NTNU
Norwegian University of
Science and Technology

Modelling of gas-solid reactions

Usage of industrial off-gas for pre-reduction
of manganese ores

Mathias Grønberg Gustum

Materials Science and Engineering (MTMT)
Submission date: June 2018
Supervisor: Kristian Etienne Einarsrud, IMA
Co-supervisor: Halvor Dalaker, SINTEF

Norwegian University of Science and Technology
Department of Materials Science and Engineering

Abstract

The traditional way of making manganese alloys are by reduction of different manganese oxides, iron oxides and silicon dioxide in a standalone electric furnace. That being said, manganese alloy producing companies may consider the implementation of a prereduction unit which will pre-reduce the ore before entering the electric furnace reducing the overall electrical consumption per ton alloy produced. How well the prereduction unit will contribute to the reduction of electrical consumption have previously been found to rely heavily on ore content. At this point, we should implement a reliable model which describes the process at hand inside a theoretical pretreatment unit.

An established framework describing the gaseous reduction of manganese oxide inside a prereduction unit has been made which relies on a continuous model taking the most important gas-solid reaction related phenomena into account. The model proved to be both fast and resilient. Several relevant cases were tested on the implemented continuous model to see how it performed. Some flaws were found. These were related to heat transfer and different kinetic parameters such as rate constants. There still exists some industrially relevant questions that go unanswered concerning prereduction. Here, our model can be used to assist further work and decisions on the matter.

Sammendrag

Den tradisjonelle måten å lage mangan legeringer på er ved reduksjon av forskjellige manganoksider, jernoksider og silikondioksid i en frittstående elektrisk ovn. Når det er sagt, kan manganlegeringsproduserende selskaper vurdere å implementere en forreduksjonsenhet som vil redusere malmen før den kommer inn i den elektriske ovnen som vil redusere det totale elektriske forbruket per tonn legering produsert. Hvor godt forreduksjonsenheten vil bidra til reduksjon av elektrisk forbruk har tidligere blitt funnet å ha en sammenheng med malminnholdet. På dette tidspunktet bør vi implementere en pålitelig modell som beskriver prosessen av den teoretiske forreduksjonsenheten.

Et etablert rammeverk som beskriver den gassavhengige reduksjonen av manganoksid inne i en pre-reduksjonsenhet er blitt utført. Den bygger på en kontinuerlig modell som tar hensyn til de viktigste gass-faste reaksjonsrelaterte fenomenene. Modellen viste seg å være både rask og robust. Flere relevante scenarier ble testet på den implementerte kontinuerlige modellen for å se hvor godt den presterte. Noen modellbegrensninger ble funnet. Disse var relatert til varmeoverføring og forskjellige kinetiske parametere som hastighetskonstanter. Det finnes fortsatt noen industrielle relevante spørsmål som er ubesvarte når det gjelder forreduksjon. Her kan vår modell brukes til å bistå videre arbeid og avgjørelser i saken.

Preface

The work presented in this thesis was produced at the Department of Materials Science and Engineering, at the Norwegian University of Science and Technology as a partial requirement to obtain the degree Master of Science and Technology.

Objectives

The main objectives of this master thesis are

- Implement a fast and resilient model describing prereduction of manganese ores using different blends of CO, H_2, CO_2 and H_2O.
- Run different cases using the implemented model to see how the model performs relative to experimental data.

Limitations

During this study, no experiments were conducted. That being said, when modeling, parallel experiments similar to what is modeled should be considered. This will improve any judgemental decisions during the modeling process. However, this study relies solely on past experiments and data gathered from the industry.

Structure of the Report

The rest of the report is structured into six chapters.

Chapter 1: Introduction gives an introduction to the topics at hand. These are are mainly related to prereduction of manganese ores by a specified syngas. Both aim of thesis and a detailed literature survey will be presented.

Chapter 2: Theoretical Framework is devoted to the reduction kinetics related to gas-solid reaction systems involving single particles. Two models will be presented, the shrinking-core model and a more general continuous model. A presentation of the numerical treatment used in this report will be given as well.

Chapter 3: Modeling The Prereduction Unit gives a suggested approach to modeling the prereduction unit.

Chapter 4: Results. In this chapter, different cases using real experimental values taken from past experiments as input values for our model will be presented. Industrial data will be used as well.

Chapter 5: Discussion will present a discussion on model limitations and how these can be solved in the future.

Chapter 6: Conclusions presents a brief conclusive summary of the findings in this report as bullet points.

Acknowledgments

First of all, I would like to thank my supervisor, Associate Professor Kristian Etienne Einarsrud at the Department of Materials Science and Engineering, NTNU, for his endless patience and support throughout the entire semester.

I would also like to thank my co-supervisor, Senior Research Scientist Halvor Dalaker, SINTEF, for his supplementary guidance and fruitful discussions.

Finally I wish to thank my family and friends for being so supportive and encouraging during my work.

Contents

List of Figures

Chapter 1

Introduction

Manganese alloys are mostly used during the production of steel. Their important properties as a deoxidant and a sulphide former makes them essential during steel manufacturing. As of today several manganese alloys exist on the market. Manganese alloys are made by reduction of different manganese oxides, iron oxides and silicon dioxide. The metallization process of manganese oxide can be represented by the reaction path

$$MnO_2 \longrightarrow Mn_2O_3 \longrightarrow Mn_3O_4 \longrightarrow MnO \longrightarrow Mn \tag{1.1}$$

In the last step from MnO to Mn, solid carbon is needed. Therefore, carbon in appropriate proportions is added as reduction material. Both oxides and carbon are added together into the electric arc furnace as shown in **Figure** 1.1. The amounts of different oxides used in the process is dependent on product specifications.

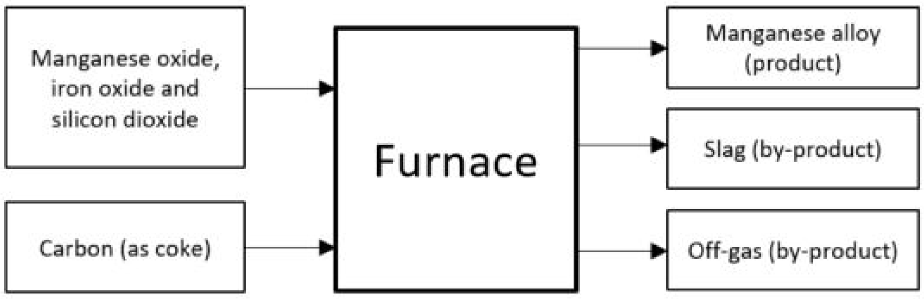

Manganese oxide, iron oxide and silicon dioxide
Carbon (as coke)
Furnace
Manganese alloy (product)
Slag (by-product)
Off-gas (by-product)

Figure 1.1: Simplified overview of input and output variables of manganese alloy production.

The furnace consumes a lot of electrical energy during the process which is trans-formed to heat making the reduction of oxides possible (Olsen et al. [1]). An important factor that should be noted is that carbon is very undesirable as a component in metal products. The use of carbon in the metallurgical industry serves only two purposes, remove oxygen from the ore and make sure it goes out of the furnace as gas. Carbon consumption and consequently CO_2 emissions in metal manufacturing are essentially process oriented and not product oriented [2]. So, the process, in general, is both energy demanding and environmentally unfavorable. That being said, manganese production is a major field in the Norwegian metallurgical industry and when considering the growing vision of a sustainable metallurgical industry that achieves CO_2-neutrality by 2050, specific energy-reducing process implementations are deemed necessary [3]. One such implementation is the supplementary pretreatment unit which could be added as an additional processing step connected to the infeed of an manganese alloy producing furnace. In principal, this unit could upgrade any commercial metallurgical manganese ore by driving off volatiles (moisture and CO_2), reducing the amount of excess oxygen (prereduction), and by improving the size distribution (agglomeration or sintering) to generally improve the performance during furnace operations. Today, there exist a handful of different technologies related to pretreatment that reduce the electrical power consumption and increase the productivity of manganese smelters (Gordon and Nell [4]). Such technologies depend on either calcination or agglomeration of manganese ore. However, in any pretreatment unit using methods of manganese ore thermal-treatment prior to smelting, some degree of prereduction is to be expected due to higher temperatures initiation reduction/decomposition paths based on atmosphere inside the pretreatment unit. This reaction path is somewhat different to the one already described (1.1), for instance, there may be no solid carbon present or high enough temperatures to start metallization of MnO. We then end up with the reaction path represented by

$$MnO_2 \longrightarrow Mn_2O_3 \longrightarrow Mn_3O_4 \longrightarrow MnO \tag{1.2}$$

How prereduction will affect the overall energy consumption is not that obvious as the reduction of higher manganese oxides to MnO is exothermic reactions. Nevertheless, in Japan by Kashima Works, prereduction of ore by coal and CO-rich off-gas from furnace operations is being done by a prereduction unit before entering the electric furnace. Raw materials going out of the prereduction unit will have a temperature of 600°C with zero water content and reduced amount of excess oxygen. In a study by Tangstad et al. [5] it was estimated that the prereduction unit reduced the total energy consumption by 20%. However, this estimate is highly dependent on charge (infeed) specifications. Nevertheless, such a decrease in energy consumption is worth pursuing in the Norwegian metallurgical industry as well.

1.1 Prereduction of Manganese Ores

During prereduction of manganese ores, four different Mn oxides will be present inside a given pellet/lump in different amounts depending on reduction-time, temperature and partial pressure of oxygen, p_{O2}. As seen in the already explained reduction paths, present Mn oxides are MnO_2, Mn_2O_3, Mn_3O_4 and MnO. Additionally there will be different iron oxides present as well. However, these are not considered in this report but can be implemented in the same way as for the various Mn oxides. Manganese oxide will start to decompose at high temperatures. Determined by Wang and Sundman [6], MnO_2 will start to decompose at about 697 K, Mn_2O_3 at 1161 K. This relates well with **Figure 1.2** when considering oxygen pressure of 0.21 atm ($log p_{O2} = -0.677$) which determines the lowest possible thermal decomposition temperatures of the respective oxides in air at atmospheric pressure (Berg [7]). For complete reduction to MnO, it is required to have a reducing agent, in this case a syngas. A syngas is a gas mixture consisting primarily of hydrogen and carbon monoxide and is often used as fuel to generate power. The suggested syngas used for reducing the different Mn oxides will be a gas mixture of CO, H_2, CO_2 and H_2O. The reactions of CO and H_2 with Mn oxides are given by Reactions 1.3 - 1.8 (Lobo [8]).

$$MnO_2 + \frac{1}{2}CO\,(g) = \frac{1}{2}Mn_2O_3 + \frac{1}{2}CO_2\,(g) \qquad \Delta H^\circ_{298} = -99.9 \text{ kJ} \qquad (1.3)$$

$$\frac{1}{2}Mn_2O_3 + \frac{1}{6}CO\,(g) = \frac{1}{3}Mn_3O_4 + \frac{1}{6}CO_2\,(g) \qquad \Delta H^\circ_{298} = -31.3 \text{ kJ} \qquad (1.4)$$

$$\frac{1}{3}Mn_3O_4 + \frac{1}{3}CO\,(g) = MnO + \frac{1}{3}CO_2\,(g) \qquad \Delta H^\circ_{298} = -16.9 \text{ kJ} \qquad (1.5)$$

$$MnO_2 + \frac{1}{2}H_2(g) = \frac{1}{2}Mn_2O_3 + \frac{1}{2}H_2O(g) \qquad \Delta H^\circ_{298} = -80.3 \text{ kJ} \tag{1.6}$$

$$\frac{1}{2}Mn_2O_3 + \frac{1}{6}H_2(g) = \frac{1}{3}Mn_3O_4 + \frac{1}{6}H_2O(g) \qquad \Delta H^\circ_{298} = -22.2 \text{ kJ} \tag{1.7}$$

$$\frac{1}{3}Mn_3O_4 + \frac{1}{3}H_2(g) = MnO + \frac{1}{3}H_2O(g) \qquad \Delta H^\circ_{298} = -4.41 \text{ kJ} \tag{1.8}$$

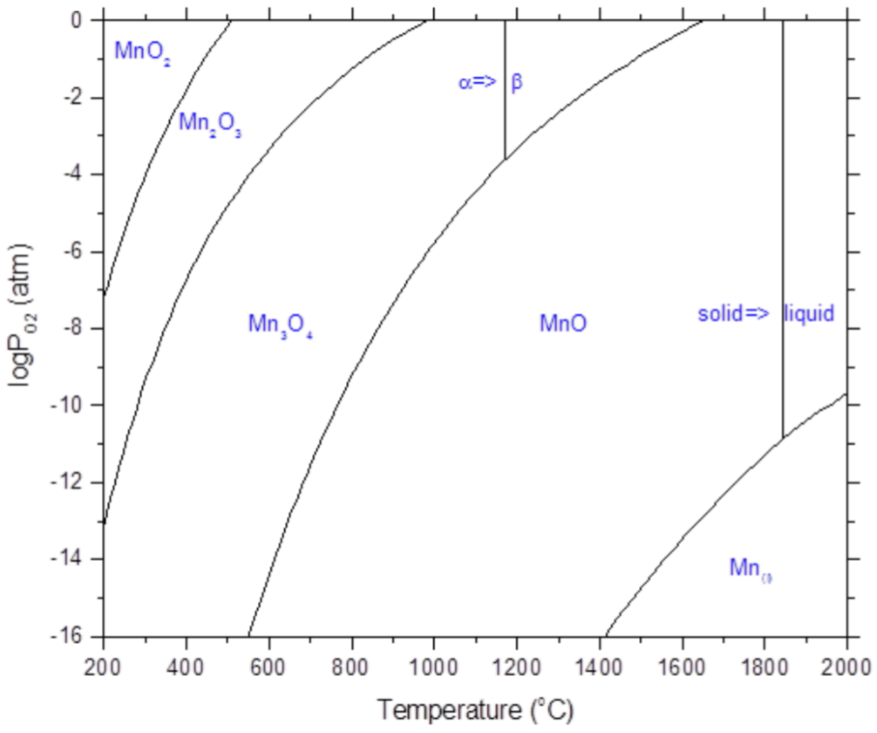


Figure 1.2: Calculated stability areas in the Mn-O system [1].

All of these reactions are exothermic and will produce heat to the system. Due to this, one has to assume that the temperature of the areas at which the reactions takes place to be higher than temperatures outside the solid. Now, with gas mixtures containing CO, H_2, CO_2 and H_2O, the water-gas shift reaction (WGSR) becomes important (Reaction 1.9 [9]). This reaction consumes carbon monoxide to produce hydrogen in the presence of water vapour. As implied by Lobo [8], the water-gas shift reaction is considered to be at equilibrium during prereduction.

$$CO + H_2O \rightleftharpoons CO_2 + H_2 \qquad \Delta H^\circ_{298} = -41.09 \text{ kJ} \tag{1.9}$$

Later on in the report, when we want to implement this phenomenon into our prereduction model we shall make good use of the Langmuir Hinshelwood Model (Smith et al. [10]). This model estimates the *reaction rate* of Reaction 1.9 which consequently favors one side of the reaction dependent on partial pressures and temperature. A more detailed explanation of reactions rates and rate constants will be given in Chapter 2. However, the WGSR *reaction rate* estimated by the Langmuir Hinshelwood Model is given by

$$rate_{WGSR} = \frac{k_{WGSR} K_{CO} K_{H_2O} \left[P_{CO} P_{H_2O} - \frac{P_{CO_2} P_{H_2}}{K_{eq}} \right]}{(1 + K_{CO} P_{CO} + K_{H_2O} P_{H_2O} + K_{CO_2} P_{CO_2})^2} * \frac{\rho_{cat}}{60} \tag{1.10}$$

where k_{WGSR} is the WGSR *rate constant* given by

$$k_{WGSR} = exp\left(-\frac{29364}{1.987 * T} + \frac{40.32}{1.987} \right) \tag{1.11}$$

in $mol/gcat./min$, together with P_i and K_i which is the partial pressure and adsorption equilibrium constant of component i respectively (Pedolski and Kim [11]). The different adsorption equilibrium constants are given by

$$K_{CO} = exp\left(\frac{3064}{1.987 * T} - \frac{6.74}{1.987} \right) \tag{1.12}$$

$$K_{CO_2} = exp\left(\frac{12542}{1.987 * T} - \frac{18.45}{1.987} \right) \tag{1.13}$$

$$K_{H_2O} = exp\left(-\frac{6216}{1.987 * T} + \frac{12.77}{1.987} \right) \tag{1.14}$$

Lastly, the equilibrium constant is given by

$$K_{eq} = exp\left(\frac{4577.8}{T} - 4.33 \right) \tag{1.15}$$

1.2 Background

Now that we have undergone a brief introduction to the general concept of prereduction, we can now begin to limit ourselves to a manageable workload as well explain what we hope to achieve with this study based on the already presented objectives.

Aim

When considering a unit operating together with other units inside a larger system, whether it is a pretreatment unit or a furnace, detailed information regarding the different variables affecting the units output as well as the output itself are seen as critical data when considering overall system performance and predictability. For instance, this information could be average particle concentration of any oxide inside a pretreatment unit or temperature gradients inside a furnace which are all variables that directly affect the overall system behavior. In other words, the system at hand is highly complex with different variables affecting different input and output streams, and we want to know as much about it as possible to be able to predict every outcome.

Now that we are considering to implement a pretreatment unit to our system given in **Figure 1.1**, staying true to our statement above, we need to implement a reliable model which describes the process at hand inside a theoretical pretreatment unit pre-reducing manganese oxide. We would then like to know what kind of oxides that are present inside a hypothetical particle of any given size at any given time and temperature using a specified initial syngas mixture. This can be done by making a progressive-conversion model (or continuous model) which means that solid reactant is converted continuously throughout the particle (Levenspiel [12]). This is, of course, no straight forward task as its mathematical complexity is endless.

When a respectable model has been made, we will start by presenting model results based on both experimental data and data gathered from the industry concerning the potential of using off-gas being as syngas in our hypothetical pretreatment unit.

Literature Survey

The main books that were used on the mathematically complex topics, are as follows: Transport Phenomena by Robert Byron Bird, Edwin N. Lightfoot and Warren A. Stewart [13], and Gas-Solid Reactions by Julian Szekely, James W. Evans and Hong Yong Sohn [14]. The first book is more general in the sense that it addresses all the different scenarios one may encounter when working with transport phenomena of liquids and gases. The second book mentioned has a narrower vision only focusing on different gas-solid reaction systems. However, both books were used to a large extent when validating equations and theories provided in this report.

To begin with we would want to make a simple scheme containing the broad idea of the process. For instance, from Szekely et al. [14], we know that the rate of a gas-solid reaction is governed by the following individual steps:

- External mass transfer from bulk to particle surface
- Mass transfer of gaseous reactant within pores of the solid matrix
- Adsorption of the gaseous reactant
- Chemical reaction
- Desorption of the gaseous product
- Mass transfer of gaseous reaction product within pores of the solid matrix

Note that the steps representing adsorption, desorption and chemical reactions are generally looked upon as part of only one rate-determining step which is the chemical reaction. Now, there exist other phenomena that may affect the rate-determining steps mentioned. These can be:

- Heat transfer
- Structural changes of the solid
- Nucleation effects.

All the phenomena listed above are expected to occur in our system, and we can now begin to grasp the complexity of the problem. Note that each of these phenomena are discussed further in Chapter 2, except the nucleation effects which are neglected in this case. However, such effects could be added in future work when the more important phenomena are well implemented.

Previous work tackling similar problems presented in this report will be used extensively. For instance, the work done by Berg [7] were used when implementing different kinetic parameters regarding Reactions 1.3 - 1.5. Kinetic parameters for the reduction of manganese oxides by a syngas containing only H_2 proved difficult to find. Nevertheless, this problem will be discussed later on in the report. Now, let us describe the work done by Berg [7] to get a better understanding of his results used in this report.

The main objective of his work (as he describes it), was to elucidate the kinetics and mechanisms of the gas-solid reactions taking place in the ferromanganese process. Prereduction of ore with carbon monoxide was heavily emphasized on. Reduction of various commercial manganese ores with mixtures of CO was studied. A thermogravimetric apparatus was used to estimate the weightloss of the samples during reduction/decomposition. Different compositions of CO-CO_2 mixtures were used together with varying temperatures between $700-1100°C$. There are a lot of results given in his work. Thus, we will only focus on the weightloss vs. time plots describing the different reduction/decomposition paths represented by

$$MnO_2 \longrightarrow Mn_2O_3 \tag{1.16}$$

$$Mn_2O_3 \longrightarrow Mn_3O_4 \longrightarrow MnO \tag{1.17}$$

Now, three types of commercial, high-grade manganese ore were investigated in the work conducted by Berg [7]. However, we will only consider results concerning the Groote Eylandt ore type (BHP). These ore lumps are high in manganese and should therefore be a good representation of reality concerning the reduction/decomposition paths given above. Chemical analysis of different BHP ore lumps can be found on page 35 in [7]. Different results concerning the reduction/decomposition paths found in [7] are presented in **Figure 1.3**. Here we see the reduction/decomposition rate for both reaction paths. The experimental results presented by Berg should also give an insight into what to expect when reducing manganese oxide in the temperature range 700-1100°C.

In the work conducted by Lobo [8], pellets of Nchwaning ore were reduced using a syngas similar to our already defined gas mixture of CO, H_2, CO_2 and H_2O only with varying partial pressures of oxygen to determine the driving forces of reduction. The porosity of the pellets were considered to be between 30-34 %. Lobo also emphasized on the effects caused by the Boudouard reaction, Reaction 1.18, as it can precipitate carbon inside the pellet by consumption of CO.

$$2\,CO\,(g) \longrightarrow CO_2\,(g) + C\,(s) \tag{1.18}$$

This reaction is neglected in our case but could be implemented in future work. The results of interest used in this report are those of pellet reduction by either CO-rich gas or pure H_2. The results can be seen in **Figure 1.4** (note that Lobo presents his results in mass change vs. time and not weightloss vs. time). One should also note that the pellets were first roasted at 800°C for 2 hours prior to reduction which resulted in a slight mass loss, however, this mass loss was argued to be a consequence of the calcination of carbonates due to decreased CO_2 content after roasting. This is also the reason why the results in **Figure 1.4** starts with a positive mass, which is equivalent to the mass loss prior to reduction. Reduction is then set to start at a mass loss of 0. It was concluded that reduction of manganese oxides is significantly affected by the degree of oxidation in the gas [8].

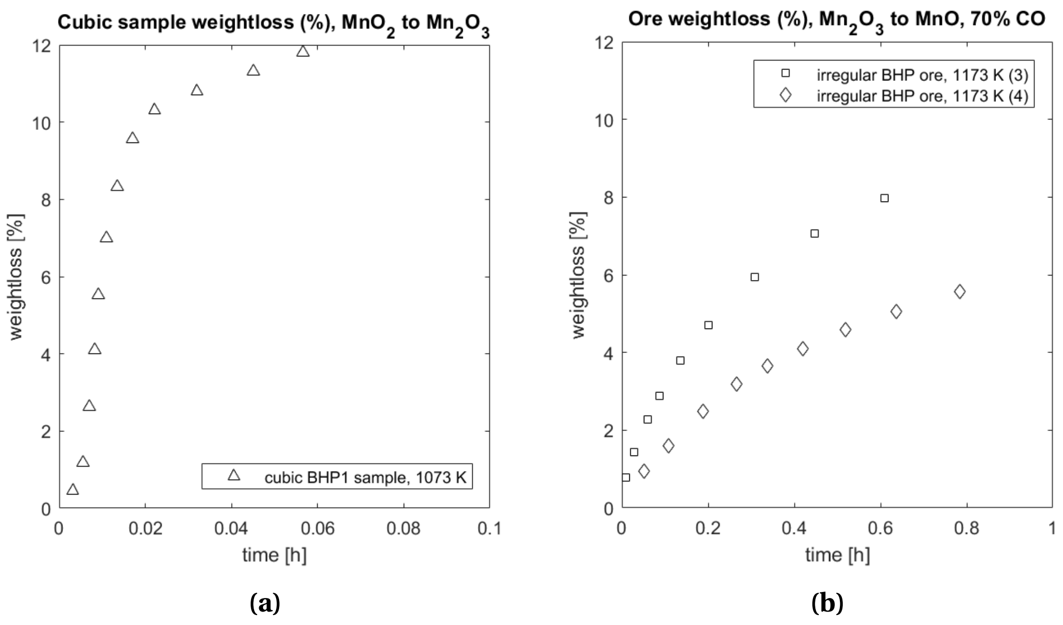


Figure 1.3: Weightloss vs. time for a cubic BHP1 sample, dimension range: 1-1.5 cm, CO atmosphere, 1073 K (a), and two irregular BHP pieces, dimension range: 1.2-1.7 cm, 70% CO, 1173 K (b). Results taken from page 41-42 [7].

Work done by Schanche [15] should also contain useful information regarding prereduction considering his effort on making a model representing the submerged arc furnace. The model consists of different submodels where one of them takes prereduction into account. The foundation is then already set which leads to inspirational inputs from a code already produced on the matter, however, it is not a continues model. Work done by Tangstad et al. [5] shows that there is no extra energy related savings in having an increased degree of prereduction in the prereduction unit. This might

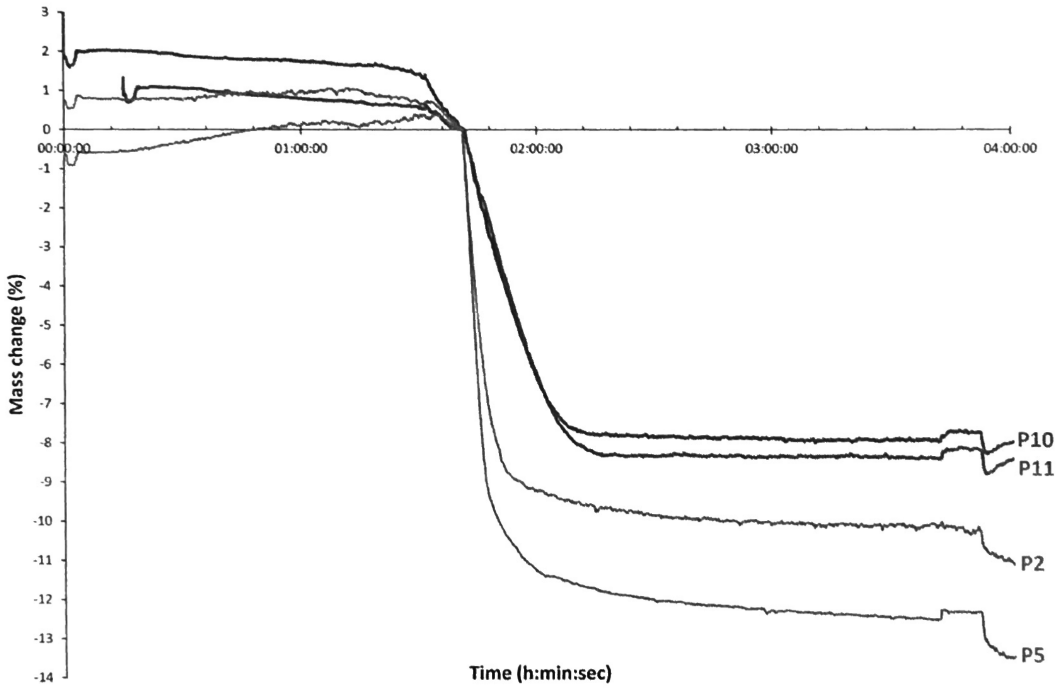


Figure 1.4: Mass change for pellets in heavily oxidizing conditions (P10 and P11) with pure H_2 (P2) and CO rich gas (P5) for comparison. Dimension range: 10-14 mm. Taken from [8].

sound demotivating considering our aim to produce a model describing reduction of manganese oxide inside a pretreatment unit. However, there exist other advantages with a separate prereduction unit which strengthens the drive to make a reliant model simulating prereduction inside a pretreatment unit. Preparatory work (by the same authors [16]), emphasized on using CO-rich off-gas from a furnace in other process related units which then had the potential of reducing the overall specific energy (energy per tonne of metal produced). So, when considering using the off-gas as syngas in a pretreatment unit similar to Kashima Works, it becomes important to have information about the off-gas content.

What Remains to be Done?

After the review of previous work on the topic of prereduction, it becomes clear that no documented attempt has been made so far on making a detailed continuous model describing the prereduction of manganese ores using CO, H_2, CO_2 and H_2O gas mixtures taking every mentioned phenomena into account described in Section 1.2. By building on the foundation of other reports, the amount of work becomes manageable as this report intends to add insight and experience to the topics presented.

Chapter 2

Theoretical Framework

The first part of this chapter is devoted to the reduction kinetics related to gas-solid reaction systems involving single particles. Two models will be presented here, the shrinking-core model, and a more general continuous model. At the end of this chapter, a presentation is given of the numerical treatment used in the modeling part of this report.

2.1 Reduction Kinetics

2.1.1 Mass Transfer Between a Particle Surface and a Gas Stream

Consider the gas-solid reaction located on a sphere shaped particle surface

$$\mathrm{aA\,(s) + bB\,(g) = cC\,(s) + dD\,(g)} \tag{2.1}$$

Here, the particle which is subjected to a gaseous reactant B will be affected by the rate at which the gaseous reactant is transferred from the bulk of the gas to the surface of the particle or the rate at which the gaseous product D is removed from the surface of the solid reactant. Regarding this matter, we shall use an empirical approach which depends on mass transfer coefficients. As proposed by Bird et al. [13], the rate at which a gas is being transferred from a gas in motion (reactant B), through a gas film of unknown thickness δ (see **Figure 2.1**) onto a solid surface A per unit solid surface area is given by

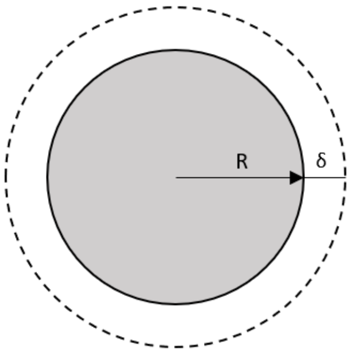


Figure 2.1: Spherical particle with radius R surrounded by a gas film of thickness δ.

$$N_B = \bar{k}_B(c_{B,surface} - c_{B,bulk}) + \chi_B(N_B + N_D) \tag{2.2}$$

where $c_{B,j}$ is the concentration of specie B at location j. N_D is the mass flux of reaction product D from Reaction 2.1 and χ_B is the mole fraction of B at the solid surface given that the mass flux N_B and N_D is given in the corresponding units which is moles per unit area per unit time. $\bar{k}_B$ is the *mass transfer coefficient* of specie B. The first term on the r.h.s. of Equation 2.2 describes the molecular flux with the concentration gradient between surface and bulk concentration of B as its driving force. The second term describes the convective flux which is tied together with the global bulk flow. When having *equimolar counter diffusion* i.e. when $-N_B = N_D$, the convective term disappears. Note that if the concentration gradient had units of mass per volume, the convective term would not necessarily disappear. This reflects that the center of mass could be displaced even though we have *equimolar counter diffusion* of gaseous species (Bakken [17]).

Now the task lies in obtaining the *mass transfer coefficient* $\bar{k}_i$ (in this case, $i = B$), which depends on the physical conditions of the gas phase and the geometry of the solid (Berg [7]). This can be done from an empirical correlation built up by dimensionless groups describing the system. Considering the Buckingham's π-theorem we have in this case a number of 6 mutually independent dimensionless groups which is linked together with the number of physical variables and base units in the system [17]. Here, we shall use the *Schmidt number*

$$Sc = \frac{\nu'}{D_{ij}} \tag{2.3}$$

where ν' is the *kinematic viscosity* (which will be defined later, Equation 2.12) and D_{ij} is the diffusion coefficient of the binary gas mixture of specie i in j, together with the *Reynolds number*

$$Re = \frac{V_\infty L}{\nu'} \tag{2.4}$$

where V_∞ is the bulk velocity and L is the *characteristic dimension* (diameter for spheres), to estimate the *Sherwood number*

$$Sh = \frac{\bar{k}_i L}{D_{ij}} \tag{2.5}$$

With Equation 2.5 it is possible to calculate $\bar{k}_i$ when the *Sherwood number* is obtained from the empirical correlation proposed for spherical particles [see 14, Chap. 2]

$$Sh = 2 + BRe^m Sc^n \tag{2.6}$$

where B, m and n are empirical constants. Note that this equation comes from

$$Sh = A + BRe^m Sc^n + CGr^p Sc^q \tag{2.7}$$

where the second term, $BRe^m Sc^n$, on the r.h.s. of Equation 2.7 represents *forced convection* and the third term, $CGr^p Sc^q$ (C, p and q are empirical constants), represents *natural convection*. Here, Gr is the *Grashof number*. Equation 2.6 simply neglects the last term due to *forced convection* being of much stronger influence on the system in general. Nevertheless, it is not straight forward obtaining the remaining empirical constants B, m and n in Equation 2.6. Luckily, due to the industrial importance of operations involving mass transfer from drops or particles, it is quit easy to find correlations in analogy to our specific case. From a study by Ranz and Marshall [18] (as cited in [14]) the experimental results were correlated with equation

$$Sh = 2.0 + 0.6 Re^{1/2} Sc^{1/3} \tag{2.8}$$

which is the equation that will be used when describing the mass transfer from bulk gas concentration to a particle surface. Note that when the bulk velocity of the gaseous species is zero, $Sh = 2$.

Estimation of Diffusivities

When estimating the diffusion coefficient D_{ij} in Equations 2.3 and 2.5, Todd and Young [19] proposed an alternative regarding non-polar binary gas mixtures by using an expression given by Fuller et al. [20] (as cited in [19])

$$D_{ij} = \frac{0.00143 * T^{1.75}}{P_b MW_{ij}^{1/2} \left(V_i^{1/3} + V_j^{1/3}\right)^2} \tag{2.9}$$

$$MW_{ij} = 2\left[\frac{1}{MW_i} + \frac{1}{MW_j}\right]^{-1} \tag{2.10}$$

where P_b is pressure in bar, T is temperature in kelvin, MW_i and MW_j are molecular weights in g per mol together with their *special atomic diffusion volumes* V_i and V_j which are tabled values found in Thibodeaux [21] of common gas species. If the gas mixture is not binary but consists of more than two species we can make use of Wilke's equation which is derived from the Stefan-Maxwell diffusion equation to estimate the *molecular diffusion coefficient* [22], we then have

$$D_{i,mixture} = \frac{1 - \chi_i}{\sum_{j=2}^{NG} \chi_j / D_{ij}} \tag{2.11}$$

where NG is the total number of gas species, and D_{ij} is the diffusivity for the binary pair i-j.

Estimation of Gas Mixture Viscosity

The *kinematic viscosity* is defined as

$$v' = \frac{\mu}{\rho} \tag{2.12}$$

where μ is the *dynamic (absolute) viscosity* and ρ is the density of the gas. The *dynamic viscosity* is a fluid property which is a measure of internal resistance i.e. shear stress divided by shear rate (Bird et al. [13]). When more than two gaseous species are present in a mixture with the assumption of constant low pressure, the Herning-Zipperer [23] correlation can be used

$$\mu_{mixture} = \frac{\sum_{i=1}^{NG} \chi_i \mu_i MW_i^{1/2}}{\sum_{i=1}^{NG} \chi_i MW_i^{1/2}} \tag{2.13}$$

The viscosity for a single specie i (μ_i) can be calculated using Sutherland's formula [24]

$$\frac{\mu_i}{\mu_{i,0}} = \frac{T_0 + C_{i,s}}{T + C_{i,s}} \left(\frac{T}{T_0}\right)^{3/2} \tag{2.14}$$

where $\mu_{i,0}$ is the reference viscosity at reference temperature T_0 (in degrees Rankine) and $C_{i,s}$ is the *Sutherland's constant* for specie i. Values of the *Sutherland's constant* $C_{i,s}$ are taken from Crane [25]. Reference viscosities together with their reference temperatures can be found in the handbook of CRC [26].

2.1.2 Heat Transfer Between a Particle Surface and a Gas Stream

Let us now consider the same particle as in Section 2.1.1. We then have a particle surface of temperature T_s that is in contact with a moving (or stagnant) gas which has a bulk temperature T_b. The heat flux from the solid surface may then be written as (Bakken [17])

$$q = \bar{h}_c(T_s - T_b) \tag{2.15}$$

where $\bar{h}_c$ is the *heat transfer coefficient*. Note that there is a complete analogy between the heat and mass transfer variables. Therefore we also get 6 independent dimensionless groups for the heat transfer correlation [17]. Here, we shall use the *Prantl number*

$$Pr = \frac{v' C_p \rho}{\hat{k}} \tag{2.16}$$

where $\hat{k}$ is the *thermal conductivity* and C_p is the *heat capacity* at constant pressure, together with the *Reynolds number* (Equation 2.4) to estimate the *Nusselt number*

$$Nu = \frac{\bar{h}_c L}{\hat{k}} \tag{2.17}$$

We shall again use the semi-empirical formula from the study by Ranz and Marshall [18] (as cited in [14]), Equation 2.8, only now in correlation with the heat transfer

$$Nu = 2.0 + 0.6 Re^{1/2} Pr^{1/3} \tag{2.18}$$

Note that we have not accounted for any convection heat transfer inside nor outside the particle due to the fact that we are only considering heat transfer in the solid particle.

Estimation of Thermal Conductivity and Heat Capacity

When estimating both *thermal conductivity* and *heat capacity* for a gas mixture, it is common to use mole fraction weighted values given by (Poling et al. [27])

$$\hat{k}_{mixture} = \sum_{i=1}^{NG} \chi_i \hat{k}_i \tag{2.19}$$

$$C_{p,mixture} = \sum_{i=1}^{NG} \chi_i C_{p,i} \tag{2.20}$$

$\hat{k}_i$ and $C_{p,i}$ are both dependent on temperature and can be estimated by regression formulas which can be found tabulated. For instance, in Felder and Rousseau [28], considering the thermal conductivity temperature dependence, the regression formula is on the form

$$C_{p,i} = a_i + b_i T + c_i T^2 + d_i T^3 \tag{2.21}$$

where a_i, b_i, c_i and d_i are tabulated constants for specie i. Similar regression formulas for the thermal conductivity can be found in Poling et al. [27] or formulas could be made from scratch using curve fitting on experimental data sets found in the literature.

2.1.3 Diffusion of Gaseous Reactants and Products Through a Porous Solid Matrix

This section will focus on a gas mixture in a porous medium from a possible gas-solid reaction, i.e. Reaction 2.1. The gas stream going through the porous medium is presumed to have its own chemical reactions (WGSR), but for now, lets assume that our gas stream is in equilibrium and in steady state going through a medium of unchanging structure. We are also considering the process to be an isobaric process which indicates a constant pressure inside the particle. Now, regarding pore diffusion, there exist many complex models which tries to mimic this phenomena, see Szekely et al. [14]. However, in this report the porous medium characteristics will be described with a constant ϵ, as well as a rate-determining function $f(X)$ (see Section 2.1.5). In the present section we will later see that ϵ is directly tied to the rate at which the concentration of gas species change over time. So, ϵ is the local composition dependent porosity which is given by (Melchiori and Canu [29])

$$\epsilon = 1 - \sum_{j=1}^{NS} c_j * \frac{MW_{(kg),j}}{\rho_{I,j}} \tag{2.22}$$

where c_j is the molar concentration of j, and $\rho_{I,j}$ its intrinsic density (that of a dense solid without pores). NS is the total number of solids present.

The Equation of Continuity

Now, consider a gas mixture going through a porous medium. If we set up a mass balance for the gas mixture over a control volume $\triangle V = \triangle x \triangle y \triangle z$, we get (Bird et al. [13])

$$\begin{aligned} \triangle x \triangle y \triangle z \frac{\partial \rho}{\partial t} = & \triangle y \triangle z \left[(\rho \mathrm{v}_x)|_x - (\rho \mathrm{v}_x)|_{x+\triangle x} \right] \\ & + \triangle z \triangle x \left[(\rho \mathrm{v}_y)|_y - (\rho \mathrm{v}_y)|_{y+\triangle y} \right] \\ & + \triangle x \triangle y \left[(\rho \mathrm{v}_z)|_z - (\rho \mathrm{v}_z)|_{z+\triangle z} \right] \end{aligned} \tag{2.23}$$

where v_i is the gas velocity in direction i. Equation 2.23 simply explains the rate of increase of mass $(\partial \rho / \partial t)$ inside $\triangle V$ which is dependent on the rate of mass going into $(\rho \mathrm{v}_i)|_i)$ and out $(\rho \mathrm{v}_i)|_{i+\triangle i})$ of the volume. Now, divide the entire equation by $\triangle x \triangle y \triangle z$ and take the limit as $\triangle x$, $\triangle y$, and $\triangle z \to 0$. Using the definition of a derivative of a function we get

$$\frac{\partial \rho}{\partial t} = -(\nabla \cdot \rho \mathbf{v}) \tag{2.24}$$

Let us now consider the porosity of the medium as well as switching out the density with the concentration of specie i, c_i. We then get

$$\frac{\partial \epsilon c_i}{\partial t} = -(\nabla \cdot \boldsymbol{N}_i) \tag{2.25}$$

where $\boldsymbol{N}_i$ is the mass flux, or in this case the molar flux of specie i as a vector field.

Similar to Equation 2.2, $\boldsymbol{N}_i$ can be expressed as the sum of the diffusive term and the the convective term given by

$$\boldsymbol{N}_i = \boldsymbol{J}_i^* + \chi_i \boldsymbol{N} \tag{2.26}$$

where $\boldsymbol{N}$ is the global molar flux which controls the convection. The diffusive term $\boldsymbol{J}_i^*$ in this case can be calculated using the generalized Fick's law which can be found in [29]. However, in this report we are simplifying the matter by only considering the *effective diffusion coefficient* $D_{i,eff}$, which leads to an ordinary representation of Fick's law

$$\boldsymbol{J}_i^* = -D_{i,eff} \nabla c_i \tag{2.27}$$

$$D_{i,eff} = D_{i,mixture} * \frac{\epsilon}{\tau} \tag{2.28}$$

where τ is the local tortuosity of the solid. Now, if we use the del operator in spherical coordinates for Equation 2.25 we get

$$\frac{\partial \epsilon c_i}{\partial t} = -\frac{1}{r^2}\frac{\partial (r^2 N_{i,r})}{\partial r} + \frac{1}{r\sin\theta}\frac{\partial (N_{i,\theta}\sin\theta)}{\partial \theta} + \frac{1}{r\sin\theta}\frac{\partial N_{i,\varphi}}{\partial \varphi} \tag{2.29}$$

However, the molecular diffusion is assumed going only radially outward or inward which cancels out the last two terms on the r.h.s. of Equation 2.29. The same procedure can be done for Equation 2.27 by changing the gradient operator and applying the same assumption. We then end up with

$$\frac{\partial \epsilon c_i}{\partial t} = -\frac{1}{r^2}\frac{\partial (r^2 N_{i,r})}{\partial r} \tag{2.30}$$

$$N_{i,r} = -\left(D_{i,eff}\frac{\partial c_i}{\partial r}\hat{r}\right) + \chi_i N_r \tag{2.31}$$

2.1.4 Conductive Heat Transfer in a Porous Solid Matrix

Now, consider the same solid particle. If the solid is a continuous phase containing pores with a gas mixture that has a thermal conductivity much lower than that of the solid phase we can assume the following relation (Szekely et al. [14])

$$\hat{k}_{eff} = \hat{k}_s(1-\epsilon) \tag{2.32}$$

where $\hat{k}_{eff}$ is the *effective thermal conductivity* and $\hat{k}_s$ is the *thermal conductivity* of the solid without pores. In analogy with the mass balance describing the diffusion of gaseous reactants in a porous solid, a similar balance equation could be used to describe the accumulated enthalpy per unit of time in the same control volume $\triangle V$. Following the same procedure, we get

$$\frac{\partial \rho h_e}{\partial t} = -(\nabla \cdot \boldsymbol{q}^*) \tag{2.33}$$

where h_e is the specific enthalpy and $\boldsymbol{q}^*$ is the heat flux given as a vector field which can be described using Fourier's law and adding a convective term $\rho h_e \mathbf{v}$ (Bakken [17])

$$\boldsymbol{q}^* = -\hat{k}_{eff}\nabla T + \rho h_e \mathbf{v} \tag{2.34}$$

Assuming only radially outward/inward heat transfer, in spherical coordinates we get

$$\frac{\partial \rho h_e}{\partial t} = -\frac{1}{r^2}\frac{\partial (r^2 q_r^*)}{\partial r} \tag{2.35}$$

$$q_r^* = -\left(\hat{k}_{eff}\frac{\partial T}{\partial r}\hat{r}\right) + \rho h_e \mathrm{v}_r \tag{2.36}$$

Note that we only consider thermal conductivity in the solid components inside the particle, we can then neglect any convective terms related to heat transfer.

2.1.5 Gas-Solid Chemical Reactions

Now that the gaseous reactants have traveled from the bulk of a moving (or stagnant) gas, through the pores of a product/ash solid matrix covering the remaining unreacted solid (see **Figure** 2.2), it is necessary to consider the surface reactions controlling the rate at which the unreacted solid is consumed. For a reaction to happen between a gas and a solid surface some sort of contact must take place between the two. Here, both *chemical adsorption* and *physical adsorption* comes into play (Szekely et al. [14]). However, the surface reaction alone is assumed controlling the kinetics of the system which makes both presented terms obsolete. We are then left with (for Reaction 2.1)

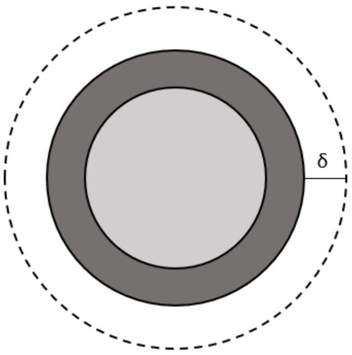


Figure 2.2: Partly reacted particle with an unreacted core (light grey) covered by a product/ash layer (dark grey) surrounded by a gas film of thickness δ.

$$\frac{dc_A}{dt} = s_A \tag{2.37}$$

where s_A describes the rate at which the concentration of solid A changes over time. This can be described by introducing the *reaction rate* of the reaction/reactions assumed present in the system.

Reaction Rate

First, let us consider only one gas-solid chemical reaction present in our system which is a general second order chemical reaction

$$\mathrm{aA\,(s) + bB\,(g)} \xrightleftharpoons[\mathrm{k}_b]{\mathrm{k}_f} \mathrm{cC\,(s) + dD\,(g)} \tag{2.38}$$

Here, *rate constants* k_f and k_b are for the forward and backward reaction paths respectively. The *reaction rate* can be described as [30]

$$rate = k_f c_A^s c_B^t - k_b c_C^u c_D^v \tag{2.39}$$

Let us assume that Reaction 2.38 goes completely to the right. This leads to a simplification of Equation 2.39 which implies that the *reaction rate* can be described assuming irreversibility and first order in the concentration of the gas reactant B as well as including a nonlinear function of solid conversion X. This is similar to what was stated in [29] on the same subject. We then have

$$rate = k * f(X) * c_B \tag{2.40}$$

$$X = 1 - \frac{c_A}{c_{A,t_0}} \tag{2.41}$$

where the *rate constant* k is now controlling the consumption of solid A in Reaction 2.38. This value is highly dependent on temperature and can be described by the Arrhenius equation [30]

$$k = k_0 e^{-E_\alpha/(RT)} \tag{2.42}$$

where k_0 is the *pre-exponential constant* and E_α is the *activation energy* of the reaction. In a closed system under isochoric conditions (constant volume) the *reaction rate* can be described through the reaction balance as defined in [see 30, Chap. 12]

$$rate = -\frac{1}{a}\frac{dc_A}{dt} = -\frac{1}{b}\frac{dc_B}{dt} = \frac{1}{c}\frac{dc_C}{dt} = \frac{1}{d}\frac{dc_D}{dt} \tag{2.43}$$

By implementing Equation 2.40 as well as multiplying both sides of Equation 2.43 by the boundary constricted volume, we get

$$V\bar{R} = A\bar{R}' = -\frac{1}{a}\frac{dn_A}{dt} \tag{2.44}$$

where the $rate$ is presented as per volume, $\bar{R}$, and as per surface area, $\bar{R}'$. Which fully written out looks like

$$Vk * f(X) * c_B = Ak' * f(X) * c_B = -\frac{1}{a}\frac{dn_A}{dt} \tag{2.45}$$

where k and k' is the *volumetric rate constant* and *superficial rate constant* respectively. The function $f(X)$ is still not accounted for. However, $f(X)$ can be seen as a description of how much of solid A that is present depending on local structure and porosity. In this case one could use the grain model to represent $f(X)$, proposed by Szekely et al. [see 14, Chap. 4]. Nevertheless, a simplified version which only considers the kinetic component can be used [29], which is given by

$$f(X) = (1 - X)^\alpha \tag{2.46}$$

This equation takes into account that there is no diffusion resistance inside the micro-grains, only intergrain diffusion is considered. Nevertheless, one should note that $f(X)$ is highly dependent on shape which is baked into the α variable which for spherical shape, $\alpha = 2/3$. This value could also change with time and temperature due to re-shaping of solid A during reaction with gas B as well as sintering of solid A under high temperatures. However, due to simplifications a constant value of $\alpha = 2/3$ is assumed. Now, the s_i notation from Equation 2.37 ends up being

$$s_i = v_i * rate = v_i \bar{R} \tag{2.47}$$

where v_i is the stoichiometric coefficient of i. If we now wish to extend this equation by considering a system of more than one gas-solid reaction, a more general solution to Equation 2.47 would be

$$s_i = \sum_{j=1}^{NR} v_{i,j} \bar{R}_j \tag{2.48}$$

where NR is the number of reactions taking place and $v_{i,j}$ is a stoichiometric matrix corresponding of the different reaction stoichiometric coefficients. $\bar{R}_j$ represents the $rate$ of reaction j.

2.1.6 Exothermic and Endothermic Reactions

When a system is subject to chemical changes in form of chemical reactions, change in energy is consequently involved as well. The chemical reaction itself means breaking and formation of bonds which involves change in enthalpy either way. So, our reactions would either produce heat (exothermic) or consume heat (endothermic). The reaction heat source term in analogy with Equation 2.48 now becomes

$$\hat{s} = \sum_{j=1}^{NR} \triangle_r H_j \bar{R}_j \tag{2.49}$$

where $\triangle_r H_j$ is the *enthalpy of rection* of reaction j given by

$$\triangle_r H_j = \int_{T_0}^{T} \triangle C_p^{\ominus} dT + \triangle_r H_j^{\ominus} \tag{2.50}$$

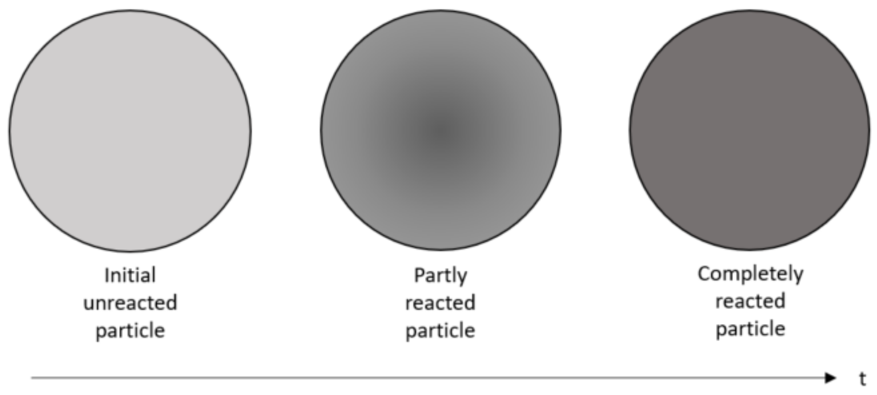


Figure 2.3: Representation of a reacting particle where solid reactant is converted continuously throughout the particle. The initial solid particle of only one component (light grey) reacting with a gas creating a solid product (which is represented as a darker grey), over time. In this case assumed to be an isochoric process.

2.2 Continuous Model (CM)

Now that we have been through the topics of gas film diffusion, particle diffusion, gas-solid reactions and thermal conductivity we can pull everything together making a theoretical model which describes the different concentration profiles of components present inside a particle at any given time as well as the overall temperature profile, see **Figure 2.3**. However, before doing this we need to make some assumptions. For instance, when it comes to estimating the porosity, the molecular weights of solid species, as well as their intrinsic densities (reactants and products) needed in Equation 2.22 are now considered to represent one specie only. In this case the initial particle solid specie. We then avoid having a porosity factor that is time dependent, due to concentration changes inside the particle of reactants and products of different porosities. This assumption was used in [29] as well. We then get the simplified version

$$\epsilon = 1 - c_j * \frac{MW_{(kg),j}}{\rho_{I,j}} \tag{2.51}$$

where j is the initial particle component. If our initial particle is of many components, an estimated average could be used. Nevertheless, it is now possible to consider a constant porosity factor independent of time and position inside the particle. The second assumption is based on the fact that we only operate with reactions that are equimolar with respect to the gas phase. We may then remove the last term on the r.h.s. of Equations 2.2 and 2.31 for all considered gas species. Now, when estimating the concentration profiles of both gas and solid components inside the sphere shaped particle we must use the full mass balance for an open system

$$accumulation = in - out + generation - consumption \tag{2.52}$$

which works over the system volume (volume of the particle). Now, if we consider the accumulation of gas specie i inside the solid particle which over time may consist of different solid components (NS), combining Equations 2.30, 2.31 and 2.48 with the assumption of time/position independent ϵ, Equation 2.52 becomes

$$\epsilon\frac{\partial c_i}{\partial t} = D_{i,eff}\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial c_i}{\partial r}\right) + \sum_{j=1}^{NR} \upsilon_{i,j} k_j (1-X_j)^{2/3} c_j \tag{2.53}$$

where the first term on the r.h.s. of the equation corresponds to the $in - out$ term and the second term corresponds to the $generation - consumption$ term. Note that c_j is not the concentration of specie j because j is not a notation of specie in this case but a reaction notation. So, c_j is the concentration of the reactant specie that in reaction j controls the *reaction rate*. X_j is the solid conversion from reaction j. Now, the same equation can be used for a solid specie i. However, the first term on the r.h.s. of Equation 2.53 is neglected due to the assumption of non existent diffusion of solid matter. We then get

$$\frac{\partial c_i}{\partial t} = \sum_{j=1}^{NR} \upsilon_{i,j} k_j (1-X_j)^{2/3} c_j \tag{2.54}$$

Let us now consider the reactions inside a particle of initial components A_1, A_2 and A_3 initiated by the gaseous reactant B

$$a_1A_1\,(s) + bB\,(g) \longrightarrow c_1C_1\,(s) + dD\,(g) \tag{2.55}$$

$$a_2A_2\,(s) + bB\,(g) \longrightarrow c_2C_2\,(s) + dD\,(g) \tag{2.56}$$

$$a_3A_3\,(s) + bB\,(g) \longrightarrow c_3C_3\,(s) + dD\,(g) \tag{2.57}$$

Note that all the reactions consumes as well as produces the same type of gas species. The stoichiometric matrix is now

$$v_{i,j} = \begin{bmatrix} -a_1 & -a_2 & -a_3 \\ -b & -b & -b \\ +c_1 & +c_2 & +c_3 \\ +d & +d & +d \end{bmatrix} \tag{2.58}$$

Equation 2.53 for specie B then becomes

$$\begin{aligned} \epsilon\frac{\partial c_B}{\partial t} = D_{B,eff}\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial c_B}{\partial r}\right) + v_{2,1}k_1(1-X_1)^{2/3}c_B \\ + v_{2,2}k_2(1-X_2)^{2/3}c_B + v_{2,3}k_3(1-X_3)^{2/3}c_B \end{aligned} \tag{2.59}$$

and for the solid specie A$_3$

$$\frac{\partial c_{A_3}}{\partial t} = 0 + 0 + v_{1,3}k_1(1-X_3)^{2/3}c_B \tag{2.60}$$

One should note that the equations presented in this section does not fully describe our system, hence whats beyond the particle radius, but takes for granted that values such as surface concentrations of gaseous reactants are known. However, this may not be the case in a real life scenario. Nevertheless, these values are possible to estimate considering what was presented in Section 2.1.1. We know that our mass balance must be fulfilled at the interface between the particle surface and the gas film. By assuming steady-state, from Equations 2.31 and 2.2 we get

$$-D_{i,eff}\frac{\partial c_i}{\partial r}\hat{r} = \bar{k}_i(c_{i,surface} - c_{i,bulk})\hat{r} \tag{2.61}$$

Now that we have equations which could be used to describe the concentration profiles inside the particle, we can consider a similar approach concerning the overall temperature profile based on Equation 2.52. By neglecting any convective terms related to heat transfer (no viscous dissipation) inside the particle and by consequence writing $dh_e = \bar{C}_{p,solid}dT$, where $\bar{C}_{p,solid}$ is the average *specific heat capacity* at constant pressure of the solids present in the particle. When using the *specific heat capacity* we also need to use a *specific enthalpy of reaction* ($h_{r,e}$) in order to be dimensionally consistent. Combining Equations 2.35, 2.36 and 2.49, gives us

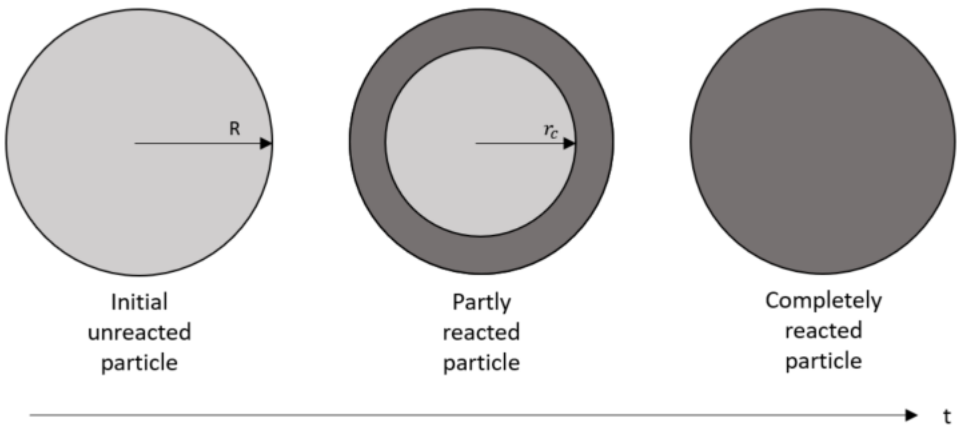


Figure 2.4: Representation of a reacting particle according to the shrinking-core model. The gas-solid reaction is first initiated at the particle surface. As the unreacted core shrinks, further reaction proceeds at the reaction surface between the "ash" layer (dark grey) and the unreacted core (light grey). The solid reactant is completely converted as the reaction front sweeps by.

$$\bar{\rho}\bar{C}_{p,solid}\frac{\partial T}{\partial t} = \hat{k}_{eff}\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial T}{\partial r}\right) + \sum_{j=1}^{NR}\left[h_{r,e}\bar{R}\right]_j \tag{2.62}$$

where $\bar{\rho}$ is the average solid density. On the particle surface we have

$$-\hat{k}_{eff}\frac{\partial T}{\partial r}\hat{r} = \bar{h}_c(T_s - T_b)\hat{r} \tag{2.63}$$

2.3 Shrinking-Core Model (SCM)

A more common and widely used model describing the process of gas-solid reactions is the so called shrinking-core model. This model might be chosen over its counterpart (CM) for a number of reasons. For instance, it is more straight forward and easier to use. SCM might also be a better representation of reality for certain gas-solid reactions were flaking ash or gaseous products cause shrinking in particle size (Levenspiel [12]). That being said, choosing a model for your system which establishes a good representation of reality without too many mathematical complexities should be prioritized either way.

SCM only considers one zone which the gas-solid reaction can take place which clearly differs from CM. Initially, this zone (or front) lies on the particle surface. It then moves inwards as the gas-solid reaction converts the solid reactant leaving behind a solid product or gas as seen in **Figure 2.4** (a solid in this case). This assumption implies that we only need a few equations describing what passes inwards or outwards at the reaction front as it moves by. If we consider a gas film as described in Section 2.1.1, the mass transfer equation must be added. As visualized by Yagi and Kunii (the developers of SCM, Levenspiel [12]), five steps were assumed occurring in succession during reaction, i.e. 2.1: Diffusion of gaseous reactant B through the gas film - Diffusion of B through the ash layer consisting of C - Reaction of B with solid A - Diffusion of gaseous product D through C - Diffusion of gaseous product D through the gas film. Note that some of these steps are not present in some situations. Nevertheless, these steps are directly connected to the overall resistance to reaction. For instance, if the product layer of solid D is of low porosity, diffusion of gaseous species might be affected increasing the resistance to reaction. However, these resistances often varies greatly and in some cases we may only consider the step with the highest resistance to be rate-controlling [12]. In this case we will present solutions to the three first steps as rate-controlling steps. These will later be combined to give the best possible depiction of reality considering SCM.

Gas Film Controls

Now that the resistance of the gas film controls, we can assume that there is no gaseous reactant B at the particle surface (from Reaction 2.1). We then end up with the concentration driving force, $c_{B,bulk} - c_{B,surface}$, which becomes $c_{B,bulk}$ due to $c_{B,surface} = c_{B,R} = 0$. Combining Equation 2.2 with Equation 2.43, we get

$$-\frac{1}{A_{ex}}\frac{dn_A}{dt} = -\frac{1}{A_{ex}}\frac{a}{b}\frac{dn_B}{dt} = a\bar{k}_B(c_{B,bulk} - c_{B,R}) = a\bar{k}_B c_{B,bulk} = constant \tag{2.64}$$

where A_{ex} is the exterior surface area of the particle with radius R. The amount of moles of specie i in the particle is $n_i = \phi_i V$, where V is the volume of the particle and ϕ_i is the molar density of specie i. Now, if we consider the general Reaction 2.1, we can write the disappearance of dn_A moles as

$$-dn_A = -\phi_A V = -\phi_A = -\phi_A d\left(\frac{4}{3}\pi r_c^3\right) = -4\pi\phi_A r_c^2 dr_c \tag{2.65}$$

where r_c is the radius of the unreacted core of solid A (see Figure 2.4). Implementing

Equation 2.65 into Equation 2.64, the change in unreacted core radius per unit time can be expressed as

$$-\frac{dr_c}{dt} = \frac{aR^2\bar{k}_B}{\phi_A r_c^2} c_{B,bulk} \tag{2.66}$$

Diffusion Through Ash Layer Controls

Now, let us consider the diffusion of gaseous reactant B (Reaction 2.1) through the present ash layer to be the rate-controlling step. We then have $c_{B,bulk} = c_{B,R}$ and $c_{B,r_c} = 0$ due to no resistance in the present gas film. By considering Equation 2.31 in the radial direction, assuming steady-state and *equimolar counter diffusion*, our rate of reaction of B at any instant is given by its rate of diffusion to the reaction surface, thus

$$-\frac{dn_B}{dt} = 4\pi r^2 D_{B,eff}\frac{dc_B}{dr} = constant \tag{2.67}$$

integrating across the ash layer, we get

$$-\frac{dn_B}{dt}\int_R^{r_c}\frac{dr}{r^2} = 4\pi D_{B,eff}\int_{[B]_{bulk}}^{[B]_{r_c}=0} dc_B \tag{2.68}$$

or

$$-\frac{dn_B}{dt}\left(\frac{1}{r_c}-\frac{1}{R}\right) = -\frac{b}{a}\frac{dn_A}{dt}\left(\frac{1}{r_c}-\frac{1}{R}\right) = 4\pi D_{B,eff}c_{B,bulk} \tag{2.69}$$

Implementing Equation 2.65 and we end up with

$$-\frac{dr_c}{dt} = \frac{aD_{B,eff}c_{B,bulk}}{b\phi_A r_c^2\left(\frac{1}{r_c}-\frac{1}{R}\right)} \tag{2.70}$$

Chemical Reaction Controls

When chemical reaction controls we have $c_{B,bulk} = c_{B,R} = c_{B,r_c}$ (Reaction 2.1). Since the presence of ash layer is not affecting the progress of the gas-solid reaction, the rate is proportional to the surface of the unreacted core, thus

$$-\frac{1}{a}\frac{dn_A}{dt} = A_{r_c}k' * c_{B,bulk} \tag{2.71}$$

Note that we are not considering a dependable factor of local structure and porosity ($f(X) = 1$). Implementing Equation 2.65, we end up with

$$-\frac{dr_c}{dt} = a\frac{k' * c_{B,bulk}}{\phi_A} \tag{2.72}$$

Combination of Resistances

The expressions presented above for SCM assumes that only a single resistance controls throughout reaction of the particle. However, this is not the case in reality were the relative importance of the gas film, ash layer, and reaction steps will vary as particle conversion progresses. As expressed in Levenspiel [12], to account for the simultaneous action of these resistances, we could combine Equations 2.66, 2.70 and 2.72 directly, thus

$$-\frac{dr_c}{dt} = \frac{a}{b}\frac{c_{B,bulk}/\phi_A}{\frac{r_c^2}{R^2\bar{k}_B} + \frac{r_c(R-r_c)}{RD_{B,eff}} + \frac{1}{bk'}} \tag{2.73}$$

Model Expansion

Implementation of different ash layers from a complex chain reaction is not implemented in this study. Nevertheless, this could be added by implementing more unreacted core radius functions for the different reaction layers, but keep in mind that more resistances must be considered for every gas-solid reaction located inside the ’onion’ layered particle. Implementing temperature dependence have not been considered either. This is due to the complicated nature considering temperature changes in a noncontinuous model. In fact, we could come up with a temperature dependent radius formula. However, this radius will not follow the unreacted core radius which indicates different values of heat and mass transfer. If this model was to be affected by temperature, position dependent temperatures must be "remembered" and source terms calculated in neighbouring radial positions to the reaction front. Nevertheless, this is beyond the scope of this study.

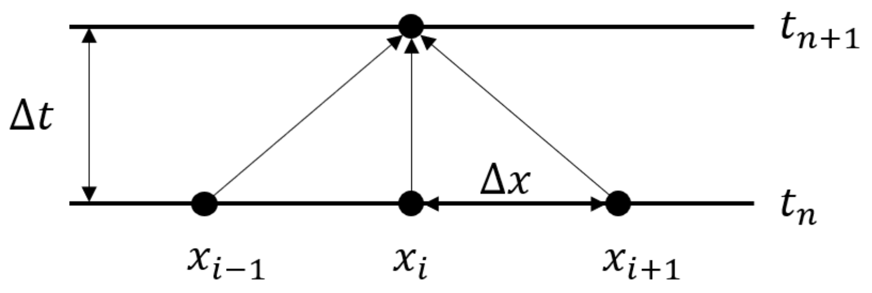


Figure 2.5: Schematic representation of the FTCS finite difference scheme, for the one dimensional diffusion Equation 2.74.

2.4 Numerical Treatment

In this section we will present two methods for estimating the diffusive term in Equation 2.53.

2.4.1 The FTCS Explicit Method

The Forward-Time Central-Space (FTCS) method is a finite difference method used for numerically solving partial differential equations. Now, let us consider the diffusion equation in one dimension given by

$$\frac{\partial c(x,t)}{\partial t} = D\frac{\partial^2 c(x,t)}{\partial x^2} \tag{2.74}$$

If we consider a grid of $x_i = i\triangle x$ and $t_n = n\triangle t$ (**Figure 2.5**), approximation of first-order derivatives could be estimated by the central difference

$$\left(\frac{\partial c}{\partial x}\right)_i \approx \frac{c_{i+1} - c_{i-1}}{2\triangle x} \tag{2.75}$$

the forward difference

$$\left(\frac{\partial c}{\partial x}\right)_i \approx \frac{c_{i+1} - c_i}{\triangle x} \tag{2.76}$$

and the backward difference

$$\left(\frac{\partial c}{\partial x}\right)_i \approx \frac{c_i - c_{i-1}}{\triangle x} \tag{2.77}$$

In turn, using Equation 2.75 with the in-between grid element boundary [-1/2,+1/2], second-order derivatives could be estimated by

$$\left(\frac{\partial^2 c}{\partial x^2}\right)_i \approx \frac{\left(\frac{\partial c}{\partial x}\right)_{i+1/2} - \left(\frac{\partial c}{\partial x}\right)_{i-1/2}}{\triangle x} = \frac{\frac{c_{i+1}-c_i}{\triangle x} - \frac{c_i - c_{i-1}}{\triangle x}}{\triangle x} = \frac{c_{i+1} - 2c_i + c_{i-1}}{(\triangle x)^2} \tag{2.78}$$

Now that we know how to estimate derivatives appropriate to a finite difference scheme, a similar implementation of the scheme in spherical coordinates considering the full mass balance (Equation 2.53) could be used. The diffusive term on the r.h.s. of Equation 2.53 is now given by the approximation

$$\begin{aligned} D_{i',eff}\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial c_{i'}}{\partial r}\right) &\approx D_{i',eff}\frac{1}{r_i^2}\frac{\left(r^2\frac{\partial c_{i'}}{\partial r}\right)^n_{i+1} - \left(r^2\frac{\partial c_{i'}}{\partial r}\right)^n_{i-1}}{2\triangle r} \\ &= D_{i',eff}\frac{1}{r_i^2}\left[\frac{r_{i+1}^2(c^n_{i',i+1} - c^n_{i',i}) - r_{i-1}^2(c^n_{i',i} - c^n_{i',i-1})}{2\triangle r^2}\right] \end{aligned} \tag{2.79}$$

where $c(i,n)$ is the position and time dependent function of concentration $c_{i'}$. Note the notation change from i to i' of the specified specie in Equation 2.79 due to i already being used as the position notation in the r-direction. The l.h.s. of Equation 2.53 is given by the approximation

$$\epsilon\frac{\partial c_{i'}}{\partial t} \approx \epsilon\frac{c^{n+1}_{i',i} - c^n_{i',i}}{\triangle t} \tag{2.80}$$

We then end up with

$$\epsilon\frac{c^{n+1}_{i',i} - c^n_{i',i}}{\triangle t} = D_{i',eff}\frac{1}{r_i^2}\left[\frac{r_{i+1}^2(c^n_{i',i+1} - c^n_{i',i}) - r_{i-1}^2(c^n_{i',i} - c^n_{i',i-1})}{2\triangle r^2}\right] + \sum_{j=1}^{NR} v_{i',j}k_j(1-X_j)^{2/3}c_j \tag{2.81}$$

for the gas species, and for the solid species we get

$$\epsilon\frac{c^{n+1}_{i',i} - c^n_{i',i}}{\triangle t} = \sum_{j=1}^{NR} v_{i',j}k_j(1-X_j)^{2/3}c_j \tag{2.82}$$

with the boundary conditions

$$r = 0 \qquad \frac{dc_{i'}}{dr} = 0 \qquad symmetric \tag{2.83}$$

$$r = R \qquad -D_{i',eff}\frac{c^n_{i',i} - c^n_{i',i-1}}{\triangle r} = \hat{k}_{i'}(c_{i',s} - c_{i',b}) \tag{2.84}$$

When considering numerical treatment of any problem by numerical algorithms for differential equations, rounding errors and/or small fluctuations in initial data sets might cause large deviations between final estimation outputs and exact solutions. The FTCS can then yield unstable solutions that oscillate and grow if $\triangle r$ is too large. The stability condition for the FTCS explicit method reads (Ames [31])

$$\frac{1}{2} \geq \alpha \frac{\triangle t}{(\triangle r)^2} \tag{2.85}$$

where $\alpha = D_{i',eff}/\epsilon$. At first glance the stability condition might look manageable. However, a doubling of spatial resolution in the r-direction requires a simultaneous reduction in the time-step by a factor of four in order to maintain numerical stability which might suddenly become unmanageable considering long reduction times.

When it comes to estimating the temperature profile inside the particle, a similar approach is used when approximating Equation 2.62. This is not shown in the report due to redundancy. However, the equation is given as

$$\bar{\rho}\bar{C}_{p,solid}\frac{T^{n+1}_i - T^n_i}{\triangle t} = \hat{k}_{eff}\frac{1}{r^2_i}\left[\frac{r^2_{i+1}(T^n_{i+1} - T^n_i) - r^2_{i-1}(T^n_i - T^n_{i-1})}{2\triangle r^2}\right] + \sum_{j=1}^{NR}\left[h_{r,e}\bar{R}\right]_j \tag{2.86}$$

with the boundary conditions

$$r = 0 \qquad \frac{dT}{dr} = 0 \qquad symmetric \tag{2.87}$$

$$r = R \qquad -\hat{k}_{eff}\frac{T^n_i - T^n_{i-1}}{\triangle r} = \bar{h}_c(T_s - T_b) \tag{2.88}$$

2.4.2 Orthogonal Collocation Method

An alternative to approximating the diffusive term in Equation 2.53 with a more manageable numerical stability, is by using the orthogonal collocation method (OC) which was first developed by Villadsen and Stewart (as cited in [32]), and is in summary a special case of the collocation method and the method of weighted residuals. First of all we want to express our differential equations at specified points, x_j. As we do not know the exact solution to every differential equation we could express them by using corresponding trial functions representing the specified points, x_j. Now, a sum of different functions describing the exact solution to an unknown function could be expressed as (Heinemann and Carberry [32])

$$trial\ function = c^*(x) = \sum_{i=1}^{N} a_i c_i(x) \tag{2.89}$$

where N is the number of interior collocation points, and a_i are unknown coefficients. This trial function is substituted into the differential equations, and their results at every collocation point are the so called residuals. We now want to determine the unknown coefficients, a_i, which satisfies the differential equations the most at every collocation point. The residual is required to be zero which again determines the unknown coefficients a_i. If the number of collocation points goes to infinity ($N = \infty$), the approximated solution would converge to the exact solution. In problems that are symmetric such as our spherical particle, it can be shown that the solution is a function of x^2 rather than just x. Our trial function then becomes

$$c^*(x^2) = b + (1 - x^2) \sum_{i=1}^{N} a_i P_{i-1}(x^2) \tag{2.90}$$

where P_i is an orthogonal polynomial, hence the name orthogonal collocation method, which improves the rate of convergence as N increases. Note that the interior collocation points are the roots to $P_N(x^2) = 0$ which are tabulated values for different N. So, the orthogonal collocation method is applied by substituting the trial function (Equation 2.90) into the differential equation at hand, which is set to zero at every collocation point x_j. Then, the derivatives of the trial function at the specified collocation points can be found by reducing the residuals in terms of a_i, and hence $c^*(x_j^2)$. This is done by taking the first derivative of the trial function and the Laplacian of this expression and evaluate them at every collocation point. These can be rewritten in matrix notation (see Finlayson [33]). We then have for every N collocation point pluss the $N+1$ collocation point where $x = 1$

$$\left(\frac{dc}{dx}\right)_j = \sum_{i=1}^{N+1} A_{j,i} \bar{c}_i \tag{2.91}$$

$$\left(\frac{d^2c}{dx^2}\right)_j = \sum_{i=1}^{N+1} B_{j,i}\bar{c}_i \tag{2.92}$$

where A and B matrices of dimension $(N+1)\times(N+1)$ represents the Laplacian at the collocation point j, $\bar{c}$ is now a vector of length $N+1$ where every element represents the function at their specified collocation point. Now, lets say we have a general gas-solid reaction system with its differential equation describing the change in concentration of gas specie i' due to diffusion inside a spherical particle given by

$$\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{dc_{i'}}{dr}\right) = \frac{d^2c_{i'}}{dr^2} + \frac{2}{r}\frac{dc_{i'}}{dr} = \phi^2 f(c_{i'}(r)) \tag{2.93}$$

with the boundary conditions

$$r = 0 \qquad \frac{dc_{i'}}{dr} = 0 \qquad symmetric \tag{2.94}$$

$$r = R \qquad -D_{i',eff}\frac{dc_{i'}}{dr} = \hat{k}_{i'}(c_{i',surface} - c_{i',bulk}) \tag{2.95}$$

Implementing Equation 2.91 for our boundary condition $r = R$, we get

$$-D_{i',eff}\sum_{i=1}^{N+1} A_{N+1,i}\bar{c}_{i',i} = \hat{k}_{i'}(c_{i',N+1} - c_{i',bulk}) \tag{2.96}$$

where $\bar{c}_{i',i}$ is a vector containing collocation point dependent concentrations of gas specie i'. Taking out the $N+1$ term in the sum, we get

$$-D_{i',eff}A_{N+1,N+1}c_{i',N+1} - D_{i',eff}\sum_{i=1}^{N} A_{N+1,i}\bar{c}_{i',i} = \hat{k}_{i'}(c_{i',N+1} - c_{i',bulk}) \tag{2.97}$$

which becomes

$$c_{i',N+1} = -\frac{D_{i',eff}\sum_{i=1}^{N} A_{N+1,i}\bar{c}_{i',i} - \hat{k}_{i'}c_{i',bulk}}{D_{i',eff}A_{N+1,N+1} + \hat{k}_{i'}} \tag{2.98}$$

Now, considering Equation 2.93 and the full mass balance 2.53, we get

$$\epsilon\frac{\partial c_{i',j}}{\partial t} = D_{i',eff}\left[\sum_{i=1}^{N+1} B_{j,i}\bar{c}_{i',i} + \frac{2}{r_j}\sum_{i=1}^{N+1} A_{j,i}\bar{c}_{i',i}\right] + \sum_{n=1}^{NR} v_{i',n}k_n(1-X_n)^{2/3}c_n \tag{2.99}$$

where j is now the collocation point we are "standing in" and n notates the specified gas-solid reactions at this position j. For present solid species, Equation 2.54 now becomes

$$\frac{\partial c_{i',j}}{\partial t} = \sum_{n=1}^{NR} v_{i',n} \left[k_n (1 - X_n)^{2/3} c_n\right]_j \tag{2.100}$$

which holds for every collocation point j. The collocation constants used for spherical symmetry are taken from Villadsen and Stewart [34]. These collocation constants can however not be implemented directly into our equations stated above. For instance, they are given on the dimensionless form. If we wish to use these matrices, we need to convert them to our specified dimension from Equation 2.99. Our matrices then becomes

$$A_{j,i} = \frac{1}{R} \hat{A}_{j,i} \tag{2.101}$$

$$B_{j,i} = \frac{1}{R^2} \hat{B}_{j,i} \tag{2.102}$$

where $\hat{A}_{j,i}$ and $\hat{B}_{j,i}$ are taken from [34]. Now, only one more adjustment is needed. The $\hat{B}_{j,i}$ matrix from Villadsen and Stewart [34] contains the whole Laplace-operator which gives our initial B matrix the same property, thus

$$\frac{1}{r^2} \frac{d}{dr} \left(r^2 \frac{dc_{i'}}{dr} \right) = \sum_{i=1}^{N+1} B_{j,i} \bar{c}_{i',i} \tag{2.103}$$

Equation 2.99 then becomes

$$\epsilon \frac{\partial c_{i',j}}{\partial t} = D_{i',eff} \sum_{i=1}^{N+1} B_{j,i} \bar{c}_{i',i} + \sum_{n=1}^{NR} v_{i',n} k_n (1 - X_n)^{2/3} c_n \tag{2.104}$$

Now that we can estimate the concentration profiles, a similar approach could be taken regarding the temperature profile. However, this is not shown in the report due to redundancy. Nevertheless, the equation is given as

$$\bar{\rho} \bar{C}_{p,solid} \frac{\partial T_j}{\partial t} = \hat{k}_{eff} \sum_{i=1}^{N+1} B_{j,i} \bar{T}_i + \sum_{n=1}^{NR} \left[h_{r,e} \bar{R}\right]_n \tag{2.105}$$

with the $N+1$ term

$$T_{N+1} = -\frac{\hat{k}_{eff} \sum_{i=1}^{N} A_{N+1,i} \bar{T}_i - \bar{h}_c T_b}{\hat{k}_{eff} A_{N+1,N+1} + \bar{h}_c} \tag{2.106}$$

Chapter 3

Modeling The Prereduction Unit

In this chapter, we would like to present our general approach to modeling the prereduction unit. Then we would like to implement two functions, one for estimating the *mass transfer coefficients* and one for estimating the *heat transfer coefficient*. The third part is devoted to presenting the different models described in Chapter 2 and compare them using a simple example. The fourth part is devoted fully to the main objective of the report. The complete code describing the prereduction unit can be found in Appendix E. Programming languages used will be Python and MATLAB®.

☞ **Remark**: We have scaled up the letter-spacing in all our represented scripts and functions for improved readability. If one wish to recreate any results from this report one should take note of any line-skips (e.g., between line 16 and 17, Script E.1) when copy-pasting scripts/functions from the appendix.

3.1 Approach

Before beginning any programming related task, a simplified model overview is needed to structure the problem. This can be seen in **Figure 3.1** where our expected input parameters goes into our idealized model producing the wanted outputs.

Our task now lies in producing a model which can use these input parameters to predict the specified output. We should then start by summarizing our most important assumptions. First of, particles described in our model are considered to be stationary spheres surrounded by a moving gas. Our process is both considered an isochoric pro-

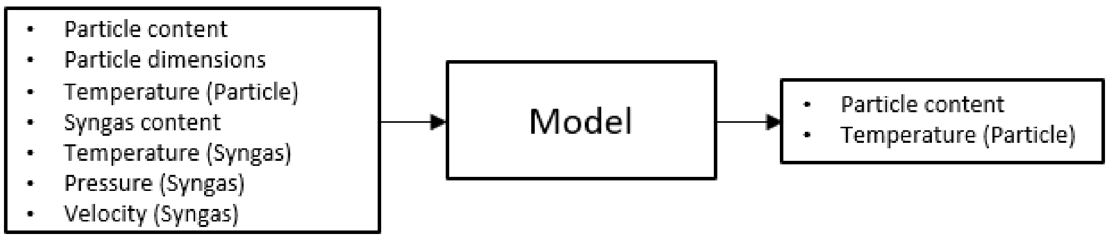


Figure 3.1: Simplified overview of input and output parameters of our idealized model.

cess as well as an isobaric process. With that being said, we also assume that the work done on the system by the moving gas is negligible. We then end up with a system dependent on both external and internal mass/heat transfer together with internal heat generation and consumption/accumulation of solid species based on gas-solid reactions. We shall then start to implement every phenomena described in Section 1.2 if not otherwise neglected. Model performance is also considered an important factor due to the fact that our intention with the model described by the thesis's main objective is to make a fast and resilient model which can quickly produce outputs based on different input scenarios.

3.2 External Mass and Heat Transfer Functions

Now that we have a general structure of our idealized model we would like to start by implementing functions describing the external mass and heat transfer phenomena occurring between our particle and syngas. These functions can be found in Appendix A. Now, let us start by describing the external mass transfer function (Function A.1). Here, line 1-58 describes the parameters needed for calculations while line 59-181 calculates the different *mass transfer coefficients* based on input i which indicates the specie of interest. As an example, let us begin with the calculation of $\bar{k}_{CO}$, first we need to calculate the overall *dynamic viscosity* of our specified gas mixture. Before doing this we need the different *dynamic viscosities* for every single specie present in the gas which can be calculated using Equation 2.14 represented by

```
14 %Dynamic viscosity (single specie):
15 muCO = 0.01720*((0.5555*518.67 + 118)/(0.5555*TOR +...
16     118))*(TOR/518.67)^(3/2); %cP
17
18 muCO2 = 0.01480*((0.5555*527.67 + 240)/(0.5555*TOR +...
    240))*(T0R/527.67)^(3/2); %cP

muH2 = 0.00876*((0.5555*528.93 + 72)/(0.5555*T0R +...
    72))*(T0R/528.93)^(3/2); %cP

muH2O = 0.0657; %cP (350C)
```

Here, *dynamic viscosity* for water vapour is considered constant. Calculation of the *dynamic viscosity* for the specified gas mixture using Equation 2.13 is represented by

```
%Dynamic viscosity for the CO-CO2-H2_H2O mixture:
mumix = (XCO*muCO*MWCO^0.5 + XCO2*muCO2*MWCO2^0.5 + XH2*muH2*MWH2^0.5 + ...
    XH2O*muH2O*MWH2O^0.5)/(XCO*MWCO^0.5 + XCO2*MWCO2^0.5 + XH2*MWH2^0.5...
    + XH2O*MWH2O^0.5);

mumixSI = 0.001*mumix; %N s/m2
```

The *kinematic viscosity* is then calculated using Equation 2.12, represented by

```
%Density of gas [rho = (MW*P)/(RT)]:
MWtot = (XCO*MWCO + XCO2*MWCO2 + XH2*MWH2 + XH2O*MWH2O)*0.001; %kg/mol

rhogas = (MWtot*PSTPPa)/(R*T0K); %kg/m3

%Kinematic viscosity:
nudot = mumixSI/rhogas; %m2/s
```

We would also need the diffusion coefficient of the binary pair, CO-mixture, which can be calculated using Equations 2.9, 2.10 and 2.11 represented by

```
    MWCOCO2 = 2*((1/MWCO)+(1/MWCO2))^(-1);
    MWCOH2 = 2*((1/MWCO)+(1/MWH2))^(-1);
    MWCOH2O = 2*((1/MWCO)+(1/MWH2O))^(-1);

    DCOCO2 = ((0.00143*(T0K^1.75))/(PSTPbar*(MWCOCO2^0.5)*(VCO^(1/3)+...
        VCO2^(1/3))^(2)))*0.0001; %m2/s
    DCOH2 = ((0.00143*(T0K^1.75))/(PSTPbar*(MWCOH2^0.5)*(VCO^(1/3)+...
        VH2^(1/3))^(2)))*0.0001; %m2/s
    DCOH2O = ((0.00143*(T0K^1.75))/(PSTPbar*(MWCOH2O^0.5)*(VCO^(1/3)+...
        VH2O^(1/3))^(2)))*0.0001; %m2/s

    DCOmix = (1 - XCO)/(XCO2/DCOCO2 + XH2/DCOH2 + XH2O/DCOH2O);
```

The *mass transfer coefficient* is then calculated using Equations 2.3, 2.4, 2.5 and 2.8 represented by

```
    %Schmidt number:
    Sc = nudot/DCOmix;

    %Reynolds number:
    Re = (vb*L)/nudot;

    %Sherwood number:
    Sh = 2.0 + 0.6*(Re^0.5)*(Sc^(1/3));

    %Mass transfer coefficient:
    kmassCO = (Sh*DCOmix)/L; %m/s
```

Now, let us consider a syngas with volume fractions CO = 0.8, $CO_2 = 0.1$, $H_2 = 0.06$ and $H_2O = 0.04$ of temperature 1300 kelvin, with a bulk velocity of 0.1 meters per second, hitting a particle of dimensions 0.002 meters in diameter. When calling the external mass transfer function for specie $i = 1$ $(1 = CO)$, we get

```
>> masstransferfun(1,1300,0.002,0.1,0.8,0.1,0.06)
ans =
    0.5262
```

where the output is the *mass transfer coefficient* of specie CO, $\bar{k}_{CO}$, between the particle surface and the gas stream.

Now that we have introduced the external mass transfer function we can do a similar presentation of the external heat transfer function (Function A.2). These functions share many similarities, however, considering Function A.2 we need to calculate both the *thermal conductivity* and the *heat capacity* for the gas mixture to be able to calculate the *heat transfer coefficient*. These can be calculated using Equations 2.19 and 2.20 represented by

```
khatmix = (XCO*khatCO + XCO2*khatCO2 + XH2*khatH2 + XH2O*khatH2O); %J/m/K/s

cpmix = (XCO*cpCO + XCO2*cpCO2 + XH2*cpH2 + XH2O*cpH2O); %J/kg/K
```

The *heat transfer coefficient* is then calculated using Equations 2.16, 2.4, 2.17 and 2.18 represented by

```
%Prantl number:
Pr = (nudot*cpmix*rhogas)/khatmix;

%Reynolds number:
Re = (vb*L)/nudot;
```

```

%Nusselt number:
Nu = 2.0 + 0.6*(Re^0.5)*(Pr^(1/3));

%Heat transfer coefficient:
heattran = (Nu*khatmix)/L; %W/m2/K
```

When calling the external heat transfer function (using the same input values) we get

```
>> heattransferfun(1300,0.002,0.1,0.8,0.1,0.06)
ans =
   154.4552
```

where the output is the *heat transfer coefficient*, $\bar{h}_c$, between the particle surface and the gas stream.

3.3 The Model (an Example)

With the implemented functions calculating the different *mass transfer coefficients* as well as the *heat transfer coefficient*, it is time to implement the phenomena occurring inside the particle which are internal mass/heat transfer, heat generation, and gas-solid reactions. From Chapter 2, we have two models to choose from, CM or SCM, were the CM can be solved by either using the FTCS method or the orthogonal collocation method. However, we will begin by implementing a simple example using SCM. Note that this model does not take the heat related phenomena into account. Nevertheless, this is an obvious starting point considering the simple nature of the SCM. To make things easier for ourselves, we shall start by implementing a simple example with only one gas-solid reaction. We can then compare the different models based on their different outputs and estimated performances.

3.3.1 SCM

Now, let us consider a spherical particle of zinc sulfide reacting with a gas stream containing oxygen given by the reaction

$$2\,\mathrm{ZnS}\,(s) + 3\,\mathrm{O_2}(g) = 2\,\mathrm{ZnO}\,(s) + 2\,\mathrm{SO_2}(g) \qquad \Delta H^\circ_{298} = -878.26\ \mathrm{kJ} \qquad (3.1)$$

This example is chosen due to being highly exothermic which will emphasis the difference of having a temperature dependent model vs a non temperature dependent model later on. Let us also consider the gas stream to have a temperature of 900°C containing 80% oxygen and 20% nitrogen at 1 atm with bulk velocity of 5 meters per second hitting our solid particle with a radius of 0.001 meters. The diffusion coefficient of the binary pair, O_2-N_2, were calculated by hand using Equations 2.9 and 2.10. The *mass transfer coefficient* were calculated using our example related Script A.3. Since this model is not temperate dependent we do not need to calculate the *heat transfer coefficient*. We then end up with our model input parameters given in Script B.1, line 8-13. With this, our main calculations can be initiated. The porosity factor of the solid is calculated using Equation 2.22, represented by

```
30 #porosity:
31 epsilon = 1 - (phi_ZnS*((MW_ZnS*0.001)/rho_ZnS))
```

The tortuosity of the solid is not considered, and is set to 1. After calculating the *effective diffusion coefficient* in line 37 (using Equation 2.28) with our previously calculated porosity factor, our model specifies both the *pre-exponential constant* and the *activation energy* of the reaction needed for Equation 2.42, represented by

```
42 #needed for the rate constant calculation:
43 k_0_r1 = 23000.00 #m/s (pre-exponential constant for the reaction)
44 E_a_r1 = 121032.00 #J/mol (activation energy for the reaction)
45 
46 k = k_0_r1*np.exp((-E_a_r1)/(R*T0_K)) #m/s
```

However, these are not the correct values for Reaction 3.1, but actually obtained from Berg [7] using linear interpolation similar to Schanche [15] concerning Reaction 1.3. Nevertheless, it is not our goal to be 100 % correct concerning Reaction 3.1 with this example, but do a comparison between CM and SCM (as long as we use the same values for each model). Now, when calculating the new position of the reaction front, r_c, we use Equation 2.73 represented by

```
64 for i in range(t):
65     R_rx = (1/(k*b))
66     R_film = ((r_c**2)/(r_0**2*kg))
67     R_ash = ((r_c*(r_0-r_c))/(r_0*D_eff))
68     drc_dt = -(((a*c_O2_b)/(phi_ZnS*b))/(R_rx + R_film + R_ash))
69     r_c = r_c + drc_dt*dt
70     if r_c < 0:
        r_c = 0
```

which updates r_c for every iteration in range t. Now, by calculating the initial particle concentration of ZnS together with the iteration dependent concentration we can produce a dimensionless conversion profile of ZnO, represented by

```
    #conversion profile:
    n_ZnS_SCM = ((4/3)*(np.pi)*r_c**3)*phi_ZnS
    n_ZnS_SCM_0 = ((4/3)*(np.pi)*r_0**3)*phi_ZnS
    c_ZnS_SCM = n_ZnS_SCM/(((4/3)*(np.pi)*r_0**3))
    c_ZnS_SCM_0 = n_ZnS_SCM_0/(((4/3)*(np.pi)*r_0**3))

    X_ZnS[i+1] = c_ZnS_SCM/c_ZnS_SCM_0
    X_ZnO[i+1] = 1 - X_ZnS[i+1]
    X_time[i+1] = X_time[i] + 1
```

Table 3.1: Example inputs without temperature dependency (*calculated from Script A.3).

$T_b[°C]$	$p_{O_2}[atm]$	$D_{O_2,N_2}[m^2/s]$	$\bar{k}_{O_2}[m/s]$	$R[m]$	$t[s]$
900	0.8	0.000228	0.71*	0.001	250

Running Script B.1 with the specified inputs given in **Table 3.1**, outputs the corresponding conversion profile of ZnO which is the time dependent fraction of ZnO (X_{ZnO}) plotted over the specified reduction time in seconds, see **Figure 3.2**. Performance-wise, we ended up with an estimated execution time of 0.087 seconds which all things considered is pretty good.

3.3.2 CM-FTCS

Now that we have implemented and described the SCM, it is time to implement the CM using the FTCS method. However, before going into a more detailed description of the CM script in Appendix C, one should note that this script was not designed with a high performance oriented goal set. It got huge performance issues related to memory leak. This problem is a direct consequence of the stability condition related to FTCS, Equation 2.85, which forces our time step to be extremely big in order to establish a small enough $\triangle t$ with our specified $\triangle r$, represented in Script C.1 by

```
#FTCS resolution:
steps_time = t*1000000
delta_t = t/steps_time
```

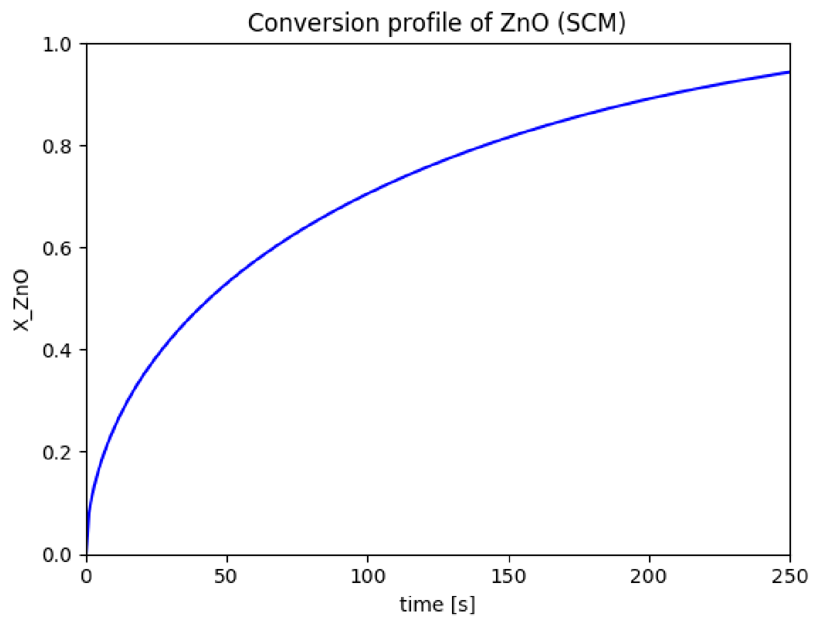


Figure 3.2: Conversion profile of ZnO at reduction time, $t = 250$, input values: **Table 3.1**, (SCM).

```
19
20 steps_radius = 10
21 delta_r = r_0/steps_radius
```

We then end up with a huge number of needed iterations. However, there exist many performance oriented alternatives which solves this problem to some degree. Nevertheless, we will later see how this problem can be avoided altogether with the orthogonal collocation method. So, the biggest difference between SCM and CM first arrive in line 95 (Script C.1). First of all, we are updating every time dependent variable (concentrations) during every iteration in our *for* loop (line 95-158). The needed position dependent variable, *reaction rate*, are calculated in line 97-107 represented by

```
97      """
98      Reaction and diffusion related calculations:
99      """
100     #rate calculations:
101     for i in range(len(k)):
102         k[i] = k_0_r1*np.exp((-E_a_r1)/(R_g*T[i]))
103     R = np.zeros(len(c_ZnS))
104     for i in range(len(c_ZnS)):
105         X = (1-(c_ZnS[i]/c_ZnS_0[i]))
106         f_X = (1-X)**(2/3)
107         R[i] = k[i]*f_X*c_O2[i]
```

Note that the calculated *effective diffusion coefficients* (line 109-113) are not position dependent in this case due to the fact that this model does not take heat transfer into account. However, this will later be implemented in our more sophisticated script using the orthogonal collocation method. With Script C.1, we want to make a comparison between SCM and CM considering only mass transfer and chemical reactions. This is due to the fact that SCM is limited when it comes to implementing temperature dependencies. Now, concentration related calculations for every point inside the particle specified by our FTCS resolution can be found in line 115-143. Here we need to convert our *rate* to be per volume instead of per surface area. This is due to our initial *pre-exponential constant* having units of meter per second (*superficial rate constant*). The conversion can be done using Equation 3.2.

$$\bar{R}' = \frac{V}{A}\bar{R} = a_0\bar{R} \tag{3.2}$$

The boundary related calculations using Equations 2.84 and 2.82 are then represented by

```
115     """
116     Concentration calculations:
117     """
118     #boundary concentration of O2 at r=r0:
119     c_O2[len(c_O2)-1] = ((kg*c_O2_b + c_O2[len(c_O2)-2]*
120         (D_eff[len(c_O2)-1]/delta_r))/(D_eff[len(c_O2)-1]/delta_r + kg))
121 
122     #boundary concentration of ZnS at r=r0:
123     a0 = ((4/3)*(np.pi)*r[len(R)-1]**3)/((4)*(np.pi)*r[len(R)-1]**2) #m
124 
125     if (a*delta_t*(1/a0)*R[len(R)-1]) >= c_ZnS[len(c_ZnS)-1]:
126         c_ZnS[len(c_ZnS)-1] = 0
127     else:
128         c_ZnS[len(c_ZnS)-1] = ((-a*delta_t*(1/a0)*R[len(R)-1]) +
129              c_ZnS[len(c_ZnS)-1])
```

The internal related calculations using Equations 2.81 and 2.82 are represented by

```
131     #concentration profile (solid and gas) calculations inside particle:
132     for n in range(len(c_ZnS)-2):
133         a0 = ((4/3)*(np.pi)*r[n+1]**3)/((4)*(np.pi)*r[n+1]**2) #m
134 
135         c_O2[n+1] = ((delta_t/epsilon)*((D_eff[n+1]/r[n+1]**2)*((((c_O2[n
+2]-
            c_O2[n+1])*r[n+2]**2)-((c_O2[n+1]-c_O2[n])*r[n]**2))/
            (2*delta_r**2))- b*(1/a0)*R[n+1]) + c_O2[n+1])
        c_O2[0] = c_O2[1]
        if (a*delta_t*(1/a0)*R[n+1]) >= c_ZnS[n+1]:
            c_ZnS[n+1] = 0
        else:
            c_ZnS[n+1] = (-a*delta_t*(1/a0)*R[n+1]) + c_ZnS[n+1]
        c_ZnS[0] = c_ZnS[1]
```

Now that the different concentrations are calculated for every iteration at every specified point given by our FTCS resolution, we could output concentration profiles for each of our species present inside the particle at any given time. For instance, lets output the concentration profiles of ZnS and O_2 inside the particles using the already specified inputs given in **Table** 3.1, only changing the reduction time to, $t = 20$. These can be seen in **Figure** 3.3.

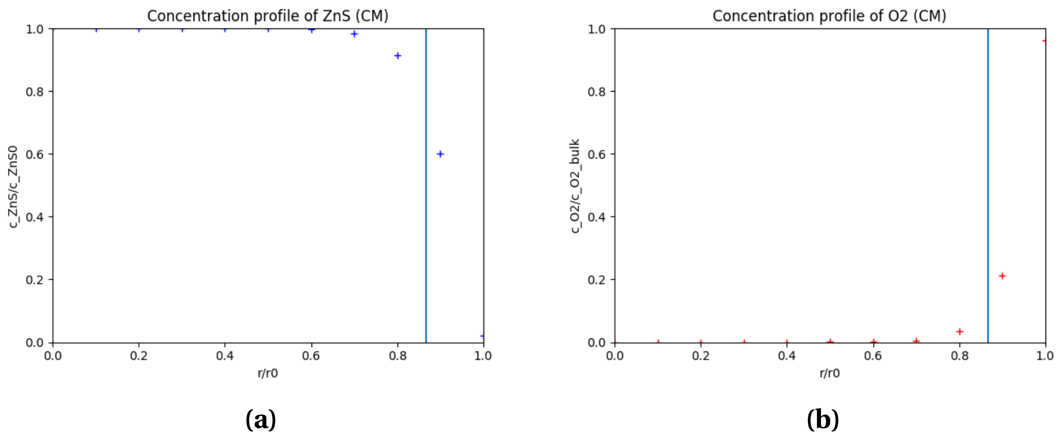


Figure 3.3: Concentration profiles of ZnS and O_2 at reduction time, $t = 20$ (CM-FTCS). The blue line represent the reaction front r_c, calculated from Script B.1 (SCM) using the same reduction time input. All values are in the range [0,1] where 1 indicates no conversion of solid species (a), and total gas saturation (b).

The conversion profile of ZnO is calculated by integrating over the ZnS profile (**Figure** 3.3a) for each iteration followed by converting the total particle concentration of ZnS to a fraction ($c_{\text{ZnS}}/c_{0,\text{ZnS}}$) which is then subtracted from the initial particle fraction of ZnS, $X^*_{\text{ZnS}} = 1$. This gives us the solid fraction of ZnO (X_{ZnO}) inside the particle represented by line 145-158 (Script C.1) which can be plotted over the specified reduction time t. The conversion profile of ZnO with reduction time, $t = 20$, can be seen in **Figure** 3.4. Performance-wise, we ended up with an estimated execution time of 4330.6

seconds (1.2 hours) which is pretty bad considering the objective of making a fast and resilient model. The performance gets even worse when increasing the time input due to the static time step implementation. This step could, however, be updated during iterations dependent on the stability condition. In Script C.1, the scaling value of 10^6 used for scaling the time input to the needed amount of time steps represented in line 17 were found by using Equation 2.85 and is set to be a constant value but could be updated as mentioned above. That being said, it is advised to use a supercomputer when executing Script C.1 for t above 20 seconds.

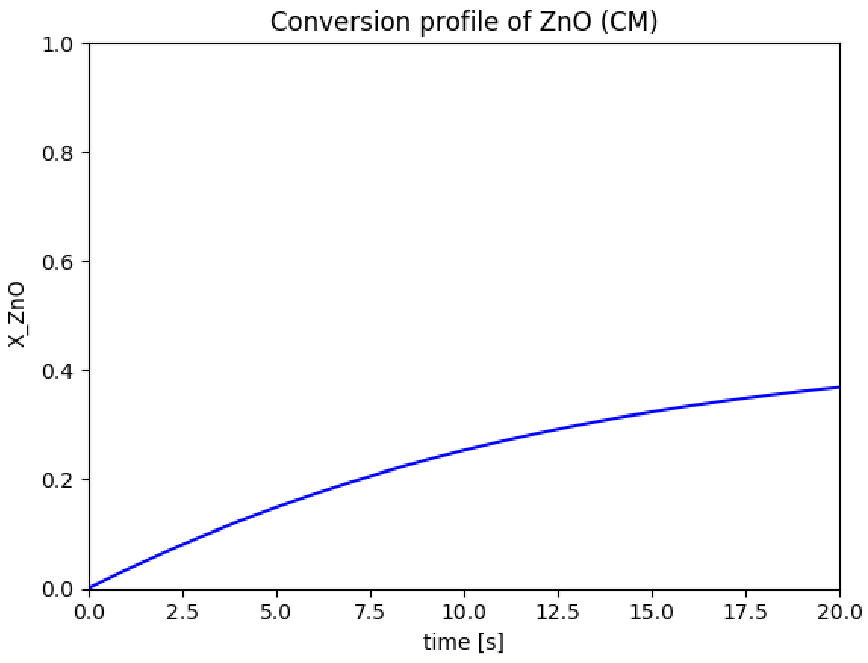


Figure 3.4: Conversion profile of ZnO at reduction time, $t = 20$, (CM-FTCS).

When running the model with the input values given in **Table 3.1**, the estimated execution time were 19.18 hours. The output can be seen in **Figure 3.5**. Comparing this output with the SCM output (see **Figure 3.7**), they look quite similar. However, the SCM estimates the particle fraction of ZnO (X_{ZnO}) at the specified time to be 0.943 while the CM-FTCS method estimates a value of 0.903 which indicates a faster conversion rate estimated from SCM. Nevertheless, considering the fact that we get more information about the particle from the CM related outputs, the huge loss in performance makes us still favor the faster, yet simpler SCM. Even though we could easily implement heat transfer as well as heat generation into our continuous model we need to go back to the drawing board. Luckily, by implementing the orthogonal collocation method instead of the FTCS method, we should expect a huge performance increase.

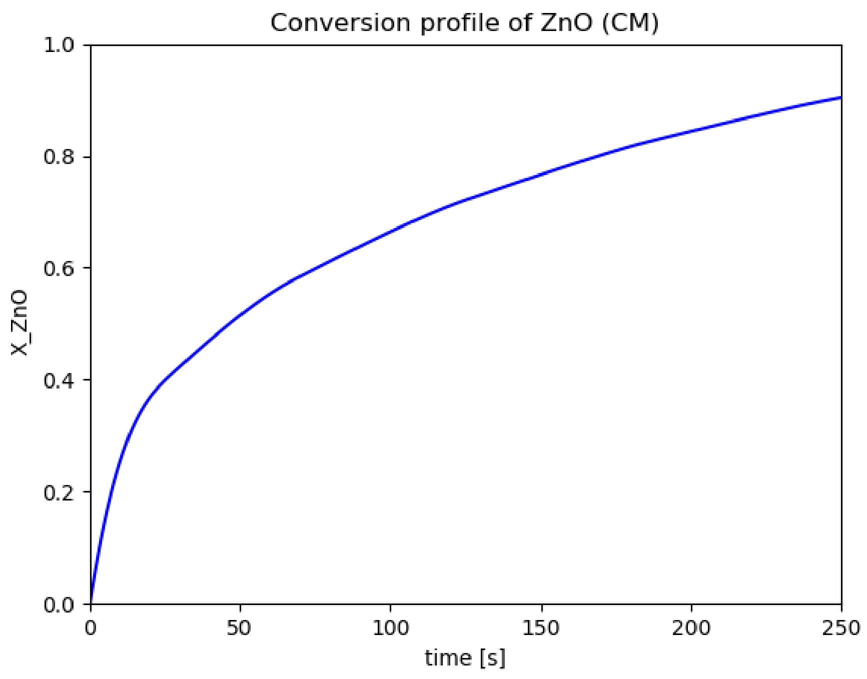


Figure 3.5: Conversion profile of ZnO at reduction time, $t = 250$, input values: **Table 3.1**, (CM-FTCS).

3.3.3 CM-OC

The CM script using the OC method (Appendix D) depends on the ode15s function in Matlab. This function is capable of integrating complex systems of differential equations by only specifying initial conditions (Shampine and Reichelt [35]). Note that Script D.1 does take heat transfer as well as heat generation into account. We then need to calculate both the *mass transfer coefficient* as well as the *heat transfer coefficient*, using Script A.3 and A.4. However, we will start by only considering mass transfer and chemical reactions similar to our implementation of the CM-FTCS method to be able to compare the outputs. Then both heat transfer and heat generation will be implemented to see the effect of a temperature dependent model. Now that we want to implement the OC method to solve our continuous model, we first need to specify the different matrices given in Equations 2.101 and 2.102 together with our collocation points inside the particle. As mentioned earlier, both matrices and collocation points on their dimensionless form are taken form Villadsen and Stewart [34]. These are implemented into Script D.1 by converting them into their correct dimensions using Equations 2.101

and 2.102 as well as converting our collocation points by multiplying with the specified input particle radius, represented by

```
rc   = rp*[0.36311746 0.67718628 0.89975799]; %Vector containing inner
    collocation points

Amat = (1/rp)*[-4.1308947 6.8819128 -4.5475385 1.7965206;
        -1.3388630 -2.2150478 5.2890548 -1.7351440;
        0.62570332 -3.7406161 -1.6671150 4.7820278;
        -1.0727282 5.3255695 -20.752841 33/2]; %First derivative
Bmat = (1/rp.^2)*[-23.853065 30.593651 -9.74629544 3.0057072;
        11.099906 -43.237662 40.818768 -8.6810122;
        -3.3228457 38.356814 -125.40927 90.375305;
        -33.675598 152.37521 -311.19961 385/2]; %Laplacian
```

Now, there is only one reaction present in our example (Reaction 3.1) which in this case, the stoichiometric matrix becomes

```
vmat = [-2 0 0;
        -3 0 0;
         2 0 0;
         2 0 0]; %Stoichiometric coefficient matrix
```

Note that the stoichiometric matrix can be expanded upon to hold as many reactions as needed. One should also note that this matrix is not present in the previous presented scrips as each stoichiometric coefficient were implemented directly. This is a direct consequence of planning ahead. In the end, we want to expand on the OC model to be able to tackle more complex problems, in this case, reduction of manganese oxides. Hence the implemented stoichiometric matrix. Now, for the ode15s to work we need to specify our initial conditions. These are represented by

```
%Initial conditions:
Tp0 =  ones(length(rc),1)*T0;
cg10 = zeros(length(rc),1); %O2
cs10 = ones(length(rc)+1,1)*41300; %ZnS

y0 = [Tp0(:);cg10(:);cs10(:)];
```

Ode15s is then called upon, represented by

```
%Solver:
[t,y] = ode15s(@(t,y) model(t,y,cs10,rp,hg,kg1,Tb,Vfracg1,rc,...
    Amat,Bmat,vmat), [0 tend], y0);
```

where the model function in line 39 (Function D.2) describes the system of differential equations $y' = f(t, y)$. This function takes in the time span (t), the specified system related functions (y), initial particle concentration and dimensions ($cs10 = c_{0,\mathrm{ZnS}}$, $rp = R$), gas related constants specified by the input in Script D.1, all the orthogonal collocation related constants ($rc, Amat, Bmat$), and lastly, the stoichiometric matrix ($vmat$) which returns a vector with all the time derivatives related to y, represented by

```
105 dydt = [dTdt(:);dcg1dt(:);dcs1dt(:)];
```

Now, in Function D.2, input vector y, is first organized by

```
 9 %Input to correct vectors:
10 T = y(1:3);
11 cg1 = y(4:6);
12 cs1 = y(7:9);
13 cs1surf = y(10);
```

However, input values concerning the present solid species are set to zero if their input values are negative represented by

```
3 for i = 1:4
4     if y(i+6) <= 0
5         y(i+6) = 0;
6     end
7 end
```

This is due to the fact that we need to force the model to stop consuming solid reactants when they are not present. Now, after specifying all the different variables needed for future calculations in line 15-35, the *enthalpy of reaction* is calculated by using Equation 2.50, represented by

```
41 %Calculates the enthalpy of reaction:
42 deltacprx1 = vmat(1)*cpZnS + vmat(2)*cpO2 + vmat(3)*cpZnO + vmat(4)*cpSO2;
43 
44 deltaHrx1s = -878260; %J/mol
45 
46 deltaHrx1T = zeros(length(rc),1);
47 for i = 1:3
48     deltaHrx1T(i,1) = (T(i)*deltacprx1 - 298.15*deltacprx1) + deltaHrx1s;
49 end
```

Due to the fact that this model share many similarities with the already presented models, we shall not explain every line in Function D.2. However, one should understand that both surface and internal calculations are treated separately with different equations in both the FTCS and OC related models. For instance, in Function D.2, all the surface oriented calculations depend on the surface boundary condition $r = R$ and its mass balance, which utilizes Equation 2.98 together with Equation 2.100 for both the gas/solid concentration profiles. The surface boundary related temperature is calculated using Equation 2.106. We then end up with

```
%Surface:
Tsurf = (hg*Tb-keff*(Amat(4,1:3)*T(1:3)))/(keff*Amat(4,4)+hg);
%Tsurf = T(3);

Deffsurf = (epsilonZnS/tauZnS)*(((0.00143*(Tsurf^1.75))/(PSTPbar*...
    (MWO2N2^0.5)*(VO2^(1/3)+VN2^(1/3))^(2)))*0.0001);
cg1surf = (kg1*cg1b - Deffsurf*(Amat(4,1:3)*cg1(1:3)))/...
    (Deffsurf*Amat(4,4)+kg1);

ksurf = k0r1*exp((-Ear1)/(Rg*Tsurf));
fXsurf = (1-(1-(cs1surf/cs10(4))))^(2/3);
Rsurf = ksurf*fXsurf*cg1surf;
dcs1dtsurf = vmat(1)*(1/a0surf)*Rsurf;
```

Now, for our internal calculations, each of the specified collocation points has its own set of equations similar to the FTCS method. However, in this case we want to use Equation 2.104 for describing the concentration change of the specified gas species and Equation 2.100 for describing the consumption/production of solid species. Temperature change is described by using Equation 2.105, we then end up with

```
%Internal:
dTdt = zeros(length(rc),1);
for i = 1:3
   dTdt(i,1) = 1/(phiZnS*cpZnS)*(keff*(Bmat(i,1:3) * T(1:3) + ...
       Bmat(i,4)*Tsurf) + -deltaHrx1T(i)*(1/a0)*R(i));
   %dTdt(i,1) = 0;
end

dcg1dt = zeros(length(rc),1);
for i = 1:3
   dcg1dt(i,1) = (1/epsilonZnS)*(Deff(i)*(Bmat(i,1:3) * cg1(1:3) + ...
       Bmat(i,4)*cg1surf) + vmat(2)*(1/a0)*R(i));
end

```

```
99  dcs1dtint = zeros(length(rc),1);
100 for i = 1:3
101     dcs1dtint(i,1) = vmat(1)*(1/a0)*R(i);
102 end
```

Now, when running Script D.1 with the inputs given in **Table 3.1** and turning off any temperature related calculations in Function D.2 (re-activate line 73 and 90), we can output the different concentration profiles as well as the conversion profile of ZnO similar to what has already been done. Note that the concentration profiles calculated by the OC method are limited in the sense that they only have three internal points with the fourth point representing the surface. We shall then use a special integration method from Finlayson [33]. This method utilizes a weighted vector, **W**, represented in Function D.2 by [34]

```
73 W = [0.04567809, 0.12589839, 0.13397916, 1/36];
```

to establish the ZnO fraction using

$$X_i(t) = \frac{\mathbf{y}_i(t)\cdot\mathbf{W}}{\mathbf{y}_i(1)\cdot\mathbf{W}} \tag{3.3}$$

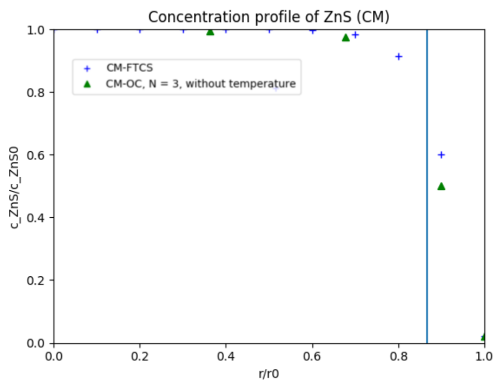
where $\mathbf{y}_i$ is a vector containing the concentration profile of specie i, represented by

```
74 XZnO = zeros(length(y),1);
75 for i = 1:length(y)
76     cZnStot = dot(y(i,7:10),W);
77     XZnS = cZnStot/(dot(y(1,7:10),W));
78     XZnO(i) = 1 - XZnS;
79 end
```

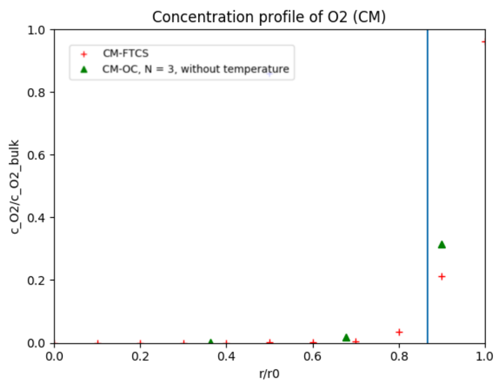
The different concentration profiles compared with each other can be found in **Figure 3.6**.

The different conversion profiles for each model can be found in **Figure 3.7**. Performance-wise, the OC model ended up with an estimated execution time of 0.826 seconds which is a huge performance increase compared to the previously tested model using the FTCS method. That being said, SCM is still faster (0.087 seconds), however, by being slightly slower, the OC method brings a lot more to the table with its concentration profiles and as will be shown in the next paragraph, implemented temperature dependency.

We have up to this point explained the temperature related phenomena in our OC



(a)



(b)

Figure 3.6: Concentration profiles of ZnS and O_2 at reduction time, $t = 20$ (FTCS and OC). The blue line represent the reaction front r_c, calculated from Script B.1 (SCM) using the same reduction time input.

model but never considered it when doing all the calculations in Function D.2. Now, by deactivating line 73 and 90 (default settings in Function D.2) we suddenly have a model which is temperature dependent. Our initial inputs are then updated to take heat transfer into account, see **Table 3.2**.

Table 3.2: Example inputs for the temperature dependent OC model (*calculated from Script A.3, **calculated from Script A.4).

$T_p[K]$	$T_b[K]$	$p_{O_2}[atm]$	$D_{O_2,N_2}[m^2/s]$	$\bar{k}_{O_2}[m/s]$	$\bar{h}_c[W/(m^2K)]$	$R[m]$	$t[s]$
298.15	1173.15	0.8	0.000228	0.71*	128.65**	0.001	250

The temperature dependent model output can be seen in **Figure 3.8**. As expected, the rate of conversion is much faster when taking heat transfer and heat generation into account due to Reaction 3.1 being highly exothermic. Now, looking at the conversion profiles related to the OC method, we see that they have sharp bends. This is due to the fact that we have fewer points concerning our OC related concentration profile of ZnO to integrate over which becomes critical over time when the outermost points are equal to zero representing full conversion. However, with more collocation points this can be avoided.

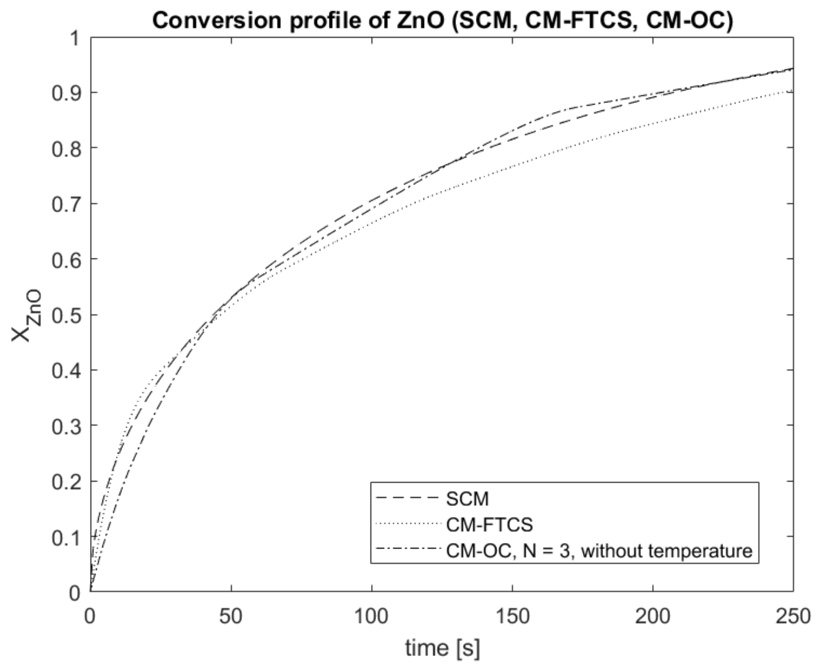


Figure 3.7: Conversion profile of ZnO at reduction time, $t = 250$, input values: **Table 3.1**, (SCM, FTCS and OC).

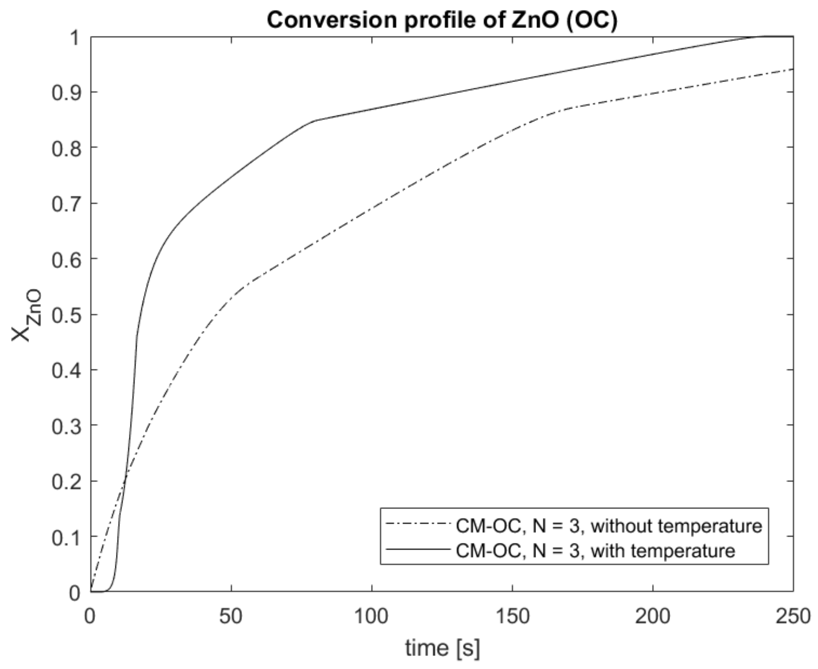


Figure 3.8: Conversion profile of ZnO at reduction time, $t = 250$, input values: **Table 3.2**, (OC with and without temperature).

3.4 The Prereduction Model

Now that we have presented the different models and experienced how they differ using an example case we should now begin to think about expanding one of them including all the reactions given in Section 1.1. We then have to keep track of every specie present inside, as well as outside our theoretical particle related to reduction of higher manganese oxides (MnO_2, Mn_2O_3 and Mn_3O_4) to MnO with blends of CO, H_2, CO_2 and H_2O. We have proven to ourselves that our more sophisticated OC model is up to the task by being very fast and resilient. However, we will not explain every detail regarding the OC model again, but rather emphasize on the main changes going from a simple four-component gas-solid reaction system to an eight-component gas-solid reaction system. Now, we can simply start expanding our example model by updating the initial conditions, $y0$, represented in Script E.1 by

```
y0 = [Tp0(:);cCO0(:);cMnO20(:);cMn2O30(:);cMn3O40(:);cMnO0(:);cH20(:);cCO20
    (:);cH2O0(:)];
```

Now, considering all the reactions given in Section 1.1, our stoichiometric matrix becomes

```
vmat = [-1.0  -0.5  -1/3  -1;
        -0.5  -1/6  -1/3  -1;
         0.5  1/3   1.0  1;
         0.5  1/6   1/3  1]; %Stoichiometric coefficient matrix
```

Note that the newly formed stoichiometric coefficient matrix is used for describing both the CO and H_2 related reactions due to having similar stoichiometric coefficients during their reaction paths. The WGSR is simply added in the last column. We also need to calculate the different *mass transfer coefficients*, $\bar{k}_i$, for every specie i, together with the *heat transfer coefficient*, $\bar{h}_c$. This is done by calling Function A.1 and Function A.2 represented in Script E.1 by

```
kgCO  = masstransferfun(1,Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);
kgH2  = masstransferfun(2,Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);
kgCO2  = masstransferfun(3,Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);
kgH2O  = masstransferfun(4,Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);

hg = heattransferfun(Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);
```

Now, when ode15s is called upon, represented by

```
64 %Solver:
65 [t,y] = ode15s(@(t,y) modelcomp(t,y,rp,hg,kgCO,kgH2,kgCO2,kgH2O,Tb,VCOfrac
       ,...
66     VCO2frac,VH2frac,rc,Amat,Bmat,vmat), [0 tend], y0);
```

a more complex model function, Function E.2, is used for describing the expanded system of differential equations. This function goes through the same procedure as Function D.2 when calculating the different time derivatives representing each component in $y0$. However, the WGSR is now considered using Equations 1.10-1.15 represented in Function E.2 by

```
163 %WGSR:
164     %%{
165     Keq = exp((4577.8/Tp(i))-4.33);
166     kWGS = exp((-29364/(1.987*Tp(i)))+(40.32/1.987));
167     KCO = exp((3064/(1.987*Tp(i)))-(6.74/1.987));
168     KCO2 = exp((12542/(1.987*Tp(i)))-(18.45/1.987));
169     KH2O = exp((-6216/(1.987*Tp(i)))-(12.77/1.987));
170     PCO = cCO(i)*(Rg*Tp(i));
171     PCO2 = cCO2(i)*(Rg*Tp(i));
172     PH2O = cH2O(i)*(Rg*Tp(i));
173     PH2 = cH2(i)*(Rg*Tp(i));
174     RWGSR(i,1) = (((kWGS*KCO*KH2O*(PCO*PH2O-((PCO2*PH2)/Keq)))/(1 + ...
175         KCO*PCO + KH2O*PH2O + KCO2*PCO2)^2)*(phicat/60))*(10^6*((4/3)*pi*rp
    ^3)); %mol/s
176     %}
177     %RWGSR(i,1) = 0;
```

Note that we need to multiply by the volume to get the required units from line 175. When running Script E.1, one is free to change the default input variables represented by

```
4  %Input:
5  rp = 5e-3;          %Particle radius, m
6  Tb = 1600;          %Bulk temperature, K
7  %T0 = 300;          %Initial particle temperature, K
8  T0 = Tb;
9  vb = 0.1;           %Bulk velocity, m/s
10 tend = 20000;       %Ending time of simulation, s
11
12 VCOfrac = 0.99;  %Volume fraction of CO in bulk
13 VCO2frac = 0.01; %Volume fraction of CO2 in bulk
14 VH2frac = 0.0;   %Volume fraction of H2 in bulk
```

as well as the initial particle content represented by

```
53 cMn020 = ones(length(rc)+1,1)*57858;
54 cMn2030 = ones(length(rc)+1,1)*0;
55 cMn3040 = ones(length(rc)+1,1)*0;
56 cMn00 = ones(length(rc)+1,1)*0;
```

Here, our default initial particle consists only of MnO_2, this could, however, be changed as one see fit. Keep in mind that we sat our pure MnO_2 particle to have the molar density of, $\phi_{MnO_2} = 57858[mol/m^3]$ (line 54). So, when considering a starting point of only Mn_2O_3, we either need to assume that our particle have completely been reduced to Mn_2O_3 which implies that our initial molar density becomes, $\phi_{0,particle} = \phi_{MnO_2} * 0.5$, from Reaction 1.3, or that our starting particle has not been previously reduced which implies, $\phi_{0,particle} = \phi_{Mn_2O_3} = 28503[mol/m^3]$. However, when considering a different starting point other than $\phi_{0,particle} = \phi_{MnO_2}$, the mole balance has to be correct. Lines concerning the dimensionless plots of present oxides in Script E.1 (line 118-172) have to be changed accordingly as well.

Since there is no inert gas in our system we should consider to fill our particle with the specified syngas as a starting point so there is no vacuum inside our initial particle. During previous experiments (Berg [7] and Lobo [8]), an inert gas were used when preheating the material of interest before filling the chamber with the reductive gas. However, this is not implemented in our model which leaves us with

```
57 cCO0 = ones(length(rc),1)*cCOb;
58 cH20 = ones(length(rc),1)*cH2b;
59 cCO20 = ones(length(rc),1)*cCO2b;
60 cH2O0 = ones(length(rc),1)*cH2Ob;
```

Now, the porosity factor is taken directly out from the experiments conducted by Lobo [8] and not calculated using Equation 2.51. This is represented in Function E.2 by

```
81 %Porosity:
82 %epsilon = 1 - (phiMnO2*((MWMnO2*0.001)/rhoMnO2));
83 epsilon = 0.30; %Pellet/ = 0.05 lump (Lobo)
```

where we consider a porosity factor of 0.30 for pellets and 0.05 for lumps. When it comes to the different rate constants specified by Function E.2 in line 63-79, one is free to change every constant as one see fit. However, the default values are taken from Schanche [15] where both the *pre-exponential constant* and the *activation energy* of both the H_2 and the CO related reactions are assumed to be similar. This was assumed due to the fact that H_2 related values were hard to find without doing similar experiments as conducted by Berg [7].

When running Script E.1 with default input values we get the following outputs shown in **Figures 3.9-3.12**. **Figure 3.9** is self-explained and **Figure 3.10** shows the particle temperature at the different collocation points. However, some changes have been made to the presentation of the different conversion profiles. For instance, **Figure 3.11** does not show the total particle conversion profile of the different oxides but rather a presentation of the different oxide fractions at the different collocation points. That being said, due to our high porosity factor, ϵ, the particle will be reduced continuously at nearly the same rate at every collocation point, hence the curves being on top of each other. With a lower porosity factor this will not necessarily be the case. One should also note that each oxide in **Figure 3.11** are normalized on their own total concentration respectively and not the particle total concentration. So, if the particle consisted of only Mn_2O_3 after reduction, the second plot in **Figure 3.11** would show, $X_{Mn_2O_3} = 1$. Performance-wise, we ended up with an estimated execution time of 1.23 seconds which is pretty good considering all the differential equations being solved.

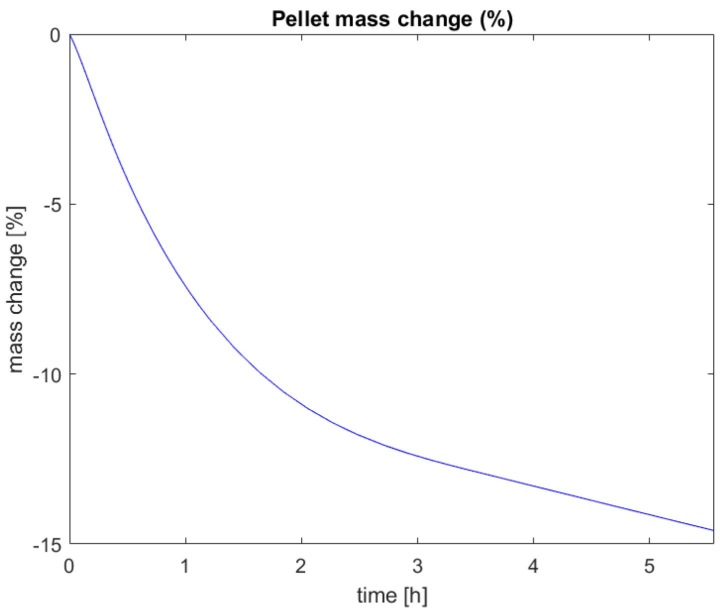


Figure 3.9: Mass change vs. time using default input values which concerns reduction of an MnO_2 particle using pure CO gas.

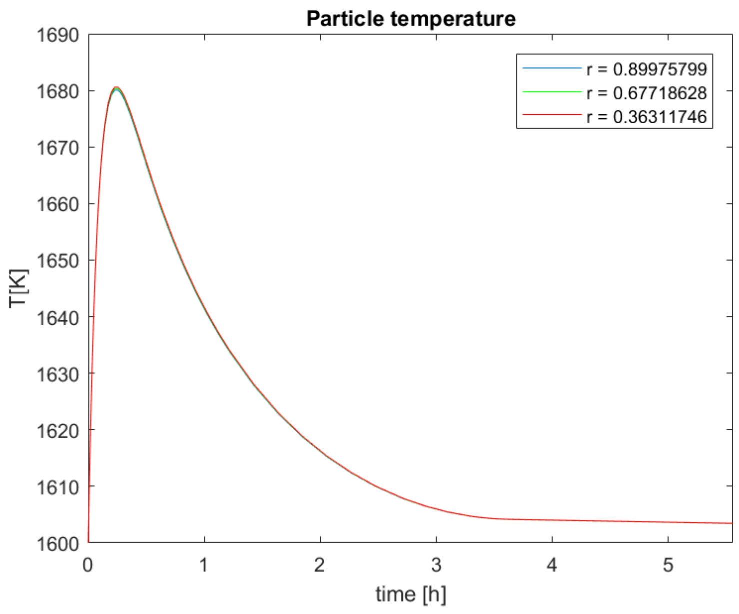


Figure 3.10: Particle temperature vs. time using default input values which concerns reduction of an MnO_2 particle using pure CO gas.

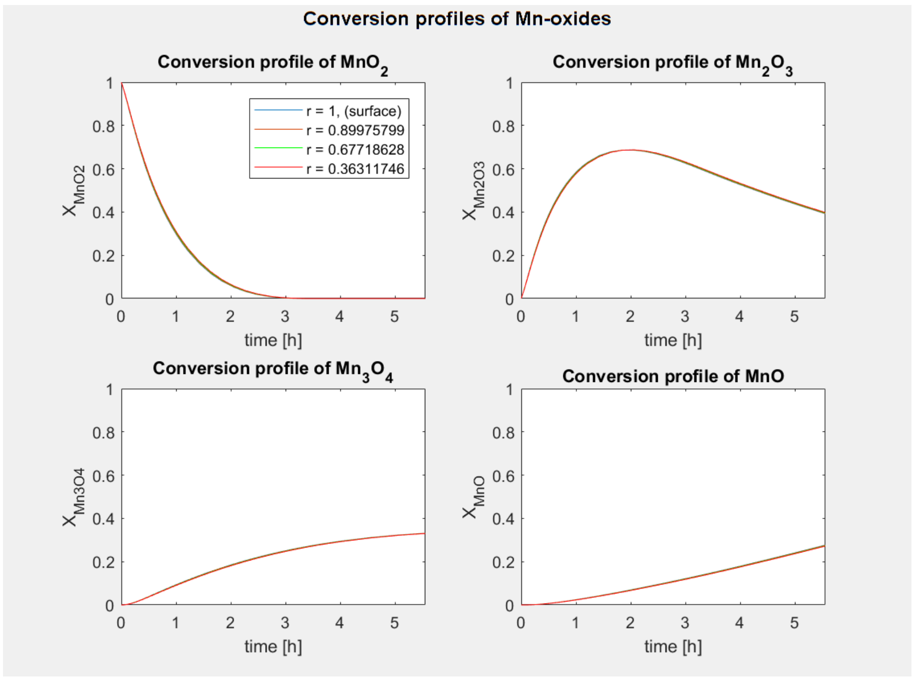


Figure 3.11: Different oxide fractions vs. time using default input values which concerns reduction of an MnO_2 particle using pure CO gas.

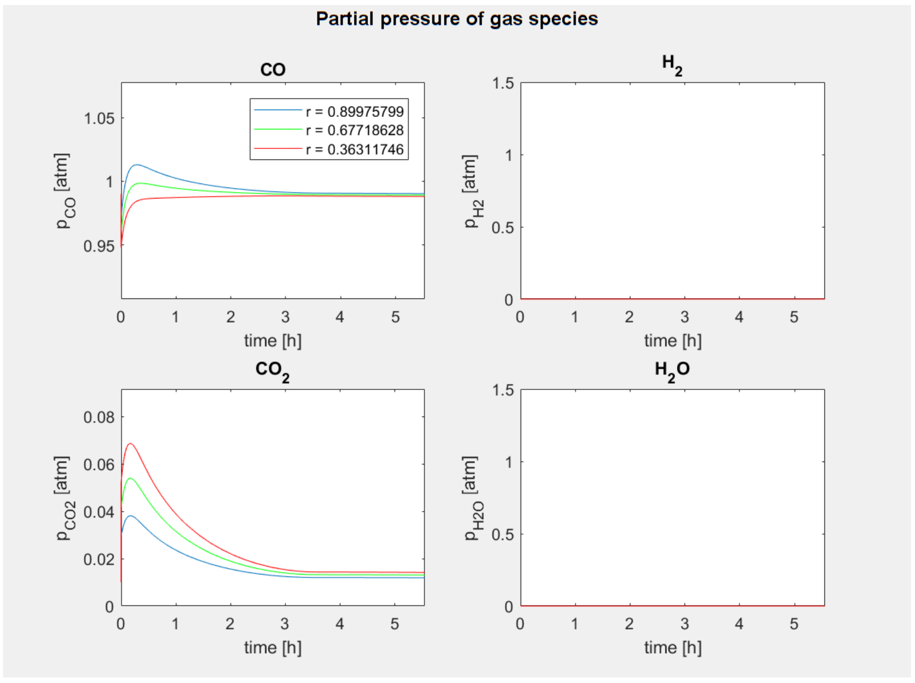


Figure 3.12: Partial pressure of different gas species vs. time using default input values which concerns reduction of an MnO_2 particle using pure CO gas.

Chapter 4

Results

In this chapter, we would like to present different cases using real experimental values taken from past experiments. Industrial data will be used as well to see how feasible it is to use off-gas from a manganese alloy producing furnace to fuel our prereduction unit. The industrial data were gathered from earlier work by the same authors [16], regarding furnace off-gas from Glencore Manganese Norway AS, now Ferroglobe Mangan Norge AS. Weightloss/mass change vs. time outputs from all the different cases will be plotted together at the end of each section to see how they differ. Conversion profiles and temperature profiles will be presented for some cases as well. A summary of all the input values for all the different cases presented in this chapter can be found in **Table 4.3**.

4.1 Particles of High Porosity with Pure Reducing Gases

First of all, before we begin to describe each case, note that the model exercise presented in this section serves as a presentation of the raw model, so no model adjustments have been made as a possible result of the model being adapted to experimental data. However, this will be considered in Section 4.2. Now, let us consider a particle consisting of only MnO_2 with a porosity factor of, $\epsilon = 0.3$. The dimension considered for our hypothetical sphere shaped particle will be of radius 5 mm. This is due to the fact that in the work conducted by both Berg [7] and Lobo [8], samples lay within the range of 10-14 mm. So why choose a particle radius of 5 mm? Now, the shapes of interest found in [7, 8] are those of irregular shapes (some cubic). For instance, these could be 10 mm in one direction and 5 mm in another. These shapes will then end up with a larger surface area where gas-solid reactions can occur. We will also end up with thinner samples in some cases which will reduce the diffusion controlled resistance. Our hypothetical particle is perfectly round with a smaller surface area and should roughly fall into the same ball park if considered a bit smaller than average size. That being said, one could also introduce a shape factor (sphericity [36]) which could be a value between 0 and 1. Now, for case one (C1), lets consider a syngas consisting of 99 % CO and 1 % CO_2 with a temperature of 1300 kelvin. It is assumed that our particle has been heated up to the same temperature with an inert gas assuming no decomposition of oxides. Our initial values then becomes

```
%Input:
rp = 5e-3;          %Particle radius, m
Tb = 1300;          %Bulk temperature, K
%T0 = 300;          %Initial particle temperature, K
T0 = Tb;
vb = 0.1;           %Bulk velocity, m/s
tend = 3600*2;      %Ending time of simulation, s

VCOfrac = 0.99;  %Volume fraction of CO in bulk
VCO2frac = 0.01; %Volume fraction of CO2 in bulk
VH2frac = 0.00;  %Volume fraction of H2 in bulk
```

Listing 4.1: Initial values for case one (C1)

The particle temperature at each collocation point can be seen in **Figure 4.1**. We clearly see the effect of having exothermic reactions, however, the temperature increase is not that significant. Nevertheless, our endless supply of syngas with constant temperature will of course have its impact on our model results. In reality we would have an finite volume which implies that our syngas around the particle should heat up as well. This has not been implemented into our model but we shall try to mimic its effect by turning off the heat transfer between the particle and the surrounding gas later on. Now, both the conversion profiles and partial pressures vs. time can be found in **Figure 4.2** and **Figure 4.3** respectively. The mass change vs. time plot can be seen in **Figure 4.10** (C1).

For case two (C2), we shall use the exact same input values as in case one, however, our syngas now consists of 99 % H_2 and 1 % H_2O. Our initial values then becomes

```
%Input:
rp = 5e-3;          %Particle radius, m
```

```
Tb = 1300;         %Bulk temperature, K
%T0 = 300;         %Initial particle temperature, K
T0 = Tb;
vb = 0.1;          %Bulk velocity, m/s
tend = 3600*2;     %Ending time of simulation, s

VCOfrac = 0.00;    %Volume fraction of CO in bulk
VCO2frac = 0.00;   %Volume fraction of CO2 in bulk
VH2frac = 0.99;    %Volume fraction of H2 in bulk
```

Listing 4.2: Initial values for case two (C2)

The particle temperature at each collocation point can be seen in **Figure** 4.4. Both the conversion profiles and partial pressures vs. time can be found in **Figure** 4.5 and **Figure** 4.6 respectively. The weightloss vs. time plot can be seen in **Figure** 4.10 (C2).

Now, let us assume that there is no heat transfer between the particle and the surrounding gas. This is accomplished by forcing our *heat transfer coefficient*, $\bar{h}_c$, to be zero by activating line 38 in Script E.1. We clearly see the exothermic effect in **Figure** 4.7 when no heat can escape the particle. Note that we have forced our model to stay at 2000 kelvin if reached for each of the collocation points (activating line 326-328 in Function E.2) due to model stability and prevention of reaching the melting point of MnO (2218 kelvin). Now, both the conversion profiles and partial pressures vs. time can be found in **Figure** 4.8 and **Figure** 4.9 respectively. The mass change vs. time plot can be seen in **Figure** 4.10 (C3).

Our next case, case four (C4), represents a similar approach only with inputs taken from **Listing** 4.2. The different outputs from case four is not presented here due to being very similar in nature to case three. However, the mass change vs. time plot can be seen in **Figure** 4.10. We clearly see that with no heat transfer we get model results similar to the experimental results seen in **Figure** 1.4.

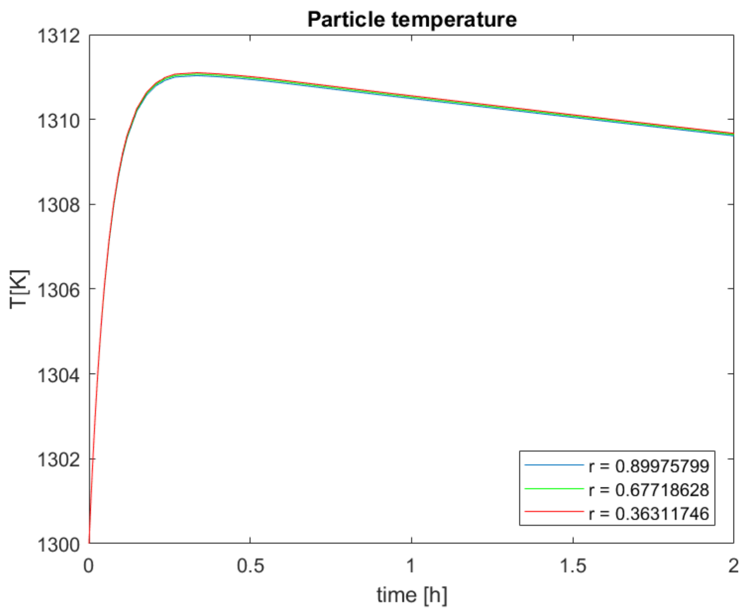


Figure 4.1: Particle temperature vs. time using input values found in **Listing 4.1** (C1) which concerns reduction of an MnO_2 particle using pure CO gas.

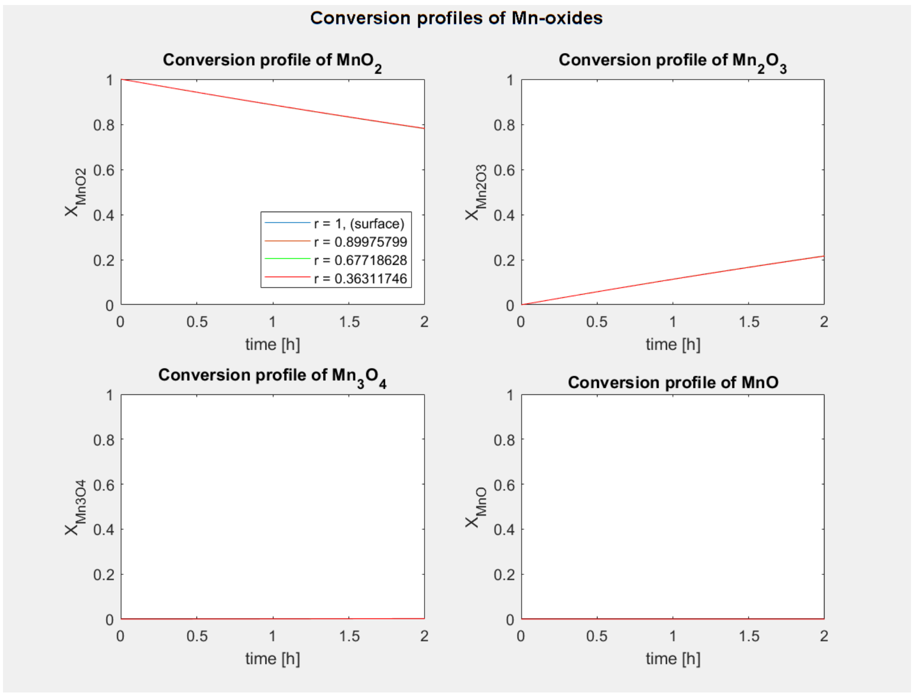


Figure 4.2: Different oxide fractions vs. time using input values found in **Listing 4.1** (C1) which concerns reduction of an MnO_2 particle using pure CO gas.

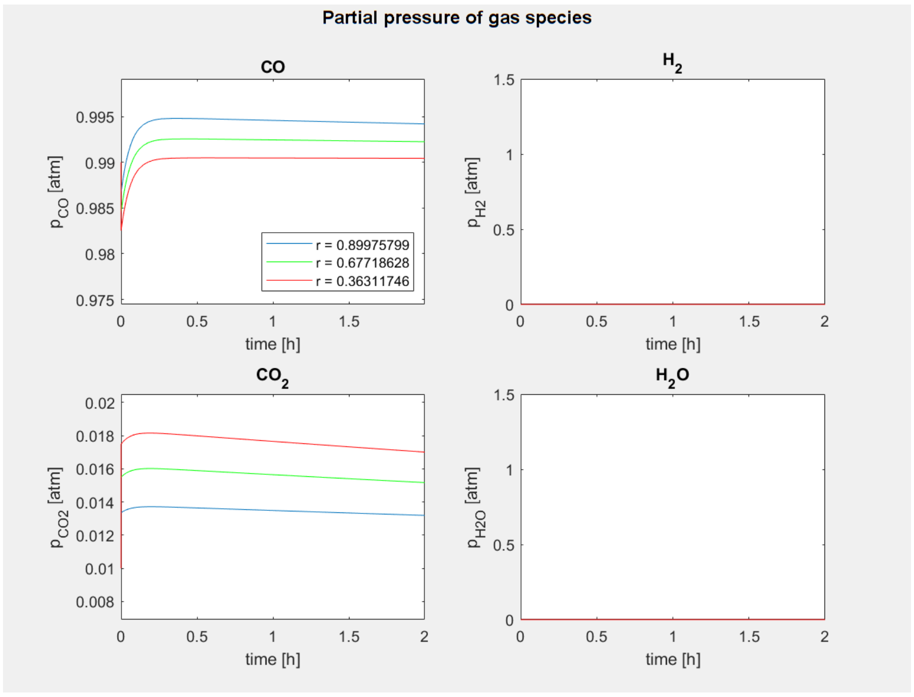


Figure 4.3: Partial pressure of different gas species vs. time using input values found in **Listing 4.1** (C1) which concerns reduction of an MnO_2 particle using pure CO gas.

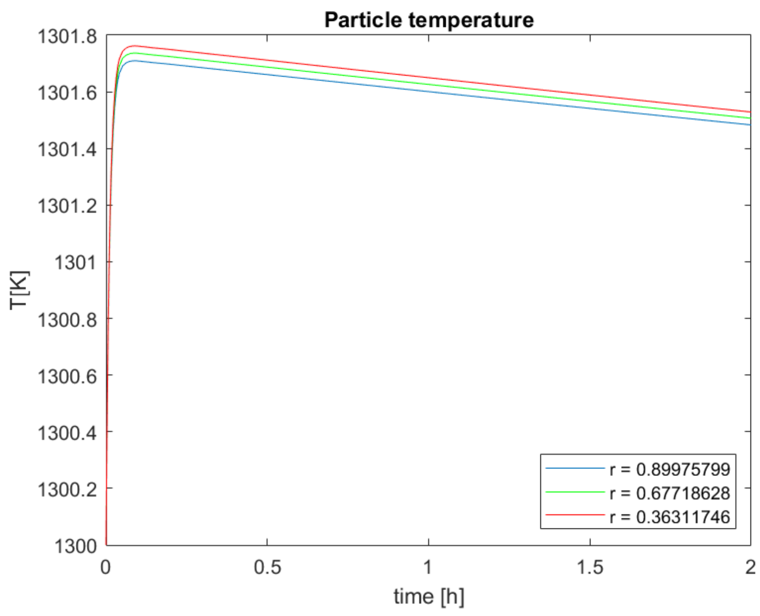


Figure 4.4: Particle temperature vs. time using input values found in **Listing 4.2** (C2) which concerns reduction of an MnO_2 particle using pure H_2 gas.

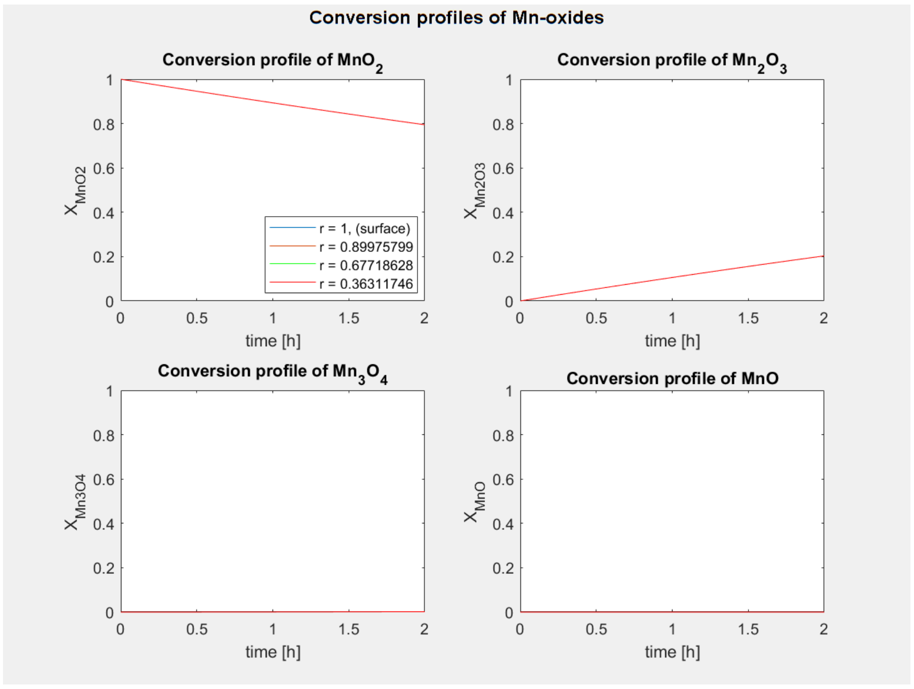


Figure 4.5: Different oxide fractions vs. time using input values found in **Listing 4.2** (C2) which concerns reduction of an MnO_2 particle using pure H_2 gas.

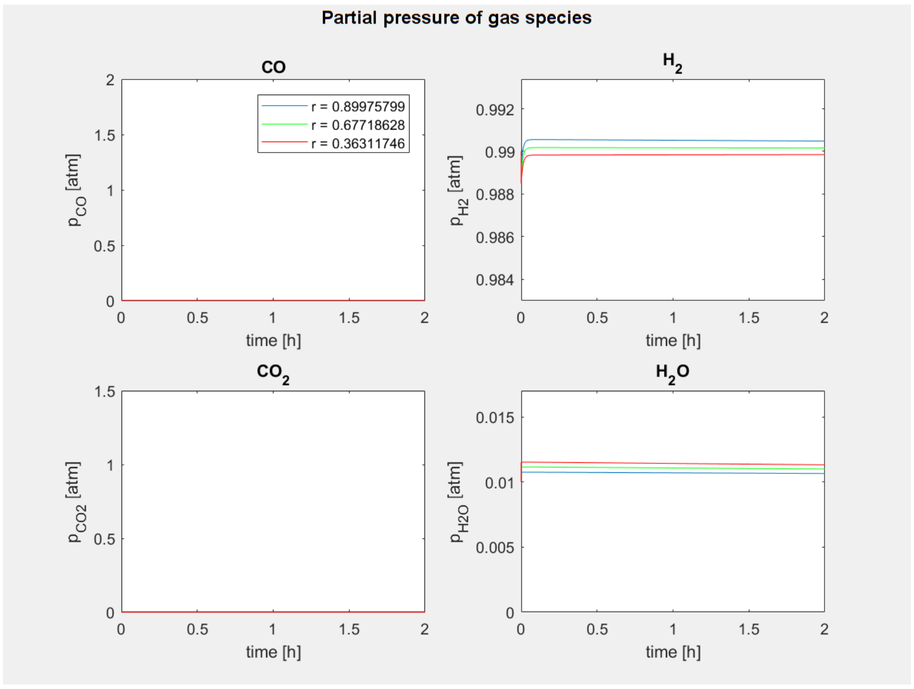


Figure 4.6: Partial pressure of different gas species vs. time using input values found in **Listing 4.2** (C2) which concerns reduction of an MnO_2 particle using pure H_2 gas.

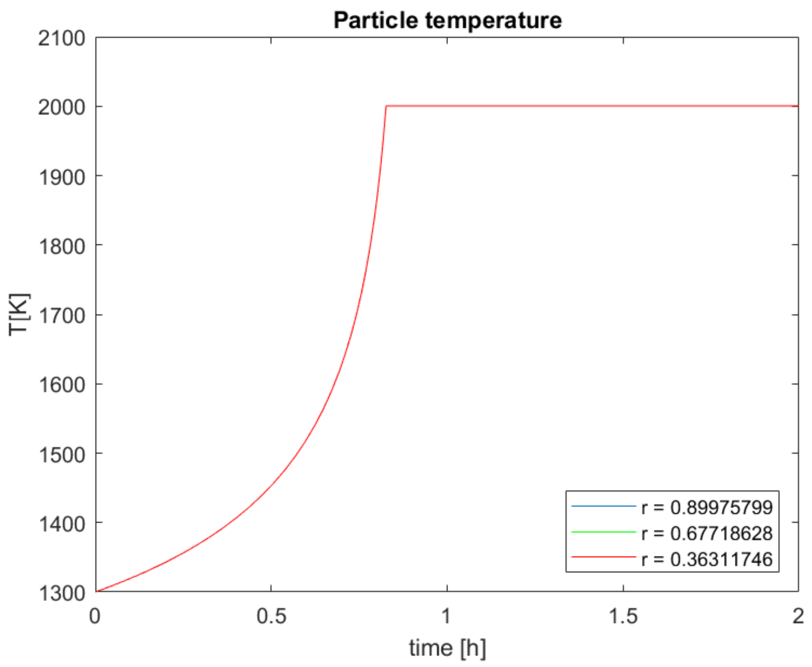


Figure 4.7: Particle temperature vs. time using input values found in **Listing 4.1** with no heat transfer (C3) which concerns reduction of an MnO_2 particle using pure CO gas.

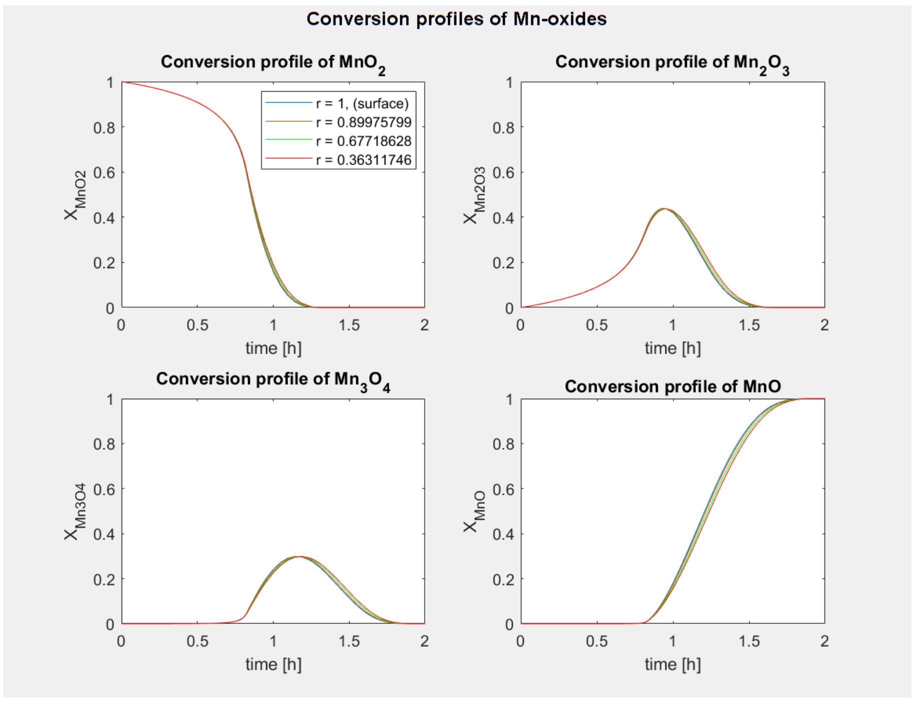


Figure 4.8: Different oxide fractions vs. time using input values found in **Listing 4.1** with no heat transfer (C3) which concerns reduction of an MnO_2 particle using pure CO gas.

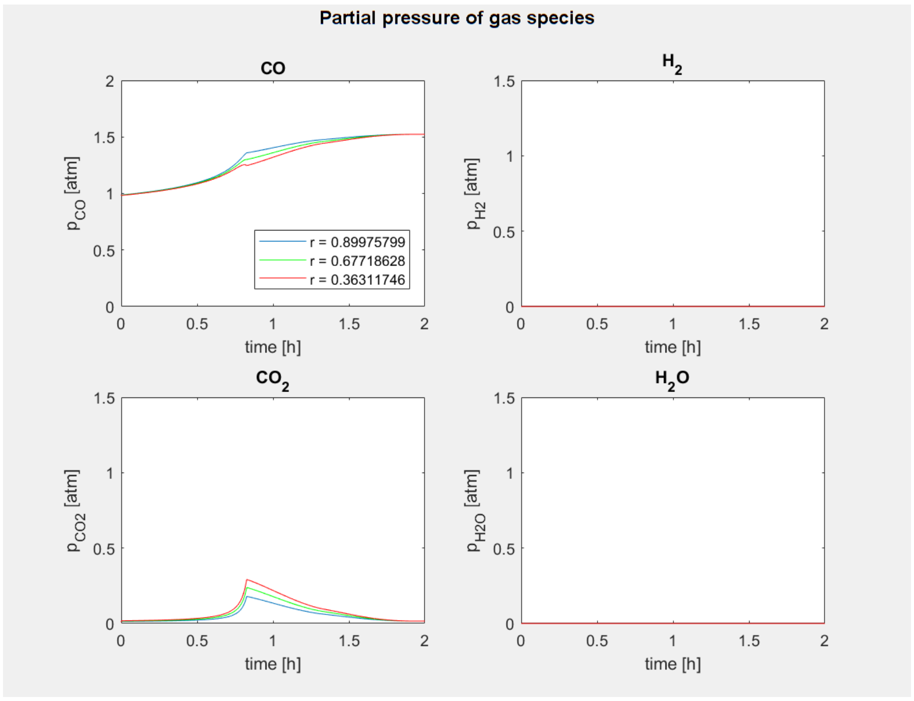


Figure 4.9: Partial pressure of different gas species vs. time using input values found in **Listing 4.1** with no heat transfer (C3) which concerns reduction of an MnO_2 particle using pure CO gas.

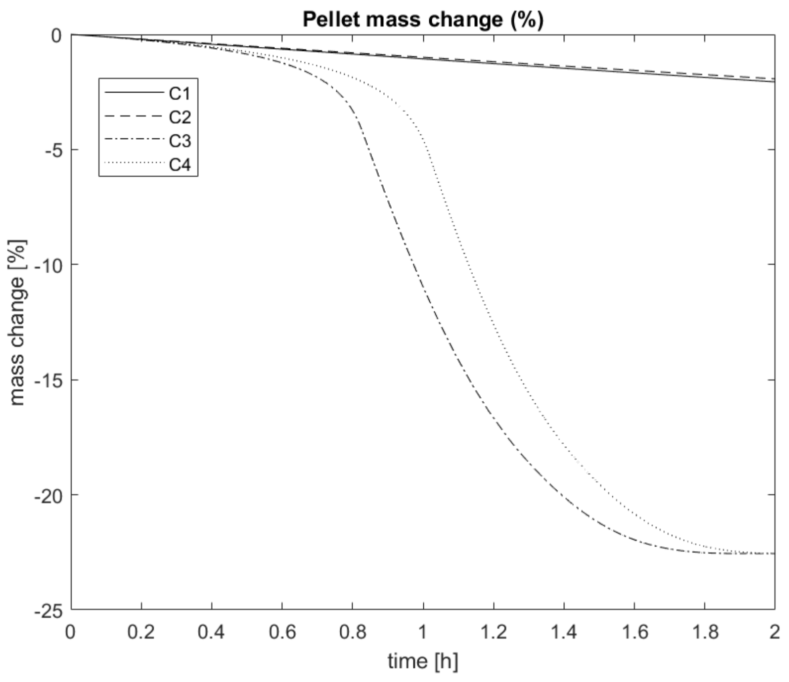


Figure 4.10: Mass change vs. time for case 1-4.

4.2 Reproduction of Experimental Results

In this section we should try to replicate some experimental results from Berg [7]. The different weightloss vs. time plots seen in **Figure 1.3**, will be used as a reference point. The particle dimensions in Lobo [8] are somewhat similar to the dimensions found in the work conducted by Berg [7]. However, some of the samples were cylindrical as well as cubic shaped. Nevertheless, in our attempt to replicate the results, we will only consider spherical shaped particles. The weightloss vs. time plots from Berg [7] differ from the mass change plots found in Lobo [8]. However, by changing to a positive sign in line 259, Script E.1. We should get similar types of plots to the ones found in Berg [7]. Now, let us start with the reduction/decomposition of MnO_2 to Mn_2O_3 (**Figure 1.3a**). The initial inputs used for all the results in this section are found in **Listing 4.3** if not otherwise pinpointed. The gas temperature is considered to be above 1600 kelvin in the first six scenarios (**Figures 4.11-4.16**) which arguably simulates the measured particle temperatures given by Berg [7] which were found to be 300 kelvin higher than the initial gas temperature of around 1300 kelvin. That being said, these temperature measurements were taken during reaction, nevertheless, we should consider similar temperatures to get our reactions to start instantly. One should note that if one wishes to start the procedure at 1300 kelvin, with no heat transfer, one should almost completely get the same result as when starting the procedure at 1600 kelvin.

```
%Input:
rp = 5e-3;          %Particle radius, m
Tb = 1600;          %Bulk temperature, K
%T0 = 300;          %Initial particle temperature, K
T0 = Tb;
vb = 0.1;           %Bulk velocity, m/s
tend = 3600*4;      %Ending time of simulation, s

VCOfrac = 0.99;   %Volume fraction of CO in bulk
VCO2frac = 0.01; %Volume fraction of CO2 in bulk
VH2frac = 0.00;   %Volume fraction of H2 in bulk
```

Listing 4.3: Initial values used for reproduction of experimental results

Now, lets see how the porosity factor affects the particle weightloss. First of all, let us turn off the reactions concerning reduction of Mn_2O_3 and Mn_3O_4 by setting their *pre-exponential constants* to zero. Three porosity factors were considered, 0.005, 0.05 and

0.3. It is assumed that our initial particle consists only of MnO_2. The results can be seen in **Figure 4.11**.

By changing the gas temperature from 1600 to 1800 kelvin with a constant porosity factor of 0.05 resulted in overall increased conversion rates as seen in **Figure 4.12**.

Considering the last reduction/decomposition path, Mn_2O_3 to MnO (**Figure 1.3b**), our initial particle now consist only of Mn_2O_3. We shall then see how both the porosity factor and the gas temperature affects the particle weightloss. The results with different porosity factors can be found in **Figure 4.13**. By changing the gas temperature from 1600 to 1800 kelvin with a constant porosity factor of 0.05 resulted in overall increased conversion rates as seen in **Figure 4.14**.

Now, let us increase the *reaction rate* of the first reaction in the reduction/decomposition path, MnO_2 to MnO (Reaction 1.3) without increasing the temperature. This is done by changing either the *pre-exponential constant* or the *activation energy* of the reaction. However, we shall only consider the *pre-exponential constant.* The results can be seen in **Figure 4.15**. The figure also includes an isolated reduction/decomposition path, MnO_2 to Mn_2O_3, we can then clearly see where Reaction 1.3 ends. By looking at the different reduction/decomposition results given in **Figure 1.3**, we clearly see that either an increase in the *pre-exponential constant* for Reaction 1.3 or an increase in overall temperature in our model fits better with the experimental results considering both the time frames and the total conversion of MnO_2 and Mn_2O_3. However, note that the heat transfer between the particle and the surrounding gas is not considered in the last two cases where the *pre-exponential constant* were considered to be above 23 m/s. The temperature were also locked at 2000 kelvin if reached. Now, if we consider heat transfer we can see the huge effect of having an endless reservoir of syngas cooling down our particle, see **Figure 4.16**. This indicates that Reactions 1.4-1.5 suddenly becomes very slow when the temperature is held around 1600 kelvin.

Let us take this a bit further by reducing the maximum model temperature from 2000 to 1500 kelvin as well as changing the *activation energy* for the different reactions (Reactions 1.3-1.5). We will, again, consider no heat transfer. The reason for our narrower temperature range of 1100-1500 kelvin in this example is to better represent the suggested outline of reaction progress 1.2 which is believed to occur in this temperature range [1, 7]. So why have we not used this temperature range for all our examples?

Now, we do not want to restrict our model to much when looking at the different factors affecting it. This is the main reason behind our decision to restrict the model partly in each example. This is also the reason why we have only considered to include the results from Berg [7] in the two last scenarios as they fit our new temperature range better. The new initial inputs used for producing the results concerning the narrower temperature range for reduction/decomposition path, MnO_2 to Mn_2O_3, are found in **Listing 4.4**.

```
%Input:
rp = 5e-3;         %Particle radius, m
Tb = 1100;         %Bulk temperature, K
%T0 = 300;         %Initial particle temperature, K
T0 = Tb;
vb = 0.1;          %Bulk velocity, m/s
tend = 3600*0.1;  %Ending time of simulation, s

VCOfrac = 0.99;   %Volume fraction of CO in bulk
VCO2frac = 0.01;  %Volume fraction of CO2 in bulk
VH2frac = 0.00;   %Volume fraction of H2 in bulk
```

Listing 4.4: Initial values used for reproduction of experimental results now with a narrower temperature range for MnO_2 to Mn_2O_3

The *activation energy* concerning Reaction 1.3 are now tested for two new values, 60 kJ/mol and 70 kJ/mol instead of 121 kJ/mol. The results can be seen in **Figure 4.17**.

Now, let us consider the reduction/decomposition path, Mn_2O_3 to MnO. The new initial inputs used for producing the results concerning the narrower temperature range for reduction/decomposition path, Mn_2O_3 to MnO, are found in **Listing 4.5**.

```
%Input:
rp = 5e-3;         %Particle radius, m
Tb = 1200;         %Bulk temperature, K
%T0 = 300;         %Initial particle temperature, K
T0 = Tb;
vb = 0.1;          %Bulk velocity, m/s
tend = 3600*1;    %Ending time of simulation, s

VCOfrac = 0.70;   %Volume fraction of CO in bulk
VCO2frac = 0.30;  %Volume fraction of CO2 in bulk
```

```
VH2frac = 0.00;  %Volume fraction of H2 in bulk
```

Listing 4.5: Initial values used for reproduction of experimental results now with a narrower temperature range for Mn_2O_3 to MnO

The *activation energy* concerning Reactions 1.4-1.5 are now tested for two new sets of values each, 120-130 kJ/mol, and 140-150 kJ/mol for Reaction 1.4 and Reaction 1.5 respectively. We have also considered sintering in the last example by increasing α to 2.5 from Equation 2.46. The results can be seen in **Figure 4.18**.

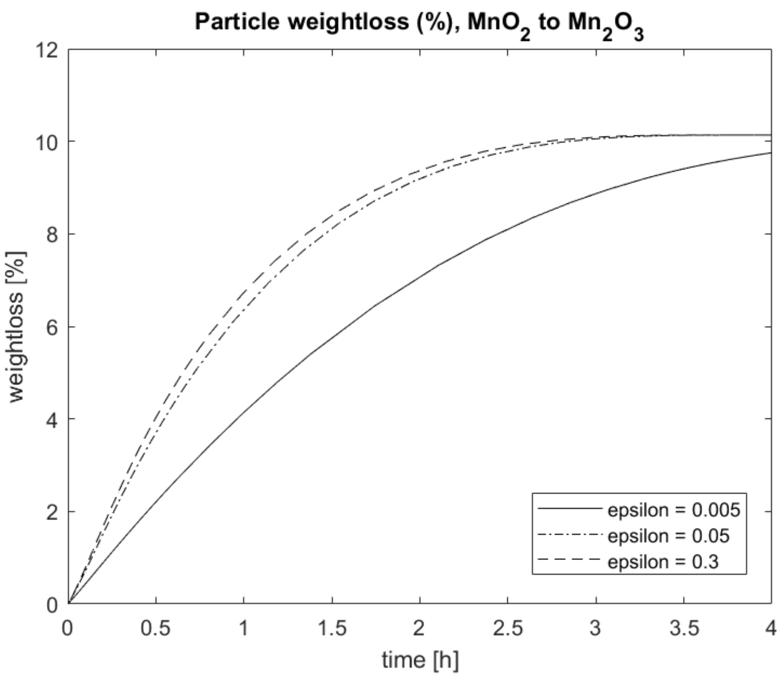


Figure 4.11: Weightloss vs. time for different porosity factors concerning the reduction path, MnO_2 to Mn_2O_3, using pure CO gas with an initial temperature of 1600 K using input values found in **Listing** 4.3 or **Table** 4.3 (R1).

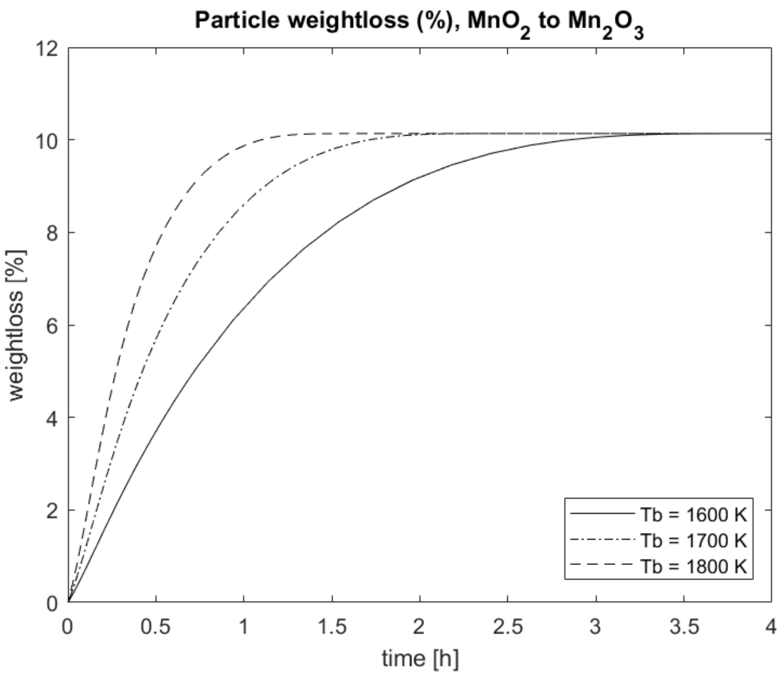


Figure 4.12: Weightloss vs. time for different gas temperatures concerning the reduction path, MnO_2 to Mn_2O_3, using pure CO gas. Porosity factor, $\epsilon = 0.05$. Input values used can be found in **Listing** 4.3 or **Table** 4.3 (R2).

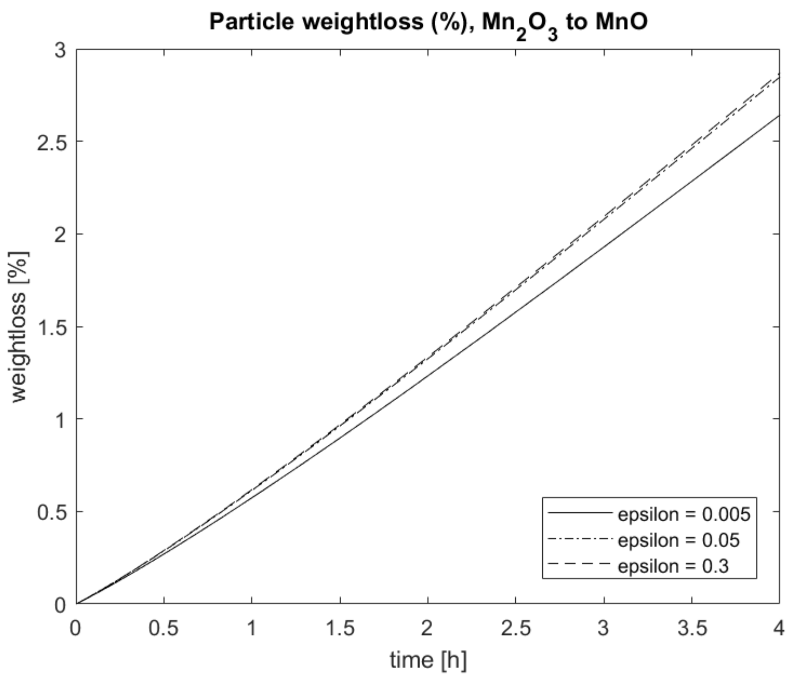


Figure 4.13: Weightloss vs. time for different porosity factors concerning the reduction path, Mn_2O_3 to MnO, using pure CO gas with an initial temperature of 1600 K using input values found in **Listing 4.3** or **Table 4.3** (R3).

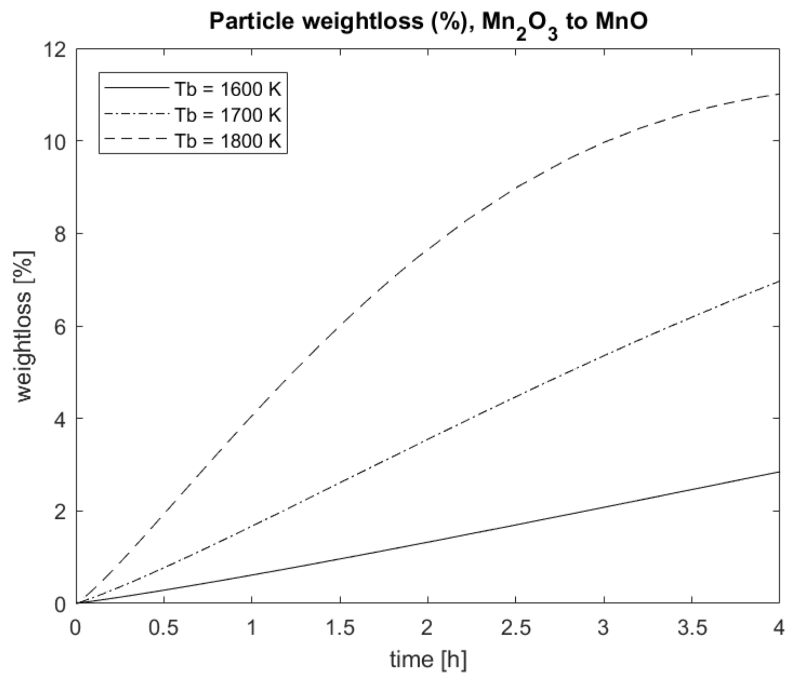


Figure 4.14: Weightloss vs. time for different gas temperatures concerning the reduction path, Mn_2O_3 to MnO, using pure CO gas. Porosity factor, $\epsilon = 0.05$. Input values used can be found in **Listing 4.3** or **Table 4.3** (R4).

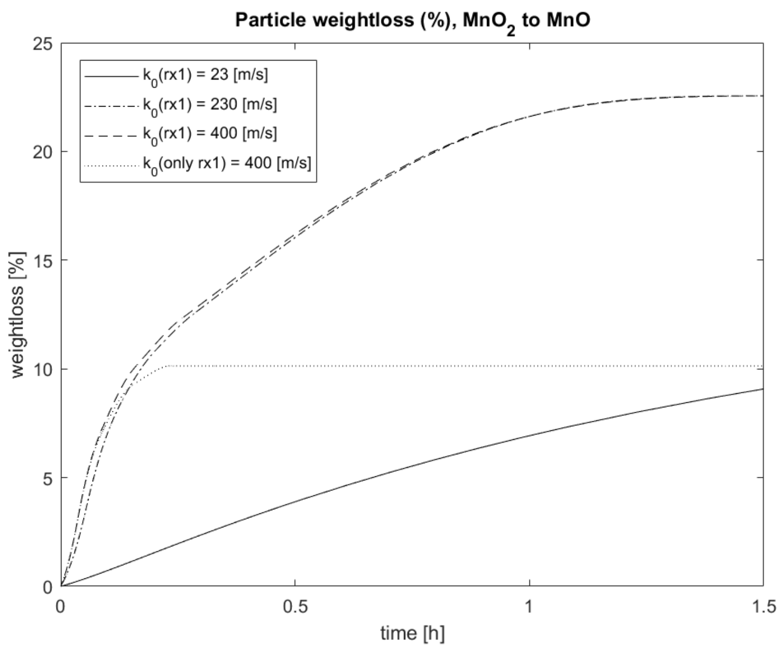


Figure 4.15: Weightloss vs. time for different *pre-exponential constants* for Reaction 1.3 concerning the reduction/decomposition path, MnO_2 to MnO, using pure CO gas. Input values used can be found in **Listing 4.3** or **Table 4.3** (R5) (with $tend = 1.5$). Porosity factor, $\epsilon = 0.05$. The line converging around 10 % weightloss represent the reduction path, MnO_2 to Mn_2O_3, with a *pre-exponential constant* of, $k_0 = 400 m/s$.

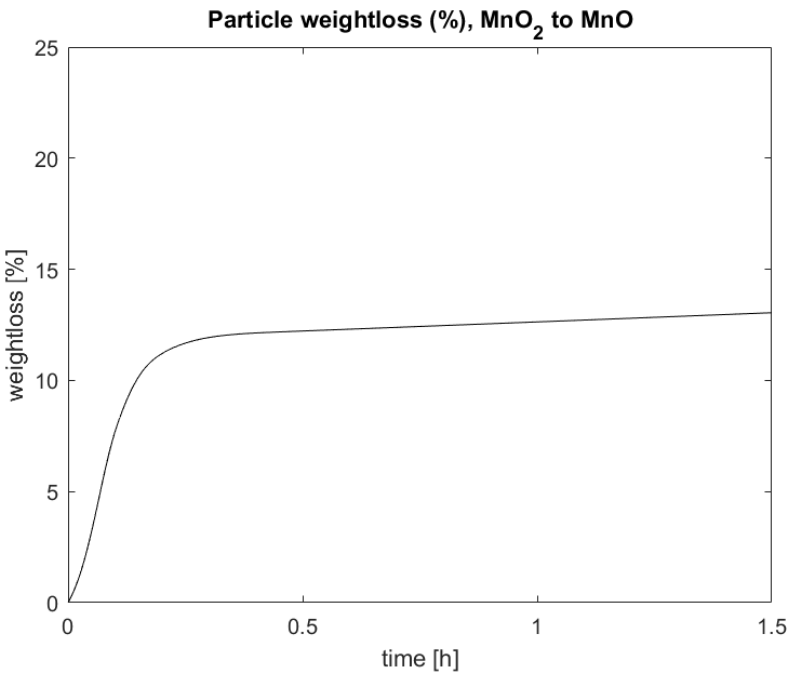


Figure 4.16: Weightloss vs. time with an increased pre-exponential constant ($k_0 = 230 m/s$) and heat transfer considered concerning the reduction/decomposition path, MnO_2 to MnO, using pure CO gas with an initial temperature of 1600 K. Input values used can be found in **Listing 4.3** or **Table 4.3** (with $tend = 1.5$ h) (R6).

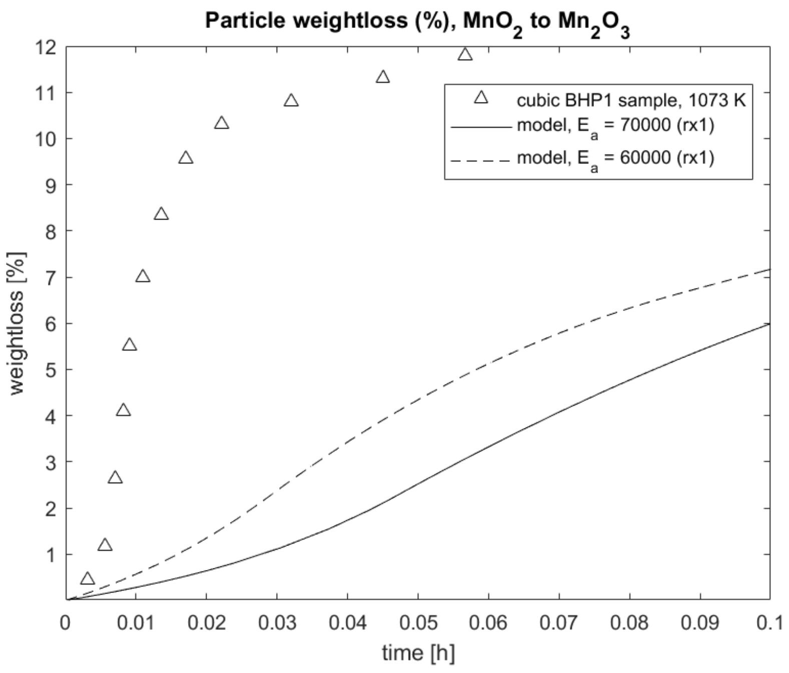


Figure 4.17: Weightloss vs. time with increased *activation energy* (60 kJ/mol and 70 kJ/mol for Reaction 1.3), no heat transfer concerning the reduction/decomposition path, MnO_2 to Mn_2O_3, CO rich gas with an initial temperature of 1100 K. Input values used can be found in **Listing 4.4** or **Table 4.3** (with $tend = 0.1$ h) (R7).

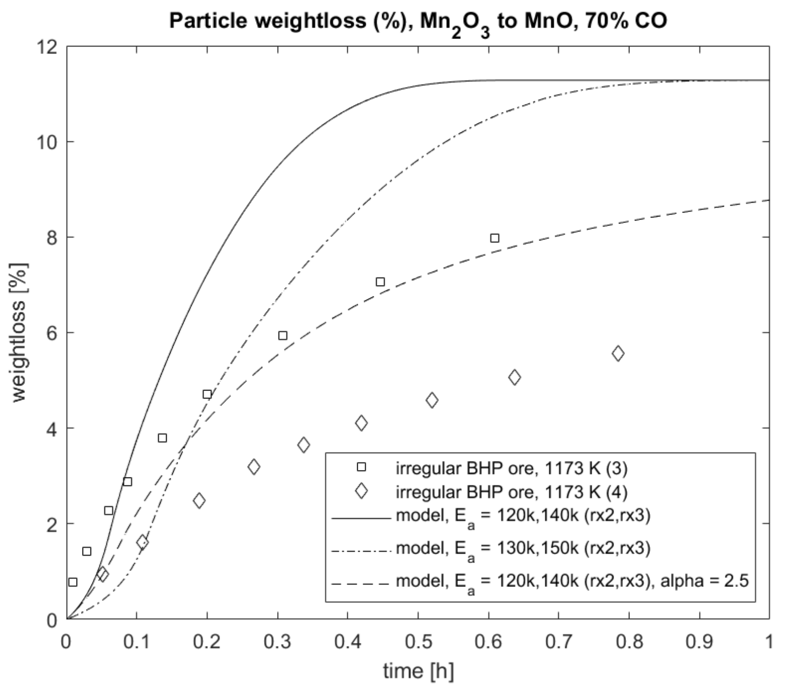


Figure 4.18: Weightloss vs. time with increased *activation energy* (120-130 kJ/mol and 140-150 kJ/mol for Reaction 1.4 and Reaction 1.5 respectively), no heat transfer concerning the reduction/decomposition path, Mn_2O_3 to MnO, CO rich gas with an initial temperature of 1200 K. Input values used can be found in **Listing 4.4** or **Table 4.3** (with $tend = 0.1$ h) (R8).

4.3 Industrial Case

We shall now present an industrial case (IC). The particle is now considered a lump of MnO_2 and Mn_2O_3 with a porosity factor, $\epsilon = 0.05$, similar to the lumps used in the experiment conducted by Lobo [8]. However, the raw material used in the experiments conducted by Lobo [8] also consisted of Braunite together with lesser amounts of Hematite and $CaCO_3$. Nevertheless, in our system, the amount of different phases in the considered lump can be found in **Table 4.1a**. Now, this is just an example, however, it should give us a feeling for how the model performs on cases concerning particles containing different oxides.

Our specified syngas mixture should be an off-gas blend from an manganese producing company, in this case Ferroglobe Mangan Norge AS to see how feasible it is to use off-gas to fuel a prereduction unit. The off-gas analysis are taken from previous work, Gustum [16]. We shall present two scenarios given in **Table 4.1b**, where the first scenario contains the lowest measured CO content from the furnace off-gas (within reason, disregarding readings during metal pouring) and a second scenario containing the highest measured CO content over a period of six months.

Table 4.1: Inputs used for the two industrially relevant scenarios.

(a) Phases in lump example

	MnO_2	Mn_2O_3
wt%	18.7	81.3

(b) Off-gas content in volume percent

	CO	CO_2	H_2	N_2
IC-S1	87	7.4	5	0.6
IC-S2	74	9.4	12.5	4.1

Now that we have a gas mixture of both CO and H_2 we would expect the WGSR to take effect. Since our model does not take N_2 into account, the amount will be neglected and set to be H_2O since one should assume some water vapour in the off-gas. For instance, the measured data is done on dry gas which explains the zero water content. With that being said, general input values used for both scenarios can be found in **Table 4.2**.

Table 4.2: Input values used for both scenarios.

$T_p[K]$	$T_b[K]$	$V_b[m/s]$	$R[m]$	$t[h]$
1300	1300	2	0.005	8

Our specified particle which is given in **Table 4.1a**, is roughly converted to a concentration profile represented in the script by

```
cMn020 = ones(length(rc)+1,1)*(10000); %mol/m3
cMn2030 = ones(length(rc)+1,1)*(47858/2); %mol/m3
```

Listing 4.6: Initial particle profiles

First of all, let us see how the WGSR react to the specified gas mixture. By assuming a porosity factor, $\epsilon = 1$, turning off any gas-solid reactions and setting the bulk velocity to zero, we should have a control volume which is the particle volume containing only the specified gas mixture. Mass transfer from the surrounding gas is still assumed to occur. We can then plot Equation 1.10 vs. time to see the effect, seen in **Figure 4.19**. We clearly see that the WGSR goes toward equilibrium, however, it is quite slow due to the constant mass transfer between the control volume and the surrounding gas mixture. The negative rate constant at the beginning implies that the WGSR goes to the left producing CO and H_2O which consumes heat as seen in **Figure 4.20**. This fits well with the work conducted by Demirel and Azcan [37] which states that the equilibrium amounts of CO and H_2O increase with increasing temperature. Now, lets consider both scenarios (IC-SI and IC-S2) with and without heat transfer. The results can be found in **Figure 4.21**.

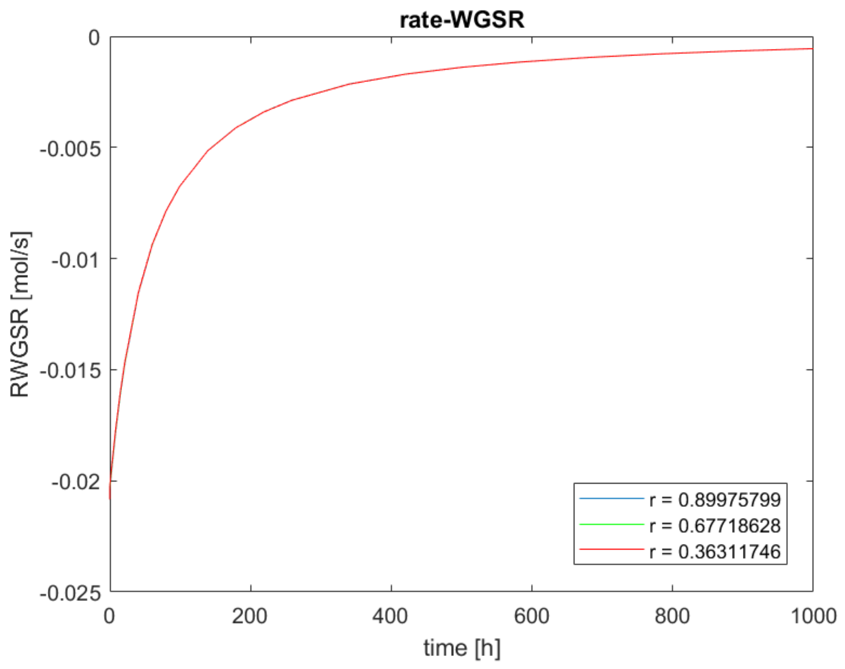


Figure 4.19: Equation 1.10 vs. time for an off-gas mixture (IC-SI) concerning the WGSR in a control volume at start temperature 1300 kelvin.

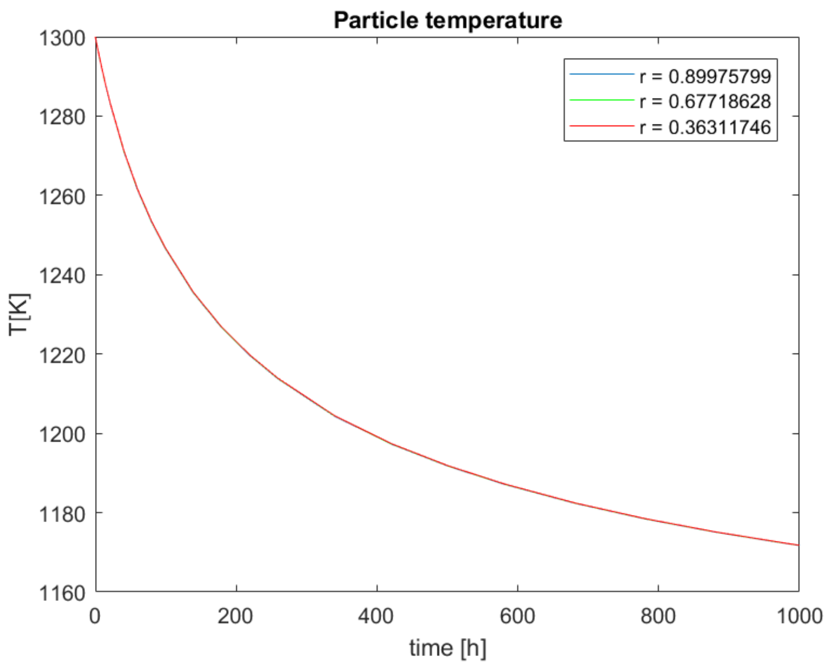


Figure 4.20: Particle temperature vs. time for an off-gas mixture (IC-SI) concering the WGSR in a control volume at start temperature 1300 kelvin.

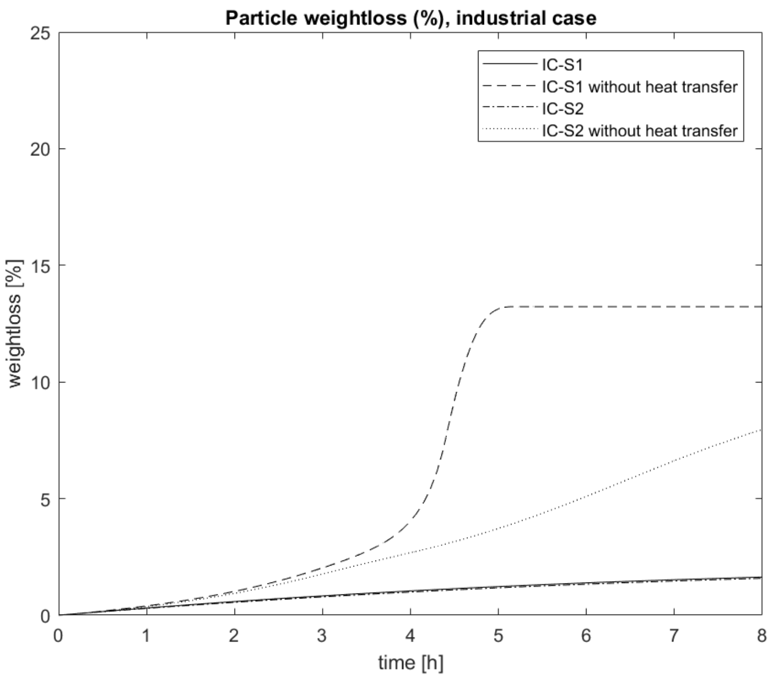


Figure 4.21: Weightloss vs. time for two industrially relevant scenarios, with and without heat transfer. Initial gas temperature of 1300 kelvin.

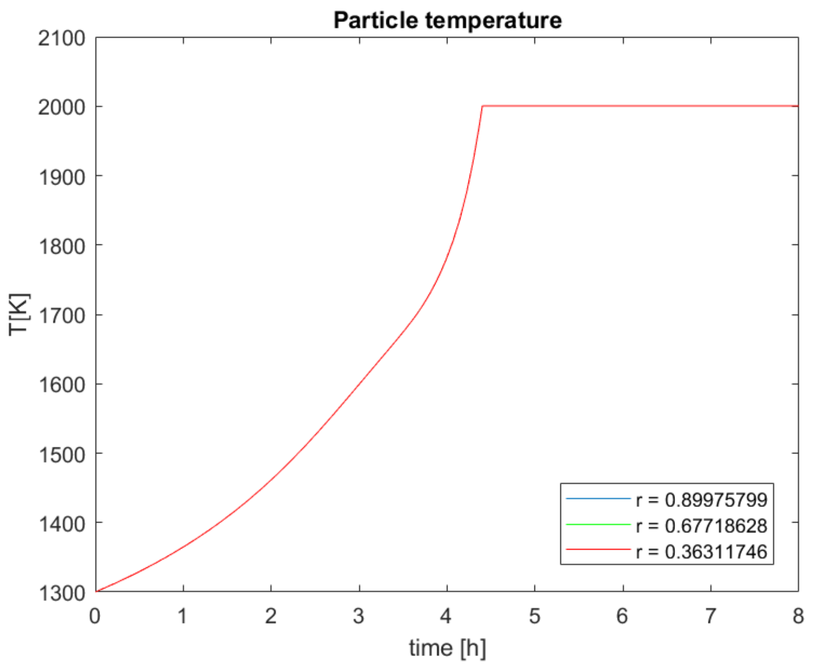


Figure 4.22: Particle temperature vs. time for IC-S1 without heat transfer.

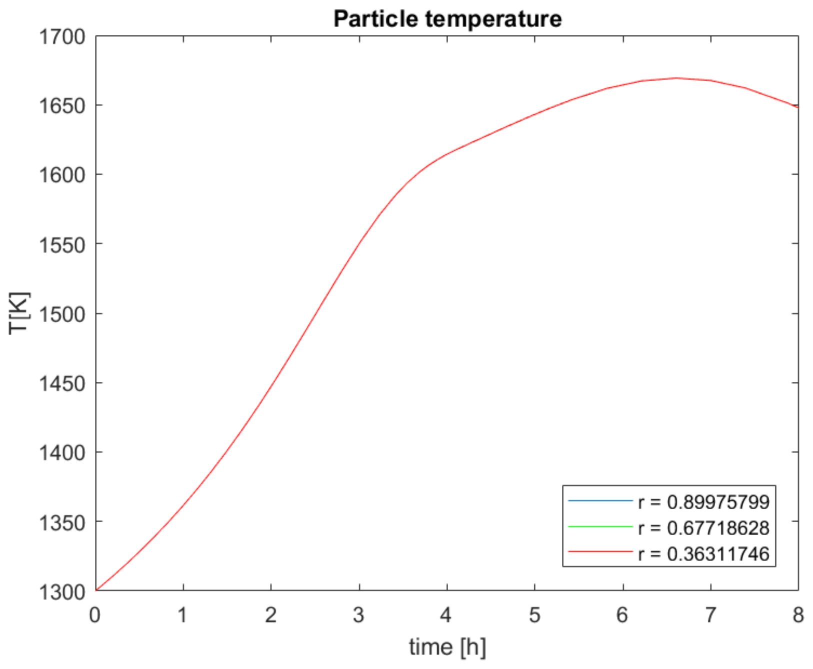


Figure 4.23: Particle temperature vs. time for IC-S2 without heat transfer.

Table 4.3: Input values for all the different cases presented in Chapter 4.

$Case$	$R_p[m]$	$T_{0,b}[K]$	$T_{0,p} = T_{0,b}[K]$	$V_b[m/s]$	V_{CO}	V_{CO_2}	V_{H_2}	$\bar{h}_c = 0$	ϵ	$k_0(rx1)$	$E_\alpha(rx1-3)$	$max(T_b)$
C1	0.005	1300	yes	0.1	0.99	0.01	0	no	0.3	default	default	1311
C2	0.005	1300	yes	0.1	0	0	0.99	no	0.3	default	default	1302
C3	0.005	1300	yes	0.1	0.99	0.01	0	yes	0.3	default	default	2000
C4	0.005	1300	yes	0.1	0	0	0.99	yes	0.3	default	default	2000
R1	0.005	1600	yes	0.1	0.99	0.01	0	no	[0.005,0.3]	default	default	1676
R2	0.005	[1600,1800]	yes	0.1	0.99	0.01	0	no	0.05	default	default	1952
R3	0.005	1600	yes	0.1	0.99	0.01	0	no	[0.005,0.3]	default	default	1605
R4	0.005	[1600,1800]	yes	0.1	0.99	0.01	0	no	0.05	default	default	1827
R5	0.005	1600	yes	0.1	0.99	0.01	0	yes/no	0.05	[23,400]	default	2000
R6	0.005	1600	yes	0.1	0.99	0.01	0	yes	0.05	230	default	2000
R7	0.005	1100	yes	0.1	0.99	0.01	0	yes	0.05	default	$[60k, 70k]_{rx1}$	1500
R8	0.005	1200	yes	0.1	0.99	0.01	0	yes	0.05	default	see figure	1500
IC-S1	0.005	1300	yes	2	87	7.4	5	yes/no	0.05	default	default	2000
IC-S2	0.005	1300	yes	2	74	9.4	12.5	yes/no	0.05	default	default	2000

Chapter 5

Discussion

This chapter provides a discussion on the previously presented results. It will include an analysis based on how well the model performed regarding the exercise of trying to recreate relevant experimental results. The big question concerning the commercial aspect of implementing a pretreatment unit using furnace off-gas as syngas will also be discussed.

5.1 Model Related Issues

5.1.1 The Heat Transfer Problem

After looking at the results from Chapter 4, it is readily apparent that there exist limitations with our model concerning heat transfer. For instance, the supplied syngas is assumed to have a constant bulk temperature which in reality is not true. In a confined space inside a crucible, the syngas temperature would increase significantly in the presence of gas-solid reactions heating up the system. In our case the endless reservoir of syngas will start to cool down our particle if the particle temperature starts to go above the bulk temperature of the gas. This is one reason to why our model performs better with increased syngas temperatures as this replicate the potential temperatures which could be reached if the temperature of the surrounding gas were set to change over time. By looking at **Figure 4.10**, we clearly see the impact of turning off the heat transfer between the particle and the surrounding gas. If we look at C1 and C3 in **Table 4.3**, we see that the only difference is the heat transfer column ($\bar{h}_c = 0$) where *yes* indicate no heat transfer. By only turning off the heat transfer, we see that C3 is completely reduced to MnO after two hours while C1 is only partly reduced. The reason behind C3's behavior is directly tied to the highly temperature dependent equation, Equation 2.42, which will rapidly increase the *reaction rate* of the present gas-sold reactions at higher temperatures. This is not the case for C1 where the particle is rapidly cooled down when reaching temperatures above $T_{0,b}$. This can also be seen in **Figure 4.12** and **Figure 4.14** were the initial temperature have been increased to certain temperatures concerning different reduction/decomposition paths. Note that the reduction/decomposition path, Mn_2O_3 to MnO (Reactions 1.4-1.5), is more affected by the temperature increase compared to Reaction 1.3. A solution to the problem would be to implement convective heat transfer through the moving gas outside the particle. Convective heat transfer depend on surface roughness and the type of fluid flow (*laminar* or *turbulent*), and could in principle be treated by adequate correlations as proposed by Whitaker [38] (as cited in [39]) concerning convective heat transfer to the gas over a sphere shaped particle.

With higher temperatures, thermal radiation will also become an important mechanism regarding heat transfer between a solid particle and its surroundings. The rate at which a solid body emits radiation is given by the Stefan Boltzmann law, Equation 5.1 (Szekely et al. [14])

$$q_R = \tilde{\epsilon}\sigma T_s^4 \tag{5.1}$$

where q_R is the radiant flux emitted by the surface, $\tilde{\epsilon}$ is the emissivity of the surface, σ is the Boltzmann constant and T_s is the surface temperature. The net radiative heat transfer will increase the total heat flux from the particle to the surroundings, further increasing the gas temperature. That being said, a more accurate model implementation regarding heat transfer should be prioritized for future work.

5.1.2 The Rate Constant Problem

From the results (especially **Figure 4.15** and **Figure 4.17**) we see that the different *rate constants*, both the *pre-exponential constant* and the *activation energy* affect the model greatly. Now, the problem comes from the fact that these values are hard to find as well as hard to interpret based on their model specific nature in that sense they could be found from either SCM, CM or some other simplified model [14]. For instance, in the work done by Berg [7] the SCM were used to obtain different *rate constants* based on the retreating reaction interface, r_c. However, SCM is considered to only be suitable for reactions that are entirely controlled by either mass transfer or surface reaction which does not correlate well with reactants that are porous [14, 7]. As Berg [7] suggests in his work, a big factor concerning the uncertainty behind obtaining the different rate constants is the inhomogeneity of the ore. Berg also states that for a meaningful study of these *rate constants*, the reactant (in this case the manganese ore) should be more homogeneous and preferably non-porous.

We have already suggested some kinetic parameters for reduction of manganese oxides by CO rich gas taken from [15, 7]. That being said, kinetic parameters related to H_2 are still not accounted for. Such values have been proven difficult to find but further investigation on the matter should be acknowledged for future work.

5.1.3 Integration Trouble

When integrating over the different concentration profiles to obtain the weightloss vs. time functions, integration related problems may occur as a consequence of low resolution (few collocation points). One should expect deviations when over half the particle is considered converted as we only have one collocation point representing the concentration profile from $r = R * 0.3631$ to the center. However, a different method was applied by taking the dot product of a weighted vector, **W**, and the concentration profile which describes the definite integral from 0 to R. That being said, a total weightloss estimate from our model (total conversion of MnO_2 to MnO) were expected to be around the theoretical value of 18.4 % [7] (see Equation 5.2), instead our model estimated a value of 22.5 %.

$$\left(1 - \frac{\phi_{\mathrm{MnO_2}} MW_{\mathrm{MnO}} * V}{\phi_{\mathrm{MnO_2}} MW_{\mathrm{MnO_2}} * V}\right) * 100\% = \left(1 - \frac{MW_{\mathrm{MnO}}}{MW_{\mathrm{MnO_2}}}\right) * 100\% = 18.4\% \tag{5.2}$$

A good place to start solving such a problem is to reevaluate the total mole balance of the system. Nevertheless, no errors were found as the different conversion profiles produce expected results as seen in **Figure 4.8**, where total conversion of MnO_2 to MnO after 2 hours outputs the expected value of $X_{\mathrm{MnO}} = 1$. After some time the error was found in line 255 in Script E.1 represented by

```
255 mloss(j+1) = (m(1)/m(j+1))-1;
```

which was corrected to

```
255 mloss(j+1) = (1-(m(j+1)/m(1))); %corrected
```

Sadly, due to time limitations the different weightloss vs. time plots were not updated. However, the script was corrected. As a final note regarding this issue, lets see how this error affect an example output of total conversion from MnO_2 to MnO. This exercise can be seen in **Figure 5.1**.

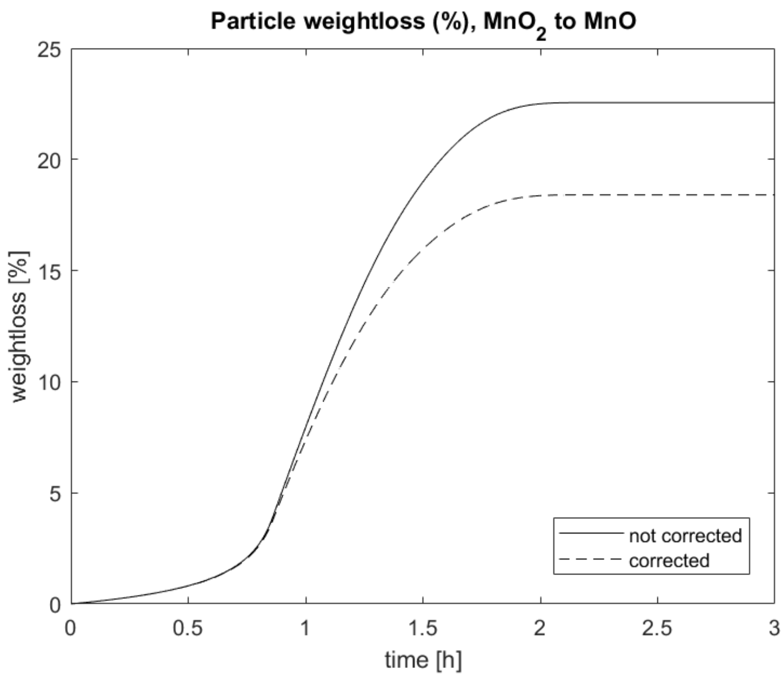


Figure 5.1: Weightloss vs. time plot (example), corrected vs. not corrected. Corrected plot reach a constant value of 18.4 %, uncorrected plot reach a constant value of 22.5 %.

5.1.4 Structural Changes in Gas-Solid Reactions

All the solid reactants in this report have been considered not to change its structure during reactions. However, when considering gas-solid reactions, the solid matrix through which diffusion is taking place may, in reality, undergo structural changes. Strictly speaking, these changes could be categorized into two types of change, one leading to bigger pores and the other causing densification (Szekely et al. [14]). In our case, sintering of present manganese oxides is expected to occur at high temperatures which increase material density. The effect of sintering is clearly visible in **Figure**

4.18 with an increase in the exponential factor, α, from Equation 2.46 which produced better results concerning the experimental data. Other types of structural changes (swelling, softening and cracking) are not considered in this report. That being said, all of these effects should be acknowledged for future work.

5.2 Commercially Applicable?

Now, for the remaining questions. Can this model be used in the industry, and is it feasible to implement a prereduction unit which utilizes off-gas as syngas? Before we start answering these questions, one should note that the answers will not cover every possibility concerning different types of off-gas from different manufacturing companies. The answers will most likely only be applicable to our specific case which is: Could Ferroglobe Mangan Norge AS use its off-gas to fuel a potential prereduction unit and could the implemented model say anything about a plausible scenario concerning prereduction of manganese oxide by CO-rich off-gas? From **Figure 4.21** we clearly see that both scenarios (IC-S1 and IC-S2) can be considered acceptable if neglecting any heat transfer. Nevertheless, the relatable temperature profiles (**Figure 4.23** and **Figure 4.22**) illuminates the problem with no heat transfer. Considering IC-S1 with no heat transfer, a gas temperature of 2000 kelvin is highly unlikely due to the different measured temperatures from the experiments conducted by Berg [7] which were found to lay in the range, 700-1100°C. IC-S2 with no heat transfer can be considered more realistic due to the maximum temperature around 1660 kelvin which corresponds well with the measured temperature increase of 360 kelvin from an initial temperate of 1300 kelvin suggested by the measured temperate range 700-1100°C [7]. That being said, the amount of oxide converted is appreciably high although considering the more realistic worst case scenario (IC-S2) of low CO content. One also has to consider the amount of water vapour in the off-gas which is in this case unknown. But seen from an industrial perspective, could be measured in the future if deemed necessary. An interesting topic that should be investigated in the future is how this model could be expanded to treat a multiple particle system. Our model only treats single particle systems which could have its drawbacks considering the fact that in reality a prereduction unit will pre-reduce multiple particles together. Here, solid to solid radiation will come into play as well as gas permeability which implies how well the gas travels through the material inside the prereduction unit.

Let us take a step back and observe the proposed scenario of implementing a prereduction unit for an manganese alloy producing company, in this case, Ferroglobe Mangan Norge AS. Thus, we will present some factors that one has to take into account if one wish to implement such a unit. These will be

- Electricity prices
- Price of carbon credit
- Carbon dioxide emissions
- Total electric power consumption
- Efficiency

The two first factors are considered more economically oriented and will not be further discussed here. The last three factors are related to the overall performance of the system (prereduction unit and furnace). Now, the motivating factor behind the decision of implementing a prereduction unit is to become more energy efficient which implies a more environmentally friendly industry. That being said, as stated in the work conducted by Tangstad et al. [5], increased degree of prereduction in a prereduction unit will increase the electrical power consumption in the electric furnace due to the fact that Reactions 1.3-1.8 are exothermic reactions. However, if the charge is reduced all the way down to MnO no endothermic Boudouard reaction producing CO (Reaction 1.18, reversed) will occur as there is no CO_2 present from the reduction of higher manganese oxides. This will decrease the electric energy consumption. There exist other factors related to decrease in energy consumption. These are mainly, removal of water and low temperature volatiles which increase furnace stability. That being said, if the ore that is being pre-reduced consist mostly of MnO_2, reduction of energy will be much less compared to a standalone electric furnace Tangstad et al. [5]. Nevertheless, our implemented framework should be of help in future work considering such calculations regarding total electric power consumption and efficiency.

Chapter 6

Conclusions

- A framework has been established concerning prereduction of manganese oxide using blends of CO, CO_2, H_2 and H_2O. That being said, it also have its flaws which needs to be accounted for in future work if one wish to make a commercially acceptable model, in particular related to variable gas temperatures.
- Comparison of different methods has been conducted. CM is considered to be the most realistic representation of prereduction of manganese ores using syngas.
- Kinetic parameters such as rate constants have to be acknowledged for future investigation and further expansion of the model as this is still considered a barrier regarding model implementation in the industry.
- There still exists some industrially relevant questions that go unanswered concerning prereduction. Here, our model can be used to assist further work and decisions on the matter.

Nomenclature

Non mathematical symbols

A	:	Area	$[m^2]$
A_{ex}	:	Exterior surface area	$[m^2]$
C_p	:	Constant pressure heat capacity	$[J/kgK]$
$c_{i,j}$	:	Molar concentration of specie i at position j	$[mol/m^3]$
$C_{i,s}$	:	Sutherland's constant for specie i	[]
$D_{i,eff}$	:	Effective diffusion coefficient of specie i in the gas mixture $i-mix$	$[m^2/s]$
$D_{i,j}$	:	Diffusion coefficient of specie i in j	$[m^2/s]$
E_α	:	Activation energy	$[J/mol]$
Gr	:	Grashof number	[]
h_e	:	Specific enthalpy	$[J/kg]$
k	:	Volumetric rate constant	$[1/s]$
k'	:	Superficial rate constant	$[m/s]$
k_0	:	Pre-exponential constant	$[m/s]$
L	:	Characteristic dimension	$[m]$
MW_i	:	Molecular weights of specie i	$[g/mol]$
N_i	:	Mass flux of specie i	$[mol/m^2s]$
Nu	:	Nusselt number	[]

P_b	:	Pressure in bar	$[bar]$
Pr	:	Prantl number	[]
q	:	Heat flux	$[W/m^2]$
r, R	:	Particle radius	$[m]$
r_c	:	Radius of the unreacted core	$[m]$
Re	:	Reynolds number	[]
s_i	:	Rate at which the concentration of specie i changes over time	$[mol/m^3s]$
Sc	:	Schmidt number	[]
Sh	:	Sherwood number	[]
T	:	Temperature	$[K,^\circ C,^\circ R]$
T_0	:	Reference temperature	$[^\circ R]$
T_b	:	Bulk temperature	$[K,^\circ C]$
T_s	:	Surface temperature	$[K,^\circ C]$
V	:	Volume	$[m^3]$
v'	:	Kinematic viscosity	$[m^2/s]$
V_b	:	Bulk velocity	$[m/s]$
V_i	:	Special atomic diffusion volume of specie i	[]
v_i	:	Gas velocity in direction i	$[m/s]$
$v_{i,j}$	:	Stoichiometric matrix	[]
X_i	:	Solid conversion of specie i	[]
$\bar{h}_c$	:	Heat transfer coefficient	$[W/m^2K]$
$\bar{k}_i$	:	Mass transfer coefficient of specie i	$[m/s]$
$\bar{R}'_i$	:	Reaction rate i	$[mol/m^2s]$

$\bar{R}_i$	:	Reaction rate if i	$[mol/m^3s]$
$\hat{k}$	:	Thermal conductivity	$[W/mK]$
$\hat{k}_eff$	:	Effective thermal conductivity	$[W/mK]$
$\triangle_r H_j$	:	Enthalpy of reaction j	$[J/mol, kJ/mol]$
$\triangle_r H_j^\ominus$	:	Standard enthalpy of reaction j	$[J/mol, kJ/mol]$

Greek Letters

ϵ	:	Local composition dependent porosity	[]
μ_i	:	Dynamic viscosity of specie i	$[cP]$
$\mu_{i,0}$	:	Reference viscosity of specie i at reference temperature T_0	$[cP]$
$\mu_{mixture}$	:	Dynamic viscosity of a gas mixture	$[Ns/M^2]$
ϕ_i	:	Molar density of specie i	$[mol/m^3]$
χ_i	:	Mole fraction of specie i	[]
ρ	:	Density	$[kg/m^3]$
$\rho_{I,j}$	:	Intrinsic density of specie j	$[kg/m^3]$
τ	:	Tortuosity	[]

Bibliography

[1] S. E. Olsen, M. Tangstad, and T. Lindstad. *Production of Manganese Ferroalloys.* Tapir Academic Press, 1st edition, 2007.

[2] H. Dalaker, E. Ringdalen, L. Kolbeinsen, and J. Mårdalen. Veikart for gass i metallindustrien, Økt verdiskaping og reduserte utslipp. *SINTEF, NTNU*, 2015.

[3] Norsk Industri. Veikart for prosessindustrien—økt verdiskapning med nullutslipp i 2050. *Rapport fra norskindustri.no*, 2016.

[4] Y. Gordon and J. Nell. Methods of manganese ore thermal-treatment prior to smelting - what to choose? *Volume I: Production Technologies and Operation. Presmelting Operations. The thirteenth International Ferroalloys Congress, June 9-13, 2013*, 2013.

[5] M. Tangstad, K. Ichihara, and E. Ringdalen. Pretreatment unit in ferromanganese production. *Kashima Works, SINTEF, NTNU*, 2015.

[6] M. Wang and B. Sundman. Thermodynamic assessment of the mn-o system. *Metall. Trans. B, vol. 23B, pp. 821-831*, 1992.

[7] K. L. Berg. Gaseous reduction of manganese ores. *Ph.D. thesis, Department of Metallurgy, The Norwegian Institute of Technology, Trondheim*, 1998.

[8] S. Lobo. Reduction of manganese ores using CO, H_2, CO_2 and H_2O blends. *NTNU*, 2015.

[9] C. A. Callaghan. Kinetics and catalysis of the water-gas-shift reaction: A microkinetic and graph theoretic approach. *Ph.D. thesis, Worcester Polytechnic Institute, Chemical Engineering Department*, 2006.

[10] B. Smith, M. Loganathan, and M. S. Shantha. A review of the water gas shift reaction kinetics. *International Journal Of Chemical Reactor Engineering, Volume 8, Review R4*, 2010.

[11] W. F. Pedolski and Y. G. Kim. Modeling the water-gas shift reaction. *Ind. Eng. Chem. Process Des. Dev. 13, 4, 415-421*, 1974.

[12] O. Levenspiel. *Chemical Reaction Engineering*. John Wiley & Sons, Inc, 3rd edition, 1999.

[13] R.B. Bird, W.E. Stewart, and E.N. Lightfoot. *Transport Phenomena*. John Wiley & Sons, Inc, revised 2nd edition, 2007.

[14] J. Szekely, J. W. Evans, and H. Y. Sohn. *Gas-solid reactions*. Academic press, Inc, 111 Fifth Avenue, New York, New York 10003, 1st edition, 1976.

[15] T. L. Schanche. Exergy relations in manganese production. *Master thesis. Norwegian University of Science and Technology (NTNU)*, 2013.

[16] M. G. Gustum. Smart utilization of co gas in industrial clusters. *Project thesis. Norwegian University of Science and Technology (NTNU)*, 2017.

[17] J. A. Bakken. *Metallurgical Engineering II, Fluid Flow and Heat Transfer, Advanced Course, Translation by Stephen Lobo and Leiv Kolbeinsen*. Department of Materials Science and Engineering, NTNU, 2009.

[18] W. E Ranz and W. R. Marshall, Jr. Evaporation from drops. parts one & two. *Chem. Eng. Progr. 48:141-6; 173-80*, 1952.

[19] B. Todd and JB. Young. Thermodynamic and transport properties of gases for use in solid oxide fuel cell modelling. *J. Power Sources. 110: 186-200*, 2002.

[20] E. N. Fuller, P. D. Schettler, and J. C. Giddings. New method for predicting of binary gas-phase diffusion coefficients. *Industrial & Engineering Chemistry 58 (5), 18-27*, 1966.

[21] L. J. Thibodeaux. *Environmental Chemodynamics: Movement of Chemicals in Air, Water, and Soil*. John Wiley & Sons, Inc, 2nd edition, 1996.

[22] S. Bretsznajder. *Prediction of Transport and Other Physical Properties of Fluids: International Series of Monographs in Chemical Engineering. Volume 11.* Elsevier, 2013.

[23] A. Kayode Coker. *Fortran Programs for Chemical Process Design, Analysis, and Simulation.* Gulf Publishing Company, 1995.

[24] Z. Tan. *Air Pollution and Greenhouse Gases.* Springer Singapore, 2014.

[25] Crane Company. Flow of fluids through valves, fittings, and pipe. *Technical Paper No. 410 (TP 410). P A-5*, 1988.

[26] CRC. *CRC Handbook of Chemistry and Physics.* In: Lide DR (ed), (94th edn), CRC Press, Inc., Boca Raton, Florida, USA, 2013.

[27] B. E. Poling, J. M. Prausnitz, and J. P. O'connell. *The Properties of Gases and Liquids.* The McGraw-Hill Companies, 5th edition, 2000.

[28] R. M. Felder and R. W. Rousseau. *Elementary Principles of Chemical Processes.* John Wiley & Sons, Inc, 3rd edition, 2005.

[29] T. Melchiori and P. Canu. Improving the quantitative description of reacting porous solids: Critical analysis of the shrinking core model by comparison to the generalized grain model. Technical report, Department of Industrial Engineering, University of Padova, Via Marzolo, 9, 35131 Padova, Italy, 2013.

[30] M. Helbæk and S. Kjelstrup. *Fysikalsk kjemi.* Fagbokforlaget Vigmostad og Bjørke AS, Postboks 6050 Postterminalen 5892 BERGEN, 2nd edition, 2006.

[31] W. F. Ames. *Numerical Methods for Partial Differential Equations.* Academic press, Inc. Boston, 3rd edition, 1992.

[32] H. Heinemann and J. J. Carberry. *Catalysis Reviews Science and Engineering. Volume 10.* Marcel Dekker, Inc. New York, 1974.

[33] B. A. Finlayson. *The Method of Weighted Residuals and Variational Principles. Volume 87.* Academic press, Inc. New York, 1972.

[34] J. V. Villadsen and W. E. Stewart. Solution of boundary-value problems by orthogonal collocation. *Reprinted from Chem. Eng. Sci, 22:1483-501, 1967*, 1967.

[35] L. F. Shampine and M. W. Reichelt. The matlab ode suite. *SIAM Journal on Scientific Computing, Vol. 18, pp. 1–22*, 1997.

[36] C. J. Geankoplis. *Transport processes and unit operations.* Allyn and Bacon, 2nd edition, 1983.

[37] E. Demirel and N. Azcan. Thermodynamic modeling of water-gas shift reaction in supercritical water. *Proceedings of the World Congress on Engineering and Computer Science 2012 Vol II, WCECS*, 2012.

[38] S. Whitaker. Forced convection heat transfer correlations for flow in pipes, past flat plates, single cylinders, single spheres, and flow in packed beds and tube bundles. *AIChE J 18, 361–371*, 1972.

[39] Z. Duan, B. He, and Y. Duan. Sphere drag and heat transfer. *Scientific Reports. 5:12304. doi:10.1038/srep12304*, 2015.

Appendix A

Mass/Heat Transfer Coefficient Functions (Matlab and Python)

The first function calculates the different *mass transfer coefficients*. The second function calculates the systems *heat transfer coefficient*. Both between a particle with diameter L and a gas mixture of CO, CO_2, H_2 and H_2O with bulk velocity vb. Input temperature is in Celsius. The last two scripts are related to the simplified example used in Chapter 3.

```
function [kmassi] = masstransferfun(i,T,L,vb,VCOfrac,VCO2frac,VH2frac)

%Temperature conversion:
TOC = T; %C
TOK = TOC + 275.15; %K
TOR = (TOK)/0.5555; %R
%Pressure:
PSTPatm = 1; %atm
PSTPPa = 101325; %Pa
PSTPbar = 1.01325; %bar

R = 8.3145; %Universal gas constant, m3*Pa/K/mol

%Dynamic viscosity (single specie):
muCO = 0.01720*((0.5555*518.67 + 118)/(0.5555*TOR +...
    118))*(TOR/518.67)^(3/2); %cP

muCO2 = 0.01480*((0.5555*527.67 + 240)/(0.5555*TOR +...
    240))*(TOR/527.67)^(3/2); %cP

```

```
muH2 = 0.00876*((0.5555*528.93 + 72)/(0.5555*T0R +...
    72))*(T0R/528.93)^(3/2); %cP

muH2O = 0.0657; %cP (350C)

%Molecular weights:
MWCO = 28.01; %g/mol
MWCO2 = 44.010; %g/mol
MWH2 = 2.016; %g/mol
MWH2O = 18.01528; %g/mol

%Special atomic diffusion volumes:
VCO = 18.9;
VCO2 = 26.9;
VH2 = 7.07;
VH2O = 12.7;

%Mole fraction calculations:
XCO = VCOfrac/PSTPatm;
XCO2 = VCO2frac/PSTPatm;
XH2 = VH2frac/PSTPatm;
XH2O = 1 - XCO - XCO2 - XH2;

%Dynamic viscosity for the CO-CO2-H2_H2O mixture:
mumix = (XCO*muCO*MWCO^0.5 + XCO2*muCO2*MWCO2^0.5 + XH2*muH2*MWH2^0.5 + ...
    XH2O*muH2O*MWH2O^0.5)/(XCO*MWCO^0.5 + XCO2*MWCO2^0.5 + XH2*MWH2^0.5...
    + XH2O*MWH2O^0.5);

mumixSI = 0.001*mumix; %N s/m2

%Density of gas [rho = (MW*P)/(RT)]:
MWtot = (XCO*MWCO + XCO2*MWCO2 + XH2*MWH2 + XH2O*MWH2O)*0.001; %kg/mol

rhogas = (MWtot*PSTPPa)/(R*T0K); %kg/m3

%Kinematic viscosity:
nudot = mumixSI/rhogas; %m2/s

if i == 1
    %Diffusion coefficient of the binary pair CO-mix:

    MWCOCO2 = 2*((1/MWCO)+(1/MWCO2))^(-1);
    MWCOH2 = 2*((1/MWCO)+(1/MWH2))^(-1);
    MWCOH2O = 2*((1/MWCO)+(1/MWH2O))^(-1);

    DCOCO2 = ((0.00143*(TOK^1.75))/(PSTPbar*(MWCOCO2^0.5)*(VCO^(1/3)+...
        VCO2^(1/3))^(2)))*0.0001; %m2/s
    DCOH2 = ((0.00143*(TOK^1.75))/(PSTPbar*(MWCOH2^0.5)*(VCO^(1/3)+...
        VH2^(1/3))^(2)))*0.0001; %m2/s
    DCOH2O = ((0.00143*(TOK^1.75))/(PSTPbar*(MWCOH2O^0.5)*(VCO^(1/3)+...
        VH2O^(1/3))^(2)))*0.0001; %m2/s

    DCOmix = (1 - XCO)/(XCO2/DCOCO2 + XH2/DCOH2 + XH2O/DCOH2O);

    %Schmidt number:
    Sc = nudot/DCOmix;

    %Reynolds number:
    Re = (vb*L)/nudot;

    %Sherwood number:
    Sh = 2.0 + 0.6*(Re^0.5)*(Sc^(1/3));

    %Mass transfer coefficient:
    kmassCO = (Sh*DCOmix)/L; %m/s

    kmassi = kmassCO;
end

if i == 2
    %Diffusion coefficient of the binary pair H2-mix:

    MWH2CO = 2*((1/MWH2)+(1/MWCO))^(-1);
    MWH2CO2 = 2*((1/MWH2)+(1/MWCO2))^(-1);
    MWH2H2O = 2*((1/MWH2)+(1/MWH2O))^(-1);

    DH2CO = ((0.00143*(TOK^1.75))/(PSTPbar*(MWH2CO^0.5)*(VH2^(1/3)+...
        VCO^(1/3))^(2)))*0.0001; %m2/s
    DH2CO2 = ((0.00143*(TOK^1.75))/(PSTPbar*(MWH2CO2^0.5)*(VH2^(1/3)+...
        VCO2^(1/3))^(2)))*0.0001; %m2/s
    DH2H2O = ((0.00143*(TOK^1.75))/(PSTPbar*(MWH2H2O^0.5)*(VH2^(1/3)+...
        VH2O^(1/3))^(2)))*0.0001; %m2/s

    DH2mix = (1 - XH2)/(XCO/DH2CO + XCO2/DH2CO2 + XH2O/DH2H2O);

    %Schmidt number:
    Sc = nudot/DH2mix;

    %Reynolds number:
    Re = (vb*L)/nudot;

    %Sherwood number:
    Sh = 2.0 + 0.6*(Re^0.5)*(Sc^(1/3));

    %Mass transfer coefficient:
    kmassH2 = (Sh*DH2mix)/L; %m/s

    kmassi = kmassH2;
end

if i == 3
    %Diffusion coefficient of the binary pair CO2-mix:

    MWCO2CO = 2*((1/MWCO2)+(1/MWCO))^(-1);
    MWCO2H2 = 2*((1/MWCO2)+(1/MWH2))^(-1);
    MWCO2H2O = 2*((1/MWCO2)+(1/MWH2O))^(-1);

    DCO2CO = ((0.00143*(TOK^1.75))/(PSTPbar*(MWCO2CO^0.5)*(VCO2^(1/3)+...
        VCO^(1/3))^(2)))*0.0001; %m2/s
    DCO2H2 = ((0.00143*(TOK^1.75))/(PSTPbar*(MWCO2H2^0.5)*(VCO2^(1/3)+...
        VH2^(1/3))^(2)))*0.0001; %m2/s
    DCO2H2O = ((0.00143*(TOK^1.75))/(PSTPbar*(MWCO2H2O^0.5)*(VCO2^(1/3)+...
        VH2O^(1/3))^(2)))*0.0001; %m2/s

    DCO2mix = (1 - XCO2)/(XCO/DCO2CO + XH2/DCO2H2 + XH2O/DCO2H2O);

    %Schmidt number:
    Sc = nudot/DCO2mix;

    %Reynolds number:
    Re = (vb*L)/nudot;

    %Sherwood number:
    Sh = 2.0 + 0.6*(Re^0.5)*(Sc^(1/3));

    %Mass transfer coefficient:
```

```matlab
    kmassCO2 = (Sh*DCO2mix)/L; %m/s

    kmassi = kmassCO2;
end

if i == 4
    %Diffusion coefficient of the binary pair H2O-mix:

    MWH2OCO = 2*((1/MWH2O)+(1/MWCO))^(-1);
    MWH2OH2 = 2*((1/MWH2O)+(1/MWH2))^(-1);
    MWH2OCO2 = 2*((1/MWH2O)+(1/MWCO2))^(-1);

    DH2OCO = ((0.00143*(TOK^1.75))/(PSTPbar*(MWH2OCO^0.5)*(VH2O^(1/3)...
        +VCO^(1/3))^(2)))*0.0001; %m2/s
    DH2OH2 = ((0.00143*(TOK^1.75))/(PSTPbar*(MWH2OH2^0.5)*(VH2O^(1/3)...
        +VH2^(1/3))^(2)))*0.0001; %m2/s
    DH2OCO2 = ((0.00143*(TOK^1.75))/(PSTPbar*(MWH2OCO2^0.5)*(VH2O^(1/3)...
        +VCO2^(1/3))^(2)))*0.0001; %m2/s

    DH2Omix = (1 - XH2O)/(XCO/DH2OCO + XH2/DH2OH2 + XCO2/DH2OCO2);

    %Schmidt number:
    Sc = nudot/DH2Omix;

    %Reynolds number:
    Re = (vb*L)/nudot;

    %Sherwood number:
    Sh = 2.0 + 0.6*(Re^0.5)*(Sc^(1/3));

    %Mass transfer coefficient:
    kmassH2O = (Sh*DH2Omix)/L; %m/s

    kmassi = kmassH2O;
end

end
```

Listing A.1: Mass transfer coefficient function (Matlab)

```matlab
function [heattran] = heattransferfun(T,L,vb,VCOfrac,VCO2frac,VH2frac)

%Temperature conversion:
```

```
T0C = T; %C
T0K = T0C + 275.15; %K
T0R = (T0K)/0.5555; %R
%Pressure:
PSTPatm = 1; %atm
PSTPPa = 101325; %Pa
PSTPbar = 1.01325; %bar

R = 8.3145; %Universal gas constant, m3*Pa/K/mol

%Dynamic viscosity (single specie):
muCO = 0.01720*((0.5555*518.67 + 118)/(0.5555*T0R +...
    118))*(T0R/518.67)^(3/2); %cP

muCO2 = 0.01480*((0.5555*527.67 + 240)/(0.5555*T0R +...
    240))*(T0R/527.67)^(3/2); %cP

muH2 = 0.00876*((0.5555*528.93 + 72)/(0.5555*T0R +...
    72))*(T0R/528.93)^(3/2); %cP

muH2O = 0.0657; %cP (350C)

%Molecular weights:
MWCO = 28.01; %g/mol
MWCO2 = 44.010; %g/mol
MWH2 = 2.016; %g/mol
MWH2O = 18.01528; %g/mol

%Special atomic diffusion volumes:
VCO = 18.9;
VCO2 = 26.9;
VH2 = 7.07;
VH2O = 12.7;

%Mole fraction calculations:
XCO = VCOfrac/PSTPatm;
XCO2 = VCO2frac/PSTPatm;
XH2 = VH2frac/PSTPatm;
XH2O = 1 - XCO - XCO2 - XH2;

%Dynamic viscosity for the CO-CO2-H2_H2O mixture:
mumix = (XCO*muCO*MWCO^0.5 + XCO2*muCO2*MWCO2^0.5 + XH2*muH2*MWH2^0.5 + ...
    XH2O*muH2O*MWH2O^0.5)/(XCO*MWCO^0.5 + XCO2*MWCO2^0.5 + XH2*MWH2^0.5...
    + XH2O*MWH2O^0.5);

mumixSI = 0.001*mumix; %N s/m2

%Density of gas [rho = (MW*P)/(RT)]:
MWtot = (XCO*MWCO + XCO2*MWCO2 + XH2*MWH2 + XH2O*MWH2O)*0.001; %kg/mol

rhogas = (MWtot*PSTPPa)/(R*TOK); %kg/m3

%Kinematic viscosity:
nudot = mumixSI/rhogas; %m2/s

%Thermal conductivity:
khatCO = (5.067*10^(-4) + (9.125*10^(-5))*TOK + (-3.524*10^(-8))*TOK^2 +...
    (8.199*10^(-12))*TOK^3); %J/m/K/s

khatCO2 = (-7.215*10^(-3) + (8.015*10^(-5))*TOK + (5.477*10^(-9))*TOK^2
    +...
    (-1.053*10^(-11))*TOK^3); %J/m/K/s

khatH2 = (8.099*10^(-3) + (6.689*10^(-4))*TOK + (-4.158*10^(-7))*TOK^2 +...
    (1.562*10^(-10))*TOK^3); %J/m/K/s

khatH2O = (7.341*10^(-3) + (-1.013*10^(-5))*TOK + (1.801*10^(-7))*TOK^2
    +...
    (-9.100*10^(-11))*TOK^3); %J/m/K/s

%Constant pressure heat capacity:
cpCO = 10^6*(28.95*10^(-3) + (0.4110*10^(-5))*TOC +...
    (0.3548*10^(-8))*TOC^2 + ((-2.220)*10^(-12))*TOC^3)/MWCO; %J/kg/K

cpCO2 = 10^6*(36.11*10^(-3) + (4.233*10^(-5))*TOC +...
    ((-2.887)*10^(-8))*TOC^2 + (7.464*10^(-12))*TOC^3)/MWCO2; %J/kg/K

cpH2 = 10^6*(28.84*10^(-3) + (0.00765*10^(-5))*TOC +...
    (0.3288*10^(-8))*TOC^2 + ((-0.8698)*10^(-12))*TOC^3)/MWH2; %J/kg/K

cpH2O = 10^6*(33.46*10^(-3) + (0.6880*10^(-5))*TOC +...
    (0.7604*10^(-8))*TOC^2 + ((-3.594)*10^(-12))*TOC^3)/MWH2O; %J/kg/K

khatmix = (XCO*khatCO + XCO2*khatCO2 + XH2*khatH2 + XH2O*khatH2O); %J/m/K/s
```

```

cpmix = (XCO*cpCO + XCO2*cpCO2 + XH2*cpH2 + XH2O*cpH2O); %J/kg/K

%Prantl number:
Pr = (nudot*cpmix*rhogas)/khatmix;

%Reynolds number:
Re = (vb*L)/nudot;

%Nusselt number:
Nu = 2.0 + 0.6*(Re^0.5)*(Pr^(1/3));

%Heat transfer coefficient:
heattran = (Nu*khatmix)/L; %W/m2/K
end
```

Listing A.2: Heat transfer coefficient function (Matlab)

```
"""
Input:
"""
T0 = 900 #C (bulk temperature)
L = 0.002 #m (particle diameter)
V_b = 5 #m/s (bulk velocity)
R = 8.3145 #Universal gas constant (m3*Pa/K/mol)

"""
Mass transfer coefficient calculation:
"""
#temperature conversion:
T0_K = T0 + 275.15 #K
T0_R = (T0_K)/0.5555 #R

#Pressure:
P_STP_atm = 1 #atm
P_STP_Pa = 101325 #Pa
P_STP_bar = 1.01325 #bar

#dynamic viscosity (single specie):
mu_O2 = 0.02018*((0.5555*526.05 + 127)/(0.5555*T0_R + 127))*(T0_R/
                526.05)**(3/2) #cP
mu_N2 = 0.01781*((0.5555*540.99 + 111)/(0.5555*T0_R + 111))*(T0_R/
                 540.99)**(3/2) #cP
```

```

#molecular weights:
MW_O2 = 31.999 #g/mol
MW_N2 = 28.02 #g/mol

#special atomic diffusion volumes:
V_O2 = 16.6
V_N2 = 17.9

#mole fraction calculations:
P_O2_bulk = 0.8 #atm

X_O2 = P_O2_bulk/P_STP_atm
X_N2 = 1 - X_O2

#dynamic viscosity O2-N2 mixture in [N s/m2]:
mu_O2_N2 = (X_O2*mu_O2*MW_O2**0.5 + X_N2*mu_N2*MW_N2**0.5
             )/(X_O2*MW_O2**0.5 + X_N2*MW_N2**0.5)

mu_O2_N2_SI = 0.001*mu_O2_N2 #N s/m2

#density of gas [rho = (MW*P)/(RT)]:
MW_tot = (X_N2*MW_N2 + X_O2*MW_O2)*0.001 #kg/mol

rho_gas = (MW_tot*P_STP_Pa)/(R*T0_K) #kg/m3

#kinematic viscosity:
nu_dot = mu_O2_N2_SI/rho_gas #m2/s

#diffusion coefficient of the binary pair O2-N2:
MW_O2_N2 = 2*((1/MW_O2)+(1/MW_N2))**(-1)

D_O2_N2 = ((0.00143*(T0_K**1.75))/(P_STP_bar*(MW_O2_N2**0.5)*
             (V_O2**(1/3)+V_N2**(1/3))**(2))))*0.0001 #m2/s

#Schmidt number:
Sc = nu_dot/D_O2_N2

#Reynolds number:
Re = (V_b*L)/nu_dot

#Sherwood number:
```

```python
Sh = 2.0 + 0.6*(Re**0.5)*(Sc**(1/3))

"""
Output:
"""
#mass transfer coefficient:
k_mass = (Sh*D_O2_N2)/L #m/s
```

Listing A.3: Mass transfer coefficient script for the ZnS example (Python)

```python
"""
Input:
"""
T0 = 900 #C (bulk temperature)
L = 0.002 #m (particle diameter)
V_b = 5 #m/s (bulk velocity)
R = 8.3145 #Universal gas constant (m3*Pa/K/mol)

"""
Heat transfer coefficient calculation:
"""
#temperature conversion:
T0_K = T0 + 275.15 #K
T0_R = (T0_K)/0.5555 #R

#Pressure:
P_STP_atm = 1 #atm
P_STP_Pa = 101325 #Pa
P_STP_bar = 1.01325 #bar

#dynamic viscosity (single specie):
mu_O2 = 0.02018*((0.5555*526.05 + 127)/(0.5555*T0_R + 127))*(T0_R/
                526.05)**(3/2) #cP
mu_N2 = 0.01781*((0.5555*540.99 + 111)/(0.5555*T0_R + 111))*(T0_R/
                 540.99)**(3/2) #cP

#molecular weights:
MW_O2 = 31.999 #g/mol
MW_N2 = 28.02 #g/mol

#mole fraction calculations:
P_O2_bulk = 0.8 #atm

```

```
X_O2 = P_O2_bulk/P_STP_atm
X_N2 = 1 - X_O2

#dynamic viscosity O2-N2 mixture in [N s/m2]:
mu_O2_N2 = (X_O2*mu_O2*MW_O2**0.5 + X_N2*mu_N2*MW_N2**0.5
             )/(X_O2*MW_O2**0.5 + X_N2*MW_N2**0.5)

mu_O2_N2_SI = 0.001*mu_O2_N2 #N s/m2

#density of gas [rho = (MW*P)/(RT)]:
MW_tot = (X_N2*MW_N2 + X_O2*MW_O2)*0.001 #kg/mol

rho_gas = (MW_tot*P_STP_Pa)/(R*T0_K) #kg/m3

#kinematic viscosity:
nu_dot = mu_O2_N2_SI/rho_gas #m2/s

#thermal conductivity:
k_hat_O2 = 0.024 #J/m/K/s  (25C)
k_hat_N2 = (3.919*10**(-4) + (9.816*10**(-5))*T0_K + (-5.067*10**(-8))*T0_K
    **2
            + (1.504*10**(-11))*T0_K**3) #J/m/K/s

k_hat_mix = X_O2*k_hat_O2 + X_N2*k_hat_N2

#constant pressure heat capacity:
c_p_O2 = 10**6*(29.10*10**(-3) + (1.158*10**(-5))*T0 + ((-0.6076)*10**(-8))
    *T0**2 +
          (1.311*10**(-12))*T0**3)/MW_O2 #J/kg/K
c_p_N2 = 10**6*(29*10**(-3) + (0.2199*10**(-5))*T0 + (0.5723*10**(-8))*T0
    **2 +
          ((-2.871)*10**(-12))*T0**3)/MW_N2 #J/kg/K

c_p_mix = X_O2*c_p_O2 + X_N2*c_p_N2 #J/kg/K

#Prantl number:
Pr = (nu_dot*c_p_mix*rho_gas)/k_hat_mix

#Reynolds number:
Re = (V_b*L)/nu_dot

#Nusselt number:
```

```python
Nu = 2.0 + 0.6*(Re**0.5)*(Pr**(1/3))

"""
Output:
"""
#mass transfer coefficient:
h_heat = (Nu*k_hat_mix)/L #W/m2/K
```

Listing A.4: Heat transfer coefficient script for the ZnS example (Python)

Appendix B

Shrinking-Core Model Script, Example Case (Python)

This script is used for the calculation of the shrinking-core model for the example case.

```python
import time
import numpy as np
import matplotlib.pyplot as plt
start = time.time()
"""
Input:
"""
T0 = 900 #C (bulk gas temperature)
P_O2_atm_b = 0.8 #atm (bulk partial pressure of O2)
D_mix = 0.00022812 #m^2/s (diffusion coefficient of the binary pair O2-N2)
kg = 0.71032 #m/s (mass transfer coefficient)
r_0 = 0.001 #m (initial particle radius)
t = 250 #duration in seconds

"""
SCM calculations:
"""
#constants:
R = 8.3145 #universal gas constant (m3*Pa/K/mol)
phi_ZnS = 41300 #mol/m3
MW_ZnS = 97.474 #g/mol
rho_ZnS = 4090 #kg/m3

#2ZnS + 3O2 = 2ZnO + 2SO2:
a = 2
```

```python
b = 3
c = 2
d = 2

#porosity:
epsilon = 1 - (phi_ZnS*((MW_ZnS*0.001)/rho_ZnS))

#tortuosity:
tau = 1

#effective diffusion coefficient:
D_eff = D_mix*(epsilon/tau)

#temperature conversion:
T0_K = T0 + 273.15 #K

#needed for the rate constant calculation:
k_0_r1 = 23000.00 #m/s (pre-exponential constant for the reaction)
E_a_r1 = 121032.00 #J/mol (activation energy for the reaction)

k = k_0_r1*np.exp((-E_a_r1)/(R*T0_K)) #m/s

#pressure to concentration (O2):
P_O2_Pa_b = P_O2_atm_b*101325 #Pa
c_O2_b = P_O2_Pa_b/(R*T0_K) #mol O2/m3

#initial values:
dt = 1 #sec
r_c = r_0 #m

#conversion vectors:
X_ZnS = [None for i in range(t+1)]
X_ZnS[0] = 1
X_ZnO = [None for i in range(t+1)]
X_ZnO[0] = 0
X_time = np.zeros(t+1)
X_time[0] = 0

for i in range(t):
    R_rx = (1/(k*b))
    R_film = ((r_c**2)/(r_0**2*kg))
    R_ash = ((r_c*(r_0-r_c))/(r_0*D_eff))
    drc_dt = -(((a*c_O2_b)/(phi_ZnS*b))/(R_rx + R_film + R_ash))
    r_c = r_c + drc_dt*dt
    if r_c < 0:
        r_c = 0
    #conversion profile:
    n_ZnS_SCM = ((4/3)*(np.pi)*r_c**3)*phi_ZnS
    n_ZnS_SCM_0 = ((4/3)*(np.pi)*r_0**3)*phi_ZnS
    c_ZnS_SCM = n_ZnS_SCM/(((4/3)*(np.pi)*r_0**3))
    c_ZnS_SCM_0 = n_ZnS_SCM_0/(((4/3)*(np.pi)*r_0**3))

    X_ZnS[i+1] = c_ZnS_SCM/c_ZnS_SCM_0
    X_ZnO[i+1] = 1 - X_ZnS[i+1]
    X_time[i+1] = X_time[i] + 1

#plot:
fig = plt.figure()
plt.plot(X_time,X_ZnO, 'b')
plt.xlabel('time [s]')
plt.ylabel('X_ZnO')
plt.axis([0, t, 0, 1])
plt.title('Conversion profile of ZnO (SCM)')

P = format((time.time()-start))
```

Listing B.1: SCM script (Python)

Appendix C

Continuous Model Script, FTCS, Example Case (Python)

This script is used for the calculation of the continuous model using the FTCS method for the example case.

```python
import time
import numpy as np
import matplotlib.pyplot as plt
start = time.time()
"""
Input:
"""
T0_b = 900 #C (bulk gas temperature)
P_O2_atm_b = 0.8 #atm (bulk partial pressure of O2)
D_mix = 0.00022812 #m^2/s (diffusion coefficient of the binary pair O2-N2)
kg = 0.71032 #m/s (mass transfer coefficient)
r_0 = 0.001 #m (initial particle radius)

t = 20 #duration in seconds

#FTCS resolution:
steps_time = t*1000000
delta_t = t/steps_time

steps_radius = 10
delta_r = r_0/steps_radius

"""
CM calculations:
```

```python
"""
#constants:
T0_K_b = T0_b + 273.15 #K
R_g = 8.3145 #Universal gas constant (m3*Pa/K/mol)
P_STP_bar = 1.01325 #1atm in bar
phi_ZnS = 41300 #mol/m3
MW_ZnS = 97.474 #g/mol
rho_ZnS = 4090 #kg/m3

#(2ZnS + 3O2 = 2ZnO + 2SO2)*:
a = 2
b = 3
c = 2
d = 2

k_0_r1 = 23000.00 #m/s (pre-exponential constant for the reaction*)
E_a_r1 = 121032.00 #J/mol (activation energy for the reaction*)

#porosity:
epsilon = 1 - (phi_ZnS*((MW_ZnS*0.001)/rho_ZnS))

#tortuosity:
tau = 1

#molecular weights:
MW_O2 = 31.999 #g/mol
MW_N2 = 28.02 #g/mol

#special atomic diffusion volumes:
V_O2 = 16.6
V_N2 = 17.9

#pressure to concentration (O2):
P_O2_Pa_b = P_O2_atm_b*101325 #Pa
c_O2_b = P_O2_Pa_b/(R_g*T0_K_b) #mol O2/m3

#initial vectors:
D_eff = np.zeros(steps_radius+1)

k = np.zeros(steps_radius+1)

c_O2_0 = np.zeros(steps_radius+1)
```

```

c_ZnS_0 = np.zeros(steps_radius+1)
for i in range(len(c_ZnS_0)):
    c_ZnS_0[i] = phi_ZnS

r = np.zeros(steps_radius+1)
for i in range(steps_radius):
    r[i+1] = r[i] + delta_r

T = np.zeros(steps_radius+1) #initial particle temperature profile
for i in range(len(T)):
    T[i] = T0_b + 273.15

#stability test:
D_eff_test = D_mix*(epsilon/tau)
s = (D_eff_test/epsilon)*(delta_t/delta_r**2)

"""
FTCS time-step loop:
"""
#concentration vectors updated per time step:
c_ZnS = np.copy(c_ZnS_0)
c_O2 = np.copy(c_O2_0)

#conversion vectors/initial counters:
X_ZnS = [None for i in range(steps_time)]
X_ZnO = [None for i in range(steps_time)]

for j in range(steps_time):

    """
    Reaction and diffusion related calculations:
    """
    #rate calculations:
    for i in range(len(k)):
        k[i] = k_0_r1*np.exp((-E_a_r1)/(R_g*T[i]))
    R = np.zeros(len(c_ZnS))
    for i in range(len(c_ZnS)):
        X = (1-(c_ZnS[i]/c_ZnS_0[i]))
        f_X = (1-X)**(2/3)
        R[i] = k[i]*f_X*c_O2[i]

    #effective diffusion coefficient:
    MW_O2_N2 = 2*((1/MW_O2)+(1/MW_N2))**(-1)
    for i in range(len(D_eff)):
        D_eff[i] = ((epsilon/tau)*(((0.00143*(T[i]**1.75))/(P_STP_bar*
             (MW_O2_N2**0.5)*(V_O2**(1/3)+V_N2**(1/3))**(2)))*0.0001)) #m2/
    s

    """
    Concentration calculations:
    """
    #boundary concentration of O2 at r=r0:
    c_O2[len(c_O2)-1] = ((kg*c_O2_b + c_O2[len(c_O2)-2]*(D_eff[len(c_O2)
    -1]/
        delta_r))/(D_eff[len(c_O2)-1]/delta_r + kg))

    #boundary concentration of ZnS at r=r0:
    a0 = ((4/3)*(np.pi)*r[len(R)-1]**3)/((4)*(np.pi)*r[len(R)-1]**2) #m

    if (a*delta_t*(1/a0)*R[len(R)-1]) >= c_ZnS[len(c_ZnS)-1]:
        c_ZnS[len(c_ZnS)-1] = 0
    else:
        c_ZnS[len(c_ZnS)-1] = ((-a*delta_t*(1/a0)*R[len(R)-1]) +
             c_ZnS[len(c_ZnS)-1])

    #concentration profile (solid and gas) calculations inside particle:
    for n in range(len(c_ZnS)-2):
        a0 = ((4/3)*(np.pi)*r[n+1]**3)/((4)*(np.pi)*r[n+1]**2) #m

        c_O2[n+1] = ((delta_t/epsilon)*((D_eff[n+1]/r[n+1]**2)*((((c_O2[n
    +2]-
            c_O2[n+1])*r[n+2]**2)-((c_O2[n+1]-c_O2[n])*r[n]**2))/
            (2*delta_r**2))- b*(1/a0)*R[n+1]) + c_O2[n+1])
        c_O2[0] = c_O2[1]
        if (a*delta_t*(1/a0)*R[n+1]) >= c_ZnS[n+1]:
            c_ZnS[n+1] = 0
        else:
            c_ZnS[n+1] = (-a*delta_t*(1/a0)*R[n+1]) + c_ZnS[n+1]
        c_ZnS[0] = c_ZnS[1]

    #conversion profile:
    int_c_ZnS = 0
    for i in range(len(c_ZnS)-1):
        int_c_ZnS = ((((4/3)*np.pi*r[i+1]**3) -
                        ((4/3)*np.pi*r[i]**3))*c_ZnS[i+1] + int_c_ZnS)

    n_ZnS_CM = int_c_ZnS
    c_ZnS_CM = (n_ZnS_CM/(((4/3)*(np.pi)*r_0**3)))

    n_ZnS_CM_0 = ((4/3)*(np.pi)*r_0**3)*phi_ZnS
    c_ZnS_CM_0 = (n_ZnS_CM_0/(((4/3)*(np.pi)*r_0**3)))

    X_ZnS[j] = c_ZnS_CM/c_ZnS_CM_0
    X_ZnO[j] = 1 - X_ZnS[j]

#conversion profile, cont.:
X_ZnO_sec = X_ZnO[::1000000] #[::steps_time/t]
X_ZnO_sec[0] = 0
X_ZnO_sec_V2 = np.append(X_ZnO_sec,X_ZnO[len(X_ZnO)-1])
X_time = np.arange(t+1)

#dimensionless form:
c_ZnS_0dim = np.zeros(len(c_ZnS))
for i in range(len(c_ZnS)):
    c_ZnS_0dim[i] = c_ZnS[i]/c_ZnS_0[i]

c_O2_0dim = np.zeros(len(c_ZnS))
for i in range(len(c_ZnS)):
    c_O2_0dim[i] = c_O2[i]/c_O2_b

r_0dim = np.zeros(len(c_ZnS))
for i in range(len(c_ZnS)):
    r_0dim[i] = r[i]/r_0

#plot:
fig = plt.figure()
plt.plot(r_0dim,c_ZnS_0dim, 'b+')
plt.xlabel('r/r0')
plt.ylabel('c_ZnS/c_ZnS0')
plt.axis([0, 1, 0, 1])
plt.title('Concentration profile of ZnS (CM)')
plt.savefig('Concentration_profile_of_ZnS_(CM)_350')

fig = plt.figure()
plt.plot(r_0dim,c_O2_0dim, 'r+')
plt.xlabel('r/r0')
plt.ylabel('c_O2/c_O2_bulk')
plt.axvline(x=(0.00086760833945/r_0))
plt.axis([0, 1, 0, 1])
plt.title('Concentration profile of O2 (CM)')
plt.savefig('Concentration_profile_of_O2_(CM)_350')

fig = plt.figure()
plt.plot(r_0dim,T, 'b+')
plt.xlabel('r/r0')
plt.ylabel('T [K]')
plt.title('Temperature profile (CM)')
plt.savefig('Temperature_profile_(CM)_350')

fig = plt.figure()
plt.plot(X_time,X_ZnO_sec_V2, 'b')
plt.xlabel('time [s]')
plt.ylabel('X')
plt.axis([0, t, 0, 1])
plt.title('Conversion profile of ZnO (CM)')
plt.savefig('Conversion_profile_of_ZnO_(CM)_350')

P = format((time.time()-start))
```

Listing C.1: CM script using the FTCS method (Python)

Appendix D

Continuous Model Script, OC, Example Case (Matlab)

This script is used for the calculation of the continuous model using the orthogonal collocation method for the example case. The needed function doing all the main calculations are given as well. For the model to work, both Script D.1 and Function D.2 need to be located inside the Matlab current folder.

```matlab
clear variables
tic

%Input:
rp = 1e-3;       %Particle radius, m
hg = 128.65;     %Heat transfer coefficient, W/m2/K
kg1 = 0.71032;   %Mass transfer coefficient, m/s
Tb = 1173.15;    %Bulk temperature, K
T0 = 298.15;     %Initial particle temperature, K
%T0 = Tb;
tend = 250;      %Ending time of simulation

Vfracg1 = 0.8;   %Volume fraction of O2 in bulk

rc   = rp*[0.36311746 0.67718628 0.89975799]; %Vector containing inner
    collocation points

Amat = (1/rp)*[-4.1308947 6.8819128 -4.5475385 1.7965206;
        -1.3388630 -2.2150478 5.2890548 -1.7351440;
        0.62570332 -3.7406161 -1.6671150 4.7820278;
        -1.0727282 5.3255695 -20.752841 33/2]; %First derivative
```

```
Bmat = (1/rp.^2)*[-23.853065 30.593651 -9.74629544 3.0057072;
        11.099906 -43.237662 40.818768 -8.6810122;
        -3.3228457 38.356814 -125.40927 90.375305;
        -33.675598 152.37521 -311.19961 385/2]; %Laplacian

vmat = [-2 0 0;
        -3 0 0;
         2 0 0;
         2 0 0]; %Stoichiometric coefficient matrix

%Initial conditions:
Tp0 =  ones(length(rc),1)*T0;
cg10 = zeros(length(rc),1); %O2
cs10 = ones(length(rc)+1,1)*41300; %ZnS

y0 = [Tp0(:);cg10(:);cs10(:)];

%Solver:
[t,y] = ode15s(@(t,y) model(t,y,cs10,rp,hg,kg1,Tb,Vfracg1,rc,...
    Amat,Bmat,vmat), [0 tend], y0);

%Plot:
%Data needed to make dimensionless plots:
Rg = 8.3145;
rplot = [rc/rp,1];
Pcg1bp = Vfracg1*101325; %Pa
cg1b = Pcg1bp/(Rg*Tb); %mol O2/m3
cg1plot = y(length(y),4:6)/cg1b;
cs1plot = y(length(y),7:10)/41300;

figure
plot(rc/rp,cg1plot,'r--o')
title('Concentration profile of O_2 (OC)')
xlabel('r/r_0')
ylabel('c_{O2}/c_{O2}_{,bulk}')
axis([0 1 0 1])

figure
plot(rplot,cs1plot,'b--o')
title('Concentration profile of ZnS (OC)')
xlabel('r/r_0')
ylabel('c_{ZnS}/c_{ZnS}_0')
```

```
axis([0 1 0 1])

figure
plot(rc/rp,y(length(y),1:3),'--o')
title('Temperature profile (OC)')
xlabel('r/r_0')
ylabel('T [K]')
xlim([0 1])

%Conversion profile:
W = [0.04567809, 0.12589839, 0.13397916, 1/36];
XZnO = zeros(length(y),1);
for i = 1:length(y)
    cZnStot = dot(y(i,7:10),W);
    XZnS = cZnStot/(dot(y(1,7:10),W));
    XZnO(i) = 1 - XZnS;
end

%Import data from python:
%SCM = importdata('Data.txt');
%tSCM = [0:1:250];
%CMFTCS = importdata('Data2.txt');
%tFTCS = [0:1:250];
%CMOCtemp = importdata('Data3.txt');
%tCMOCtemp = importdata('Data3_time.txt');

figure
box on
hold on
%title('Conversion profile of ZnO (SCM, CM-FTCS, CM-OC)')
title('Conversion profile of ZnO (OC)')
xlabel('time [s]')
ylabel('X_{ZnO}')
%plot(tSCM,SCM,'k--')
%plot(tFTCS,CMFTCS,'k:')
plot(t,XZnO,'k-.')
%plot(tCMOCtemp,CMOCtemp,'k-')
legend('CM-OC, N = 3, with temperature')
%legend('CM-OC, N = 3, without temperature','CM-OC, N = 3, with temperature
    ')
%legend('SCM','CM-FTCS','CM-OC, N = 3, without temperature')
axis([0 tend 0 1])
```

```

figure
box on
title('Conversion profile of ZnS (OC)')
xlabel('time [s]')
ylabel('X_{ZnS}')
hold on
plot(t,y(:,10)/41300) % Plot surface
plot(t,y(:,9)/41300) % Plot outermost collocation point
plot(t,y(:,8)/41300,'g') % Plot middle collocation point
plot(t,y(:,7)/41300,'r') %Plot inner collocation point
legend('r = 1, (surface)','r = 0.89975799','r = 0.67718628','r = 0.36311746
    ')
axis([0 tend 0 1])

figure
box on
title('Temperature particle')
xlabel('time [s]')
ylabel('T[K]')
hold on
plot(t,y(:,3)) % Plot outermost collocation point
plot(t,y(:,2),'g') % Plot middle collocation point
plot(t,y(:,1),'r') %Plot inner collocation point
legend('r = 0.89975799','r = 0.67718628','r = 0.36311746')
xlim([0 tend])

toc
```

Listing D.1: CM script using the OC method (Matlab)

```
function [dydt,cg1surf,Tsurf] = model(t,y,cs10,rp,hg,kg1,Tb,Vfracg1,rc,Amat
    ,Bmat,vmat)

for i = 1:4
    if y(i+6) <= 0
        y(i+6) = 0;
    end
end

%Input to correct vectors:
T = y(1:3);
cg1 = y(4:6);
```

```
cs1 = y(7:9);
cs1surf = y(10);

%Data:
phiZnS = 41300;          %Particle molar density, mol/m3
epsilonZnS = 0.015727;   %Porosity of ZnS
tauZnS = 1;              %Tortuosity of ZnS
keff = 24.705;           %Heat conductivity, W/m/K
Rg = 8.3145;             %Universal gas constant, m3*Pa/K/mol
PSTPbar = 1.01325;       %1atm in bar

MWO2 = 31.999;           %Molecular weight of O2, g/mol
MWN2 = 28.02;            %Molecular weight of O2, g/mol

VO2 = 16.6;              %Special atomic diffusion volume of O2
VN2 = 17.9;              %Special atomic diffusion volume of N2

cpZnS = 45.358;          %Particle heat capacity, J/mol/K
cpO2 = 29.39;            %O2 heat capacity, J/mol/K
cpZnO = 41.086;          %ZnO heat capacity, J/mol/K
cpSO2 = 39.94;           %SO2 heat capacity, J/mol/K

k0r1 = 23000.00;         %Pre-exponential constant for rx1, m/s
Ear1 = 121032.00;        %Activation energy for rx1, J/mol

%Volume fraction to concentration:
Pcg1bp = Vfracg1*101325; %Pa
cg1b = Pcg1bp/(Rg*Tb); %mol O2/m3

%Calculates the enthalpy of reaction:
deltacprx1 = vmat(1)*cpZnS + vmat(2)*cpO2 + vmat(3)*cpZnO + vmat(4)*cpSO2;

deltaHrx1s = -878260; %J/mol

deltaHrx1T = zeros(length(rc),1);
for i = 1:3
    deltaHrx1T(i,1) = (T(i)*deltacprx1 - 298.15*deltacprx1) + deltaHrx1s;
end

%Ratio between surface area and volume:
a0 = ((4/3)*pi*rp^3)/(4*pi*rp^2);
a0surf = a0;
```

```

%Rate constant calculations:
R = zeros(length(rc),1);
for i = 1:3
    k = k0r1*exp((-Ear1)/(Rg*T(i)));
    fX = (1-(1-(cs1(i)/cs10(i))))^(2/3);
    R(i,1) = k*fX*cg1(i);
end

%Effective diffusion coefficient of specie1:
MWO2N2 = 2*(1/MWO2 + 1/MWN2)^(-1);
Deff = zeros(length(rc),1);
for i = 1:3
    Deff(i,1) = (epsilonZnS/tauZnS)*(((0.00143*(T(i)^1.75))/(PSTPbar*...
        (MWO2N2^0.5)*(VO2^(1/3)+VN2^(1/3))^(2)))*0.0001);
end

%Surface:
Tsurf = (hg*Tb-keff*(Amat(4,1:3)*T(1:3)))/(keff*Amat(4,4)+hg);
%Tsurf = T(3);

Deffsurf = (epsilonZnS/tauZnS)*(((0.00143*(Tsurf^1.75))/(PSTPbar*...
    (MWO2N2^0.5)*(VO2^(1/3)+VN2^(1/3))^(2)))*0.0001);
cg1surf = (kg1*cg1b - Deffsurf*(Amat(4,1:3)*cg1(1:3)))/...
    (Deffsurf*Amat(4,4)+kg1);

ksurf = k0r1*exp((-Ear1)/(Rg*Tsurf));
fXsurf = (1-(1-(cs1surf/cs10(4))))^(2/3);
Rsurf = ksurf*fXsurf*cg1surf;
dcs1dtsurf = vmat(1)*(1/a0surf)*Rsurf;

%Internal:
dTdt = zeros(length(rc),1);
for i = 1:3
   dTdt(i,1) = 1/(phiZnS*cpZnS)*(keff*(Bmat(i,1:3) * T(1:3) + ...
       Bmat(i,4)*Tsurf) + -deltaHrx1T(i)*(1/a0)*R(i));
   %dTdt(i,1) = 0;
end

dcg1dt = zeros(length(rc),1);
for i = 1:3
   dcg1dt(i,1) = (1/epsilonZnS)*(Deff(i)*(Bmat(i,1:3) * cg1(1:3) + ...
       Bmat(i,4)*cg1surf) + vmat(2)*(1/a0)*R(i));
end

dcs1dtint = zeros(length(rc),1);
for i = 1:3
    dcs1dtint(i,1) = vmat(1)*(1/a0)*R(i);
end

dcs1dt = [dcs1dtint;dcs1dtsurf];
dydt = [dTdt(:);dcg1dt(:);dcs1dt(:)];
end
```

Listing D.2: CM function doing all the calculations related to gas-solid reactions

Appendix E

The Prereduction Model (Matlab)

The main script, Script E.1, calculates the particle weight loss, conversion profiles, concentration profiles, partial pressures and temperature profiles for the relevant species present during reduction of manganese oxides using blends of CO, CO_2, H_2 and H_2O. Function E.2 does all the important calculations regarding gas-solid reactions. For the model to work, both Script E.1 and Function E.2 need to be located inside the Matlab current folder.

```
clear variables
tic

%Input:
rp = 5e-3;          %Particle radius, m
Tb = 1600;          %Bulk temperature, K
%T0 = 300;          %Initial particle temperature, K
T0 = Tb;
vb = 0.1;           %Bulk velocity, m/s
tend = 20000;       %Ending time of simulation, s

VCOfrac = 0.99;  %Volume fraction of CO in bulk
VCO2frac = 0.01; %Volume fraction of CO2 in bulk
VH2frac = 0.00;  %Volume fraction of H2 in bulk

rc   = rp*[0.36311746 0.67718628 0.89975799]; %Vector containing inner
    collocation points

Amat = (1/rp)*[-4.1308947 6.8819128 -4.5475385 1.7965206;
        -1.3388630 -2.2150478 5.2890548 -1.7351440;
        0.62570332 -3.7406161 -1.6671150 4.7820278;
        -1.0727282 5.3255695 -20.752841 33/2]; %First derivative
Bmat = (1/rp.^2)*[-23.853065 30.593651 -9.74629544 3.0057072;
        11.099906 -43.237662 40.818768 -8.6810122;
        -3.3228457 38.356814 -125.40927 90.375305;
        -33.675598 152.37521 -311.19961 385/2]; %Laplacian

vmat = [-1.0 -0.5 -1/3 -1;
        -0.5 -1/6 -1/3 -1;
         0.5 1/3  1.0 1;
         0.5 1/6  1/3 1]; %Stoichiometric coefficient matrix

kgCO = masstransferfun(1,Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);
kgH2 = masstransferfun(2,Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);
kgCO2 = masstransferfun(3,Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);
kgH2O = masstransferfun(4,Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);

hg = heattransferfun(Tb-273.15,rp*2,vb,VCOfrac,VCO2frac,VH2frac);
%hg = 0;

%Volume fraction to concentration (bulk)
Rg = 8.3145; %Universal gas constant, m3*Pa/K/mol
PCObp = VCOfrac*101325; %Pa
cCOb = PCObp/(Rg*Tb); %mol CO/m3
PH2bp = VH2frac*101325; %Pa
cH2b = PH2bp/(Rg*Tb); %mol H2/m3
PCO2bp = VCO2frac*101325; %Pa
cCO2b = PCO2bp/(Rg*Tb); %mol CO2/m3
PH2Obp = (1- (VCO2frac + VH2frac + VCOfrac))*101325; %Pa
cH2Ob = PH2Obp/(Rg*Tb); %mol H2O/m3

%Initial conditions:
Tp0 =  ones(length(rc),1)*T0; %K
cMnO20 = ones(length(rc)+1,1)*57858; %mol/m3
cMn2O30 = ones(length(rc)+1,1)*0; %mol/m3
cMn3O40 = ones(length(rc)+1,1)*0; %mol/m3
cMnO0 = ones(length(rc)+1,1)*0; %mol/m3
cCO0 = ones(length(rc),1)*cCOb; %mol/m3
cH20 = ones(length(rc),1)*cH2b; %mol/m3
cCO20 = ones(length(rc),1)*cCO2b; %mol/m3
cH2O0 = ones(length(rc),1)*cH2Ob; %mol/m3

```

```
y0 = [Tp0(:);cCO0(:);cMnO20(:);cMn2O30(:);cMn3O40(:);cMnO0(:);cH20(:);cCO20
    (:);cH2O0(:)];

%Solver:
[t,y] = ode15s(@(t,y) modelcomp(t,y,rp,hg,kgCO,kgH2,kgCO2,kgH2O,Tb,VCOfrac
    ,...
    VCO2frac,VH2frac,rc,Amat,Bmat,vmat), [0 tend], y0);

%Plot:
%Adding surface temperature to our plot:
for i = 1:length(y)
    phiMnO2 = 57858;
    MWMnO2 = 86.9368;
    rhoMnO2 = 5030;
    epsilon = 1 - (phiMnO2*((MWMnO2*0.001)/rhoMnO2));
    kMnO2 = 3.5;
    keff = kMnO2*(1 - epsilon);
    Tadd = y(i,1:3);
    Tsurfadd = (hg*Tb-keff*(dot(Amat(4,1:3),Tadd(1:3))))/...
        (keff*Amat(4,4)+hg);
    Tplot = [y(i,1:3),Tsurfadd];
end
rplot = [rc/rp,1];

%Data needed to make dimensionless plots:
cCOplot = y(length(y),4:6)/cCOb;
cMnO2plot = y(length(y),7:10)/57858;
cMn2O3plot = y(length(y),11:14)/28503;
cH2plot = y(length(y),23:25)/cH2b;

figure
plot(rplot,Tplot,'--o')
title('Temperature profile')
xlabel('r/r_0')
ylabel('T [K]')
xlim([0 1])

figure
plot(rc/rp,cCOplot,'r--o')
title('Concentration profile of CO')
xlabel('r/r_0')
ylabel('c_{CO}/c_{CO}_{,bulk}')
```

```
axis([0 1 0 1])

figure
plot(rc/rp,cH2plot,'r--o')
title('Concentration profile of H_2')
xlabel('r/r_0')
ylabel('c_{H2}/c_{H2}_{,bulk}')
axis([0 1 0 1])

figure
plot(rplot,cMnO2plot,'b--o')
title('Concentration profile of MnO_2')
xlabel('r/r_0')
ylabel('c_{MnO2}/c_{MnO2}_0')
axis([0 1 0 1])

f = figure;
p = uipanel('Parent',f,'BorderType','none');
p.Title = 'Conversion profiles of Mn-oxides';
p.TitlePosition = 'centertop';
p.FontSize = 12;
p.FontWeight = 'bold';

subplot(2,2,1,'Parent',p)
title('Conversion profile of MnO_2')
xlabel('time [h]')
ylabel('X_{MnO2}')
hold on
box on
plot(t/3600,y(:,10)/57858) % Plot surface
plot(t/3600,y(:,9)/57858) % Plot outermost collocation point
plot(t/3600,y(:,8)/57858,'g') % Plot middle collocation point
plot(t/3600,y(:,7)/57858,'r') %Plot inner collocation point
legend('r = 1, (surface)','r = 0.89975799','r = 0.67718628','r = 0.36311746
    ')
axis([0 tend/3600 0 1])

subplot(2,2,2,'Parent',p)
title('Conversion profile of Mn_2O_3')
xlabel('time [h]')
ylabel('X_{Mn2O3}')
hold on
```

```
box on
plot(t/3600,y(:,14)/28929) % Plot surface
plot(t/3600,y(:,13)/28929) % Plot outermost collocation point
plot(t/3600,y(:,12)/28929,'g') % Plot middle collocation point
plot(t/3600,y(:,11)/28929,'r') %Plot inner collocation point
axis([0 tend/3600 0 1])

subplot(2,2,3,'Parent',p)
title('Conversion profile of Mn_3O_4')
xlabel('time [h]')
ylabel('X_{Mn3O4}')
hold on
box on
plot(t/3600,y(:,18)/19286) % Plot surface
plot(t/3600,y(:,17)/19286) % Plot outermost collocation point
plot(t/3600,y(:,16)/19286,'g') % Plot middle collocation point
plot(t/3600,y(:,15)/19286,'r') %Plot inner collocation point
axis([0 tend/3600 0 1])

subplot(2,2,4,'Parent',p)
title('Conversion profile of MnO')
xlabel('time [h]')
ylabel('X_{MnO}')
hold on
box on
plot(t/3600,y(:,22)/57858) % Plot surface
plot(t/3600,y(:,21)/57858) % Plot outermost collocation point
plot(t/3600,y(:,20)/57858,'g') % Plot middle collocation point
plot(t/3600,y(:,19)/57858,'r') %Plot inner collocation point
axis([0 tend/3600 0 1])

figure
title('Particle temperature')
xlabel('time [h]')
ylabel('T[K]')
hold on
box on
plot(t/3600,y(:,3)) % Plot outermost collocation point
plot(t/3600,y(:,2),'g') % Plot middle collocation point
plot(t/3600,y(:,1),'r') %Plot inner collocation point
legend('r = 0.89975799','r = 0.67718628','r = 0.36311746')
xlim([0 tend/3600])
```

```

f = figure;
p = uipanel('Parent',f,'BorderType','none');
p.Title = 'Partial pressure of gas species';
p.TitlePosition = 'centertop';
p.FontSize = 12;
p.FontWeight = 'bold';

subplot(2,2,1,'Parent',p)
title('CO')
xlabel('time [h]')
ylabel('p_{CO} [atm]')
hold on
box on
plot(t/3600,y(:,6)*Rg.*y(:,3)*(9.86923267e-6)) % Plot outermost collocation
     point
plot(t/3600,y(:,5)*Rg.*y(:,2)*(9.86923267e-6),'g') % Plot middle
    collocation point
plot(t/3600,y(:,4)*Rg.*y(:,1)*(9.86923267e-6),'r') %Plot inner collocation
    point
legend('r = 0.89975799','r = 0.67718628','r = 0.36311746')
xlim([0 tend/3600])
ylim([0 2])

subplot(2,2,2,'Parent',p)
title('H_2')
xlabel('time [h]')
ylabel('p_{H2} [atm]')
hold on
box on
plot(t/3600,y(:,25)*Rg.*y(:,3)*(9.86923267e-6)) % Plot outermost
    collocation point
plot(t/3600,y(:,24)*Rg.*y(:,2)*(9.86923267e-6),'g') % Plot middle
    collocation point
plot(t/3600,y(:,23)*Rg.*y(:,1)*(9.86923267e-6),'r') %Plot inner collocation
     point
xlim([0 tend/3600])
ylim([0 1.5])

subplot(2,2,3,'Parent',p)
title('CO_2')
xlabel('time [h]')
```

```
ylabel('p_{CO2} [atm]')
hold on
box on
plot(t/3600,y(:,28)*Rg.*y(:,3)*(9.86923267e-6)) % Plot outermost
    collocation point
plot(t/3600,y(:,27)*Rg.*y(:,2)*(9.86923267e-6),'g') % Plot middle
    collocation point
plot(t/3600,y(:,26)*Rg.*y(:,1)*(9.86923267e-6),'r') %Plot inner collocation
     point
xlim([0 tend/3600])
ylim([0 1.5])

subplot(2,2,4,'Parent',p)
title('H_2O')
xlabel('time [h]')
ylabel('p_{H2O} [atm]')
hold on
box on
plot(t/3600,y(:,31)*Rg.*y(:,3)*(9.86923267e-6)) % Plot outermost
    collocation point
plot(t/3600,y(:,30)*Rg.*y(:,2)*(9.86923267e-6),'g') % Plot middle
    collocation point
plot(t/3600,y(:,29)*Rg.*y(:,1)*(9.86923267e-6),'r') %Plot inner collocation
     point
xlim([0 tend/3600])
ylim([0 1.5])

%Weightloss:
W = [0.04567809, 0.12589839, 0.13397916, 1/36];
m = zeros(length(y),1);
for j = 1:length(y)
    nMnO2CM = dot(y(j,7:10),W)*(((4/3)*pi*rp^3));
    nMn2O3CM = dot(y(j,11:14),W)*(((4/3)*pi*rp^3));
    nMn3O4CM = dot(y(j,15:18),W)*(((4/3)*pi*rp^3));
    nMnOCM = dot(y(j,19:22),W)*(((4/3)*pi*rp^3));
    m(j) = (nMnO2CM*86.9368 + nMn2O3CM*157.8743 + nMn3O4CM*228.812 + nMnOCM
    *70.937449);
end

mloss = zeros(length(y),1);
for j = 1:(length(y)-1)
    mloss(j+1) = (1-(m(j+1)/m(1))); %corrected
```

```matlab
end

figure
plot(t/3600,mloss*-100,'b')
title('Particle mass change (%)')
xlabel('time [h]')
ylabel('mass change [%]')
xlim([0 tend/3600])
%ylim([0 25])

toc
timehours = tend/(60*60);
disp('hours:')
disp(timehours)
```

Listing E.1: The prereduction model (Matlab)

```matlab
function [dydt,cCOsurf,Tsurf,cH2surf,cCO2surf,cH2Osurf] = ...
    modelcomp(t,y,rp,hg,kgCO,kgH2,kgCO2,kgH2O,Tb,VCOfrac,VCO2frac,VH2frac,
   rc,Amat,Bmat,vmat)

for i = 1:16
    if y(i+6) <= 0
        y(i+6) = 0;
    end
end

%Input to correct vectors:
Tp = y(1:3);
cCO = y(4:6);
cMnO2 = y(7:9);
cMnO2surf = y(10);
cMn2O3 = y(11:13);
cMn2O3surf = y(14);
cMn3O4 = y(15:17);
cMn3O4surf = y(18);
cMnO = y(19:21);
cMnOsurf = y(22);
cH2 = y(23:25);
cCO2 = y(26:28);
cH2O = y(29:31);

%Data:
```

```
phiMnO2 = 57858;        %Molar density of MnO2, mol/m3
phiMn2O3 = 28503;       %Molar density of Mn2O3, mol/m3
phiMn3O4 = 21240;       %Molar density of Mn3O4, mol/m3
phiMnO = 75700;         %Molar density of MnO, mol/m3
Rg = 8.3145;            %Universal gas constant, m3*Pa/K/mol
PSTPbar = 1.01325;      %1atm in bar
PSTPatm = 1;            %atm
phicat = 60;

MWMnO2 = 86.9368;       %Molecular weight of MnO2, g/mol
rhoMnO2 = 5030;         %Density of MnO2, kg/m3

MWCO = 28.01;           %Molecular weight of CO, g/mol
MWCO2 = 44.010;         %Molecular weight of CO2, g/mol
MWH2 = 2.016;           %Molecular weight of H2, g/mol
MWH2O = 18.01528;       %Molecular weight of H2O, g/mol

VCO = 18.9;             %Special atomic diffusion volume of CO
VCO2 = 26.9;            %Special atomic diffusion volume of CO2
VH2 = 7.07;             %Special atomic diffusion volume of H2
VH2O = 12.7;            %Special atomic diffusion volume of H2O

cpMnO2 = 54;            %MnO2 heat capacity, J/mol/K
cpMn2O3 = 108;          %Mn2O3 heat capacity, J/mol/K
cpMn3O4 = 140;          %Mn3O4 heat capacity, J/mol/K
cpMnO = 45;             %MnO heat capacity, J/mol/K

deltaHrx1 = -99900;     %J/mol rx1: (MnO2 + 1/2CO = 1/2Mn2O3 + 1/2CO2)
deltaHrx2 = -31300;     %J/mol rx2: (1/2Mn2O3 + 1/6CO = 1/3Mn3O4 + 1/6CO2)
deltaHrx3 = -16900;     %J/mol rx3: (1/3Mn3O4 + 1/3CO = MnO + 1/3CO2)

deltaHrx4 = -80342;     %J/mol rx1: (MnO2 + 1/2H2 = 1/2Mn2O3 + 1/2H2O)
deltaHrx5 = -22223;     %J/mol rx2: (1/2Mn2O3 + 1/6H2 = 1/3Mn3O4 + 1/6H2O)
deltaHrx6 = -4411;      %J/mol rx3: (1/3Mn3O4 + 1/3H2 = MnO + 1/3H2O)

deltaHWGSR = -41090;    %J/mol WGSR: (CO + H2O <=> CO2 + H2)

k0rx1 = 23.00;          %Pre-exponential constant for rx1, m/s
Earx1 = 121032.00;      %Activation energy for rx1, J/mol

k0rx2 = 3250;           %Pre-exponential constant for rx2, m/s
Earx2 = 208832.00;      %Activation energy for rx2, J/mol
```

```

k0rx3 = 3250;           %Pre-exponential constant for rx3, m/s
Earx3 = 208832.00;      %Activation energy for rx3, J/mol

k0rx4 = 23.00;          %Pre-exponential constant for rx4, m/s
Earx4 = 121032.00;      %Activation energy for rx4, J/mol

k0rx5 = 3250;           %Pre-exponential constant for rx5, m/s
Earx5 = 208832.00;      %Activation energy for rx5, J/mol

k0rx6 = 3250;           %Pre-exponential constant for rx6, m/s
Earx6 = 208832.00;      %Activation energy for rx6, J/mol

%Porosity:
%epsilon = 1 - (phiMnO2*((MWMnO2*0.001)/rhoMnO2));
epsilon = 0.35; %Pellet/ = 0.05 lump (Lobo)

%Tortuosity:
tau = 1;

%Thermal conductivity of MnO2:
kMnO2 = 10;             %W/m/K (for MnO at 573 K)
keff = kMnO2*(1 - epsilon);

%Calculates the enthalpy of reaction:
%Reaction 1,2,3,4,5 and 6:
deltaHrx1T = zeros(length(rc),1);
deltaHrx2T = zeros(length(rc),1);
deltaHrx3T = zeros(length(rc),1);

deltaHrx4T = zeros(length(rc),1);
deltaHrx5T = zeros(length(rc),1);
deltaHrx6T = zeros(length(rc),1);
for i = 1:3
    cpH2 = (-1e-07*Tp(i)^2 + 0.002*Tp(i) + 13.359)*MWH2; %J/mol/K
    cpCO = (5e-12*Tp(i)^3 - 7e-08*Tp(i)^2 + 0.0003*Tp(i) + 0.9627)*MWCO; %J
    /mol/K
    cpCO2 = (-7e-15*Tp(i)^4 + 1e-10*Tp(i)^3 - 5e-07*Tp(i)^2 + 0.001*Tp(i) +
     0.5775)*MWCO2; %J/mol/K
    cpH2O = (7e-12*Tp(i)^3 - 1e-07*Tp(i)^2 + 0.0008*Tp(i) + 1.6083)*MWH2O;
    %J/mol/K

    deltacprx1 = vmat(1)*cpMnO2 + vmat(2)*cpCO + vmat(3)*cpMn2O3 + vmat(4)*cpCO2;
    deltacprx2 = vmat(1,2)*cpMn2O3 + vmat(2,2)*cpCO + vmat(3,2)*cpMn3O4 + vmat(4,2)*cpCO2;
    deltacprx3 = vmat(1,3)*cpMn3O4 + vmat(2,3)*cpCO + vmat(3,3)*cpMnO + vmat(4,3)*cpCO2;

    deltacprx4 = vmat(1)*cpMnO2 + vmat(2)*cpH2 + vmat(3)*cpMn2O3 + vmat(4)*cpH2O;
    deltacprx5 = vmat(1,2)*cpMn2O3 + vmat(2,2)*cpH2 + vmat(3,2)*cpMn3O4 + vmat(4,2)*cpH2O;
    deltacprx6 = vmat(1,3)*cpMn3O4 + vmat(2,3)*cpH2 + vmat(3,3)*cpMnO + vmat(4,3)*cpH2O;

    deltaHrx1T(i,1) = (Tp(i)*deltacprx1 - 298.15*deltacprx1) + deltaHrx1;
    deltaHrx2T(i,1) = (Tp(i)*deltacprx2 - 298.15*deltacprx2) + deltaHrx2;
    deltaHrx3T(i,1) = (Tp(i)*deltacprx3 - 298.15*deltacprx3) + deltaHrx3;

    deltaHrx4T(i,1) = (Tp(i)*deltacprx4 - 298.15*deltacprx4) + deltaHrx4;
    deltaHrx5T(i,1) = (Tp(i)*deltacprx5 - 298.15*deltacprx5) + deltaHrx5;
    deltaHrx6T(i,1) = (Tp(i)*deltacprx6 - 298.15*deltacprx6) + deltaHrx6;
end

%Ratio between surface area and volume:
a0 = ((4/3)*pi*rp^3)/(4*pi*rp^2);
a0surf = a0;

%Rate constant calculations:
R1 = zeros(length(rc),1);
R2 = zeros(length(rc),1);
R3 = zeros(length(rc),1);

R4 = zeros(length(rc),1);
R5 = zeros(length(rc),1);
R6 = zeros(length(rc),1);

RWGSR = zeros(length(rc),1);
for i = 1:3
    k1 = k0rx1*exp((-Earx1)/(Rg*Tp(i)));
    fX1 = (1-(1-(cMnO2(i)/phiMnO2)))^(2/3);
    R1(i,1) = k1*fX1*cCO(i);

    k2 = k0rx2*exp((-Earx2)/(Rg*Tp(i)));
    fX2 = (1-(1-(cMn2O3(i)/phiMn2O3)))^(2/3);
    R2(i,1) = k2*fX2*cCO(i);

    k3 = k0rx3*exp((-Earx3)/(Rg*Tp(i)));
    fX3 = (1-(1-(cMn3O4(i)/phiMn3O4)))^(2/3);
    R3(i,1) = k3*fX3*cCO(i);

    k4 = k0rx4*exp((-Earx4)/(Rg*Tp(i)));
    fX4 = (1-(1-(cMnO2(i)/phiMnO2)))^(2/3);
    R4(i,1) = k4*fX4*cH2(i);

    k5 = k0rx5*exp((-Earx5)/(Rg*Tp(i)));
    fX5 = (1-(1-(cMn2O3(i)/phiMn2O3)))^(2/3);
    R5(i,1) = k5*fX5*cH2(i);

    k6 = k0rx6*exp((-Earx6)/(Rg*Tp(i)));
    fX6 = (1-(1-(cMn3O4(i)/phiMn3O4)))^(2/3);
    R6(i,1) = k6*fX6*cH2(i);

    %WGSR:
    %%{
    Keq = exp((4577.8/Tp(i))-4.33);
    kWGS = exp((-29364/(1.987*Tp(i)))+(40.32/1.987));
    KCO = exp((3064/(1.987*Tp(i)))-(6.74/1.987));
    KCO2 = exp((12542/(1.987*Tp(i)))-(18.45/1.987));
    KH2O = exp((-6216/(1.987*Tp(i)))-(12.77/1.987));
    PCO = cCO(i)*(Rg*Tp(i));
    PCO2 = cCO2(i)*(Rg*Tp(i));
    PH2O = cH2O(i)*(Rg*Tp(i));
    PH2 = cH2(i)*(Rg*Tp(i));
    RWGSR(i,1) = (((kWGS*KCO*KH2O*(PCO*PH2O-((PCO2*PH2)/Keq)))/(1 + ...
        KCO*PCO + KH2O*PH2O + KCO2*PCO2)^2)*(phicat/60))*(10^6*((4/3)*pi*rp
    ^3)); %mol/s
    %}
    %RWGSR(i,1) = 0;
end

%Volume fraction to concentration:
PCObp = VCOfrac*101325; %Pa
cCOb = PCObp/(Rg*Tb); %mol CO/m3

```

```
PH2bp = VH2frac*101325; %Pa
cH2b = PH2bp/(Rg*Tb); %mol CO/m3

PCO2bp = VCO2frac*101325; %Pa
cCO2b = PCO2bp/(Rg*Tb); %mol CO/m3

PH2Obp = (1- (VCO2frac + VH2frac + VCOfrac))*101325; %Pa
cH2Ob = PH2Obp/(Rg*Tb); %mol CO/m3

%Mole fraction calculations:
XCO = VCOfrac/PSTPatm;
XCO2 = VCO2frac/PSTPatm;
XH2 = VH2frac/PSTPatm;
XH2O = 1 - XCO - XCO2 - XH2;

%Effective diffusion coefficient of CO:
MWCOCO2 = 2*((1/MWCO)+(1/MWCO2))^(-1);
MWCOH2 = 2*((1/MWCO)+(1/MWH2))^(-1);
MWCOH2O = 2*((1/MWCO)+(1/MWH2O))^(-1);
DCOeff = zeros(length(rc),1);
for i = 1:3
    DCOCO2 = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWCOCO2^0.5)*(VCO^(1/3)+...
        VCO2^(1/3))^(2)))*0.0001; %m2/s
    DCOH2 = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWCOH2^0.5)*(VCO^(1/3)+...
        VH2^(1/3))^(2)))*0.0001; %m2/s
    DCOH2O = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWCOH2O^0.5)*(VCO^(1/3)+...
        VH2O^(1/3))^(2)))*0.0001; %m2/s
    DCOeff(i,1) = (epsilon/tau)*((1 - XCO)/(XCO2/DCOCO2 + XH2/DCOH2 + ...
        XH2O/DCOH2O));
end

%Effective diffusion coefficient of H2:
MWH2CO = 2*((1/MWH2)+(1/MWCO))^(-1);
MWH2CO2 = 2*((1/MWH2)+(1/MWCO2))^(-1);
MWH2H2O = 2*((1/MWH2)+(1/MWH2O))^(-1);
DH2eff = zeros(length(rc),1);
for i = 1:3
    DH2CO = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWH2CO^0.5)*(VH2^(1/3)+...
        VCO^(1/3))^(2)))*0.0001; %m2/s
    DH2CO2 = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWH2CO2^0.5)*(VH2^(1/3)+...
        VCO2^(1/3))^(2)))*0.0001; %m2/s
    DH2H2O = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWH2H2O^0.5)*(VH2^(1/3)+...
        VH2O^(1/3))^(2)))*0.0001; %m2/s
    DH2eff(i,1) = (epsilon/tau)*((1 - XH2)/(XCO/DH2CO + XCO2/DH2CO2 + ...
        XH2O/DH2H2O));
end

%Effective diffusion coefficient of CO2:
MWCO2CO = 2*((1/MWCO2)+(1/MWCO))^(-1);
MWCO2H2 = 2*((1/MWCO2)+(1/MWH2))^(-1);
MWCO2H2O = 2*((1/MWCO2)+(1/MWH2O))^(-1);
DCO2eff = zeros(length(rc),1);
for i = 1:3
    DCO2CO = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWCO2CO^0.5)*(VCO2^(1/3)+...
        VCO^(1/3))^(2)))*0.0001; %m2/s
    DCO2H2 = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWCO2H2^0.5)*(VCO2^(1/3)+...
        VH2^(1/3))^(2)))*0.0001; %m2/s
    DCO2H2O = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWCO2H2O^0.5)*(VCO2^(1/3)
    +...
        VH2O^(1/3))^(2)))*0.0001; %m2/s
    DCO2eff(i,1) = (epsilon/tau)*((1 - XCO2)/(XCO/DCO2CO + XH2/DCO2H2 + ...
        XH2O/DCO2H2O));
end

%Effective diffusion coefficient of H2O:
MWH2OCO = 2*((1/MWH2O)+(1/MWCO))^(-1);
MWH2OH2 = 2*((1/MWH2O)+(1/MWH2))^(-1);
MWH2OCO2 = 2*((1/MWH2O)+(1/MWCO2))^(-1);
DH2Oeff = zeros(length(rc),1);
for i = 1:3
    DH2OCO = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWH2OCO^0.5)*(VH2O^(1/3)+...
        VCO^(1/3))^(2)))*0.0001; %m2/s
    DH2OH2 = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWH2OH2^0.5)*(VH2O^(1/3)+...
        VH2^(1/3))^(2)))*0.0001; %m2/s
    DH2OCO2 = ((0.00143*(Tp(i)^1.75))/(PSTPbar*(MWH2OCO2^0.5)*(VH2O^(1/3)
    +...
        VCO2^(1/3))^(2)))*0.0001; %m2/s
    DH2Oeff(i,1) = (epsilon/tau)*((1 - XH2O)/(XCO/DH2OCO + XH2/DH2OH2 + ...
        XCO2/DH2OCO2));
end

%Surface:
Tsurf = (hg*Tb-keff*(Amat(4,1:3)*Tp(1:3)))/(keff*Amat(4,4)+hg);
%Tsurf = Tp(1);
```

```

DCOCO2surf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWCOCO2^0.5)*(VCO^(1/3)+VCO2
    ^(1/3))^(2)))*0.0001;
DCOH2surf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWCOH2^0.5)*(VCO^(1/3)+VH2
    ^(1/3))^(2)))*0.0001;
DCOH2Osurf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWCOH2O^0.5)*(VCO^(1/3)+VH2O
    ^(1/3))^(2)))*0.0001;
DCOeffsurf = (epsilon/tau)*((1 - XCO)/(XCO2/DCOCO2surf + XH2/DCOH2surf +
    XH2O/DCOH2Osurf));
cCOsurf = (kgCO*cCOb - DCOeffsurf*(Amat(4,1:3)*cCO(1:3)))/(DCOeffsurf*Amat
    (4,4)+kgCO);

DH2COsurf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWH2CO^0.5)*(VH2^(1/3)+VCO
    ^(1/3))^(2)))*0.0001;
DH2CO2surf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWH2CO2^0.5)*(VH2^(1/3)+VCO2
    ^(1/3))^(2)))*0.0001;
DH2H2Osurf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWH2H2O^0.5)*(VH2^(1/3)+VH2O
    ^(1/3))^(2)))*0.0001;
DH2effsurf = (epsilon/tau)*((1 - XH2)/(XCO/DH2COsurf + XCO2/DH2CO2surf +
    XH2O/DH2H2Osurf));
cH2surf = (kgH2*cH2b - DH2effsurf*(Amat(4,1:3)*cH2(1:3)))/(DH2effsurf*Amat
    (4,4)+kgH2);

DCO2COsurf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWCO2CO^0.5)*(VCO2^(1/3)+VCO
    ^(1/3))^(2)))*0.0001;
DCO2H2surf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWCO2H2^0.5)*(VCO2^(1/3)+VH2
    ^(1/3))^(2)))*0.0001;
DCO2H2Osurf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWCO2H2O^0.5)*(VCO2^(1/3)+
    VH2O^(1/3))^(2)))*0.0001;
DCO2effsurf = (epsilon/tau)*((1 - XCO2)/(XCO/DCO2COsurf + XH2/DCO2H2surf +
    XH2O/DCO2H2Osurf));
cCO2surf = (kgCO2*cCO2b - DCO2effsurf*(Amat(4,1:3)*cCO2(1:3)))/(DCO2effsurf
    *Amat(4,4)+kgCO2);

DH2OCOsurf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWH2OCO^0.5)*(VH2O^(1/3)+VCO
    ^(1/3))^(2)))*0.0001;
DH2OH2surf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWH2OH2^0.5)*(VH2O^(1/3)+VH2
    ^(1/3))^(2)))*0.0001;
DH2OCO2surf = ((0.00143*(Tsurf^1.75))/(PSTPbar*(MWH2OCO2^0.5)*(VH2O^(1/3)+
    VCO2^(1/3))^(2)))*0.0001;
DH2Oeffsurf = (epsilon/tau)*((1 - XH2O)/(XCO/DH2OCOsurf + XH2/DH2OH2surf +
    XCO2/DH2OCO2surf));
```

```
cH2Osurf = (kgH2O*cH2Ob - DH2Oeffsurf*(Amat(4,1:3)*cH2O(1:3)))/(DH2Oeffsurf
    *Amat(4,4)+kgH2O);

k1surf = k0rx1*exp((-Earx1)/(Rg*Tsurf));
k2surf = k0rx2*exp((-Earx2)/(Rg*Tsurf));
k3surf = k0rx3*exp((-Earx3)/(Rg*Tsurf));
fX1surf = (1-(1-(cMnO2surf/phiMnO2)))^(2/3);
fX2surf = (1-(1-(cMn2O3surf/phiMn2O3)))^(2/3);
fX3surf = (1-(1-(cMn3O4surf/phiMn3O4)))^(2/3);
R1surf = k1surf*fX1surf*cCOsurf;
R2surf = k2surf*fX2surf*cCOsurf;
R3surf = k3surf*fX3surf*cCOsurf;

k4surf = k0rx4*exp((-Earx4)/(Rg*Tsurf));
k5surf = k0rx5*exp((-Earx5)/(Rg*Tsurf));
k6surf = k0rx6*exp((-Earx6)/(Rg*Tsurf));
fX4surf = (1-(1-(cMnO2surf/phiMnO2)))^(2/3);
fX5surf = (1-(1-(cMn2O3surf/phiMn2O3)))^(2/3);
fX6surf = (1-(1-(cMn3O4surf/phiMn3O4)))^(2/3);
R4surf = k4surf*fX4surf*cH2surf;
R5surf = k5surf*fX5surf*cH2surf;
R6surf = k6surf*fX6surf*cH2surf;

dMnO2dtsurf = vmat(1,1)*(1/a0surf)*R1surf + vmat(1,1)*(1/a0surf)*R4surf;
dMn2O3dtsurf = vmat(3,1)*(1/a0surf)*R1surf + vmat(1,2)*(1/a0surf)*R2surf
    +...
    vmat(3,1)*(1/a0surf)*R4surf + vmat(1,2)*(1/a0surf)*R5surf;
dMn3O4dtsurf = vmat(3,2)*(1/a0surf)*R2surf + vmat(1,3)*(1/a0surf)*R3surf
    +...
    vmat(3,2)*(1/a0surf)*R5surf + vmat(1,3)*(1/a0surf)*R6surf;
dMnOdtsurf = vmat(3,3)*(1/a0surf)*R3surf + vmat(3,3)*(1/a0surf)*R6surf;

%Internal:
dTpdt = zeros(length(rc),1);
for i = 1:3
    %%{
    dTpdt(i,1) = 1/(phiMnO2*cpMnO2)*(keff*(Bmat(i,1:3) * Tp(1:3) + Bmat(i
    ,4)*Tsurf) +...
        -deltaHrx1T(i)*(1/a0)*R1(i) + -deltaHrx2T(i)*(1/a0)*R2(i) + -
    deltaHrx3T(i)*(1/a0)*R3(i) + ...
        -deltaHrx4T(i)*(1/a0)*R4(i) + -deltaHrx5T(i)*(1/a0)*R5(i) + -
    deltaHrx6T(i)*(1/a0)*R6(i) + ...
```

```
        -deltaHWGSR*RWGSR(i));
    if Tp(i) >= 2000
        dTpdt(i,1) = 0;
    end
    %}
    %dTpdt(i,1) = 0;
end

dCOdt = zeros(length(rc),1);
for i = 1:3
   dCOdt(i,1) = (1/epsilon)*(DCOeff(i)*(Bmat(i,1:3) * cCO(1:3) + Bmat(i,4)*
    cCOsurf)...
       + vmat(2)*(1/a0)*R1(i) + vmat(2,2)*(1/a0)*R2(i) + vmat(2,3)*(1/a0)*
    R3(i) + vmat(1,4)*RWGSR(i));
end

dH2dt = zeros(length(rc),1);
for i = 1:3
   dH2dt(i,1) = (1/epsilon)*(DH2eff(i)*(Bmat(i,1:3) * cH2(1:3) + Bmat(i,4)*
    cH2surf)...
       + vmat(2)*(1/a0)*R4(i) + vmat(2,2)*(1/a0)*R5(i) + vmat(2,3)*(1/a0)*
    R6(i) + vmat(4,4)*RWGSR(i));
end

dCO2dt = zeros(length(rc),1);
for i = 1:3
   dCO2dt(i,1) = (1/epsilon)*(DCO2eff(i)*(Bmat(i,1:3) * cCO2(1:3) + Bmat(i
    ,4)*cCO2surf)...
       + vmat(4)*(1/a0)*R1(i) + vmat(4,2)*(1/a0)*R2(i) + vmat(4,3)*(1/a0)*
    R3(i) + vmat(3,4)*RWGSR(i));
end

dH2Odt = zeros(length(rc),1);
for i = 1:3
   dH2Odt(i,1) = (1/epsilon)*(DH2Oeff(i)*(Bmat(i,1:3) * cH2O(1:3) + Bmat(i
    ,4)*cH2Osurf)...
       + vmat(4)*(1/a0)*R4(i) + vmat(4,2)*(1/a0)*R5(i) + vmat(4,3)*(1/a0)*
    R6(i) + vmat(2,4)*RWGSR(i));
end

dMnO2dtint = zeros(length(rc),1);
for i = 1:3
```

```
    dMnO2dtint(i,1) = vmat(1,1)*(1/a0)*R1(i) + vmat(1,1)*(1/a0)*R4(i);
end

dMn2O3dtint = zeros(length(rc),1);
for i = 1:3
    dMn2O3dtint(i,1) = vmat(3,1)*(1/a0)*R1(i) + vmat(1,2)*(1/a0surf)*R2(i)
    +...
        vmat(3,1)*(1/a0)*R4(i) + vmat(1,2)*(1/a0surf)*R5(i);
end

dMn3O4dtint = zeros(length(rc),1);
for i = 1:3
    dMn3O4dtint(i,1) = vmat(3,2)*(1/a0)*R2(i) + vmat(1,3)*(1/a0)*R3(i) +...
        vmat(3,2)*(1/a0)*R5(i) + vmat(1,3)*(1/a0)*R6(i);
end

dMnOdtint = zeros(length(rc),1);
for i = 1:3
    dMnOdtint(i,1) = vmat(3,3)*(1/a0)*R3(i) + vmat(3,3)*(1/a0)*R6(i);
end

dcsoliddt = [dMnO2dtint;dMnO2dtsurf;dMn2O3dtint;dMn2O3dtsurf;dMn3O4dtint
    ;...
    dMn3O4dtsurf;dMnOdtint;dMnOdtsurf];
dydt = [dTpdt(:);dCOdt(:);dcsoliddt(:);dH2dt;dCO2dt;dH2Odt];
end
```

Listing E.2: Function doing all the important calculations needed for Script E.1