Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Materials Science and Engineering

Aren Lekve Nordhagen

Non-transferred arc hydrogen plasma torch simulations in OpenFOAM

Master's thesis in Materials Science and Engineering
Supervisor: Kristian Etienne Einarsrud
Co-supervisor: Sverre Johnsen and Halvor Dalaker
June 2022

NTNU
Norwegian University of
Science and Technology

Aren Lekve Nordhagen

Non-transferred arc hydrogen plasma torch simulations in OpenFOAM

Master’s thesis in Materials Science and Engineering
Supervisor: Kristian Etienne Einarsrud
Co-supervisor: Sverre Johnsen and Halvor Dalaker
June 2022

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Materials Science and Engineering

Preface

This work is written for the Norwegian University of science and technology(NTNU) as a requirement of the course TMT4905 to complete the master's degree in materials science and technology. The project was conducted during the spring semester of 2022 under the supervision of Professor Kristian Etienne Einarsrud, and with help from the institute of material science at NTNU. Support has also been given from Sintef, which I'm truly grateful for.

I want to thank Kristian Etienne Einarsrud for the supervision and guidance throughout the whole project, which would not have been completed without your help. I'm most thankful for the help with difficult physics, fluid mechanics, and thermodynamics. However, the help with OpenFOAM has been the most crucial, and this work would not have gone anywhere without your explanations of its inner workings. I also want to thank Sverre Johnsen and Halvor Dalaker for feedback on the report and explanations of complex and confusing plasma physics.

Last but not least, I would also like to thank my family and friends for being there for me and listening to my problems, even though you had no interest in thermal plasma physics or OpenFOAM scripting. A special thanks to my classmates who have been with me all semester and cheered me up with lunch breaks and meaningful conversations.

Abstract

This project conducted a literature study and numerical simulations on thermal hydrogen plasma. The main goal was to model hydrogen plasma in a non-transferred plasma torch. The primary motivation behind the simulation is the move from carbon dioxide-based reduction processes of metal oxides to cleaner reduction processes with hydrogen plasma. Plasma reactors are expensive to build, and the inner workings of the torch are not entirely understood as it is challenging to extract experimental data. Hence, modeling it in software such as OpenFOAM can create data that might help understand hydrogen plasma behavior and design better plasma reactors. A case and solver procedure were constructed in OpenFOAM using data and models from the literature study. The case was a steady-state laminar system with an axisymmetric geometry. The axisymmetry was centered on the symmetry axis of the cathode in the non-transferred plasma torch. Governing equations for mass, momentum, and energy were implemented, along with an electromagnetic model. These models enabled the coupling of an electric current with the conservation equations of the plasma through the source terms of Joule heating and Lorentz force. In addition, polynomials of thermophysical properties were implemented to capture the dynamic response of the plasma to the heat from the electric current.

Hydrogen and argon plasma was simulated using the case setup, and a simple parameter study was performed. The argon results were compared to earlier work, which uses the same geometry and parameters. In addition, the results of the hydrogen and argon parameter study were also compared to the literature of parameter studies on hydrogen and argon. The results show that the temperature field resembles the earlier work, but the temperature was lower than expected. The low temperature could be attributed to the polynomial regression, which is not accurate enough to capture the plasma properties. Furthermore, the response to the parameters such as inflow velocity and current resembles litterateur results, but the arc behaves drastically different from what is expected. This result shows that the electromagnetic model and Lorentz force implementation needs more attention. In addition, the hydrogen results are questionable as they did not meet the requirement for convergence.

Sammendrag

Dette prosjektet utførte en litteratur studie og numeriske simuleringsforsøk på termisk hydrogen plasma. Hovedmålet var å modellere hydrogen plasma i en ikke-overført lysbue plasma fakkel. Hovedmotivasjonen bak er å gå fra karbondioksid basert metalloksidreduksjon til renere prosesser som bruker hydrogenplasma. Plasma reaktorer er dyre å bygge, og oppførselen til plasma er dårlig forstått siden det er vanskelig å hente ut eksperimentell data fra plasma fakler. Simulering i OpenFOAM er derfor et perfekt verktøy for å genere data som kan hjelpe til med forståelsen av hvordan plasma oppføre seg, og hvordan en designer bedre reaktorer. En modell og løser prosedyre i OpenFOAM ble bygget ved hjelp av dataene og modellen fra litteraturstudiet. Modellen var et stasjoner, stabilt laminært system med en aksesymmetrisk geometri. Denne aksesymmetrien var sentrert på symmetriaksen til katoden. Styreligninger for masse, moment og energi ble implementert sammen med en modell for elektromagnetiske felter. Disse modellen muliggjorde koblingen mellom den elektriske strømmen og bevaringsligningen til plasmaet via Joule oppvarming og Lorentzkraften. I tillegg ble thermofysiske polymer implementert for gjenskaper den dynamiske responsen plasmaet har til varme fra den elektriske strømmen.

Simuleringen av hydrogen og argon plasma ble gjennomført på modellen og en enkel parameter studie ble gjennomført. Argon resultatene ble sammenlignet med tidligere arbeid på samme modell som i dette prosjektet. Parametertestene på hydrogen og argon ble sammenlignet med tidligere arbeid i feltet. Resultatene viser at temperaturfeltet ligner på tidligere arbeid, men temperaturen var lavere enn forventet. Denne lave temperaturen kan skyldes at polynomene implementert ikke var nøyaktige nok. Parametertestingen viser at noe av responsen til plasmaet er som forventet, men lysbuen oppførte seg ikke slik litteraturen tilsier. Dette resultatet viser at den elektromagnetiske modellen og implementasjonen av Lorentzkraft trenger mer arbeid.

Table of Contents

List of Figures

List of Tables

1 A cleaner process

The world has experienced a rise in greenhouse gases, temperatures, and extreme weather throughout the last decades. Clean and more efficient processes need to replace old ones to combat this trend and cut emissions. One such sector where this shift is necessary is the material sector, where the metal production industry requires new processes and chemical routes to combat their contribution to global emissions[1]. The steel industry alone generates 5% of the total carbon dioxide contribution of the whole world. The release of carbon dioxide from such industries is a consequence of their dependence on carbothermal processes to turn oxides into metals. In such processes, metallic oxides react with carbon to form stable metallic compounds. The carbothermic reaction for iron oxide is listed below:

$$2Fe_2O_3(s) + 3C(s) \implies 2Fe_2(s) + 3CO_2(g) \tag{1}$$

The byproduct of the carbothermic reaction is carbon dioxide. Unfortunately, this byproduct also occurs when producing other metals such as titanium, silicon, manganese. Thus new technologies are necessary to reduce the emission of carbon dioxide, and spearhead the metal industry towards the green shift. An example of new technology which could be used is hydrogen. The reaction between hydrogen species and metallic oxides will not create carbon dioxide as a byproduct. Instead, it produce water vapor. This can be seen from the equation bellow:

$$FeO(s) + H_2(g) \implies Fe(s) + H_2O(g) \tag{2}$$

It has been suggested that the move to this reaction, using hydrogen as the reducing agent for the production of iron, will be advantageous as prior experiments with hydrogen used in metal production[1] reveal a possible cost reduction. In addition, the move to hydrogen might be advantageous as the gas might become cheaper and more readily available as society moves toward a hydrogen-based economy[2].

However, thermodynamic calculations show that the gas state is not adequate if hydrogen gas is used for all processes that use a carbothermic reaction, as it does not have the thermodynamic properties necessary to reduce certain metal oxides. A solution to this problem could be to look at more energetic species as an alternative. The plasma state for example, which is the ionized state of gases is highly energetic and contains reactive excited species[1]. Conveniently the plasma state does have the necessary thermodynamics to reduce most oxides in the following reaction:

$$\text{FeO(s)} + \left(H_2^*/2H/2H^+\right)(g) = Fe(s) + H_2O(g) \tag{3}$$

Nevertheless, the transition possibly comes at a higher price in relation to emissions and the economically aspect. For example the net emission could be higher if the source of hydrogen and the energy source used to produce hydrogen plasma releases more emissions than the charbothermic reaction. In addition, according to Dalaker et al.[3] the profitability of using hydrogen plasma depends on the hydrogen price and cost of producing CO_2 in the coming decades. These prices, and the availability of clean sources for hydrogen and energy are difficult to predict, but the concept of using hydrogen plasma is still worth pursuing.

Trying to carry out a project that uses plasma technologies is expensive and takes time. Therefore, before anything is built, simulations can be used to model the behavior of the plasma, find optimal process parameters or to test if designs and concept ideas are fruitful. In addition, it is hard to extract experimental data from the arc reactor due to the large amount of radiation and high temperatures of plasma reactors[4]. Hence, simulations could give information not possible to obtain by other methods. Computational fluid dynamics(CFD) is the perfect simulation tool for a dense plasma system, especially at atmospheric pressure, which is a more suitable condition for a future up-scaling of a theoretical reactor. Over the recent years, simulation tools have advanced to the point where they can handle complex and extreme physics such as plasma systems. Open-source software is free and accessible and potentially easier and faster to customize than its commercial counterparts. Customizability is essential, as plasma contains physics for which not all software has a standard solution. Therefore, free software such as OpenFOAM is an excellent platform to start a case study.

1.1 Goals and Limitations

This project aims to do a case study on the implementation of modeling procedures that can predict the properties of thermal hydrogen plasma. The software chosen is OpenFOAM, as the source code is open-source, it is free, and has a vast collection of user guides available. The work restricts itself to a thermal plasma generator, suitable for industry and metallic ores, which is chosen to be a non-transferred arc torch. Data will be gathered from adequate sources, prioritizing availability over accuracy. The governing equations, coupled electromagnetic effect, and non-constant thermodynamic properties will be explored and implemented. An exploration of the simulation procedure's accuracy, stability, and efficiency through a study of different discretization schemes and algorithm procedures is not conducted, as the thesis is written from the point of view from a material scientist in the metallurgical industry. The goal is to first explore the literature on thermal plasma and thermal plasma simulation. This will give insight into the physics and behaviour governing thermal plasma, the existing simulation technique paradigm, and available data on properties and parametric behaviour. Next, with the available tools in hand, the work aims to see if it is possible to recreate plasma behavior in OpenFOAM by starting with a compressible gas, and then add the additional elements of thermal plasma. This is then compared to older studies with argon plasma to see if it matches the results and behavior predicted in earlier studies. The resulting procedure is then used in a parametric study on thermal hydrogen and argon plasma to reveal the unique characteristics of hydrogen plasma, and compare it to existing simulation results. Specific assumptions are laminar flow, two-dimensional axisymmetric simulation, the use of polynomials to curve fit thermophysical data, and the use of the tools given to the user by the OpenFOAM software.

2 Plasma

Plasma is a hot gas ionized gas. It is often referred to as new state om matter because of its different attributes and behaviour such as highly temperature and pressure dependent thermodynamic and transport coefficients. The difference originates from the ionization degree of the gas, which is higher than regular gases. This makes plasma electrically conductive, which is in contrast to most gases. However, ionization is not enough to be classified as plasma, as the hot ionized gas need to be electrically neutral as well, which is refereed to in the work of Boulos et al.[5] as quasi-neutrality. In addition, plasma contains different species of the same original molecule, where ionization and molecular dissociation create new constituents with different properties and molecular make up. This changes the property of the gas, as the species relative amount will change how the gas dynamics behaves. The mechanics behind this lies in the different attributes of each molecule, as each has a different mass and electronic structure. Thus the average thermodynamic degrees of freedom of the plasma composition will vary creating different thermodynamic properties. The transport properties varies as well as the collision cross section is different for each specie. Hence a changing chemical composition will alter the behaviour of the gas. Hydrogen plasma gas for example contains H, H^+, H^-, H_2, and ${H_2}^+$, which is the result of different ionization and dissociation reactions in the plasma according to Murphy et al.[6]. The extent of ionization, or ionization degree, is often quite low around 10^{-4} to 10^{-7} according to Fridman[7].

Since the plasma consist of free charges, even though it is net zero as mentioned earlier, it will be affected by electromagnetic forces. Hence, phenomenas like Joule heating and Lorentz force will affect the transport equations of the plasma. In addition it will also interact with external magnetic fields.

The interest towards plasma in material science and chemistry comes from its potential in the synthesis of materials, as well as economic benefits as reasoned by Sabat et al.[1]. Two main attributes stand out, where the first is the existence of highly energetic species which are not created in other less energetic phases. These enable reaction pathways which are only possible for plasma. The second effect is the high energy density and temperature which enhances the kinetics and make the reactions faster and more efficient.

3 Hydrogen plasma in metallurgy

Before trying to model hydrogen plasma, it is important to know if there is any substance to the argument of using hydrogen plasma in metallurgy. Fortunately the usage of hydrogen in metallurgy has already been researched by several researcher. Some research has been done on iron production with use of both hydrogen gas and plasma.

when it comes to reducing iron oxides, the work of Spreitzner et al.[8] shows that hydrogen gas can be used as a reducing agent as its reaction has low enough Gibbs energy and good kinetics. For other oxides this is not true, however in this article from Sabat and Murphy[9] hydrogen plasma is stated as a solution as they presents its reaction mechanics with iron. Their reasoning is their Ellingham diagram, showed in Figure 1 of both oxides and different species of hydrogen.

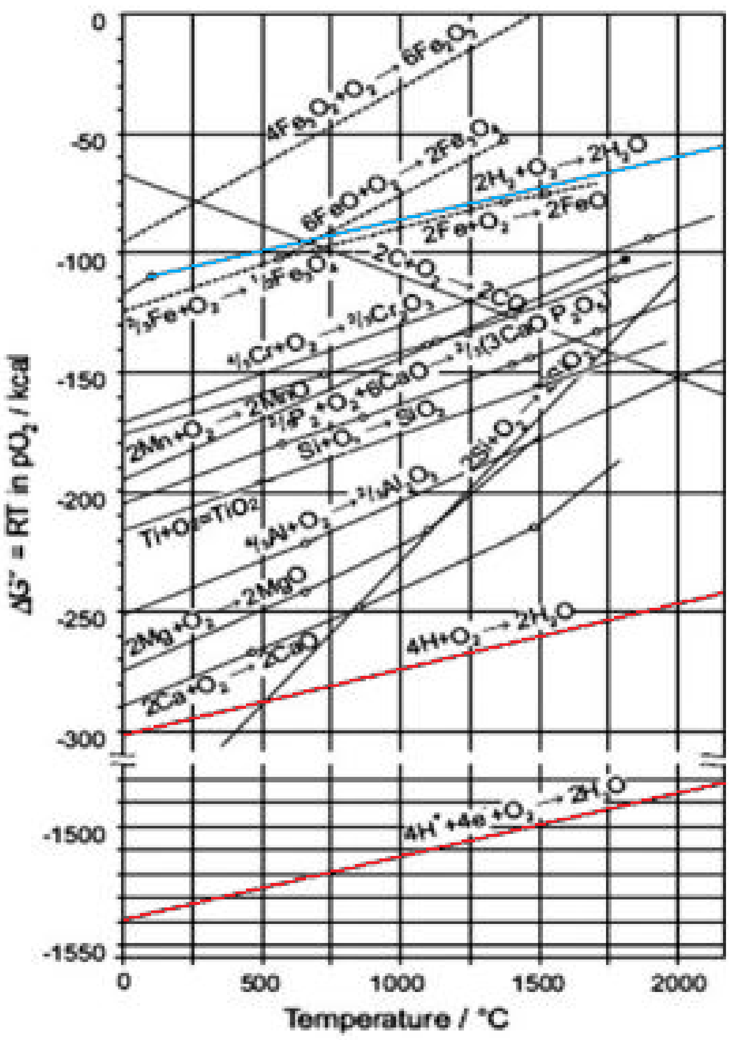

Figure 1: Ellingham diagram with common metal oxides and excited hydrogen species [1].

The Figure 1 shows that both atomic and ionized hydrogen theoretically have the Gibbs energy required to reduce most oxides as the Gibbs energy of reaction of excited hydrogen species with oxygen lie below most oxidation reactions of metal oxides.

In the article from Sabat and Murphy[9] the thermodynamic reactions of hydrogen plasma are quickly summarised, and the reaction mechanics using hydrogen plasma as a reducing agent with iron is briefly shown. Hydrogen gas reacts with iron oxide through the following pathway at atmospheric pressure:

$$3Fe_2O_3(s) + H_2(g) = 2Fe_3O_4(s) + H_2O(g)$$
$$\Delta G_1 = 3.0 - 0.1096T \tag{4}$$

$$Fe_3O_4(s) + H_2(g) = 3Fe_3O(s) + H_2O(g)$$
$$\Delta G_2 = 65.75 - 0.702T \tag{5}$$

$$FeO(s) + H_2(g) = Fe(s) + H_2O(g)$$
$$\Delta G_3 = 18.45 - 0.102T \tag{6}$$

The overall reaction becomes:

$$Fe_2O_3 + 3H_2 = 2Fe + 3H_2O$$
$$\Delta G_4 = 37.57 - 0.0432T \tag{7}$$

This reaction still has a relatively high Gibbs energy which will not be negative for other oxides. However by replacing hydrogen gas by plasma the Gibbs energy drops substantially:

$$H_2 + Energy = (H_2^*/2H/2H^+)$$
$$\Delta G_5 >> 0 \tag{8}$$

$$FeO + (H_2^*/2H/2H^+) = Fe + H_2O$$
$$\Delta G_6 = \Delta G_3 - \Delta G_5 < 0 \tag{9}$$

A lower Gibbs energy is achieved by adding Equation 8 and Equation 6 together, as ΔG_6 has to be below zero. Hence this is an argument for investigating the behaviour of hydrogen thermal plasma in an arc reactor.

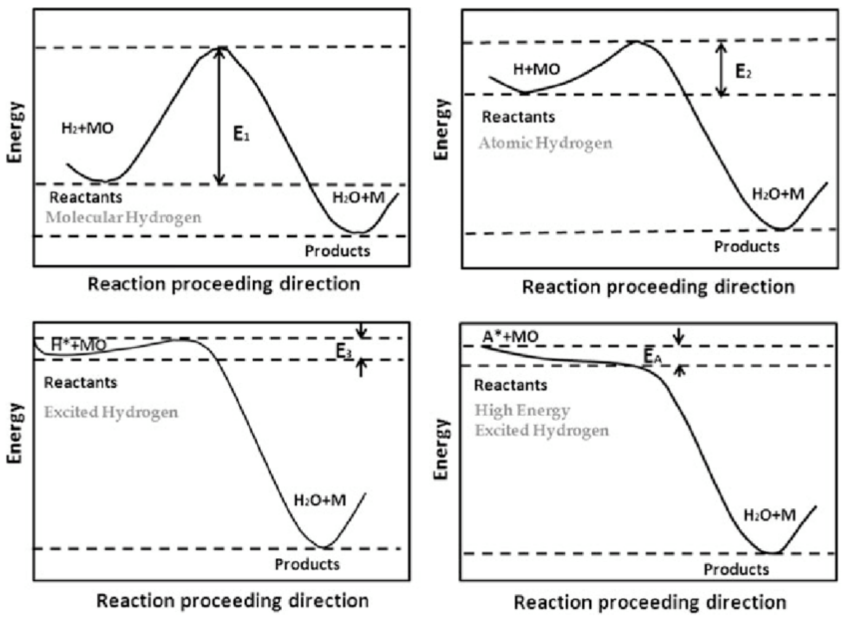

Figure 2: Diagram of activation energy of reduction reactions between hydrogen species and metal oxides[1].

The next argument presented by Sabat and Murphy is the activation energy of the reactions between excited species of hydrogen and oxides. This is the energy needed to initialize a reaction. It should be lower for plasma state compared to regular gas, as demonstrated in Figure 2. This gives more favorable reaction kinetics and a more efficient reaction process if the creation of the plasma is excluded.

4 Thermal plasma sources

There are different methods used to create thermal plasma, all described in this article from Murpy et al.[4], and some are more commonly used than others. Today three different sources exist: ICP, electric arc and microwave plasma. These function in different ways and utilize different systems, but their function is the same, which is to create and maintain gas in plasma state.

ICP stands for inductively coupled plasma which heats the gas using the electromagnetic effect of induction to create Joule heating. This is done with a set up where a coil is wrapped around a cylinder. A potential difference is created in the cylinder by sending an alternating current through the wire around it. This is the phenomena known as Lenz's law. Joule heating is then created as a result of the resistance of the gas and its moving charge with the potential difference. Thus, the plasma is energized. Normally this plasma is a lot colder than plasma from other thermal plasma generators, as the heating process can be maintained at lower energies. As a result, the flow is more often in the laminar regime as well.

Microwave is more specific in its usage, and only used in elemental analysis. It creates plasma by superimposing microwaves in a central ceramic tube, creating an electric field which energizes the plasma. This source is not often used however as creates plasma with higher degree of non-thermal characteristic.

The third option is most often used in industry today. This is the arc option where a large, applied potential between a cathode and anode creates joule heating along a arc in the generator. The arc locally creates both strong heating and exerts Lorentz force on the gas. Two different concepts exist today, one where the arc is established between a cathode and anode, and either the nozzle wall is used as the anode, or the sample at the end. These kind of plasma generators often creates turbulent high plasma. In this work the type of plasma arc where the current travels from the nozzle wall to the cathode is used. This is called a non-transferred plasma torch, and generates electrons at the cathode as shown by the current path in the work of Trelles et al.[10].

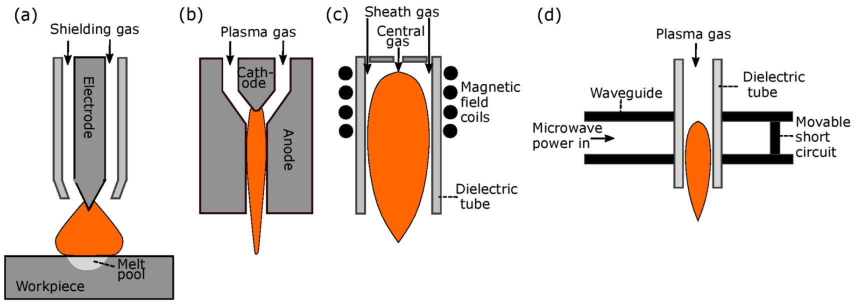

Figure 3: Illustration of different thermal plasma sources from the work of Murphy et al. [4], where a represents (a) transferred arc, (b) a non-transferred, (c) a inductively coupled plasma reactor, and (d) a microwave induced thermal plasma.

Figure 3 shows the chosen plasma torch geometry as model (b). As seen the gas flows perpendicular to the anode surface, however the electric current travels in an arched path from the anode to the cathode, attaching at the anode Wall.

5 Governing equations

To be able to model thermal plasma, a set of governing equations are needed which will predict the dynamic evolution of the state variables of the plasma. An example of such robust and tested equations are the conservation equations of mass and momentum and energy equation used in continuum mechanics. However, to use these equation the system needs to be verified within the limits of a continuum. The verification of plasma as continuum is done through checking the Knudsen number(Kn) as done by Lopes[11] in his thesis. The Knudsen number is defined as the ratio between the mean free path of a system and the physical length of it. In thermal plasma the Knudsen number is low, as argued by Lopes[11] which verifies the continuum approach. This allows the usage of Reynold's transport theorem to model the conservation of extensive properties of the system. The resulting conservation equations from the theorem are the most common model equation for a continuum system and known as continuity equation, the energy conservation equation, and the conservation of momentum. These will be shown in the section on conservation equations.

As mentioned earlier, the plasma is ionized thus consisting of charged particles which have good conductivity, hence can transport a large current. This current is the source of energy for the plasma, which keeps it in the plasma state, however the current and source term is governed by the electromagnetic force. Maxwell's equations, which governs the electromagnetic forces for a continuum is thus need to be solved to achieve a dynamic model between the current and plasma flow.

5.1 Conservation equations

This section will be dedicated to the deduction, explanation, and simplification of important thermal plasma governing equations.

5.1.1 Reynolds transport theorem

The derivation of the conservation equation requires the knowledge of the Reynold's transport theorem, which is described in the work of White et al.[12]. This theorem describes the properties of intensive values in a control volume. The definition of an intensive values (Ψ) is the per unit mass of its extensive variable ($\Psi = \frac{d\psi}{dm}$). This holds for stationary control volume and gives the formula as:

$$\frac{d\psi}{dt} = \frac{d}{dt}\left(\int_{\Omega} \Psi \rho dV\right) + \int_{d\Omega} \Psi \rho \left(\mathbf{U} \cdot \mathbf{n}\right) dA \tag{10}$$

In essence Equation 10 describes the change of ψ with respect to time, and relates it to the flux of it through a volume and its change inside it. This relates conservation from Lagrange to Euler view. Here ρ is density, t is time, surface area is A, $\mathbf{n}$ is the normal vector to the surface of the control volume.

Before moving on to the conservation equations it is important to mention the divergence theorem, which is often used when solving the conservation equations. It allows surface integrals to be written as volume integrals given the relation below:

$$\int_{d\Omega} \mathbf{F} \cdot \mathbf{n} dA = \int_{\Omega} \nabla \cdot \mathbf{F} dV \tag{11}$$

The equation allows all therms in Reynold's theorem to be written as the same volume integral, thus can be simplified to differential form. **F** is defined as any vector field.

5.1.2 Continuity equation

The conservation equation for mass is created by first assuming that mass should be conserved in a system:

$$\frac{D}{Dt}(M) = 0 \tag{12}$$

Here M is the mass of the system. Using mass as the extensive variable in Reynold's theorem Equation 10, and Equation 13, The resulting integral form of the mass conservation equation can be expressed as:

$$\frac{d}{dt}\left(\int_{\Omega} \rho dV\right) + \int_{d\Omega} \rho\left(\mathbf{U} \cdot \mathbf{n}\right) dA = 0 \tag{13}$$

The symbol **U** refers to the velocity in the continuum. Equation 13 is transformed to the continuity equation by Muller[13] by using the Divergence theorem, stated as Equation 11, which leaves the equation in the given differential form:

$$\frac{\partial}{\partial t}(\rho) + \nabla \cdot (\rho \mathbf{U}) = 0 \tag{14}$$

5.1.3 Momentum equation

The derivation of the Momentum equation is done in a similar manner to the mass equation. Momentum is used instead of mass as the extensive variable in Reynold's theorem, Equation 10. Conservation of momentum with respect to time is equivalent to the sum of forces acting on the control volume:

$$\frac{Dm\mathbf{U}}{Dt} = \Sigma F \tag{15}$$

This gives a relationship to work with. The sum of forces acting on the control volume can be categorized into two groups: surface forces and body forces. Surface forces are acting on the surface of the control volume, and are forces like pressure and viscosity. Body forces on the other hand are acting in the volume as for example gravity or Lorentz force. Using Newtons law Equation 15, and Equation 10 on the left side of, Muller[13] creates the Momentum equation in integral form:

$$\frac{d}{dt}\left(\int_{\Omega}\rho\mathbf{U}dV\right)+\int_{d\Omega}\rho\left(\mathbf{U}\mathbf{U}\cdot\mathbf{n}\right)dA=-\int_{d\Omega}\left(p\mathbf{n}\right)dA+\int_{d\Omega}\left(\tau_{ij}\cdot\mathbf{n}\right)dA+\int_{\Omega}\left(\rho\mathbf{f}\right)dV \tag{16}$$

The term $\mathbf{f}$ represents any body force, and the field τ_{ij} is the viscous stress tensor. In addition ρ is defined as the density, and p the pressure. The stress tensor can be expresses as:

$$\tau_{ij}=\mu\left(\nabla\mathbf{U}+\left(\nabla\mathbf{U}\right)^{\top}\right)-\frac{2}{3}\mu\nabla\cdot\mathbf{UI} \tag{17}$$

The simplification requires that the viscosity coefficient represented as μ is isotropic with respect to spatial parameters. This assumption, with the Divergence theorem, stated in Equation 11, creates the differential form of the momentum equation without the source term:

$$\begin{gathered}\frac{\partial\left(\rho\mathbf{U}\right)}{\partial t}+\nabla\cdot\left(\rho\mathbf{U}\mathbf{U}\right)-\mathbf{U}\nabla\cdot\left(\rho\mathbf{U}\right)=\\-\nabla\cdot\left[\mu\left(\nabla\mathbf{U}+\left(\nabla\mathbf{U}\right)^{\top}\right)-\frac{2}{3}\mu\left(\nabla\cdot\mathbf{U}\right)\mathbf{I}\right]-\nabla p\end{gathered} \tag{18}$$

5.1.4 Energy equation

The final conservation equation defines the energy flow of the system. Total energy E is the extensive variable to be conserved, which is the sum of kinetic and potential energy of the system. Using Equation 10, and thermodynamics's first law:

$$\frac{d\left(E\right)}{dt}=W+Q \tag{19}$$

In this equation W is the rate of work on the system, and Q is the heat added to it. The rate of energy added is the work through body forces, pressure forces, and viscous forces on the volume. Most often viscous dissipation energy is neglected. Heat is given to the volume by Fourier's law over the surface of the volume.

Through these assumptions, and Reynolds' theorem Equation 10, Muller[13] creates the integral form of the conservation of total energy.

$$\frac{d}{dt}\left(\int_{\Omega}\rho EdV\right)+\int_{d\Omega}\rho\left(E\mathbf{U}\cdot\mathbf{n}\right)dA=-\int_{d\Omega}\left(p\mathbf{U}\cdot\mathbf{n}\right)dA+\int_{\Omega}\left(\rho\mathbf{f}\cdot\mathbf{U}\right)dV-\int_{d\Omega}\left(q\cdot\mathbf{n}\right)dA \tag{20}$$

The conservation equation for energy in enthalpy form is also stated in the work of Sass-Tisovskaya[14], and given in the differential form without the source term as:

$$\frac{\partial (\rho h)}{\partial t} + \nabla \cdot (\rho \mathbf{U} h) - h\nabla \cdot (\rho \mathbf{U}) - \nabla \cdot (\alpha \nabla h) = \nabla \cdot (\mathbf{U} p) - p \nabla \cdot \mathbf{U} \tag{21}$$

In this representation α represents thermal diffusivity and h is the enthalpy of the plasma.

5.1.5 Equation of state

The systems of equations now consist of three conservation equations which defines the system. However, there are not enough equations to define a solution for the system for a plasma, as compressible flow in the form of thermally expansive flow is assumed for plasma. Equations of state are used to solve this problem, where an equation for each non constant transport or thermodynamic property is needed as stated by Anderson et al.[15]. These state equations have to be defined as functions of already used variables to not introduce new variables into the system of equations. The main state equation for plasma system is the Ideal gas law, with modifications to account for compressible effects. In its simplest form it can be expressed as:

$$pv = nRT \tag{22}$$

Here n is the moles in the system, and R is the gas constant.

5.2 Coupled equations

In the chapter concerning general plasma, Joule heating is mentioned as heating the plasma through the electric field. To incorporate this into the conservation equations, Joule heating is added as a source term in the Energy equation. The equation of the source term is stated as:

$$\dot{Q} = \mathbf{j} \cdot \mathbf{E} \tag{23}$$

E is defined as the electric field strength, and **j** the current density. The source term, as stated by the formulation, is the energy exchanged by the collisions of charged particles as they are accelerated by the electric field. In addition electron enthalpy transport and radiation is also added as a source term, as the plasma is hot enough to radiate a significant amount of energy. This gives the full energy equation with source terms as:

$$
\begin{gathered}
\frac{\partial(\rho h)}{\partial t} + \nabla\cdot(\rho\mathbf{U}h) - h\nabla\cdot(\rho\mathbf{U}) - \nabla\cdot(\alpha\nabla h) = \\
\nabla\cdot(\mathbf{U}p) - p\nabla\cdot\mathbf{U} + \mathbf{j}\cdot\mathbf{E} + 4\pi\epsilon_n + \nabla\cdot\left(\frac{5k_b\mathbf{j}}{2eC_p}h\right)
\end{gathered}
\tag{24}
$$

Here k_b is the Boltzmann constant, e the elementary charge, and C_p the specific heat capacity at constant pressure. The Moment equation also possess such a source term, which is called the Lorentz force, a body centered volume source. It comes form the interaction between moving charges and a magnetic field, which creates a force on the charges.

$$
\mathbf{F} = \mathbf{j}\times\mathbf{B} \tag{25}
$$

As the arc from the plasma reactor will creates it own magnetic field, the charges will interact with it and add a force to the plasma. The coupled momentum equation can be written as:

$$
\begin{gathered}
\frac{\partial(\rho\mathbf{U})}{\partial t} + \nabla\cdot(\rho\mathbf{U}\mathbf{U}) + \nabla\cdot\left[\mu\left(\nabla\mathbf{U} + (\nabla\mathbf{U})^{\top}\right)\right] = \\
\nabla\cdot\left[\frac{2}{3}\mu(\nabla\cdot\mathbf{U})\mathbf{I}\right] - \nabla p + \mathbf{j}\times\mathbf{B}
\end{gathered}
\tag{26}
$$

The coupled terms comes from the fact that the source terms couple together the conservation equations with the electromagnetic, meaning that the flow of the plasma will react according to the physics and conditions of the electromagnetic phenomena. However this means that electromagnetic equations need to be solved to be able to accurately calculate the source terms. Maxwell's equations are the key to understand the electromagnetic phenomena, and are explored in the next chapter.

5.3 Maxwell's equations

From Griffith[16] the Maxwell equations governing the plasma system can be obtained. These are Faraday's law , Ampere's law, Gauss's law for electric and magnetic fields:

$$
\nabla\times\mathbf{E} = -\tfrac{\partial\mathbf{B}}{\partial t} \tag{27}
$$

$$
\nabla\times\mathbf{B} = \mu_0\mathbf{j} + \mu_0\epsilon_0\tfrac{\partial\mathbf{E}}{\partial t} \tag{28}
$$

$$\nabla \cdot \mathbf{E} = \tfrac{\varrho}{\epsilon_0} \tag{29}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{30}$$

the symbol μ_0 is magnetic permeability, and $\mathbf{B}$ is the magnetic field vector. In addition ϱ represesnt the charge density.

5.4 Magneto-hydrodynamic approach

These equations are simplified using assumptions from the magneto-hydrodynamic model(MHD), which are simplifications taken from the nature of thermal plasma systems. Through these simplifications the calculations become easier to implement and solve. These assumptions are the Quasi-steady electromagnetic phenomena and charge neutrality, both present in steady thermal plasma arch torches. These assumptions are thoroughly investigated in this paper from Sass-Tisovskaya[14], however in this section the results from Sass-Tisovskaya[14] are briefly mentioned to justify the MHD approach.

Firstly, electro-neutrality simplifies Gauss's law and conservation of charge to:

$$\nabla \cdot \mathbf{E} = 0 \tag{31}$$

$$\nabla \cdot \mathbf{j} = 0 \tag{32}$$

Assumption number two, the quasi-steady electromagnetic phenomena, implies that the electric and magnetic fields to not vary in time. This changes Faraday's and Ampere's law to:

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{j} \tag{33}$$

$$\nabla \times \mathbf{E} = 0 \tag{34}$$

The full Ohm's law governs the current density for conductive materials:

$$\mathbf{j} = \sigma \left(\mathbf{E} + \mathbf{U} \times \mathbf{B} - \left(\frac{1}{n_e} \right) (\mathbf{j} \times \mathbf{B}) \right) \tag{35}$$

Here **U** represent the velocity field, and σ represents the electrical conductivity of the continuum. The last term is the Hall's current. This current is effectively cancelled out because of the high collisions frequency of charged particles in the plasma, as stated by Sass-Tisovskaya[14]. The induction current, which is the second part of the Equation 35 is also neglected, which is proven by calculating the magnetic Reynolds number the ratio between drift current from the applied electric field, and the induction current from the moving charged medium. In most plasma reactors with low inflow of charged particles the magnetic Reynolds number will be much lower than 1, thus the induction current is negligible compared to the drift current.

The result is a simpler formulation of Ohm's law relating current density to the electric field.

$$\mathbf{j} = \sigma \mathbf{E} \tag{36}$$

Below, the potential formulation of the Maxwell's equation system is introduced.

$$\mathbf{B} = \nabla \times \mathbf{A} \tag{37}$$

$$\mathbf{E} = -\nabla \phi \tag{38}$$

Equation 32, which is the conservation of charge, and ohms law results in a equation for the electric potential which is solvable for correct boundary condition.

$$\nabla \cdot (\sigma \nabla \phi) = 0 \tag{39}$$

A definition of the magnetic vector potential is also possible to construct using Ampere's law Equation 33, which is explicitly dependent on Equation 39.

$$\nabla^2 A = \sigma \mu_0 \nabla \phi \tag{40}$$

The full derivation of the Vector potential formulation can be found in the work of Sass-Tisovskaya [14]

6 Thermal considerations

Equilibrium between the different temperatures of each species is established when the mechanism of heating and redistribution is equal. These mechanisms are Joule heating and inelastic collisions between electrons and heavier species. As the translational temperature is the dominant one, all heavy species can be assumed to have the same temperature regardless of size and other molecular attributes in simple monoatomic plasma. The ratio between these forces is proportional to the ratio between the electric field and the mean free path of the plasma, as this relationship represent how fast the electrons can exchange energy before they are energized. If the ratio is small the system is defined as in thermal equilibrium according to Dijk et al.[17].

However such a system is rarely in complete thermal equilibrium(CTE) where the whole system has the same translational temperature. Instead a truer assumption is local thermal equilibrium(LTE). Here each point in the plasma, according to Fridman[7], has one temperature which defines all processes and species present at the point in the plasma. The kinetic energy of the species is Maxwellian distributed among the species present at the point, hence the average energy is statistically distributed among the species. This is represented as $T_e = T_g$. where T_e is the electron temperature and T_g as the temperature of the gas.

To simplify the model of plasma and later models the use of a local thermodynamic model is necessary. This model however does not only require local thermal equilibrium, but also chemical equilibrium. This require that the reactions, which in monoatomic plasmas are ionization and dissociation, to be faster than any convective processes in the plasma. This ensures that chemical equilibrium is established in a given volume. The result of these assumption is a system where all properties are dependent on the average point or volume temperature and pressure, and equations for the chemical equilibrium can be used to calculate composition according to Lopes[11].

6.1 Thermophysical properties

Thermodynamic properties of the plasma is needed to be able to model it. As LTE models simplifies these to functions of temperature and pressure, they are called thermophysical properties. These are the specific heat capacity, specific enthalpy, and density. As well as these there are the transport coefficients governing diffusion of different processes. These are thermal conductivity, viscosity, diffusion coefficient for species, and electrical conductivity. This is mentioned in the work by Busse et al.[18].

6.2 Composition

To find the thermophysical properties of the plasma, the composition has to be known and calculated. The LTE condition simplifies this process by leaving reactions in equilibrium at one temperature for each species. The reactions present in simple monoatomic plasmas are often the Gulberg-Waage and Saha equation for dissociation and ionization. Thus the coupled equilibrium system of these equations for all species present, as well as a modified state equation, is needed to solve for the composition as discussed by Dijk et al.[17].

The Saha equation governs the equilibrium fraction of an ionization reaction of the form:

$$A^{r+} = A^{(r+1)+} + e^{-} \tag{41}$$

Here a general ionization reaction is displayed, ionizing from the nth ionization level to the nth + 1, which frees an electron. The general equation for the equilibrium depends on multiple factors such as internal partition function, the ionization level, and the electromagnetic system around the species and is presented in work of Murphy et al.[4].

The second reaction equilibrium use the Gulberg-Waage equation, which is the dissociation of molecule consisting of two atoms, as in the dissociation in the hydrogen gas system:

$$AB = A + B \tag{42}$$

The equation for the equilibrium of the fraction is in the same form as the Saha-equation, however it depends on the dissociation energy and is less sensitive to the electromagnetic fields around it.

The last piece of the puzzle to complete the system of equations, and define a solution, is the state equation, in this work given at constant pressure as a function of temperature and the density of each species. It is important to mention that as the system is a gas at high pressures, thus van der waals interactions and compressibility needs to be accounted for by using a virial gas equation as mentioned by Gonzales et al.[19].

6.3 Thermodynamic properties

The composition is now known, hence thermodynamic properties such as density, heat capacity and enthalpy can be calculated for each given temperature. In the paper from Gonzales et al. [19] it is shown that the properties of density and enthalpy is the sum of the contribution of each constituent specie in the plasma at the given temperature. For density the formula reads:

$$\rho = \Sigma n_i m_i \tag{43}$$

Where the density is the sum of the mass of each species and its number density. Enthaply is displayed below as the sum of the product of the number denisty, mass, and enthalpy for each species divided by density:

$$h = \frac{1}{\rho}\Sigma n_i m_i h_i \tag{44}$$

When enthalpy is known, the heat capacity at constant pressure can be found through its relation with enthalpy and temperature:

$$C_p = \frac{dh}{dT} \tag{45}$$

The behaviour of these thermodynamic properties are discussed in the article from Gonzalez et al [19]. As mentioned earlier, these properties are dependent upon the composition of the thermal plasma, which when pressure is constant is a function of temperature only. Its dependency of temperature can be deduced from the compostions, and how the capacity changes with the ionization and dissociation reactions. As the energy available increases with increasing temperature, the different reactions in the LTE plasma will change their equilibrium compositions, giving different dominant species. This will affect the overall attribute of the plasma, pushing it towards the dominant species. This can be seen as clear peaks in the thermodynamic attributes of the plasma.

6.4 Transport coefficients

The second family of thermophysical properties are the transport coefficients. These are more difficult to calculate than thermodynamic values, as their dependency on composition is more complex, as well as a dependency on the kinetic energy present for each species. The complexity stems from the transport coefficients dependency on the collision cross section between each species. The collision cross section is defined as the are formed from the radius between the two species who collide to interact with each other. Thus if one species enters the collision area of another an interaction will take place[20].

In his paper, Gonzales et al.[19] explains a method to approximate the transport coefficients using the Chapman Enskog method. This project will not calculate the transport coefficients, but use sources instead, hence the introduction of the Chapman Enskog is a short overview. The method itself solves the Boltzmann transport equation by an approximation. The approximation requires collision integrals, which are shown in the the work from Gonzales et al.[19], but not explored in this project.

In the work from Murphy et al.[4] and Sabat et all.[9], Simple relations using the collision integral polynomial is presented. These are summarized down below and gives the electrical conductivity σ, thermal conductivity$k$, and viscosityμ:

$$\mu \approx \frac{\sqrt{m_h}T}{\Omega_{hh}} \tag{46}$$

$$\sigma \approx \frac{n_e}{\sqrt{T}n\Omega_{eh}} \tag{47}$$

$$k_h \approx \frac{\sqrt{T}C_p}{\Omega_{hh}\sqrt{m_h}} \tag{48}$$

$$k_e \approx \frac{\sqrt{T}nC_p}{\Omega_{ee}\sqrt{m_h}} \tag{49}$$

Diffusion coefficient for species is not needed when local chemical equilibrium is assumed as the fast equilibrium process will counteract any diffusion process such that species conservation equations and chemical reaction source terms are not needed to fulfill the system. However if local chemical equilibrium is not present, species conservation equations are required as well as diffusion coefficients. A full scale implementation would require $\frac{1}{2}q(q-1)$ diffusion coefficients, however there exists simplification methods as for example those discussed by Murphy[21]. Here for LTE plasma with multiple non-reacting homonuclear species, it is possible to combine the diffusion coefficient for each species into their atomic base gases. This creates four independent diffusion coefficients: a combined ordinary, pressure, thermal and electrical for each gas.

6.5 Radiation

Most theory on radiation in thermal plasmas are summarized in key points in the work from Gonzales et al.[19]. Heat transfer by radiation is important to consider in any hot environment where radiation is generated. This is especially true for hot ionized gases such as thermal plasmas. Radiation account for a percentage of the heat lost in the arc, and in the case of re-absorption it will account for the transfer from the arc to surroundings. The arc is defined as the heated plasma body, and show considerably different properties to the surrounding gas. It can be see n as the coloured are of Figure 3. Re-absorption only happens if the plasma is optically dense, however in most thermal plasma models, and in this work, the plasma is considered relatively thin.

The hot plasma will radiate different frequencies of radiation at different intensity. This creates a spectral intensity field $I_{v,T}$ which gives the intensity of any wavelength of radiation as a function of temperature. As radiation travels through a medium it might be absorbed depending on the absorption coefficient $a_{v,T}$ of the medium. Radiation can also be emitted from the medium related to the emission coefficient $e_{v,T}$ and its local intensity emission field. For a thermal plasma it can be assumed to be a blackbody spectrum $B_{v,T}$. This gives rise to the radiative transfer equation(RTE), which governs the transport of radiation through the medium and is formulated as:

$$\frac{dI_{v,T}}{dr} = e_{v,T}\left(B_{v,T}\right) - a_{v,T}\left(I_{v,T}\right) \tag{50}$$

This equation is difficult to solve, however using LTE and further simplifications, as done in Gleizes et al.[22] and Johnsen et al.[23], it is possible to use a simplified model:

$$\frac{dI_{v,T}}{dr} = K_v\left(B_{v,T} - I_{v,T}\right) \tag{51}$$

The absorption and emission coefficients are simplified by LTE into the same coefficient K_v. Further simplifications can be done as the equation is still computationally heavy to calculate. This simplification is the net emission coefficient(NEC) which is the net divergence of the radiation intensity inside a isothermal sphere. As the temperature is constant, and the calculation only consider the divergence inside this sphere of constant temperature, it is no longer dependent on surrounding cells with different temperatures. Hence it can be pre-calculated and used as a source term. In this project pre-calcululated values of the NEC is used from works of authors who have calculated it given the arc radius and plasma compostion. The arc radius decides the size of the isothermal sphere, where the arc radius is defined as the radius of the plasma arc body.

6.6 Two-temperature models

Because of the strong gradients developed at edges, anodes, or strong convection flow, such as in arch plasma torches, the plasma will diverge from thermal equilibrium at these locations. According to Murphy et al.[4], the LTE condition is no longer valid, which means that the assumption and calculation method employed in the previous sections are not applicable anymore, and the conclusion derived from these methods is inaccurate. Non-LTE models are an area of further research, and models such as the Two-temperature equilibrium are being investigated. Here T_e and T_g are separate temperatures not in thermal equilibrium, which gives rise to two separate energy equations coupled through source terms accounting for elastic collisions. The work of Trelles et al.[24] highlights that the non-LTE characteristics of the plasma increases as the ration of flow to current increases.

7 Non-transferred arc plasma torches

This section will serve as a more in-depth look into what a non-transferred arc plasma torch is, its usage, and its operational behavior. Non-transferred plasma torches have been used in the industry for a long time. Recently, it has found its use in plasma spraying, destruction of waste material, and melting different materials, as mentioned by Murphy et al.[4]. According to Trelles et al.[10] its main advantages are high energy density of around $10^7[\frac{J}{m^3}]$, its high quenching rate and processing rate due to its high velocities. In addition, its ability to create heated plasma with many reactive species might cause it to find its use in chemical reactions requiring energetic species. As a result, the torch might find use in reducing metallic oxides.

From the description of Trelles et al.[10], a non-transferred arc plasma torch is defined as a heating reactor that uses electric power in terms of an electric arc to heat the incoming gas. The system uses a current between electrodes, the cathode and anode, to form this arc. The gas is flushed in through nozzles around the cathode and is heated on its way through the torch. Opposite to a transferred plasma arc, the arc itself is not attached to the workpiece or an anode at the end. However, the walls of the reactor itself are of a conducting material. Therefore, the arc will attach itself to the cylindrical anode wall. The anode will function as a nozzle, thus acting as a constrictor which increases the velocity of the plasma. However, different setups exist, which are non-symmetric. For example, a non-conductive cylinder wall with an anode and cathode attached opposite to each other onto the cylinder wall, with gas flowing between these. This setup is also considered non-transferred as the current is not following the flow of the plasma. The downside of this setup is that the wear of the anode is a lot quicker than in the previous configuration. Furthermore, the non-transferred arc is not an axisymmetric simulation, thus the phenomena must be described in three dimensions to understand the physics acting in the system thoroughly. This also makes the system unpredictable as it is challenging to know where the strike will occur. An illustration is given below to better describe the non-transferred plasma arc torch:

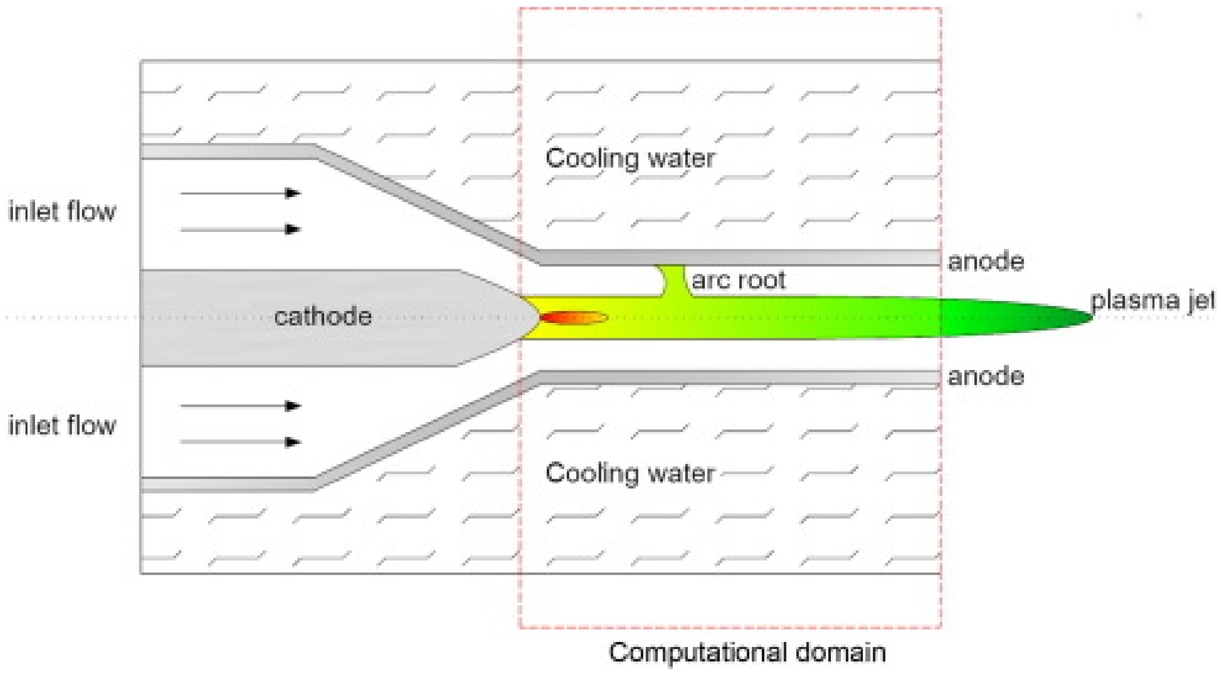

Figure 4: Plasma torch illustrated in the work of Guo et al.[25]

As seen in the figure below, the plasma jet is driven outwards while the current forms an attachment to the anode, which is shown as a small pocket of high temperature at the anode surface. This is often called the arc root location. In this project, however, it is called the anode arch attachment, as the arc attaches at this point. The length along the symmetry line of the torch from the cathode tip to the anode arc attachment is called the arc length. Hence, the curvature of the arc is not considered in its length.

7.1 Behaviour of non-transferred arc plasma torches

Why the arc behaves in such different ways is explained by exploring the effects of the source terms acting on the arc, which are Joule heating and the Lorentz force. Firstly Lorentz force will affect the arc by the pinch effect near the cathode arc, which increases pressure, and contributes to the pumping effect already applied by the Joule heating. The Lorentz force also contributes to the placement of the arc at the anode. The placement or anode arc attachment, according to Paik et al.[26], is dependent on two factors: The drag force pushing in the downstream direction from the plasma flow, and the net Lorentz force pushing it in the upstream direction. The balance between these forces will give the position at the anode for the arc to strike and attach itself. The relationship of this balance is given by Paik et al.[26] as:

$$\nabla \cdot (\rho \mathbf{U}\mathbf{U} + \mathbf{P}) = \mathbf{j}\mathbf{B} \tag{52}$$

One would expect the net contribution of the Lorentz force on the anode arc to be net zero and only exert a pinching force. However, in reality the arc curves, which according to Perambadur et al.[27], give a net force in the opposite direction of the curvature vector. The curvature vector will point in the downstream direction for the top part of the anode arc, as velocity is zero at the no-slip boundary layer close to the wall and increases towards the center axis, and the anode arc current is normal to the anode surface. According to the two authors, Paik et al.[26] and Perambadur et al.[27], a two-dimensional arc will act to the increase of current and flow by decreasing and increasing the arc length. The overall relationship is summarized by Paik et al.[26] by the approximate relationship given in the equation below:

$$x_s \approx \rho^{\frac{5}{4}} \frac{U}{I} \tag{53}$$

Here x_s represents the length from the anode attachment to the cathode surface in the axial direction. Perambadur et al.[27] also show in their results that the movement of the arc increases the outflow temperature. It is important to note that the simulation of Perambadur et al.[27] uses a fixed arc current radius R_c and a free moving anode arc for a given current and flow value, while Paik et al.[26], in contrast, has the anode attachment as a boundary condition. The arc length, which gives the lowest voltage value, is used as the true anode arc attachment.

The trend according to Equation 53 is also valid for the three-dimensional case, according to the article from Guo et al.[25]. However, in this article, the arc radius is varied, while the arc attaches freely to the anode. The arc radius, which gives the lowest heat transferred through the anode, is given as the true arc radius. This is done to create a more realistic simulation as the arc radius is influenced and affects the properties of the non-transferred plasma arc system, hence cannot be given as constant unless experimental values are known for the specific plasma torch. Nevertheless, the results of Guo et al.[25] show that the length of the arc will decrease for the same arc radius if the current is decreased. The work of Ramachandran et al.[28] varies both the arc radius and the anode arc attachment and uses the minimum entropy production to verify the true arc parameters given current and inflow velocity. The results support the trend of a decreasing arc given an increasing current. However, in contrast to the other works, this work shows a decreasing length of anode attachment for increasing flow. The trend is explained by their derived equation for a simplified arc which is also used by Guo et al.[25] and stated here as:

$$x_s = \pi R_c^2 \sigma(T) \frac{U}{I^2} \tag{54}$$

Here Ramachandran et al.[28] explain that the increase of flow will reduce the maximum temperature and thus the electrical conductivity of the plasma. This effect for their case resulted in a decrease in the arc length. This is not seen in the results of Guo et al.[25], however, they note that increasing arc radius for a constant current and inflow will reduce the arc length, as the conductivity falls drastically as the temperature in the arc decreases. This trend was also shown in the work of Westhoff et al.[29] where the effect of multiple parameters of changing current density through altering the arc radius over multiple currents and gas flows with an angled inflow where tested. The results showed that the arc length decreases for increasing arc radius and current but increases for higher flow. In addition, max temperature and velocity in the torch increase with increasing current and decreasing arc radius at a constant current. For a straight inflow, Westhoff et al.[29] also concluded that anode attachment should increase for an increase in flow and decrease for an increase in current. The exit velocity and temperature in the torch increased for a higher current. The increase of the maximum velocity as a function of the applied current is also supported by Paik et al.[26], which pointed to the Maecker effect and the equation for U_{max} in the form of:

$$u_{max} = (\frac{\mu_0 \mathbf{I}\mathbf{j}}{2\pi\rho})^{\frac{1}{2}} \tag{55}$$

Where the maximum velocity is governed by the strength of the electromagnetic source terms affecting the flow through the pinch force and expansion through the increase of temperature. The term $\mathbf{I}$ refers here to the current applied.

It is important to highlight the trends for increasing flow, from the work of Ramachandran et al.[28] and Guo et al.[25], is only for increasing mass flow of argon gas in pure argon or hydrogen argon non-transferred plasma torch system. The works of authors Paik et al.[26] and Ramachandran et al.[28] also highlighted trends for increasing the flow of hydrogen gas. Paik et al.[26] highlighted that hydrogen gas at the same mass flow as argon will create a more constricted anode arc attachment because of the higher thermal conductivity and a lower temperature because of the higher specific heat and shorter anode attachment. In addition, hydrogen shows a shorter arc length at the same mass flow. In the work of Ramachandran et al.[28], the increase of mass flow of hydrogen in an argon-hydrogen mixture was explored and showed an increasing axial temperature and velocity.

In addition, the arc behavior is different in a full three-dimensional simulation compared to a two-dimensional axisymmetric, and the authors of Paik et al.[26] and Perambadur et al.[27] where the only ones to use an axisymmetric geometry. As the arc only attaches to a single spot on the anode in a three-dimensional model, it will cause non-symmetric effects on the temperature and velocity distributions of the torch, as seen in the work of Li et al[30]. This spatial difference will give a higher voltage and longer arc attachment in the two-dimensional case compared to the three-dimensional case, according to Li et al.[30], as the effect of the drag is overestimated for a symmetric attachment. In addition, a three-dimensional arc will, according to the description of Trelles et al.[31] often, if not in steady-state, move along the anode, pushed by the flow, then detach and move to the opposite wall in an upstream position, forming an attachment here, forming a cyclic process. This is the takeover mode, and as described by the paper from trelles et al.[32] the factors causing this behavior are the Lorentz force from the curvature, an anode jet produced at the anode arc attachment from the Lorentz force and Joule heating, and the non-symmetric drag experienced by the arc. These effects create a net momentum moving the arc attachment to the opposite side of the anode nozzle.

7.2 Modeling of plasma torches

Most works aimed at solving steady state non-transferred plasma torch cases use the SIMPLE algorithm as the work of Westhoff et al.[29] for example. To solve the system, different approaches must be taken to get an approximation of the plasma system using CFD software. The base solver algorithm chosen needs to have the capability to solve for thermally expansible flows with heat transfer, which is needed to solve a thermal plasma system. In addition, a method to solve the electromagnetic equations is required in the software. To implement the LTE condition, the transport and thermodynamic properties need to be implemented as functions of temperature, as mentioned in the section on LTE.

Initializing the simulation by some method of recreating plasma conditions or ignition at the start of the simulation is necessary to achieve a steady solution, as most numerical schemes cannot handle the large temperature gradients occurring when applying a large joule heating source term at the start of the simulation, as stated by Murphy et al.[4]. Here the conductivity of the plasma is close to zero, thus resulting in high Joule heating when a constant current density is applied. This means that it's extremely hard to include natural ignition and its gradient in a thermal plasma simulation. This is one of the main issues facing researchers modeling thermal plasma. However, there are known approaches, and two different approaches in OpenFOAM have been explored by earlier case studies such as by Sass-Tisovskaya et al.[14] on transferred thermal plasma systems, and Busse et al.[18] on ICP thermal plasma systems. In the work of Busse et al.[18] a simpler system is solved first, which only solves the Energy equation given in Equation 24. The simulation with only the energy equation is solved without convection and with a restricted conductivity to a minimum of 1 $[\frac{S}{m}]$. In the work of Westhoff et al.[29] a restricted conductivity model is also used, but without a pre-calculated initial field. Sass-Tisovskaya[14], on the other hand, uses a coefficient to slowly ramp up the joule heating source term. This avoids conflicts with large gradients as the whole system is slowly heated up. The coefficient will approach one after a certain amount of iterations. A third method is used by Perambadur et al.[27], where a zone of high conductivity between the cathode and anode is established and removed after a couple of iterations.

The last consideration is the electrodes boundary conditions, as a current profile must be applied, and even though the initialization of the plasma is executed successfully, the simulation might still be unstable because of high permanent gradients at the boundary.

Since all modeling of thermal plasma requires boundary conditions, the arc behavior is also decided by the given boundary condition value at the cathode and anode. For example, the distribution of current along the cathode gives a fixed arc radius. The works referred to in this section use different radial dependent profiles for their current distributions, hence there are no standards. To apply the current, authors like Perambadur et al.[27], who use the vector potential model Equation 39, apply a gradient of the form -$\frac{j}{\sigma}$ at the cathode boundary. At the anode, a zero potential is given as the reference electrical potential needed to solve the electromagnetic equations. In addition, for two-dimensional simulations, some authors like Paik et al.[26] use only a small zone of zero potential at the anode, while the rest is left as zero gradient. This forces the arc to attach to only one spot at the anode. For temperature conditions, authors such as Ramachandran et al.citeramachandran2006modelling use a temperature gradient representing a water cooled wall, while Westhoff et al.[29] prescribes a fixed temperature. Lastly, authors like Guo et al.[25] and Trelles et al.[32] uses a thin high conductive layer in front of their anodes to get a stable anode attachment in three-dimensional cases, and mimic the high conductive non-LTE layer presents close to the electrodes.

The final piece needed to model non-transferred thermal plasma is the procedural algorithm. As mentioned at the start of the section, most authors used CFD and the SIMPLE algorithm for steady-state simulations. The scripts used to create the simulations are rarely included in articles, except for the user-defined script of Westermoen et al.[33] in Fluent, the electromagnetic ICP model of Busse et al.[18], the pseudocode of the altered SIMPLE algorithm in the work of Busse et al.[18], and the procedural implementation of Sass-Tisovskaya[14]. A semi-transient and steady procedure is explained in the bullet point list below created from the pseudocode of the authors Busse et al.[18] and Sass-Tisovskaya[14]

1. First create polynomials or maps of data of thermophysical and transport properties as function of an expected temperature range.
2. Start the solving loop with initial guess of temperature, velocity and pressure.
3. Calculate the electrical conductivity.
4. Solve the electromagnetic vector and scalar potentials equations.
5. Calculate the coupling source terms Lorentz and Joule heating used in Equation 26 and 24.
6. Solve the Momentum equation given in Equation 26 using initial vlaues to get the guessed velocity field.
7. Compute new radiation source term through the NEC model.
8. Solve the energy equation to get new temperature and thermophysical properties.

9. Use a compressible pressure equation to calculate new pressure.
10. Update the velocity guess.
11. Repeat from step (2) if the system has converged, if not repeat from step (5).

8 OpenFOAM

OpenFOAM is the software used to construct the specified geometry and solve the governing equations given as Equation 13, and Equation 16 and electromagnetic equations, Equation 40 and Equation 39. It is an open-source, FVM-driven simulation software which is easily customized. This feature allows the algorithm to be altered such that plasma properties can be implemented. OpenFOAM's user interface is a directory system of input text files and programs, where users alter the input files the software reads. As a result, there isn't a graphical user interface. OpenFOAM is a CFD software, where CFD is the standard method of solving the complex differential equation system, and can be applied to the governs the dynamics of a non-transferred plasma torch system. This chapter briefly explains the ideas behind computational fluid dynamics, and the difficulties faced when using such a tool, and the structure of OpenFOAM.

8.1 The finite volume method

The finite volume method (FVM) is the most commonly used numerical method in CFD analysis and is employed in this project and OpenFOAM, whered the software organizes the geometry into a meshed network. Surfaces in the grid are created by connecting vertexes with edges, enclosing the cell structures. The average value of that region in space, such as specific enthalpy, density, velocity, or any other parameter defining the system, is stored in cells as the cell average value. First, the governing equations are shifted from their volume integral form to surface integrals through the Divergence equation given as Equation 11, as done in the work of Muller[13]. The cell average values and the transport coefficients are then interpolated to the shared faces by two cells according to the discretization method and interpolation method used. This results in sets of equations for each face which are coupled together. The sets of equations are solved through an iterative matrix solution procedure to achieve an approximate solution, as explained in the work of Fang[34]

8.1.1 SIMPLE

According to Anderson et al.[15] the SIMPLE algorithm stands for semi implicit method used on pressure linked equations(SIMPLE). It is a decoupled approach to solving the

Navier-Stokes equations, and is used by steady state simulations in OpenFOAM. In this section the SIMPLE procedure of OpenFOAM is shown, and quickly explained to grasp the outlines of how it operates. The script used in OpenFOAM is listed below:

```

while (simple.loop(runTime))
    {
        #include "UEqn.H"  // Pressure-velocity SIMPLE corrector
        #include "EEqn.H"
        #include "pEqn.H"
        turbulence->correct();
        thermophysicalTransport->correct();
    }
```

Listing 1: Snippet from rhoSimpleFoam.C/ The SIMPLE procedure is run bellow.

The procedure is presented in the work of Fang[34], and the most important steps are quickly highlighted. To run the SIMPLE algorithm uses guessed initial values from the user. The file *UEqn.H* constructs the momentum equation, in the form of Equation 26, and solves it with the given guessed values of thermodynamic and transport properties, as well as pressure and velocity. It then updates using a relaxation procedure:

$$U_{n+1} = U_n + \alpha_p * (U_n^{'} - U_n) \tag{56}$$

This is referred to as the momentum prediction step, and creates a new guessed velocity. Equation 56 is equal for enthalpy and pressure, where the new value U_{n+1} is equal to a weighted sum of the old iteration value U_n and the new guessed value shown by $U^{'}$. The weight is decided by the relaxation factor α_p, which decides how much of the old or new value to use. The effect of this is better stability when a low α_p is applied, as the solution procedure changes slower with each iteration. However, this makes a convergence solution require more iteration steps. The velocity is then used in the energy equation in file *EEqn.H*, with the previous pressure, to get the new temperature value, which is used to update the thermodynamic properties of the system.

The continuity equation, given as Equaiton 31 and Momentum equation, stated as Equation 18 is then used to create a pressure equation of the form, found in file *pEqn.H* , which gives a new pressure value $p^{'}$. Pressure is then updated by relaxation as in Equation 56 by the step known as pressure correction. The velocity is then updated by Equation 56 which gives the new velocity. Turbulence is then handled if added to the simulation, and the transport coefficients are then updated. The loop is repeated until the tolerance of the velocity, enthalpy and pressure fields are satisfied. In addition, the continuity criterion should be satisfied. This criterion is based on the usage of the Continuity equation, given as Equation 31. The criterion most hold true, as the the pressure equation is created using Equation 31, which is shown in the work of Anderson et al.[15].

8.1.2 Discretization

The discretization of each term in the governing equation can be done in a variety of ways, all explained in the work of Greenshields[35], where OpenFOAM discretizes each vector operation term, then adds it to the matrix to be solved. The discretization procedure used for the volume integral convection terms are the Divergence theorem in Equation 11

$$\int_{d\Omega} (\rho\mathbf{U})\,\psi \cdot \mathbf{n}dA = \Sigma_f \mathbf{n}A_f \cdot (\rho\mathbf{U})_f\,\psi_f \tag{57}$$

Where f represents the the face in question between a given cell center N and P. The average cell value ψ is interpolated to the face using various methods such as Central differencing or Upwind differencing which is based on cell center distance to the face, and the direction of the flux at the face. The same procedure with the use of Equation 11 is also used for laplacian terms, which gives:

$$\int_{d\Omega} \Gamma\,(\nabla\psi) \cdot \mathbf{n}dA = \Sigma_f \Gamma\mathbf{n}A_f \cdot (\nabla\psi)_f \tag{58}$$

Here Γ is a coefficient within the cell, as for example the diffusion coefficient or viscosity. ψ is any conserved property. It is interpolated from the cell center to the face. The face gradient of ψ is calculated using the the distance between N and P and the values ψ_N and ψ_P in a first order approximation. However, OpenFOAM uses the divergence theorem, and linear interpolation for the grad term.

8.1.3 Solver applications and scripting

The calculations of the system of equations are done by OpenFOAM using a solver application. It takes data from the input files described in the previous sections and uses it to create a functioning simulation by combining time, mesh, and discretization commands. OpenFOAM solvers use built-in functions and objects and follow an object-oriented framework. Editing solvers is simple, thanks to their intuitive layout. It provides the user with a powerful toolbox for solving novel physics and systems using existing physics. The solver can be compiled with *wmake* once all of the necessary code and physics have been added. Most solvers have a C-file that contains the primary solution procedure and H-files that define constants and mesh variables or subroutines.

The fvm or fvc namespaces contain functions that construct a discretization of the input fields and add it to the appropriate component of the coupled matrix system, making them essential syntax for solving differential equations. The fvm namespace functions add implicit terms to the matrix, adding them as coefficients. fvc functions, on the other hand, add the term to the constant side as explained by Fang[34].

Functions like $grad()$, $div()$, and $curl()$ in the namespaces fvm and fvc create discretization of each word based on the scheme selected by the user. Operators like $+$, $-$, and $==$ are used to specify the separate mathematical pieces of the equation and express the equality. Using $solve()$, the solution to the matrix is calculated.

8.2 Folder structure

The folder structure is separated into three subgroups: time directories, constant folder, and system folder, each of which has a specific function that will be described in the following chapters , shown in the Figure 5 below:

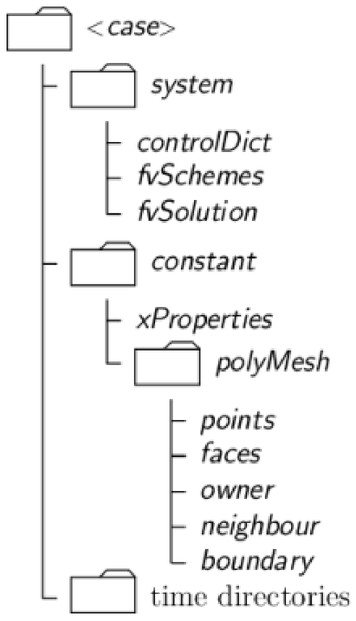

Figure 5: The folder structure of OpenFOAM with each folder and attached files shown in the OpenFoam Userguide[36].

8.2.1 Time folder

Initial values are the other requirement that must be met to solve the system. The guess of the user defines the initial state of the system. Time folders store these values and the BC, which are given for each time step. The 0 folder is the initial folder, which defines the simulation's initial values. The following folders will be named by adding the time increment and used as the SIMPLE algorithm's next guess. This flexible structure allows the simulation to commence at any point in its prior iteration history.

8.2.2 Constant Folder

The *constant* folder holds the properties that remain constant during the simulation, such as thermophysical properties, *polymesh* data, transport properties, chemical composition, turbulence model, and mixture models. All information on these folders are prescribed in the user guide[37].

8.2.3 System

The system folder contains the *blockMesh* file, numerical schemes, time control, and algorithm control files. The *blockMesh* file is responsible for creating the mesh used in the simulation. According to the OpenFOAM user guide[38], a mesh in OpenFOAM is formed as a polymesh, which implies that the mesh is made up of a large number of polyhedral cells. Squares, rectangles, wedges, and other forms are possible geometries. The model geometry is defined by adding these cells together. The technique begins with the formation of points in space, which are subsequently grouped into a geometric shape. A group of points in the geometry is used to identify boundary faces. The selected group form one of the geometry's outside faces. The last part of the *blockMesh* file assigns BC characteristics to patches through the use of keywords explained in the OpenFOAM User guide[39].

The file *controlDict* manages the input and output data streams from the time folders. Files like *fvSchemes* control the discretization technique for variables in the governing equations, while the tolerances and relaxation factors are controlled by *fvSolution*.

9 Case setup

This section presents the case setup for geometry and physics. This project has chosen to base the cases set up on the work of Westhoff et al.[29]. In the work of Westhoff et al.[29] a non-transferred arc torch under steady conditions and laminar flow was simulated, hence it fits the assumptions of this work. Thus both the boundary conditions and the geometry of Westhoff et al.[29] is used in this work, forming the basis of all further simulations.

9.1 The model

It is important to understand the physics being implemented to grasp and understand the implementation procedure developed here and the later results. In this case, the goal is to implement a non-transferred plasma torch by setting a flow through a cylinder around the cathode and a current from the anode to the cathode. The current is the heat source providing the Joule heating, Lorentz force, and other source terms to the plasma model. The physics and differential equations have been explained in section 5. The models assume that the fields in the flow and electromagnetic fields are axisymmetric. The flow is also considered laminar, which is reasonable according to the work of Westhoff et al.[29]. The plasma system is assumed to be steady state. The system is also assumed to be in LTE. The plasma is assumed optically thin, thus radiation loss can be accounted for using the NEC model. Heating caused by viscous dissipation, buoyancy, and compressible effects is also neglected in this project.

The solution procedure is adapted from the discussion on steady state simulation in section 7.2 using the SIMPLE algorithm designed for thermal plasma. Source terms are added to the governing equation according to OpenFOAM protocol. Radiation is present through the NEC model, and LTE by implementing temperature dependent polynomials. Custom Boundary condition values have also been created to implement the correct physics from the base case of Westhoff et al.[29]. In addition, a custom initialization procedure has been implemented according to the methods and problems discussed by sass-tisovskaya[14] and Busse et al.[18] and problems faced in this work. It will be presented in the later solver section.

The system will have a flow that expands as it passes above the cathode and is heated by the Joule effect acting in front of the cathode. The arc in this system is not attached to any specific spot on the anode by any boundary condition but is allowed to distribute itself freely at the surface. Since it is a plasma arc system, the system should be the hottest closest to the cathode. Figure 4 is an example of how the system should behave and look. However, in this work, the cathode is flat.

9.2 Geometry

It is important to note that the geometry is axisymmetrical around the line A - I, which is the Z axis in the cylindrical coordinates (Z, R, θ), and x in the OpenFOAM coordinates x y z. The axis R is the radial direction always orthogonal to Z, while θ is in the rotational direction around Z. In OpenFOAM the axis are give in Cartesian coordinates. Hence to achieve axisymmetry, a wedge is created, and wedge conditions are given. The wedge is made by first creating the geometry in two dimensions in the y and x plane using the $blockMesh$ utility. The points along the symmetry axis keep a z coordinate value of zero. However, all points not situated along the symmetry axis are duplicated into two points and given a positive and negative z value according to the formula:

$$\pm z = \pm tan\left(0.5 \cdot \frac{\pi}{72}\right) \cdot y \tag{59}$$

Here $\frac{\pi}{72}$ is 2.5 degrees, which is small enough for the axi-symmetry condition to work on the wedge. The wedge surfaces with their normals pointing in rotational direction θ are given the wedge condition, which is a periodic boundary condition, hence establishing an axi-symmetrical simulation shown in Figure 6.

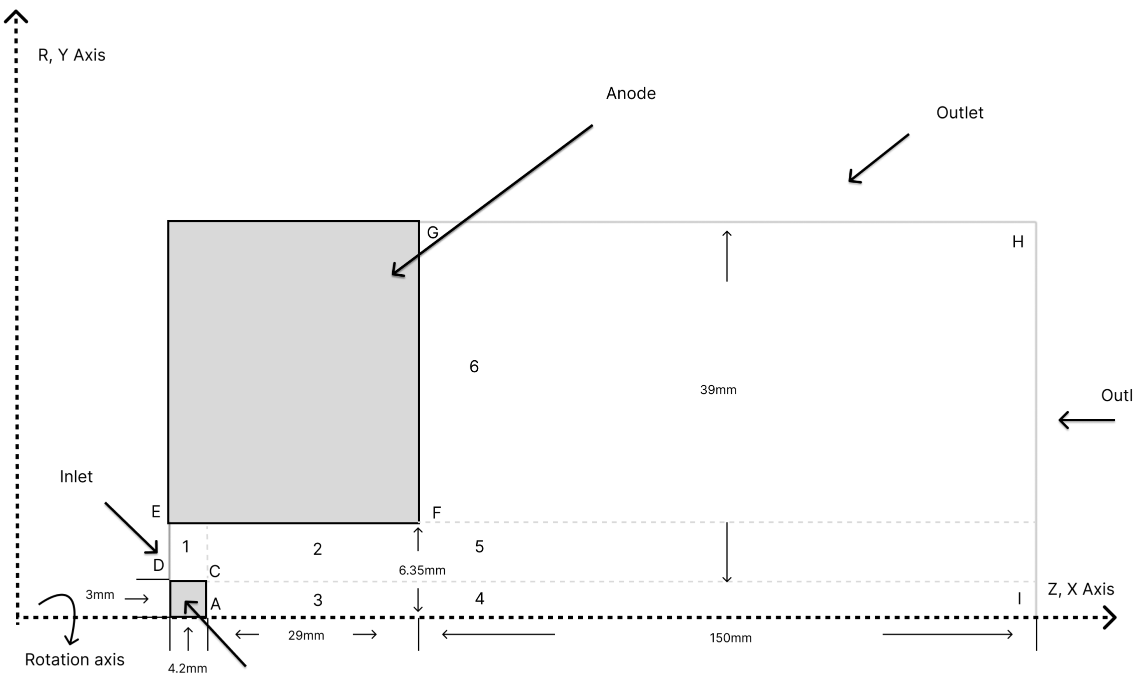

Figure 6: The geometry is adapted from Westhoff et al.[29], and for all cases in this work.

The geometry resembles a non-transferred thermal plasma torch with a central cathode and a conical tube as the anode. The geometry also includes the outside region of the torch, where the torch exits into a larger area representing the area outside the torch.

The lines $G - H$ and $H - I$ mark the outflow boundaries into a pure atmospheric argon atmosphere. $E - F$ marks the anode surface and $F - G$ the front surface of the torch. The line $E - D$ marks the gas inflow into the torch and is situated between the anode and cathode. The cathode has two surfaces, $D - E$ and $C - E$, where the first surface is the top of the cathode, and the latter is the current carrying surface that creates the arc. Here the current will flow by establishing an electric potential.

9.3 Initial Values

Initial values are the values used to initialize the procedure in Figure 22. These are used for the first iteration as guesses for the matrix calculations used at the start of the iterative solution procedure.

Table 1: Initial values for the ThermoFOAMRamp solver

Initial values						
Fields	A $[NA^{-1}]$	$\phi[V]$	$\sigma[Sm^{-1}]$	$p[Pa]$	T $[K]$	U $[\frac{m}{s}]$
Value	0, 0, 0	0	1200	101325	1000	(0, 0, 0)

The velocity of 0 $\frac{m}{s}$ is chosen as ThermoFoamRamp don't use a velocity field.

Table 2: Initial values for the RhoFoamProcedureInitial solver

Initial values						
Fields	A $[NA^{-1}]$	$\phi[V]$	$\sigma[Sm^{-1}]$	$p[Pa]$	T $[K]$	U $[\frac{m}{s}]$
Value	0, 0, 0	0	$\sigma(T)$	101325	1000	(5, 0, 0)

Conductivity is given as a temperature function in the compressible solver's initial conditions as seen in Table 2. As mentioned in the later solution procedure, this field is the final conductivity field from the solver only solving the Energy equation. The initial values from Westhoff et al.[29] are not mentioned in the article hence this work does not have access to them, and the values chosen here are used as they achieved a convergent solution procedure. They are all chosen from a trial and error method, which gave the conditions most steady for the solution of the plasma with the coupled source terms. The pressure is kept at 101325 $[Pa]$ for the start of the simulation.

In the article from Westhoff et al.[29] a source term at the cathode is included to account for ionization of the gas close to the cathode. This source is connected to the discussion on non-thermal plasma and the non-thermal effects close to the electrodes. It is not included in this work as it led to divergence.

9.4 Boundary conditions

The boundary conditions define the system's behavior and its steady solution. All values are taken from Westhoff et al.[29] to the extent information on the conditions can be retrieved from the article. Below each boundary condition is summarized in the table for each boundary. The most important BC conditions, and their values, will be discussed further down in this section.

Table 3: Boundary conditions for all simulations with boundaries from Figure 6.

Boundary conditions						
Boundaries	A $[NA^{-1}]$	$\phi[V]$	$\sigma[Sm^{-1}]$	$p[Pa]$	T $[K]$	U $[\frac{m}{s}]$
ED	$\frac{\partial A}{\partial n}=0$	$\frac{\partial \phi}{\partial n}=0$	σ(T)	$\frac{\partial p}{\partial n}=0$	1000	U(r)
CA	$\frac{\partial A}{\partial n}=0$	$\frac{\partial \phi}{\partial n}$(r)	σ(T)	$\frac{\partial p}{\partial n}=0$	3000	(0, 0, 0)
EF	$\frac{\partial A}{\partial n}=0$	0	σ(T)	$\frac{\partial p}{\partial n}=0$	1000	(0, 0, 0)
FG	$\frac{\partial A}{\partial n}=0$	$\frac{\partial \phi}{\partial n}=0$	σ(T)	$\frac{\partial p}{\partial n}=0$	600	(0, 0, 0)
GH	0, 0, 0	$\frac{\partial \phi}{\partial n}=0$	σ(T)	101325	600	(0, 0, 0)
HI	0, 0, 0	$\frac{\partial \phi}{\partial n}=0$	σ(T)	101325	$\frac{\partial T}{\partial n}=0$	$\frac{\partial U}{\partial n}=0$

The boundary conditions are implemented to replicate the physical model stated in section 7.1. The magnetic vector potential A is set as zero gradient at all boundaries, except those far from the magnetic field. This choice, according to Sass-Tisovskaya[14], gives the best solution stability. On most surfaces, the electric potential is set as zero gradients, using the *zeroGradient* boundary condition. This stops any electric current from moving through the boundary. A zero potential is given to the anode $E-F$, which creates the reference potential for the electric potential. The cathode boundary condition sets the current density distribution at the cathode by defining an electric potential gradient. The function describing the distribution is given as:

$$\frac{\partial \phi}{\partial n} = \frac{j_c}{\sigma} = \frac{I}{\pi R_c^2 \sigma}, R < R_c \tag{60}$$

$$\frac{\partial \phi}{\partial n} = 0, R > R_c \tag{61}$$

The value R_c is the arc radius at the cathode boundary surface. This is applied to the cathode surface $C-A$. The rest of the surface is given a zero gradient according to the solution procedure. Here j_c is the current density, and I is the current. The benchmark article gives the value of the electrical current density as $3 \cdot 10^7$, and the current as 250 $[A]$. This gives a R_c value of 1.6 $[mm]$, which used for the verification case. The current distribution is given in the figure below:

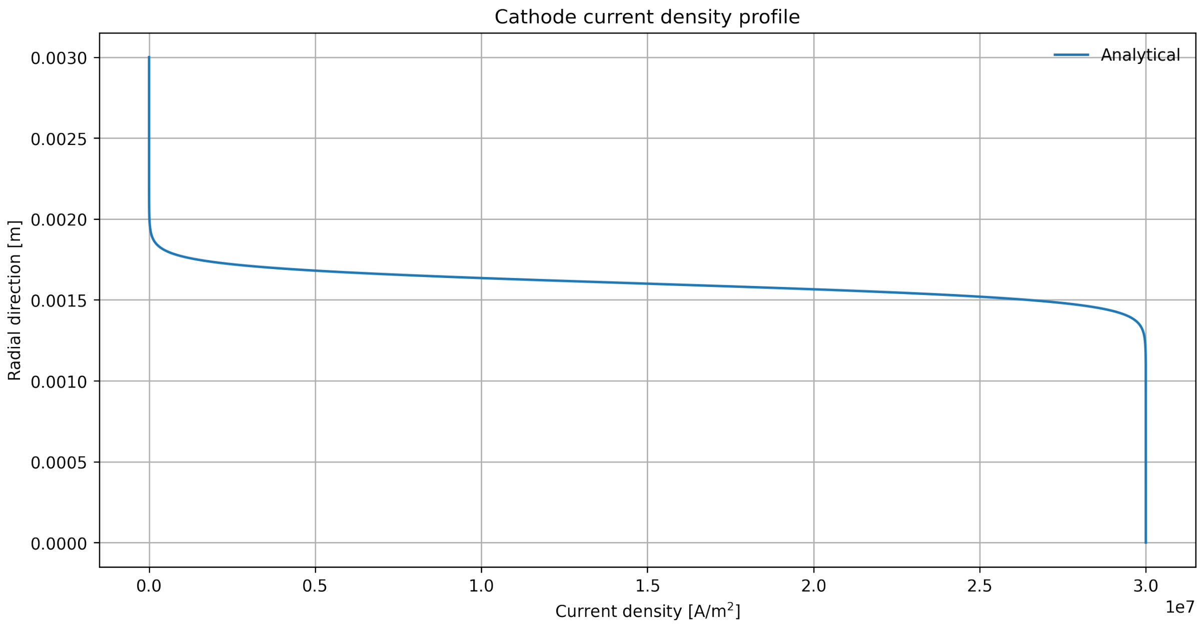

Figure 7: The analytical current density profile

The conductivity is set according to temperature using a polynomial connecting conductivity to temperature. Pressure and velocity are set with $fixedValue$ velocity at the inflow and $fixedValue$ pressure at the outflow. The pressure at the inlet $E-D$ and walls are set to a zero gradient via $zeroGradient$. The velocity is given a parabolic profile, the walls the no slip condition via $noSlip$, and outflow is given the $InletOutlet$ conditions to prevent outflow. These boundary conditions are given by the web pages of OpenFOAM[40]. The outlet is given the fixed pressure of 101325 $[Pa]$ and a $zeroGradient$ for the temperature at the outlet. The boundary $G-H$ is treated differently in this project compared to the work from Westhoff et al.[29]. In their work, this wall is treated as no slip for the axial flow in the x-direction, but given a zero flux gradient in the y-direction. This project has chosen to treat it as an outlet With the $InletOutlet$ condition. In addition, the temperature at boundaries $F-G$ and $G-H$ are set to 600 $[K]$, which is a 100 $[K]$ above the value of 500 $[K]$ in the work of Westhoff et al.[29]. This is because the enthalpy solution procedure is unstable at temperatures below 600 $[K]$. In addition, the pressure boundary conditions are not given in the benchmark work. Hence the conditions in this report are taken from the work of Sass-Tisovskaya[14]. Lastly, the conductivity boundary condition treatment is not stated in the article from Westhoff et al.[29]. Hence, this work will explore different mechanics for testing the boundary by assuming that the boundary value of the electrical conductivity field is the same as the internal cell value next to it.

The inflow condition is stated as parabolic with an inflow of 0.59 $[scmh]$. OpenFOAM uses $\frac{m}{s}$, hence U_{avg} is calculated from the benchmark case value. The unit scmh stands for standard cubic meters per hour and is the mass flux of the inlet divided by the density of the argon at standard conditions[41], which gives:

$$Q_{inlet} \cdot \rho_{inlet} = Q_{std} \cdot \rho_{std} \tag{62}$$

Hence the volumetric flow is found by multiplying by the density of argon at standard conditions and dividing by density at the boundary of 1000 $[K]$. This gives a value of 1.958 $[\frac{m^3}{s}]$ for the volumetric flow. The average velocity U_{avg} is then found trough dividing the volumetric flow by the inlet area, which is the area of the inlet radius to the center axis minus the cathode area. From these short calculations the U_{avg} is 5.556 $[\frac{m}{s}]$. As the flow profile is not axisymmetric around the center axis but instead a full parabolic flow profile, it can be approximated as the two-dimensional flow profile between two fixed plates. From White[12] this is given as:

$$U(r) = U_{max}\left(1 - \frac{r^2}{R^2}\right) \tag{63}$$

As U_{avg} and velocity profile are known, except for U_{max}, it is possible to calculate U_{max} by integrating the velocity profile over the inlet and dividing it by the inlet area. This gives an U_{max} of 8.541 $[\frac{m}{s}]$. This is implemented by the groovyBc library[42] and shown in the appendix.

9.5 Constants

This section displays the constants used by the OpenFOAM, both temperature dependent polynomials and other constants used. The input constants are shown below in table ??.

Constant	Value
μ_0 $[\frac{N}{A^2}]$	$1.26 \cdot 10^{-6}$
k_b $[\frac{m^2 kg}{s^2 K}]$	$1.3806452 \cdot 10{-}23$
e_c $[C]$	1.60217663410^{-19}

Here the values for the elementary charge, Boltzmann constant, and the magnetic permeability is given. These are constant throughout the simulation.

9.6 Polynomials

Next, the polynomials for each transport coefficient and thermodynamic property are shown. An extension to 28000 $[K]$, a minimum value of the lowest value of the data set, is used. A seventh degree polynomial was used for the polynomial regression. This procedure will be explained more in depth in the solver procedure. The Numpy package[43] is used for the polynomial regression. The results are the following polynomials with their percentage errors for argon:

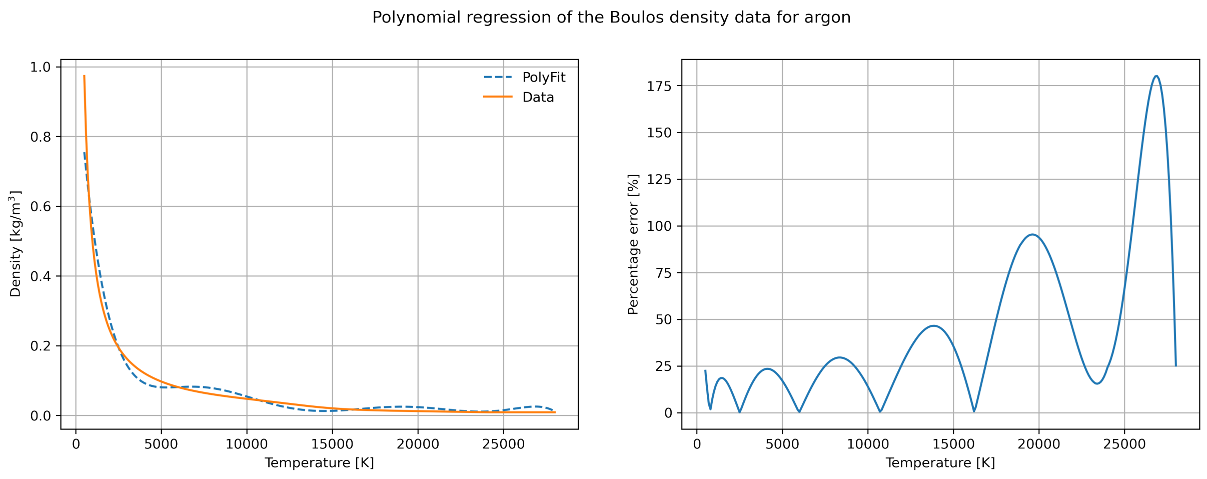

Figure 8: A temperature dependent density polynomial made using the polyfit function of the Numpy package.

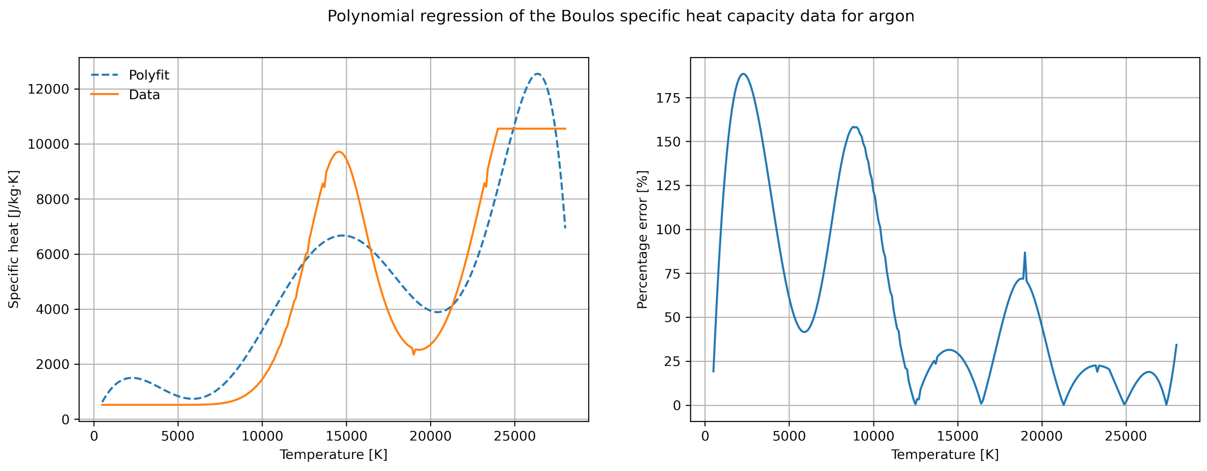

Figure 9: A temperature dependent heat capacity polynomial made using the polyfit function of the Numpy package.

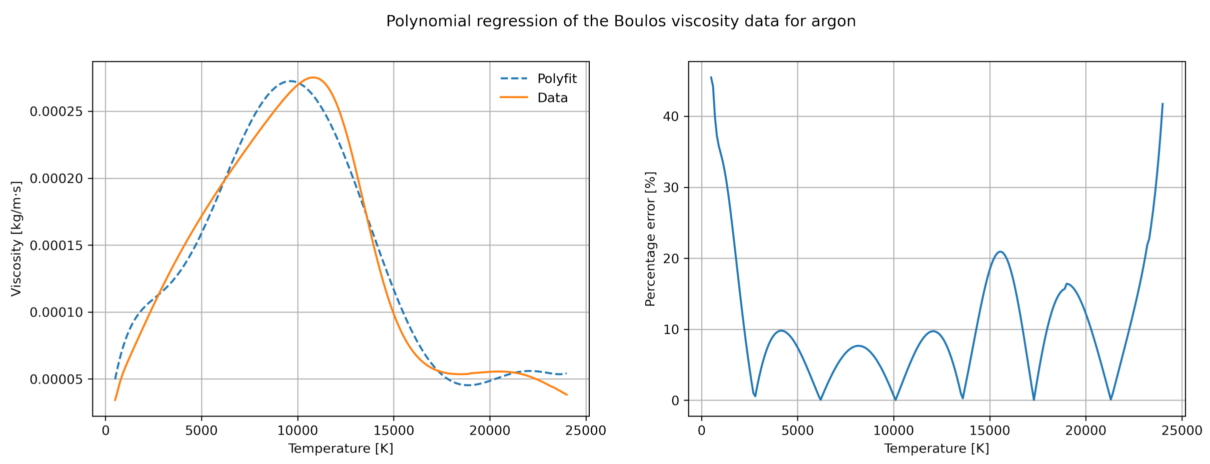

Figure 10: A temperature dependent viscosity polynomial made using the polyfit function of the Numpy package.

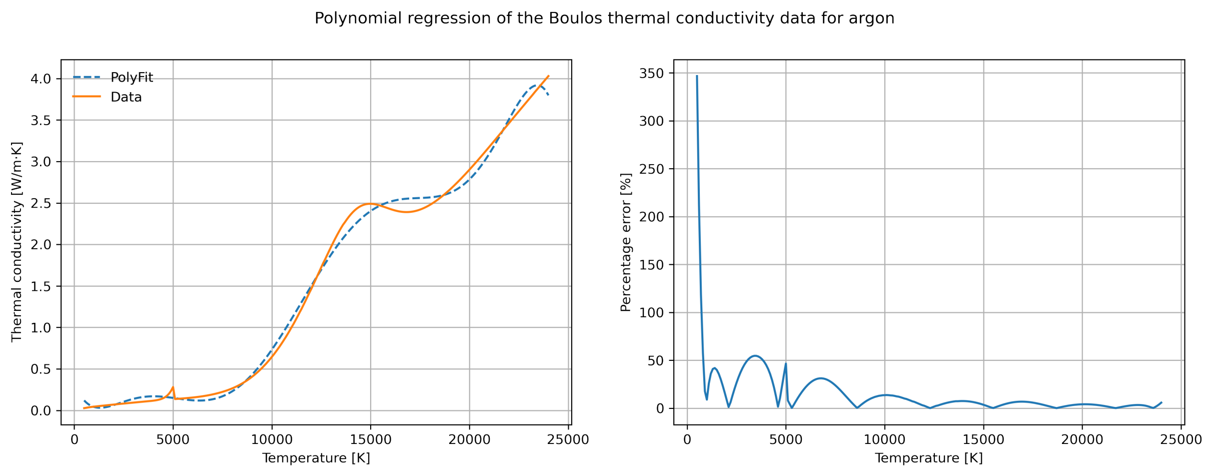

Figure 11: A temperature dependent thermal conductivity polynomial made using the polyfit function of the Numpy package.

In addition, two conductivity polynomials are used for argon, one for the values given by the paper from Westhoff et al.[29], with conductivity values from Devoto et al.[44], and a model of the conductivity below 9000 $[K]$ taken from Scott et al.[45] correlating conductivity to temperature as:

$$\sigma(T) = 0.2e^{\frac{T}{2000}} \tag{64}$$

The author does this to counteract the non-LTE effects close to the cathode and the high heating effect. To create the polynomial, this project fitted a twelfth degree polynomial to the tenth logarithm of the data set of Devoto et al.[44] from 9000 to 24000 $[K]$. This created the following polynomial with coefficients referred to as the Devoto et al.[44] conductivity shown in Figure 12.

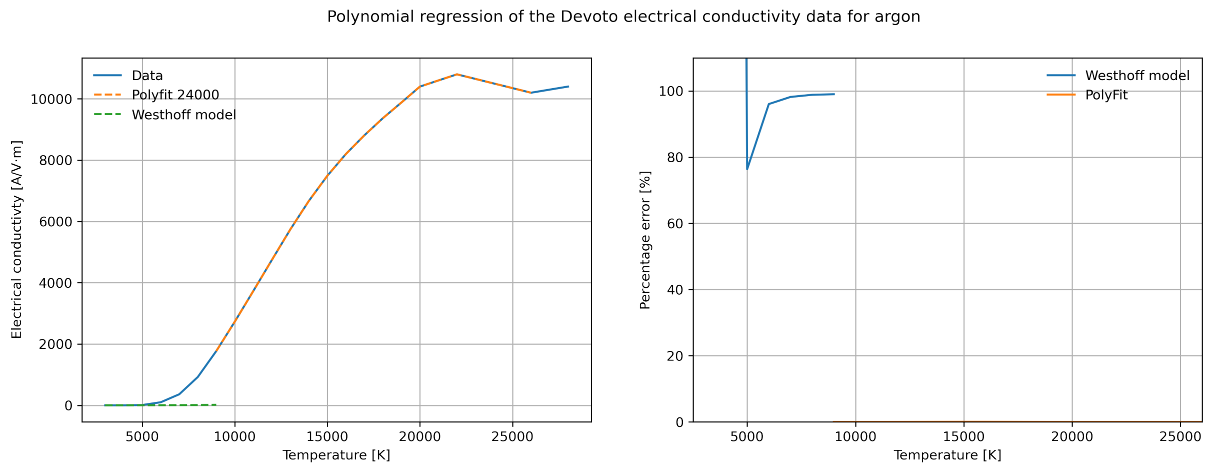

Figure 12: Conductivity model developed and used for the argon cases from the data of Devoto's and the low temp model used by Westhoff et al.[29].

In addition, a polynomial for the values from Boulos et al.[5] is also created. Here the data set was divided into three parts, below 1300 $[K]$, in between 1300 and 6000 $[K]$, and above 6000 $[K]$. The values between 1300 and 6000 were approximated using the tenth logarithm of the data. When implemented values of conductivity below $5.4577{\cdot}10^{-5}$ caused divergence. Hence the conductivity was capped at a constant value of $5.4577 \cdot 10^{-5}$ at lower values. The polynomial is presented in Figure 13 and called the Boulos conductivity.

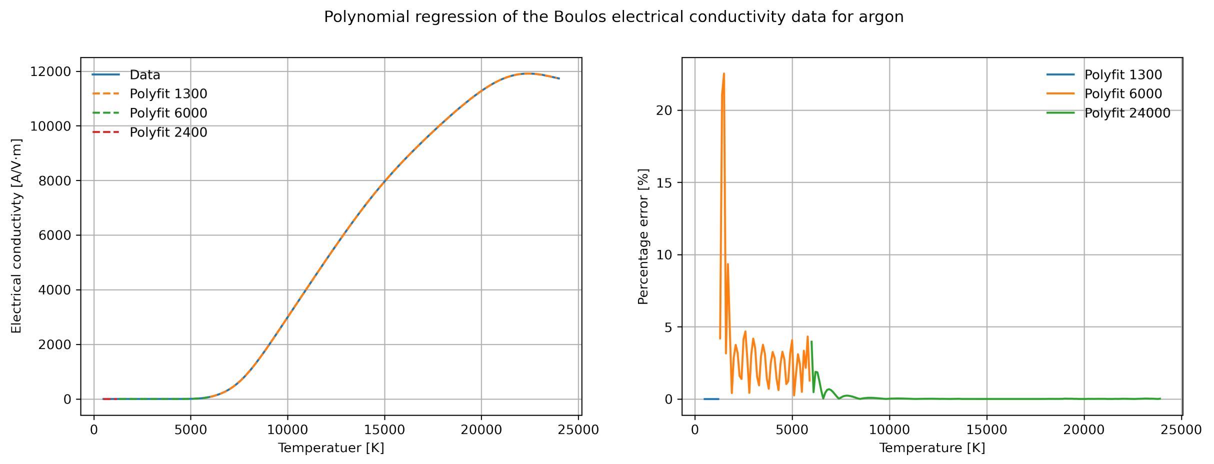

Figure 13: Conductivity model developed and used for the argon cases from the data of Boulos.

The data used by the NEC radiation model in the article of Westhoff et al.[29] originates from the work of Evans et al.[46]. However, this article only gives data from 10000 to 18000 $[K]$. The rest is given as private communication by Westhoff. Hence, the data must be sourced from a different article that matches the original data. The data of the articles from Essoltani et al.[47] and Cressault et al.[48] were compared with the known data in Figure 15. It was evident that the values from Cressault were more accurate in the given area. The derivative of the logarithm of the Cressault data was used to reconstruct the data with a simple Euler technique below 10000 $[K]$ using the data from Essoltani et al.[47] as the starting point, as shown in Figure 14.

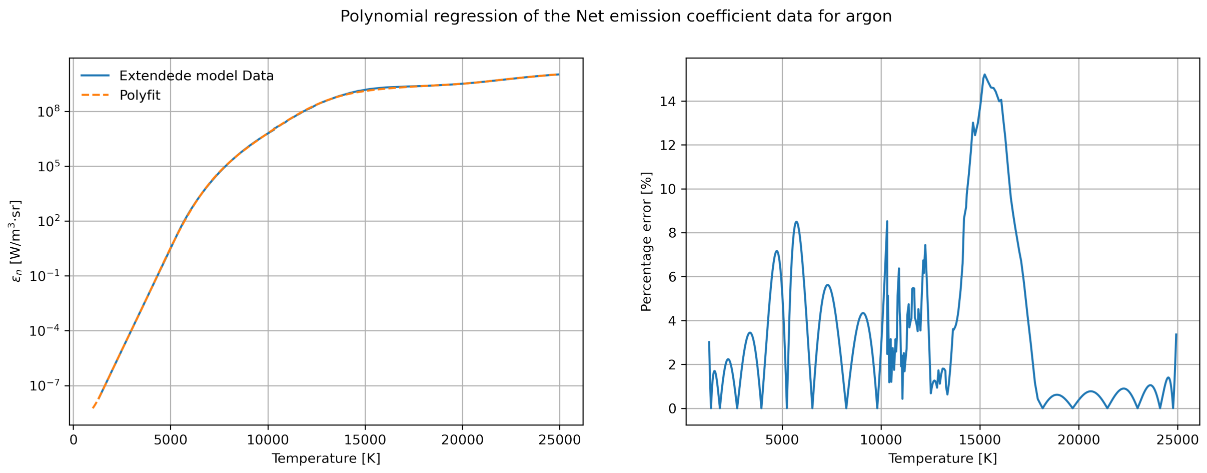

Figure 14: NEC radiation model of argon.

The comparison can be shown in Figure 15 given bellow:

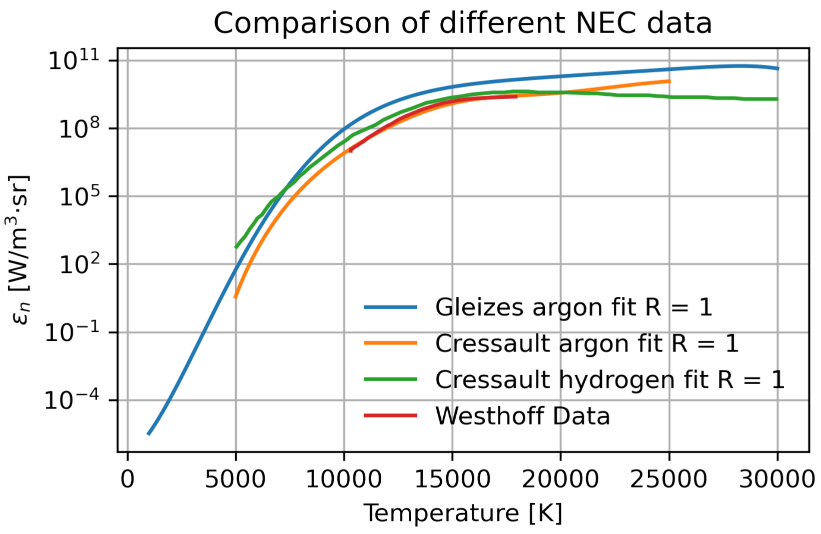

Figure 15: Comparisons of the NEC radiation model of argon.

The transport coefficients and thermodynamic properties were approximated for hydrogen using the same polynomial procedure as stated for argon. The density and heat capacity polynomials were extended to 27000 $[K]$. The min value was kept to $\frac{1}{10}$ of the minimum value of the data set, as this gave the best stability when checking the effect of stability of varying the min value and extension of temperatures on heat capacity and density. Unfortunately, the highest stable temperature achieved was 20000 $[K]$. Hence the simulation procedure is capped at this temperature. The transport coefficients' minimum values were kept to $\frac{1}{10}$ of the minimum value of the data set as well. The polynomials are presented in the figures below:

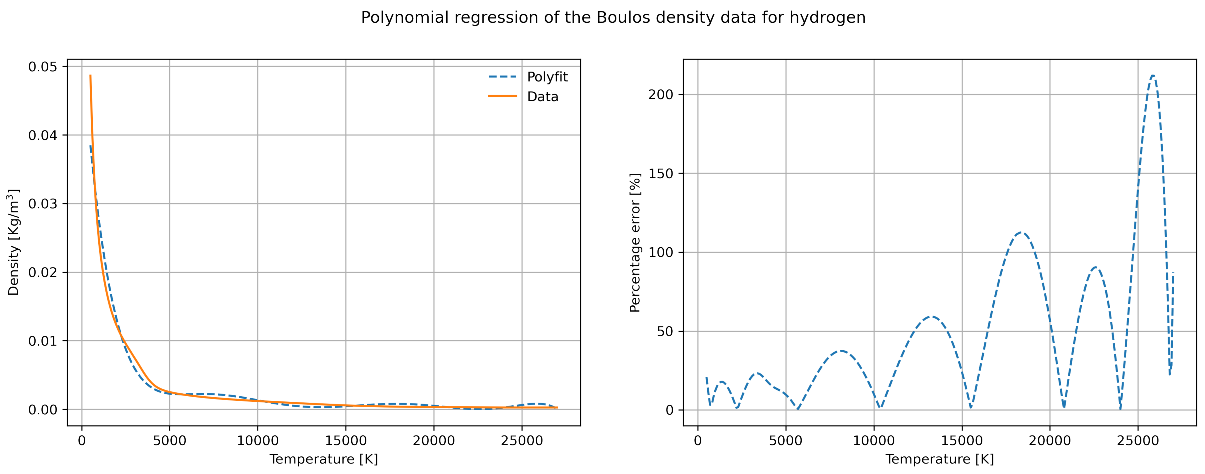

Figure 16: A temperature dependent density polynomial of hydrogen made using the polyfit function of the Numpy package.

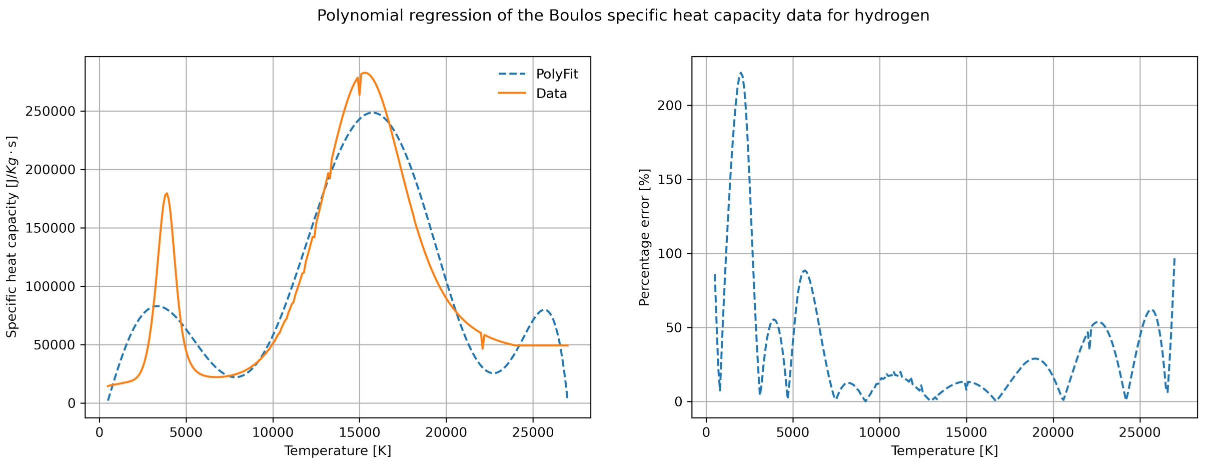

Figure 17: A temperature dependent heat capacity polynomial of hydrogen made using the polyfit function of the Numpy package.

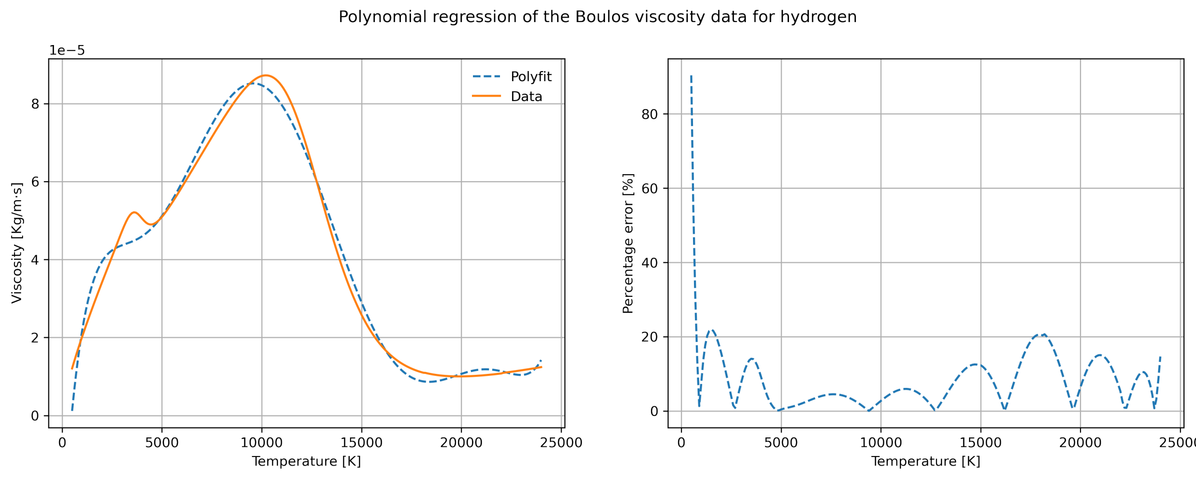

Figure 18: A temperature dependent viscosity polynomial of hdyrogen made using the polyfit function of the Numpy package.

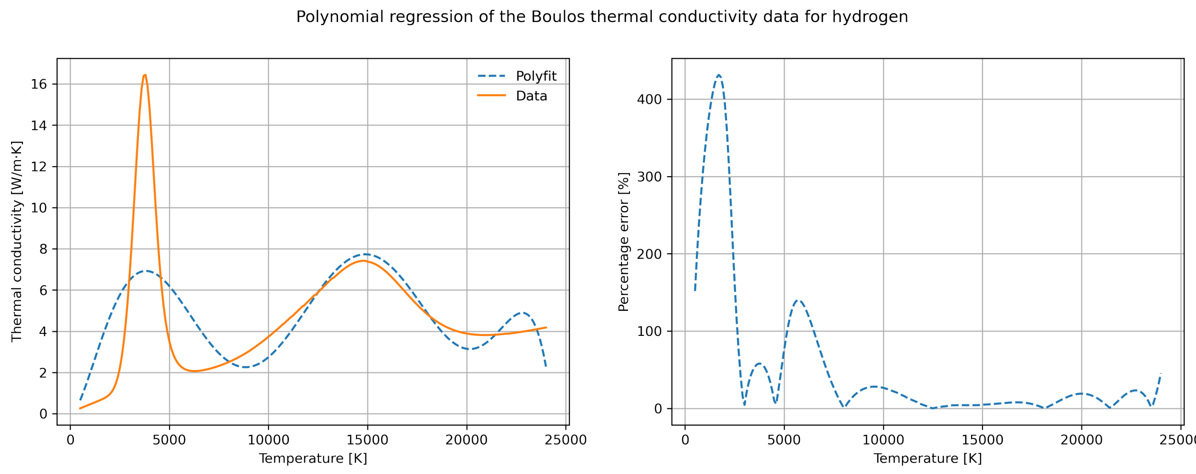

Figure 19: A temperature dependent thermal conductivity polynomial of hydrogen made using the polyfit function of the Numpy package.

Only the Boulos data set was used for the conductivity. The conductivity of hydrogen was approximated using the same procedure as argon. However, the data set was divided into two parts instead of three. It was expected that the solution would diverge for the lower conductivity values. As this was proven true, the conductivity had to be capped at $6.5421 \cdot 10^{-1}$ $[\frac{S}{m}]$ which is at 5000 $[K]$. The polynomial is given below:

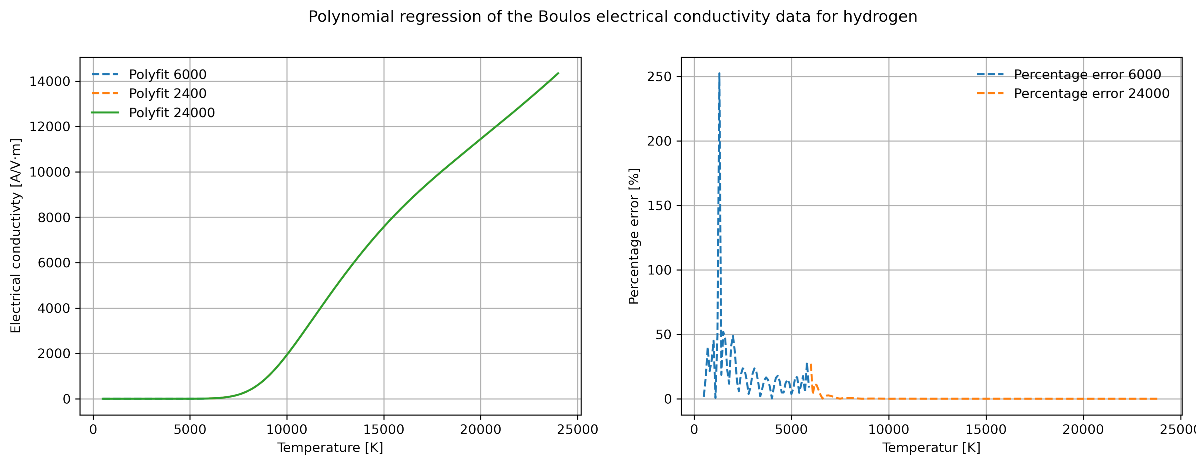

Figure 20: A temperature dependent thermal conductivity polynomial of hydrogen made using the polyfit function of the Numpy package.

The radiation data were taken from Cressault et al.[48] as the resulting model is most similar to the radiation model used for argon. Again the data were approximated using polynomial regression with a tewlth degree polynomial. The polynomial is listed below:

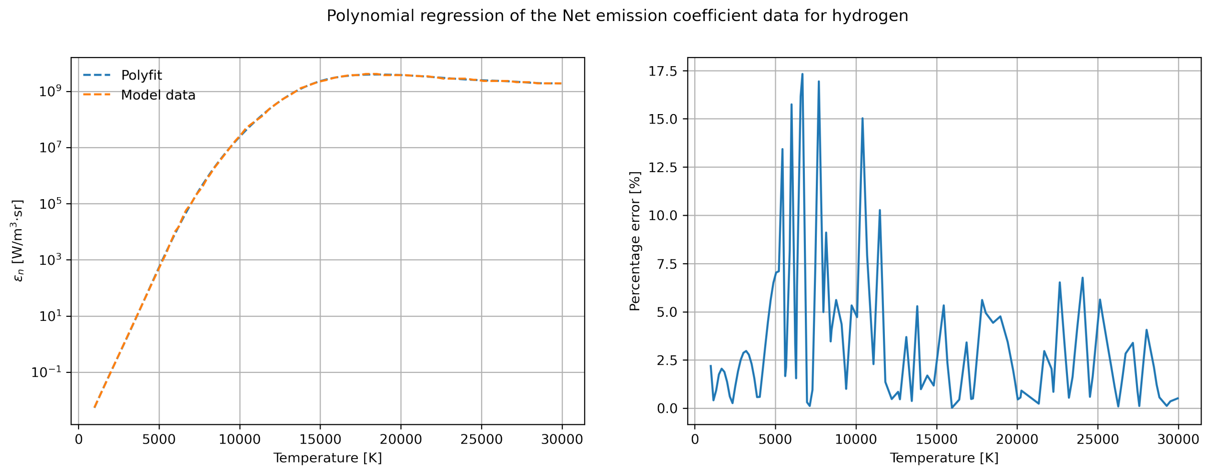

Figure 21: NEC radiation model of hydrogen.

All polynomial coefficients can be found in the source code in the appendix, where they are used.

9.7 Cases and simulation runs

This section lists the simulation runs performed during this project and group them in their respective cases. Each case looks at different aspects of the non-transferred thermal plasma torches litterateur and tries to verify the procedure used in this work and show how the plasma behaves. All important simulation defining models are listed in the tables below, such as what conductivity model is used and the applied inlet flow and current. This information gives the simulations' names, making it easier to distinguish between them. Naming rules are given in the tables listing the simulation runs. The cases presented are the verification case, comparison of hydrogen against argon, and parameter testing on both argon and hydrogen. Some simulations are reused, as their results contribute information to multiple cases.

The meshes used are small mesh(SM), which only uses the torch part of the simulation, hence capes the geometry at point F. The boundary conditions of $H - I$ are then used at the new outlet formed at F. The big mesh(BM) includes the area outside of the torch. The mesh BM takes a lot of computation power to run. Therefore, only one is performed in this project, and the rest are small meshes. The uniform mesh grading is given in the table below:

Table 4: Mesh grading values are given here

Mesh name	Small mesh(SM)	Big mesh(BM)
zone 1/ nr(x,y)	(40×40)	(40×40)
zone 2/ nr(x,y)	(80×40)	(80×40)
zone 3/ nr(x,y)	(80×40)	(80×40)
zone 4/ nr(x,y)	(0×0)	(160×40)
zone 5/ nr(x,y)	(0×0)	(160×40)
zone 6/ nr(x,y)	(0×0)	(160×80)

9.7.1 Verification case

The verification case is meant to test the solver developed in this work to verify if it produces results matching earlier simulations done in the field of non-transferred thermal plasma. If not for the verification case, the accuracy of the developed solvers cannot be known. Hence it is a vital step in qualifying the solver developed in OpenFOAM. This verification case will use the setup and values from the paper from Westhoff et al.[29], and compare the results from OpenFOAM with their given result. The values are given in the table below with the names and their meaning

Table 5: Simulation cases for argon verification/A = argon gas, DC = Devoto conductivity, BC = Boulos conductivity, SM = small mesh, BM = big mesh, U = inflow velocity, A = current applied.

Case Name	U $[\frac{m}{s}]$	I$[A]$	Mesh type	gass	Conductivity
ADCSMU2498A250	2.498	250	Small mesh	Argon	Devoto
ADCSMU8335A250	8.335	250	Small mesh	Argon	Devoto
ABCSMU8335A250	8.335	250	Small mesh	Argon	Boulos
ADCBMU8335A250	8.335	250	Big mesh	Argon	Devoto

These simulation runs are chosen as they are the closest to the case pf Westhoff et al.[29]. Furthermore, a value of 2.498 $[\frac{m}{s}]$ is chosen as it is the equivalent to the volumetric flow at $300[K]$ as reassurance in case the inflow from Westhoff et al.[29] has been misinterpreted.

9.7.2 Comparison of argon and hydrogen

This case explores the differences between argon and hydrogen to reveal both characteristics and compare them to other works that perform the same comparison, for example, Paik et al.[26]. Only Boulos conductivity is used, as it is the only source of data for hydrogen in this work. The runs are given below:

Table 6: Simulation cases for argon vs hydrogen/H = hydrogen gas, A = argon gas, DC = Devoto conductivity, BC = Boulos conductivity, SM = small mesh, BM = big mesh, U = inflow velocity, A = current applied.

Case Name	U $[\frac{m}{s}]$	I$[A]$	Mesh type	gass	Conductivity
ABCSMU0415A250	0.415	250	Small mesh	Argon	Boulos
ABCSMU8335A250	8.335	250	Small mesh	Argon	Boulos
HBCSMU8335A250	8.335	250	Small mesh	Hydrogen	Boulos

The simulation runs are chosen as they use the same current, inflow velocity, and inflow mass flow, as a velocity of 8.335 $[\frac{m}{s}]$ at 1000 $[K]$ is the same mass flow as argon at 8.335 $[\frac{m}{s}]$.

9.7.3 Argon parameter study

This case is analyzed as this work found a lot of theories on the effects of current and inflow velocity on the behavior of thermal argon plasma. Thus this analysis is crucial as it helps this work verify the results by comparing it to work from other authors, even do they use different parameters. The simulation runs are listed below:

Table 7: Simulation cases for argon parameter study/A = argon gas, DC = Devoto conductivity, BC = Boulos conductivity, SM = small mesh, BM = big mesh, U = inflow velocity, A = current applied.

Case Name	U $[\frac{m}{s}]$	I$[A]$	Mesh type	gass	Conductivity
ABCSMU0415A250	0.415	250	Small mesh	Argon	Boulos
ABCSMU0415A150	0.415	150	Small mesh	Argon	Boulos
ABCSMU8335A250	8.335	250	Small mesh	Argon	Boulos
ABCSMU8335A150	8.335	150	Small mesh	Argon	Boulos

The current is changed from 150 $[A]$ to 250 $[A]$ to see how the plasma behaves. Inflow velocity is also changed, from 8.335 $[\frac{m}{s}]$ to 0.415 $[\frac{m}{s}]$ respectively to see how the argon behaves to this input.

9.7.4 Hydrogen parameter study

This case compares the behavior of hydrogen when changing current and inflow velocity to other studies., which helps to verify the solver and results. However, not a lot of data was found on hydrogen parameter testing. Nevertheless, the behavior of hydrogen can be compared to the fundamental equations of the plasma, such as Lorentz force and Joule heating. In addition, insight into the behavior of hydrogen plasma is relevant for its usage for reducing metal oxides. The simulation runs are given in the table below:

Table 8: Simulation cases for hydrogen parameter study/H = hydrogen gas A = argon gas, DC = Devoto conductivity, BC = Boulos conductivity, SM = small mesh, BM = big mesh, U = inflow velocity, A = current applied.

Case Name	U $[\frac{m}{s}]$	I$[A]$	Mesh type	gass	Conductivity
ABCSMU0415A250	0.415	250	Small mesh	Argon	Boulos
ABCSMU8335A250	8.335	250	Small mesh	Argon	Boulos
HBCSMU8335A250	8.335	250	Small mesh	hydrogen	Boulos

The hydrogen plasma is altered with the same procedure as the argon parameter case in Table 7.

10 Solver procedure

This chapter explains in detail the solution procedure created by this project to solve the thermal plasma system. The procedure chosen is similar to the ones found and explained in the literature in section 7.2. In short, the procedure assumes LTE and LCE to simplify the necessary thermodynamic and transport properties to temperature dependent functions. Then electromagnetic equations are solved in vector potential form, which uses the electric potential equation, given as Equation 39 and magnetic vector equation, given as Equation 40. A given electric potential gradient and conductivity are used, producing a current and magnetic field. The electric fields are then coupled to the fluid governing equations, Equation 18 and Equation 21, through the Lorentz force and Joule heating ,given as Equation 25and Equation 23. Radiation is added through the NEC model, and Electron enthalpy transport is added as the last source term. Solving the governing equations results in a new temperature, which gives new transport coefficients and thermodynamic properties. Steady state is reached as the SIMPLE procedure solves the governing equations. After convergence, the conductivity is updated and the electromagnetic equations solved, which gives updated source terms. The semi-steady procedure is continued until the residual of the conductivity field has reached an acceptable limit, or the system has reached the maximum iteration value. A temperature field calculated using a different solver is used as the initial value. The specific solver solves for only the energy equation without any flow through the system but does it using the same procedure as the complete solver. The full procedure is summarized in a flow chart below in Figure 22.

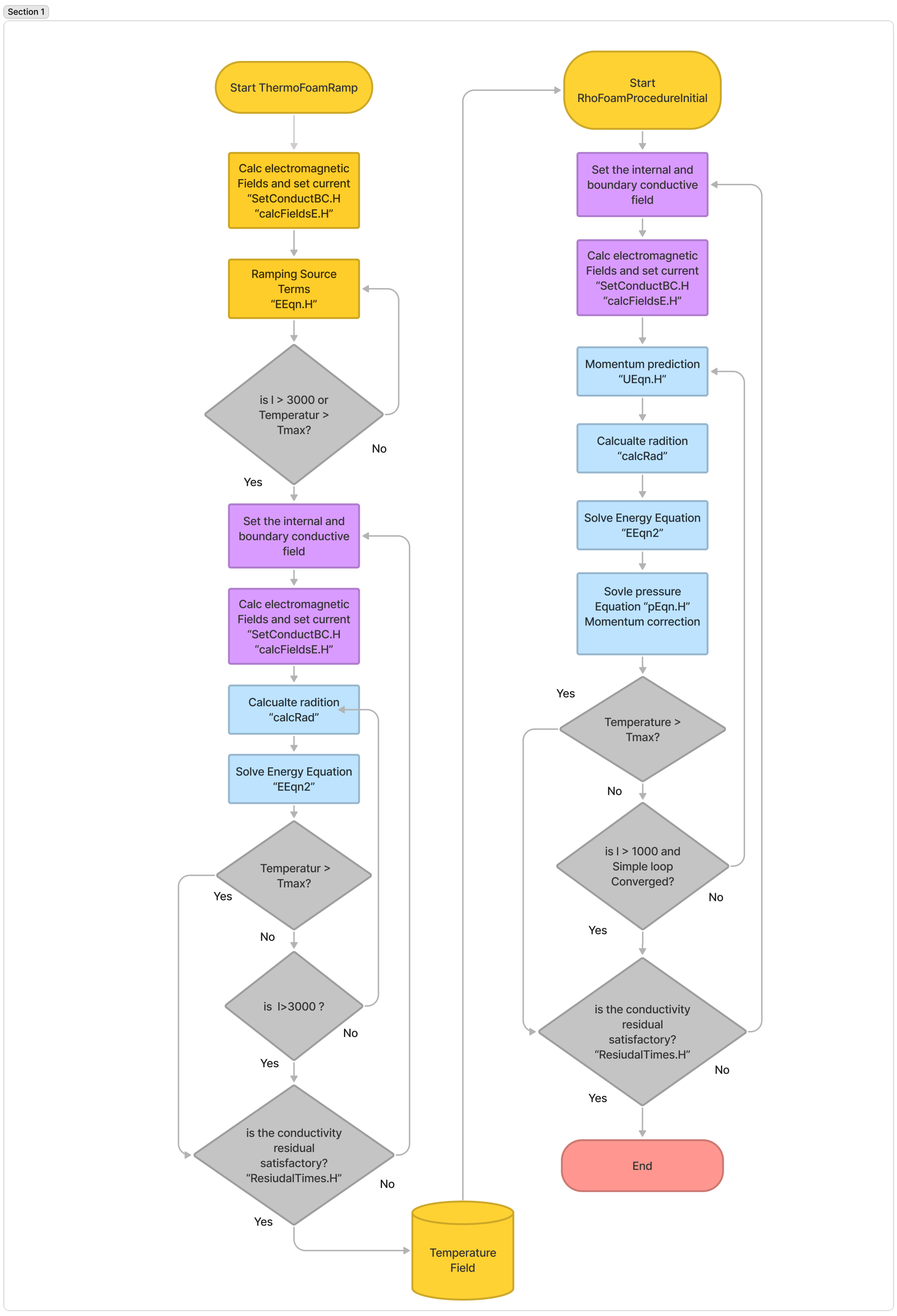
Section 1
Start ThermoFoamRamp
Calc electromagnetic Fields and set current "SetConductBC.H "calcFieldsE.H"
Ramping Source Terms "EEqn.H"
is I > 3000 or Temperatur > Tmax?
No
Yes
Set the internal and boundary conductive field
Calc electromagnetic Fields and set current "SetConductBC.H "calcFieldsE.H"
Calcualte radition "calcRad"
Solve Energy Equation "EEqn2"
Temperatur > Tmax?
Yes
No
is I>3000 ?
No
Yes
is the conductivity residual satisfactory? "ResiudalTimes.H"
No
Yes
Temperature Field
Start RhoFoamProcedureInitial
Set the internal and boundary conductive field
Calc electromagnetic Fields and set current "SetConductBC.H "calcFieldsE.H"
Momentum prediction "UEqn.H"
Calcualte radition "calcRad"
Solve Energy Equation "EEqn2"
Sovle pressure Equation "pEqn.H" Momentum correction
Yes
Temperature > Tmax?
No
is I > 1000 and Simple loop Converged?
No
Yes
is the conductivity residual satisfactory? "ResiudalTimes.H"
No
Yes
End

Figure 22: The Flow chart of the working procedure for the thermal plasma simulation

In the next sections the files and solvers used in Figure 22 are explained to give a better overview of how the procedure operates. The code shown in these procedures are found in the OpenFOAM source code guide[49].

10.1 Implementing governing equations

The governing equations are implemented using the standard SIMPLE algorithm from the standard *rhoSimpleFoam* solver, where the momentum-predictor with the pressure equation procedure is used. Finally, the Energy equation is defined, solved, and used to calculate the new density value for the pressure equation. This procedure and structure have been presented in the theory section of OpenFOAM in section 8.1.3.

The first equations to be implemented are the electromagnetic ones, which are not standard in OpenFOAM in the vector potential form. Next, code has been added to assign electromagnetic fields, electromagnetic transport coefficients and constants to the simulation, as they are not standard.

```

dimensionedScalar muMag //Magnetic permeability is added.
(
    "muMag",
    dimensionSet(1, 1, -2, 0, 0, -2, 0),
    physicalProperties
);

dimensionedScalar k_b  //Boltzmann constant is added.
(
    "k_b",
    dimensionSet(1, 2, -2, -1, 0, 0, 0),
    physicalProperties
);


Info<< "Reading field sigma\n" << endl;
volScalarField sigma #Creating the conductivty field.
(
    IOobject #Creating the field object
    (
        "sigma",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

Listing 2: Snippet from createFieldsE.H/ Electromagnetic fields are created.

The fields and constants are created, defined in Listing 2, and explained in section 5 on governing equations. In line with the theory, the electric potential ϕ and magnetic vector potential A are defined as implicit variables of the mesh. The magnetic vector field B and current density J are explicitly stated as operations of ϕ and A. Sigma σ is defined as a *volScalarField*, however standard practice usually define a transport coefficient as a *dimensionedScalar*. This choice is rooted in the manipulations later done with the conductivity fields in the simulation, where it was found to be easier to access and alter internal mesh and boundary mesh values of a *volScalarField*. The whole file is given in the appendix.

The next step is to implement the equations in OpenFOAM, which is done by using OpenFOAM's library to solve differential vector equations:

```

solve(fvm::laplacian(sigma, ElPot));//Fvm::laplacian used to dricritize
    the Laplacian term into an implicit discritization.

solve(fvm::laplacian(A)==sigma*muMag*(fvc::grad(ElPot))); //Used in the
    same manner here, however

B = fvc::curl(A); //Magnetic field implemented

Je = -sigma*(fvc::grad(ElPot)); //Current Implemented

F_l = (((Je^B))); //Lorentz Force source term implemented

E_J =  (Je&(-(fvc::grad(ElPot)))); //Joule source term implemented
```

Listing 3: Snippet from calcFieldsE.H/ Electromagnetic fields and sources are created.

In listing 3, the differential equations are created and discretized using the functions *laplacian*() and *grad*(). The *solve*() function solves the matrix with unknowns and constants given by the *fvm* and *fvc* methods. The exact implementation procedure has been tested on a conductive rod with an applied electric potential by Nordhagen[50], where the model shows satisfactory results. The procedure code originates from the work of the Chalmers university of science[51]. The file *calcFieldsE.H* creates the current, the magnetic field, the Lorentz source term, and the Joule heating source term. Operators are used according to the definition of the source terms and OpenFOAM operators in section 8.

10.2 Source Terms

It is necessary to connect the electromagnetic source terms to the governing equations to connect the electromagnetic fields and the plasma continuum. Connecting them is done by adding the previously defined source terms into the momentum prediction and energy equations. The Lorentz source is added to the momentum equation in $UEqn$:

```

tmp<fvVectorMatrix> tUEqn
   (
        fvm::div(phi, U)
      + MRF.DDt(rho, U)
      + turbulence->divDevTau(U)
     ==
        fvOptions(rho, U) + F_l// The Lorentz force is added here as a
 source term.
   );
```

Listing 4: Snippet from UEqn.H/ Electromagnetic fields and sources are created.

The soruce term is added to the constant side, as all its discretization functions are fvc, as can bee seen in listing 3. The source terms for the energy equation is added in the same way as listing 4:

```

tfvScalarMatrix EEqn
     (
         fvm::div(phi, he)
       + (
             he.name() == "e"
           ? fvc::div(phi, volScalarField("Ekp", 0.5*magSqr(U) + p/rho))
           : fvc::div(phi, volScalarField("K", 0.5*magSqr(U)))
         )
       + thermophysicalTransport->divq(he)
      ==
       fvOptions(rho, he) + C_T/*Ramp coefficient*/*E_J /*Joule heating
 */  - C_T*Rad  /* Radiation*/ + C_T*((5*(k_b))/(2*thermo.Cp()*(e_c)))
 *(Je&fvc::grad(he)) /*Electron enthalpy diffusion*/
     );
```

Listing 5: Snippet from EEqn.H/ Electromagnetic fields and sources are created.

In listing 5, the radiation sink term, the net emission coefficient, Joule heating, and the electron enthalpy diffusion are added. The term C_T is a coefficient that will be explained later. The electron enthalpy diffusion is implemented as the dot product between the current density and the gradient of the enthalpy to create the source term in the form as presented in Westhoff et al.[29]. Specifically, he stores the enthalpy field and $thermo.Cp()$ returns the heat capacity field of the model. The procedure for adding and declaring Joule heating is the same as done in the source code of the $fvOptions$ Joule source option and explained in the OpenFOAM API guide[52].

10.3 LTE condition and polynomials

To enforce the LTE condition, temperature-dependent polynomials are created to relate radiation, transport coefficients, and thermodynamic variables to the defined temperature in each cell. The data used is found in the book of Boulos et al.[5]. The data sets limitation is a temperature increment of a 100 $[K]$ per data point and the limited temperature range of 500 to 24000 $[K]$. In comparison, Westhoff et al.[29] derives their thermodynamic data and transport coefficient from the author R.S. Devoto[44]. The accuracy of the data compared to other authors is debated in this paper from Murphy[6], which compares both Boulos and Devoto for hydrogen and argon hydrogen mixtures. Devoto's viscosity values are higher than Murphy's at the peak of 10000 $[K]$. Boulos et al. are between these values at the peak but higher than both at values above 15000 $[K]$. Thermal conductivity is generally similar among all authors. However, for electrical conductivity, the values of Boulos are higher than those of Murphy and Devoto.

The polynomial regression and implementation into OpenFOAM differ for conductivity and the NEC compared to other thermophysical variables. Thus, the general procedure for the other thermophysical variables is discussed first, which are density ρ, heat capacity c_p, viscosity, and μ. These are fitted using the least square method of the Polyfit function found in the Numpy package in python. Some adjustments are made to the curve fitting procedure because of the divergence and instability of the simulation at certain temperatures when using the standard procedure. This adjustment comes at the cost of the accuracy of the polynomial fitting. In addition, the polynomial input of the thermophysical properties is limited to a seventh degree polynomial. This limitation is due to restrictions implemented in the thermophysical library in OpenFOAM.

The adjustment was to extend the temperature range by duplicating the last value of the data set to extend the temperature range. Figure 9 shows this extension. The second adjustment was to limit the minimum value of the polynomial. The curve fit of polynomials is often oscillatory around the value of the data set. Hence negative values might occur, which is considered nonphysical and makes the procedure unstable. Hence a loop was implemented, which increases the value of a data point if the polynomial regression returns a value below a certain threshold mentioned in section 9.6. After several iterations, the regression curve is lifted. The threshold is always above zero, as thermodynamic values close to zero would not help with stability.

The conductivity polynomial is implemented in the file *SetConductInternalREal.H*, where the internal cell value temperature of the mesh is used to create a new conductivity field from the supplemented polynomial.

```

scalarField& sigmaCells = sigma.primitiveFieldRef(); //Accessing the field through pointer.
     forAll(sigmaCells, sigmacelli) //Iterate through all cell center
```

```
    values of the mesh.
            {
                sigmaCells[sigmacelli] =
                (-2.68872190e-46)*pow(thermo.T()[sigmacelli],13) +
    (5.71816721e-41)*pow(thermo.T()[sigmacelli],12) + (-5.56635110e-36)*
    pow(thermo.T()[sigmacelli],11) + (3.28386804e-31)*pow(thermo.T()[
    sigmacelli],10) + (-1.30989753e-26)*pow(thermo.T()[sigmacelli],9) +
    (3.73058177e-22)*pow(thermo.T()[sigmacelli],8) + (-7.80413235e-18)*
    pow(thermo.T()[sigmacelli],7) + (1.21412711e-13)*pow(thermo.T()[
    sigmacelli],6) + (-1.40469700e-09)*pow(thermo.T()[sigmacelli],5) +
    (1.19363669e-05)*pow(thermo.T()[sigmacelli],4) + (-7.24077915e-02)*
    pow(thermo.T()[sigmacelli],3) + (2.96922898e+02)*pow(thermo.T()[
    sigmacelli],2) + (-7.37705762e+05)*thermo.T()[sigmacelli] +
                8.38636954e+08; //The implemented polynomial.



                 if ( thermo.T()[sigmacelli] < 9000 ){ //Set
    temperatures values below to other model.
                sigmaCells[sigmacelli] = 0.2*Foam::exp(thermo.T()[
    sigmacelli]/2000);
            }
        }
```

Listing 6: Snippet from SetConductinternalReal/ Conductivity polynomial created and used.

As seen in listing 6, the internal mesh values are altered directly according to the polynomial using a *forAll* loop and pointer. The line *thermo.T*() allows access to the mesh's temperature value. The procedure presented here is an example from argon plasma simulations with a specific low temperature model.

The radiation model chosen is the NEC model, and it's implemented using the same method as in listing 6 in the file called *calcRadLower.H*

```

scalarField& RadCells = Rad.primitiveFieldRef();//Acessing the NEC
    radience field.





forAll( RadCells, Radcelli)
    {
        RadCells[Radcelli] = 4*Foam::constant::mathematical::pi*pow(10,(
     (-9.74738451e-52)*pow(thermo.T()[Radcelli],13) + (1.72988685e-46)*
    pow(thermo.T()[Radcelli],12) + (-1.36979930e-41)*pow(thermo.T()[
    Radcelli],11) +
        (6.37817811e-37 )*pow(thermo.T()[Radcelli],10) +
        (-1.93728139e-32 )*pow(thermo.T()[Radcelli],9) + (4.02444763e
   -28)*pow(thermo.T()[Radcelli],8) +
        (-5.83025572e-24)*pow(thermo.T()[Radcelli],7) +
        (5.89782481e-20)*pow(thermo.T()[Radcelli],6) +
        (-4.11017114e-16)*pow(thermo.T()[Radcelli],5) +
        (1.92036935e-12)*pow(thermo.T()[Radcelli],4) +
        (-5.78117589e-09)*pow(thermo.T()[Radcelli],3) +
        (1.05596042e-05)*pow(thermo.T()[Radcelli],2) +
        (-8.26080120e-03)*thermo.T()[Radcelli] +
        -6.32519497e+00));}
```

Listing 7: Snippet from calcRadLower.H/ Radiation polynomial created and used.

The value of the polynomial is used as the power of ten, as all radiation values are fitted by using the $log(y)$ of the data set, as it gives better accuracy. The advantage of directly accessing and altering the cell values for listing 5 and 6 is the flexibility of easily changing the polynomial and adding extra models through conditional statements.

10.4 Boundary conditions

The BC values have to be altered during the procedure to set the current and deal with the non-LTE phenomena occurring in the region adjacent to the electrodes. These procedures are not standard in OpenFOAM, hence have to be set by altering the boundary values directly. This is done in the file *SetConductBC.H*.

```

const fvPatchList& patches = mesh.boundary(); //pointer to access
   boundary list
        forAll (patches , patchi ) {
            const fvPatch& SigmaPatch = patches[patchi]; //Access each
   patch.


            forAll(SigmaPatch, faceI) //Iterate through each boundary
   patch  {
                label faceCelli = SigmaPatch.faceCells()[faceI];//
   retrieving the label og the face cell value.

                sigma.boundaryFieldRef()[patchi][faceI] = sigma[
   faceCelli];//set boundary equal to face value
                }
}
sigma.correctBoundaryConditions();
```

Listing 8: Snippet from SetConductBC.H/ setting boundary equal to internal cell value.

The code from listing 8 essentially sets the boundary equal to the cell it is facing, hence the conductivity of the neighbour cell. This avoids the non-LTE effects close to electrodes, as discussed in theory. Thus the simulation becomes more stable but less accurate.

This next piece of code in *SetConductBC.H* calculates and sets the current boundary condition at the cathode. This is done by calculating an appropriate gradient for the electric potential and applying it to the boundary where the current flows.

```


    label patchIDLeftRCWall = mesh.boundary().findPatchID("leftRCWall"); // Retrieving the id of the cathode boundary patch.
     const polyPatch& Elpatches = mesh.boundaryMesh()[patchIDLeftRCWall];
    fixedGradientFvPatchScalarField& ElPatch = refCast<fixedGradientFvPatchScalarField>(ElPot.boundaryFieldRef()[patchIDLeftRCWall]);
    // Retrieving the gradient of the patch to set the gradient.
    forAll (ElPatch, EfaceI)
    {

        ElPatch.gradient()[EfaceI] = -((3e7)/(sigma.boundaryField()[patchIDLeftRCWall][EfaceI]))*pow(1 + Foam::exp(2*250000*(Elpatches.faceCentres()[EfaceI].y() - 1.6e-03)),-1);
        // The gradient is set according to the given current distribution and current density
        if (Elpatches.faceCentres()[EfaceI].y() > 1.61e-3) {
            ElPatch.gradient()[EfaceI] = 0;
        }//Zero gradient set where the current is not to flow.

    }

    ElPot.correctBoundaryConditions();//make sure that the boundary is updated.
```

Listing 9: Snippet from SetConductBC.H/ setting the current.

The gradient is set to the specifics of the user, where a Heaviside approximation is used to achieve a smoother transition from a set gradient at the boundary to a zero gradient. Current density is divided by the conductivity of the boundary patch to create the appropriate Neumann boundary condition for the electric potential equation. As the temperature and conductivity of the wall changes, the gradient condition will change dynamically to keep the current density at the cathode boundary surface.

10.5 Initialization and Solution procedure

In this section, the files explained earlier and in theory, are put together to show the full solution procedure. Its structure is connected to the solution procedure explained in section 7.2. First, the residual model is explained, then the initialization procedure built on $thermoFoam$ is presented, then the edited $rhoSimpleFoam$ is explained.

The residual file which is used and accesses the internal values of the mesh is $ResiudalTimes.H$. It calculates the residual of the conductivity through the sum of the absolute difference between two sequentially conductive fields. This is demonstrated in the equation below. The full code is in the appendix.

$$r_{abs} = \left(\Sigma\sigma_i^n\right) - \left(\Sigma\sigma_i^{n-1}\right) \tag{65}$$

Here n is the iteration number, and i is the cell number. The ratio between sequential residuals is calculated to find how the residual change changes with time. A low residual would indicate that the change between consecutive conductivity fields is low. The ratio between consecutive residuals indicates if the residual is falling or increasing. The ratio between the new residual and the first is also calculated to see how the residual evolves, where a small residual would indicate a steady state solution. If the residual of the conductivity change does not drop to zero or a satisfying limit, the simulation run is stopped at 3 million iterations.

The solver $thermoFoam$ is used to calculate the initial value for the full solver procedure. It is used because it is easier to keep the energy equation stable compared to the full decoupled system with applied plasma features. This gives a useful initial value as a large Joule heating source is bypassed in the initialization of the full procedure.

When starting the edited $thermoFoam$ solver known as $thermoFoamRamp$, it starts by slowly ramping up the source terms as a linear function of the *runtime*. This is done in the energy equation file $EEqn.H$

```

scalar C_T = (  1/(  1+(5000/runTime.value())  )  ); //Ramping
    coefficient.

    fvScalarMatrix EEqn
    (
        fvm::div(phi, he)
      + (
            he.name() == "e"
          ? fvc::div(phi, volScalarField("Ekp", 0.5*magSqr(U) + p/rho))
          : fvc::div(phi, volScalarField("K", 0.5*magSqr(U)))
        )
      + thermophysicalTransport->divq(he)
     ==
        fvOptions(rho, he) + C_T*E_J  - C_T*Rad + C_T*((5*(k_b))/(2*
    thermo.Cp()*(e_c)))*(Je&fvc::grad(he))//coefficient multiplied with
    source terms.
    );
```
Listing 10: Snippet from EEqn.H/ Slowly ramping up the source terms.

The coefficient C_T, in listing 10, increases with the run time of the simulation. This makes it possible to slowly ramp up the joule heating source term for a constant conductivity decoupled from the temperature. The result is a high temperature area next to the cathode. This initial ramp procedure is vital as the next step, where the conductivity is coupled to the temperature field, diverges if the temperature dependent conductivity is too small. When the conductivity is coupled to the temperature, a new energy equation file is used called *EEqn2.H*.

```

scalar C_T = (  1/(  1+(8000/pow(l,1.45))  )  );
    if (C_T > 0.95) {
      C_T = 1;
    }
    fvScalarMatrix EEqn
    (
```

Listing 11: Snippet from EEqn2.H/ Solving coupled energy equation.

The difference in listing 9 compared to listing 10 is the change of the coefficient C_T. It will here quickly ramp up to 1 within 6000 iterations. This is done for each iteration of the conductivity field as the first iterations give highly varying joule heating fields, which will cause divergence if not ramped up slowly.

The full solver *thermoFoamRamp* is shown below, demonstrating the solution procedure and usages of the different energy equations. All files described earlier are included in the order required for the simulation procedure.

```
bool l1 = true;
for (int i = 0; i <= 1000000; i++){ //Outer loop, could have used a
   while loop.
    int l = 1; //Ramp variable.
    #include "SetConductBC.H"//Set the conductivity field and current
   for boundary.
    #include "calcFieldsE.H" // Calculate the electromagnetic fields.
    #include "ResidualTimes.H" //Keep track of residual value.

    while (runTime.loop()) {//runTime.loop() Moves time forward in the
   system
        Info<< "Time = " << runTime.timeName() << nl << endl;
        simple.storePrevIterFields(); //Store the previous fields.
        #include "calcRadLower.H" //Calculate the radiation.

        if (l1) { //If statement to choose between solver modes
            Info<< "We are warming up "<< endl;
            #include "EEqn.H" //Solve energy equation initial ramp.
        }
        else {
            Info<< "We are solving!"<< endl;
            #include "EEqn2.H"//Solve energy equation.
        }
        double T_max = max(thermo.T()).value(); //Max temperature.
        l+= 1;//move counter.
        if ((( (l > 10000)) ) || (T_max > 26000)) //Statement to break
    inner loop{
            l1 = false;
            runTime.writeNow();
            runTime.write();
            #include "SetConductInternalReal.H"//Temperature dependent
    conductivity.
            Info<< "It's breaking!!! "  << nl << endl;
            break;//Used to move out of inner loop.
        }
    }
}
```

Listing 12: Snippet from thermoFoamRamp.C/ Procedure for initializing the temperature field

The algorithm in listing 12 is a double loop system. As mentioned in theory, the outer loop alters the conductivity and current, and the inner loop solves the energy equation to account for the change in the electromagnetic system. The inner loop breaks if the ramp variable l has reached beyond its threshold limit or the max temperature of the field is too large. The reason behind this is the divergence of the simulation if the thermophysical polynomials reach temperatures beyond the stability limit. The ramp coefficient C_T and break criteria for l are designed such that the loop spends half the inner loop ramping up and half solving at the exact value of the source terms. The work of Busse et al.[18] and Sass-Tisovskaya[14] inspired the procedure. The function of C_T and the breakage point of the inner loop for l is not found from any theory but tested by trial and error. It is not optimized to create a fast solution, but to achieve a non-divergent for the specific case setup in this work.

The temperature field from the previous simulation is then used as the starting point for the *rhoSimpleFoam* edited solver named *rhoFoamProcedureInitial*, whose structure closely resembles that of *thermoFoamRmp*, except for the inclusion of momentum prediction procedure in *UEqn.H*, and pressure equation solution procedure in *pEqn.H*. The final solver does not use the C_T ramp coefficient as the initial value is close enough to make the initialization steady with a low enough relaxation factor for enthalpy h, velocity U, and pressure p.

```
bool l1 = true;
 for (int i = 0; i <= 1000000; i++)  //Outer loop, could have used a
    while loop.
        {
            int l = 1; //Ramp variable.
            #include "SetConductBC.H"//Set the conductivity field and
    current for boundary.
            #include "calcFieldsE.H" // Calculate the electromagnetic
    fields.
```

```
            #include "ResidualTimes.H" //Keep track of residual value.

            while (runTime.loop()) //runTime.loop() Moves time forward
   in the system
            {//Inner running loop.
                Info<< "Time = " << runTime.timeName() << nl << endl;
                    simple.storePrevIterFields(); //Store the previous
   fields.
                    // Pressure-velocity SIMPLE corrector
                    #include "UEqn.H"
                    #include "calcRadLower.H"
                    #include "EEqn2.H"
                    #include "pEqn.H"
            turbulence->correct();
            thermophysicalTransport->correct();
                    l+= 1;//move counter.
                    double T_max = max(thermo.T()).value(); //Retrieve
   max temp value.


                    if ( ((simple.converged()) && (l > 1000)) || (T_max
   > 26000)) //Statement to break inner loop
                    {
                        l1 = false;
                        runTime.writeNow();
                        runTime.write();
                        #include "SetConductInternalReal.H"//Create temp
    dependent
                        electrical conductivity.
                        Info<< "It's breaking!!! "  << nl << endl;
                        break;//Used to move out of inner loop.
                    }
```

Listing 13: Snippet from rhoFoamProcedureInitial.C/ Procedure for initializing the temperature field

The radiation and momentum predictor files must be included before the energy equation for the procedure in Listing 13, as these give all the guessed fields necessary to calculate the enthalpy. The thermophysical properties are updated before being used in the pressure equation procedure. One major difference between the solvers is the inclusion of the $Simple.converged()$ check with the iteration constant l. The function $converged()$ forces the inner loop to break if the l is satisfied and the system has reached its convergence tolerance criteria, which were set at 10^{-3} for p, 10^{-3} for U, and 10^{-5} for h. Hence, the fluid fields have to converge for each new electrodynamic state of the system. The relaxation factors are not given, as they were not constant during the simulation, as the start of the simulation required smaller relaxation factors than at the end. Generally, the starting relaxation factors were given as 10^{-1} for all fields except pressure and enthalpy. They were given the value of 10^{-2} and 10^{-3}, respectively. After a couple of convergences, the pressure relaxation factor was changed to $3 \cdot 10^{-1}$. For argon, the relaxation factor for enthalpy was pushed up 10^{-1}. However, the hydrogen had to be kept at a value of 10^{-2} to prevent solution divergence.

11 Result

In this section, the results from the simulation runs are presented. They are divided into four cases, all with their own goals of either verifying the solver procedure or showing relevant results for hydrogen plasma. The results are presented as graphs with two axes. The data is extracted from the two-dimensional axisymmetric data either at a constant point in the axial direction, where data is taken along the radial direction, or at a constant radial value in the axial direction. Information is given in plots by either stating the axial value or the boundary surface at which the data is extracted. A percentage difference is used to quantify the difference between different data sets more easily. Note that this makes comparing values within a bar diagram not as straightforward as errors in the simulation will be more weighted at lower temperatures. However, it is still a valuable tool for comparing simulation runs against other results.

Results using a bar diagram are linearly interpolated to find the percentage difference. This is done as the data sets of values extracted from the case of Westhoff et al.[29] do not contain the same incremental data set structure. All values from Whesthoff et al.[29] are extracted using the data tool ScanIt[53], which adds some uncertainty to the values. Arc lengths are presented in the result, and are taken as an approximation using the anode current profile. The value is an approximation as the current is distributed along a larger zone of the anode. The approximation used in Westhoff et al.[29] is not known. Thus this work decided to use the maximum value of the anode current profile to set the axial point for the arc length.

11.1 Argon verification case

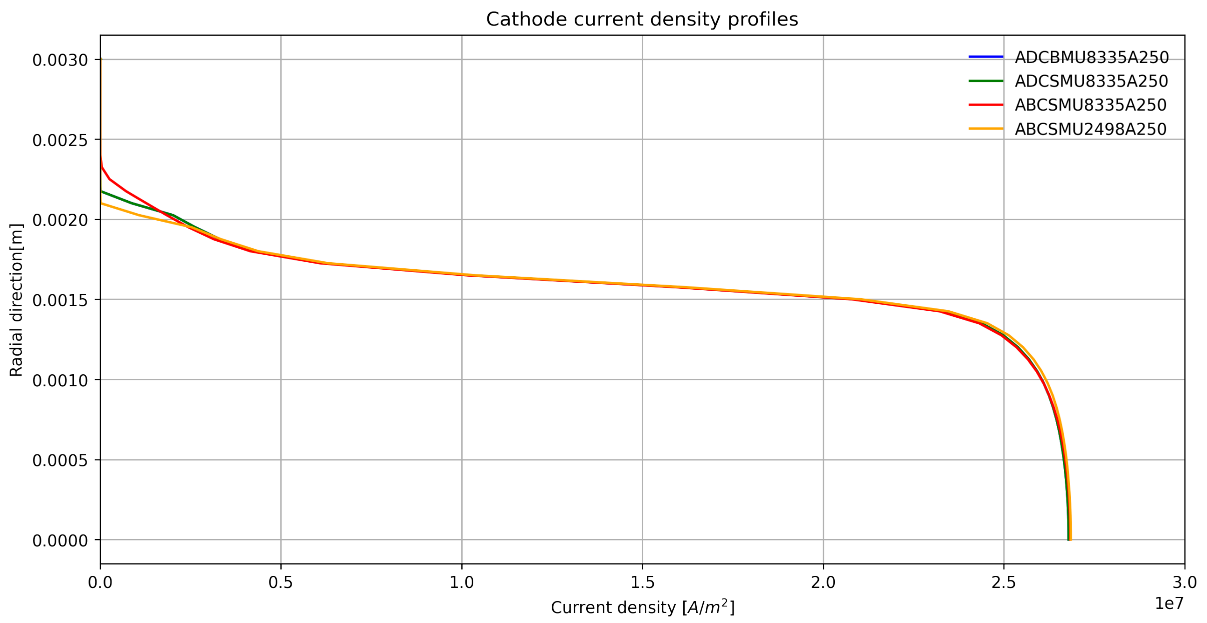

Figure 23: The current distribution at the cathode of the argon verification simulation runs are shown here. Current density is on the x-axis, and radial direction on the y-axis.

The cathode current density shows a current that slowly reaches zero as it reaches a radial value of around 22 $[mm]$ for all runs. The max current density is here given as 2.685 $[\frac{A}{m^2}]$.

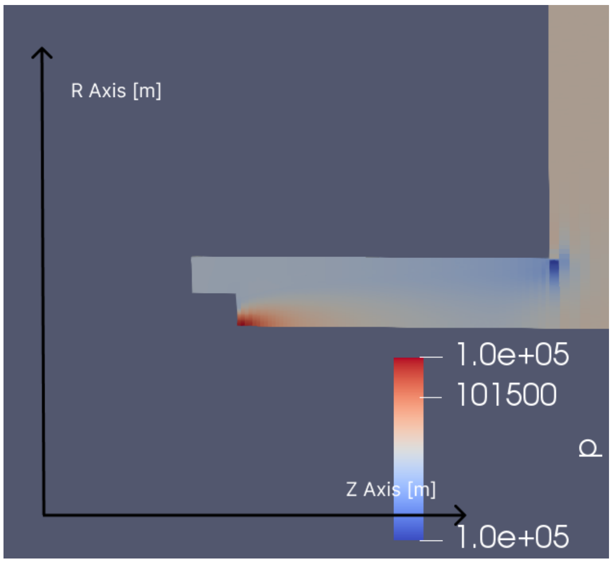

Figure 24: The pressure field of the $ADCBMU8335A250$ simulation run. Axial and radial direction is shown.

The pressure field shows a clear high pressure zone close to the cathode of simulation run $ADCBMU8335A250$, and a decreased pressure at the torch outlet.

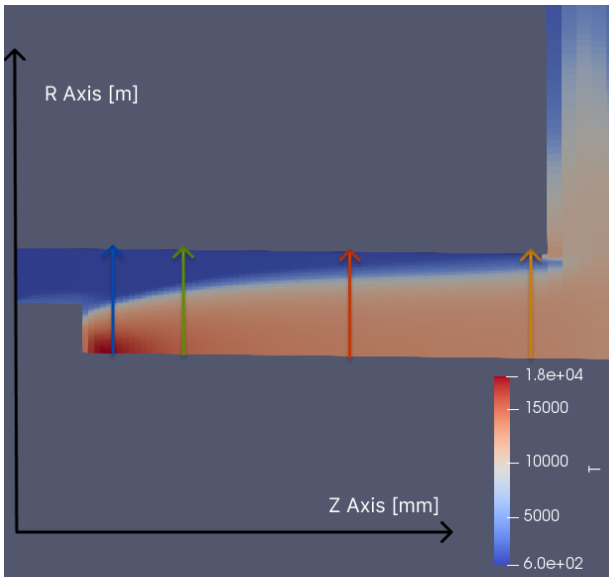

Figure 25: The temperature field of the $ADCBMU8335A250$ simulation run. Arrows indicates the extraction line of data for radial temperature profiles given at $z = (1.7, 5.7, 15.8, 27.8)$ $[mm]$. The model is zoomed in at the torch section.

The result from the case $ADCBMU8335A250$ is presented in Figure 25 and shows qualitatively a hot zone close to the cathode and a decrease of temperature in the positive Z-direction. From the contour colors, a drastic temperature change is observed, defining a clear arc profile. The temperature curls around from the axis to the outer point of the anode.

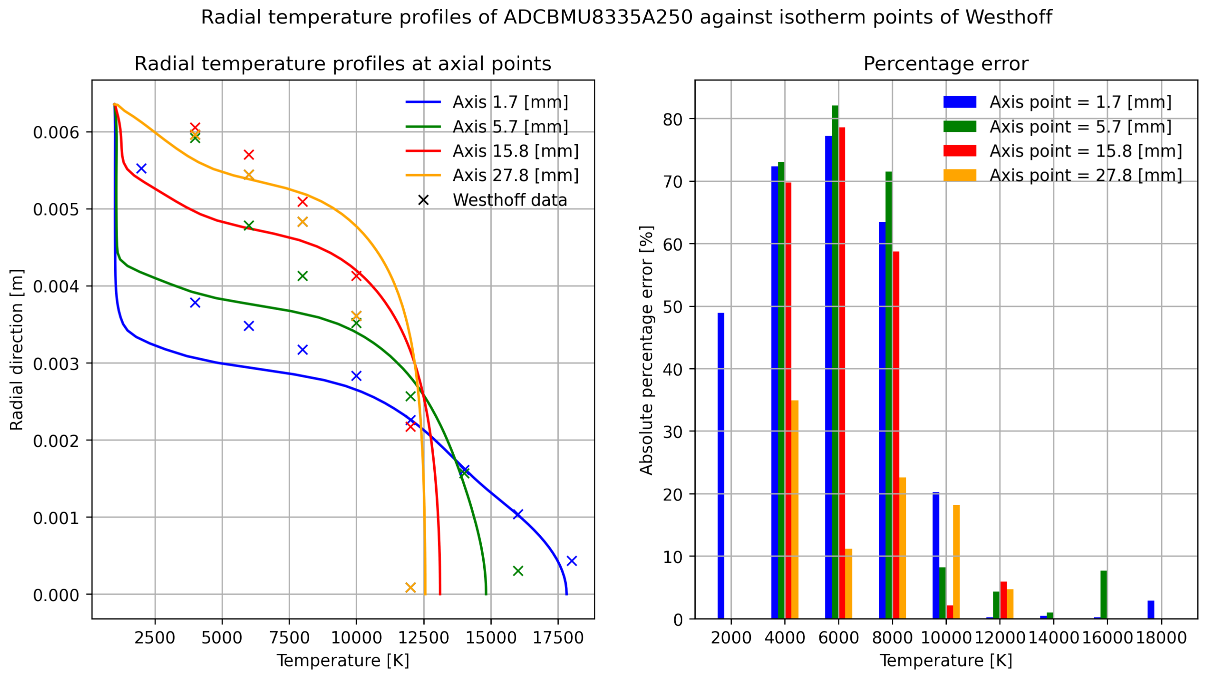

Figure 26: The radial temperature profile at given location at the z-axis of $ADCBMU8335A250$ compared against the isothermal data of Westhoff et al.[29], represented as crosses. The second graph shows the absolute percentage difference between a cross and the temperature.

Figure 26 shows that the maximum temperatures at axial positions of 1.7 and 5.7 $[mm]$ are lower than the benchmark case. The temperature drops a lot faster for all points at higher radial values of 3 $[mm]$. However, the drop is not as large at the far downstream point of 27.8 $[mm]$. The percentage error also follows this trend as the yellow bar shows the lowest percentage error for the isotherms.

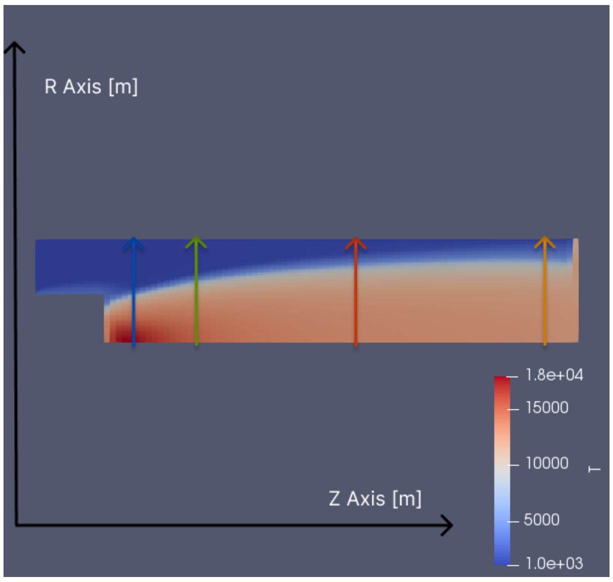

Figure 27: The temperature field of the $ADCsMU8335A250$ simulation run. Arrows indicates the extraction line of data for radial temperature profiles given at $z = (1.7, 5.7, 15.8, 27.8)$ $[mm]$.

The temperature field in Figure 27 shows a sudden extension of the hot temperature field at the exit but has the same maximum temperature as Figure 25.

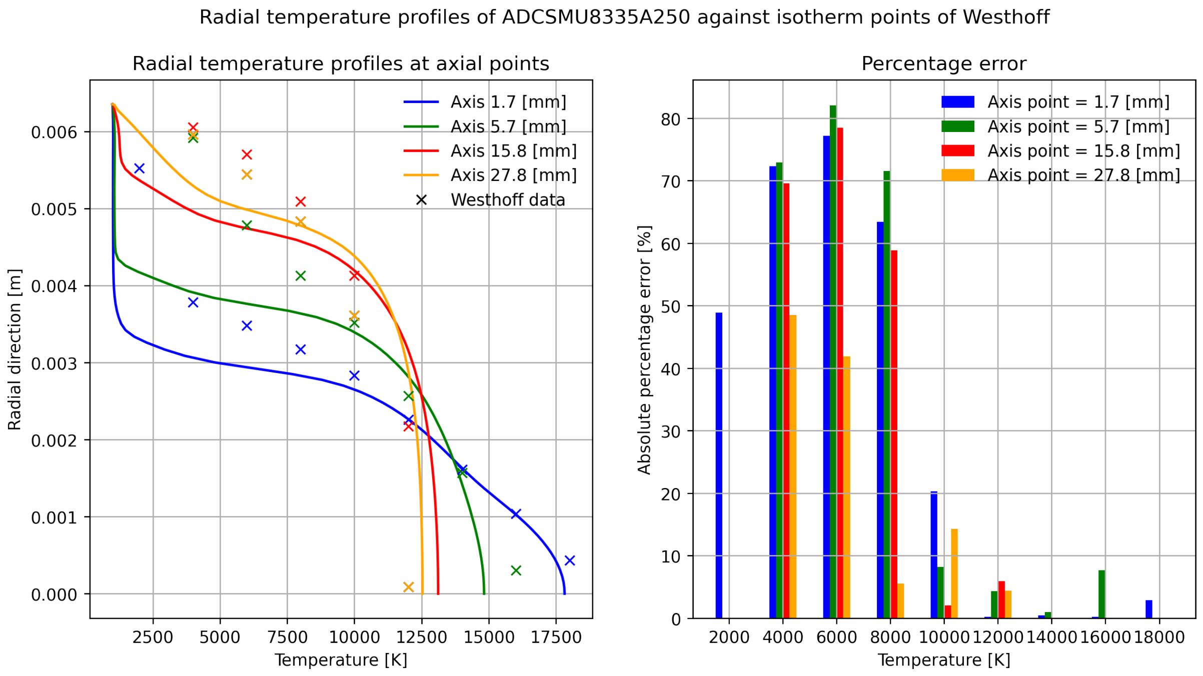

Figure 28: The radial temperature profile at given location at the z-axis of $ADCSMU8335A250$ compared against the isothermal data of Westhoff et al.[29], represented as crosses. The second graph shows the absolute percentage difference between a cross and the temperature.

The results in Figure 28 shows a high percentage error of above 40% for all isotherm points below 8000 $[K]$. Intersection of radial temperature profile occur at a radial distance below 4 $[mm]$, while crosses for 15.8 and 27.8 $[mm]$ intersect at around 6 $[mm]$.

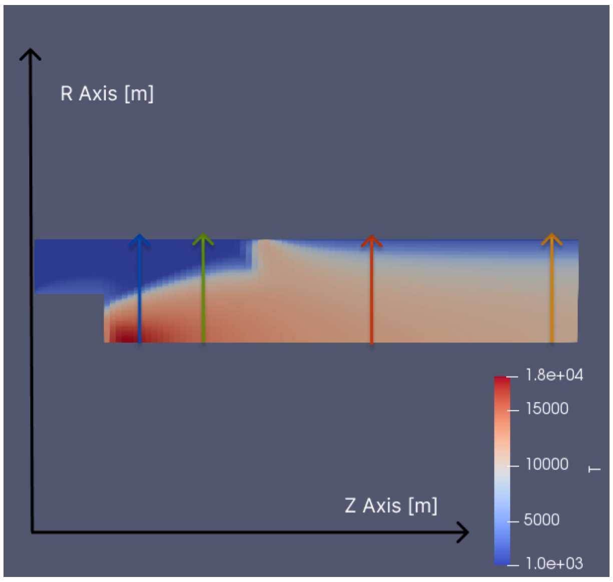

Figure 29: The temperature field of the $ADCSMU2498A250$ simulation run. Arrows indicates the extraction line of data for radial temperature profiles given at $z = (1.7, 5.7, 15.8, 27.8)$ $[mm]$.

The third run displayed in Figure 29 shows a temperature field where the temperature is above 8000 $[K]$ close to the anode surface between the axial points of 5.7 and 15.8 $[mm]$. A high elevated temperature close to the anode is present until the exit of the torch.

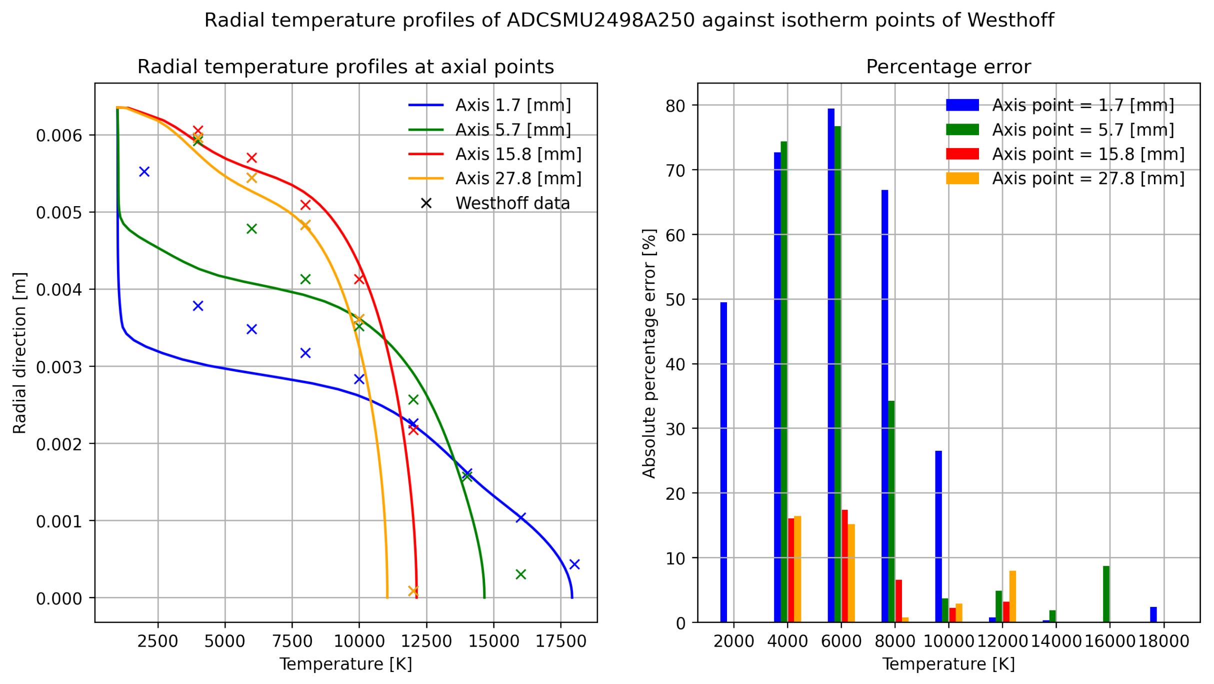

Figure 30: The radial temperature profile at given location at the z-axis of $ADCSMU2498A250$ compared against the isothermal data of Westhoff et al.[29], represented as crosses. The second graph shows the absolute percentage difference between a cross and the temperaturen.

The percentage error in Figure 30 shows an error below 20% for the axial points 15.8 and 27.8 $[mm]$ after the point of contact of the high temperature part of the arc with the anode. The axial points closer to the cathode show a much larger discrepancy between the simulation and benchmark values. The intersection of radial temperature profiles occurs at a radial distance below 4 $[mm]$ for the two radials closest to the cathode, and 6 $[mm]$ for the others, while crosses follow the same trend.

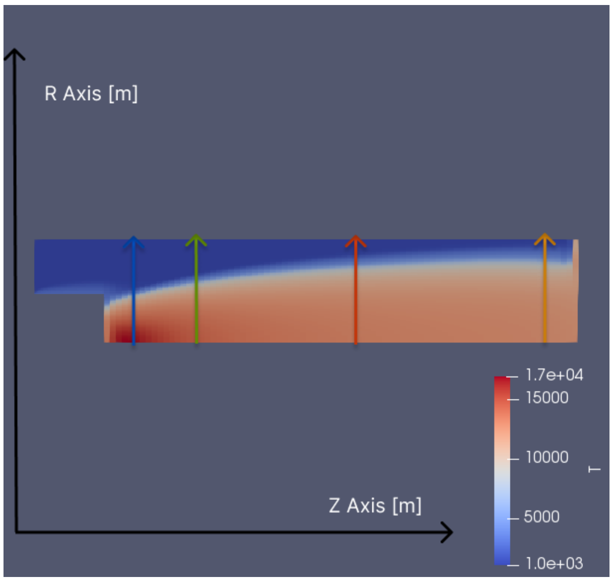

Figure 31: The temperature field of the $ABCSMU8335A250$ simulation run. Arrows indicates the extraction line of data for radial temperature profiles given at $z = (1.7, 5.7, 15.8, 27.8)$ $[mm]$.

The simulation $ABCSMU8335A250$ represented in Figure 31 shows a lower temperature profile, where the temperature does not surpass 17000 $[K]$, and a high temperature part of the flow touching the anode at the exit.

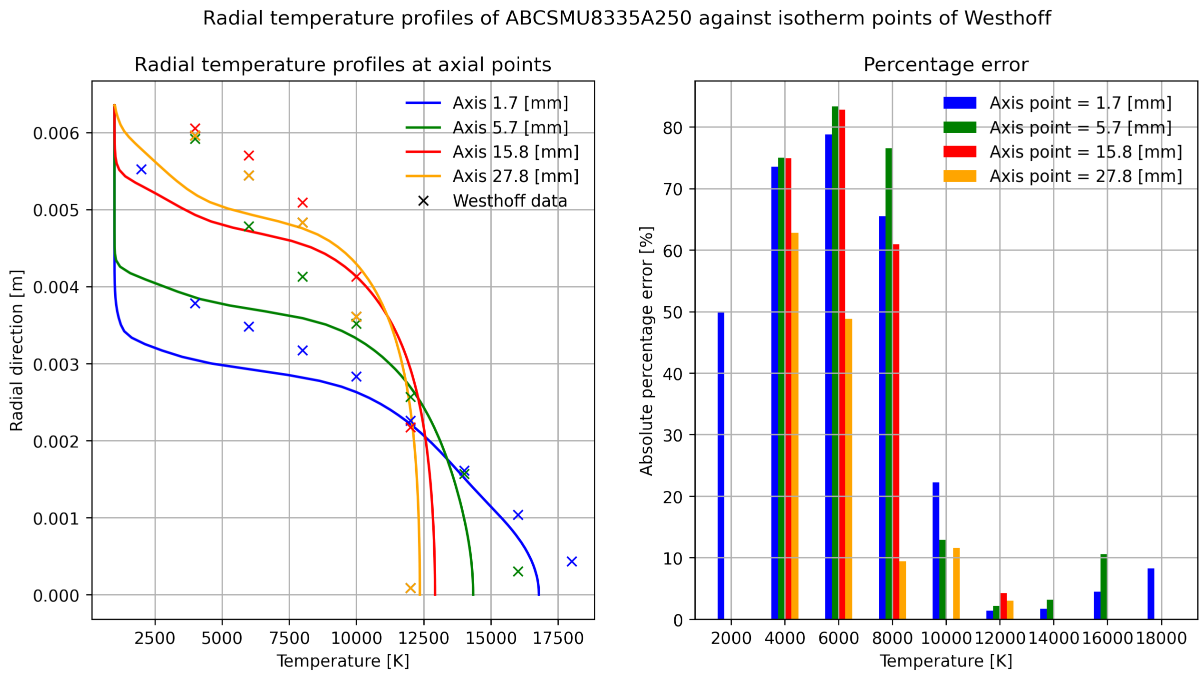

Figure 32: The radial temperature profile at given location at the z-axis of $ABCSMU8335A250$ compared against the isothermal data of Westhoff et al.[29], represented as crosses. The second graph shows the absolute percentage difference between a cross and the temperature.

The percentage error bars for the Bolous conductivity shown in Figure 32 also show a high percentage error at almost 50% of temperatures below 8000 $[K]$, and a high percentage error of 8% at 18000 $[K]$. The temperature profile shows a drastic temperature decrease for all axial points relative to the data of Westhoff et al.[29] shown as crosses. Intersection of radial temperature profile occur at a radial distance below 4 $[mm]$, while crosses for 15.8 and 27.8 $[mm]$ intersect at around 6 $[mm]$.

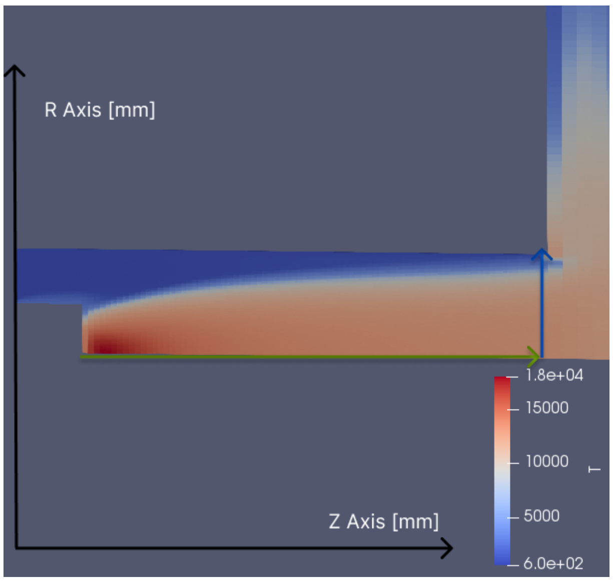

Figure 33: The temperature field of the $ADCBMU8335A250$ simulation run. Arrows indicates the extraction line of data for outlet and axial data as blue and green arrows. The model is zoomed in at the torch section.

The Figure 33 Show at which places data is retrieved for the figure below. The same method is used on all simulations to retrieve the data.

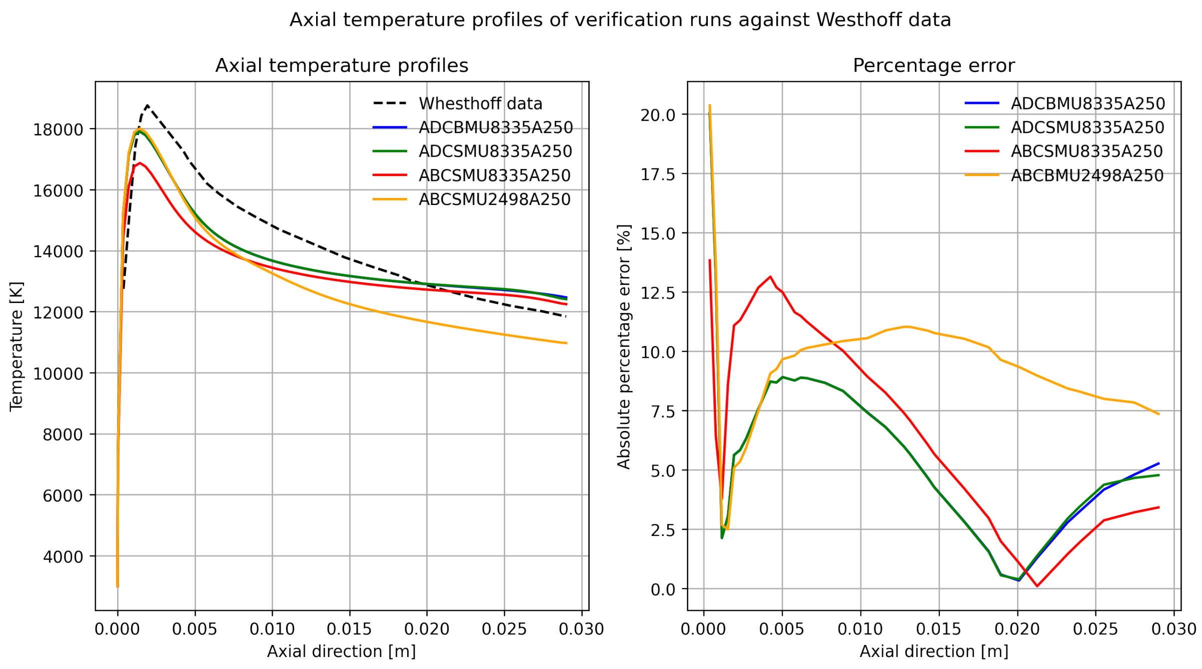

Figure 34: The axial temperature profile of the benchmark verification cases are displayed in the graph. Westhoff et al.[29] data is shown as a black line. The right graph indicate the percentage error.

The axial profiles show that the Westhoff et al.[29] data to be situated at a higher temperature for the axial length of the simulation domain until an intersection around 20 $[mm]$. Furthermore, a clear trend is seen where all simulation runs using 8.335 $[\frac{m}{s}]$ follow a similar trend with a less negative gradient of the data after 10 $[mm]$, while the Whesthoff et al.[29] data and 2.498 $[\frac{m}{s}]$ show a similar continuously decreasing trend.

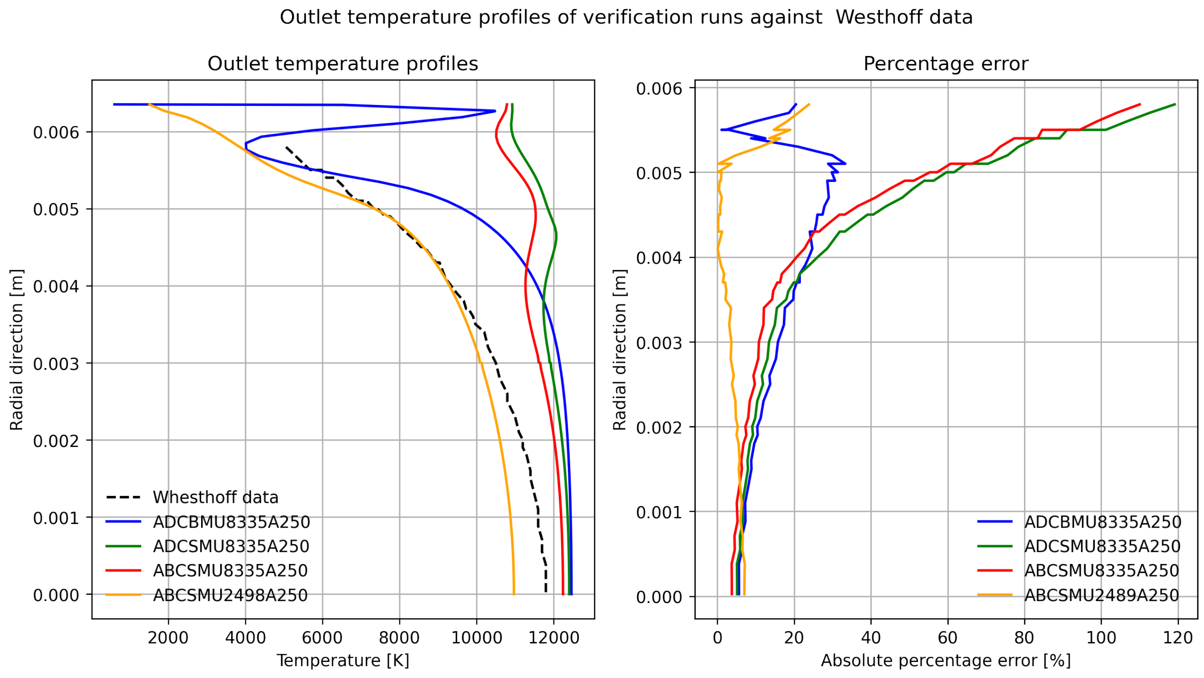

Figure 35: The outlet temperature profile of the benchmark verification cases are displayed in the graph, with the radial direction as the y-axis. Westhoff et al.[29] data is shown as a black line. The right graph indicate the percentage error.

A clear trend is shown where the difference between the temperature profiles in Figure 35 increases at higher radial values. Run $ADCBMU8335A250$ show an irregular fluctuating behavior, whereas runs using a small mesh at a velocity of 8.335 $[m/s]$ generally remain above 10000 $[K]$. This is in contrast to the last runs, which portray a similar decreasing behavior.

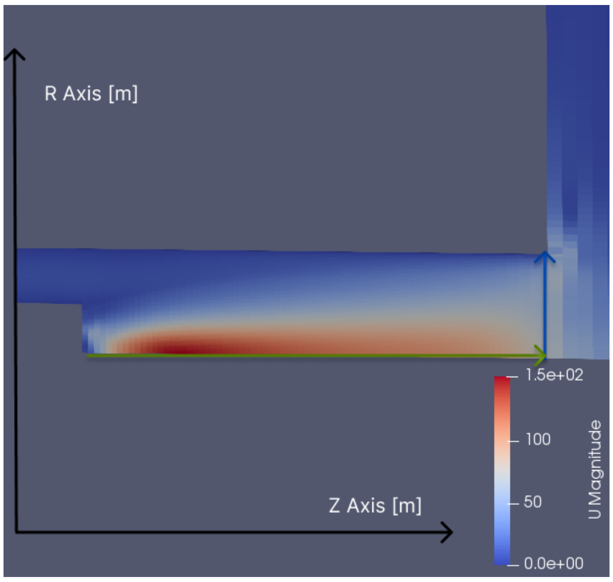

Figure 36: The velocity field of the $ADCBMU8335A250$ simulation run. Arrows indicates the extraction line of data for Outlet and axial data. The model is zoomed in at the torch section.

The velocity field is displayed in Figure 36, where a high velocity zone is visible close to the cathode and along the axial. All other simulation runs have their data extracted, as the arrow shows.

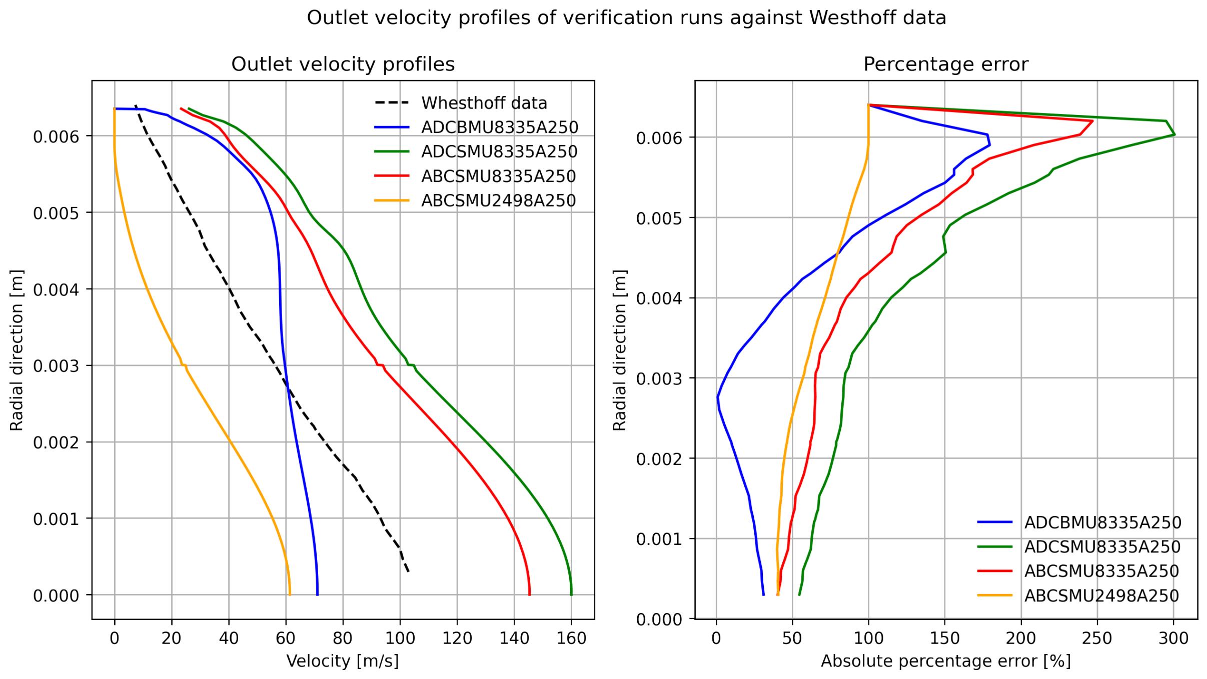

Figure 37: The radial velocity profile at the outlet displayed with the radial values at the y-axis.

Except for simulation run $ADCBMU8335A250U$ in Figure 37, all runs show the same trend as their gradient is similar.

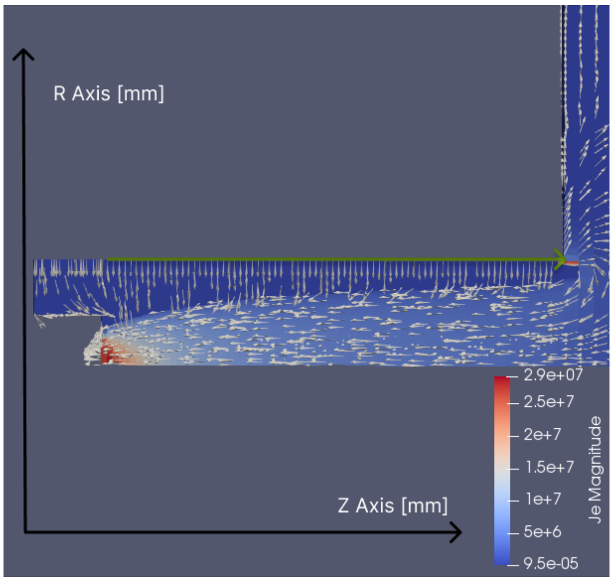

Figure 38: The current density field of the $ADCBMU8335A250$ simulation run. Arrows indicates the extraction line of data for anode data. The model is zoomed in at the torch section.

The current in Figure 38 clearly shows a higher value close to the cathode and the current flowing from the anode to the cathode. It is worth noting that the current curls around the torch exit to the outermost point of the anode, creating a large current density at this position.

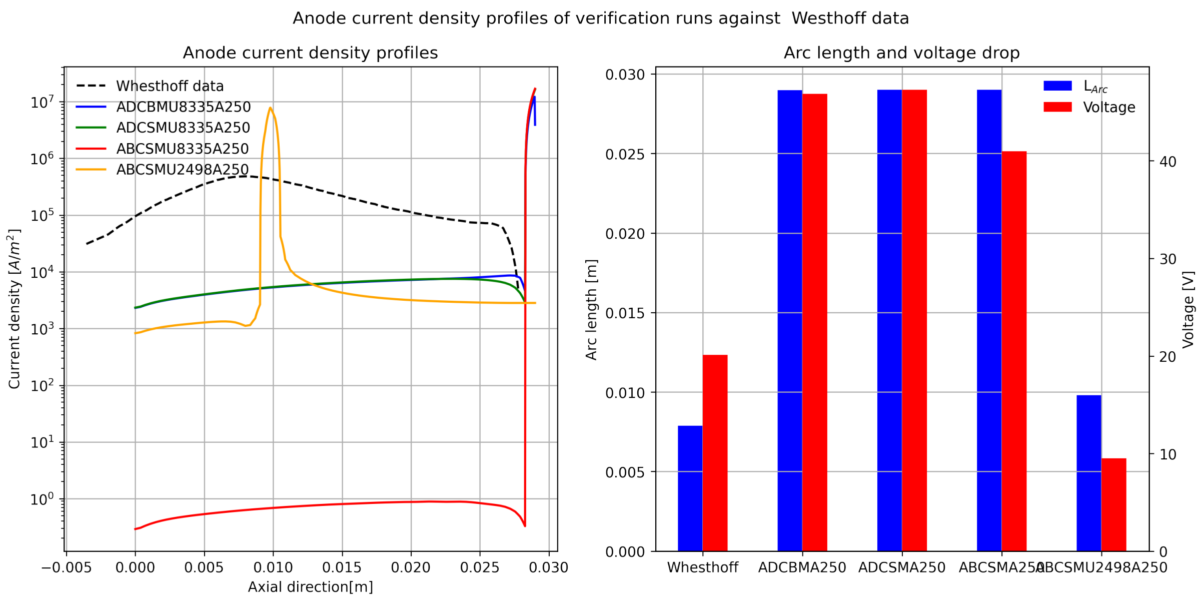

Figure 39: The anode current density profile at the outlet displayed with the axial direction at the x-axis. The second graph shows the measured arc length of each, and the maximum voltage drop in electric potential field. A logarithmic scale is used

All simulation runs show a clear peak around 10^7 $[\frac{A}{m^2}]$ in the current density profiles shown in Figure 39, with a large difference between the arc attachment at the anode, and the rest of the profile. The profile from Westhoff et al.[29] shows a current profile that is more evenly distributed along the anode and with a peak current density below 10^6 $[\frac{A}{m^2}]$. The arc length stretches from the whole domain for simulation runs using a velocity of 8.335 $[\frac{m}{s}]$, with an arc length below 10 $[mm]$ for the other runs. Voltage drop follows the trend of arc length as shown in the second graph of Figure 39.

11.2 Argon and hydrogen comparison study

The simulation data below compares argon and hydrogen at the same applied current and velocity and mass flow to show the differences between argon and hydrogen plasma.

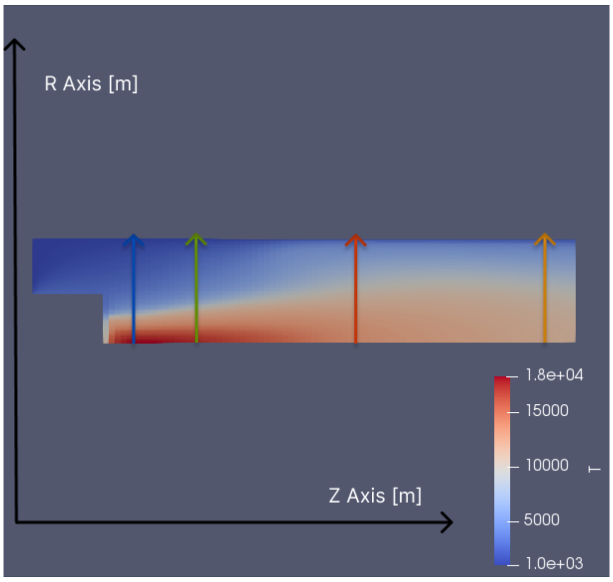

Figure 40: The temperature field of the $HBCSMU8335A250$ simulation run. Arrows indicates the extraction line of data for radial temperature profiles given at $z = (1.7, 5.7, 15.8, 27.8)$ $[mm]$.

The Figure 40 shows the hydrogen plasma's temperature. A high temperature region develops close to the cathode. The field's downstream temperature profile and the boundary between arc and external environment are quite diffuse. The temperature field is also more diffusively attached to the anode wall.

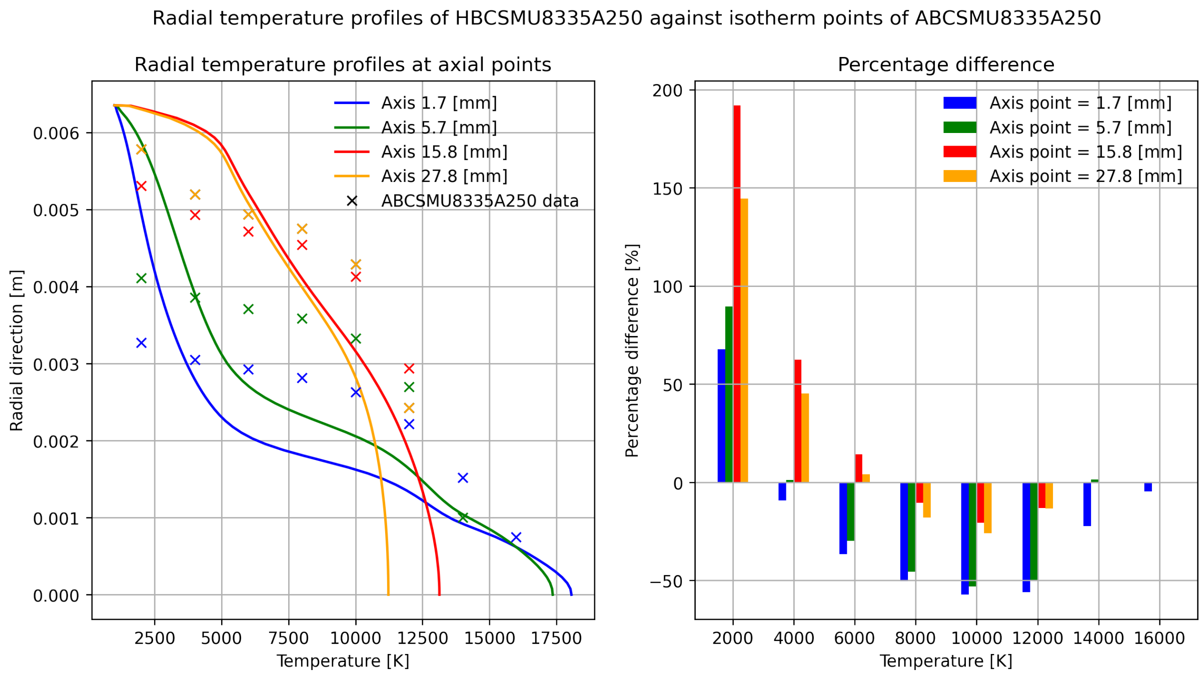

Figure 41: The radial temperature profile at given location at the z-axis of $HBCSMU8335A250$ compared against the isothermal points of $ABCSMU8335A250$, represented as crosses. The second graph shows the percentage difference between hydrogen and isothermal points of argon.

The graph shows that the radial temperature profiles of hydrogen drop below that of run $ABCSMU8335A250$ from the radial value of all profiles between 1 and 3 $[mm]$, while after this point, the hydrogen temperature more slowly approaches than the argon ones, especially farther downstream in the geometry.

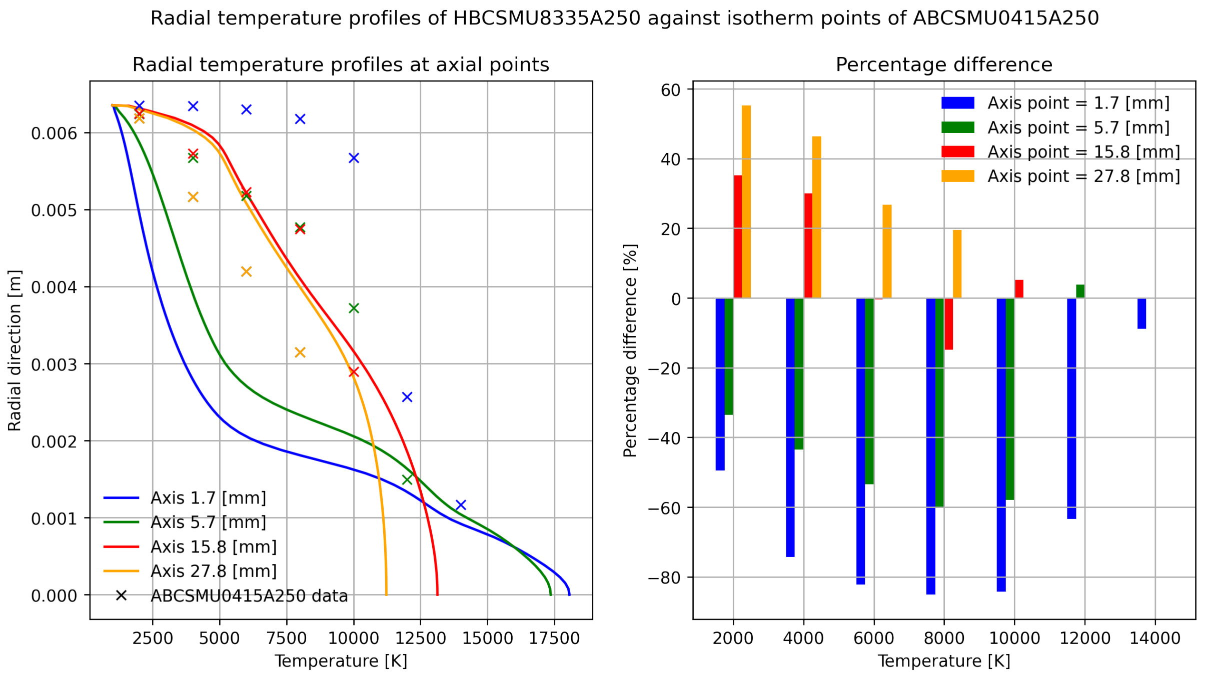

Figure 42: The radial temperature profile at given location at the z-axis of $HBCSMU8335A250$ compared against the isothermal data of $ABCSMU0415A25$, represented as crosses. The second graph shows the percentage difference between isotherm values of argon and the temperature of hydrogen.

The temperature profiles at the same mass flow of hydrogen and argon show different trends for different axial points. The two closest to the cathode show the hydrogen plasma cooling fast along the radial direction, while the argon plasma maintains a higher temperature. On the other hand, downstream profiles show a higher cooling rate along the radial for argon.

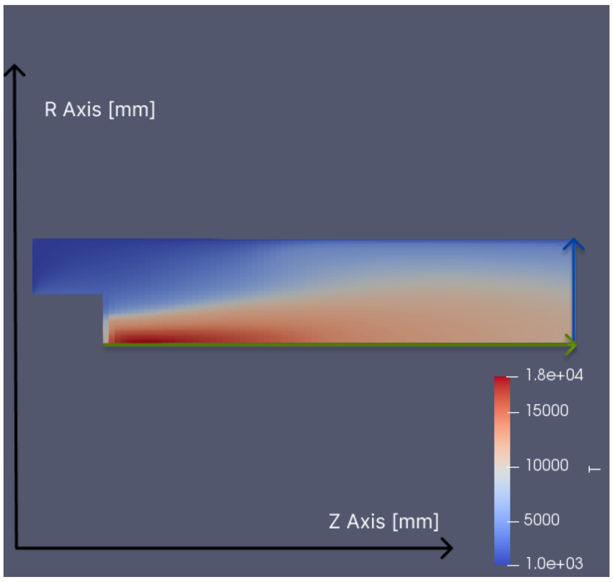

Figure 43: The temperature field of the $HBCSMU8335A250$ simulation run. Arrows indicates the extraction line of data for axial and outlet data.

Figure 43 demonstrates where the data is extracted from. All data for outlet and axial values are extracted in the same manner.

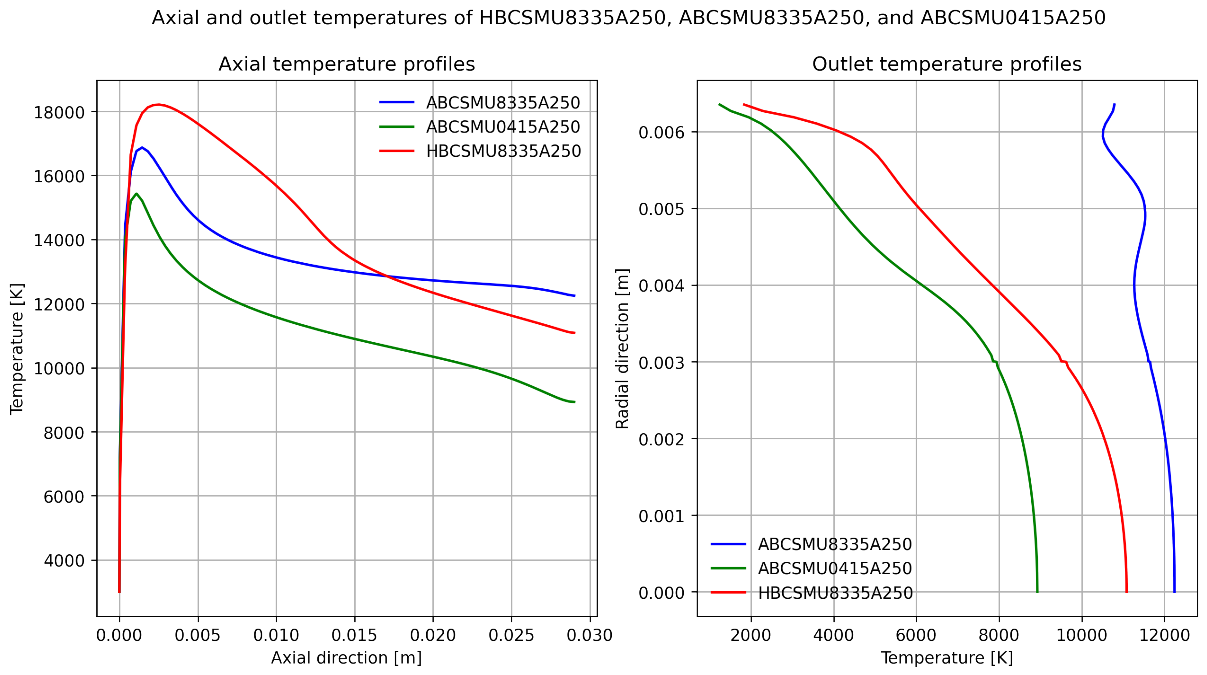

Figure 44: Two graphs displaying the axial temperatures and the outlet temperatures.

As seen from Figure 44, hydrogen has the highest temperature close to the cathode, while the same mass flow of argon has the lowest along the axial. The runs at the same mass flow, $ABCSMU0415A25$ and $HBCSMU8335A250$, display the temperature trend for the outlet temperature profiles.

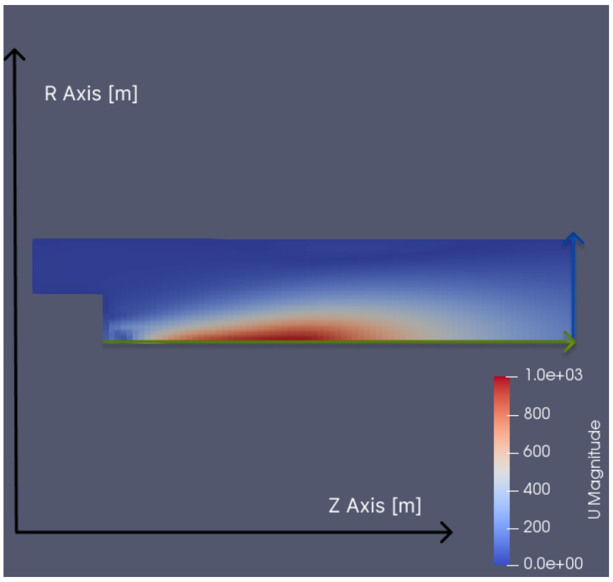

Figure 45: The velcoity field of the $HBCSMU8335A250$ simulation run. Arrows indicates the extraction line of data for axial and outlet data.

The Figure 45 shows a large velocity increase close to the symmetry axis. The velocity reaches a magnitude of around 1000 $[\frac{m}{s}]$ around its peak. All velocity values are gathered from the same lines as shown in Figure 45.

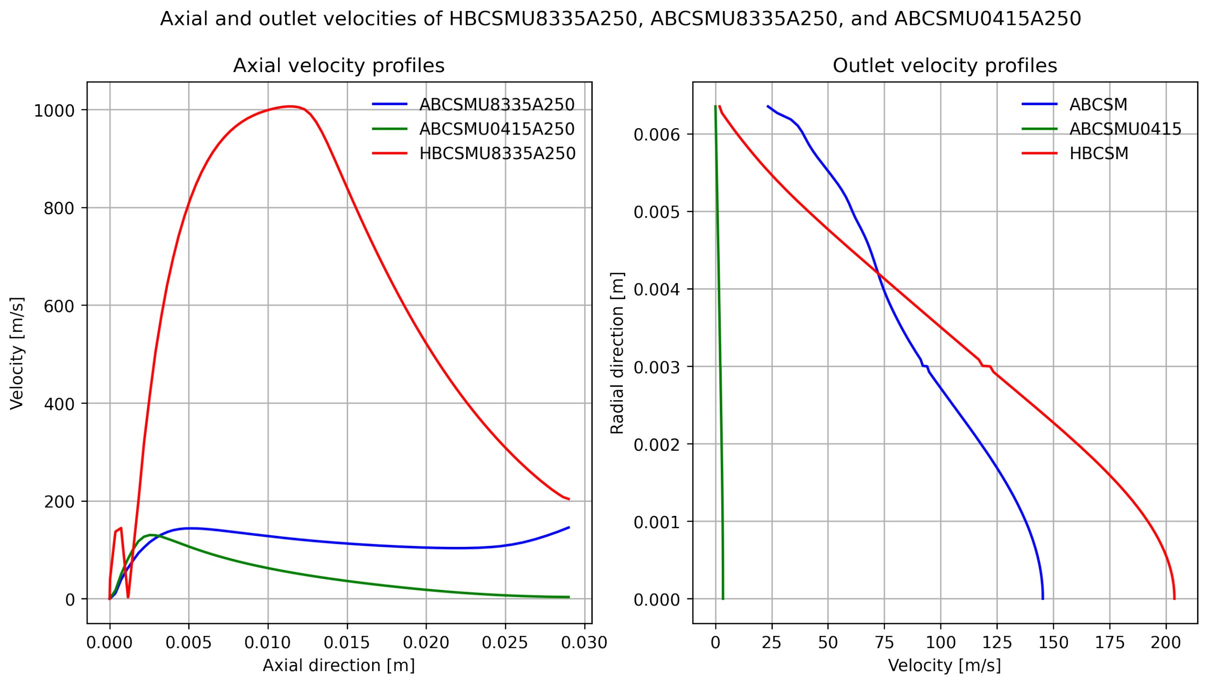

Figure 46: Two graphs displaying the axial and outlet temperatures velocity, and the percentage difference.

Hydrogen shows the highest axial velocity as seen from Figure 46, where the same mass flow for argon shows the lowest velocity. The same is true for the outlet velocity, where the low velocity argon run has an outlet velocity around 95% lower than the other runs in the graph.

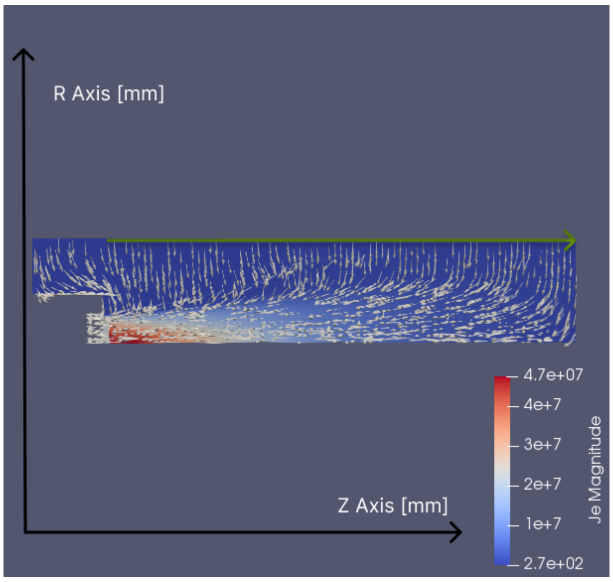

Figure 47: The anode current field of the $HBCSMU8335A250$ simulation run. Arrows indicates the extraction line of data for anode data.

The anode current density field in Figure 47 shows a clear current density concentration along the center of the cathode surface. The current along the anode is relatively evenly distributed along the anode surface downstream in Figure 47.

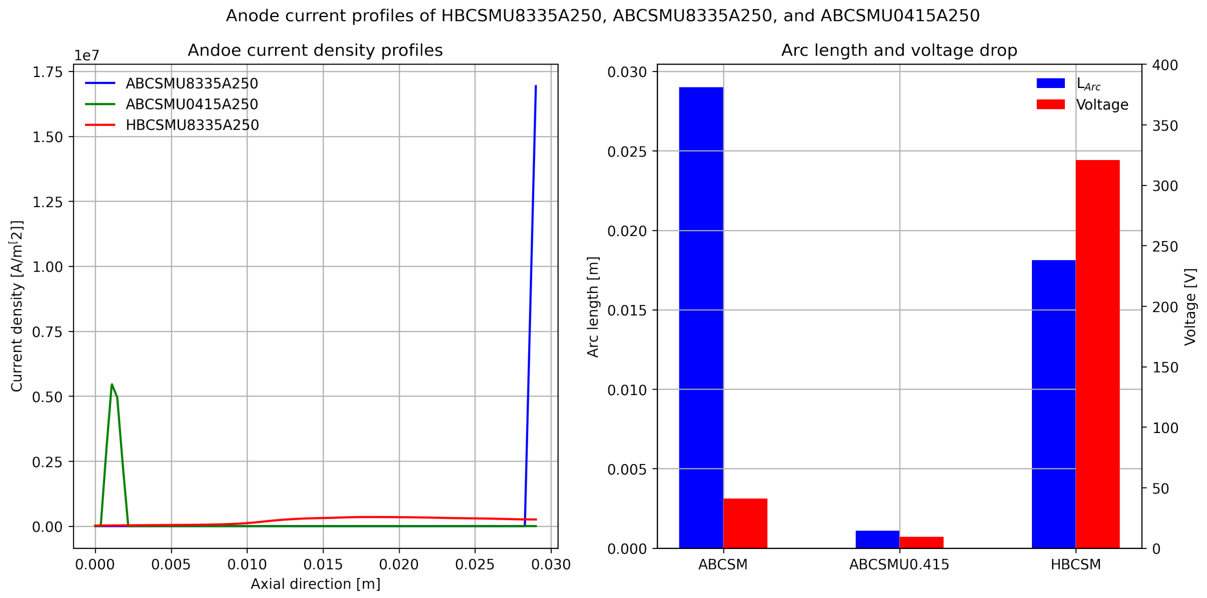

Figure 48: The anode current density profile at the anode displayed with the axial direction at the x-axis. The second graph shows the measured arc length of each, and the maximum voltage drop in electric potential field.

The profile of the anode current density is more evenly distributed for hydrogen. Sharper peaks are shown for argon. The arc lengths for the same mass flow show a large difference where the arc length of argon is 94% lower than hydrogen. The voltage drop for hydrogen is above six times larger than that of both argon cases.

11.3 Argon parameter study

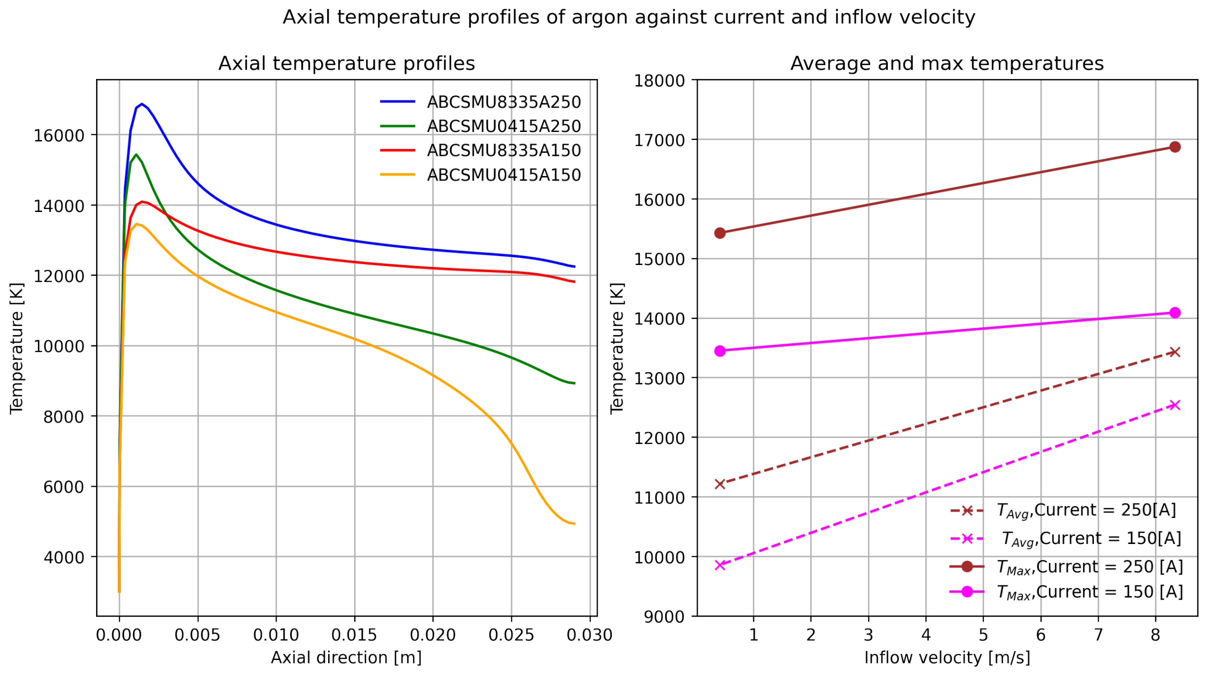

Figure 49: Axial temperatures shown for 4 simulation runs, where the current and inflow velocity is varied. The second graph shows the average and max temperature values as functions of inflow velocity at different applied currents. Brown indicate 250 $[A]$, and pink 150 $[A]$. Circles and a dotted line and excess separate the max for the average values.

Maximum axial temperatures are higher for higher currents regardless of inflow velocity. However, the average temperature of the 150 $[A]$ at high inflow velocity runs surpasses the 250 $[A]$ run at low inflow velocity. The axial development trend shows that simulation runs with the same inflow share the same axial development towards the outflow.

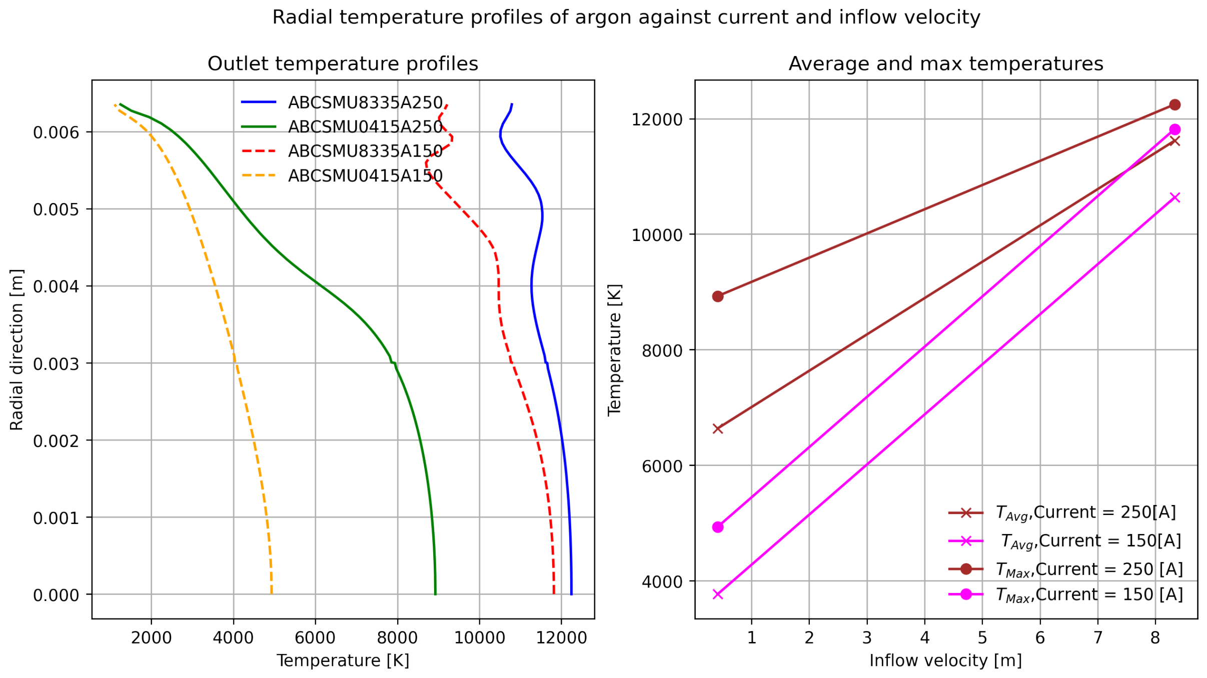

Figure 50: Outlet temperatures are shown for four simulation runs, where the current and inflow velocity is varied. The second graph shows the average and max temperature values as functions of inflow velocity at different applied currents. Brown indicate 250 $[A]$, and pink 150 $[A]$. Circles and a dotted line and crosses separate the max for the average values.

Figure 50 shows that the inflow velocity has a large effect on the temperature values, as both the characteristic radial development of the temperature and the average and maximum temperatures show a clear trend for the different Inflow velocities.

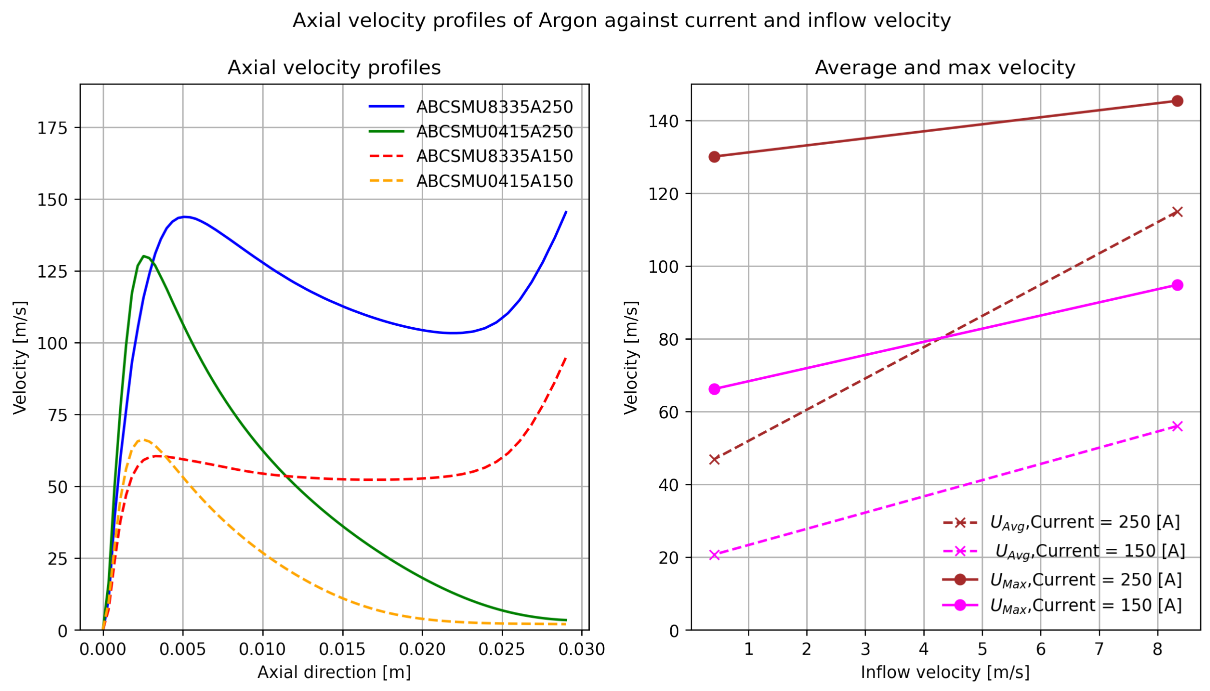

Figure 51: Axial velocity shown for 4 simulation runs, where the current and inflow velocity is varied. The second graph shows the average and max velcoity values as functions of inflow velocity at different applied currents. Brown indicate 250$[A]$, and pink 150$[A]$. Circles and a dotted line and crosses separate the max for the average values.

The current is shown as the main influence on the velocity achieved as a higher current gives above a 100% increase in maximum velocity close to the cathode. However, the inflow velocity heavily influences the characteristic development as the curves at the same inflow show similar development along the axial direction.

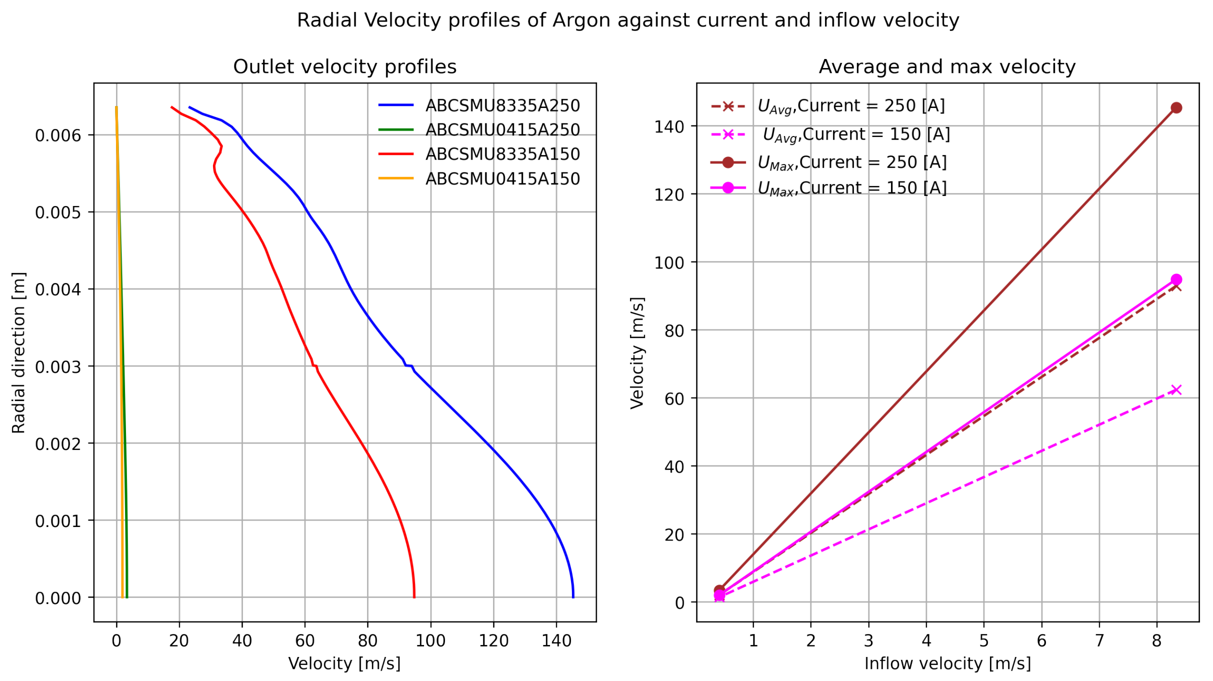

Figure 52: Outlet velocity shown for 4 simulation runs, where the current and inflow velocity is varied. The second graph shows the average and max velocity values as functions of inflow velocity at different applied currents. Brown indicate 250 $[A]$, and pink 150 $[A]$. Circles and a dotted line and crosses separate the max for the average values.

For Figure 52 the inflow velocity seems like the main driving factor, as the outflow velocity rapidly increases when inflow is increased.

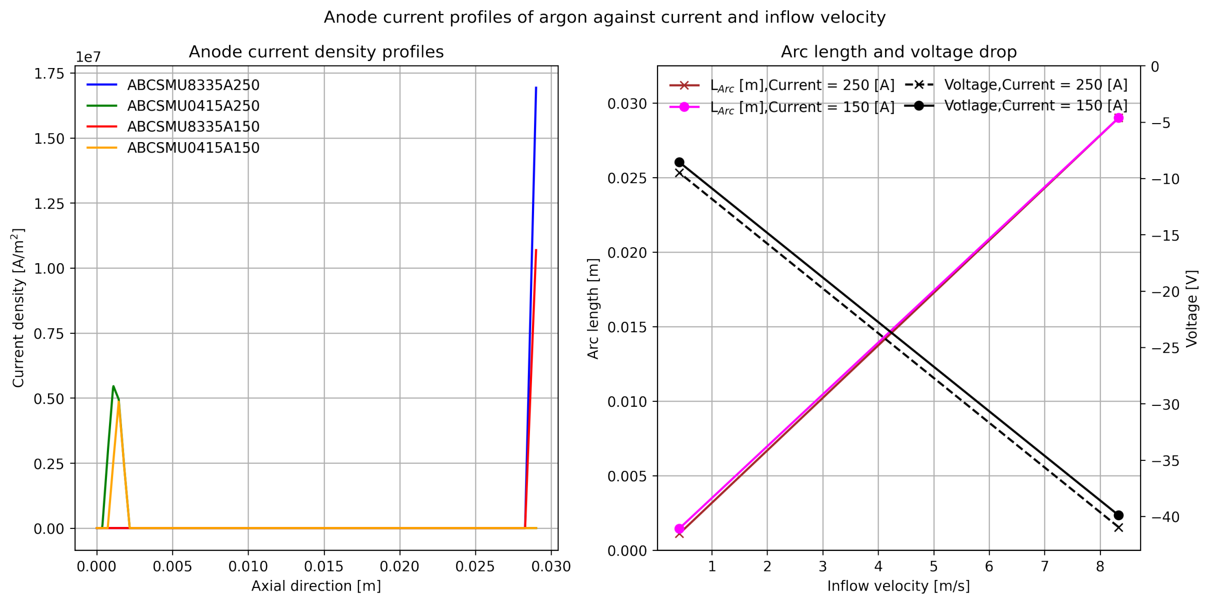

Figure 53: The anode current shown for four simulation runs, where the current and inflow velocity is varied. The second graph shows the voltage and arc length values as functions of inflow velocity at different applied currents. Brown indicate 250 $[A]$, and pink 150$[A]$. Circles and a dotted line and crosses separate the max for the average values.

For the current profile shown in figure 53 the inflow velocity is shown to create a large difference, as the peak of velocities of 8.335 $[\frac{m}{s}]$ are a lot higher, with around 100% to 200% increase. The same is seen for the arc length and the voltage.

11.4 Hydrogen parameter study

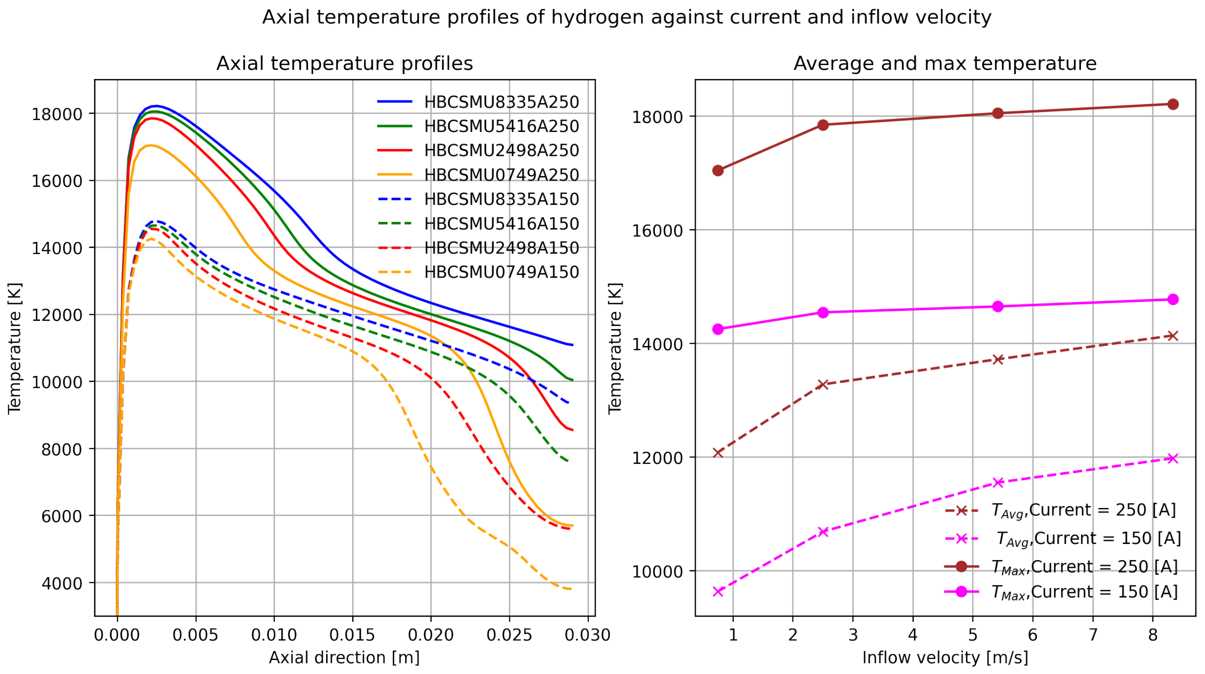

Figure 54: The axial temperature profile shown for eight simulation runs, where the current and inflow velocity is varied. Dotted lines represent a current of 150 $[A]$. The second graph shows the average and max temperature values as functions of inflow velocity at different applied currents. Brown indicate 250 $[A]$, and pink 150 $[A]$. Circles and a dotted line and crosses separate the max for the average values.

The result from the axial temperature show from Figure 54 that the maximum and average temperature is strongly dependent on the current applied, separating the curves into two distinct groups, which is also shown for the brown and pink curves of the max and average temperature. The inflow velocity increases the temperature but to a lower extent.

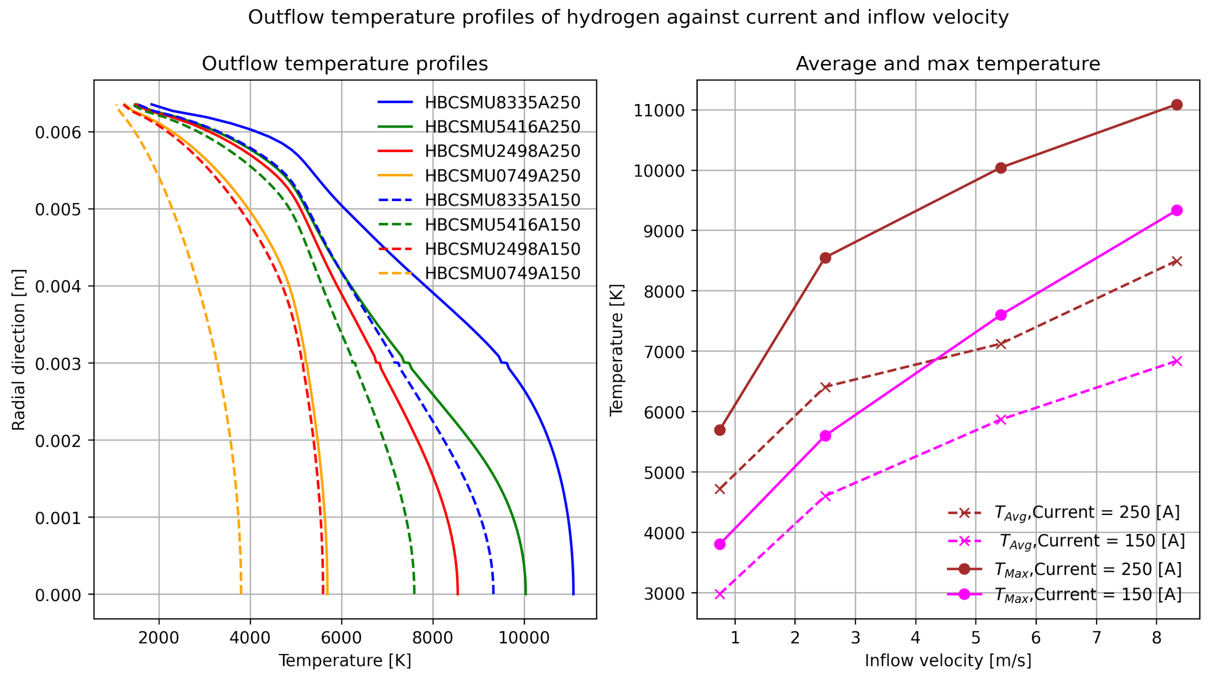

Figure 55: The outflow temperature profile shown for eight simulation runs, where the current and inflow velocity is varied. Dotted lines represent a current of 150 $[A]$. The second graph shows the average and max temperature values as functions of inflow velocity at different applied currents. Brown indicate 250 $[A]$, and pink 150 $[A]$. Circles and a dotted line and crosses separate the max for the average values.

The outflow temperature shown in Figure 55 shows a similar dependence on both current and inflow velocity, where both cause an increase in temperature.

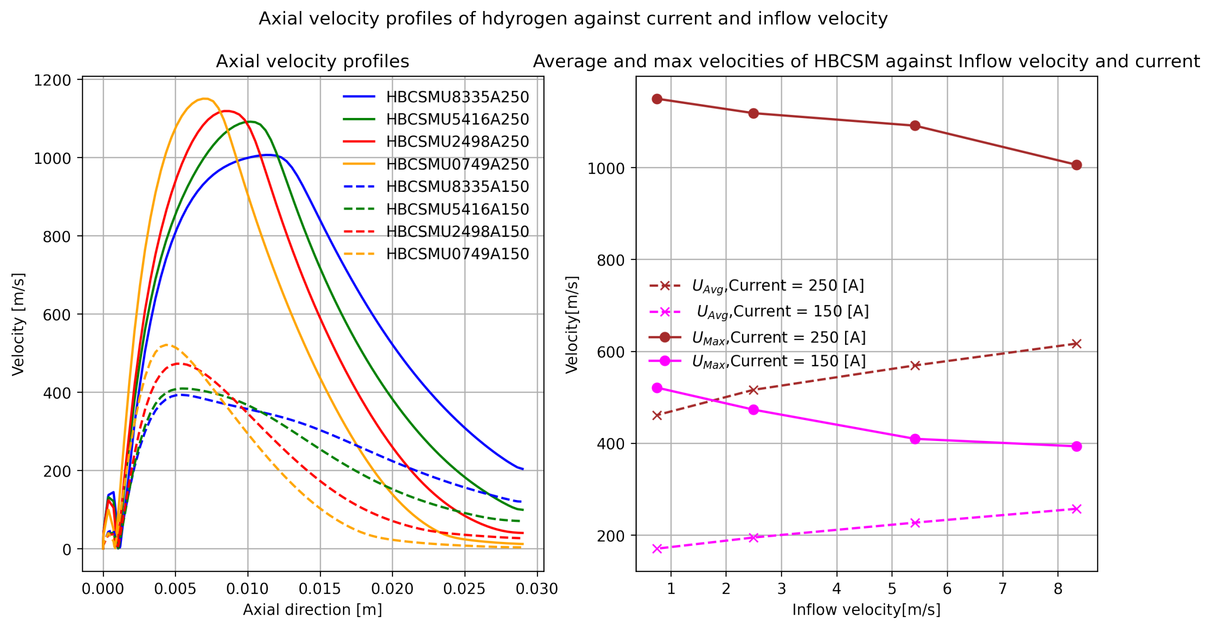

Figure 56: The axial velocity profile shown for eight simulation runs, where the current and inflow velocity is varied. Dotted lines represent a current of 150 $[A]$. The second graph shows the average and max velocity values as functions of inflow velocity at different applied currents. Brown indicate 250 $[A]$, and pink 150 $[A]$. Circles and a dotted line and crosses separate the max for the average values.

The current creates a large difference between the axial velocity profiles, as shown in Figure 56, and an increase in current creates a large difference in the velocity. Both currents show a decrease in maximum velocity as the inflow velocity is decreased.

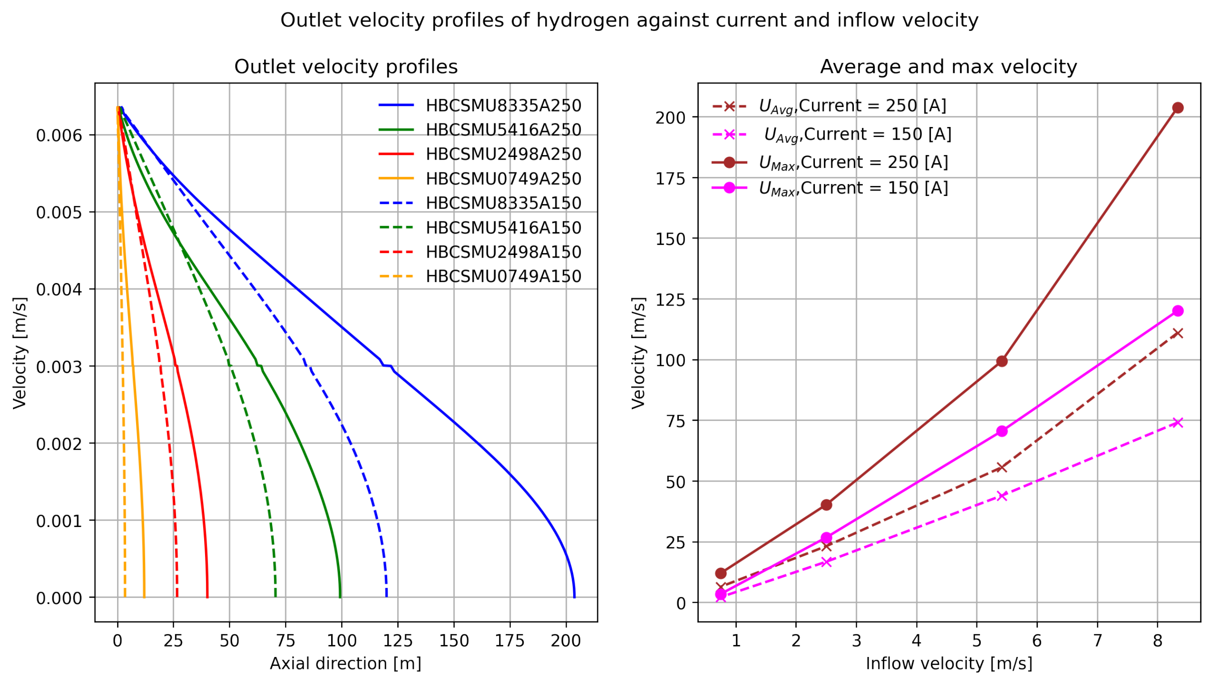

Figure 57: The outlet velocity profile shown for eight simulation runs, where the current and inflow velocity is varied. Dotted lines represent a current of 150 $[A]$. The second graph shows the average and max velocity values as functions of inflow velocity at different applied currents. Brown indicate 250 $[A]$, and pink 150 $[A]$. Circles and a dotted line and crosses separate the max for the average values.

The outlet velocity is shown to increase for both an increase in inflow velocity and current, with a larger dependence on the inflow velocity. All profiles approach zero as they approach the no-slip boundary wall.

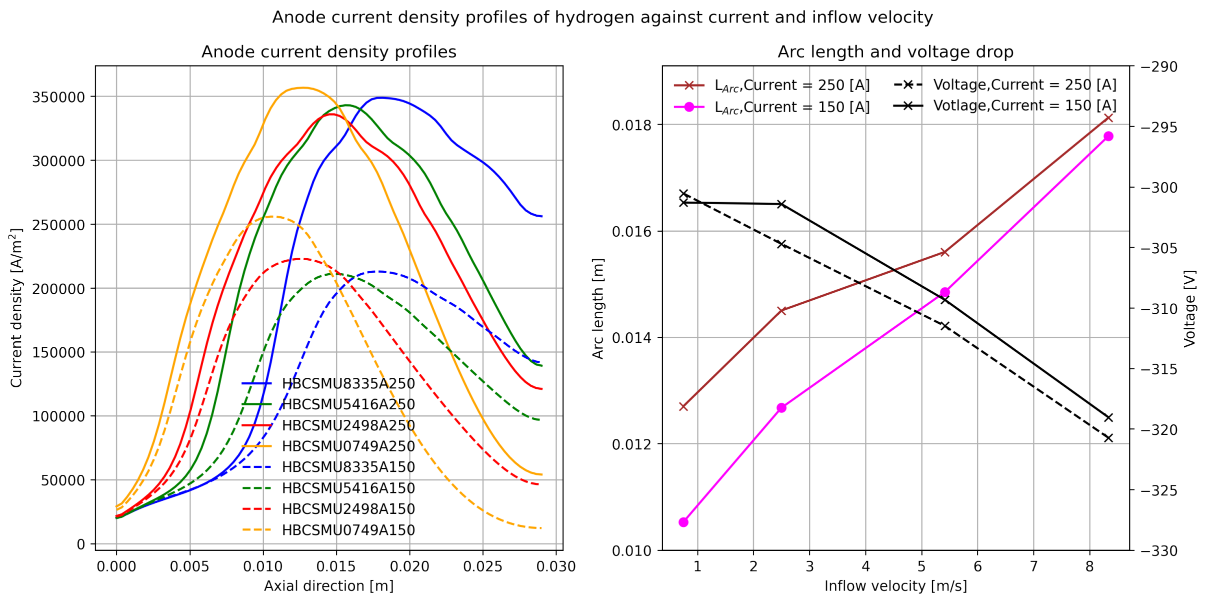

Figure 58: The anode current density profile is shown for eight simulation runs, where the current and inflow velocity is varied. Dotted lines represent a current of 150 $[A]$. The second graph shows the voltage drop and arc length as functions of inflow velocity at different applied currents. Brown indicate 250 $[A]$, and pink 150 $[A]$. Circles and a dotted line, and crosses separate the voltage drop and arc length.

The current applied drastically increases the current density at the anode. In contrast, the inflow velocity has a non-linear effect on this max value but increases the arc length and the voltage drop. Increasing current increases the arc length and slightly increases the absolute magnitude of the voltage drop.

11.5 Residuals

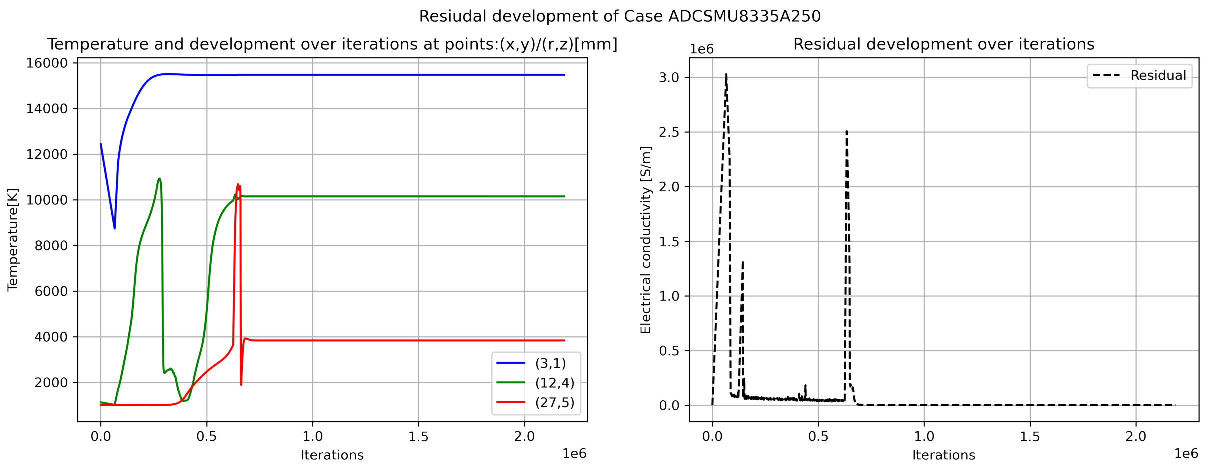

Figure 59: The left graph shows temperature values extracted from the simulation over multiple iterations. The right shows the residual level

The Figure 59 shows temperature values that stabilize over time, as well as a residual which drops to zero.

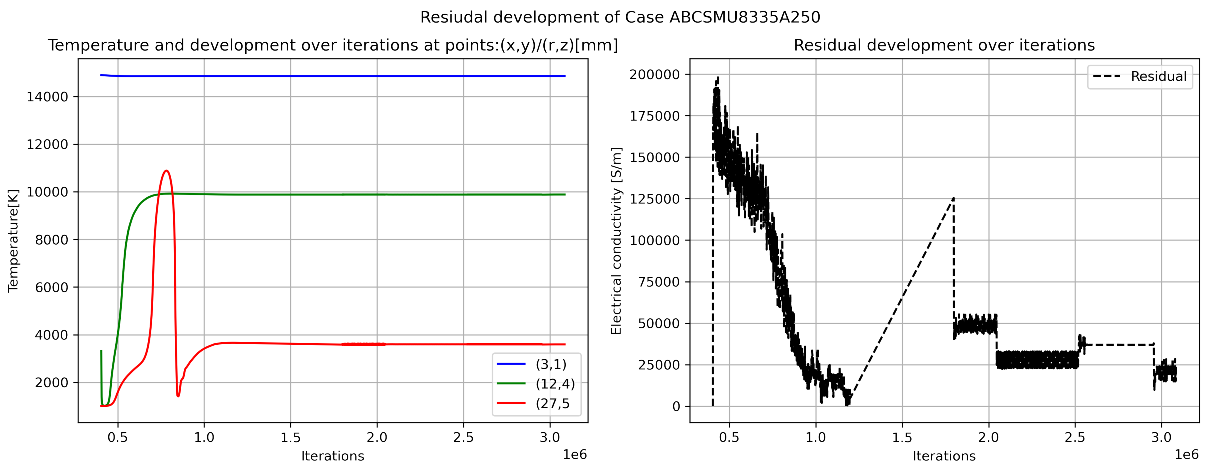

Figure 60: The left graph shows temperature values extracted from the simulation over multiple iterations. The right shows the residual level

Here for Figure 60 the temperature values do stabilize. However, the residual does not reach zero but stays at $\frac{1}{8}$ of the max value.

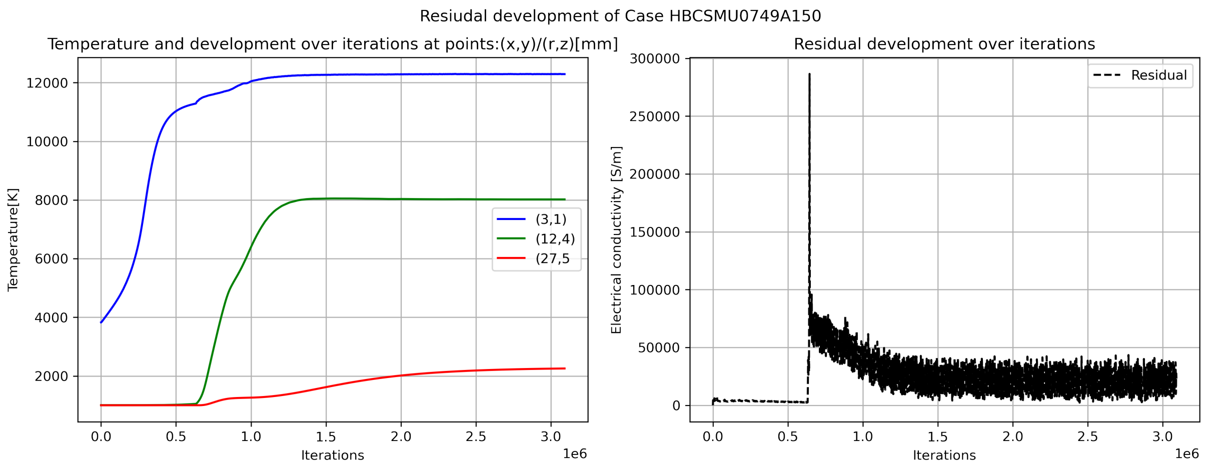

Figure 61: The left graph shows temperature values extracted from the simulation over multiple iterations. The right shows the residual level

Figure 61 shows stabilizing temperature where point (27,5) is still increasing slowly. The residual does not drop below $\frac{1}{12}$ for the given simulation.

12 Discussion

The discussion will aim to verify or not verify the thermal plasma solver procedure developed during this project, as well as the hydrogen plasma results. This is done by comparing the hydrogen-argon comparison study and the interaction on temperature, velocity, and arc length of different parameters, with results in the literature described in section 7. The insight from the discussion is then used to verify if and why there might be a difference between the chosen benchmark case of Westhoff et al.[29], and the results obtained in this project. This will give a verdict on the credibility of the results produced by the solver procedure in this project. To do this, the discussion will first look at the implemented source terms and physical features of the argon verification simulations and see if it matches the physics described in theory. Then hydrogen is compared to argon to see if this also corresponds to the literature. Next, the parameter study of both argon and hydrogen is analyzed to see if their response to the parameter study is equal to known behavior. This knowledge is then used to analyze the verification results and give a verdict on why the results are different from the work of Westhoff et al.[29].

12.1 Thermal plasma features and source terms

It is expected that an arc forms with a heated area close to the cathode and a more or less clear distinction between the arc body and the surrounding gas as shown in Figure 3 and described by Trelles et al.[10]. This is seen in Figure 29 from the results, where an arc body is clearly visible, and a heated zone has developed close to the anode. In addition, from the temperature field and the anode profiles in Figure 39 it is clear that a current is established between cathode and anode. As the temperature follows this current path, it is possible to establish that the Joule source term has been implemented to some extent according to the source term in Equation 24, and is heating the gas on its way from the inlet to the outlet.

The velocity field in Figure 36 show an increased velocity field close to the axial symmetry line with a peak close to the cathode. This velocity increase indicates an increased pressure region from a pinch force and temperature driven decrease of density according to the Continuity equation, given in Equation 31 and the Maecker effect stated in Equation 55. The increased pressure field can be seen in Figure 24. The pinch force and the heat increase are indications of the combined effect of the Lorentz force and Joule heating source, as the Lorentz force is coupled through the Momentum equation, given in Equation 26. In addition, the evidence of a steady state anode arc attachment, as seen in the temperature field in Figure 29 and the anode current profile of Figure 39, shows that an equilibrium between the Lorentz force and convective drag according to Equation 52 might exists. This result highlights that the applied Lorentz force or some similar effect affects the arc and creates this defining feature of non-transferred plasma torches.

12.2 Cathode current

The current density along the cathode for Figure 23 of argon shows a current density average which is slightly lower then what is expected from Figure 7 by a value of 0.315 $[\frac{A}{m^2}]$]. This could contribute to a loss of power in the simulation runs, which means that the procedure listed in Listing 8, which approximates the non-LTE effect by giving the boundary the same conductivity as the cell facing it, is not a good approximation. However, the total current could be the same as the area is slightly large for Figure 23 has not been tested. Nevertheless, as the area is larger, the arc radius is also longer. According to Ramachandran et al.[28] an increased arc radius will lower the torch power. Thus the result might nevertheless be a reduction in the temperature.

12.3 residual and divergence

The residual result for all cases shows a similar trend for the combination of conductivity, flow, and mesh sizes. For example, small mesh and Devoto conductivity gave a residual, which became zero after around 600000 iterations. As seen from Figure 59, the temperature values extracted from the given point in the mesh show stable values. This was also true for argon at a velocity of 0.415 $[\frac{m}{s}]$ and 2.498 $[\frac{A}{m^2}]$ using the Boulous conductivity, regardless of the current applied.

However, for big mesh simulations and the combination of a velocity of 8.335 $[\frac{m}{s}]$ and Boulos conductivity, the residual followed the trend shown in Figure 59, only dropping to $\frac{1}{8}$ of the value. On the other hand, the temperature extracted from points in the mesh remains within the scale of the graph, showing that the conductivity field remains fairly constant. However, as the residuals have not reached zero, the value of the information in these simulations must be questioned.

The same is true for all hydrogen simulations, as seen in Figure 61, as the oscillating residual only drop to $\frac{1}{12}$ of the original value. However, the temperature and conductivity field is seen stabilizing as the simulation run reaches 3 million iterations. Nevertheless, as the residual does not drop towards zero, the simulation data has to be questioned in these simulations. The verdict is that the simulations have to be run to a greater value than 3 million iterations. However, that was the limit of this work because of time constraints. In addition, another residual calculation method or check to stop the simulation has to be implemented, as the residual does not, to a great extent, reflect the convergence of the data.

12.4 Polynomial implementation

The polynomial fitting for conductivity and the NEC model is relatively good as a twelfth degree polynomial is used to approximate the data values. However, the radiation values for argon are approximated from a different paper than what Westhoff et al.[29] used for temperature values below 10000 $[K]$. This might affect the accuracy of the temperature values of argon below this threshold.

The argon and hydrogen are only approximated by a seventh degree polynomial, which has caused significant approximation errors as seen in Figure 9 in section 9.6. For example, the peaks in heat capacity of both argon and hydrogen are greatly lowered. The peak for hydrogen at lower temperatures is cut by around 50% and starts at a 1000 $[K]$, which dramatically alters its behavior at low temperature values. The same is true for thermal conductivity, where the first peak is around 60% of its actual value, significantly reducing the conductivity of hydrogen for low temperatures. The conclusion is that seventh degree polynomials are not accurate enough to describe the thermophysical and thermodynamic values of hydrogen and argon.

12.5 Comparison of Hydrogen and argon thermal plasma

The hydrogen results compared to argon show some contradictions to the litterateur presented in the theory section. For example, Paik et al.[26] found that the axial temperature should decrease due to high heat capacity and a shorter arc length. However, the result from Figure 44 shows a higher axial temperature present for the whole axes for the same mass flow of argon and hydrogen, which is only surpassed by the argon flow at the same velocity. In addition, argon shows a shorter arc length than hydrogen at the same mass flow, opposite to the theory presented by Paik et al.[26]. On the other hand, Ramachandran et al.[28], who used a mixture, found that the axial temperature increased at a higher hydrogen mass flow. However, the thermodynamic properties of a mixture of hydrogen and argon are quite different from using pure mono-atomic gases. Hence, a conclusion from literature can not be concluded here as it is performed under different conditions to the results in this project.

Looking at Figure 40 and Figure 43 a better perspective of the whole arc domain is seen. For both the same mass and inflow velocity, the axial temperature of hydrogen is higher. However, this is only true close to the axial, whereas moving up in the radial direction reveals that in all radial profiles between 1 and 3 - 5 $[mm]$, the argon plasma dominates in temperature, which would be expected from the higher product of heat capacity and density in hydrogen compared to argon. The percentage difference graph shows that the sum of the differences for all profiles is at its highest at the temperature range between 8000 - 14000 $[K]$, where the heat capacity of hydrogen is higher than argon. The axial points of 15.8 and 27.8$[mm]$ go opposite to the trend as hydrogen has a higher temperature than the isotherm of the argon runs. This can be contributed to the arc length and anode current density profile shown in Figure 48. The increased current in this region of hydrogen compared to the argon cases might add extra power, skewing the results. Also, the lower temperature isotherms generally show a higher hydrogen temperature, which could be caused by the increased heat conductivity in the area of hydrogen. As seen for the polynomial fit, the peak in the thermal conductivity of hydrogen is not represented very well, which counteracts the main contributor to a constricted arc suggested by Paik et al.[26], and might be the mechanism behind the large difference of the width of the arc channel between results and literature.

For the maximal axial velocity, the literature and results in Figure 46 are more similar, as both Paik et al.[26], and the results show hydrogen to have a maximal velocity 670% larger than that of argon. This velocity difference follows Equation 55 as the density of hydrogen is lower than argon at the same temperatures, as seen in Figure 16 and Figure 8. However, the axial velocity difference is much larger than the literature value of approximately 157% from Paik et al.[26].

The results of hydrogen and argon simulation runs under the same conditions show some differences which are not supported by the findings in Paik et al.[26], hence do not support the verification of the procedure used in this work. If this is because of the source term implementation along the axial or the inaccuracy of the polynomial fit is unknown. However, the effect on the plasma of temperature dependent thermodynamic and transport properties needs to be further explored, as it would help point to where the procedural inaccuracy might be.

12.6 effect of current and inflow velocity on hydrogen and argon

The parameter testing analysis gives information on the argon plasma behavior, which can be compared to the existing literature on how argon plasma behaves. First, the result for the arc shows a clear trend where the arc length and voltage drop increase with increasing inflow velocity. This is also found in the studies of Westhoff et al.[29], and Perambadur et al.[27], which is supported by Equation 53 and the data given for the relationship of voltage to arc length. However, the voltage drop and arc length barely change as the current increases, which is in contrast to the findings of Perambadur et al.[27] and Equation 53. This could suggest that the Lorentz force is not acting as it should in Equation 52, and that there are faults in its procedural implementation in Listing 4. The change in the current profile is caused by the inflow velocity, which pushes the arc attachment to the boundary wall. Here the electric current can't flow, as seen by the BC Table 3, where the electric gradient is set to zero. Thus, the current must flow upwards to a small area, which gives a large peak current density.

The temperature also follows the expected trend from earlier studies, as the increase of current pushes the temperature profile upwards, as seen in Figure 49, and thus the max and average temperature, which is shown in the earlier studies of Westhoff et al.[29]. It is connected to the definition of Joule heating, as the source term increases with a larger current, as shown by Equation 23. The increase of max temperature as a function of inflow velocity is not shown in litterateur. However, the large increase of voltage drop in Figure 53 could increase Joule heating, as the Joule heating is dependent on the current, which is again dependent on the voltage drop. This increase in temperature for higher inflow velocity is also shown for the outlet temperature in Figure **??**, which is supported by Perembadur et al.[27]. The mechanism is not explicitly given in the literature. However, the presence of the arc attachment further downstream could cause a more significant current density downstream, especially for the arc attachment located at the outlet position.

The velocity from Figure 51 follows Equation 55, where the increased current increases the maximum and average velocity as the curves are moved upwards. This follows as the Lorentz force and Joule heating increase with the current, creating a density decrease and increased pressure to drive the flow. The influence on outflow velocity also follows the same trend, but here the increase of inflow velocity has a larger effect on the maximum and average velocity as seen from Figure 52. This is contributed to the increased arc length compared to literature.

For the hydrogen, litterateur studies on the parameters current and inflow velocity are lacking, and the only found was parameter testing on mixtures from Ramachandran et al.[28]. Here the author increased the hydrogen inflow for a base volumetric flow of 40 [slpm] argon, and 8 [slpm] hydrogen, making it heavily influenced by the properties of argon. Nonetheless, the findings of these papers correlate to the volumetric increase of flow for pure hydrogen gas, as both Figure 55 and Figure 57 show increasing outflow temperature and velocity. However, this is weak evidence.

The effect of increasing current on hydrogen follows the given definition of Joule heating in Equation 23 and the Maecker effect in Equation 55, as the max and average temperature and velocity increases with current in Figure 54 and Figure 56. The same trend is followed for the outflow values, but no results in the literature were found to support these results. A surprising result is the electric current's effect on the anode current attachment behavior, as seen in Figure 58, as the increase of current increases the arc length, especially at low inflow rates. This contradicts the argon findings of all authors mentioned in the literature and does not support the theory and balance between convective drag and Lorentz force, as stated in Equation 52. A factor affecting this might be that the Maecker effect in Equation 55 increases more than the contribution of Lorentz force due to arc curvature in the case of hydrogen. However, no evidence directly supports this, and the conclusion must be that either the force balance in Equation 52 behaves differently for hydrogen or that the Lorentz force implementation is not behaving properly for hydrogen due to the procedural implementation in Listing 4.

Another great outlier in Figure 58 is the decrease of voltage drop for 150$[A]$ current when inflow is increased from 0.7495 to 2.498 $[\frac{m}{s}]$. Here the voltage drop increases slightly, opposite to the other results for voltage drop using argon. No definite cause for this is found in the literature in this work. This also includes the behavior of the maximum velocity in Figure 56, where the maximum decreases with increasing inlet flow while the average flow increases.

To summarize the parameter test, some aspects of the response of argon and hydrogen plasma follow the findings and theory found in the literature. As there is too little information on hydrogen's response to current and inflow velocity, no definite conclusion can be made on the procedural implementation. On the other hand, for argon, the evidence of the result and theory suggest that the Joule heating source is implemented and behaving according to theory and prior results. The results found in this work do not explicitly support the Lorentz force's behavior. Hence the implementation in Listing 4 needs further work and testing.

12.7 Verification case of argon

The comparison between the isothermal temperature field of Westhoff et al.[29] to the simulation runs used in the verification case, listed in Table 5, show that the temperatures fields of these simulations show differences from the work of Westhoff et al.[29]. Especially true is the difference in the maximum temperature of any of the simulations, where the Devoto conductive simulations are closer than the Boulos one. As the max temperature is driven mainly by the Energy equation, Equation 24, an implementation error might be present in either of the source terms here. The electron enthalpy diffusion source has not been tested. Hence, no conclusions can be derived from it. However, radiance and Joule heating are relevant candidates. As the cathode current density is lower and the arc radius larger than what is expected, a correct current might be the problem. A better implementation could add extra power to the temperature torch profile, thus raising its temperature. The isothermals show that the simulation runs lack power input over the whole arch body, as all radial temperature profiles at the axial points of 1.7 $[mm]$, 5.7 $[mm]$, 15.84 $[mm]$ and 27.8 $[mm]$, which supports the idea of raising the power in the torch.

However, from the radial profiles, such as Figure 26, it is evident that the percentage error drastically increases from around 8000 $[K]$ - 6000 $[K]$ and downwards regardless of where the radial profile is extracted. Radiation could be a factor here, as the values from a different model than the work of Westhoff et al.[29] are used for temperatures below 10000 $[K]$. However, it can't explain the significant temperature drop occurring in the radial profiles. Another factor could be the inaccuracy and oscillating behavior of the polynomial approximation procedure at this temperature range. As seen from Figure 9 and Figure 11, the heat capacity is over 200% as high as it should be at this temperature range, and the thermal conductivity over 70% as small, which might lead to an unnaturally high gradient.

However, another problem is the arc length and, as an extension, the Lorentz force. The radial temperature profiles of Figure 29, which has a lower velocity than what is used in the work of Westhoff et al.[29], show a drastically smaller percentage error for the downstream radial temperature profiles. It also show a more similar curve characteristic in Figure 34 for the axial temperature and Figure 35 for the outflow temperature. The mechanism behind this is the arch length shown in Figure 39, which attaches around the same spot as the work of Westhoff et al.[29], around 10 [mm], because of a lower inflow velocity applied. Thus Equation 52 shows that the Lorentz force implemented in this work is most likely not creating as much force on the anode attachment counteracting the drag from the flow as the Westhoff et al.[29] simulations.

Lastly, the maximum current of the anode current density profile is around 10 - 100 times larger than that of Westhoff et al.[29], creating a sharp, distinct peak. The profile here depends on the boundary condition set for the anode, which is the procedure used in Listing 8. This procedure could cause a high current density peak as it overestimates the electrical conductivity.

To conclude, the verification runs show a promising result, as the temperature field throughout the arc body is close to that of Westhoff et al.[29], and the axial only show a 5% difference. However, the error increase in the radial direction. Low Joule heating from a lower than expected applied current density or the effects of inaccurate radiance and thermodynamic and thermophysical properties could be to blame. In addition, the arc length is over three times as long, and the current density is over ten times as high for the simulation result in this work. This inaccuracy points to procedural faults in the Lorentz force implementation and the electrical conductivity boundary procedure.

13 Conclusion

This project has created a new solver procedure to solve thermal plasma systems from the literature and theory available. However, the results show that the procedure developed in this work show inaccuracies and some large discrepancies compared to older results with the same case setup and literature studies on parameter testing. The same was true for the arc's response to inflow velocity and current. In addition, the electric current distribution at the anode shows large differences compared to the benchmark case. The Lorentz force is the main factor affecting the arc, according to the theory known to the author. Hence the Lorentz force implementation procedure should be researched and analyzed in depth before being used by other authors. The cathode current profile is also slightly different from the theory. The primary mechanism might be this project's electrical conductivity boundary procedure and the electromagnetic model itself. Hence alternative methods should be researched.

Furthermore, the use of OpenFOAM's polynomial thermophysical implementation has been shown inaccurate for thermal plasma simulations, as it could be causing features in the results incompatible with the theory. Hence it is not recommended for future works. Lastly, only argon simulations run with a small mesh and Devoto conductivity reached zero residual. This discredits the hydrogen results as it is not known if the plasma has reached a steady state solution. Even though the procedure and results are not fully satisfactory in their current state, the procedure still gives some insight into the subject area. In addition, this work still fulfills some of its goals as some plasma features are seen from the implementation in OpenFOAM.

14 Further work

This most obvious goal is to run the simulations for longer, as three million iterations are clearly not enough to reach a steady state. In addition, better residual methods need to be explored. For the procedure, the effects of the Lorentz force need to be explored by running simulations without it and see what effect this has on the anode attachment. Different techniques to handle boundary conductivity must also be explored, where for example, linear interpolation could be used. Finally, the polynomial implementation must be replaced by a more accurate method, which could be a linear interpolation between tabulated values of the thermophysical data.

In reality, the plasma deviates from both LTE and LCE near edges and with higher flow rates, which needs to be implemented before such a solver can be industrially relevant on a large scale. In addition, the real arc is not in a steady state or two-dimensional axisymmetric. Hence three-dimensional transient effects must be considered to develop a complete working simulation as the movement of the arc drastically changes its behavior compared to the result in this work.

Finally, comparing the procedure of this work or future work to simulation results from other authors will not verify the procedure for industrial use, as only experimental values can truly verify a modeling procedure. Hence experimental results from real plasma torches are needed.

Bibliography

[1] K. C. Sabat, P. Rajput, R. K. Paramguru, B. Bhoi and B. K. Mishra, ‘Reduction of Oxide Minerals by Hydrogen Plasma: An Overview’, en, *Plasma Chemistry and Plasma Processing*, vol. 34, no. 1, pp. 1–23, Jan. 2014, ISSN: 1572-8986. DOI: 10.1007/s11090-013-9484-2. [Online]. Available: https://doi.org/10.1007/s11090-013-9484-2 (visited on 28/06/2021).

[2] M. Mihovsky, ‘THERMAL PLASMA APPLICATION IN METALLURGY (REVIEW)’, en, *Journal of the University of Chemical Technology and Metallurgy*, p. 16, 2010.

[3] H. Dalaker, N. Eldrup, R. Jensen and R. Kvande, ‘Techno-economic pre-feasibility study of a hydrogen plasma-based ferromanganese plant’, in *REWAS 2022: Developing Tomorrow’s Technical Cycles (Volume I)*, Springer, 2022, pp. 647–658.

[4] A. B. Murphy and D. Uhrlandt, ‘Foundations of High-Pressure Thermal Plasmas’, en, *Plasma Sources Science and Technology*, vol. 27, no. 6, p. 063 001, Jun. 2018, Publisher: IOP Publishing, ISSN: 0963-0252. DOI: 10.1088/1361-6595/aabdce. [Online]. Available: https://doi.org/10.1088/1361-6595/aabdce (visited on 24/06/2021).

[5] M. l. Boulos, *Thermal Plasmas*. Plenum Press, 1994, ISBN: 0-306-44607-3.

[6] A. Murphy, ‘Transport coefficients of hydrogen and argon–hydrogen plasmas’, *Plasma Chemistry and Plasma Processing*, vol. 20, no. 3, pp. 279–297, 2000.

[7] A. Fridman, *Plasma Chemistry*. Cambridge University Press, 2008, pp. 1–4, ISBN: 978-0-521-84735-3. (visited on 08/06/2021).

[8] D. Spreitzer and J. Schenk, ‘Reduction of Iron Oxides with Hydrogen—A Review’, en, *steel research international*, vol. 90, no. 10, p. 1 900 108, 2019, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/srin.201900108, ISSN: 1869-344X. DOI: 10.1002/srin.201900108. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/srin.201900108 (visited on 20/07/2021).

[9] K. C. Sabat and A. B. Murphy, ‘Hydrogen Plasma Processing of Iron Ore’, en, *Metallurgical and Materials Transactions B*, vol. 48, no. 3, pp. 1561–1594, Jun. 2017, ISSN: 1073-5615, 1543-1916. DOI: 10.1007/s11663-017-0957-1. [Online]. Available: http://link.springer.com/10.1007/s11663-017-0957-1 (visited on 28/06/2021).

[10] J. Trelles, C. Chazelas, A. Vardelle and J. Heberlein, ‘Arc plasma torch modeling’, *Journal of thermal spray technology*, vol. 18, no. 5, pp. 728–752, 2009.

[11] S. Lopes, ‘Integrated cfd model for nanoparticle production in inductively coupled plasma reactors: Implementation and application’, Ph.D. dissertation, Université Libre de Bruxelles, 2016.

[12] F. White, *Fluid Mechanics*. McGraw-Hill, 2011, ISBN: 978-007-131121-2.

[13] B. Muller, *Introduction to computational fluid dynamics*, 2021.

[14] M. Sass-Tisovskaya, 'Plasma Arc Welding Simulation with OpenFOAM', en, p. 85, 2009.

[15] R. H. e. a. Pletcher, *Computational Fluid Mechanics and Heat Transfer*. Taylor and Francis Group, 2013, ISBN: 978-1-59169-037-5.

[16] D. J. Griffiths, *Introduction to Electrodynamics*. Cambridge University Press, 2017, ISBN: 978-1-108-42041-9.

[17] J. v. Dijk, K. Peerenboom, M. Jimenez, D. Mihailova and J. v. d. Mullen, 'The plasma modelling toolkit Plasimo', en, *Journal of Physics D: Applied Physics*, vol. 42, no. 19, p. 194 012, Sep. 2009, Publisher: IOP Publishing, ISSN: 0022-3727. DOI: 10.1088/0022-3727/42/19/194012. [Online]. Available: https://doi.org/10.1088/0022-3727/42/19/194012 (visited on 25/06/2021).

[18] C. Busse, I. Tsivilskiy, J. Hildebrand and J. P. Bergmann, 'Numerical modeling of an inductively coupled plasma torch using OpenFOAM', en, *Computers & Fluids*, vol. 216, p. 104 807, Feb. 2021, ISSN: 0045-7930. DOI: 10.1016/j.compfluid.2020.104807. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045793020303777 (visited on 25/06/2021).

[19] J. J. Gonzalez, F. Lago, P. Freton, M. Masquère and X. Franceries, 'Numerical modelling of an electric arc and its interaction with the anode: Part II. The three-dimensional model—influence of external forces on the arc column', en, *Journal of Physics D: Applied Physics*, vol. 38, no. 2, pp. 306–318, Jan. 2005, Publisher: IOP Publishing, ISSN: 0022-3727. DOI: 10.1088/0022-3727/38/2/016. [Online]. Available: https://doi.org/10.1088/0022-3727/38/2/016 (visited on 14/06/2021).

[20] S. Education, *Kinetic theory*, Accessed: 07-12-2021, 2021.

[21] A. Murphy, 'Diffusion in equilibrium mixtures of ionized gases', *Physical Review E*, vol. 48, no. 5, p. 3594, 1993.

[22] A. Gleizes, J.-J. Gonzalez and P. Freton, 'Thermal plasma modelling', *Journal of Physics D: Applied Physics*, vol. 38, no. 9, R153, 2005.

[23] S. G. Johnsen and A. J. SIMONSEN, 'CFD modeling of a rotating arc plasma reactor', *10th International Conference on CFD in Oil abd gas, Metallurgical and Process Industries*, 2014.

[24] J. P. Trelles, 'Computational study of flow dynamics from a dc arc plasma jet', *Journal of Physics D: Applied Physics*, vol. 46, no. 25, p. 255 201, 2013.

[25] Z. Guo, S. Yin, H. Liao and S. Gu, 'Three-dimensional simulation of an argon–hydrogen dc non-transferred arc plasma torch', *International Journal of Heat and Mass Transfer*, vol. 80, pp. 644–652, 2015.

[26] S. Paik, P. Huang, J. Heberleinand and E. Pfender, 'Determination of the arc-root position in a dc plasma torch', *Plasma chemistry and plasma processing*, vol. 13, no. 3, pp. 379–397, 1993.

[27] J. Perambadur, A. Y. Klimenko, V. Rudolph and P. Shukla, 'The investigation of arc fluctuations in thermal plasma torch using 3d modeling approach', *International Journal of Heat and Mass Transfer*, vol. 165, p. 120 666, 2021.

[28] K. Ramachandran, J. Marqués, R. Vaßen and D. Stöver, 'Modelling of arc behaviour inside a f4 aps torch', *Journal of Physics D: Applied Physics*, vol. 39, no. 15, p. 3323, 2006.

[29] R. Westhoff and J. Szekely, 'A model of fluid, heat flow, and electromagnetic phenomena in a nontransferred arc plasma torch', *Journal of applied physics*, vol. 70, no. 7, pp. 3455–3466, 1991.

[30] H.-P. Li and X. Chen, 'Three-dimensional modelling of a dc non-transferred arc plasma torch', *Journal of Physics D: Applied Physics*, vol. 34, no. 17, p. L99, 2001.

[31] J. Trelles and J. Heberlein, 'Simulation results of arc behavior in different plasma spray torches', *Journal of Thermal Spray Technology*, vol. 15, no. 4, pp. 563–569, 2006.

[32] J. P. Trelles, E. Pfender and J. Heberlein, 'Multiscale finite element modeling of arc dynamics in a dc plasma torch', *Plasma chemistry and plasma processing*, vol. 26, no. 6, pp. 557–575, 2006.

[33] A. Westermoen, 'Modelling of Dynamic Arc Behaviour in a Plasma Reactor', Norwegian University of Science and Technology, Tech. Rep., Mar. 2007.

[34] A. Fang, *Advanced process simulation*, Accessed: 13-06-2022, 2017. [Online]. Available: %5Curl%7Bhttps://folk.ntnu.no/preisig/HAP_Specials/AdvancedSimulation_files/2017/project%5C%20reports/CFD/Amos%5C%20Fang%5C%20-%5C%20openFOAM_Amos_Fang%5C%20.pdf%7D.

[35] C. J. Greenshields, *Openfoam programmer's guide: The open source cfd toolbox*, 2015.

[36] ——, *File structure of openfoam cases*, Accessed: 12.12.2021, 2021. [Online]. Available: https://cfd.direct/openfoam/user-guide/v9-case-file-structure/#x16-1220004.1.

[37] ——, *Openfoam v9 user guide: 7 models and physical properties*, Accessed: 12.12.2021, 2021. [Online]. Available: https://cfd.direct/openfoam/user-guide/v9-models/#x35-2680007.

[38] ——, *Openfoam v9 user guide: 5.1 mesh description*, Accessed: 12.12.2021, 2021. [Online]. Available: https://cfd.direct/openfoam/user-guide/v9-mesh-description/#x24-1680005.1.

[39] ——, *Openfoam v9 user guide: 5.2 boundaries*, Accessed: 12.12.2021, 2021. [Online]. Available: https://cfd.direct/openfoam/user-guide/v9-boundaries/#x25-1770005.2.

[40] OpenFOAM.com, *Standard boundary conditions*, Accessed: 26-05-2022, 2022. [Online]. Available: %5Curl%7Bhttps://www.openfoam.com/documentation/user-guide/a-reference/a.4-standard-boundary-conditions%7D.

[41] N. institute of standards and technology, *Nist chemistry webbook*, Accessed: 12-05-2022, 2022. [Online]. Available: %5Curl%7Bhttps://webbook.nist.gov/chemistry/%7D.

[42] OpenFOAMwiki, *Groovybc*, Accessed: 15-03-2022, 2015. [Online]. Available: %5Curl%7Bhttps://openfoamwiki.net/index.php/Contrib/groovyBC%7D.

[43] numpy, *Numpy.polyfit*, Accessed: 10-3-2022, 2022. [Online]. Available: %5Curl%7Bhttps://numpy.org/doc/stable/reference/generated/numpy.polyfit.html%7D.

[44] R. Devoto, 'Transport coefficients of ionized argon', *The Physics of fluids*, vol. 16, no. 5, pp. 616–623, 1973.

[45] D. Scott, P. Kovitya and G. Haddad, 'Temperatures in the plume of a dc plasma torch', *Journal of applied physics*, vol. 66, no. 11, pp. 5232–5239, 1989.

[46] D. L. Evans and R. Tankin, 'Measurement of emission and absorption of radiation by an argon plasma', *The Physics of Fluids*, vol. 10, no. 6, pp. 1137–1144, 1967.

[47] A. Essoltani, P. Proulx, M. Boulos and A. Gleizes, 'Volumetric emission of argon plasmas in the presence of vapors of fe, si, and al', *Plasma Chemistry and Plasma Processing*, vol. 14, no. 4, pp. 437–450, 1994.

[48] Y. Cressault, M. Rouffet, A. Gleizes and E. Meillot, 'Net emission of ar–h2–he thermal plasmas at atmospheric pressure', *Journal of Physics D: Applied Physics*, vol. 43, no. 33, p. 335 204, 2010.

[49] H. Weller, *C++ source guide*, Accessed: 5-3-2022, 2021. [Online]. Available: %5Curl%7Bhttps://cpp.openfoam.org/v8/%7D.

[50] A. Nordhagen, 'Plasma simulations in openfoam', 2021.

[51] H. Nilsson, *Implement a simple electromagnetic solver*, Accessed: 02-3-2022, 2020. [Online]. Available: %5Curl%7Bhttp://www.tfd.chalmers.se/~hani/kurser/OS_CFD_2020/lectureNotes/12_implementElectromagneticSolver.pdf%7D.

[52] O. Olesen, *Openfoam api guide v2112*, Accessed: 22-05-2022, 2021. [Online]. Available: %5Curl%7Bhttps://www.openfoam.com/documentation/guides/latest/api/classFoam_1_1fv_1_1jouleHeatingSource.html#details%7D.

[53] J. van Baten, *Amsterchem*, Accessed: 3-03-2022, 2017. [Online]. Available: %5Curl%7Bhttps://www.amsterchem.com/scanit.html%7D.

Appendix

A Solver procedure scripts

```

Info<< "Reading physicalProperties\n" << endl;
IOdictionary physicalProperties
(
    IOobject
    (
            "physicalProperties",
            runTime.constant(),
            mesh,
            IOobject::MUST_READ,
            IOobject::NO_WRITE
    )
);
dimensionedScalar muMag
(
    "muMag",
    dimensionSet(1, 1, -2, 0, 0, -2, 0),
    physicalProperties
);

dimensionedScalar k_b
(
    "k_b",
    dimensionSet(1, 2, -2, -1, 0, 0, 0),
    physicalProperties
);

dimensionedScalar e_c
(
    "e_c",
    dimensionSet(0, 0, 1, 0, 0, 1, 0),
    physicalProperties
);

dimensionedScalar First_res_Scalar
(
    "First_res_Scalar",
    dimensionSet(0, 0, 0, 0, 0, 0, 0),
    physicalProperties
);

double First_res = First_res_Scalar.value();



```

```
Info<< "Reading field sigma\n" << endl;
volScalarField sigma
(
    IOobject
    (
        "sigma",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);

 volScalarField Rad
 (
   IOobject
   (
     "Rad",
     runTime.timeName(),
     mesh,
     IOobject::READ_IF_PRESENT,
     IOobject::AUTO_WRITE
   ),
   mesh,
   dimensionSet(1,-1,-3, 0,0,0,0)
 );


volScalarField ElPot
(
    IOobject
    (
        "ElPot",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);

Info<< "Reading field A\n" << endl;
volVectorField A
    (
    IOobject
    (
        "A",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);

Info << "Calculating magnetic field B \n" << endl;
volVectorField B
    (
    IOobject
    (
        "B",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    fvc::curl(A)
);

volVectorField Je
(
    IOobject
    (
        "Je",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    -sigma*(fvc::grad(ElPot))
);

volVectorField F_l
(
    IOobject
    (
        "F_l",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    (((Je^B)))
);


volScalarField  E_J
(
    IOobject
    (
        "E_Jb",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    (Je&(-(fvc::grad(ElPot))))
);
```
Listing 14: Snippet from createFieldsE.H/

```

        solve(fvm::laplacian(sigma, ElPot));


        solve(fvm::laplacian(A)==sigma*muMag*(fvc::grad(ElPot)));



        B = fvc::curl(A);
        Je = -sigma*(fvc::grad(ElPot));

        F_l = (((Je^B)));

        E_J =  (Je&(-(fvc::grad(ElPot))));
```
Listing 15: Snippet from caclFieldsE.H/ .

```

scalarField& RadCells = Rad.primitiveFieldRef();


forAll( RadCells, Radcelli)
    {
        RadCells[Radcelli] = 4*Foam::constant::mathematical::pi*pow(10,(
     (-9.74738451e-52)*pow(thermo.T()[Radcelli],13) + (1.72988685e-46)*
   pow(thermo.T()[Radcelli],12) + (-1.36979930e-41)*pow(thermo.T()[
   Radcelli],11) + (6.37817811e-37 )*pow(thermo.T()[Radcelli],10) +
   (-1.93728139e-32 )*pow(thermo.T()[Radcelli],9) + (4.02444763e-28)*pow
   (thermo.T()[Radcelli],8) + (-5.83025572e-24)*pow(thermo.T()[Radcelli
   ],7) + (5.89782481e-20)*pow(thermo.T()[Radcelli],6) + (-4.11017114e
   -16)*pow(thermo.T()[Radcelli],5) + (1.92036935e-12)*pow(thermo.T()[
   Radcelli],4) + (-5.78117589e-09)*pow(thermo.T()[Radcelli],3) +
   (1.05596042e-05)*pow(thermo.T()[Radcelli],2) + (-8.26080120e-03)*
   thermo.T()[Radcelli] +  -6.32519497e+00));

    }
```
Listing 16: Snippet from caclRadLower.H/.

```

{
    volScalarField& he = thermo.he();

    scalarField F = ((sigma*(fvc::grad(ElPot)&fvc::grad(ElPot)))/(thermo
   .rho()*he));

    scalar F_max = max(F);

    scalar W_max = Foam::sqrt(1/(F_max));

    scalar C_T = (  1/(  1+(1000/runTime.value())  )  );

    fvScalarMatrix EEqn
    (
        fvm::div(phi, he)
      + (
            he.name() == "e"
          ? fvc::div(phi, volScalarField("Ekp", 0.5*magSqr(U) + p/rho))
          : fvc::div(phi, volScalarField("K", 0.5*magSqr(U)))
        )
      + thermophysicalTransport->divq(he)
     ==
        fvOptions(rho, he) + C_T*E_J  - C_T*Rad + C_T*((5*(k_b))/(2*
   thermo.Cp()*(e_c)))*(Je&fvc::grad(he))
    );

    EEqn.relax();

    fvOptions.constrain(EEqn);

    EEqn.solve();

    fvOptions.correct(he);

    thermo.correct();


}
```

Listing 17: Snippet from EEqn.H/thermoFoam

```

{
     volScalarField& he = thermo.he();




    scalar C_T = (  1/(  1+(1000/pow(1,1.45)  )  ));
    if (C_T > 0.95) {
      C_T = 1;
    }

     fvScalarMatrix EEqn
     (
         fvm::div(phi, he)
       + (
             he.name() == "e"
           ? fvc::div(phi, volScalarField("Ekp", 0.5*magSqr(U) + p/rho))
           : fvc::div(phi, volScalarField("K", 0.5*magSqr(U)))
         )
       + thermophysicalTransport->divq(he)
      ==
       fvOptions(rho, he) + C_T*E_J  - C_T*Rad + C_T*((5*(k_b))/(2*
   thermo.Cp()*(e_c)))*(Je&fvc::grad(he))
     );


     EEqn.relax();

     fvOptions.constrain(EEqn);

     EEqn.solve();

     fvOptions.correct(he);

     thermo.correct();

      Info<< "C_T is coefficient is :" << C_T << endl;
 }
```

Listing 18: Snippet from EEqn2.H/thermoFoam

```
#include "fvCFD.H"
#include "rhoThermo.H"
#include "fluidThermoMomentumTransportModel.H"
#include "fluidThermophysicalTransportModel.H"
#include "LESModel.H"
#include "radiationModel.H"
#include "fvOptions.H"
#include "simpleControl.H"

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
    * * //

int main(int argc, char *argv[])
{
    #include "postProcess.H"

    #include "setRootCaseLists.H"
    #include "createTime.H"
    #include "createMesh.H"
    #include "createControl.H"
    #include "createFields.H"
    #include "createFieldsE.H"
    #include "OFstream.H"

    fileName name = ("yorFileName");
    OFstream OS(name);


    OS << "Time" << "\t" << "Residual Sum" << "\t" << "rel_tol" << "\t" << "rel_tol_first" "\t" << "Temp at: ("
    << mesh.C()[5927].x() << "," << mesh.C()[5927].y() << ")" << "\t" << "Conductivity at: ("
    << mesh.C()[5927].x() << "," << mesh.C()[5927].y() << ")" << "\t"<< "Temp at: ("
    << mesh.C()[2513].x() << "," << mesh.C()[2513].y() << ")" << "\t"<< "Conductivity at: ("
    << mesh.C()[2513].x() << "," << mesh.C()[2513].y() << ")" << "\t"<< "Temp at: ("
    << mesh.C()[3913].x() << "," << mesh.C()[3913].y() << ")" << "\t"<< "conductvity at: ("
    << mesh.C()[3913].x() << "," << mesh.C()[3913].y() << ")" << endl;
    //#include "calcFieldsE.H"

    // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

    Info<< "\nEvolving thermodynamics\n" << endl;



        bool l1 = true;
        double Sum_Old = 1;
        for (int i = 0; i <= 1000000; i++)
        {
            int l = 1;





            #include "SetConductBC.H"
            #include "calcFieldsE.H"
            sigma.correctBoundaryConditions();
            #include "ResidualTimes.H"

            while (runTime.loop())
            {
                Info<< "Time = " << runTime.timeName() << nl << endl;
```

```
                    simple.storePrevIterFields();

                    #include "calcRadLower.H"

                    if (l1) {
                    Info<< "Time warming up "<< endl;
                    #include "EEqn.H"

                    }

                    else {
                   Info<< "We are solving!"<< endl;
                    #include "EEqn2.H"

                    }



                    double T_max = max(thermo.T()).value();

                    /*if ( T_max > 26000) {

                    Info<< "It's breaking Because of temp!!! "  << nl << endl;

                    runTime.write();
                    break;

                    }*/

                    l+= 1;
                    //if (((simple.converged()) && (l > 10000)) || (T_max > 26000) || (runTime.value() == 10000 )) {
                    if (( (l > 3000) || (T_max > 24000) ))
                    {

                        l1 = false;
                        runTime.writeNow();
                        runTime.write();
                        #include "SetConductInternalReal.H"

                        Info<< "It's breaking!!! "  << nl << endl;
                        break;
                    }

                Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
                    << "  ClockTime = " << runTime.elapsedClockTime() << " s"
                    << nl << endl;
                Info << "i is: " << i << endl;
                runTime.write();




            }

        }


    Info<< "End\n" << endl;

    return 0;
}
```

Listing 19: Snippet from thermoFoamJERamp

```

{
     volScalarField& he = thermo.he();




    scalar C_T = (  1/(  1+(1000/pow(1,1.45)  )  ));
    if (C_T > 0.95) {
      C_T = 1;
    }

     fvScalarMatrix EEqn
     (
         fvm::div(phi, he)
       + (
             he.name() == "e"
           ? fvc::div(phi, volScalarField("Ekp", 0.5*magSqr(U) + p/rho))
           : fvc::div(phi, volScalarField("K", 0.5*magSqr(U)))
         )
       + thermophysicalTransport->divq(he)
      ==
       fvOptions(rho, he) + C_T*E_J  - C_T*Rad + C_T*((5*(k_b))/(2*
   thermo.Cp()*(e_c)))*(Je&fvc::grad(he))
     );


     EEqn.relax();

     fvOptions.constrain(EEqn);

     EEqn.solve();

    fvOptions.correct(he);

    thermo.correct();

     Info<< "C_T is coefficient is :" << C_T << endl;
}
```

Listing 20: Snippet from EEqn2.H/thermoFoam

```
const fvPatchList& patches = mesh.boundary();
        forAll (patches , patchi ) {
            const fvPatch& SigmaPatch = patches[patchi];
            //fvPatchScalarField& faces_sigma = sigma.boundaryFieldRef()
   [ipatch];
            forAll(SigmaPatch, faceI)  {
                label faceCelli = SigmaPatch.faceCells()[faceI];

                sigma.boundaryFieldRef()[patchi][faceI] = sigma[
   faceCelli];

                }

        }

        label patchID = mesh.boundaryMesh().findPatchID("leftRCWall");
        const polyPatch& Elpatches = mesh.boundaryMesh()[patchID];
        fixedGradientFvPatchScalarField& ElPatch = refCast<
   fixedGradientFvPatchScalarField>(ElPot.boundaryFieldRef()[patchID]);
        forAll (ElPatch, EfaceI)
        {

            ElPatch.gradient()[EfaceI] = -((3e7)/(sigma.boundaryField()[
   patchID][EfaceI]))*pow(1 + Foam::exp(2*10000*(Elpatches.faceCentres()
   [EfaceI].y() - 1.6e-03)),-1);
        }
```

Listing 21: Snippet from SetConductBC.H

```



scalarField& sigmaCells = sigma.primitiveFieldRef();

        forAll(sigmaCells, sigmacelli)
            {
                //sigmaCells[sigmacelli] = (1+l)*1e-1*pow(10,(l+1));

                sigmaCells[sigmacelli] = (-2.68872190e-46)*pow(thermo.T
   ()[sigmacelli],13) + (5.71816721e-41)*pow(thermo.T()[sigmacelli],12)
   + (-5.56635110e-36)*pow(thermo.T()[sigmacelli],11) + (3.28386804e-31)
   *pow(thermo.T()[sigmacelli],10) + (-1.30989753e-26)*pow(thermo.T()[
sigmacelli],9) + (3.73058177e-22)*pow(thermo.T()[sigmacelli],8) +
(-7.80413235e-18)*pow(thermo.T()[sigmacelli],7) + (1.21412711e-13)*
pow(thermo.T()[sigmacelli],6) + (-1.40469700e-09)*pow(thermo.T()[
sigmacelli],5) + (1.19363669e-05)*pow(thermo.T()[sigmacelli],4) +
(-7.24077915e-02)*pow(thermo.T()[sigmacelli],3) + (2.96922898e+02)*
pow(thermo.T()[sigmacelli],2) + (-7.37705762e+05)*thermo.T()[
sigmacelli] + 8.38636954e+08;



                if ( thermo.T()[sigmacelli] < 9000 ){

                    sigmaCells[sigmacelli] = 0.2*Foam::exp(thermo.T()[
sigmacelli]/2000);

               }
          }
```

Listing 22: Snippet from SetConductInernalREal.H

```

 // Solve the Momentum equation

 MRF.correctBoundaryVelocity(U);

 tmp<fvVectorMatrix> tUEqn
 (
     fvm::div(phi, U)
   + MRF.DDt(rho, U)
   + turbulence->divDevTau(U)
  ==
     fvOptions(rho, U) + F_l
 );
 fvVectorMatrix& UEqn = tUEqn.ref();

 UEqn.relax();

 fvOptions.constrain(UEqn);

 solve(UEqn == -fvc::grad(p));

 fvOptions.correct(U);
```

Listing 23: Snippet from UEqn.H

```
        #include "setRootCaseLists.H"
 #include "createTime.H"
 #include "createMesh.H"
 #include "createControl.H"
 #include "createFieldsE.H"
 #include "createFields.H"
```

```
    #include "createFieldRefs.H"
    #include "initContinuityErrs.H"
    #include "OFstream.H"

    fileName name = ("yorFileName");
    OFstream OS(name);


    OS << "Time" << "\t" << "Residual Sum" << "\t" << "rel_tol" << "\t"
   << "rel_tol_first" << "\t" << "Temp at: ("
    << mesh.C()[5927].x() << "," << mesh.C()[5927].y() << ")" << "\t" <<
    "Conductivity at: ("
    << mesh.C()[5927].x() << "," << mesh.C()[5927].y() << ")" << "\t"<<
   "Temp at: ("
    << mesh.C()[2513].x() << "," << mesh.C()[2513].y() << ")" << "\t"<<
   "Conductivity at: ("
    << mesh.C()[2513].x() << "," << mesh.C()[2513].y() << ")" << "\t"<<
   "Temp at: ("
    << mesh.C()[3913].x() << "," << mesh.C()[3913].y() << ")" << "\t"<<
   "conductvity at: ("
    << mesh.C()[3913].x() << "," << mesh.C()[3913].y() << ")" << endl;
    turbulence->validate();

    // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
    * * //
    double Sum_Old = 1;
    Info<< "\nStarting time loop\n" << endl;
    for (int i = 0; i <= 10000000; i++)
    {

        int l = 1;
        if (runTime.value() > 1) {
            #include "SetConductInternalReal.H"
             Info<< "New Field is calculated "  << nl << endl;
        }

        #include "SetConductBC.H"
        #include "calcFieldsE.H"
        sigma.correctBoundaryConditions();
        #include "ResidualTimes.H"

        while (runTime.loop())
        {
            simple.storePrevIterFields();
            Info<< "Time = " << runTime.timeName() << nl << endl;

            // Pressure-velocity SIMPLE corrector
            #include "UEqn.H"
            #include "calcRadLower.H"
            #include "EEqn2.H"
            #include "pEqn.H"

            turbulence->correct();
            thermophysicalTransport->correct();



            l += 1;
            double T_max = max(thermo.T()).value();

            if ( T_max > 26000) {

                Info<< "It's breaking Because of temp!!! "  << nl << endl;

                runTime.write();
                runTime.writeNow();
                break;

            }
            if (((simple.converged()) && (l > 1000)))
            {
                Info<< "It's breaking!!! "  << nl << endl;
                runTime.writeNow();
                runTime.write();
                break;

            }



            runTime.write();
            Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
                << "  ClockTime = " << runTime.elapsedClockTime() << " s"
                << nl << endl;
            Info << " i is: " << i << endl;

        }
    }
    Info<< "End\n" << endl;

    return 0;
}
```

Listing 24: Snippet from rhoSimeFoamJBJEProcedureInitialArgonTCEW.C

```

scalarField Sigma_New = sigma.primitiveField();

scalarField Sigma_Old = sigma.oldTime().primitiveField();


double rel_tol = 0;
double Sum_New = 0;
double rel_tol_first = 0;
forAll(Sigma_New, sigmacelli)
            {
               Sum_New += abs(Sigma_New[sigmacelli] - Sigma_Old[
   sigmacelli]);



            }

rel_tol = (Sum_New)/(Sum_Old);
rel_tol_first = (Sum_New)/(First_res);

OS << runTime.value() << "\t" << Sum_New << "\t" << rel_tol << "\t" <<
   rel_tol_first << "\t" << thermo.T()[5927] << "\t"
            << sigma[5927] << "\t" << thermo.T()[2513] << "\t"
            << sigma[2513] << "\t" << thermo.T()[3913] << "\t"
            << sigma[3913] << endl;

if (Sum_New != 0) {
   Sum_Old = Sum_New;
}
```

Listing 25: Snippet from ResidualTimes.H.C

B OpenFOAM input files

```

FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      thermophysicalProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
    * * //

thermoType
{
    type            heRhoThermo;
    mixture         pureMixture;
    transport       polynomial;
    thermo          hPolynomial;//hPolynomial;
    equationOfState icoPolynomial;//rhoConst;
    specie          specie;
    energy          sensibleEnthalpy;
}

mixture
{

    specie
    {
        molWeight       28.96;
    }
    thermodynamics
    {
      Hf                0;
      Sf                0;

     CpCoeffs<8>     (-5.07474830e+01  1.56654977e+00 -4.62359244e-04
   2.27408407e-08 5.65570634e-12 -6.92028050e-16  2.74343339e-20
   -3.67614299e-25);
    }
    transport
    {
       muCoeffs<8>     ( 3.42040546e-06  1.16613952e-07 -5.38454942e-11
    1.28686692e-14 -1.46723082e-18  8.36492358e-23 -2.32758969e-27
   2.52709431e-32);

       kappaCoeffs<8>  (3.20579852e-01 -5.67583838e-04  3.78501374e-07
   -1.03199717e-10 1.35381415e-14 -8.85818265e-19  2.80780919e-23
   -3.44117489e-28);
    }

    equationOfState
     {
         rhoCoeffs<8>   (1.03979008e+00 -6.50987510e-04  1.74990497e-07
   -2.39575730e-11 1.80406291e-15 -7.56785048e-20  1.65802452e-24
   -1.47844663e-29);//(1.04969382e00 -6.66367551e-04  1.82058219e-07
   -2.53672677e-11 1.94534065e-15 -8.31018195e-20  1.85283448e-24
   -1.67959812e-29);
     }
}
```

Listing 26: Snippet from thermophysicalProperteis.C Argon

```

FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location     "constant";
    object       thermophysicalProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
    * * //

thermoType
{
    type             heRhoThermo;
    mixture          pureMixture;
    transport        polynomial;
    thermo           hPolynomial;//hPolynomial;
    equationOfState  icoPolynomial;//rhoConst;
    specie           specie;
    energy           sensibleEnthalpy;
}

mixture
{

    specie
    {
        molWeight       28.96;
    }
    thermodynamics
    {
      Hf               0;
      Sf               0;
      CpCoeffs<8>     (-2.20935264e+04  4.57818378e+01  7.87150545e-03
   -6.38383446e-06 1.03155851e-09 -7.00810224e-14  2.16062776e-18
   -2.49509487e-23);//(-4.19143607e+04  9.82249167e+01 -2.10622794e-02
   1.25737724e-08 3.39679986e-10 -3.11886260e-14  1.07056905e-18
   -1.29241923e-23);//(2.76873127e+04  4.57331026e+01 -1.47609479e-02
   1.09879947e-06 6.99245182e-11 -1.08884599e-14  4.13985576e-19
   -5.10407529e-24);//(-4.06558881e+04  9.40093915e+01 -1.86396364e-02
   -5.27686346e-07 3.97889926e-10 -3.44243081e-14  1.15976001e-18
   -1.38872999e-23);//(3.53304253e+04 -9.27096913e+01  8.67165895e-02
   -2.48135237e-05 3.15929023e-09 -1.98386850e-13  6.03113368e-18
   -7.10293347e-23);
    }
    transport
    {
       muCoeffs<8>     (-3.20814094e-05  8.24356173e-08 -3.54632281e-11
    7.50549592e-15 -8.02832952e-19  4.46275302e-23 -1.23644051e-27
   1.35255632e-32);//(-8.96583239e-06  4.49260250e-08 -1.71914345e-11
   3.59950875e-15 -3.78923951e-19  2.02589591e-23 -5.29892153e-28
   5.39783370e-33);

       kappaCoeffs<8>  (-2.34443250e-01  1.05616165e-03  1.74255186e-06
   -6.88621143e-10 9.74045610e-14 -6.47370557e-18  2.05128297e-22
```

```
    -2.50224652e-27);//(-3.44339017e+00  8.49182730e-03 -2.42338388e-06
    2.75809285e-10 -1.32046062e-14  1.64504732e-19  5.56081561e-24
    -1.31936268e-28);
    }

    equationOfState
     {
         rhoCoeffs<8>   (5.34249058e-02 -3.41351644e-05  9.11994293e-09
    -1.25874944e-12 9.68641678e-17 -4.18964083e-21  9.51499116e-26
    -8.82194320e-31);//(4.08342859e-02 -1.77549253e-05 2.87563684e-09
    -2.13045928e-13 7.31820142e-18 -9.45055650e-23 0 0);//(5.34249058e-02
     -3.41351644e-05  9.11994293e-09 -1.25874944e-12 9.68641678e-17
    -4.18964083e-21  9.51499116e-26 -8.82194320e-31);//(5.26036102e-02
    -3.28606188e-05  8.53220769e-09 -1.14040803e-12 8.48051538e-17
    -3.54025010e-21  7.75437725e-26 -6.93109212e-31);//( 5.26154996e-02
    -3.28790828e-05  8.54072335e-09 -1.14212105e-12 8.49792663e-17
    -3.54958160e-21  7.77948104e-26 -6.95773872e-31);//(5.61367655e-02
    -3.85920098e-05  1.13233545e-08 -1.73703450e-12 1.49572010e-16
    -7.26591263e-21  1.85694550e-25 -1.93936483e-30);//(1.04969382e00
    -6.66367551e-04  1.82058219e-07 -2.53672677e-11 1.94534065e-15
    -8.31018195e-20  1.85283448e-24 -1.67959812e-29);
     }



}
```

Listing 27: Snippet from thermophysicalProperteis.C Hydrogen

```
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
    * * //

dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (3 0 0);

boundaryField
{

      leftRCWall
    {
        type            noSlip;

    }

    leftReactorWall
    {
        type        groovyBC;
        value       uniform (0 0 0);

        valueExpression "vector(8.335*(1-(pow((pos().y-4.675e-03),2)/(
pow(1.675e-03,2)))),0,0)";

    }

     atmosphereTopRight

    {
        type            inletOutlet;
        inletValue      uniform (0 0 0);



    }

    atmosphereReactorRight
    {
        type            inletOutlet;
        inletValue      uniform (0 0 0);

    }
    /*
    atmosphereCathodeRight
    {
        type  pressureInletOutletVelocity;
    value uniform (5 0.0 0.0);
    }
    */
    atmosphereButtomRight
    {
        type            inletOutlet;
        inletValue      uniform (0 0 0);


    }

    atmosphereTop

    {

      type            inletOutlet;
      inletValue      uniform (0 0 0);


     }

    atmosphereTopLeft

    {

       type             noSlip;


     }


    CathodeTop
    {

       type             noSlip;


     }

     /*
    leftCathodeWall

    {

        type             noSlip;


    }
    */
     ReactorSmallTop
     {

        type             noSlip;


      }
    ReactorBigTop
      {

        type             noSlip;


      }

    ReactorSmallBack
    {
        type             wedge;
    }

    ReactorSmallFront
    {
        type            wedge;
    }

    ReactorBigBack
    {
        type            wedge;
    }

    ReactorBigFront
    {
        type            wedge;
    }
    /*
   CathodeBack
    {
        type            wedge;
    }

    CathodeFront
    {
        type            wedge;
    }
    */
    RCBack
    {
        type            wedge;
    }

    RCFront
    {
        type            wedge;
    }

     atmosphereTopBack
    {
        type            wedge;
    }

    atmosphereTopFront
    {
        type            wedge;
    }

    atmosphereReactorBack
    {
        type            wedge;
    }

    atmosphereReactorFront
    {
        type            wedge;
    }
    /*
    atmosphereCathodeBack
    {
        type            wedge;
    }

    atmosphereCathodeFront
    {
        type            wedge;
    }
    */
    atmosphereButtomBack
    {
        type            wedge;
    }

    atmosphereButtomFront
    {
        type            wedge;
    }

    axisReactor
    {
        type            empty;
    }

    axisAir
    {
        type            empty;
    }

}
```

Listing 28: Snippet from U file

Aren Lekve Nordhagen

Non-transferred arc hydrogen plasma torch simulations in OpenFOAM

NTNU

Norwegian University of
Science and Technology