a formal requirement has been derived for the number of boundary conditions for each boundary. Then three numerical methods are constructed and analysed to find one that can be used for linear system control. The methods have been compared to each other and to a benchmark model. Due to the stability conditions the methods with implicit time integration proved to be much faster. Therefore the Semi-explicit scheme is not suitable for our purpose: control. The Implicit scheme could not compute a solution for situations where shock-waves made the velocity negative, or negative velocities at the left boundary. Supersonic flow was not implemented either, even though supersonic velocities do occur for some parameters. So the Implicit scheme is not robust enough to model detachment. The Flux-splitting scheme is though, making it the only method in this report that can be used to model detachment. Due to the implementation of the Analytic Jacobian the complexity is also linear in the number of grid points. Its computation time for computing a solution is also less than that of the benchmark model.

Due to the complexity of the plasma equations and the small difference in time scales, using the stationary plasma solutions and dynamic neutral equations results in different solutions. However this semi-dynamic approach is faster in calculating the stationary solution for attached plasmas.

Outlook

The improvements that are needed are both in its performance and in its physical correctness. Many physical phenomena are missing. Important additions to the model would be:

- Remove the assumption that the field lines are straight lines. The model should contain the option for the domain to be parallel along a general magnetic field line because it would make the model more physically accurate. Flux expansion should be included as well because when the magnetic field lines are spread over a wider area, it will directly influence the ion particle density.

- The source terms that are implemented are possibly not physically correct enough. Heuristic functions are used that are a good fit for $n \approx 10^{20}[\text{m}^{-3}]$, but when detachment occurs large differences in n appear. As a result the temperatures between the target and the detachment front are around 10^{-3} [eV], which is approximately 12 [K]. This is close to absolute zero and is much too low. Better rate-coefficients, including their analytic Jacobian, should be implemented instead of heuristic functions.

- The function used for the neutral diffusion coefficient ((2.2.24)) neglects the influence of the neutral dynamics on the diffusion coefficient. For low values of n_n these influences are not significant, but when simulating divertor detachment these influences can not be neglected any more. To have a more physically correct model for the neutral dynamics another function for the diffusion coefficient is required. Using (2.2.23) is already an improvement. Baffling should also be included in the function for the neutral dynamics, this will increase the neutral particle density in the divertor [27], which will influence the velocity of the detachment front.

To improve performance both the number of grid points and the number of required integration steps can be reduced. Possible adaptions to the model to do so are:

- Most of the upstream grid points are not needed. The gradients before the x-point are usually low and the most important behaviour is at the target and/or the detachment front. Using non-uniform grid sizes will reduce the number of grid points and therefore reduce the number of computations, which results in less computation time.
- When the detachment front passes the x-point neutral particles will mix with the main plasma. This will cool down the main plasma and stop nuclear fusion. The numerical model is needed to control the detachment front, once it reaches the x-point it is already to late to intervene. This makes all of the grid points before the x-point unnecessary. Replacing the upstream plasma with a boundary condition will remove a large portion of the domain for which computations have to be done.
- An easily implemented improvement is a higher order upwind/downwind scheme. If a second order scheme is used then all the spatial discretization schemes will be second order, making the entire scheme second order.

A List of symbols

Symbol	Name	Value [unit]	Notes
n	Ion/electron particle density	$[\mathrm{m}^{-3}]$ if not scaled	variable
v	Ion/electron velocity	[m/s] if not scaled	variable
T	Ion/electron temperature	[eV] if not scaled	variable
n_n	Neutral particle density	$[\mathrm{m}^{-3}]$ if not scaled	variable
m	Mass of a proton	$1.67262 \cdot 10^{-27}$ [kg]	constant
q_e	Elementary charge	$1.60218 \cdot 10^{-19}$ [C]	constant
$\hat{n}$	Scale for the particle densities	10^{20} $[\mathrm{m}^{-3}]$	constant
$\hat{v}$	Scale for the velocity	$\sqrt{\frac{q_e \hat{T}}{m}}$ [m/s]	constant
$\hat{T}$	Scale for the temperature	100 [eV]	constant
$\hat{x}$	Length scale	L [m]	constant
$\hat{t}$	Time scale	$\hat{x}/\hat{v}$ [s]	constant
L	Length of domain	[m]	parameter
x_{pt}	Distance from x-point to mid-plane	[m]	parameter
N	Number of grid points	-	numerical parameter
Δx	Distance between grid points	$L/\hat{x}/N$	numerical parameter
Δt	Numerical time step	-	numerical parameter
γ	Sheath energy transmission factor	-	constant
c_s	ion thermal velocity	$\sqrt{\frac{2 q_e T}{m}}$ [m/s]	variable
$\langle \sigma v \rangle_{rec}$	Recombination rate coefficient	$[\mathrm{m}^3/\mathrm{s}]$	function
$\langle \sigma v \rangle_{ion}$	Ionisation rate coefficient	$[\mathrm{m}^3/\mathrm{s}]$	function
$\langle \sigma v \rangle_{cx}$	Charge exchange rate coefficient	$[\mathrm{m}^3/\mathrm{s}]$	function
W_{imp}	Impurity radiation	$[\mathrm{Jm}^3/\mathrm{s}]$	function
M	velocity w.r.t. to ion thermal velocity	v/c_s	variable
$\underline{y}$	dimensionless plasma variables	$(\underline{n}; \underline{v}; \underline{T})$	variable
$\underline{n}$	dimensionless particle density for each grid point	$(n(x_i); ...; n(x_N))$	variable
$\underline{v}$	dimensionless velocity for each grid point	$(v(x_i); ...; v(x_N))$	variable
$\underline{T}$	dimensionless temperature for each grid point	$(T(x_i); ...; T(x_N))$	variable
$\underline{n}_n$	dimensionless neutral particle density for each grid point	$(n_n(x_i); ...; n_n(x_N))$	variable

B Modeling detachment

B.1 Scaling Nakazawa's system of PDEs

To scale equations (2.2.1), (2.2.2) and (2.2.3) define the following scaling:

$$\begin{aligned} t &\equiv t\hat{t} \qquad & [\hat{t}] &= \mathrm{s}, \\ x &\equiv x\hat{x} \qquad & [\hat{x}] &= \mathrm{m}, \\ n &\equiv n\hat{n} \qquad & [\hat{n}] &= \mathrm{m}^{-3}, \\ v &\equiv v\hat{v} \qquad & [\hat{v}] &= \mathrm{m/s}, \\ T &\equiv T\hat{T} \qquad & [\hat{T}] &= \mathrm{eV}, \end{aligned}$$

where t, x, n, v and T on the r.h.s. are dimensionless.
Substitute these to get:

$$\frac{\hat{n}}{\hat{t}}\frac{\partial n}{\partial t} + \frac{\hat{n}\hat{v}}{\hat{x}}\frac{\partial (nv)}{\partial x} = s_n, \tag{B.1.1}$$

$$\frac{m\hat{n}\hat{v}}{\hat{t}}\frac{\partial (nv)}{\partial t} + \frac{m\hat{n}\hat{v}^2}{\hat{x}}\frac{\partial (nv^2)}{\partial x} + \frac{q_e\hat{n}\hat{T}}{\hat{x}}\frac{\partial (2nT)}{\partial x} = s_v, \tag{B.1.2}$$

$$\frac{q_e\hat{n}\hat{T}}{\hat{t}}\frac{\partial (3nT)}{\partial t} + \frac{m\hat{n}\hat{v}^2}{\hat{t}}\frac{\partial (\frac{1}{2}nv^2)}{\partial t} + \frac{q_e\hat{n}\hat{T}\hat{v}}{\hat{x}}\frac{\partial (5nTv)}{\partial x} + \frac{\hat{n}\hat{v}^3}{\hat{x}}\frac{\partial (\frac{1}{2}mnv^3)}{\partial x} - \frac{\hat{T}}{\hat{x}^2}\frac{\partial}{\partial x}\left(\kappa_\parallel^e\frac{\partial T}{\partial x}\right) = Q. \tag{B.1.3}$$

Mind that the variables are now dimensionless, the same symbols are still used, but that is to avoid a lot of different notations. Dividing Equation (B.1.1) by $\frac{\hat{n}}{\hat{t}}$, (B.1.2) by $\frac{m\hat{n}\hat{v}}{\hat{t}}$ and (B.1.3) by $\frac{m\hat{n}\hat{v}^2}{\hat{t}}$ results in:

$$\begin{aligned} \frac{\partial n}{\partial t} + \frac{\hat{t}\hat{v}}{\hat{x}}\frac{\partial (nv)}{\partial x} &= \frac{\hat{t}}{\hat{n}}s_n, \\ \frac{\partial (nv)}{\partial t} + \frac{\hat{t}\hat{v}}{\hat{x}}\frac{\partial (nv^2)}{\partial x} + \frac{q_e\hat{t}\hat{T}}{m\hat{x}\hat{v}}\frac{\partial (2nT)}{\partial x} &= \frac{\hat{t}}{m\hat{n}\hat{v}}s_v, \\ \frac{q_e\hat{T}}{m\hat{v}^2}\frac{\partial (3nT)}{\partial t} + \frac{\partial (\frac{1}{2}nv^2)}{\partial t} + \frac{q_e\hat{t}\hat{T}}{m\hat{v}\hat{x}}\frac{\partial (5nTv)}{\partial x} + \frac{\hat{t}\hat{v}}{\hat{x}}\frac{\partial (\frac{1}{2}nv^3)}{\partial x} - \frac{\hat{T}\hat{t}}{m\hat{n}\hat{x}^2\hat{v}^2}\frac{\partial}{\partial x}\left(\kappa_\parallel^e\frac{\partial T}{\partial x}\right) &= \frac{\hat{t}}{m\hat{n}\hat{v}^2}Q. \end{aligned}$$

To decrease the amount of constants choose $\hat{x} = \hat{v}\hat{t}$ and $\hat{v} = \sqrt{q_e\hat{T}/m}$. The equations then become:

$$\frac{\partial n}{\partial t} + \frac{\partial(nv)}{\partial x} = \frac{\hat{t}}{\hat{n}}s_n,$$
$$\frac{\partial(nv)}{\partial t} + \frac{\partial(nv^2)}{\partial x} + \frac{\partial(2nT)}{\partial x} = \frac{\hat{t}}{m\hat{n}\hat{v}}s_v,$$
$$\frac{\partial(3nT)}{\partial t} + \frac{\partial(\frac{1}{2}nv^2)}{\partial t} + \frac{\partial(5nTv)}{\partial x} + \frac{\partial(\frac{1}{2}nv^3)}{\partial x} - \frac{\hat{t}}{q_e\hat{n}\hat{x}^2}\frac{\partial}{\partial x}\left(\kappa_\parallel^e\frac{\partial T}{\partial x}\right) = \frac{\hat{t}}{m\hat{n}\hat{v}^2}Q.$$

Applying the same scaling to the neutral equations, with $n_n \equiv \hat{n}n_n$ results in:

$$\frac{\partial n_n}{\partial t} - \frac{\hat{t}}{(\hat{x}\sin\theta)^2}\frac{\partial}{\partial x}\left(D\frac{\partial n_n}{\partial x}\right) = \frac{\hat{t}}{\hat{n}}s_{n_n}.$$

The scaling that is chosen is:

- $\hat{x} = L$, to make the scaled length of the domain equal to 1.
- $\hat{T} = 100$ [eV], which is a typical plasma temperature in the SOL.
- $\hat{n} = 10^{20}[\mathrm{m}^{-3}]$, which is a typical particle density for divertor plasmas.
- $\hat{v} = \sqrt{q_e\hat{T}/m}$, which implies that $\hat{v} = \sqrt{\frac{100q_e}{m}}$ [m/s].
- $\hat{x} = \hat{v}\hat{t}$, which implies that $\hat{t} = L\sqrt{\frac{m}{100q_e}}$ [s].

B.2 Equivalence of systems of PDEs

To show that the homogeneous systems of PDEs of [4] and [3] are equivalent, start with the system of PDEs of [3]:

$$\frac{\partial n}{\partial t} = -\frac{\partial(nv)}{\partial x}, \tag{B.2.1}$$
$$\frac{\partial}{\partial t}\left(\frac{3}{2}p\right) = -\frac{\partial q}{\partial x} + v\frac{\partial p}{\partial x}, \tag{B.2.2}$$
$$\frac{\partial}{\partial t}(mnv) = -\frac{\partial}{\partial x}\left(mnv^2 + p\right), \tag{B.2.3}$$
$$q = \frac{5}{2}pv - \kappa_\parallel^e\frac{\partial T}{\partial x}, \tag{B.2.4}$$
$$p = 2q_enT. \tag{B.2.5}$$

To make this system equivalent to Nakazawa's equations, the time derivative of the energy equation (B.2.2) must become $\frac{\partial}{\partial t}\left(\frac{3}{2}p + \frac{1}{2}mnv^2\right)$. First derive an expression for $\frac{\partial}{\partial t}\left(\frac{1}{2}mnv^2\right)$. To do this rewrite it to

$$\frac{\partial}{\partial t}\left(\frac{1}{2}mnv^2\right) = \frac{\partial}{\partial v}\left(\frac{1}{2}mnv^2\right)\frac{\partial v}{\partial t} + \frac{\partial}{\partial n}\left(\frac{1}{2}mnv^2\right)\frac{\partial n}{\partial t} = mnv\frac{\partial v}{\partial t} + \frac{1}{2}mv^2\frac{\partial n}{\partial t}.$$

Do the same for $\frac{\partial}{\partial t}(mnv)$:

$$\frac{\partial}{\partial t}(mnv) = mn\frac{\partial v}{\partial t} + mv\frac{\partial n}{\partial t}.$$

This implies that

$$\frac{\partial}{\partial t}\left(\frac{1}{2}mnv^2\right) - v\frac{\partial}{\partial t}(mnv) = -\frac{1}{2}mv^2\frac{\partial n}{\partial t}.$$

Substitute equations (B.2.1) and (B.2.3):

$$\frac{\partial}{\partial t}\left(\frac{1}{2}mnv^2\right) = -v\frac{\partial}{\partial x}\left(mnv^2 + p\right) + \frac{1}{2}mv^2\frac{\partial(vn)}{\partial x}.$$

Add this to equation (B.2.2):

$$\frac{\partial}{\partial t}\left(\frac{3}{2}p + \frac{1}{2}mnv^2\right) = -\frac{\partial q}{\partial x} + v\frac{\partial p}{\partial x} - v\left(\frac{\partial}{\partial x}(mnv^2 + p)\right) + \frac{1}{2}mv^2\frac{\partial(vn)}{\partial x},$$

which is equivalent to,

$$\frac{\partial}{\partial t}\left(\frac{3}{2}p + \frac{1}{2}mnv^2\right) = -\frac{\partial q}{\partial x} - v\left(\frac{\partial}{\partial x}(mnv^2)\right) + \frac{1}{2}mv^2\frac{\partial(vn)}{\partial x}.$$

To prove that this equation is equivalent to the energy conservation equation of [4], it must be shown that:

$$\frac{\partial}{\partial x}\left(\frac{1}{2}mnv^3\right) = v\frac{\partial}{\partial x}(mnv^2) - \frac{1}{2}mv^2\frac{\partial(vn)}{\partial x},$$

which can be done using the product rule. Hence the two homogeneous systems of PDEs are equivalent.

□

B.3 SD1D settings

To make the SD1D model parameters and rate coefficients as similar as possible to those of this report some adjustments were made. The most important ones are listed here.

1. `NOUT` is the number of solutions that are saved. The time between the solutions is always the same. `TIMESTEP` is the number of steps between the saved solutions. One step is $\Delta t = 1/\Omega_{c_s} \approx 2.1 \times 10^{-10}$ [s], where Ω_{c_s} is the ion cyclotron frequency. The steps can be longer than Δt, if a solution is in between two steps then it is found by interpolation. So `TIMESTEP` is taken large and `NOUT` small such that interpolating influences the run time as little as possible.

2. `evolve_vn` and `evolve_pn` are false, if they are true then the particle, momentum and energy equation conservation equations for the neutral particles will be used instead of a single diffusion equation.

3. The functions for the rate coefficients are in the file radiation.cxx. Choosing the class `HydrogenRadiatedPower` instead of `UpdatedRadiatedPower` for the rate coefficients and changing the recombination rate function to the one in Section 2.2.5 will make the source terms the same as those used in our schemes.

4. The calculation of the neutral diffusion coefficient in the source code of SD1D is changed to be similar to the neutral coefficient in Equation (2.2.24).

5. In the SD1D code there are several optional sources that can be included in the model. For this thesis only ionisation, recombination, charge exchange and radiation are used. If only a diffusion model is used for the neutral dynamics, as is in our case, then $v_n = 0$ and $T_n = T$. Taking this into account, 4 source terms can be defined.
   A particle sink:
   $$S = -nn_n\langle\sigma v\rangle_{\text{ion}} + n^2\langle\sigma v\rangle_{\text{rec}}.$$
   A friction sink:
   $$F = mv\left(nn_n\langle\sigma v\rangle_{\text{cx}} + n^2\langle\sigma v\rangle_{\text{rec}}\right).$$
   An energy sink term:
   $$E = \frac{3}{2}T\langle\sigma v\rangle_{\text{rec}} - \frac{3}{2}\cdot T \text{ eV}\langle\sigma v\rangle_{\text{ion}}.$$
   A radiation term:
   $$R = (1.09T - 13.6\text{eV})\langle\sigma v\rangle_{\text{rec}} + 30 \text{ eV}\langle\sigma v\rangle_{\text{ion}} + W_{imp}.$$

C Numerical models

C.1 The diffusion discretization

Well-posedness of the diffusion scheme

The diffusion terms are discretized according to the scheme in Equation (3.1.7). We prove it is well-posed using the same method as in [15, Chapter 9]. Define the discretization operator S by:

$$Su_j = \frac{1}{2\Delta x^2}\left((\alpha_{j+1}+\alpha_j)(u_{j+1}-u_j) - (\alpha_j+\alpha_{j-1})(u_j-u_{j-1})\right),$$

where α_j is the diffusion coefficient at x_j which is positive. Assume that $e^{i\phi_l j}$ is an eigenvector of S for some ϕ_l with eigenvalue $\Omega(\phi_l)$ and substitute it for u_j:

$$\frac{1}{2\Delta x^2}\Big((\alpha_{j+1}+\alpha_j)(e^{i\phi_l(j+1)}-e^{i\phi_l j}) - (\alpha_j+\alpha_{j-1})(e^{i\phi_l j}-e^{i\phi_l(j-1)})\Big).$$

This is equal to

$$\frac{1}{2\Delta x^2}e^{i\phi_l j}\Big((\alpha_{j+1}+\alpha_j)(e^{i\phi_l}-1) - (\alpha_j+\alpha_{j-1})(1-e^{-i\phi_l})\Big).$$

So the eigenvalue $\Omega(\phi_j)$ is equal to $\frac{1}{2\Delta x^2}\big((\alpha_{j+1}+\alpha_j)(e^{i\phi_l}-1) - (\alpha_j+\alpha_{j-1})(1-e^{-i\phi_l})\big)$. For the operator S to be well-posed, the real part of the eigenvalues must be non-positive, so:

$$\frac{1}{2\Delta x^2}\left((\alpha_{j+1}+\alpha_j)(\cos\phi_l-1) - (\alpha_j+\alpha_{j-1})(1-\cos\phi_l)\right) \leq 0.$$

This is equavalent to

$$(\cos\phi_l - 1)(\alpha_{j+1}+2\alpha_j+\alpha_{j-1}) \leq 0,$$

which is non-positive because $\cos\phi_l \leq 1$. Hence the operator is well-posed.

Convergence of the diffusion scheme

To calculate the order of convergence of this scheme, substitute the Taylor approximations for α and u around x_j in the scheme. This results in:

$$\frac{1}{2\Delta x^2}\Bigg(\left(2\alpha+\Delta x\alpha'+\frac{\Delta x^2}{2}\alpha''+\frac{\Delta x^3}{6}\alpha'''+O(\Delta x^4)\right)\left(\Delta x u'+\frac{\Delta x^2}{x}u''+\frac{\Delta x^3}{6}u'''+O(\Delta x^4)\right)$$
$$-\left(2\alpha-\Delta x\alpha'+\frac{\Delta x^2}{2}\alpha''-\frac{\Delta x^3}{6}\alpha'''+O(\Delta x^4)\right)\left(\Delta x u'-\frac{\Delta x^2}{x}u''+\frac{\Delta x^3}{6}u'''+O(\Delta x^4)\right)\Bigg),$$

where $\alpha \equiv \alpha(x_j)$ and $u \equiv u(x_j)$ to shorten the equation, the same notation holds for their derivatives. Rewrite the equation to:

$$\frac{1}{2\Delta x^2}\left(\Delta x u'\left(2\Delta x\alpha' + \frac{\Delta x^3}{3}\alpha'''\right) + \frac{\Delta x^2}{2}u''\left(4\alpha + \Delta x^2\alpha''\right)\right.$$
$$\left. + \frac{\Delta x^3}{6}u'''\left(2\Delta x\alpha' + \frac{\Delta x^3}{3}\alpha'''\right) + O(\Delta x^4)\right).$$

This is equal to

$$u'\alpha' + u''\alpha + O(\Delta x^2) = \left(\alpha u'\right)' + O(\Delta x^2),$$

hence the scheme is second order convergent.

C.2 Alternative boundary conditions

In earlier models the condition $\frac{\partial T}{\partial x} = 0$ at $x = L$ was implemented on the heat diffusion term. This implied that the sheath heat transmission became $\frac{1}{2}nv^3 + 5nvT = \gamma nvT$. This was implemented as a condition on the first order derivatives, so according to Section 2.2.2 there could not be any more boundary conditions. Hence the Bohm-criterion was not implemented at first. The results are given in Figure C.1. The velocity becomes negative which makes the system unable to remove the heat causing T to diverge until it becomes too large for the simulation.

Hence to still have the Bohm-criterion, it is implemented as a source term in the momentum (2.2.6) equation. This also coincides with the physical meaning of the Bohm-criterion, namely a magnetic force. The source only acts on the last grid point since the Debye sheath is too thin for any reasonable grid size. Lacking a more scientific formula for the magnetic force, a crude control function was used to force the velocity to be equal to the ion thermal velocity, which reads:

$$S_{\text{Bohm}} = C \cdot n_N(\sqrt{2T_N} - v_N). \tag{C.2.1}$$

The result is given in Figure C.2, in Figure C.3 is the solution for the same parameters with the boundary conditions of table 2.1 for comparison. The solutions are almost identical, but we chose to use the conditions in table 2.1 because these agree with Section 2.2.2.

C.3 Analytic Jacobian of the Flux-splitting scheme

In this section we briefly explain the calculation and implementation of the analytic Jacobian that is implemented to improve the computation time for `ode15i()` and can be used for system control. We also discuss how we checked the implementation in Matlab for mistakes.

The diffusion and source terms in Equation (3.3.7) are of the form $B\underline{f}$, where B is a matrix and $\underline{f}$ is a vector, both are dependent on $\underline{n}$, $\underline{v}$, $\underline{T}$ and $\underline{n}_n$. The terms of the neutral equation are only multiplied by the identity matrix so these will be ignored. Again the notation $\underline{y} = (\underline{n}; \underline{v}; \underline{T})$ is used. The Jacobian of $B\underline{f}$ is

$$\frac{\partial(B\underline{f})}{\partial\underline{y}} = \left(\frac{\partial B}{\partial y_1}\underline{f}, ..., \frac{\partial B}{\partial y_{3N}}\underline{f}\right) + B\frac{\partial\underline{f}}{\partial\underline{y}}. \tag{C.3.1}$$

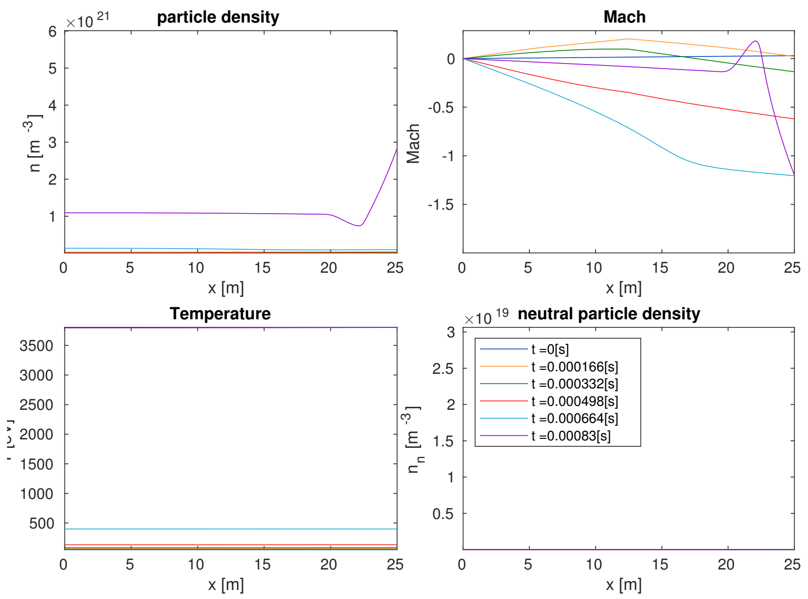


Figure C.1: Solution without the Bohm-criterion.

In this case $B = \left(\frac{\partial \underline{F}}{\partial \underline{y}}\right)^{-1}$ where $\underline{F}$ consists out of three functions $\underline{F}^p$, $\underline{F}^m$ and $\underline{F}^E$ for the particle, momentum and energy equation respectively. By definition F_i^E is dependent on the variables at grid point i, idem for $\underline{F}^p$ and $\underline{F}^m$. The same holds for $\underline{f}$, therefore the Jacobian can be given by:

$$\frac{\partial (B\underline{f})}{\partial \underline{y}} = \left(\frac{\partial B}{\partial \underline{n}} \begin{pmatrix} \mathrm{diag}(\underline{f}^p) \\ \mathrm{diag}(\underline{f}^m) \\ \mathrm{diag}(\underline{f}^E) \end{pmatrix}, \ldots, \frac{\partial B}{\partial \underline{T}} \begin{pmatrix} \mathrm{diag}(\underline{f}^p) \\ \mathrm{diag}(\underline{f}^m) \\ \mathrm{diag}(\underline{f}^E) \end{pmatrix} \right) + B\frac{\partial \underline{f}}{\partial \underline{y}}. \tag{C.3.2}$$

By the definition of $\left(\frac{\partial \underline{F}}{\partial \underline{y}}\right)^{-1}$ in Equation (3.3.8) the matrix $\frac{\partial B}{\partial \underline{n}}$ is defined by a block matrix consisting out of nine $N \times N$ diagonal matrices:

$$\left(\begin{array}{ccccccccc} 0 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 0 & & & & & & \\ v_1/n_1^2 & & & -1/n_1^2 & & & & & \\ & \ddots & & & \ddots & & & & \\ & & v_N/n_N^2 & & & -1/n_N^2 & & & \\ -\frac{v_1^2/6-T_1}{n_1^2} & & & v_1/3/n_1^2 & & & -1/3/n_1^2 & & \\ & \ddots & & & \ddots & & & \ddots & \\ & & -\frac{v_N^2/6-T_N}{n_N^2} & & & v_N/3/n_N^2 & & & -1/3/n_N^2 \end{array} \right). \tag{C.3.3}$$

A similar definition is used for $\frac{\partial B}{\partial \underline{v}}$ and $\frac{\partial B}{\partial \underline{T}}$. Using definition (C.3.2) instead of (C.3.1) allows you to have 3 sparse matrix multiplications instead of $3N$ sparse matrix vector multiplications. To get the Jacobian of $A^+PD_{\mathrm{upw}}\underline{y}$ consider that A_i^+ is only dependent on n_i, v_i and T_i, hence

$$\frac{\partial A_i^+}{\partial n_i} = \frac{\partial V_i^R}{\partial n_i}\Lambda_i^+ V_i^L + V_i^R \frac{\partial \Lambda_i^+}{\partial n_i} V_i^L + V_i^R \Lambda_i^+ \frac{\partial V_i^L}{\partial n_i}.$$

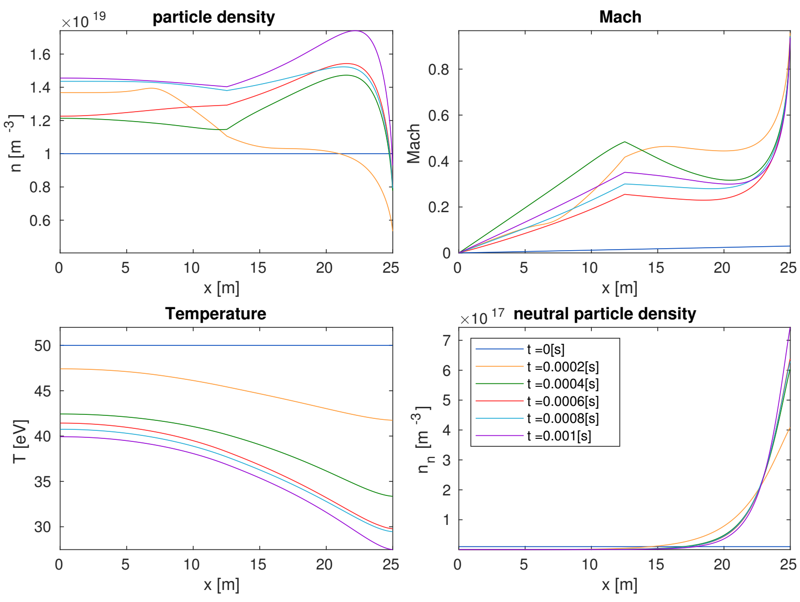


Figure C.2: Solution with the Bohm-criterion implemented as a source term.

The positive and negative eigenvalue matrices are defined by:

$$\Lambda^+ = \frac{1}{2}\left(\Lambda + |\Lambda|\right),$$
$$\Lambda^- = \frac{1}{2}\left(\Lambda - |\Lambda|\right).$$

Because A^+ is a block diagonal matrix we can define $\frac{\partial A^+}{\partial \underline{n}}$ by,

$$\frac{\partial A^+}{\partial \underline{n}} = \begin{pmatrix} \frac{\partial A_1^+}{\partial n_1} & & \\ & \ddots & \\ & & \frac{\partial A_N^+}{\partial n_N} \end{pmatrix}.$$

Idem for A^- and the derivatives w.r.t. $\underline{v}$ and $\underline{T}$. The Jacobian of $A^+PD_{\text{upw}}\underline{y}$ is

$$\frac{\partial}{\partial \underline{y}}\left(A^+PD_{\text{upw}}\underline{y}\right) = \left(\frac{\partial A^+}{\partial y_1}PD_{\text{upw}}\underline{y}, ..., \frac{\partial A^+}{\partial y_{3N}}PD_{\text{upw}}\underline{y}\right) + A^+PD_{\text{upw}}.$$

The term

$$\left(\frac{\partial A^+}{\partial y_1}PD_{\text{upw}}\underline{y}, ..., \frac{\partial A^+}{\partial y_{3N}}PD_{\text{upw}}\underline{y}\right)$$

is found by permuting the entries of

$$\left(\frac{\partial A^+}{\partial \underline{n}}PD_{\text{upw}}\underline{y}, \frac{\partial A^+}{\partial \underline{v}}PD_{\text{upw}}\underline{y}, \frac{\partial A^+}{\partial \underline{T}}PD_{\text{upw}}\underline{y}\right).$$

Using these equations the Jacobian of (3.3.7) can be constructed with addition and concatenation of matrices. The Jacobian was tested by approximating it numerically with central difference and comparing the numerical one with the analytic Jacobian. Errors could then be located by

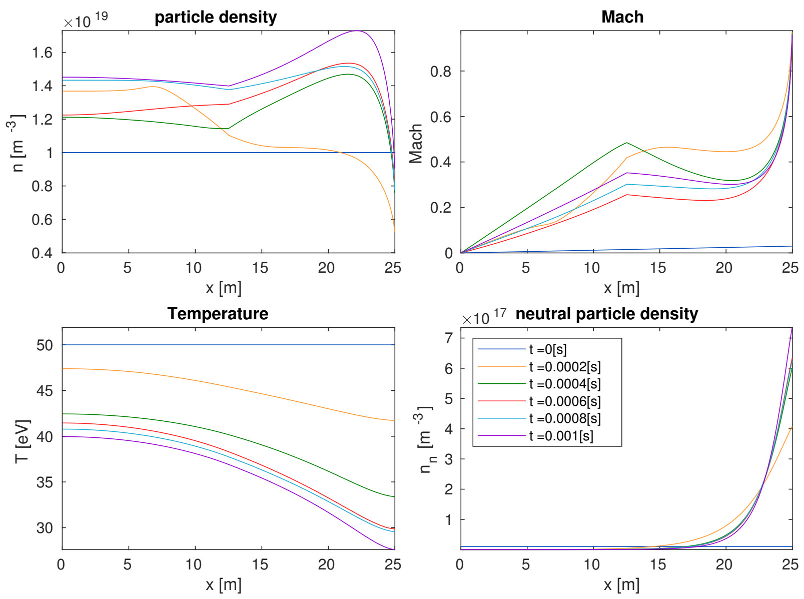


Figure C.3: Solution using the current model for the same parameters instead of implementing the Bohm-criterion using a source term.

using Matlabs rounding function and the sparse matrix representation. An example is in Figure C.4.

We continued looking for mistakes in the implementation of the Jacobian until the difference between the analytic and the numerical Jacobian was small enough. The most dominant term is the derivative of the scheme of the heat conduction. The functions in this scheme are of the form $CT^{7/2}$ or a lower power, only a power of $7/2$ is considered because this is the biggest one. To find the minimal difference we can expect when comparing the numerical Jacobian with the implementation of the analytic Jacobian we will compute the difference for this function, because this function was the most dominant cause of the differences. In this case $C = \frac{\kappa_0 \hat{T} \hat{t}}{\hat{x}^2 \hat{n} q_e \Delta x^2} \approx \frac{8}{\Delta x^2}$ and $T \in (0, 1]$ because it is scaled. The derivative is calculated analytically which results in $\frac{7}{2}CT^{5/2}$ and approximated numerically by central difference and then we compare the differences. The differences are shown in Figure C.5. These difference are used to determine whether or not the differences between the numerical and analytic Jacobian are small enough.

A second method to evaluate the implementation of the Jacobian is to compare simulations that use the Jacobian with simulations that only use the `JPattern` option of `odeset`. With `JPattern` we can define which entries of the Jacobian are nonzero. These will then be approximated numerically. `JPattern` must be implemented else wise the simulation will have complexity $O(N^2)$ which makes it too slow for checking for errors. If both simulations are similar then the implementation of the Jacobian is correct, or contains only negligible errors. Using `Jpattern` is also an easy to implement alternative to the Jacobian that reduces the calculation of the Jacobian to order $O(N)$ instead of $O(N^2)$.

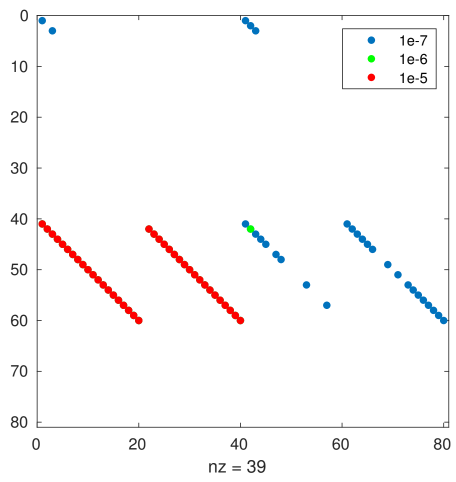


Figure C.4: Example of the visual representation of errors in the analytic Jacobian. The colours indicate the approximate size of the error. The entire matrix contains $4N \times 4N$ points, $N = 20$ in this case.

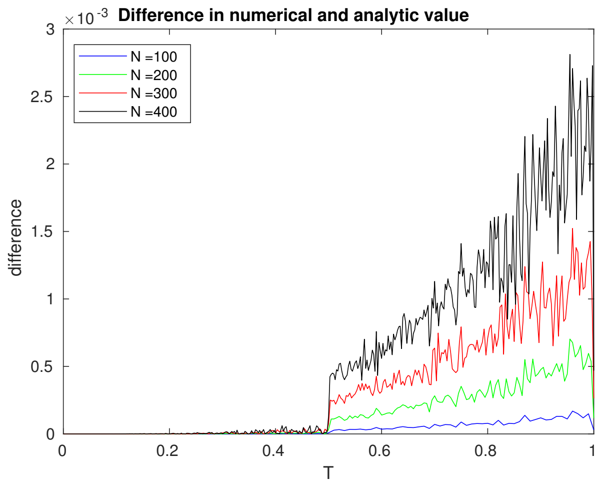


Figure C.5: Differences in the numerical approximation of a function and its analytical value.

C.4 Method of Manufactured Solutions

The manufactured solutions (3.4.2)-(3.4.5) have to meet the following scaled boundary conditions:

$$\frac{\partial n}{\partial x} = 0 \qquad \text{at } x = 0, \tag{C.4.1}$$

$$v = 0 \qquad \text{at } x = 0, \tag{C.4.2}$$

$$\frac{\partial T}{\partial x} = 0 \qquad \text{at } x = 0, \tag{C.4.3}$$

$$\frac{\partial n_n}{\partial x} = 0 \qquad \text{at } x = 0, \tag{C.4.4}$$

$$T = \frac{v^2}{2} \qquad \text{at } x = L, \tag{C.4.5}$$

$$\frac{1}{2}nv^3 + 5nvT - \kappa_\parallel^e \frac{\partial T}{\partial x} = \gamma nvT \qquad \text{at } x = L, \tag{C.4.6}$$

$$D\frac{\partial n_n}{\partial x} = \hat{x}\hat{v}nv\sin^2\theta \qquad \text{at } x = L. \tag{C.4.7}$$

To satisfy (C.4.1) and (C.4.2) define n and v as:

$$n(x,t) = e^{-t}\left(2 + \cos\left(\frac{\pi x}{L}\right)\right), \tag{C.4.8}$$

$$v(x,t) = e^{-t}\sin\left(\frac{2.5\pi x}{L}\right). \tag{C.4.9}$$

These definitions also imply that $n(L,t) = v(L,t) = e^{-t}$, so to satisfy conditions (C.4.4) and (C.4.7) choose n_n as

$$n_n(x,t) = \frac{\sin^2\theta \cdot 2L\hat{x}\hat{v}}{D \cdot 3\pi} e^{-2t}\left(2 + \cos\left(\frac{3\pi}{2L}\right)\right). \tag{C.4.10}$$

Define T as the product of two functions, $T(x,t) = T_t(t)T_x(x)$. Substitute $v(x,t)$ and $n(x,t)$ into conditions (C.4.5) and (C.4.6):

$$T_t(t)T_x(L) = \frac{v(L)^2}{2} = \frac{1}{2}e^{-2t}.$$

This implies that $T_t(t) = e^{-2t}$ and $T_x(L) = \frac{1}{2}$. Substitute this into Equation (C.4.6):

$$e^{-4t}\left(\frac{1}{2} + 5T_x(L)\right) - \kappa_\parallel^e e^{-2t}\left.\frac{\partial T_x}{\partial x}\right|_{x=L} = \gamma e^{-4t}T_x(L).$$

So choose $\kappa_\parallel^e(t) = \kappa_0 e^{-2t}$.
Substitute (C.4.5) into (C.4.6) to get

$$\kappa_\parallel^e \frac{\partial T}{\partial x} = (3 - \gamma/2)nv^3 \text{ at } x = L.$$

Insert the values and functions of n, v, T and $\kappa_\parallel^e$:

$$\kappa_0 \frac{\partial T_x}{\partial x} = (3 - \gamma/2) \text{ at } x = L. \tag{C.4.11}$$

Choose $T_x(x) = A\cos(Bx) + C$. Then condition (C.4.3) is satisfied automatically. If $B = \frac{\pi}{2L}$ then to satisfy condition (C.4.5) it must be that $C = \frac{1}{2}$. Substituting the resulting function for T into (C.4.6) results in the following equation for A:

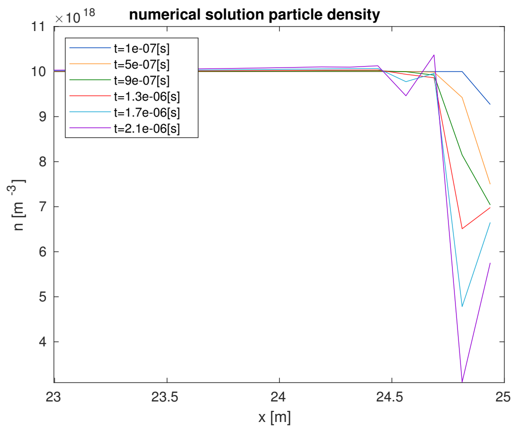
$$-\kappa_0 A\frac{\pi}{2L}\sin\frac{\pi}{2L}L = -\kappa_0 A\frac{\pi}{2L} = 3 - \gamma/2.$$

Hence $A = \frac{3-\gamma/2}{-\kappa_0\pi/2/L}$. So T becomes

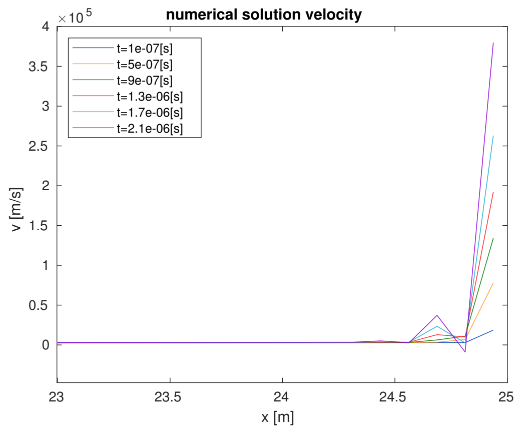
$$T(x,t) = e^{-2t}\left(\frac{3 - \frac{\gamma}{2}}{-\kappa_0\frac{\pi}{2L}}\cos\left(\frac{\pi}{2L}x\right) + \frac{1}{2}\right). \tag{C.4.12}$$

C.5 Semi-explicit scheme

For the first order derivatives in the Semi-Explicit scheme several discretization methods have been applied. Some of these methods result in numerical errors, examples of those errors are shown in Figure C.6 - C.8.



(a) Numerical solution of n.



(b) Numerical solution of v.

Figure C.6: Errors in the numerical solutions calculated using the central difference scheme.

C.5.1 Central Difference

The results are in Figure C.6.

C.5.2 Central difference and upwind scheme

Using a mix of central difference and upwind creates numerical diffusion terms, the appearance of these terms and the results of this scheme are discussed below.

Only the homogeneous particle conservation equation and the convection part of the momentum equation are considered (Equations (C.5.1), (C.5.2)). The pressure term and source terms are ignored, because these behave as expected. First take a general spatial discretization, this is denoted by $\frac{\partial v}{\partial x} \approx \frac{\delta v_j}{\delta x}$. The remaining scheme is:

$$\frac{\partial n_j}{\partial t} = -\frac{\delta v_j}{\delta x}n_j - \frac{\delta n_j}{\delta x}v_j, \tag{C.5.1}$$

$$\frac{\partial n_j}{\partial t}v_j + \frac{\partial v_j}{\partial t}n_j = -2\frac{\delta v_j}{\delta x}n_j v_j - \frac{\delta n_j}{\delta x}(v_j)^2. \tag{C.5.2}$$

Subtract Equation (C.5.1) v_j times from (C.5.2):

$$\frac{\partial n_j}{\partial t} = -\frac{\delta v_j}{\delta x}n_j - \frac{\delta n_j}{\delta x}v_j, \tag{C.5.3}$$

$$\frac{\partial v_j}{\partial t}n_j = -\frac{\delta v_j}{\delta x}n_j v_j. \tag{C.5.4}$$

So if the same spatial discretization is used for all derivatives, there is no numerical diffusion term $\frac{u_{j+1}-2u_j+u_{j-1}}{\Delta x^2}$.

We chose to use upwind for n in the particle conservation equation and for v in the momentum equation. This is done because the next solution for n is calculated using the particle equation and the next solution for v using the momentum equation. The other spatial derivatives are discretized with central difference. Equations (C.5.1), (C.5.2) become:

$$\frac{\partial n_j}{\partial t} = -\frac{v_{j+1}-v_{j-1}}{2\Delta x}n_j - \frac{n_j - n_{j-1}}{\Delta x}v_j, \tag{C.5.5}$$

$$\frac{\partial n_j}{\partial t}v_j + \frac{\partial v_j}{\partial t}n_j = -2\frac{v_j - v_{j-1}}{\Delta x}n_j v_j - \frac{n_{j+1}-n_{j-1}}{2\Delta x}(v_j)^2. \tag{C.5.6}$$

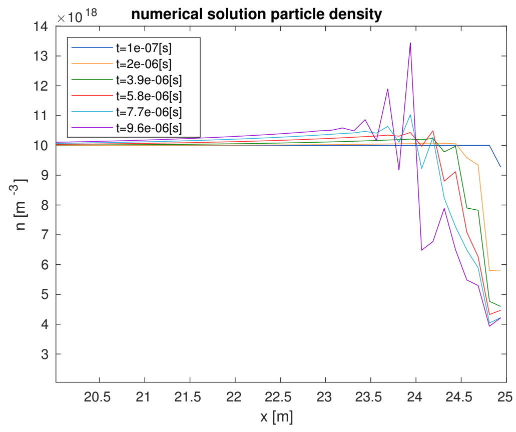
Again subtracting the Equation (C.5.5) v_j times from Equation (C.5.6) results in

$$\frac{\partial v_j}{\partial t}n_j = -(v_j)^2\left(\frac{n_{j+1}-n_{j-1}}{2\Delta x} - \frac{n_j - n_{j-1}}{\Delta x}\right) - n_j v_j\left(2\frac{v_j - v_{j-1}}{\Delta x} - \frac{v_{j+1}-v_{j-1}}{2\Delta x}\right).$$

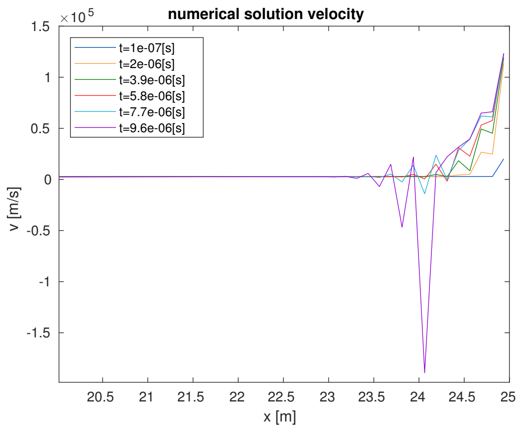
This can be rewritten to

$$\frac{\partial v_j}{\partial t}n_j = -\frac{\Delta x}{2}(v_j)^2\frac{n_{j+1}-2n_j+n_{j-1}}{\Delta x^2} - n_j v_j\left(-\frac{\Delta x}{2}\frac{v_{j+1}-2v_j+v_{j-1}}{\Delta x^2} + \frac{v_j - v_{j-1}}{\Delta x}\right).$$

So we have to same equation for the momentum equation as with the first method (C.5.4), but with 2 additional numerical diffusion terms: $\frac{n_{j+1}-2n_j+n_{j-1}}{\Delta x^2}$ and $\frac{v_{j+1}-2v_j+v_{j-1}}{\Delta x^2}$.



(a) Numerical solution of n.



(b) Numerical solution of v.

Figure C.7: Errors in the numerical solutions calculated using a mix of central difference and upwind.

C.5.3 Biased upwind scheme

The biased upwind scheme is used because of the conjecture that it is less prone to have numerical oscillations than the central difference scheme, because it is a combination between second order upwind and central difference. If we consider the central difference scheme for the simple equation $a\frac{\partial u}{\partial x}+bu=0$ we get $a\frac{u_{j+1}-u_{j-1}}{2\Delta x}+bu_j=0$. Substituting λ^j for u_j results in the following expression for λ,

$$\lambda_{1,2} = \frac{-\frac{2\Delta xb}{a} \pm \sqrt{\left(\frac{2\Delta xb}{a}\right)^2+4}}{2}.$$

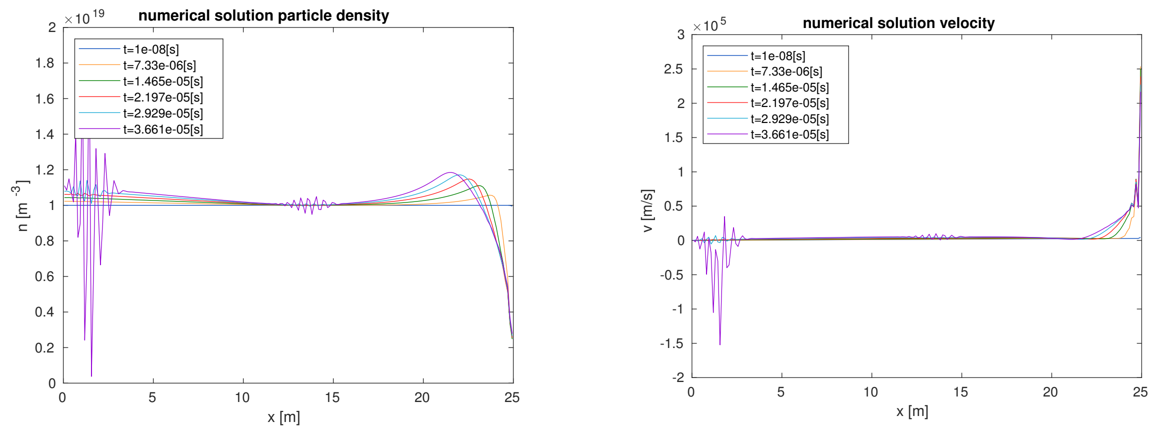
Thus $u_j = c_1\lambda_1^j + c_2\lambda_2^j$ where c_1 and c_2 are constants. From the definition of λ it follows that one of them is negative and will switch sign for each iteration. We apply the same calculation to the biased upwind scheme,

$$a\frac{3u_j - 4u_{j-1}+u_{j-2}}{2\Delta x}+bu_j = 0.$$

This results in the following expression for λ:

$$\lambda_{1,2} = \frac{4 \pm \sqrt{16 - 4\left(3 + b\frac{2\Delta x}{a}\right)}}{2\left(3 + b\frac{2\Delta x}{a}\right)},$$

now both solutions for λ are non-negative if $3 + b\frac{2\Delta x}{a} \geq 0$. So for this condition there are no numerical oscillations. Hence we presume that a combination of the second order upwind scheme and central difference is less prone to have numerical oscillations. Applying the same calculation to the biased upwind scheme results in a third-order polynomial for λ where the sign of λ is not trivial.



(a) Numerical solution of n.

(b) Numerical solution of v.

Figure C.8: Errors in the numerical solutions calculated using the biased upwind scheme.

C.6 Robustness against shock waves for the Implicit scheme

To study the solution of the Implicit scheme for shock waves the following toy problem is considered:

$$\frac{\partial n}{\partial t} + \frac{\partial (nv)}{\partial x} = 0, \tag{C.6.1}$$

$$\frac{\partial (nv)}{\partial t} + \frac{\partial (nv^2)}{\partial x} + \frac{\partial (2nT)}{\partial x} = 0. \tag{C.6.2}$$

This is equivalent to

$$\frac{\partial n}{\partial t} + \frac{\partial (nv)}{\partial x} = 0, \tag{C.6.3}$$

$$\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} + T/n\frac{\partial n}{\partial x} = 0. \tag{C.6.4}$$

The domain is the interval $[0, 1]$. Only two variables are used, T is a constant. All variables and parameters are dimensionless. The boundary conditions are $v = 0$ and $\frac{\partial n}{\partial x} = 0$ at $x = 0$. The time integration is solved with `ode15i()` The spatial integration with the upwind scheme for convection and downwind for the pressure terms. The upwind and downwind direction changes with the sign of v. Two sets of initial values are considered. In both cases

$v(x,0) = \frac{1}{10}x$. For n there is first a shock wave in the positive direction, here the initial value is $n(x,0) = \begin{cases} 2 \text{ if } x < 1/2 \\ 1 \text{ if } x > 1/2 \end{cases}$. For the second example the shock wave will move in the negative direction, here $n(x,0) = \begin{cases} 1 \text{ if } x < 1/2 \\ 2 \text{ if } x > 1/2 \end{cases}$. The results for the first example are given in Figure C.9 and for the second example in Figure C.10. For both cases the solution is also calculated using a flux-vector splitting scheme for comparison. The flux-vector splitting scheme is implemented in the same manner as the Flux-splitting scheme.

The solution for the upwind scheme has small numerical oscillations at the front of the shock-wave. But the flux-vector splitting solution has more numerical diffusion. The numerical diffusion however decreases for increasing the number of grid points, whereas the numerical oscillations stay. Also the numerical oscillations demand a lot of computation time.

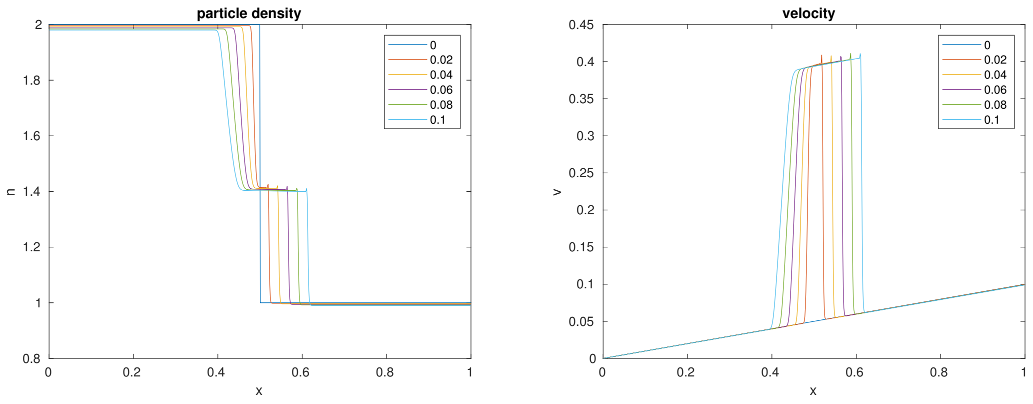
For the second set of initial conditions two different approaches are used for the upwind and downwind scheme. In one case the direction is based on the sign of v, in the other the direction is always positive. The solutions were only calculated for a small time interval. For larger times the first method became unstable. The velocity for the first method also increases for one grid point, even though it should decrease at this point. This is most likely caused by the fact that v should be negative at this point, resulting in the method having to switch the upwind direction twice on these three grid points. If the upwind direction is always positive, then it is possible to compute a solution. This however can only be done if v is not too small, if v becomes too negative then an upwind scheme in the positive direction will become unstable, as is explained in Appendix C.7. The flux-vector splitting scheme does compute a feasible solution, it is also the only method that is stable for longer time intervals.
The upwind direction must be dependent on v as is explained in Appendix C.7. So to circumvent the errors that arise when only one grid point has a negative velocity the central difference scheme is used. This is done in the following way:

$$\text{Method } = \begin{cases} \text{Upwind} & v > \varepsilon \\ \text{Central difference} & |v| \leq \varepsilon \\ \text{Downwind} & v < -\varepsilon \end{cases},$$

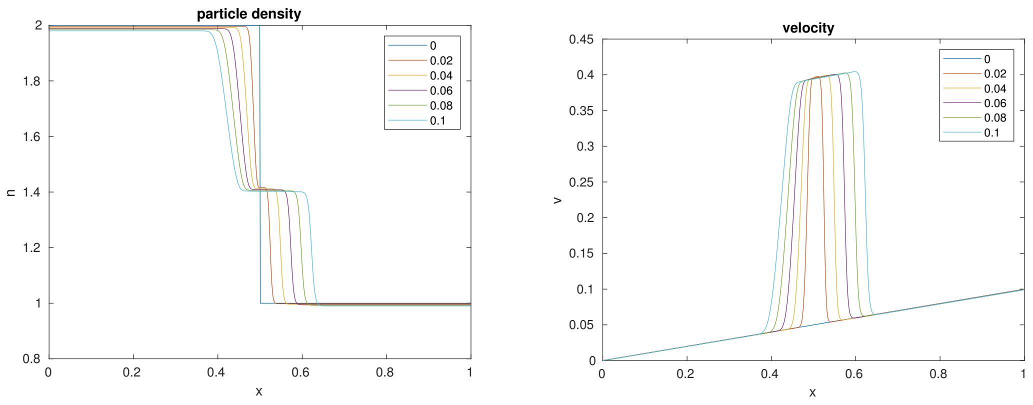
but this scheme also results in numerical errors. Hence we did not find a scheme without flux-vector splitting that is suited to simulate this phenomenon.

C.7 Stability of the upwind scheme with negative velocity

The direction of the upwind and downwind discretization in the Implicit and Semi-Explicit scheme is not dependent on the sign of the velocity. For the convection equation $\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x}$ with $a < 0$ this would cause the problem to become ill-posed. But for our problem (2.2.5)-(2.2.7) the method stays stable as long as v does not become too negative. In this appendix we briefly consider the well-posedness to demonstrate our conjecture for the reason that the problem remains stable. Only a toy-problem is considered because it proved to be too complicated to prove well-posedness for the complete system of PDEs. The proof is done according to the



(a) Upwind/downwind scheme dependent on sign of v.



(b) Flux-vector splitting scheme.

Figure C.9: Solutions for the first set of initial conditions, a shock-wave in the positive direction.

stability analysis in [15, Chapter 9]. The toy problem is:

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + bu = 0,$$

where $a < 0$. Define $\underline{u}$ as the vector such that each entry $u_i = u(x_i)$, define A as the matrix representation of the upwind discretization. Then you get the following semi discrete equation:

$$\frac{\partial \underline{u}}{\partial t} = -(A + bI)\underline{u}.$$

Then for each entry we have the ODE:

$$\frac{\partial u_j}{\partial t} = -a\frac{u_j - u_{j-1}}{\Delta x} - bu_j = Su_i,$$

where S is the discretization operator. Assume that $e^{i\phi_l j}$ is an eigenvector of S for some ϕ_l with eigenvalue $\Omega(\phi_l)$, substitute $u_j = e^{i\phi_l j}$:

$$Se^{i\phi_l j} = -e^{i\phi_l j}\left(\frac{a}{\Delta x}(1 - e^{-i\phi_l}) + b\right).$$

Split the complex power of e into its real and complex part:

$$-e^{i\phi_l j}\left(\frac{a}{\Delta x}(1 - \cos(\phi_l) + i\sin(\phi_l)) + b\right).$$

This gives us an expression for the eigenvalues of S, $\Omega_j = -\left(\frac{a}{\Delta x}(1-\cos(\phi_l)+i\sin(\phi_l))+b\right)$. The system is well-posed if $\mathrm{Re}(\Omega_j) \leq 0$. The real part of Ω_j is:

$$\mathrm{Re}(\Omega_j) = -\frac{a}{\Delta x}(1-\cos(\phi_l)) - b.$$

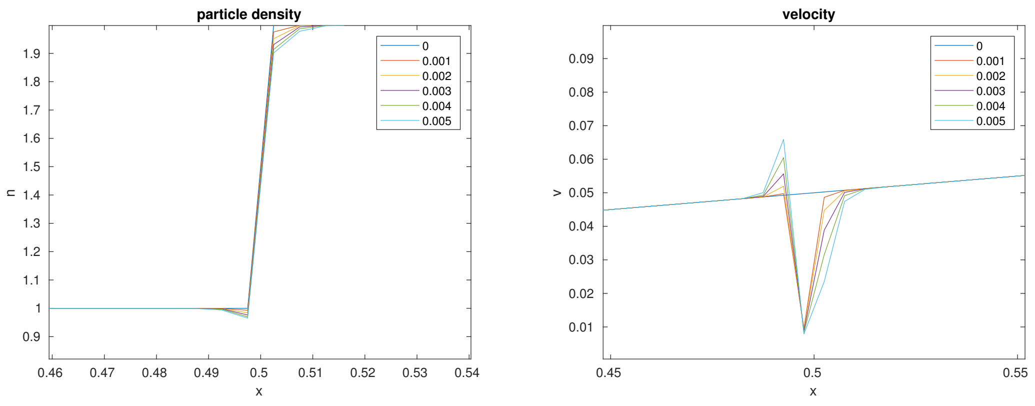
This is less than 0 if $b \geq \frac{a}{\Delta x}(\cos\phi_l - 1)$. To now put this in perspective of the discretization of the system of PDEs (2.2.5)-(2.2.7), consider Equations (3.1.8)-(3.1.10) because these only have one time derivative per equation and the set of equations is equivalent. In each equation v_j represents a, the terms representing b are listed below:

- For the particle equation b is represented by $\frac{v_j - v_{j-1}}{\Delta x}$ plus its source term s_n.
- For the momentum equation the only other term containing v_j without a derivative of v is the source term s_v, so s_v/v represents b.
- For the energy equation there are three terms containing T_j but not containing a derivative of T: $\frac{2}{3}T\frac{\partial v}{\partial x}$, the source term s_Q and the heat diffusion. The last term is dominant for most values of T because of its large diffusion coefficient. So b is represented by $\frac{2}{3}\frac{\partial v}{\partial x}$, the terms in s_Q that contain T and $\kappa_{\|}^{e}$.

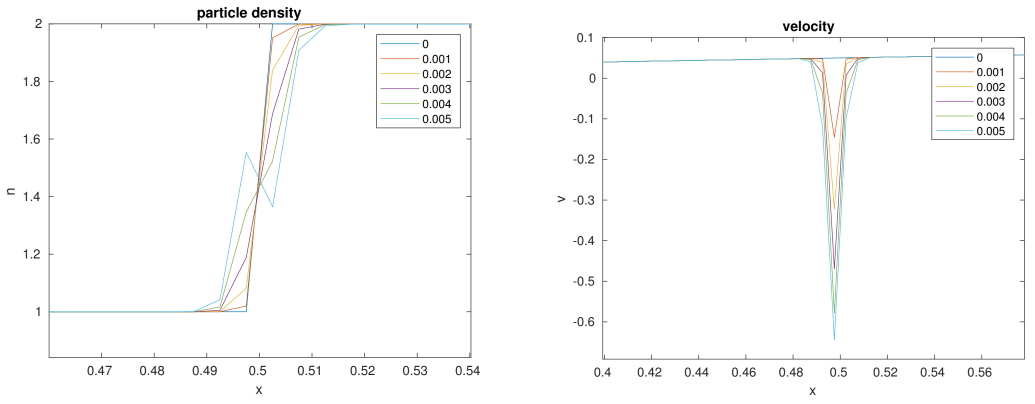
The method is not always stable if it is well-posed. For this the stability condition has to be met as well. For example for Explicit Euler the stability condition is $|1+\Delta t\Omega(\phi_l)| < 1$. This is equivalent with

$$\left(1 - \frac{a\Delta t}{\Delta x}(1-\cos\phi_l) - \Delta t b\right)^2 + \left(\frac{a\Delta t}{\Delta x}\sin\phi_l\right)^2 < 1,$$

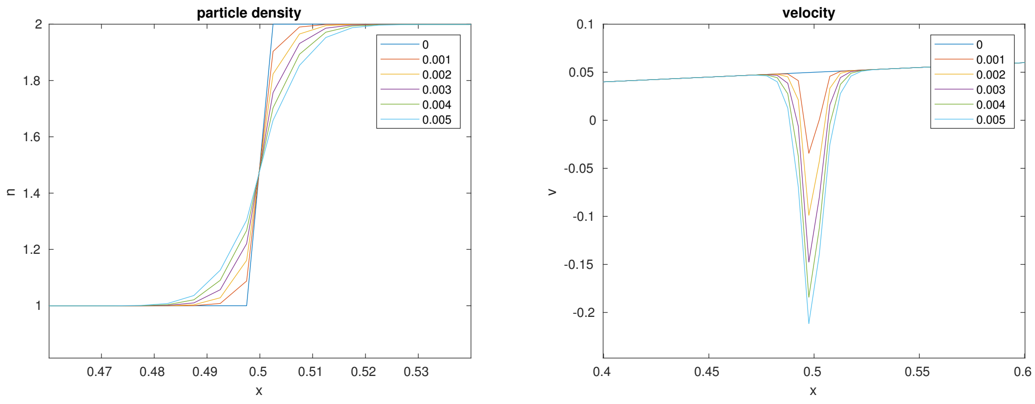
which can not hold if $b < \frac{a}{\Delta x}(\cos\phi_l - 1)$. So the well-posedness condition has to be met.



(a) Upwind/downwind scheme dependent on sign of v.



(b) Upwind/downwind scheme not dependent off the sign of v.



(c) Flux-vector splitting scheme.

Figure C.10: Solutions for the second set of intial conditions, a shock-wave in the negative direction.

Bibliography

[1] Kevin Verhaegh, *Spectroscopic investigations of detachment on TCV*, University of York, December 2019

[2] A. Kallenbach, et al., *analytic calculations for impurity seeded partially detached divertor conditions*, Plasma Phys. Control. Fusion 58

[3] Ben Dudson, *SD1D: 1D divertor model for detachment studies*, December 2016

[4] Nakazawa, Nakajima, Okamoto, Ohyabu, *One-dimensional simulation on stability of detached plasma in a tokamak divertor*, Plasma Phys. Control. Fusion 42, pages 401-413

[5] James Robert Harrison, *Characterisation of Detached Plasmas on the MAST Tokamak*, University of York, September 2010

[6] Niek Lopes Cardozo, *Fusion on the back of an envelope*, TU/e 3MF100

[7] `https://en.wikipedia.org/wiki/Tokamak`, 24-01-2019

[8] `https://www.iter.org/sci/makingitwork`, 24-01-2019

[9] Xiong Hao, Liu Ming-Hai, Chen Ming, Rao Bo, Chen Jie, Chen Zhao-Quan, Xiao Jin-Shui, Hu Xi-Wei. *Radial magnetic field in magnetic confinement device.* Chinese Physics B

[10] G.F.Matthews, *Plasma detachment for divertor targets and limiters*, Journal of Nuclear Materials Volumes 220-222, April 1995, pages 104-116

[11] Hsu W L, Yamada M and Barrett P J, *Experimental Simulations of the gaseous divertor detachment* 1982 Physics Review Letter 49

[12] Sigurd Skogestad, Ian Postlethwaite, *Multivariable Feedback Control*, John Wiley & Sons, LTd, ISBN 978-0-470-01168-3

[13] Ravensbergen et al. 2019, to be submitted

[14] P.C.Stangeby, *The Plasma Boundary of Magnetic Fusion Devices*, Bristol: IOP Publishing Ltd, 2000

[15] Charles Hirsch, *Numerical Computation of Internal and External Flows, Volume 1 Fundamentals of Computational Fluid Dynamics, second edition*, Oxford: John Wiley & Sons, Ltd, 2007

[16] John Wesson, *Tokamaks second edition, Oxford Science Series 48*, Oxford: Oxford University Press, 1997

[17] N. Horsten, , W. Dekeyser, , G. Samaey, and M. Baelmans, *Comparison of fluid neutral models for one-dimensional plasma edge modeling with a finite volume solution of the Boltzmann equation*, Physics of Plasmas 23, Issue 1, Januari 2016

[18] Naval Research Laboratory, Washington D.C. *NRL Plasma Formulary 2018*

[19] `https://nl.mathworks.com/help/matlab/ref/ode15i.html`, Matlab verion 2018a

[20] `https://nl.mathworks.com/help/matlab/math/choose-an-ode-solver.html`, Matlab verion 2018a

[21] Charles Hirsch, *Numerical Computation of Internal and External Flows, Volume 1 Fundamentals of Computational Fluid Dynamics*, Chichester: John Wiley & Sons, Ltd, 2007

[22] Alan C. Hindmarsh, Radu Serban and Daniel R. Reynolds *User Documentation for cvode v3.1.2 (sundials v3.1.2)*

[23] C.Theiler, et al., *Results from recent detachment experiments in alternative divertor configurations on TCV*, Nuclear fusion 55, march 2017

[24] C.Vuik, P. van Beek, F. Vermolen, J. van Kan, *Numerical Methods for Ordinary Differential Equations*, Delft: VSSD, 2007

[25] `https://nl.mathworks.com/help/matlab/ref/ode15s.html`, Matlab version 2018a

[26] `https://nl.mathworks.com/help/matlab/ref/ode45.html`, Matlab verion 2018a

[27] M. Wensing, et al., *SOLPS-ITER simulations of the TCV divertor upgrade*, Plasma Phys. Control. Fusion