BACHELOR

Multidimensional intrinsic low-dimension manifolds in plasma modelling

van Laanen, Joris C.A.

Award date:
2019
Multidimensional Intrinsic Low-Dimension Manifolds in plasma modelling

EINDHOVEN UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF APPLIED PHYSICS
BACHELOR FINAL PROJECT

Eindhoven, July 5, 2019

BACHELOR THESIS

Supervisors:  Ir. C.E.M. Schoutrop
              Dr. Ir. J. van Dijk

J.C.A. van Laanen  0935824
Abstract

An increased interest in complex plasmas demands a more thorough knowledge on this topic. Mathematical simulations show potential to provide this knowledge but are obstructed by the simulations times that can be excessively long. An Intrinsic Low-Dimensional Manifold (ILDM) approach can be used to reduce system kinetics, to simplify models and decrease computational costs. In this thesis a generic approach is adopted to construct ILDM manifolds for different dimensions. The results show that this generic approach is able to construct up to two-dimensional manifolds for linear systems and that there is room to extend this to non-linear systems. Furthermore, a method is developed that allows for locating and identifying stable and unstable equilibrium points without solving the time evolution of a dynamical system first.
Contents

1 Introduction .................................................. 1
  1.1 Challenges in the energy transition ...................... 1
  1.2 Chemical complexity and reduction techniques .......... 1
  1.3 Project goals and outline of thesis ..................... 2

2 Intrinsic Low-Dimensional Manifold method ................. 3
  2.1 Introduction .............................................. 3
  2.2 Conceptual method ....................................... 3
    2.2.1 Conceptual design of ILDM method .................... 4
    2.2.2 System stiffness and time scale analysis ............. 4
    2.2.3 Parametrization of the system ....................... 7
    2.2.4 Determination and use of manifold ................... 9
  2.3 Implementation .......................................... 10
  2.4 Practical aspects of implementation .................... 11
  2.5 Summary .................................................. 12

3 Mathematical aspects of method .................................. 13
  3.1 Introduction .............................................. 13
  3.2 Mathematical aspects of method ......................... 13
    3.2.1 Species densities and mass fractions ................. 13
    3.2.2 Reaction source terms ................................ 13
    3.2.3 Jacobian linearization ................................ 14
    3.2.4 Schur decomposition .................................. 15
  3.3 Case Study I: A three level hydrogen system (1D) .... 15

4 Determination of chemical equilibrium ....................... 21
  4.1 Introduction .............................................. 21
  4.2 Mathematical description of method ..................... 21
  4.3 Concept Case I: A Townsend avalanche mechanism ....... 22
  4.4 Conclusions .............................................. 24

5 Multidimensional ILDM approach ............................... 25
  5.1 Introduction .............................................. 25
  5.2 Case Study II: A three level hydrogen system (2D) ..... 25
  5.3 Case Study III: An argon molecular plasma .............. 28
  5.4 Conclusions .............................................. 31

6 General conclusions, discussion and outlook .................. 33
  6.1 Alternative method to obtain chemical equilibrium points 33
  6.2 Multidimensional ILDM for linear systems ................ 33
  6.3 Multidimensional ILDM for non-linear systems ............ 33
  6.4 Outlook .................................................. 34

References ................................................... 35

Appendix ..................................................... 37
1 Introduction

1.1 Challenges in the energy transition

In the past 300 years, disruptive changes altered the way we live, work and relate to one another. The first industrial revolution started around 1760 and created fundamentally new machinery driven by the power of water and steam.\[1\] A century later, the second industrial revolution enabled mass production by the use of electrical power and consequently, a rapid growth of industrialization followed. New technological systems such as railroad networks, telegraph and production lines emerged and facilitated an enormous increase in productivity.\[2\] This transition motivated people to start using coal as a complete new source of energy.\[3\] Not much later centralized electricity, televisions and radios represented another technological revolution that was enabled by use of oil as a new energy source and the demand for energy only increased ever since.

This increased demand for energy resulted in excessive use of fossil fuels accompanied by a serious emission of greenhouse gasses associated with the burning of these fuels. Motivated by the risk of serious problems due to climate change as a consequence of these high emissions, a energy transition towards renewable energy sources is taking place.\[4\] However, the production and consumption of these renewable energy sources can variety a lot dependent upon natural sources like sun, wind and water. Therefore, a constant supply of energy is not always possible and efficient storage is essential to avoid waste and maintain a stable energy supply. New technologies enabled by plasma physics show potential to overcome the problems in both generating and storing energy.

Plasma is often referred to as the fourth fundamental state of matter besides solid, liquid and gas.\[5\] It represents a gas state with enough energy to enable ions and electrons to coexist. Due to the high energy, some electrons are able to break free from their atom and remain present in the gas. The most famous plasma is the sun, but this is only a very small part of the 99+ percent of the visible universe that is estimated to be in this fourth state of matter.\[6\] Generating energy with plasmas sounds very attractive, since it shows potential to replicate the fusion processes from the sun inside a power plant. However, many challenges are yet to be beaten before this process can efficiently still the craving for energy.

In the meantime efficient storage of energy is another hot topic if it comes to sustainable energy. More established techniques like batteries, thermal energy, and hydro storage are already used for this purpose. Thermal energy storage in the form of heated water stored in large tanks underneath the ground and hydro storage that transports water to higher reservoirs to fuel hydro power plants. However, most of these techniques do not support long time energy storage without significant losses. Plasma can be used to store energy in the form of a synthetic fuel that are suitable to store energy for a longer period of time. This can also be done by the dissociation of carbon dioxide (CO₂), one of the important heat-trapping greenhouse gasses that causes climate change. Dissociation refers to the process of separating CO₂ in CO and O, with the use of the Fischer-Tropsch process these two products can be converted to the synthetic fuel methane. Plasmas can play an important role in efficient dissociation of CO₂.\[7\]

1.2 Chemical complexity and reduction techniques

The dissociation of CO₂ can be compared with a swing, if you push a CO₂ molecule exactly at the right time it will start to gain more and more energy until the energy will become high enough to dissociate the molecule. In a sense, plasmas are able to push such a molecule. However, only if the plasma has the exact right conditions it can efficiently perform this job. Experimenting with plasmas to find these conditions can by costly and very time consuming. Therefore, simulations of plasmas can help to find better methods and conditions that yield results that are worth experimenting with. Nevertheless, plasmas are often represented by a very complex chemistry containing tenths of species and hundreds of reactions. This complexity results in very large systems of equations that require enormous computational costs to be solved. Reduction techniques are helpful to reduce the complexity of such systems while at the same time maintaining accurate results of the simulated behavior.
Chemical complexity, as this problem is referred to, does also occur in other disciplines. To overcome this problem, different techniques known as Chemical Reduction Techniques (CRT) have been developed. However, these techniques have to be modified to be applicable to plasma modelling and sometimes this modification can be really problematic. An example is the technique Flamelet Generated Manifold (FGM), this method requires collinear gradients while plasmas contain many charged particles that show vibrating electronic and magnetic field vectors in many possible directions. A technique that is successfully modified to be used in plasma simulations is Principle Component Analysis (PCA). PCA is a statistical method that aims to find correlations in a set of observations. These correlations can be used to construct predictions on the behavior of new observations. A method that is proven to be very useful in generating useful results if enough observations are available and if more exact system kinetics are not required. However, sometimes not enough observations are available, or more detailed system kinetics are wanted. In these cases Intrinsic Low-Dimensional Manifold (ILDM) can be useful, this method allows detailed insights in the system kinetics, effective reduction of chemical complexity and does not require any observations besides a detailed reaction scheme with rate coefficients. The research in this thesis focuses on the use of ILDM applied to plasma physical systems.

1.3 Project goals and outline of thesis

The aim of this research is to generate new knowledge on the use of higher dimensional ILDM manifolds in plasma modelling. The main question of this research is how to construct a general way to extend the work of ILDM in plasma physics up to two dimensions and more dimensions? This question is answered by the following research questions:

- What method can help to locate stable equilibrium points in a dynamical system without solving the time evolution of such a system?
- How can multidimensional manifolds be created by the use of additional parameters and to what extend can this be done?
- What are the limitations in scaling up the dimensions of of the ILDM method and how to overcome these limitations?
- What additional information can be retrieved from the variation of dimensions of the ILDM manifolds?

This thesis will start with a brief introduction on the ILDM method and corresponding mathematics. It follows with the reproduction of earlier results on this topic and ends with an multidimensional extension on previous work. The thesis is structured as follows:

Chapter 2 explains the ILDM method in a conceptual manner to provide a deeper understanding and to introduce frequently used jargon. Also, the implementation of the ILDM method is discussed together with the practical aspects and limitations.

Chapter 3 is dedicated to provide supplementary mathematics that are avoided in the previous chapter. A more mathematical approach is maintained to explain how the Boltzmann equation leads to the dynamical systems discussed in this thesis. This chapter is finished with a case study that provides and in depth example on the modelling approach for a one-dimensional system.

Chapter 4 answers some of the first research questions. An analytical solution is presented to locate stable equilibria in a dynamical system without solving the time evolution. This chapter is finished with an example on a physical problem that is known for its different types of equilibria.

Chapter 5 presents the results that are obtained on a linear and non-linear systems. If possible the system is analyzed by using manifolds with a different number of dimensions. In other cases a careful reflection is provided on the modelling choices and its implications.

Chapter 6 concludes the thesis with the final answers to the research questions and provides new insights that can support further research on this topic.
2 Intrinsic Low-Dimensional Manifold method

2.1 Introduction

Even though plasmas make up more than 99 percent of the visible universe, a first quantitative understanding of plasmas only came in 1928. Nowadays plasmas are used in many applications, from fluorescent lights to potential nuclear fusion reactors like ITER. An outstanding example is the semiconductor industry that uses plasmas to generate extreme ultraviolet light for their lithography devices and also use it to treat microchips with different processes like plasma etching and plasma enhanced chemical vapor deposition. This increased interest demands a more thorough understanding of complex plasmas and as a result different chemical models and simulations have been developed. However, as explained in section 1.2, the large amount of different species and reactions causes simulations to get computationally expensive rather easily. This results in excessively long simulation times and therefore obstructs the usage of these models. A method that shows potential to overcome this problem of chemical complexity in plasma modeling is called Intrinsic Low-Dimension Manifold (ILDM).

The ILDM method was first elaborated upon in 1992 as a new way to simplify chemistry kinetics in combustion sciences. In comparison to most conventional methods at the time, ILDM didn’t require a priori assumptions with respect to species and reactions being in steady state or partial equilibrium. Besides this, the method was able to provide accurate results in regimes with relatively difficult conditions such as low temperatures for example. The ILDM method is based on a dynamical system approach. Hence, it aims to mathematically describe the time evolution of a system represented by linear and non-linear ordinary differential equations. The major advantage of this kind of approach is that it provides a clear insight in the reaction mechanisms of the system and prevents it from becoming a black box model. Another important feature of the ILDM method is its ability to automatically divide the system of reactions into slow and fast reaction groups. As will be explained in subsection 2.2.2, fast reaction groups typically increase the occurrence of stiffness in a system. In general this means that the presence of fast reactions will make it computationally more expensive to solve the system of equations over a longer period of time. By introducing steady-state approximations for the fast reaction groups, it is able to separate them from the system such that it becomes much easier to numerically solve the system of reactions represented by these differential equations.

In this chapter the ILDM method is discussed conceptually. Without going into too much mathematical detail, section 2.2 serves as a first step towards a better understanding of the ILDM method and as an introduction to frequently used jargon. Additionally, section 2.3 gives a clear description of the implementation of the method, which is done in MATLAB. It also explains the motivation behind several modeling choices. In section 2.4 a brief discussion is provided on the gains and limitations of the implementation. This section aims to elaborate on both the intrinsic limitations of the ILDM method as well as the limitations introduced because of modeling assumptions. Thereafter, a focus is put on the potential gains of the method and to explain why it is worth being explored further. Finally, a short summary is given in section 2.4.

2.2 Conceptual method

Although the ILDM method is used extensively in combustion sciences, it has only been applied to the field of plasma physics recently. It is important to mention that the thermal systems used in combustion sciences are significantly different from the low temperature plasma models used in plasma physics. An example is the larger variety of time scales, the presence of an electron temperature that may be very different from the ion temperature and the multidirectional gradients of a plasma. Due to this inherent different nature of plasmas, the ILDM method should be applied slightly different compared to its previous applications in combustion sciences. Some of the first steps in making the ILDM method suitable for plasmas is done in the work of (Rehman, 2018). This specific implementation is used as a cornerstone in the description of the ILDM method provided in this chapter.
2.2.1 Conceptual design of ILDM method

The design of the ILDM method is characterized by the small amount of required input data. The only information needed is a detailed reaction scheme with the corresponding reaction rate coefficients. Although these coefficients are involved in the determination of the time scales, it is important not to confuse them with each other. In order to clarify some basic concepts, an example of a simple argon plasma is given below.\[7\]

\[
\begin{align*}
Ar + e &\rightarrow Ar^* + e + E & : k_1 \\
Ar^* + e + E &\rightarrow Ar + e & : k_2 \\
Ar^* + E &\rightarrow Ar + [h\nu] & : k_3
\end{align*}
\]

The system stated is referred to as a two level argon plasma since there are two energy levels that can be occupied by an atom, the ground state and the excited state. If the atom in the higher energy state relaxes to the ground state, a photon can be emitted as shown in the bottom equation. The photon energy \([h\nu]\) and the electron energy \(E\) are explicitly stated. In this example three different species can be distinguished: \(Ar\), \(Ar^*\) and \(e\). The emitted photon, denoted with \([h\nu]\), is very common in plasma reactions and is not considered a species. The corresponding reaction rate coefficients are stated behind each equation and indicated with \(k_i\). Note that the time scales are different from the reaction rate coefficients and generally not known beforehand.

Complex plasmas usually contain many more different species and reactions that typically have a large variety of time scales in which they equilibrate. This introduces the problem of stiffness. However, short time scale processes equilibrate rapidly and are not of influence on the long term evolution of a system. One is interested in a method to separate these fast reactions from the system without altering the system mechanics in itself. The ILDM method delivers a significant contribution to this separation. It is able to transform a dynamical system into a parameterized system of equations that allows for reduction based on the time scales represent in the system. To understand the validity of this approach one could argue that the fast reactions in the system reach an equilibrium quickly. By reaching this equilibrium, the reaction serves no significant further contribution to the evolution of the system. However, as long as the reaction is present in the system it has to be dealt with in every time step and thereby drastically obstructs the computational efficiency of the simulation that describes the long term evolution of the system.

In terms of a geometric approach one could say that by separating these equilibrated reactions, the system can be represented in a lower dimensional composition space. I.e., the system can be described with less variables. As time progresses more reactions will reach an equilibrium and consequently the reaction trajectory can be represented in a composition space with a decreasing number of dimensions until it reaches the zero dimensional equilibrium point.\[22\] This state is referred to as the chemical equilibrium and is used as the starting point in determining the intrinsic low-dimensional manifold. This manifold captures the composition of the plasma after the short time scale processes have equilibrated. In practice different points on this manifold are determined and by interpolation the entire trajectory is constructed. The points describing the manifold are stored in look-up tables.\[18;22\]

2.2.2 System stiffness and time scale analysis

The phenomena of stiffness is mentioned several times already. This subsection goes into detail on this phenomena and explains how it is related to the different time scales in a system. Generally speaking, the stiffness of a system is expressed as the the ratio of the slowest time scale to the fastest time scale present in the system.\[8\] However, there exist examples that a high stiffness ratio does not give rise to the typical problems that are expected within stiff systems.\[23\] By using a linear dynamical system, the role of different properties of this phenomena are discussed and coupled to their implications with respect to the considered physical plasma systems.
Since plasma models are not necessarily linear differential equations, it is valuable to elaborate the relevance of this linear model with respect to the ILDM method. The ILDM method uses a linearization to approximate a system of non-linear differential equations by a system of equations that is locally linear. This linearized system can be represented by the product of the local Jacobi-matrix with the vector containing the species densities. This can be solved trivially compared to the initial system of non-linear differential equations. A simple mathematical example of such a linear system of differential equations is given based on a model by J.D. Lambert presented in his work on linear stability theory.\[23\]

\[
\frac{d}{dt} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 998 & -999 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}
\]

For these linear systems of differential equations it is rather easy to obtain the full solution. Typically the solution of such a system consists of at least one term at the form \(\exp(\lambda t)\). Here \(\lambda\) corresponds to one of the eigenvalues of the system and encloses important information for both the stability and the time scales of the system. This means that for eigenvalues greater then zero the solution of the differential equation will increase exponentially if time \(t\) proceeds and the system will thus become unstable. Other behaviour due to the properties of the eigenvalues is best explained using the solution stated below. Note that the eigenvalues are computed from the square matrix stated in equation (2.1) and have the corresponding values \(\lambda_1 = -1\) and \(\lambda_2 = -1000\). As will be explained in more detail later, this indicates that the initial system of equations is stiff. Solving equation (2.1) yields the full solution stated by

\[
\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = C_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} \exp(-t) + C_2 \begin{bmatrix} 1 \\ -998 \end{bmatrix} \exp(-1000t).
\]

For simplicity equation \(y_1\) is considered and both constants \((C_1\) and \(C_2)\) are chosen to be equal to one. Now it can already be seen that lambda has unit \(1/t\), a first indication that they are related to the time scales as will be explained later. If the full solution is analyzed, it is observed that the term with \(\lambda_1\) decays very slowly compared to the term with \(\lambda_2\). Therefore, \(\exp(\lambda_1 t)\) is indicated as the slow transient and \(\exp(\lambda_2 t)\) as the fast transient. When numerically computing an approximation of the exact solution given by \(y_1\), it is important to maintain stability and accuracy. However, this gives rise the problem that the simulation is forced to take excessively small time steps due to the presence of the fast transient. However, for computing the long term behavior of the slow transient this means that although the contribution of the fast transient is minimal, it must be taken into account until it reaches below the tolerance level of the simulation. This phenomena is referred to as stiffness. As explained in the previous section, the ILDM method’s goal is reducing the stiffness of a system by eliminating fast processes. Since it is claimed that these fast processes are of little influence for the long term evaluation of the system a figure is added to provide a better understanding on how the reduction of a system by the elimination of fast reactions can affect the long term solution. In figure 2.1 the evolution over time of the fast transient, the slow transient and the full solution are plotted on a logarithmic axis. As can be seen in the figure, the fast transient only delivers a significant contribution to the full solution in the very first time steps. Afterwards, the general behaviour is approximately equal to the evolution of the slow transient.

However, due to the presence of the fast transient the computation time can be excessively large. In the ILDM method it is possible to eliminate fast reactions from this linear system of differential equations by adopting a steady-state approach for the short time scale processes. From \(\exp \lambda t\), it can be derived that the unit of \(\lambda\) equals \(1/t\). Therefore, the time scale can be expressed as the inverses of the absolute value from the real part of an eigenvalue. This can be written in the form of equation (2.3). Time scale are thus a measure that indicate the rate of decay for a specific term in the general solution. The time scale with the largest \(\lambda\) and thus the fastest transient is denoted as \(\tau_{min}\), while the time scale with the smallest lambda and thus the slowest transient is denoted as \(\tau_{max}\). The order of the eigenvalues is defined as \(|Re(\lambda_1)| \geq |Re(\lambda_2)| \geq \ldots \geq |Re(\lambda_i)|\) with \(i\) a positive integer and \(\lambda_i \neq 0\).

\[
\tau_i = \frac{1}{|Re(\lambda_i)|}
\]

Note that in equation (2.3) only the real parts of the eigenvalues are taken into account. This can be justified relatively easily if the general solution for a Jacobi-matrix with complex eigenvalues is observed.
Let's assume the complex conjugated pair of eigenvalues $\lambda_1 = a + ib$ and $\lambda_2 = a - ib$ are obtained by solving the local Jacobi-matrix. The general solution for this situation is given by equation (2.4) with $C$ and $\tilde{C}$ complex coefficients:

$$
y = C_1 \exp(\lambda_1 t) + C_2 \exp(\lambda_2 t) = \exp(at)[\tilde{C}_1 \cos(bt) + \tilde{C}_2 \sin(bt)]$$

(2.4)

In equation (2.4) it can be observed that the exponential term only depends on the real part of the eigenvalue. This term is multiplied with a periodic function that implies that a decay in the exponential term will lead to a smaller amplitude of the periodic function. This periodic function will therefore decrease with the same rate of decay as the exponential term and therefore follows the same time scale, for purely imaginary functions the time scale equals the periodic time. Evidently, a positive value for $a$ will again result in unstable behaviour. To summarize, the following six cases depending on the eigenvalues can be obtained:

1. **Real and negative:** The term reaches a stable equilibrium
2. **Real and positive:** The term explodes and imposes instability to the system.
3. **Complex and negative real part:** The term introduces oscillations in the system and reaches a stable equilibrium
4. **Complex and positive real part:** The term introduces oscillations in the system and imposes instability to the system.
5. **Purely imaginary:** The term introduces oscillations in the system and imposes oscillations around the equilibrium point to the system.
6. **Zero:** The term does not change over time and the densities in the system remain constant.

Note that an eigenvalue of zero represents a term without a rate of decay and therefore it is not considered a time scale since it typically is a conserved quantity. Consequently, in order to have a stable system that reaches an equilibrium all eigenvalues must be negative or zero. One is now able to express the stiffness ratio of a system given by

$$S_t = \frac{\tau_{max}}{\tau_{min}}$$

(2.5)
2.2.3 Parametrization of the system

An important step in the ILDM method is the mapping of a reduced dynamical system onto a set of equations describing the parametrization. This way the set of equations only has to be solved for the specific parameters that corresponds to the number of species that are evaluated. After solving the system for these parameters it is possible to map this parametric solution back onto the desired solution by the use of a so called manifold, this will be explained in further detail in section 2.2.4. In this section the set of equations describing the parametrization \( g(n) \), stated in equation (2.6) are examined in further detail. This set of equations consists of three parts, from top to bottom: the manifold equation, the parameter equation and the element conservation equation. Here \( \omega_p \) represents a vector of parameters. Other parameters and variables are discussed below together with the specific equations. Mathematics that do not immediately serve the conceptual understanding are avoided in this section, a complete mathematical analysis of the ILDM method will be provided in chapter 3.

\[
g(n) = \begin{cases} AS(n) \\ Pn - \omega^p = 0 \\ \Phi n - \omega^c = 0 \end{cases}
\]  

(2.6)

The manifold equation

In the previous section the time scales of a dynamical system were discussed. As mentioned before, these time scales play an important role in reducing the system of differential equations. So far it has not been discussed how the ILDM method is able to perform this reduction based on these specific time scales.

Roughly speaking, the manifold equation part in equation (2.6) imposes the condition that all reactions in the dynamical system that correspond to a short time scale must be zero. The manifold equation captures the short timescale processes on a low-dimensional manifold; this approximation is equivalent to the statement that fast processes have finished and that therefore the configuration space shrinks in dimension. This condition is thus given by

\[
AS(n) = 0.
\]  

(2.7)

Here two new variables are introduced, the source vector \( S \) as a function of the species densities and the reduced matrix \( A \). The source generally represents a complicated function that precisely describe how all species interact with each other and in what rate they are produced or destroyed. In chapter 3 it will be explained in more detail how this source vector is obtained via the reaction set and rate coefficients. The reduced matrix exist of the transposed eigenvectors that belong the to eigenvalues of the Jacobi-matrix with the shortest time scales. The amount of short time scale processes that can be removed depends upon the dimension of the manifold and the number of different elements present in the complete reaction scheme. Note that for non-linear dynamical systems the Jacobi-matrix change for different densities such that the eigenvalues and thus the reduced matrix are not constant over time.

The parameter equation

The parameter equation part of equation (2.6) depends on modeler input. First of all it enables the modeler with the option to choose the dimensionality of the manifold and secondly it is of vast importance for the smoothness of the manifold curve that is obtained. The number of parameters that are present in equation (2.8) is equal to the number of dimensions of the manifold.

\[
Pn - \omega^p = 0
\]  

(2.8)

Here \( P \) represents a \( N_d \times N_s \) matrix where \( N_d \) equals the number of dimensions and \( N_s \) the number of species. \( \omega^p \) represents a vector of length \( N_d \) with values that have to be chosen by the modeler. When speaking of dimensions one refers to the number of parameters for which the manifold is evaluated. So for three different parameters one can obtain a 3 dimensional manifold curve embedded in a \( N_s \)-dimensional space configuration. However, the more species that are taken into account the more time consuming
it becomes to compute the manifold. The species for which the manifold is evaluated is determined by matrix $P$. If a system has three different species a two dimensional parameter equation could be obtained by using the matrix

$$P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$  (2.9)

As shown above we add a new row for every dimension with exactly one 1 that corresponds to the species that is included. So in this case the first and the last species with respect to their ordering in the vector $n$. The final step is to adjust the parameters in such a way that different densities on the manifold can be obtained by solving the system of equations $g(n)$. Here the initial value from the parameter of the considered species should be equal to its density in the chemical equilibrium point. The point from which it is known that it is located on the manifold as explained in section 2.2.1. Subsequently, the parameters can be adjusted to find the points on the manifold. The adjustment of this parameter should be done with caution as can be explained with figure 2.2.

![Figure 2.2](image)

Figure 2.2: Hypothetical linear interpolation of curve representing a $y = \sin(t)$ shape for a large, medium and small step size of the parameter.

By adjusting the parameter equation different points on the manifold curve are generated. However, if the step size between two parameters is too large an interpolation of points will lead to the wrong manifold curve as demonstrated in figure 2.2. On the other hand, if the step size is very small it will take much longer to compute all points. This asks special attention when scaling up the dimensions of the manifold since the total number of points that have to be computed scales with $O(N^{Nd})$ where $N$ is the number of generated points in one dimension.

The element conservation equation

For physical systems there must be element conservation, i.e. no elements are destroyed or created spontaneously. By element a chemical element is meant or if preferred a conserved building block of elements that is not decomposed during the reaction. However, the electrons that are frequently present in plasma reactions are also considered a single element in this method. This conservation law is imposed by the element conservation relation given by

$$\Phi n - \omega^e = 0.$$  (2.10)

In this equation $\omega^e$ represents a vector that represents the total amount of each element in a closed system. $\Phi$ represents the matrix that indicates what elements a species is build with. As an example an hypothetical closed system is considered with species $A, A^+$ and $B$, each having an initial density of one. Here two different elements are observed, $A = A^+$ and $B$. This situation can be represented in the following way using the elemental conservation equation.

$$\Phi n - \omega^e = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} A \\ A^+ \\ B \end{pmatrix} - \begin{pmatrix} 2 \\ 1 \end{pmatrix} = 0.$$  (2.11)
Using the top row of matrix $\Phi$ it can be observed that there are two elements of type $A$ in the system. One in the species of the first column and one in the species of the second column. In the same manner it can be observed that there is one element $B$ in the system that belongs to the species of the third column. In this example the density of the species is used, this can also be expressed using mass fractions instead. This would imply that $\sum_i^m \omega_i^e = 1$, where $n$ equals the number of elements.

2.2.4 Determination and use of manifold

After the system is parameterized correctly, it is solved for the species densities given the chosen parameter. The list of species densities in terms of the parameter are stored in a so called look-up table. However, this table does not have a value for every value of the parameter and therefore a fit or interpolation is necessary such that every species can be expressed as

$$n = n(n_p).$$  \hfill (2.12)

Here $n_p$ represent density of the parameter species. It is important that this dependency is continuous, since it is used to solve the dynamical system for the one or more parameters that are the only species densities in this parameterized system expressed as

$$\frac{\partial n_p}{\partial t} = S_p(n(n_p)).$$  \hfill (2.13)

However, before a manifold can be created manifold points have to be generated through solving the system $g(n)$. It makes sense to first compute the one-dimensional manifold before starting with the computations of a two-dimensional manifold because it is always located on the two-dimensional manifold. Every point on the one-dimensional manifold thus serves as a starting point in searching for points on a higher dimensional manifold. If the entire one-dimensional manifold is known, the search for higher dimensional point can be done simultaneously with every entry of the one-dimensional manifold as a different initial configuration for the parameters in system $g(n)$. This approach allows for parallel computations. This doesn’t decrease computational costs, but it can significantly decrease time if enough computational power is available.

There are multiple ways to generalize the relation between the different points on the manifold. Basically, one is trying to fit a curve that perfectly captures all points on the manifold in terms of the parameter. However, if it is not possible to fit such a curve, different interpolation techniques can than be useful. For example, assuming that there is a straight line between every point. Especially if the obtained manifold points are close enough to each other this can be a pretty accurate approximation. Figure 2.4 gives an idea of how a surface can be interpolated by such a method.

(a) Actual surface.  
(b) Approximated surface.

Figure 2.4: Interpolation of a surface based on a specific set of points.

If equation (2.13) is purely written in terms of the parameter by using these interpolation techniques, it can be solved such that $y_p(t)$ is obtained. Since equation (2.12) is already known the time evolution of other species densities can now be computed as $n(t) = n(n_p(t))$. 

9
2.3 Implementation

The programming language used for the implementation of this method is MATLAB. This is mainly because of convenience. Some of the first steps of the implementation were already written in MATLAB and given the relative short period of time, this was a very welcome head start. An alternative programming language would have been C++, this is the most frequently used language in the group for Elementary Processes in Gas Discharges at Eindhoven university. However, a lack of experience with this language would obstruct the main goal of this project.

Within this implementation an object oriented approach is adopted. Because of this approach, the input file with the initial properties of a plasma can be separated from the main script and easily be exchanged with an input file of choice which are implemented as class definitions in MATLAB. In this way the work in this thesis can be reproduced effortlessly, while at the same time other modelers can use the backbone of this implementation for different plasma simulations. Another cornerstone in the design is the fact that all functions are stored in a separate class as well. This has as major advantage that functions can be tested individually and also be optimized or exchanged without changing the main model.

Conclusively, there are three required files in order to run this simulation. First, the input file with all the properties specific to the plasma system. Second, the function directory file containing all the manually designed functions used in the model. And finally, the main file that generates the manifold. In the following part, some important details for the implementation are discussed using the three level hydrogen model as an example. The complete input files can be consulted in the appendix.

Configuration of main file

In the lines of code below the typical way of configuring the main model is shown. First, the modeler can couple the input file of choice to the the variable `inputFile`. This will load the object that contains all relevant information and stores this in the `inputFile` variable. Second, a time frame for the simulations is chosen and the last step is to estimate an initial value for the chemical equilibrium. This step is explained in further detail in chapter [3]. For now it is important to note that the number of entries in this array is equal to the number of species in the system. In this specific case there are three species: $H$, $H^*$ and $H^+$. Therefore, the initial guess array contains three entries.

```matlab
% Configuration of main file

% The input file with corresponding properties of the plasma system
inputFile = hydrogen.inputFile();

% The time frame for which the system is solved
T = [0 1e-4]; %[s]

% Initial guess of chemical equilibrium
initGuess = [57624e-3 699097e3 9300903e3]; %H, H^*, H^+ [m^-3]
```

Configuration of function directory

As mentioned before, one of the advantages of this model is the option to optimize or test functions in the function directory without changing the main file. The code below is an example of such a function used in the ILDM method. New functions can be added to the directory in a similar way. To call a function one can use the MATLAB environment with the command: `funDirectory.gradientMatrix(input1,input2)`. Here `input1` and `input2` represent the input arguments stated at line 10 and 11. To adjust this function one could change lines 14 to 33, if line 13 is changed the main model should be revised as well.

```matlab
% Configuration of function directory

% Name: gradientMatrix(inputFile, densities)
% -- Description --
% This function is able to numerically compute the gradient matrix
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
Configuration of input file

In the input file there are two different parts, the properties and the methods. The part with properties represents all the different constant values in the ILDM method. For example the number of species and the initial densities of these species. However, many of these constant properties can be derived from other constants. The number of species can be derived from the number of rows in the stochiometric matrix for example. To avoid mistakes this input file only asks for the most important input values and is then able to derive all other constants itself. It is good to note that a change in class definition will break these derivations and as far as the MATLAB implementation goes it is thus recommended to carefully check these derivations when changing the class definition. Furthermore, it is of vast importance that only the values are changed, or, the right hand sides of the equations. The variable names itself are linked to the main model and a change in these names is only possible if the entire model is revised afterwards.

The second part of the input file consists of methods. I.e., functions with specific information of the modeled system. For example the source equation and the names of the species that show up in the figures created by the main script. Above every function in this part a specific instruction is stated on how to properly adjust this to another system and will therefore not be discussed in further detail here.

2.4 Practical aspects of implementation

For the designed implementation there are some limitations that should be addressed. First, it is assumed that both the gas temperature and the electron temperature are constant during the entire process. Furthermore, the Jacobian is reevaluated in every step, but if the step size is to large this linearization does not hold anymore for non-linear systems. Also, the method can only identify the zero-dimensional manifold if the system reaches an equilibrium. Finally, the non-linear solver fSolve that is used for solving non-linear systems has limited capabilities in doing so; an initial guess is required and a large difference in magnitude of species densities or the presence of complex numbers can cause problems in finding physically interesting values.
2.5 Summary

An increased interest in the use of plasmas demands a more thorough understanding of this topic. This can be done with the use of computational simulations that solve systems of coupled differential equations that describe the system kinetics of a chemical process. These simulations tend to get computationally expensive rather easily and Intrinsic Low-Dimension Manifold (ILDM) can help to reduce the computational costs by decreasing system stiffness and a parametrization of the dynamical system. This method provides clear insights and requires little input.

The required input consists of a detailed reaction scheme with corresponding reaction rate coefficients. These reaction schemes generally consist of many different species and reactions. An important part in the ILDM method is separating the fast time scale reactions from the system to impose a steady state approximation on these reactions. This is done automatically without *a priori* assumptions. After reduction a manifold is created that captures all systems kinetics of a system expressed in one or more specified parameters.

The phenomena of stiffness increases computational costs because it forces a solver to take excessively small steps due to the presence of a fast time scale process that is much faster than another slow time scale process in the system. For a stiff system the fast transient contains little information on the long term evolution of a system. The time scales of a system can be identified by an analysis of the eigenvalues of the Jacobian matrix of the dynamical system.

The parametrization of the system is dependent upon three different equations, the manifold equations that imposes a steady state approximation on the fast time scale processes, the parameter equations that allows the modeler to choose in what parameter the manifold should be expressed and what the dimension of the manifold must be, the element conservation equation that ensures that no elements are produced or destroyed spontaneously.

From the manifold, a parametrization can be derived to express all species in terms of the parameter species. This reduces the system of equations to the equations that belong to the parameter only. The interpolation that allows for the parametrization of the manifold can be done in several ways and is generally more accurate if the obtained manifold points lay closer together.

The ILDM method in this thesis is implemented in MATLAB. A object oriented approach is adopted such that the input file can be exchanged easily with an input file of choice. This ensures consistent and reproducible results. This input file is coupled with a main file that allows for computations of manifolds up to two dimensions. The final object to be connected with the main script contains all different functions in the method, in this way functions can be tested or optimized without altering the main method in itself.

Finally, the design is limited by the electron temperature that is assumed to be constant. Also, a non-linear solver is used that has limited capabilities with respect to the required input parameters and the complexity of the system to be solved.
3 Mathematical aspects of method

3.1 Introduction

For the simulation of plasma fluid models, the Boltzmann equation is of great importance. This equation describes the time evolution of different species in a plasma and how they interact with each other. In this thesis, transport properties of this equation are neglected to simplify computations. This chapter explains some basic mathematical concepts that serve as the cornerstone of this modelling approach. Furthermore, essential mathematics and numerical approaches are discussed that play an important role in the method. At the end of this chapter, a case study is provided that explains how the ILDM technique can be applied to a hydrogen system. The mathematical derivations in the chapter are based on the work of T. Rehman [8].

3.2 Mathematical aspects of method

3.2.1 Species densities and mass fractions

In this thesis species densities and species fractions are used by each other. Since a significant part of this work is concerned with linear systems, these can be used interchangeable. However, non-linear problems ask for a more careful approach. Both are used frequently and for a system with \( N_s \) species the species densities vector \( \mathbf{n} \) is represented as

\[
\mathbf{n} = \begin{bmatrix}
n_1 \\
n_2 \\
\vdots \\
n_{N_s}
\end{bmatrix}
\]  

(3.1)

In many cases it can be easy to solve the system in terms of species fractions instead. The mass fraction vector \( \mathbf{y} \) is represented as

\[
\mathbf{y} = \begin{bmatrix}
y_1 \\
y_2 \\
\vdots \\
y_{N_s}
\end{bmatrix}
\]  

(3.2)

If one is interested to go from densities to fractions this is possible with the relation between both given by

\[
y_i = \frac{m_i n_i}{\sum_{j=1}^{N_s} m_j n_j}.
\]  

(3.3)

Here \( m \) represents the atomic mass of a specific species. The definition of species fractions is particularly useful since it allows for an easy way to verify if mass is conserved in a system expressed by

\[
\sum_{i=1}^{N_s} y_i = 1.
\]  

(3.4)

3.2.2 Reaction source terms

Boltzmann equation and particle balance

The particle balance is the most important equation throughout the ILDM method. This equation captures all information on how species densities change over time as a result of chemical interactions and therefore, it explains how a conserved system is maintained by describing what species are produced and destroyed. The particle balance equation is derived from the Boltzmann equation given by

\[
\frac{\partial f_i}{\partial t} + \vec{v} \cdot \nabla f_i + \frac{\vec{F}}{m_i} \cdot \nabla \sigma f_i = \left( \frac{\partial f_i}{\partial t} / c \right)
\]  

(3.5)
Here \( f_i(\vec{r}, \vec{v}, t) \) represents the particle distribution function of species and \( \vec{F} \) denotes for the different forces that act on the particle. On the left hand side, the different terms from left to right denote for the temporal evolution, the spatial gradients and the change is species distribution due to forces. On the right hand side the temporal evolution of species distribution because of collisions is stated. In this thesis the so called zeroth moment of the Boltzmann equation is considered. This zeroth moment is obtained by integrating equation (3.5) over the velocity coordinates and defines the particle balance in the system,

\[
\frac{\partial n_i}{\partial t} + \nabla \cdot (n_i \vec{u}_i) = S_i. \tag{3.6}
\]

Here \( \vec{u}_i \) represented the averaged velocity of particles, and \( S_i \) is referred to as the source term. The source term represents the production and destruction rate of species. The second term of the equation represents the transport in the system, in this thesis transport is neglected such that

\[
\frac{\partial n_i}{\partial t} = S_i. \tag{3.7}
\]

This equation will be used extensively throughout this paper. The right hand side of this equation is generally obtained via the product of the stoichiometric matrix \((\nu)\) and the reaction rate vector \((Z)\) as will be explained in more detail in the following two sections.

**Stoichiometric matrix and volumetric rate**

The reaction scheme of a plasma system containing of \( N_s \) different species \([A]_i\), can be expressed mathematically as

\[
\sum_{i=1}^{N_s} v^L_{ij}[A]_i \rightarrow \sum_{i=1}^{N_s} v^R_{ij}[A]_i \tag{3.8}
\]

Here \( v^L_{ij} \) and \( v^R_{ij} \) denote for the number of consumed and produced species in reaction \( j \). These constants are referred to as the stoichiometric coefficients and make up the so called stoichiometric matrix \( \nu \) with elements

\[
v_{ij} = v^R_{ij} - v^L_{ij}. \tag{3.9}
\]

In order to obtain a complete equation for the production and destruction of species the volumetric rate in which these reactions take place is also important. For a reaction \( j \) this is given by

\[
Z_j = k_j \prod_{i=1}^{N_s} n_i^{v^L_{ij}} \tag{3.10}
\]

Here \( k_j \) represents the reaction rate coefficient and \( n_i \) the species densities of a species \( i \). If these expressions are merged it is possible to express the source term for \( N_r \) reactions and \( N_s \) species as

\[
S_i = \sum_{j=1}^{N_r} v_{ij} Z_j. \tag{3.11}
\]

And therefore,

\[
\frac{\partial n}{\partial t} = S = \nu Z \tag{3.12}
\]

### 3.2.3 Jacobian linearization

As mentioned in the previous chapter, the ILDM method uses a linearization of the source to approximate a non-linear system of differential equations to a system that is locally linear. This makes it much easier to solve such a system of non-linear differential equations. This linearization is performed with the help of a Taylor expansion of the source as

\[
S(n) = S(n^0) + \frac{\partial S(n^0)}{\partial n} (n - n^0) + \mathcal{O}(\delta n)^2. \tag{3.13}
\]
This expansion can be simplified by neglecting the higher order terms and introducing the Jacobian matrix of the source vector \( J \) as

\[
S(n) \approx S(n^0) + J|_{n_0}(n - n^0).
\]  

(3.14)

This Jacobian matrix can also be written in terms of the stoichiometric matrix and the volumetric rate vector as

\[
J = \frac{\partial S}{\partial n} = \nu \frac{\partial Z}{\partial n} = \nu \mathcal{G}.
\]  

(3.15)

Here the gradient matrix \( \mathcal{G} \) is introduced since in equation (3.11), \( Z \) is expressed as a function of \( n_i \) and the gradient can therefore be computed analytically. The different elements of this gradient matrix can thus be computed as

\[
G_{ij} = \frac{\partial Z_i}{\partial n_i} = \begin{cases} 
0 & \text{if } v_{L,i,j} = 0 \\
\nu_{L,i,j} - v_{L,i,j} & \text{if } v_{L,i,j} \neq 0 
\end{cases} n_{\prod_{k=1; k \neq j}^{N_s} n_k}. 
\]  

(3.16)

In order to use this equation in the analytical notation analytically, the two following requirements must be satisfied:

1. The rate coefficient does not depend on the densities.
2. If the reaction is reversible, the forward and backward processes must be dealt with separately (no detailed balancing is employed).

### 3.2.4 Schur decomposition

In the ILDM method, the fast time scale process are identified and separated from the system by a analysis of the Jacobian matrix. This is done with the help of a Schur decomposition given by

\[
J = Q N Q^\top.
\]  

(3.17)

Here \( Q \) represents the Schur vector matrix with the unitary matrix \( Q^\top = Q^{-1} \). \( N \) represents a upper triangular matrix with the arranged eigenvalues on the diagonal in decreasing order of magnitude. From the unitary matrix \( Q^\top \) the fast \((Q^\top_s)\) and slow \((Q^\top_s)\) time scale processes can be separated as

\[
Q^\top = \begin{pmatrix} Q^\top_s \\ Q^\top_j \end{pmatrix}.
\]  

(3.18)

The Schur eigenvectors are thus ordered based on the time scale of the corresponding reaction. Depending on how many dimensions are taken into account, a steady state approximation is imposed to a complementory number of bottom rows of this matrix. Therefore, matrix containing the Schur eigenvectors that are separated from the system is called the reduced matrix \( A \). Mathematically this is expressed as

\[
Q^\top_j S(n) = A S(n) = 0.
\]  

(3.19)

### 3.3 Case Study I: A three level hydrogen system (1D)

The implementation of the ILDM method up to this part is very similar to the work presented by T. Rehman. In this section a three level hydrogen system is reproduced that is introduced originally by E.H. Kamaneci. To verify if the results obtained by using the discussed implementation, correspond with the results in earlier work. This model is also discussed in the work of T. Rehman. Furthermore, this section aims to provide at the same time a clear explanation on how the implemented method comes into practice. It should be mentioned that the discussed model is hypothetical, however, because of its simplicity it best serves the purpose of a first example.
The stoichiometric matrix can be derived from the reaction set (3.20). For example, in the first reaction

\[ H + e \rightarrow H^* + e \quad : k_1 = 2.7 \cdot 10^{10} \text{ [s}^{-1}\text{]} \]  \hspace{1cm} (3.20)

This results in the dynamical system represented in equation (3.22)

\[
\begin{align*}
\frac{d}{dt} & (y_1, y_2, y_3) = \\
& = \begin{pmatrix}
-1 & -1 & 0 & 0 & 1 & 1 \\
1 & 0 & -1 & 1 & -1 & 0 \\
0 & 1 & 1 & -1 & 0 & -1
\end{pmatrix}
\begin{pmatrix}
k_1 y_1 \\
k_2 y_1 \\
k_3 y_2 \\
k_4 y_3 \\
k_5 y_2 \\
k_6 y_3
\end{pmatrix}
= \begin{pmatrix}
-k_1 y_1 - k_2 y_1 + k_3 y_2 + k_6 y_3 \\
k_1 y_1 - k_3 y_2 + k_4 y_3 - k_5 y_2 \\
k_2 y_1 + k_3 y_2 - k_4 y_3 - k_6 y_3
\end{pmatrix} \hspace{1cm} (3.22)
\end{align*}
\]

The transition scheme with corresponding transition frequencies of the considered hydrogen model is given below.

\[
\begin{align*}
H + e & \rightarrow H^* + e \quad : k_1 = 2.7 \cdot 10^{10} \text{ [s}^{-1}\text{]} \quad \hspace{1cm} (3.20) \\
H^* + e & \rightarrow H^+ + e + e \quad : k_2 = 9.0 \cdot 10^8 \text{ [s}^{-1}\text{]} \\
H^* + e & \rightarrow H^+ + e + e \quad : k_3 = 1.0 \cdot 10^6 \text{ [s}^{-1}\text{]} \\
H^+ + e & \rightarrow H^* + e \quad : k_4 = 7.5 \cdot 10^4 \text{ [s}^{-1}\text{]} \\
H^* + e & \rightarrow H + e \quad : k_5 = 3.8 \cdot 10^1 \text{ [s}^{-1}\text{]} \\
H^+ + e & \rightarrow H + e \quad : k_6 = 1.7 \cdot 10^2 \text{ [s}^{-1}\text{]} \\
\end{align*}
\]

In this model three different energy levels of hydrogen are taken into account. The ground level given by \(H\), an intermediate level \(H^*\) and the ion level \(H^+\). Note that in the work of T. Rehman the electrons in these reactions are neglected, i.e. the electron density is taken to be constant. This implies that there is only one element and three species instead two elements and four species. A reason to neglect these electrons is the fact that they introduce non-linearity in the system which makes things more complicated. For example, in a linear system the Jacobi-matrix remains independent of the species densities and therefore the reduced matrix \(A\), in equation (2.7), as well. However, to obtain similar results and maintain simplicity this approach is preserved. Since most plasma physical system are not linear, an additional section will be added to provide detailed information on how non-linear systems differ from the discussed one.

**Mathematical setup of dynamical system**

In this case a vector \(y\) is defined as \(y = (H_y \ H^*_y \ H^+_y)^T\). Furthermore, mass fractions will be used instead of species densities. This makes it easier to determine if the condition for mass conservation remains valid during calculations. The initial mass fractions that are used are \(H_y = 1.0 \cdot 10^{-2}, H^*_y = 3.0 \cdot 10^{-1}\) and \(H^+_y = 6.9 \cdot 10^{-1}\). Note that \(\sum_{i=1}^{N_i} y_i = 1\) and \(n\) is replaced by \(y\) to clarify that it concerns mass fractions instead of densities. The following step is to setup the dynamical system that is now represented by

\[
\frac{dy}{dt} = S(y). \hspace{1cm} (3.21)
\]

The source equation \(S(y)\) can be computed as the product of the stoichiometric matrix and the vector that contains the product of the consumed species and the rate vector. Every column in the stoichiometric matrix represents a reaction from the reaction set such that a negative sign represents a consumed species and a positive sign a produced species. If a column is then multiplied with the corresponding species and reaction rate coefficients it is exactly known in what speed different species are produced or destroyed. This results in the dynamical system represented in equation (3.22)

\[
\frac{d}{dt} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix}
-1 & -1 & 0 & 0 & 1 & 1 \\
1 & 0 & -1 & 1 & -1 & 0 \\
0 & 1 & 1 & -1 & 0 & -1
\end{pmatrix} \begin{pmatrix} k_1 y_1 \\ k_2 y_1 \\ k_3 y_2 \\ k_4 y_3 \\ k_5 y_2 \\ k_6 y_3 \end{pmatrix} = \begin{pmatrix}
-k_1 y_1 - k_2 y_1 + k_3 y_2 + k_6 y_3 \\
k_1 y_1 - k_3 y_2 + k_4 y_3 - k_5 y_2 \\
k_2 y_1 + k_3 y_2 - k_4 y_3 - k_6 y_3
\end{pmatrix} \hspace{1cm} (3.22)
\]

The stoichiometric matrix can be derived from the reaction set (3.20). For example, in the first reaction of the reaction set it can be seen that 1 atom of the species \(H\) results in exactly 1 atom of the species \(H^*\). This implies that an amount of \(y_1 k_1\) per second of species \(H\) is used to produce \(H^*\) in the same amount per second. As a result the mass fraction of \(H\) at a certain moment in time depends on the mass fraction of \(H^*\) at that same moment in time and vice versa. This is now represented by the set of coupled differential equations given in equation 3.22. The right hand side of this equation represents the source vector.
System stiffness and time scale analysis

The next step is to obtain the different time scales in the system such that it becomes possible to separate the short time scale processes from the system. This can be done by analyzing the Jacobian \( J \) which can be computed analytically using

\[
J \overset{\text{def}}{=} \frac{\partial S(y)}{\partial y} = \nu \mathcal{G}.
\]

Here \( \nu \) denotes for the stoichiometric matrix and \( \mathcal{G} \) the gradient matrix. Here the analytical expression is used to compute the gradient matrix. Therefore, the following two requirements must be satisfied:

1. The rate coefficient does not depend on the densities.
2. If the reaction is reversible, the forward and backward processes must be dealt with separately (no detailed balancing is employed).

These requirements are satisfied and via the direct analytical method using the gradient matrix, the following Jacobian matrix is obtained.

\[
J = \begin{pmatrix}
-(k_1 + k_2) & k_5 & k_6 \\
k_1 & -(k_3 + k_5) & k_4 \\
k_2 & k_3 & -(k_6 + k_4)
\end{pmatrix}.
\]

Because of linearity the Jacobian is represented by a constant matrix. By performing an ordered Schur decomposition, the eigenvalues and corresponding Schur vectors of the Jacobian matrix will be obtained. This step is taken care of inside the MATLAB function \texttt{schur} and yields the result

\[
J = QNQ^T.
\]

Here the transposed Schur vector matrix \( Q^T \) is given by

\[
Q^T \approx \begin{pmatrix}
-6.1 \cdot 10^{-9} & -7.5 \cdot 10^{-2} & -1.0 \cdot 10^0 \\
9.0 \cdot 10^{-10} & 1.0 \cdot 10^{-1} & -7.5 \cdot 10^{-2} \\
-1.0 \cdot 10^4 & 1.4 \cdot 10^{-9} & 6.1 \cdot 10^{-9}
\end{pmatrix}.
\]

And matrix \( N \) with the eigenvalues of the Jacobian on the diagonal is given by

\[
N \approx \begin{pmatrix}
-4.2 \cdot 10^{-11} & -9.2 \cdot 10^5 & 2.9 \cdot 10^9 \\
0.0 & -1.1 \cdot 10^6 & -2.7 \cdot 10^{10} \\
0.0 & 0.0 & -2.8 \cdot 10^{10}
\end{pmatrix}.
\]

The eigenvalues are ordered in such a way that \(|Re(\lambda_1)| \leq |Re(\lambda_2)| \leq ... \leq |Re(\lambda_i)| \) where \( i \) is an integer greater than zero and 3 in this specific case. It can be observed that \( \lambda_1 \) is excessively small compared to the other values what indicates that this value is most likely the result of numerical errors in our model. For this reason this eigenvalue is said to be zero and thus corresponds to a conserved species mass fraction. From the other eigenvalues the time scales \( \tau_{\text{min}} = 1/\lambda_3 = 3.6 \cdot 10^{-11} \) and \( \tau_{\text{max}} = 1/\lambda_2 = 9.1 \cdot 10^{-7} \) are obtained via equation (2.3). These time scales are approximately equal to the time scales expected from the work of T. Rehman. Consequently, the stiffness ratio of this hydrogen system approximately equals \( 2.5 \cdot 10^4 \) as follows from equation (2.5) .

Parameterization of the system

In order to compute the parametric mass fractions the parameterized system must be composed. This is established by constructing the manifold equation, the parameter equation and the element conservation equation.

The manifold equation in this problem is the product of a constant reduced matrix with the source vector that is represented by equation (3.28).

\[
AS(y) = Q^T_j S(y) = 0
\]
The only unknown in this expression is the fast Schur matrix $Q_f^T$. From the transposed Schur matrix stated in equation (3.26) it is known that the top rows belong to the slow time scales processes while the bottom row belongs to the short time scale processes. One is interested removing as many short time scale processes from the system what corresponds to including as many bottom rows of the transposed Schur matrix as possible. The maximum amount of rows that can be included equals $N_s - N_e - N_d$. Here $N_s$ is the number of species, $N_e$ the number of elements and $N_d$ the number of dimensions. The matrix with the the maximum amount of bottom rows is called $Q_f^T$. In this case there are three species from one specific element and a one dimensional manifold is constructed. Therefore, the complete manifold equation is given by

$$Q_f^T S(y) = \begin{pmatrix} -1.0 \cdot 10^4 & 1.4 \cdot 10^{-9} & 6.1 \cdot 10^{-9} \end{pmatrix} S(y) = 0. \quad (3.29)$$

Since a one dimensional manifold is created $P$ has one row. Furthermore, the manifold for the intermediate level is constructed such that it corresponds to the second species and thus the second entry of $P$ becomes 1. Note that the parameter will thus become a scalar instead of a vector. The resulting parameter equation is given by

$$Py - \omega_p = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix} y - \omega_p = 0. \quad (3.30)$$

In order to know the initial value of the parameter $\omega_p$ the chemical equilibrium must be known since this chemical equilibrium point is in essence the zero dimensional manifold. In chapter 4 a method is introduced to obtain this value without solving the dynamical system. For now the ode15s solver is used to find the mass fraction in the chemical equilibrium of the intermediate level, $H^*$. According to the implemented model this equals $6.9 \cdot 10^{-2}$. This parameter $\omega_p$ is adjusted with a step size of $4.0 \cdot 10^{-11}$ to ensure that two consecutive fractions lay within a range of 5 percent from each other.

The final step is to derive the elemental matrix and the total amount of species in the system. Since the system consists of only of the element hydrogen the elemental matrix will have only one row. Every species is represented by a single element and therefore the final elemental matrix is just a vector of length three with a 1 on every entry. As mass fractions are used the mass conservation can be expressed by $y_1 + y_2 + y_3 = 1$, i.e. the sum of mass fractions must equal 1 such that the final element conservation equation can be expresses as

$$\Phi y - \omega_e = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} y - 1 = 0. \quad (3.31)$$

All together this yields the set of equations describing the parametrization of the system that is given by

$$g(y) = \begin{bmatrix} -1e-4 & 1.4e-9 & 6.1e-9 & 0 \\ 0 & 1 & 0 & -1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \omega_p \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (3.32)$$

**Determination and use of manifold**

From the manifolds points that are obtained using the implemented method, it is observed that there is a clear linear dependency between both species and the parameter. This one-dimensional manifold is shown in figure XX, where the chemical equilibrium point is indicated with the blue point. Since it is a linear system the Jacobian matrix is constant. Therefore, it is possible to solve the system (3.32) such that $y = y(y_p)$ can be expressed in terms of a straight line or surface as solution given by

$$y_1(y_2) = 5.76 \cdot 10^{-9} + \frac{y_2 - 0.0699}{-0.66} \cdot 3.12 \cdot 10^{-9} \quad (3.33)$$

and

$$y_3(y_2) = 0.9301 + \frac{y_2 - 0.0699}{-0.66} \cdot 0.66. \quad (3.34)$$
By using these relations for the species densities in terms of the parameter. Therefore the manifold is given in terms of a parametric equation as \((H, H^*, H^+) = (y_1(y_2), y_2, y_3(y_2))\) and the initial system can be expressed in terms the parameter as

\[
\frac{dy_p}{dt} = S_p(y(y_p)) = k_1y_1(y_2) - k_3y_2 + k_4y_3(y_2) - k_5y_2.
\] (3.35)

This differential equation can be solved analytically with the initial value \(y_2 = 0.3\). Consequently, the evolution of \(y_2(t)\) is obtained and with this also, \(y_1(t) = y_1(y_2(t))\) and \(y_3(t) = y_3(y_2(t))\). The results that are obtained via the implemented work are compared to previous results in the figures below.

(a) Results implemented method.
(b) Results from T. Rehman.

Figure 3.2: Comparison of one dimensional three level hydrogen manifold with respect to the results obtained in the work of T. Rehman.

(a) Results implemented method.
(b) Results from T. Rehman.

Figure 3.4: Comparison of the reduced solution of the hydrogen species in the ground state.

(a) Results implemented method.
(b) Results from T. Rehman.

Figure 3.6: Comparison of the reduced solution of the hydrogen species in the excited state.
Differences between linear and non-linear systems

As mentioned before, the considered hydrogen system represent a linear system. However, in many cases a genuine plasma model is not linear. Therefore, this section serves a short summary of the major differences between a linear and non-linear system with respect to the applied ILDM method.

A system becomes non-linear if there are two or more species consumed in a reaction. In such a case, the entries in the source vector typically contain terms with at product between one or more different species fractions. In contrast to a linear system, non-linear systems can have multiple stable chemical equilibrium points. This implies that in non-linear systems, a difference in initial densities can lead to a distinct solution and thus another manifold. The Jacobi-matrix of a non-linear system will always contain one or more elements that are not constant and depend on one or more time dependent species densities. Therefore, the Jacobi-matrix from a non-linear system changes as the reaction proceeds and the time scales are not guaranteed to remain the same. As an result the reduced matrix $A$, that is dependent on the short time scale eigenvectors of the Jacobi-matrix, also changes over time and should be adjusted together with the changing densities.
4 Determination of chemical equilibrium

4.1 Introduction

As time progresses more and more reactions in the system will reach an equilibrium and fall down on the manifold as explained in section 2.2.1 and visualized in figure 2.1. If the system is able to reach a stable equilibrium, the densities of the different species in this point will be constant. At this point both the fast and slow processes have equilibrated and are represented in a zero dimensional manifold. Since the number of dimensions decreases with every reaction that reaches an equilibrium, one could say that the minimum number of dimensions or the intrinsic low dimensional point is now obtained. A chemical equilibrium point that must certainly lay on the intrinsic low-dimensional manifold and can thus be used to serve as a starting point for determining the entire low dimensional manifold. However, in order to obtain this chemical equilibrium point previous studies in the field of plasma physics relied on solving the time evolution of non linear dynamical system such that one can observe at which densities the species become stable. As stated in section 1.2 these plasma systems are generally rather complex, as such, solving a system before a reduction is applied can be excessively expensive.

In this section an alternative method is introduced that allows the modeler to identify the stable equilibrium points without numerically evaluating the non linear dynamical system over time first. This approach is based on a stability analysis of the source. By setting the source vector to zero one can compute with the help of a linear or non-linear solver for what densities the source is independent of time. These solutions can then be investigated for stability by the eigenvalues of the corresponding local Jacobi-matrix. The eigenvalues of the local Jacobi-matrix have been mentioned before while speaking about system stiffness in section 2.2.2. As explained these eigenvalues play a major role in the solution of the linearized system of equations and a positive eigenvalue would lead to an exploding exponential term in the general solution. Examining a positive eigenvalue would introduce instability in the system and can therefore in no way correspond to a stable solution. When examining these extreme values one is thus interested in the densities that correspond to a local Jacobi-matrix with only zero or eigenvalues with a real part smaller than zero.

In section 4.2 a mathematical derivation is presented that serves as a dense description on how to apply this method to plasma physical systems. These mathematical concepts are then applied to a conceptual physical system in section 4.3. This system represents a Townsend avalanche mechanism and depending on the initial conditions this system has both a stable and unstable equilibrium. Finally, a conclusion is drawn on the validity of this method and its limitations will be addressed in section 4.4.

4.2 Mathematical description of method

In chapter 3 the particle balance equation is elaborated. This differential equation shows that without transport the time evolution of species densities equal the source vector. Therefore, the source vector can be set to zero to retrieve the species densities that do not evolve over time, i.e. the species densities that are in equilibrium. This relation is given by

\[
\frac{\partial n}{\partial t} = S(n) = 0.
\]  

(4.1)

Solving equation (4.1) will yield the different equilibria in a system. However, the chemical equilibrium of a plasma system refers to a stable equilibrium position. Species densities in such an equilibrium quickly move back to this state if small perturbations is applied. These stable equilibria are identified by a Jacobian analysis of the source.

This can be shown with the help of the following derivation. First, the linearized source for a specific point is written as

\[
\frac{\partial n}{\partial t} = S(n) = Jn.
\]  

(4.2)

By introducing a diagonalization of the Jacobian, equation (4.2) can be written as

\[
J = V \Lambda V^{-1}.
\]  

(4.3)
Here $V$ represent the right eigenvectors, and $\Lambda$ is a matrix that is represented by the corresponding eigenvalues on the diagonal. Using equation (4.3), equation (4.2) can be written as

$$V^{-1} \frac{\partial n}{\partial t} = \Lambda V^{-1} n.$$  
(4.4)

By the substitution $V^{-1} n = \psi$, equation (4.4) can be simplified to

$$\frac{\partial \psi}{\partial t} = \Lambda \psi.$$  
(4.5)

The solution for the obtained system in equation (4.5) yields the solution

$$\psi(t) = \sum_i C_i \exp \lambda_i t.$$  
(4.6)

Here $\lambda_i$ corresponds to one of the eigenvalues of the Jacobian and $C_i$ to an integration constant. For a physical system, it can be concluded that this solution converges if the eigenvalues of the Jacobian represented in the matrix $\Lambda$ are zero or negative. For positive eigenvalues unstable behavior is obtained due to an exploding exponential term. In order to check the an obtained equilibrium for stability, the Jacobian eigenvalues can thus be evaluated for the specific point.

### 4.3 Concept Case I: A Townsend avalanche mechanism

The method described in this chapter can be clarified by the use of a physical avalanche mechanism. In this mechanism an exploding growth of electrons can be observed due to a small perturbation that is applied to the system. In these types of mechanisms this will generate a so-called snowball effect, an event of very small significance that triggers new events that quickly build up to something of large significance.

In this concept case the developed method is applied to a physical system and aims to analyze it from a more pragmatic point of view. The considered model is given by a single reaction that represents a very simple argon system, this reaction is stated below:

$$Ar + e \rightarrow Ar^+ + e + e,$$

: $k_1$.

This reaction represents a non-linear system that will generate two new electrons for every consumed electron as long as there is argon in the system. It is thus expected that the system will have two equilibrium points. The first point corresponds to the situation without electrons in the system. This equilibrium is expected to be unstable since even a very small amount of electrons will trigger the avalanche mechanism. The second point corresponds to the situation without argon in the system. This equilibrium is expected to be stable because even when a small amount of argon is added to the system, it will quickly move back to its initial state without argon. The Jacobi-matrix is thus expected to have a positive eigenvalue for the first situation and a negative eigenvalue for the second situation. The considered system can be written in the usual dynamical form given by

$$\frac{d}{dt} \begin{pmatrix} n_{Ar} \\ n_e \end{pmatrix} = S(n) = \begin{pmatrix} -k_1 n_{Ar} n_e \\ k_1 n_{Ar} n_e \end{pmatrix}.$$  
(4.7)

For an equilibrium point it is known that the species densities no longer depends on time. This is the case when the source equation represents a steady state solution, i.e. when the system of equations represented by the source vector equal to the zero vector. This condition is stated as

$$S(n) = \begin{pmatrix} -k_1 n_{Ar} n_e \\ k_1 n_{Ar} n_e \end{pmatrix} = 0.$$  
(4.8)

It can be seen rather easily that this condition is satisfied if $n_{Ar} = 0, n_e = 0$ or if both densities are zero simultaneously. For the last case it can already be seen that the absence of both species would result in
constant densities since there will not be any reaction at all. Therefore, the eigenvalues of the Jacobian are expected to be zero in this case. However, for the other cases it helps to do an analysis based on the Jacobian eigenvalues that are computed as

\[
J = \frac{\partial S}{\partial n} = \begin{pmatrix} -k_1 \cdot n_e & -k_1 \cdot n_{Ar} \\ k_1 \cdot n_e & k_1 \cdot n_{Ar} \end{pmatrix}.
\] (4.9)

The eigenvalues of this matrix are obtained by solving the characteristic equation for \( \lambda \). The characteristic equation is given by \( \det(J - \lambda I) = 0 \), where \( I \) is the associated identity matrix. The gathered eigenvalues of the Jacobi-matrix are then given by

\[
\lambda = \begin{bmatrix} 0 \\ k_1(n_{Ar} - n_e) \end{bmatrix}.
\] (4.10)

From these eigenvalues the following is observed at the equilibrium densities:

1. If \( n_{Ar} = 0 \) and \( n_e > 0 \): all eigenvalues are smaller than or equal to zero which implies that the equilibrium point is stable.
2. If \( n_{Ar} > 0 \) and \( n_e = 0 \): one of the eigenvalues is greater than zero which implies that the equilibrium point is not stable.
3. If \( n_{Ar} = 0 \) and \( n_e = 0 \): all eigenvalues are equal to zero which implies all densities are constant.

These results are thus in line with the expectations from the discussed method. In figure 4.2, the numerical and analytical solution of system (4.7) are compared for initial values \( n_{Ar} = 1 \) and \( n_e = 0.01 \) at \( t = 0 \). This corresponds to the analytical solutions for \( n_{Ar} \) and \( n_e \) stated below.

\[
n_{Ar} = \frac{101}{100 + \exp(1.01t)} \quad \text{and} \quad n_e = \frac{1.01 \cdot \exp(1.01t)}{100 + \exp(1.01t)}
\] (4.11)

In this thesis, a numerical solver for stiff differential equations (ode15s) is used for solving the dynamical systems. By comparing both the numerical and the analytical solutions with each other it can be established that this numerical solver yields the expected outcome. In the figures below the analytical solution is compared to the numerical solutions.

![Graphs showing analytical and numerical evaluations of the system](image1.png)  
(a) Analytical evaluation of system.  
(b) Numerical evaluation of system.

Figure 4.2: Time evolution of the discussed system with a small perturbation in the unstable equilibrium point. Initial values: \( n_{Ar} = 0.99 \) and \( n_e = 0.01 \).

Figure 4.2 shows that a small perturbation in the electron density results in an unstable system with a rapidly increasing amount of electrons. This continues until all argon is consumed and the stable equilibrium is thus reached. Furthermore, both the analytical and numerical system show similar results. Conclusively, the described method is able to correctly locate the different equilibrium points and to correctly identify whether they are stable or not.
4.4 Conclusions

The analyzed case shows that it is possible to compute the chemical equilibrium directly from the source term. Furthermore, this case indicates that one is able to determine whether an equilibrium point is stable or not by investigating the eigenvalues of the local Jacobian. However, not all systems reach a constant, stable equilibrium point. For systems with complex eigenvalues extra care should be applied to the behaviour of the chemical equilibrium point since oscillations are imposed to the system. This also holds for pulsed systems for example. Furthermore, the method is limited by the non-linear solver. In this work, the MATLAB fSolve algorithm is used, a solver that requires an initial guess. For more complex systems with sometimes many equilibrium points, it can be very challenging to actually find an actual stable point that is of physical relevance. However, with some adjustments to the method it is possible to exclude the search for equilibrium densities to the positive real numbers. Conclusively, the method is able to locate chemical equilibrium points and can be very useful to investigate stability properties without solving the entire time evolution for a dynamical system. Nonetheless, different limitations should be taken into account and the quality of the non-linear solver is the major bottleneck in how efficient this method can be applied.
5 Multidimensional ILDM approach

5.1 Introduction

In the previous sections, the ILDM method is elaborated thoroughly together with an alternative approach to search for stable equilibria in plasma systems. So far, a one-dimensional procedure is maintained in the application of the ILDM method. This section goes into further detail in how to scale up the dimensions of the manifold by including additional parameters into the manifold equation. The hydrogen system from section 3.3 will be used as a first example to show the profits of adding an extra dimension. Furthermore, it serves as a good introduction to get familiar with the limitations of the approach as well. Then, the effort on a more complex argon system is discussed. This case study, in particular, is evaluated extensively to support future research with such a system.

5.2 Case Study II: A three level hydrogen system (2D)

In this case study, the three level hydrogen system of section 3.3 is discussed again. However, this time a two-dimensional approach is applied. The ILDM method for creating multidimensional manifolds is for a significant part very similar compared to creating a one-dimensional manifold. To avoid redundancy, this section will focus purely on the differences between both. In short the set of reactions given by 5.1 is discussed with known transition frequencies $k_1, k_2, ..., k_6$. The electrons in this problem are neglected such that a linear system is obtained. Consequently, the Jacobian matrix computed in the previous case study remains constant throughout the problem.

\[
\begin{align*}
H + e & \leftrightarrow H^* + e \\
H + e & \leftrightarrow H^+ + e + e \\
H^* + e & \leftrightarrow H^+ + e + e
\end{align*}
\]  

(5.1)

Parameterization of the system

When a multidimensional manifold is created for a linear system, nothing changes in the setup of the system and also the stiffness and corresponding time scales remain the same. The first changes show up in the manifold and the parameter equation during the parameterization of the system. The element conservation equation is independent of the amount of dimensions and therefore doesn’t change.

The first and foremost equation that changes is the parameter equation. In the first case study the second species ($H^*$) is selected as the parameter species. For every additional dimension that is added to the manifold another parameter species has to be included. Note that the manifold that corresponds to two particular species can be very different to a manifold corresponding to different species in the reaction mechanism. In this case study $H^+$ is chosen as the second parameter species that gives rise to the parameter equation given by

\[
P y - \omega p = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} y - \omega p = 0.
\]  

(5.2)

Again, $y$ is given by $(H, H^*, H^+)$ with corresponding initial mass fractions $(1.0 \cdot 10^{-2}, 3.0 \cdot 10^{-1}, 6.9 \cdot 10^{-1})$. Due to this additional parameter, an extra equation is created that is added to the parameterized system of equations. This allows the modeler to solve the system for a more complete system of equations (i.e. less reduced system) and therefore, more details in the chemical kinetics of the system are represented. However, at the same time this means that the stiffness of the system will remain larger since less fast time scale processed are separated from the system. This follows from $N_f = N_e + N_\gamma - N_d$, where $N_f$ is the maximum amount of fast time scale process that can separated from the system. In this example a two-dimensional manifold is created from a system consisting of three species containing only one...
element. This implies that in order to create a two-dimensional manifold no fast time scale processes can be separated from the system. This results in a fast Schur matrix given by

$$Q_f^T = 0.$$  \hspace{1cm} (5.3)

Consequently, the manifold equation is no longer present in the parameterized system of equations. In practice this means that the initial system is not reduced and therefore the manifold should include all information such that all system kinetics are represented by the solution obtained via this manifold. All derivations are taken into account the parameterized system of equations can be expressed as

$$g(y) = \begin{cases} y_2 - \omega p_1 \\ y_3 - \omega p_2 \\ y_1 + y_2 + y_3 - 1 \end{cases} = 0.$$  \hspace{1cm} (5.4)

**Determination and use of manifold**

From system \(5.4\) it follows trivially that the manifold is essentially defined through mass conservation expressed by \(y_1 + \omega p_1 + \omega p_2 = 1\). Where \(\omega p_1\) and \(\omega p_2\) are the modeling parameters with values between 0 and 1. However, one is able to achieve the same result using a naive approach that is also adopted for more complex systems.

In this approach, the first step is to search for a point that is located on the manifold to be constructed. Generally, this point is the the chemical equilibrium point if there is one present in the observed system. The initial value of the parameters are chosen to be equal to these equilibrium values. Since the chemical equilibrium is located on the manifold, the solution of the parameter equation should yield the same values as the equilibrium point for all remaining species densities. Manifold points are obtained by the use of the designed model that uses a step-size of \(4 \cdot 10^{-11}\) in this particular case. This step-size is chosen such that subsequent manifold points are located within a range of 5 percent from each other.

The obtained manifold points clearly describe a linear surface embedded in three-dimensional space. Therefore, the generalized manifold can be obtained rather easily by solving equation \(5.4\) for \(Y_1, y_2\) and \(y_3\). As expected, this corresponds to the surface given by \(y_1 + y_2 + y_3 = 1\). Consequently, the species fractions can be defined as \(y = y(y_p)\), i.e. as a function of the parameter species \(y_2\) and \(y_3\). In figure \(5.1\) the one- and two-dimensional manifold that correspond to this situation are shown. It can be seen that the one-dimensional manifold represents a line on the two-dimensional manifold surface.

![Figure 5.1: Comparison between all manifolds of a three dimensional hydrogen system.](image-url)
By using the parameterized species fractions, the full system of differential equations can be expressed in terms of the parameters as

\[ \frac{dy_p}{dt} = S_p(y(y_p)). \] (5.5)

The initial system of equations stated in section 3.3, equation (3.22) can now be expressed in as

\[ \frac{d}{dt} \left( y_2 y_3 \right) = \left( k_1 y_1(y_2, y_3) - k_3 y_2 + k_4 y_3 - k_5 y_2 \right) \]
\[ k_2 y_1(y_2, y_3) + k_3 y_2 - k_4 y_3 - k_6 y_3 \]. (5.6)

Here \( y_1(y_2, y_3) = 1 - y_2 - y_3 \). Dependent on the relation between the different points, the function \( y_i(y_p) \) can also represent another interpolation if an accurate fit like this is not possible. After the parametrization of the system, it is no longer needed to solve for all species.

From equation (5.6) the evolution of the parameter species over time is computed, \( y_2(t) \) and \( y_3(t) \). Generally, this corresponds to a reduced solution since fast time scale processes are separated from the system. However, as explained in the parameterization of the system, for this problem all fast time scale processes are still present in the system. The evolution of the final species is obtained by \( y_1(t) = 1 - y_2(t) - y_3(t) \). In figures below the comparison between the solution of all manifolds is shown for the three different species.

Figure 5.2: Comparison of time evolution’s from the ground state hydrogen species.

Figure 5.3: Comparison of time evolution’s from the excited state hydrogen species.
5.3 Case Study III: An argon molecular plasma

In this final case study, the attempts on a more complex, molecular argon plasma are considered. In the modeling phase of this plasma some problems are encountered that are worth sharing to improve future results on this system. The considered system contains 7 different species: $Ar, Ar^+, Ar_2^+, Ar(4p), Ar(4s)^m, Ar(4s)^r$ and $e$. Most species denote a specific energy level of argon for which some are in a meta stable (m) or resonant (r) state, $e$ represents the electrons that are taken into account. In this model, the electron temperature (3eV) and the heavy particle temperature (600K) are kept constant. $T_e$ is expressed with respect to $T_{ref}$ as $T_e/T_{ref}$, with $T_{ref}$ equal to 1 eV in order to obtain correct dimensions. The set of reactions is retrieved from the work of T. Rehman and stated in table 1.

Table 1: Reaction scheme of 7 level argon system.

<table>
<thead>
<tr>
<th>#</th>
<th>Reaction</th>
<th>Rate Coefficient</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$Ar + e \rightarrow Ar(4s)^m + e$</td>
<td>$5.00 \times 10^{-15} T_e^{0.74} e^{-11.56/T_e}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>2</td>
<td>$Ar + e \rightarrow Ar(4s)^r + e$</td>
<td>$5.00 \times 10^{-15} T_e^{0.74} e^{-11.56/T_e}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>3</td>
<td>$Ar + e \rightarrow Ar(4p) + e$</td>
<td>$1.40 \times 10^{-14} T_e^{0.71} e^{-13.20/T_e}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>4</td>
<td>$Ar(4s)^r + e \rightarrow Ar + e$</td>
<td>$4.30 \times 10^{-16} T_e^{0.74}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>5</td>
<td>$Ar(4s)^m + e \rightarrow Ar + e$</td>
<td>$4.30 \times 10^{-16} T_e^{0.74}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>6</td>
<td>$Ar(4p) + e \rightarrow Ar + e$</td>
<td>$3.90 \times 10^{-16} T_e^{0.71}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>7</td>
<td>$Ar(4s)^r + e \rightarrow Ar(4p) + e$</td>
<td>$8.90 \times 10^{-13} T_e^{0.51} e^{-1.59/T_e}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>8</td>
<td>$Ar(4s)^m + e \rightarrow Ar(4p) + e$</td>
<td>$8.90 \times 10^{-13} T_e^{0.51} e^{-1.59/T_e}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>9</td>
<td>$Ar(4s)^r + e \rightarrow Ar(4s)^r + e$</td>
<td>$2.00 \times 10^{-13}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>10</td>
<td>$Ar(4s)^r + e \rightarrow Ar(4s)^m + e$</td>
<td>$3.00 \times 10^{-13}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>11</td>
<td>$Ar(4p) + e \rightarrow Ar(4s)^r + e$</td>
<td>$3.00 \times 10^{-13} T_e^{0.51}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>12</td>
<td>$Ar(4p) + e \rightarrow Ar(4s)^r + e$</td>
<td>$3.00 \times 10^{-13} T_e^{0.51}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>13</td>
<td>$Ar + e \rightarrow Ar^+ + e + e$</td>
<td>$2.30 \times 10^{-14} T_e^{0.68} e^{-15.76/T_e}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>14</td>
<td>$Ar(4s)^m + e \rightarrow Ar^+ + e + e$</td>
<td>$6.80 \times 10^{-15} T_e^{0.67} e^{-4.20/T_e}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>15</td>
<td>$Ar(4s)^r + e \rightarrow Ar^+ + e + e$</td>
<td>$6.80 \times 10^{-15} T_e^{0.67} e^{-4.20/T_e}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>16</td>
<td>$Ar(4p) + e \rightarrow Ar^+ + e + e$</td>
<td>$1.80 \times 10^{-13} T_e^{0.61} e^{-2.61/T_e}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>17</td>
<td>$Ar(4s)^r \rightarrow Ar$</td>
<td>$3.00 \times 10^7$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>18</td>
<td>$Ar(4p) \rightarrow Ar(4s)^r$</td>
<td>$3.00 \times 10^7$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>19</td>
<td>$Ar(4p) \rightarrow Ar(4s)^m$</td>
<td>$3.00 \times 10^7$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>20</td>
<td>$Ar^+ + e + e \rightarrow Ar + e$</td>
<td>$8.75 \times 10^{-39} T_e^{-4.5}$</td>
<td>$m^6 \cdot s^{-1}$</td>
</tr>
<tr>
<td>21</td>
<td>$Ar_2^+ + e \rightarrow Ar(4s)^r + Ar$</td>
<td>$0.60 \times 10^{-12} \left(\frac{T_e(K)}{300}\right)^{0.66}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>22</td>
<td>$Ar_2^+ + e \rightarrow Ar(4s)^m + Ar$</td>
<td>$0.60 \times 10^{-12} \left(\frac{T_e(K)}{300}\right)^{0.66}$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
</tbody>
</table>
Table 1 – continued from previous page

<table>
<thead>
<tr>
<th>#</th>
<th>Reaction</th>
<th>Rate Coefficient</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>$\text{Ar}^+ + \text{Ar} + \text{Ar} \rightarrow \text{Ar}_2^+ + \text{Ar}$</td>
<td>$2.25 \times 10^{-43} \left( \frac{T_h(\text{K})}{300} \right)^{-0.3}$</td>
<td>$m^6 \cdot s^{-1}$</td>
</tr>
<tr>
<td>24</td>
<td>$\text{Ar}_2^+ + e \rightarrow \text{Ar}^+ + \text{Ar} + e$</td>
<td>$1.11 \times 10^{-12} \exp \left( - \frac{-2.94-3[T_h(\text{eV})]-0.026}{T_e(\text{eV})} \right)$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
<tr>
<td>25</td>
<td>$\text{Ar}_2^+ + \text{Ar} \rightarrow \text{Ar}^+ + \text{Ar} + \text{Ar}$</td>
<td>$5.22 \times 10^{-16} \exp \left( - \frac{1.304}{T_h(\text{eV})} \right)$</td>
<td>$m^3 \cdot s^{-1}$</td>
</tr>
</tbody>
</table>

Mathematical setup of dynamical system

The vector containing all species densities $\mathbf{n}$, is defined as $\mathbf{n} = (\text{Ar}_n, \text{Ar}^+_n, \text{Ar}_2^+_n, \text{Ar}(4p)_n, \text{Ar}(4s)_{r,n} \text{Ar}(4s)_{m,n} \epsilon_n)$. Note that species densities are used instead of mass fractions, this approach is adapted to ensure that the numerical methods given in terms of species densities can be applied without conversions. The final results are converted to mass fractions to compare it with the work of Rehman. First, the initial system is solved with the use of the MATLAB function ODE15s to ensure that all input configurations are correct. The solution, shown in figure 5.5, indicates that the behavior of the system is similar to the system considered in the work from T. Rehman. Furthermore, the sum of the species remains 1 over the entire time, this indicates that mass is indeed conserved. The initial densities and input file that are used to solve the system, are stated in the appendix.

![Figure 5.5: Solution in mass fractions of seven level argon system obtained with the ODE15s function.](image)

System stiffness and time scale analysis

Since the system is non-linear the Jacobian depends on the species densities and changes over time. This is also important in the determination of the reduced matrix that due to this property changes for every change in densities. Since the Jacobian represents a linearization of the system around a very specific point, it must be updated in every step during the determination of the manifold. This will be explained in further detail later. The Jacobian as a function of the species densities is given by

$$J(n) = \frac{\partial S(n)}{\partial n}.$$ (5.7)
At the equilibrium point with densities $\mathbf{n}_{eq} = (8.3 \cdot 10^{19}, 3.8 \cdot 10^{22}, 4.1 \cdot 10^7, 4.6 \cdot 10^{17}, 2.6 \cdot 10^{17}, 3.0 \cdot 10^{17}, 3.8 \cdot 10^{22})$ the time scales are $9.8 \cdot 10^{-13}, 1.4 \cdot 10^{-11}, 1.9 \cdot 10^{-11}, 4.3 \cdot 10^{-10}$ and $3.5 \cdot 10^{-8}$. The other two remaining time scales approaching infinity. Although the exact numbers differ with the work of Rehman, the results are in the same order. The stiffness of the system is $3.5 \cdot 10^8$ what indicates that the system is not very stiff.

**Difficulties in the implementation**

During the later stages, some problems occurred with the implementation of the method. Therefore, the standard approach is no longer maintained and a brief discussion on these problems is given instead. A better understanding of these problems can hopefully support further research on this topic.

Initially, the manifold equation in parametrization of the system is done by using a standard Schur decomposition of the Jacobian. However, in this case, the Schur decomposition introduced complex numbers of negligible magnitude. Due to the presence of these complex numbers the composition space increased significantly. As a result, the non-linear solver struggled a lot to come up with positive real species densities. To overcome this problem the eigen decomposition of the Jacobian matrix is performed as

$$ J = V \Lambda V^{-1} $$

(5.8)

Here $V$ represent the eigenvectors of the Jacobian and $\Lambda$ a diagonal matrix with the corresponding sorted eigenvalues on the diagonal. For the eigenvalues holds that $|Re(\lambda_1)| \geq |Re(\lambda_2)| \geq \ldots \geq |Re(\lambda_i)|$. In this way the lower rows of $V^{-1}$ correspond to the fast vectors and the upper rows to the slow vectors. Similar to the approach with the Schur decomposition. However, this approach is less favorable since if $V$ is not conditioned in the right way MATLAB can’t solve this and returns that it is to close to zero. For the Schur composition $Q$ is orthogonal and therefore always conditioned correctly.

The complex values are most likely introduced due to the residuals from the eigenvalues of the Jacobian that where not exactly zero. However, using equation (5.8) this problem was solved and the reduced matrix could be expressed only in terms of real numbers. Nevertheless, this didn’t help in getting the solver to yield valuable results. Even though the composition space decreased, the solver was not able to solve the problem anymore. This indicates that there are no equilibrium densities to start with. Which is odd, since figure 5.3 indicates that there actually is a stable equilibrium. Evaluating this equilibrium point in the source term did not yield zero as a result. An indication that during the evaluation of the dynamical system several rounding errors occur that cause fSolve to stall. This was confirmed by the convergence log of the fSolve that indicated that it could not converge.

Further investigation suggested that there was a difference in magnitude around $10^{15}$ between the species densities around the equilibrium point. Due to this large variety scaling the numbers did not solve the problem either. Relative to other species, some species where completely neglected due to machine precision. fSolve also seemed to have trouble with this high variety in magnitude. To overcome this problem a new approach is adapted. The system is normalized such that the minimum value of the system should yield a value around zero. In this approach a derivative-free solver is used to avoid convergence problems. The solver called fMinSearch is used to search for the smallest value that could be obtained. This method created some additional space for the numerical errors that where encountered previously. Usually, the results only accepted if the final value of the system obtained by fMinSearch was between 0 and 20, which is very small compared to the initial values in the order of 1e22. This approach turned out to be successful, but at excessive computational costs for a relative small system like this.

However, some interesting behavior occurred given the mathematical background of the ILDM method. According to the results, the step size did not really show a difference in behavior while it is expected that a large step size will violate the linearization condition of the Jacobian. With increasing step size no significant errors are noticed. Also, it became extremely hard to find manifold points further in the process. Figure 5.7 shows the different manifold points that are obtained and how the fit aligns perfectly, even for larger step sizes.
Nevertheless, the behavior of the manifold was rather similar with the work of Rehman. Although, the values are somewhat higher, the slope and general behavior are almost the same. The other relations between the species and the parameter species (Ar\(^+\)) show a suspicious linear behaviour for a non linear system. If Ar is plotted against Ar\(^+\), the relation between the species fractions is clearly linear. Similar results are obtained for all other species densities compared to the parameter species. Consequently, the obtained behavior is very different from the expected results retrieved in previous implementations.

5.4 Conclusions

The implemented ILDM method is used to create the manifold of a three level hydrogen system in three different dimensions. In this first system, electrons were neglected such that a linear system was obtained. The method is shown to be successful for determining the manifold in different dimensions. For the studied system three different time scales are retrieved, a fast time scale process, a slow time scale processes and conserved process.

The results obtained via a zero-dimensional manifold clearly show that the chemical equilibrium point is the only information that is captured by the manifold. The solution represents a straight horizontal line through the equilibrium point. This is in line with the expectations, since both the long en short time scale processes are separated from the system. Therefore, the only behavior that is represented on the solution is from the conserved quantity. The one-dimensional manifold is compared with previous implementations of the ILDM method and shows consistent results. Namely, a reduced system in which the fast time scale processes are separated from the system and other processes remain present. Finally, the results using the two-dimensional manifold are compared with the solution of the full initial system. These results where in line with each other. The created manifold included all time scale processes since all Schur eigenvectors were considered in order to create a two-dimensional manifold.
From the time scales of the system no direct conclusions can be drawn on when the corresponding reaction is going to equilibrate. However, it can be concluded that a higher dimensional manifold will provide a more accurate insight in the reaction mechanism in the first period of time. This is expected since a higher dimensional manifold can only be constructed if less fast time scale processes are approximated to be in steady state and therefore less information is neglected by the method. It can also be concluded that an increase in the number of dimensions will also lead to an increase in system stiffness. This is understood by the $\tau_{min}$ decreases for every additional fast time scale process that is taken into account.

Finally, a non-linear argon system is investigated. The implemented method showed difficulties with a correct determination of the manifold. It can be concluded that these difficulties are partly introduced by limitations with the used non-linear solver. Due to the large variety in species densities in the equilibrium point the non-solver is not able to come up with a solution. Changing from species densities to species fractions did not solve this problem either. The alternative approach to normalize the entire system and search for the minimum value of the system using \texttt{fMinSearch} showed to be an effective option in searching for points close to the wanted solution. However, the obtained manifold obtained via this method did not yield the correct results. Further investigation is required to draw additional conclusions on the precise cause.
6 General conclusions, discussion and outlook

The research of this thesis has generated a deeper understanding on how a multidimensional ILDM approach can serve plasma physical modelling. The existing methods on ILDM in plasma physics have been extended with an alternative approach to search for chemical equilibria and also the work on linear two-dimensional manifolds confirms knowledge on the topic. Finally, our understanding is improved on how to start with establishing similar results for non-linear systems. The main conclusion from this thesis is that it is possible to come up with a general way to extend the work on ILDM in plasma physics up to two dimensions for linear systems and there is room to extend this to non-linear systems as well. In the following subsections the research questions of the introduction are discussed and afterwards an outlook is provided for potential future research to improve upon this work.

6.1 Alternative method to obtain chemical equilibrium points

It is possible to compute the chemical equilibrium point of both linear and non-linear systems without solving the time evolution of the system. The results show for a non-linear Townsend avalanche mechanism that the stable and unstable equilibrium are correctly identified. This is done solving the the source vector that is set to zero for the species densities. By inspection from the local Jacobian matrix at this point the stability can be identified. They are shown to be stable if the eigenvalues of this Jacobian matrix are zero or negative and unstable when positive.

However, dependent on what solver is used, it can be challenging to identify all equilibria in the system and to efficiently locate the stable equilibria without the bycatch of sometimes many unstable equilibria. Also, the non-linear solver sometimes shows to have difficulties with solving more complex non-linear systems and a required initial value for the these solver hinders the use of this method to use it for a general orientation of the behavior of the system. Overall, it can be concluded that at this moment the designed method should be seen as an extension on the existing method more than a replacement.

6.2 Multidimensional ILDM for linear systems

It is possible to come up with a general approach to create up to two dimensional ILDM manifolds for linear dynamical systems. From the comparison of the solutions obtained with this approach is can be concluded that additionally included dimensions yield a more precise simulation of the system kinetics in first period of time. It also shows that including additional dimensions will increase the stiffness of a system. In this work no direct relation between the obtained time scales and the descent of the different time scale reactions is found. Current results show that the use of naive interpolation techniques is sufficient to obtain accurate results with two-dimensional manifolds for linear systems. However, more test cases are required to confirm this conclusions. For the linear systems discussed in this thesis it was possible to come up with an accurate fit that generalized the relation with the parameter species. This approach outperformed the use of an ordinary look up table.

6.3 Multidimensional ILDM for non-linear systems

The implemented method is not able to come up with a general approach to create one- or more dimensional manifolds for non-linear systems. It can be concluded that this is partly because of limitations of the used non-linear solver. Further research suggested that this problem can be solved somewhat by a normalization of the system and using a minimum search approach instead. However, no accurate results are obtained using this approach so far. Nevertheless, it can be concluded that there is room to improve these results since further investigation could draw additional conclusions on the precise cause of why the current approach failed to deliver.
6.4 Outlook

The results obtained in this thesis are interesting in general. However, in this thesis no method is presented for efficiently use higher dimensional manifolds or the use of more advanced interpolation methods. In order to do further research on this topic it is important to improve upon this implemented method such that non-linear problems are treated more accurate. The first suggestions for improvement consider the non-linear solver, it is useful to create a better understanding in how to deal with a large variety in species densities and how to effectively search for physical relevant solutions. Furthermore, the required initial guess and the bycatch of many unstable equilibria obstructed the search for stable equilibria in the designed method from this thesis. Improvements could help to overcome this problem. Also, the use of look-up tables is not thoroughly discussed in this thesis, efficient arrangement of these tables or maybe more advanced fitting techniques could help to reduce computational costs with the use of higher dimensional look-up tables. Finally, the method from this thesis is implemented in MATLAB. Possibly, much faster implementations can already be achieved by switching to another programming language like C++ for example. Using more performance oriented programming languages and improved code itself could yield more accurate information of the actual performance of the method.
References


Appendix

A Implementation of general model

Main file used to determine the manifolds.

```matlab
% Author: Joris van Laanen
% Date: 18/03/2019
% - Description -
% This file should be used together with the class 'funDirectory.m' and an
% input file that is formatted like 'hydrogen_inputFile.m'. Your input
% file can be linked with this script by using the Input area, all other
% operations in this script are adjusted automatically. This script can be
% used in the implementation of the ILDM method. It is able to compute
% properties that belong to a plasma system with a given source using an
% ILDM approach. This work is based upon 'Studies on plasma chemical
% reduction' (Rehman, 2018).
% ------------------------------------------------------------------------
clear all;
close all;
% ==========================================================================
% Set initial values for variables in script
figCount = 0;
% Create array with an estimate of the initial conditions of the species
if systConstr == 0
    initCon = inputFile.initDens;
else
    initCon = [inputFile.initDens; zeros(systConstr)];
end
% Solves the system of coupled differential equations
etime, specDens) = ode15s(evoSystem, t, initCon);
```
% Convert species densities to mass fractions
speDens = speDens ./ n;

% Find the densities in an equilibrium point and express in mass fractions
eqDens = fsolve(evoSystemSolve, initGuess);
eqDens = eqDens ./ n;

% Create a figure showing the evolution of the system
figCount = figCount + 1;
figure(figCount);
plotEq(time, speDens);

% Compute jacobian matrix at equilibrium point
gradientMat = funDirectory.gradientMatrix(inputFile, eqDens);
stoichioMat = stoichioMat * gradientMat;

% Compute the standard Schur decomposition of the jacobian
transSchurMat = transpose(schurVecMat);

% Compute the fast Schur vector matrix and
fastSchurMat = transSchurMat(1,:);

% TODO: Make this part into an adjustable function
% Compute the manifold points
parameter = eqDens(1);
stepsize = 5e-11;
manifoldArray = [];
direction = 1;
i = 1;
while i == 1 || (manifoldEq.y1 <= 1 && manifoldEq.y1 >= 0) && ... (manifoldEq.y2 <= 1 && manifoldEq.y1 >= 0) && ... (manifoldEq.y3 <= 1 && manifoldEq.y3 >= 0)
syms y1 y2 y3
eq1 = y1 + y2 + y3 == 1;
eq2 = y1 == parameter;
eq3 = fastSchurMat * (jacobianMat * [y1; y2; y3]) == 0;
manifoldEq = solve([eq1, eq2, eq3], [y1, y2, y3]);
if direction == 1
manifoldArray(i,1) = manifoldEq.y1;
manifoldArray(i,2) = manifoldEq.y2;
manifoldArray(i,3) = manifoldEq.y3;
p
parameter = parameter + stepsize;
i = i + 1;
else
manifoldArray(size(manifoldArray,1)+1,1) = manifoldEq.y1;
manifoldArray(size(manifoldArray,1),2) = manifoldEq.y2;
manifoldArray(size(manifoldArray,1),3) = manifoldEq.y3;
p
parameter = parameter - stepsize;
i = i + 1;
end
if direction == 1
direction = 2;
parameter = eqDens(1) - stepsize;
manifoldArray(size(manifoldArray,1),:)=[];
i = 1;
else
end
Function directory file that contains all manually created functions.

```matlab
classdef funDirectory < handle
    properties (Constant)
    end
    methods (Static)
    % Constructor
    function obj = functionDirectory()
    end
    % Additional functions can be added below this line.
    % ----------------------------------------------------------------------
    % Name: gradientMatrix(inputFile, densities)
    % -- Description --
    % This functions is able to numerically compute the gradient matrix
    % from which the numerical Jacobian can be derived. This function is
    % based on equation 4.24, page 36 from studies on plasma chemical
    % reduction. (Rehman, 2018)
    % -- Input --
    % inputFile: input file with properties of the plasma system
    % densities: density array with format [d1, d2, ..., dn]
    function gradientMatrix = gradientMatrix(inputFile, densities)
        gradientMatrix = zeros(inputFile.numReactions, inputFile.numSpecies);
        consumedStoichi = inputFile.consumedStoichi;
        for j = 1 : inputFile.numSpecies
            for r = 1 : inputFile.numReactions
                if consumedStoichi(j,r) == 0
                    gradientMatrix(r,j) = 0;
                else
                    prod = consumedStoichi(j,r) * densities(j)^ ...
                        (consumedStoichi(j,r) - 1) * inputFile.k(r);
                    for i = 1 : inputFile.numSpecies
                        if i == j
                            prod = prod * densities(i)^(consumedStoichi(i,r));
                        end
                        else
                            prod = prod * densities(i)^(consumedStoichi(i,r));
                        end
                        for (i = 1 : inputFile.numSpecies)
                            gradientMatrix(r,j) = prod;
                        end
                    end
                end
            end
        end
    end
    % Name: sortedSchur(A)
    %
```

This function performs a standard Schur decomposition such that it returns the decomposition \( Q \times N \times Q' \) of a square matrix \( A \). The function returns both the Schur vector matrix \( Q \) and the upper triangular matrix with the eigenvalues of \( A \) on the diagonal \( N \). These eigenvalues are sorted based on an ordering method and sorting direction. Authored by Chris Shoutrop and based on https://nl.mathworks.com/help/matlab/ref/ordschur.html.

% -- Input --
% A: matrix of type nxn

function \([Q,N]=\text{sortedSchur}(A,\text{orderingmethod},\text{sortingdirection})\)

\([Q,N]=\text{schur}(A,\text{'complex'})\);

if nargin ~= 3
    sortingdirection='\text{descend}'
end % if

if nargin == 1
    \([~,ix]=\text{sort}(|\text{diag}(\text{abs}(\text{real}(N)))|,\text{sortingdirection})\);
    a = [1:length(ix)];
    a(ix) = a;
    \([Q,N]=\text{ordschur}(Q,N,a)\);
    return
else
    switch orderingmethod
    case '\text{absreal}'
        \([~,ix]=\text{sort}(|\text{diag}(\text{abs}(\text{real}(N)))|,\text{sortingdirection})\);
    case '\text{absimag}'
        \([~,ix]=\text{sort}(|\text{diag}(\text{abs}(\text{imag}(N)))|,\text{sortingdirection})\);
    case '\text{abs}'
        \([~,ix]=\text{sort}(|\text{diag}(\text{abs}(N))|,\text{sortingdirection})\);
    case '\text{real}'
        \([~,ix]=\text{sort}(|\text{diag}(\text{real}(N))|,\text{sortingdirection})\);
    case '\text{imag}'
        \([~,ix]=\text{sort}(|\text{diag}(\text{imag}(N))|,\text{sortingdirection})\);
    end % switch

    if exist('ix','var')
        a = [1:length(ix)];
        a(ix) = a;
        \([Q,N]=\text{ordschur}(Q,N,a)\);
        return
    else
        \([Q,N]=\text{ordschur}(Q,N,\text{orderingmethod})\);
        return
    end % if
end % if
end % if

B Implementation of three level hydrogen system

Input file for the three level hydrogen system.

% Author: Joris van Laanen
% Date: 19/03/2019
% -- Description --
% This file serves as an input file for the ILDM method and should be used together with a 'main.m' file and corresponding 'funDirectory.m'. In this file all constant properties of the system must be defined together with the entire source equation. This file can be used as an
example on how to do format you input data in the right way. In this
case it describes an 3 level hydrogen system based on page 24 of
studies on plasma chemical reduction. (Rehman, 2018)
=================================================================

```matlab
classdef hydrogen_inputFile < handle
    properties (Constant)
        % The initial densities of the species
        initDens = [1e-2; 3e-1; 0.69]*1e10;
        numDistElmt = 1;
        maniDim = 1;
        % The number of constraints expressed with an equation
        numConstraints = 0;
        % Reaction rate coefficients [k1; k2; ...; kn]
        k = [2.7e10; ...
            9.0e8; ...
            1.0e6; ...
            7.5e4; ...
            3.8e1; ...
            1.7e2; ];
        % Stoichiometric matrix and corresponding consumption and production matrices
        stoichioMatrix = [-1 -1 0 0 1 1; ...]
            1 0 -1 -1 0 0; ...
            0 1 1 -1 0 -1 ...
        ];
        consumedStoichi = [1 1 0 0 0 0; ...]
            0 0 1 0 1 0; ...
            0 0 0 1 0 1 ...
        ];
        producedStoichi = [0 0 0 0 1 1; ...]
            1 0 0 1 0 0; ...
            0 1 1 0 0 0 ...
        ];
        % Derived parameters
        %
        numSpecies = size(hydrogen_inputFile.stoichioMatrix,1);
        numReactions = size(hydrogen_inputFile.stoichioMatrix,2);
        n = sum(hydrogen_inputFile.initDens)
    end
    methods (Static)
        % Constructor
        function obj = hydrogen_inputFile()
    end
    % Name:  evoSystem(n)
    %
    % -- Description --
    % This function should represent the evolution of the system. This
    % can be represented as a product of matrices or one can use
    % separate equations as an input.
    %
    % -- Input --
```
```
function sourceEq = evoSystem(n)
    sourceEq = hydrogen_inputFile.stoichioMatrix * [hydrogen_inputFile.k(1) * n(1); ...
                        hydrogen_inputFile.k(2) * n(1); ...
                        hydrogen_inputFile.k(3) * n(2); ...
                        hydrogen_inputFile.k(4) * n(3); ...
                        hydrogen_inputFile.k(5) * n(2); ...
                        hydrogen_inputFile.k(6) * n(3)];
end % function

% Name: equilibriumPlot(time, speDens)
% -- Description --
% This function contains all information about the plot that is
% generated from solving the source equation and should show the
% evolution of the densities over time. Therefore the amount of
% densities and the corresponding names should be adjusted to the
% model that is used.
% -- Input --
% time: a one collumn array of arbitrary length
% speDens: an array of the length as time with the same numer of
% collums as species present in the model

function eqPlot = equilibriumPlot(time,speDens)
    loglog( time, ... [speDens(:,1), speDens(:,2), speDens(:,3)], 'LineWidth', 1.5 );
    set(gca,'LineWidth',1.5); axis tight;
    legend('H','N*','N+','location','southeast');
    title( 'Solution of three level hydrogen system');
    xlabel('time (s)');
    ylabel('Mass Fraction (H, H* and H+)')
end % function
end % methods
end

C Implementation of molecular argon plasma

Input file for the three molecular argon plasma.

% Author: Joris van Laanen
% Date: 19/03/2019
% -- Description --
% This file serves as an input file for the ILDM method and should be
% used together with a 'main.m' file and corresponding 'funDirectory.m'.
% In this file all constant properties of the system must be defined
% together with the entire source equation. This file can be used as an
% example on how to do format you input data in the right way. In this
% case it describes an 7 level hydrogen system based on page 118 of
% studies on plasma chemical reduction. (Rehman, 2018)
%============================================================================
classdef argon_inputFile < handle
    properties (Constant)
        % Electron and heavy particle temperature
        T_e = 3
        T_ek = 34813.515
        T_h = 0.051704058 %eV
\[ T_{hk} = 600 \, ^\circ\text{K} \]

% Reaction rate coefficients \([k_1; k_2; \ldots; k_n]\)
\[
k = [ \\
 5.0e-15 \cdot \text{argon file.}T_e \cdot 0.74 \cdot \exp(-11.56 / \text{argon file.}T_e); \ldots \\
 5.0e-15 \cdot \text{argon file.}T_e \cdot 0.74 \cdot \exp(-11.56 / \text{argon file.}T_e); \ldots \\
 1.4e-14 \cdot \text{argon file.}T_e \cdot 0.71 \cdot \exp(-13.20 / \text{argon file.}T_e); \ldots \\
 4.3e-16 \cdot \text{argon file.}T_e \cdot 0.74; \ldots \\
 4.3e-16 \cdot \text{argon file.}T_e \cdot 0.74; \ldots \\
 3.9e-16 \cdot \text{argon file.}T_e \cdot 0.71; \ldots \\
 8.9e-13 \cdot \text{argon file.}T_e \cdot 0.51 \cdot \exp(-1.59 / \text{argon file.}T_e); \ldots \\
 8.9e-13 \cdot \text{argon file.}T_e \cdot 0.51 \cdot \exp(-1.59 / \text{argon file.}T_e); \ldots \\
 2.0e-13; \ldots \\
 3.0e-13; \ldots \\
 3.0e-13 \cdot \text{argon file.}T_e \cdot 0.51; \ldots \\
 3.0e-13 \cdot \text{argon file.}T_e \cdot 0.51; \ldots \\
 2.3e-14 \cdot \text{argon file.}T_e \cdot 0.71 \cdot \exp(-13.20 / \text{argon file.}T_e); \ldots \\
 4.3e-16 \cdot \text{argon file.}T_e \cdot 0.74; \ldots \\
 4.3e-16 \cdot \text{argon file.}T_e \cdot 0.74; \ldots \\
 3.9e-16 \cdot \text{argon file.}T_e \cdot 0.71; \ldots \\
 8.9e-13 \cdot \text{argon file.}T_e \cdot 0.51 \cdot \exp(-1.59 / \text{argon file.}T_e); \ldots \\
 8.9e-13 \cdot \text{argon file.}T_e \cdot 0.51 \cdot \exp(-1.59 / \text{argon file.}T_e); \ldots \\
 2.0e-13; \ldots \\
 3.0e-13; \ldots \\
 8.75e-39 \cdot \text{argon file.}T_e \cdot (-4.5); \ldots \\
 0.6e-12 \cdot (\text{argon file.}T_e / 300) \cdot 0.66; \ldots \\
 0.6e-12 \cdot (\text{argon file.}T_e / 300) \cdot 0.66; \ldots \\
 2.25e-43 \cdot (\text{argon file.}T_h / 300) \cdot -0.4; \ldots \\
 1.1e-12 \cdot \exp((-2.94 - 3 \cdot (\text{argon file.}T_h - 0.026) / \text{argon file.}T_h); \ldots \\
 (5.22e-16 / \text{argon file.}T_h) \cdot \exp(-1.304 / \text{argon file.}T_h); \ldots \\
 ]; \]

% Stoichiometric matrix and corresponding consumption and production matrices

\text{stoichioMatrix} = [ \\
 -1 -1 -1 1 1 1 0 0 0 0 0 -1 0 0 1 0 0 1 1 1 1 -1 \\
 1 1 ; \ldots \\
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 -1 0 0 -1 \\
 1 1 ; \ldots \\
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1 -1 1 -1 -1 ; \ldots \\
 0 0 1 0 0 -1 1 1 0 0 -1 -1 1 0 0 -1 0 -1 -1 0 0 0 0 0 0 0 \\
 0 0 ; \ldots \\
 0 1 0 -1 0 0 -1 0 1 -1 1 0 0 0 -1 0 -1 1 0 0 1 0 0 0 0 0 0 \\
 0 0 ; \ldots \\
 1 0 0 0 -1 0 0 -1 -1 1 0 1 0 -1 0 0 0 1 0 0 1 0 0 0 0 0 0 \\
 0 0 ; \ldots \\
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 -1 -1 -1 0 0 ... \\
 0 0 ... ];

end % properties

methods (Static)

% Name: evoSystem\(n)\)

% -- Description --

% This function should represent the evolution of the system. This
% can be represented as a product of matrices or one can use
% separate equations as an input.

% -- Input --

% \(n\): array of type \([d_1,d_2, \ldots, d_n]\); containing the species
% densities

function sourceEq = evoSystem\(n)\)

\text{sourceEq} = \text{argon file.stoichioMatrix} \ast [\text{argon file.k(1)} \ast n(1) \ast n(7) ; \ldots \\
 \text{argon file.k(2)} \ast n(1) \ast n(7) ; \ldots \\
 \text{argon file.k(3)} \ast n(1) \ast n(7) ; \ldots \\
 \text{argon file.k(4)} \ast n(5) \ast n(7) ; \ldots \\
 \text{argon file.k(5)} \ast n(6) \ast n(7) ; \ldots \\
 \text{argon file.k(6)} \ast n(4) \ast n(7) ; \ldots ];
function paraSystem = parasystem2(n, fastSchurMat, parameter, atomicMass)
    paraSystem(5) = abs(n(7)) - parameter;
end % function

function jacobianEq = jacobian(n)
    jacobianEq = [-2 .* argon_inputFile.k(23) .* n(1) .* n(2) + argon_inputFile.k(25) .* n(3) - argon_inputFile.k(2) 
          -argon_inputFile.k(23) .* n(1) .* n(2) + argon_inputFile.k(20) .* n(7) .* 2, ... 
          argon_inputFile.k(25) .* n(1) + argon_inputFile.k(21) .* n(7) + argon_inputFile.k(22) 
          argon_inputFile.k(6) .* n(7), ... 
          argon_inputFile.k(17) + argon_inputFile.k(4) .* n(7), ... 
          argon_inputFile.k(5) .* n(7), ... 
          -argon_inputFile.k(1) .* n(1) - argon_inputFile.k(13) .* n(1) - argon_inputFile.k(2) .* n(7), ... 
          -2 .* argon_inputFile.k(23) .* n(1) .* n(2) + argon_inputFile.k(25) .* n(3) + argon_inputFile.k(2) 
          -argon_inputFile.k(23) .* n(1) .* n(2) - argon_inputFile.k(20) .* n(7) .* 2, ... 
          argon_inputFile.k(25) .* n(1) + argon_inputFile.k(24) .* n(7), ... 
          argon_inputFile.k(16) .* n(7), ... 
          argon_inputFile.k(15) .* n(7), ... 
          argon_inputFile.k(14) .* n(7), ... 
          argon_inputFile.k(13) .* n(1) + argon_inputFile.k(24) .* n(3) + argon_inputFile.k(16) 
          2 .* argon_inputFile.k(23) .* n(1) .* n(2) - argon_inputFile.k(25) .* n(3), ... 
          argon_inputFile.k(23) .* n(1) .* n(2) - argon_inputFile.k(21) .* n(7) - argon_inputFile.k(22) 
          0, ... 
          0, ... 
          0, ... 
          -argon_inputFile.k(21) .* n(3) - argon_inputFile.k(22) .* n(3) - argon_inputFile.k(24) 
          0, ... 
          0, ... 
          0, ... 
          -argon_inputFile.k(18) - argon_inputFile.k(19) - argon_inputFile.k(11) .* n(7) - argon_inputFile.k(7) .* n(7), ... 
          -argon_inputFile.k(8) .* n(7), ... 
          argon_inputFile.k(3) .* n(1) - argon_inputFile.k(11) .* n(4) - argon_inputFile.k(12) .* n(7), ... 
          argon_inputFile.k(2) .* n(7), ... 
          0, ... 
          -argon_inputFile.k(18) + argon_inputFile.k(11) .* n(7), ... 
          -argon_inputFile.k(17) - argon_inputFile.k(10) .* n(7) - argon_inputFile.k(15) .* n(7) 
          argon_inputFile.k(9) .* n(7), ... 
          argon_inputFile.k(2) .* n(1) + argon_inputFile.k(21) .* n(3) + argon_inputFile.k(11) ];
function eqPlot = equilibriumPlot(time, speDens)
loglog( time, ...
[ speDens(:,1), speDens(:,2), speDens(:,3), speDens(:,4), speDens(:,5), speDens(:,6), speDens(:,7) ], 'LineWidth', 1.5 );
set(gca,'LineWidth',1.5); axis tight;
legend('Ar', 'Ar+', 'Ar2+', 'Ar(4p)', 'Ar(4s)r', 'Ar(4s)m', 'e', 'location','southeast');