MODELING CHEMISTRY OF COPPER-BASED OXYGEN CARRIERS IN CHEMICAL LOOPING COMBUSTION SYSTEMS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Eli A. Goldstein
August 2015
© 2015 by Eli Adam Goldstein. All Rights Reserved.
Re-distributed by Stanford University under license with the author.

This work is licensed under a Creative Commons Attribution-Noncommercial 3.0 United States License.
http://creativecommons.org/licenses/by-nc/3.0/us/

This dissertation is online at: http://purl.stanford.edu/cp118dr9233
I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Turgut Gur, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Reginald Mitchell, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Xiaolin Zheng

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost for Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.
Abstract

Development of a model that can be used to predict oxidation rates of copper to cuprous and cupric oxide (Cu$_2$O and CuO, respectively) in the air reactor of a Chemical Looping Combustion (CLC) system is the primary focus of this thesis. The proposed oxidation model, which is based in Wagner Theory and defect chemistry, describes the fundamental processes occurring during copper oxidation. Consequently, it provides better predictive capabilities over a wide range of temperatures and pressures, as well as characteristic particle geometries (spheres, cylinders and plates) than the phenomenological models that are currently being used to predict oxidation rates of Cu particles in CLC systems. In addition to developing this oxidation model, Cu$_2$O and CuO reduction experiments with gaseous (CO, H$_2$ and CH$_4$) and solid (char, wyodak coal, and corn stover) fuels were performed to characterize reaction rates in the fuel reactor of CLC systems. Oxidation rates obtained from the oxidation model, and measured rates from the reduction experiments were used to estimate the size and oxygen carrier loading for a 10 MW$_{th}$ CLC system.

Chemical Looping Combustion is a novel combustion strategy that enables the efficient capture of CO$_2$ without significant energy losses. As a result, CLC systems are expected to have thermal efficiencies that are similar to those of traditional fossil fuel-based power plants however with the advantage of having inherent CO$_2$ capture. To achieve this, CLC systems rely on the cyclic oxidation and reduction of metals and their oxides, known as oxygen carriers, to separate oxygen from air so that fuels can then be burned in an oxygen environment. Metal oxidation with air occurs in an air reactor and metal oxide reduction with a fuel occurs in the fuel reactor. In this way, fuel and air never mix and a stream of near capture ready CO$_2$ leaves the fuel reactor.
The oxygen carrier particles are typically composed of a transition metal, in this work copper, that is supported with an inert material to improve particle utilization, conversion rates, and particle robustness at CLC operating conditions. One of the primary challenges in the advancement of CLC is the development of oxygen carriers with sufficiently fast oxidation and reduction kinetics that will minimize the size of the air and fuel reactors.

To support these efforts, an oxidation model based on defect chemistry has been developed to predict copper oxidation rates to cuprous and cupric oxide. The model is described by the flux of lattice defects diffusing through the growing Cu$_2$O and CuO oxide layers. The modeling supports the conclusion that the dominant defects in Cu$_2$O and CuO are copper vacancies. While interstitial defects were included in the model, their concentrations and diffusion coefficients were found to be too low to impact oxidation rates at conditions relevant to CLC. Defect diffusion coefficients were extracted from experimental data obtained by oxidizing copper wires in a flow reactor at temperatures between 600 and 900 °C in 21% O$_2$ at 1 atm. These defect diffusion coefficients were found to be in good agreement with available values published in the literature, thereby validating the model. The model was further validated by comparing model predictions with and experimental oxidation rates that were obtained by oxidizing unsupported spherical Cu particles at temperatures between 500 and 800 °C in 6 % and 21 % O$_2$ at 1 atm in a thermogravimetric analyzer (TGA).

The oxidation model was also adapted to predict conversion rates of supported copper particles, denoted cermet particles. The cermet particles considered consisted of 20% CuO and 80% ZrO$_2$, and were manufactured using a spray-drying method. In the cermet particle model, both gas phase oxygen transport through pores in the ZrO$_2$ support material and defect diffusion through the oxide layers were considered. The model permits the impact of such variables as CuO and ZrO$_2$ particle sizes, ZrO$_2$ particle porosity, specific surface area, overall cermet particle size, and cermet particle copper loading on copper oxidation rates. Oxidation rates of these particles, which were obtained in a TGA in 21% O$_2$ at temperatures between 600 and 900 °C, are characterized by a fast initial rate followed by a slow, asymptotic approach to
full conversion. The model indicates that the observed particle oxidation rates are chemically controlled. The model can also be used to explain the character of these rates: the initial fast rate correspond to Cu conversion to Cu$_2$O, which is followed by the slower conversion of Cu$_2$O to CuO.

Based on the oxidation model and reduction experiments, a thermodynamic and residence time analysis was performed to estimate the oxygen carrier recirculation rate, oxygen carrier loading and residence time of the oxygen carrier in a direct-fired CLC system. Results from the analysis indicate that the particle morphology can have a significant impact on the size of the reactor system. Oxygen carrier particles with 1 $\mu$m copper sub-particles were found to have air and fuel reactor loadings between 1 and 10 kg per MW$_{th}$ heat released from the air or fuel reactor respectively, while oxygen carrier particles with 10 or 100 $\mu$m sub-particles required significantly larger inventories.
Acknowledgment

There are many people that I want to appreciate for empowering me to reach the milestone of completing my PhD thesis. First and foremost, my progress through the PhD program at Stanford would not have been possible without the guidance and unwavering support from Professor Mitchell. He taught to always question what I know, and inspired me to be a better researcher. While the journey was not easy, it involved working on five different projects and TA’ing six of his classes, I am grateful for all that he has taught me and enabled me to become. I would also like to thank Dr. Turgut Gür for volunteering as my second adviser. He challenged me to understand the fundamentals of defect chemistry and to strive for greater scientific precision in my writings. Additionally, I would like to thank Professor Zheng for serving as the third member of my reading committee and also for being present during all of my major milestones at Stanford from my qualification exam to my thesis defense. The journey that was my PhD would have never started without Deborah Goodings, my faculty adviser at The University of Maryland. She encouraged me to apply to Stanford and reminding me that life is more than just studying.

As I mentioned, I worked on many research projects during my time at Stanford. The most important was the first one with George Touchton and Marisa Zuzga. They introduced me to the concept of Chemical Looping Combustion and provided the seedling funding that ultimately enabled me to join the PhD program at Stanford. While George was unable to see the completion of my thesis, I will always remember his zeal for Chemical Looping Combustion and desire to make the world a better place.

I would also like to thank my friends at Stanford, around the Bay, and across the
country for making these last seven years so enjoyable. I would specifically like to acknowledge David, Yi, Kai and Brandon for keeping me focused on my progress, yet reminding me to celebrate all the milestones along the way. Jason and Miriam for encouraging me be creative. And Matt, Brentan, and Pratap for keeping me humble.

Lastly, I would like to thank my family, Sarahann, Michal, Stephen and Mindy for their never ending love and support. This was for you.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>v</td>
</tr>
<tr>
<td>Acknowledgment</td>
<td>viii</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Motivation: Global Electricity</td>
<td>1</td>
</tr>
<tr>
<td>1.1.1 Fossil fuel resource availability</td>
<td>2</td>
</tr>
<tr>
<td>1.1.2 Environmental Impact From Fossil Fuel Usage</td>
<td>5</td>
</tr>
<tr>
<td>1.2 Background: Carbon Capture Technologies</td>
<td>8</td>
</tr>
<tr>
<td>1.3 Chemical Looping Combustion</td>
<td>10</td>
</tr>
<tr>
<td>1.3.1 Metal Oxide Selection Criteria</td>
<td>12</td>
</tr>
<tr>
<td>1.3.2 Chemical Looping Combustion with Solid Fuels</td>
<td>14</td>
</tr>
<tr>
<td>1.4 Oxidation and Reduction Kinetics</td>
<td>23</td>
</tr>
<tr>
<td>1.5 Thesis Objectives</td>
<td>26</td>
</tr>
<tr>
<td>2 Copper Oxidation Chemistry</td>
<td>27</td>
</tr>
<tr>
<td>2.1 Literature Review on Cu Oxidation</td>
<td>27</td>
</tr>
<tr>
<td>2.2 Theoretical Approach</td>
<td>29</td>
</tr>
<tr>
<td>2.2.1 Equilibrium Defect Concentrations</td>
<td>31</td>
</tr>
<tr>
<td>2.2.2 Defect Transport Model</td>
<td>35</td>
</tr>
<tr>
<td>2.2.3 Defect Concentration Profiles</td>
<td>37</td>
</tr>
<tr>
<td>2.2.4 Oxide Layer Growth</td>
<td>38</td>
</tr>
<tr>
<td>2.3 Materials and Methods</td>
<td>40</td>
</tr>
<tr>
<td>2.3.1 Flow Reactor Experiments</td>
<td>42</td>
</tr>
</tbody>
</table>
2.3.2 TGA Experiments ........................................... 44
2.4 Results and Data Analysis ..................................... 49
2.5 Discussion ....................................................... 52
  2.5.1 Impact Of Impurities ......................................... 56
  2.5.2 Sensitivity Analysis .......................................... 59

3 Cermet Particle Model ............................................ 63
  3.1 Theoretical Approach .......................................... 63
  3.2 Experimental Studies .......................................... 66
    3.2.1 The Cermet Particles ..................................... 67
    3.2.2 TGA Experiments ........................................... 71
  3.3 Results and Data Analysis ..................................... 75
  3.4 Estimating Oxidation Rates ................................... 77

4 Copper Oxide Reduction Chemistry ............................. 80
  4.1 Reduction Chemistry Background and Theory ................. 81
    4.1.1 Background on CuO Reduction ............................. 83
  4.2 Experimental Section .......................................... 84
    4.2.1 Oxygen Release Pathway .................................. 87
    4.2.2 Batch Reactor .............................................. 88
    4.2.3 Gaseous Fuel Experimental Results ....................... 90
    4.2.4 Solid Fuels ............................................... 93
    4.2.5 TGA Experiments .......................................... 98
    4.2.6 TGA Experimental Results ............................... 101

5 Residence Time Analysis ......................................... 105
  5.1 Thermodynamic Model ......................................... 106
    5.1.1 Parametric Study Results ................................. 109
  5.2 Residence Time Analysis ...................................... 115
    5.2.1 Air Reactor Results and Discussion ...................... 119
    5.2.2 Fuel Reactor Results and Analysis ....................... 122
    5.2.3 CLC System Analysis Results and Discussion .......... 126
6 Summary of Major Results and Conclusions

6.1 Oxidation .................................................. 129
6.2 Reduction .................................................... 130
6.3 System Analysis .............................................. 131
6.4 Future Work ................................................ 132

A Surface Chemistry Reduction Model ............................................. 134

B Experimental Procedures ......................................................... 139

B.1 Thermogravimetric Analyzer ............................................. 139
B.2 Coulter Counter Procedure .............................................. 140
B.3 Tap Density Procedure ............................................... 140
B.4 Gas Chromatography .................................................... 141

C Matlab Codes ........................................................................... 142

C.1 Chemical Model ............................................................... 142
  C.1.1 Driver Code ....................................................... 142
  C.1.2 Initialization Function ........................................... 143
  C.1.3 Oxidation Model .................................................. 162
  C.1.4 Oxidation Model .................................................. 164
C.2 Cermet Particle Model .......................................................... 174
  C.2.1 Driver Code ....................................................... 174
  C.2.2 Initialization Function ........................................... 188
  C.2.3 Oxidation Model .................................................. 199
  C.2.4 Cermet particle model- distributed copper sub-particles .... 204
  C.2.5 Cermet particle model- core copper .................................. 214
  C.2.6 Copper oxidation subroutine ..................................... 223
  C.2.7 Defect Equilibrium .............................................. 242
C.3 System Model ................................................................. 250
  C.3.1 CLC Plant Driver .................................................. 251
  C.3.2 Initialization function for the CLC plant model ............... 255
  C.3.3 CLC Plant Function .............................................. 257
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Shale gas resources around the world, 2013 estimates [4]</td>
<td>6</td>
</tr>
<tr>
<td>1.2</td>
<td>Oxygen transport capacity of Cu, Fe, Ni and Mn based oxygen carriers</td>
<td>14</td>
</tr>
<tr>
<td>1.3</td>
<td>Summary of experimental conditions from [29,33,43,44]</td>
<td>22</td>
</tr>
<tr>
<td>2.1</td>
<td>Key defects in the Cu$_2$O and CuO layers during Cu oxidation</td>
<td>30</td>
</tr>
<tr>
<td>2.2</td>
<td>Defect reactions, equilibrium constants and formation energies for de-</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>fects in Cu$_2$O and CuO</td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td>Impurity atom weight and atom fractions measured in the wire and</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>sphere samples; the analysis is provided by Alfa Aesar and measured</td>
<td></td>
</tr>
<tr>
<td></td>
<td>using GDMS for the Cu wires or ICPA for the Cu spheres. * Atom</td>
<td></td>
</tr>
<tr>
<td></td>
<td>fractions calculated from the impurity analysis provided by Alfa Aesar</td>
<td>73</td>
</tr>
<tr>
<td>2.4</td>
<td>Estimated Impurity concentrations in the Cu$_2$O layer. This table only considers the metallic species found in the base copper of the wires and spheres.</td>
<td>57</td>
</tr>
<tr>
<td>2.5</td>
<td>Estimated Impurity concentrations in the CuO layer. This table only considers the metallic species found in the base copper of the wires and spheres.</td>
<td>58</td>
</tr>
<tr>
<td>3.1</td>
<td>Properties of Spray Granulated CuO/ZrO$_2$ particles</td>
<td>73</td>
</tr>
<tr>
<td>4.1</td>
<td>Copper oxides and Cu structure</td>
<td>83</td>
</tr>
<tr>
<td>4.2</td>
<td>Initial conditions for gaseous fuel oxidation with copper oxide</td>
<td>90</td>
</tr>
<tr>
<td>4.3</td>
<td>Kinetic parameters from CuO reduction with CO, H$_2$ and CH$_4$</td>
<td>93</td>
</tr>
<tr>
<td>4.4</td>
<td>Ultimate, Elemental and Heating Value Analysis</td>
<td>95</td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Global electricity production in (a) 2010 and, (b) projected electricity out to 2040 [2].</td>
</tr>
<tr>
<td>1.2</td>
<td>Proven natural gas reserves around the world as of 2013, total world reserves of 6,845 trillion ft$^3$ (b) global coal reserves as of 2011, total coal reserves 979 billion short tons.</td>
</tr>
<tr>
<td>1.3</td>
<td>Chemical Looping Combustion System utilizing a copper based oxygen carrier.</td>
</tr>
<tr>
<td>1.4</td>
<td>Pure Cu oxygen carrier oxidized and reduced in a TGA by de Diego et al. using air as the oxidizing agent and CH$_4$ as the reducing agent [19].</td>
</tr>
<tr>
<td>1.5</td>
<td>(a) Direct and (b) indirect CLC systems [18].</td>
</tr>
<tr>
<td>1.6</td>
<td>Kinetic Pathways for solid fuel combustion in a fuel reactors of CLC systems.</td>
</tr>
<tr>
<td>1.7</td>
<td>Equilibrium oxygen partial pressure vs temperature.</td>
</tr>
<tr>
<td>2.1</td>
<td>Number fraction of $V''<em>{Cu,e'}$ and $h'$ as a function of $\Delta H''</em>{R}^{V''_{Cu}}$ in CuO at 500 °C (a) and 900 °C (b). The dark band represents the region of reaction enthalpies based on the expected defect concentrations in CuO.</td>
</tr>
<tr>
<td>2.2</td>
<td>Lattice defect number fraction in the Cu$_2$O layer at the Cu/Cu$_2$O and Cu$_2$O/CuO interface (a) and electronic defects in the Cu$_2$O layer at the $Cu/Cu_2O$ and Cu$_2$O/CuO interface (b). Dotted lines are the number fraction at the Cu$_2$O/CuO interface and solid lines for the Cu/Cu$_2$O interface.</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>2.3</td>
<td>a) Vacancy, hole and electron number fraction and b) copper interstitial number fraction in the CuO layer at the Cu$_2$O/CuO and CuO/O$_2$ interface. Dotted lines are number fraction of defects at the CuO/O$_2$ interface and solid lines for the Cu$_2$O/CuO interface.</td>
</tr>
<tr>
<td>2.4</td>
<td>Schematic of a copper film during oxidation, illustrating the Cu/Cu$_2$O interface, Cu$_2$O/CuO interface and the CuO/O$_2$ interface and the equilibrium concentration profiles through the Cu$_2$O and CuO layers at a given oxide layer thickness.</td>
</tr>
<tr>
<td>2.5</td>
<td>Vacancy, hole, electron, and interstitial number fraction profiles of charged defects, across the Cu$_2$O (a) and CuO layer (b) in 21% O$_2$ for a spherical at 500, 600 and 900 °C as a function of normalized distance through the Cu$_2$O and CuO layers. Note that the electron concentration profile in Cu$_2$O is not shown.</td>
</tr>
<tr>
<td>2.6</td>
<td>Mass distributions for the nominally 10 µm (a) and 35 µm spherical copper particle samples (b) used in the TGA experiments.</td>
</tr>
<tr>
<td>2.7</td>
<td>Flow reactor setup with sample in the cooling section.</td>
</tr>
<tr>
<td>2.8</td>
<td>SEM images of copper wires in cross section, at 700 °C exposed for 0, 1, 2 and 3 hours in panels a through d, respectively.</td>
</tr>
<tr>
<td>2.9</td>
<td>(a) Thicknesses of the Cu$_2$O, CuO oxide layers and total oxide thickness (Cu$_2$O+CuO layer) at 800 °C and 21% O$_2$, and (b) the total oxide thickness as a function of the square root of time. Dotted lines represent calculated model predicted results and triangles are data points measured from the flow reactor experiments.</td>
</tr>
<tr>
<td>2.10</td>
<td>Conversion vs time of the nominally 10 µm (a) and 35 µm (b) particles exposed to 21% O$_2$ obtained from the TGA. The lines are TGA data and the error bars are based on repeat experiments.</td>
</tr>
<tr>
<td>2.11</td>
<td>Conversion vs time of the nominally 10 µm and 35 µm Cu particles at 600 °C (a) and 800 °C (b) exposed to 6 or 21% O$_2$. The lines are TGA data and the error bars are based on repeat experiments.</td>
</tr>
</tbody>
</table>
2.12 Vacancy diffusion coefficients in Cu$_2$O (a) and CuO (b) where the data points are from this study, red solid line is a best fit, and the dashed lines in (a) are: MSG – Mrowec et al. [95], GMM – Grzesik et al. [80] and MFP Mahendra et al. [96], dashed lines (b) GM – Grzesik and Migdalska. [53] ................................. 50

2.13 Measured (triangles) by TGA experiments and calculated (solid lines) extent of conversion profiles for the nominally 35 $\mu$m (a,b) and 10 $\mu$m (c,d) copper particles exposed to 6% (a,c) and 21% (b,d) oxygen at 500 (black), 600 (blue), 700 (green) and 800 (red) °C. The dark gray circle in (a) denotes the point in the model where Cu depletion occurs. 52

2.14 Overall self-diffusion coefficients for defects in Cu$_2$O evaluated under P$_{O_2}$ at the Cu/Cu$_2$O phase boundary (a) and self-diffusion coefficient in CuO evaluated under 21% O$_2$ (b). Data points, this study; red solid line best fit; dashed lines in (a): WP– Wiley Peterson [81], MC– Moore Castellan [97], MSMoore and Selikson [98], HN– Haugsrud Norby [99], MSG– Mrowec, Stoklosa and Godlewski [95]; dashed lines in (b): GM Grzesik and Migdalska [53] ................................. 55

2.15 Equilibrium defect number fractions in a) Cu$_2$O and b) CuO as a function of temperature for positive impurity loadings of pure Cu, —-wires (2 $\times$ 10$^{-7}$ in Cu$_2$O and 8.5 $\times$ 10$^{-10}$ in CuO), and spheres (8.3 $\times$ 10$^{-4}$ in Cu$_2$O and 1.5 $\times$ 10$^{-4}$ in CuO) ................................. 58

2.16 Predicted Cu-sphere conversion at 700 °C in 21% O$_2$ for impurity atom fraction of a positively charged defect between 0 and 10$^{-4}$ ................................. 59

2.17 Sensitivity coefficient at 50% conversion for a ± 10% perturbation of $\Delta H_j$, for $j = V^z_{Cu}$, $V'_{Cu}$ in Cu$_2$O and $V''_{Cu}$ in CuO for T = 600, 733, 866, and 1000 °C ................................. 61

2.18 Sensitivity coefficient at 50% conversion for a ± 10% perturbation of $D_j$, for $j = V^{Cu}_{Cu}$, $V^{Cu_{2}O}_{Cu}$ in Cu$_2$O and $V^{CuO}_{Cu}$ in CuO for T = 600, 733, 866, and 1000 °C ................................. 62
### Section 3.1
Schematic of a model cermet particle showing different distributions of
the sub-particles for a given copper weight fraction

### Section 3.2
SEM of particles used in this work. Panel a) shows a far field image
of the CuO/ZrO$_2$ particles, b) a close up of the grain structure of the
CuO/ZrO$_2$ c) d) individual particles. x’s identify CuO sub-particles.

### Section 3.3
SEM of the cross-section of the particles used in this work.

### Section 3.4
Cross section of the reduced particles, 16 wt.% Cu/ZrO$_2$ particles, x’s
identify Cu sub-particles. Panel (d) is a close up of the image in panel
(c).

### Section 3.5
In panel a) is the conversion of the OC at 600 °C at different cycles
and in panel b) is the conversion of the oxygen carrier at temperatures
between 600 and 900 °C and c) early conversion time.

### Section 3.6
Predicted oxygen mole fraction as a function of time at 600 to 900 °C

### Section 3.7
The measured and modeled extent of conversion for oxidation of the 16
wt. % Cu cermet particle at 600 °C in 21% O$_2$ balance N$_2$ and 1 atm.
The model predictions are for several characteristic copper diameters
ranging from 5 µm to 20 µm.

### Section 3.8
Experimentally measured conversion with modeled predicted output
at a) 600, b) 700, c) 800 and d) 900 °C for a characteristic Cu diameter
of 8 µm.

### Section 3.9
Average reactivity, as predicted by the cermet particle model and dif-
fusion limited oxidation rate.

### Section 3.10
Average reactivity as a function of the characteristic copper diameter

### Section 4.1
Size distributions of the CuO particles used in the flow reactor and TGA

### Section 4.2
Raw Samples (a) CuO, (b) close up of the CuO particles, (c) Cu$_2$O.

### Section 4.3
CuO sample after partial reduction (a) particle view, (b) close up of
the surface.

### Section 4.4
Equilibrium analysis of the thermal dissociation of CuO.

### Section 4.5
(a) Drawing of the flow reactor and (b) flow reactor experiment encased
in insulation.
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6</td>
<td>Carbon monoxide oxidation via CuO, with initial concentration 100% CO</td>
</tr>
<tr>
<td>4.7</td>
<td>Hydrogen oxidation via CuO, with initial concentration 20% H₂ in Ar</td>
</tr>
<tr>
<td>4.8</td>
<td>Methane oxidation via CuO, (a) major species, (b) minor species</td>
</tr>
<tr>
<td>4.9</td>
<td>Maximum reactivity from the flow reactor (dashed lines) and TGA reduction experiments (data points) along with the diffusion limited rates (solid lines)</td>
</tr>
<tr>
<td>4.10</td>
<td>Size distribution for coal sieved with the 75 to 106 μm screens</td>
</tr>
<tr>
<td>4.11</td>
<td>CuO and Activated Carbon</td>
</tr>
<tr>
<td>4.12</td>
<td>Wyodak Coal Oxidation</td>
</tr>
<tr>
<td>4.13</td>
<td>CuO and Rice Straw</td>
</tr>
<tr>
<td>4.14</td>
<td>CuO and Corn Stover</td>
</tr>
<tr>
<td>4.15</td>
<td>Repeatability and bed diffusion (a) 200 °C, (b) 300 °C. Note with a bed mass larger than 20 mg, bed diffusion starts to influence the observed reactivity</td>
</tr>
<tr>
<td>4.16</td>
<td>CuO reduction with 5% CO in N₂</td>
</tr>
<tr>
<td>4.17</td>
<td>CuO reduction with 1.6% CO in N₂</td>
</tr>
<tr>
<td>4.18</td>
<td>Cu₂O reduction with 1.6% CO in N₂</td>
</tr>
<tr>
<td>4.19</td>
<td>CuO and Cu₂O conversion at 300 °C</td>
</tr>
<tr>
<td>5.1</td>
<td>Model system for the direct-fired CLC system; Solid Gas Separator (SGS)</td>
</tr>
<tr>
<td>5.2</td>
<td>a) the O₂ concentration in the exhaust of the AR, b) the carbon fraction in the exhaust of the FR</td>
</tr>
<tr>
<td>5.3</td>
<td>Heat input to the (a) air reactor and (b) fuel reactor</td>
</tr>
<tr>
<td>5.4</td>
<td>OC recirculation rate exiting the AR and FR per MWth of fuel (LHV · mFuel)</td>
</tr>
<tr>
<td>5.5</td>
<td>OC conversion from the exhaust of the fuel and air reactors with N.G., S.G. and W.C. as the fuels</td>
</tr>
<tr>
<td>5.6</td>
<td>Mass fractions of the OC exiting the a) FR and b) AR with natural gas as the fuel</td>
</tr>
<tr>
<td>5.7</td>
<td>Average Reaction Rate in the AR, φFR = 0.8</td>
</tr>
</tbody>
</table>
5.8 Oxygen Carrier Inventory of the AR, φFR = 0.8 ........................................... 120
5.9 Residence Time of the OC thorough the AR, For a fuel reactor equivalence ratio of 0.8 ................................................................. 121
5.10 Air reactor inventory of for OC4 as a function of the AR and FR equivalence ratio when a) N.G., b) S.G. and c) W.C. are the primary fuel .................................................................................................................. 122
5.11 Average Reaction Rate in the FR, φAR = 1 ...................................................... 123
5.12 OC inventory in the FR, φAR = 1 .................................................................... 124
5.13 Residence Time of the FR, φAR = 1 ................................................................. 125
5.14 Fuel reactor inventory of for OC4 as a function of the AR and FR equivalence ratio when a) N.G., b) S.G. and c) W.C. are the primary fuel .................................................................................................................. 126
5.15 Oxygen carrier inventory for the AR and FR when a) N.G., b) S.G. and c) W.C. are the fuel ............................................................................................................................................. 127
B.1 TGA Calibration test with calcium oxalate ..................................................... 140
B.2 Signal measured by GC from calibration run. Gas mixtures used: (1) 50% CO, 50% CO2, (2) 21% CO, 21% CO2, (3) 14% CO, 14% CO2 .............................................................. 141
Chapter 1

Introduction

Producing environmentally benign, affordable electricity is one of the most pressing issues facing the current generation of scientist and engineers. Presently, a majority of electricity is produced from fossil fuel combustion. While fossil fuels like coal and natural gas are currently inexpensive and abundant, the extraction and combustion of fossil fuels are harmful to the environment. In particular, the mining of fossil fuels reduces air quality and endangers many natural ecosystems, and the combustion of fossil fuels results in the uncontrolled release of carbon dioxide (CO$_2$) to the environment, which is the primary cause of global climate change \[1\]. As a result, the continued use of fossil fuels at the present rate of consumption is unsustainable.

In an effort to support the development of technologies that reduce global CO$_2$ emissions from fossil fuel combustion, this thesis investigates a novel combustion strategy, Chemical Looping Combustion (CLC), which enables the efficient capture of CO$_2$. This first chapter motivates why carbon capture systems should be studied, discusses the benefits and challenges when using CLC, and also serves to introduce the specific objectives of this thesis.

1.1 Motivation: Global Electricity

In 2010, nearly 20 trillion kilowatt-hours of electricity were generated globally, providing electricity for roughly 81% of the world’s population (5.5 billion people) \[2\].
The remaining 19% of the global population (1.3 billion people) lacks access to electricity. As shown in Fig. 1.1a, more than 60% of the electricity generated in 2010 was produced via the combustion of fossil fuels like coal and natural gas. According to the 2013 International Energy Outlook published by the International Energy Agency (IEA), global electricity production is expected to nearly double to 38 trillion kilowatt-hours by 2040 [2]. During this period, nearly 75% of the new electricity capacity will be driven by economic growth in developing countries [2], where currently 19% of the global population lacks access to electricity [3].

IEA projections of the primary fuels and technologies likely to be used to generate electricity in 2040 are shown in Fig. 1.1b. The fastest growing generation source is renewables, which includes hydro-power, wind, and solar. Renewable sources are expected to grow to 25% of the global electricity capacity from its current value of 21%, providing 9.6 trillion kilowatt-hours of electricity, up from 4.2 trillion kilowatt-hours from 2010. While electricity generated from renewables is expected to grow to 25% of the total capacity, nearly 60% of the remaining electricity generated in 2040 will come from the combustion of fossil fuels, specifically coal and natural gas. In the IEA report, natural gas is expected to remain a favorable fuel for electricity generation because of the relatively low initial cost to build natural gas power plants and because it releases less CO$_2$ upon combustion than coal for a given amount of energy released [2]. Coal is predicted to remain the largest resource used to generate electricity around the world. This is a consequence of the large coal reserves and electricity demand from populations in India, China, the United States, and Australia and the extensive infrastructure to mine/transport coal around the globe [2]. As a result, coal and natural gas will continue to be the dominant resource used to generate electricity for well beyond the next 30 years.

1.1.1 Fossil fuel resource availability

When discussing future electricity demand, it should be acknowledged that fossil fuels like coal and natural gas are limited resources, and future production of electricity can only be met by fossil fuels provided the resources are available. Shown in Fig.
1.1. MOTIVATION: GLOBAL ELECTRICITY

Figure 1.1: Global electricity production in (a) 2010 and, (b) projected electricity out to 2040 [2]
CHAPTER 1. INTRODUCTION

1.2a is the global distribution of conventional natural gas reserves estimated in 2013 and in Fig. 1.2b the global distribution of coal reserves estimated in 2011. As of 2013, there were nearly 6,845 trillion ft\(^3\) of proven natural gas reserves and the majority of it is located in the Middle East and Eurasia (mainly Russia). As for coal reserves, in 2011 there were an estimated 979 billion short tons of coal spread throughout North America, Eurasia and Asia Oceania. From Fig. 1.2a it can be seen that nearly 75% of the world’s coal reserves is concentrated in five countries: China, India, United States, Australia and Russia [2].

The annual consumption of natural gas and coal are 119 trillion ft\(^3\) and 8.3 billion short tons, respectively. Considering an average increase in consumption per year of 1.8% and 2.5% for coal and natural gas, respectively, corresponding to the increase in consumption estimated by the IEA (Fig. 1.1b), the current coal and natural gas resources will be depleted in the next 35 to 60 years (this assumes that coal and natural gas power plants remain at their current efficiencies).

![Proven natural gas reserves around the world as of 2013, total world reserves of 6,845 trillion ft\(^3\) (b) global coal reserves as of 2011, total coal reserves 979 billion short tons.](image)

Figure 1.2: Proven natural gas reserves around the world as of 2013, total world reserves of 6,845 trillion ft\(^3\) (b) global coal reserves as of 2011, total coal reserves 979 billion short tons.

More recently, there has been additional efforts to mine non-conventional fossil fuel resources. These are natural gas and oil deposits that cannot be extracted using traditional drilling and mining processes and thus, they are different from the coal
1.1. MOTIVATION: GLOBAL ELECTRICITY

and natural gas resources presented in Fig. 1.2. For the past several decades, the extraction of non-conventional fuels was not economical. As the price for conventional fuels increased over the past decade, resulting in $100 per barrel oil and natural gas prices above of $10 per thousand ft$^3$, there has been a greater economic incentive to develop new extraction technologies for non-conventional fuels. As a result, the cost of extracting some non-conventional fuels has significantly decreased. Examples of non-conventional fossil fuels are heavy oil, shale oil, shale gas, methane hydrates, and coal bed methane.

The most disruptive example of non-conventional fuels on natural gas markets is shale gas. The global estimate of recoverable shale gas by the Energy Information Agency (EIA) is 7,299 trillion ft$^3$, which is commensurate with the estimate for conventional natural gas reserves of 6,845 trillion ft$^3$ made in 2013 [4]. Within Table 1.1 is a list of 10 countries with the largest shale gas deposits along with their proven conventional natural gas deposits. The information presented in Table 1.1 indicates that shale gas reserves in almost all countries is much larger than the conventional natural gas reserves. Considering both conventional and non-conventional gas resources as well as the abundance of coal, there will likely be a sufficient supply of natural gas and coal to generate electricity over the next century.

1.1.2 Environmental Impact From Fossil Fuel Usage

There are significant negative consequences associated with coal- and natural gas-based electricity generation. In part, these result from the fact that fossil fuel combustion emits CO$_2$ and that the processes used to extract fossil fuels are destructive to the environment. Carbon dioxide is a greenhouse gas and is the primary pollutant responsible for global warming [1]. Atmospheric carbon dioxide concentrations have been increasing since the pre-industrial era, and this is associated with the growing global population and expanding global economies [1]. Over 85% of anthropogenic CO$_2$ emissions results comes from activities or processes that use fossil fuels and this includes electricity generation, transportation, industrial processes and residential heating. However, global electricity generation is by far the largest single source of
CHAPTER 1. INTRODUCTION

Table 1.1: Shale gas resources around the world, 2013 estimates [4]

<table>
<thead>
<tr>
<th>Rank</th>
<th>Country</th>
<th>Estimated Shale Gas Reserves, trillion ft(^3)</th>
<th>Proven Reserves, Conventional Natural Gas (2013), trillion ft(^3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>China</td>
<td>1,115</td>
<td>141</td>
</tr>
<tr>
<td>2</td>
<td>Argentina</td>
<td>802</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>Algeria</td>
<td>707</td>
<td>159</td>
</tr>
<tr>
<td>4</td>
<td>U.S.</td>
<td>665</td>
<td>308</td>
</tr>
<tr>
<td>5</td>
<td>Canada</td>
<td>573</td>
<td>68</td>
</tr>
<tr>
<td>6</td>
<td>Mexico</td>
<td>545</td>
<td>17</td>
</tr>
<tr>
<td>7</td>
<td>Australia</td>
<td>437</td>
<td>44</td>
</tr>
<tr>
<td>8</td>
<td>South Africa</td>
<td>390</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>Russia</td>
<td>285</td>
<td>1,688</td>
</tr>
<tr>
<td>10</td>
<td>Brazil</td>
<td>245</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>World Total</td>
<td>7,299</td>
<td>6,845</td>
</tr>
</tbody>
</table>

CO\(_2\) emissions; the coal and natural gas burned in 2013 to meet the global electricity demand accounted for nearly 41% of the global fossil-fuels related CO\(_2\) emissions, or nearly 13 billion tons of CO\(_2\) [5].

According to the Intergovernmental Panel on Climate Change: Fifth Assessment Report [6], the continued release of greenhouse gases (i.e., if nothing is done to reduce CO\(_2\) or CH\(_4\) emissions) will likely result in increased global mean surface temperatures by 3.7 \(^\circ\)C to 4.8 \(^\circ\)C (relative to pre-industrial levels), with CO\(_2\) concentrations exceeding 750 parts per million (ppm) by 2100 [6]. While the impact from these changes on the environment is uncertain, the more that is done today to reduce global greenhouse gas emissions will limit the overall impact to the environment and global economies [6].

In addition to CO\(_2\) emissions from coal and natural gas combustion, fossil fuel extraction processes like natural gas drilling, hydraulic fracturing for shale gas, and
coal mining also have a negative impact on the environment. Natural gas drilling and hydraulic fracturing of shale gas result in the emissions of unburnt hydrocarbons, mainly methane (CH$_4$). Methane, like CO$_2$, is also a greenhouse gas and has an even stronger global warming potential than CO$_2$ [7,8]. Howarth et al. [8] have estimated methane emissions from natural gas drilling and shale gas hydraulic fracturing to be between 1.7% to 6%, and 3.6% to 8% of total production of a given well, respectively. In general, power produced from natural gas results in less greenhouse gas emission than coal produced power. However when accounting for this auxiliary emissions from hydraulic fracturing, Alvarez et al. [7] have found that burning methane produced from hydraulic fracturing only results in less pollutant emissions than coal combustion, provided leakage rates during drilling are less than 3.2%.

While coal mining does not result in the direct emissions of unburnt hydrocarbons as hydraulic fracturing, it is often performed by strip mining or mountain top removal mining which results in the displacement of natural wildlife, the destruction of natural habitats, leads to the degradation of air quality, and the creation of tailing ponds. Tailing ponds contain the waste streams from the mining process (e.g., mercury, sulfur, naturally occurring radioactive materials found in fossil fuels) as well as chemicals used to extract and separate coal from the surrounding mineral and overburden materials.

Despite the known implications to the environment from continued fossil fuels use, new advancements in the extraction of unconventional fuels along with the abundance of coal and natural gas resources in the United States, China and India, and the vast infrastructure associated with coal and natural gas, signal that these carbon-based fuels will continue to be utilized instead of switching to more renewable, non-carbon dioxide emitting fuels. In order to maintain the current electricity generation in the developed world and future electrification of the developing world, CO$_2$ capture technologies should be implemented on existing and new coal- and natural gas-based power plants. Chemical Looping Combustion (CLC) is one promising CO$_2$ capture strategy because it produces near capture ready CO$_2$ without the additional complexity and energetic penalties associated with alternative CO$_2$ capture solutions.
1.2 Background: Carbon Capture Technologies

In conventional power plants that burn fossil fuels with air, the major species in the flue gas are the combustion products CO$_2$ and H$_2$O, N$_2$ which enters the system with air, and unreacted O$_2$. By volume, the CO$_2$ concentration in the flue gas is typically between 12 to 14% when coal is the fuel and 3 to 5% when natural gas is the fuel [9]. In order to reduce CO$_2$ emissions to the environment from power plants, CO$_2$ should be captured. Since the CO$_2$ makes up between 15% and 3% of the exhaust stream, it does not make sense to capture all power plant flue gases. However, it is challenging to separate CO$_2$ from N$_2$ in the exhaust of a power plant.

To address the challenges related to CO$_2$ capture from power plants, several combustion strategies have been developed that produce separate streams of CO$_2$. These strategies focus on either reducing the energy to (1) separate CO$_2$ from N$_2$ or (2) separate N$_2$ and O$_2$ such that the flue gas is predominantly CO$_2$. Carbon capture technologies can be divided into three groups based on how fuels are burned and the CO$_2$ captured from the exhaust of a power plant. They are 1) post-combustion, 2) pre-combustion and 3) oxyfuel-combustion [9][11].

Post-combustion strategies enable CO$_2$ separation from the flue gas of conventional power plants and typically use solvents, sorbents or membranes to accomplish this separation process. In solvent-based post-combustion systems, solvents physically or chemically absorb CO$_2$ to separate it from the exhaust stream of a power plant. The most common solvent used for post-combustion CO$_2$ capture is monodiethanolamine (MDEA); other solvents are based around amines, ammonia or potassium bicarbonate [10][11]. In sorbent-based post-combustion systems, a high surface area solid called a sorbent physically or chemically adsorbs CO$_2$ to separate it from the exhaust stream [10]. Common sorbent materials are zeolites, solid amines or metal organic frameworks [9][11]. Membrane-based systems operate on a principle that utilizes materials that are preferentially permeable to CO$_2$ [9].

Solvent and sorbent post combustion strategies are the most mature of the three CO$_2$ capture strategies because they are similar to processes currently performed in industrial facilities [9]. They also require the least modification to the combustion
1.2. BACKGROUND: CARBON CAPTURE TECHNOLOGIES

process and would likely be implemented as a separate unit after conventional pollution controls [9]. Both solvent/sorbent and membrane systems have drawbacks. Solvent and sorbent based systems require a significant amount of energy to regenerate the solvent or sorbent material and if added to a power plant, researchers predict a reduction in the overall plant efficiency between 5 and 13% points [9,10,12,13]. Most membrane technologies have drawbacks because they lack sufficient selectivity to separate CO₂ and N₂ and have a low permeance to CO₂ thereby requiring excessively large membrane areas [9].

Pre-combustion CO₂ capture technologies utilize gasification systems to convert solid fuels like coal or biomass to a synthesis gas composed of CO, CO₂, H₂ and H₂O. Since gasification is an endothermic process, it often requires the partial burning of the fuel in air or oxygen to occur. If air is used, there will be some N₂ in the exhaust syngas stream; however, it will be in lower concentrations than the N₂ in the flue gas of a conventional power plant. A water-gas-shift reactor is then used to convert the remaining CO and H₂O present in the syngas to CO₂ and H₂ (following the water gas shift reaction CO+H₂O ↔ CO₂+H₂). The exhaust of the water gas shift reactor is a mixture of CO₂, H₂ with a small amount of CO, H₂O and N₂. With the CO₂ in high concentration within the exhaust, it can be separated using solvent, sorbent or membrane CO₂ capture process used in post-combustion systems. The benefit to gasifying the fuel and separating out CO₂ before combustion occurs is that the CO₂ is in a much higher concentration than in the flue gas of a conventional power plant and the volume of gas that needs to be processed in pre-combustion system is much less than a post-combustion system [9]. Ultimately being able to process a smaller volume of gas translates to smaller CO₂ capture systems and lower capital costs [9].

Another strategy for carbon capture is oxyfuel combustion. Oxyfuel combustion technologies rely on separating O₂ from air instead of separating CO₂ from the fuel as in pre-combustion, or CO₂ from N₂ as in post-combustion. In oxyfuel combustion systems, the fuel is burned with oxygen such that the combustion products are predominantly CO₂ and H₂O, and are devoid of N₂ (solid fuels often contain some nitrogen and this will be present in the exhaust regardless of the combustion strategy). The CO₂ in the exhaust can then easily be separated from the remaining H₂O
via condensation, yielding a stream of capture ready CO$_2$. Separating oxygen from air is typically accomplished by cryogenic separation processes or membrane based systems. The benefit to oxyfuel systems is that the flue gas streams are capture ready CO$_2$ \[9\]. The primary challenge with oxyfuel systems is that cryogenic oxygen separation systems are expensive to build and require significant amounts of energy to operate. Estimates for the energy penalty associated with cryogenic separation facilities are as high as 15% of the plants gross power output \[14\]-\[16\]. Additionally, membrane systems currently lack sufficient throughput and selectivity.

Chemical Looping Combustion (CLC) is a novel oxyfuel combustion strategy that enables the efficient capture of CO$_2$. However it differs from traditional oxyfuel strategies (i.e., cryogenic and membranes) because it relies on the cyclic oxidation and reduction of metals and their oxides, denoted oxygen carriers (OCs) to separate O$_2$ from air. As a result, CLC systems do not have a significant energy penalty associated with O$_2$ separation from air, and as will be shown, they are expected to have thermal efficiencies that are similar to traditional fossil-fuel based power plants but with the advantage of having inherent CO$_2$ capture.

### 1.3 Chemical Looping Combustion

The concept of Chemical Looping Combustion (CLC) was first proposed by Lewis and Gilliland in the early 1950s with the purpose of producing pure CO$_2$ for industrial applications \[17\]. It wasn’t until global warming became a prominent issue in the early 1990s did researchers begin to apply CLC to methane combustion with the aim of CO$_2$ capture.

CLC systems are comprised of two interconnected reactors, as illustrated in Figure 1.3. These two reactors are denoted as the air and fuel reactors, where the OC is oxidized in the air reactor, described by Eq. 1.1a and reduced in the fuel reactor, described by Eq. 1.1b. In these global reactions, $\text{Me}_a\text{O}_b$ represents the oxidized state of the OC (e.g. CuO or Cu$_2$O), $\text{Me}_a\text{O}_{b-1}$ the reduced state (e.g. Cu), and $\text{C}_x\text{H}_y\text{O}_z$ a hydrocarbon based fuel like coal or natural gas. When the fuel is completely oxidized, the exhaust of the fuel reactor will be a mixture of CO$_2$ and H$_2$O, from which a pure...
A stream of CO\textsubscript{2} can be extracted. Combining the global reactions Eq. 1.1\textsubscript{a} and Eq. 1.1\textsubscript{b} to write Eq. 1.1\textsubscript{c} shows that the net reaction of a CLC system is that of direct fuel combustion with air. As a result, CLC systems have the same energetics as conventional combustion systems but with the added benefit of producing capture ready CO\textsubscript{2}.

\begin{equation}
\text{Air Reactor: } \text{Me}_a\text{O}_{b-1} + \frac{1}{2} \cdot (\text{O}_2 + 3.76 \cdot \text{N}_2) \rightarrow \text{Me}_a\text{O}_b + 1.88 \cdot \text{N}_2 \tag{1.1a}
\end{equation}

\begin{equation}
\text{Fuel Reactor: } (2x + \frac{y}{2} - z) \cdot \text{Me}_a\text{O}_b + \text{C}_x\text{H}_y\text{O}_z \rightarrow (2x + \frac{y}{2} - z) \cdot \text{Me}_a\text{O}_{b-1} + x \cdot \text{CO}_2 + \frac{y}{2} \cdot \text{H}_2\text{O} \tag{1.1b}
\end{equation}

Figure 1.3: Chemical Looping Combustion System utilizing a copper based oxygen carrier.
Overall: $C_xH_yO_z + \frac{(2x + y/2 - z)}{2} \cdot (O_2 + 3.76 \cdot N_2) \rightarrow (2x + y/2 - z) \cdot 1.88 \cdot N_2 + x \cdot CO_2 + y/2 \cdot H_2O$ \hfill (1.1c)

Oxygen carriers in CLC systems are typically comprised of a transition metal, e.g., Cu, Fe, Ni or Mn, supported on an inert material such as ZrO$_2$, Al$_2$O$_3$, SiO$_2$, or YSZ. An extensive list of metals and supports that have been used in CLC applications can be found elsewhere [18]. The inert support material is included in the OC to improve particle utilization, conversion rates, and particle robustness at CLC operating conditions. In the case of copper, without such a support, the cyclic conversion of copper would decrease significantly after several oxidation and reduction cycles as can be seen in Fig 1.4. In this figure, the extent of conversion of a pure copper based oxygen carrier being oxidized with air and reduced with CH$_4$ at 800 °C is shown. As was reported by de Diego et al. [19], after several oxidation and reduction cycles, the particles sintered together and consequently required longer oxidation time to reach full conversion.

The particular metal used in the oxygen carrier will determine how much of that energy is released in the air reactor verses the fuel reactor. For all metals, the oxidation in the air reactor is exothermic and reduction in fuel reactor is endothermic, with the exception of a few metals, most notably copper and manganese, for which the fuel reactor is also exothermic.

### 1.3.1 Metal Oxide Selection Criteria

There are a number of criteria to consider when selecting a metal and support material, and designing the oxygen carrier for use in CLC system. The base metal and support material must:

- be capable of transporting a large amount of oxygen in an oxidation-reduction cycle
- have favorable thermodynamics (exothermic in oxidation and reduction and have no limitations to oxidize fuels to CO$_2$ and H$_2$O
1.3. CHEMICAL LOOPING COMBUSTION

- have fast oxidation and reduction kinetics
- have a high melting point relative to CLC system operating temperatures
- not form undesirable side products (e.g., not react with sulfur or chlorine)
- be non-toxic
- be inexpensive

Figure 1.4: Pure Cu oxygen carrier oxidized and reduced in a TGA by de Diego et al. using air as the oxidizing agent and CH$_4$ as the reducing agent [19].

A measure of the OCs ability to transfer oxygen from the AR to FR is defined by its oxygen transport capacity, $R_o$, given by Eq. 1.2 as the weight fraction of active oxygen in the OC. Values of the oxygen transport capacity for commonly used metal/metal oxides in CLC systems are in Table 1.2. The different oxide pairs listed in Table 1.2 reflect some of the different oxidation states that are possible for the metal oxides. While certain transitions may be more desirable because they transfer more oxygen or release more energy, thermodynamics limits which transitions are possible. An asterisk (*) in Table 1.2 denotes the thermodynamically favorable transition. Copper was selected for this study because it has one of the highest oxygen transport capacities, favorable thermochemistry, fast oxidation and reduction kinetics, and is non-toxic [18]. The main objection to using copper is that it has a relatively low
melting point of 1085 °C, and as a result Cu particles have problems with agglomeration [18].

\[
R_o = \frac{m_{ox} - m_{red}}{m_{ox}}
\]  

Table 1.2: Oxygen transport capacity of Cu, Fe, Ni and Mn based oxygen carriers.

<table>
<thead>
<tr>
<th>Reduced/Oxidized State</th>
<th>Oxygen Transport Capacity, [-]</th>
<th>Heat of Reaction, kJ/mole</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cu/CuO *</td>
<td>0.21</td>
<td>-312</td>
</tr>
<tr>
<td>Cu/Cu₂O</td>
<td>0.11</td>
<td>-341</td>
</tr>
<tr>
<td>Cu₂O/CuO</td>
<td>0.10</td>
<td>-283</td>
</tr>
<tr>
<td>FeO/Fe₂O₃</td>
<td>0.10</td>
<td>-560</td>
</tr>
<tr>
<td>Fe₃O₄/Fe₂O₃*</td>
<td>0.034</td>
<td>-472</td>
</tr>
<tr>
<td>Ni/NiO</td>
<td>0.21</td>
<td>-479</td>
</tr>
<tr>
<td>MnO/Mn₂O₃</td>
<td>0.10</td>
<td>-377</td>
</tr>
<tr>
<td>Mn₃O₄/Mn₂O₃*</td>
<td>0.034</td>
<td>-202</td>
</tr>
<tr>
<td>MnO/Mn₃O₄</td>
<td>0.070</td>
<td>-464</td>
</tr>
</tbody>
</table>

(*) denotes thermodynamically favored state in oxidizing and reducing conditions

1.3.2 Chemical Looping Combustion with Solid Fuels

While CLC was first developed for use with gaseous fuels, there is also interest to use solid fuels in CLC systems because solid fuels have a higher carbon content and emit more CO₂ per heat released. The two most prominent configurations being considered for solid fuel CLC are: 1) Direct CLC and 2) Indirect CLC. These two configurations, illustrated in Fig. 1.5 and discussed in the subsequent text, rely on the same two reactor architecture described above, however these configurations
must also have systems to address 1) ash production, 2) removal of sulfur and chlorine impurities that are present in solid fuels and 3) char burn-out.

The primary difference between direct and indirect CLC systems is how the fuel is supplied to the fuel reactor. In direct CLC systems, the solid fuel is supplied to the fuel reactor where it can react with the oxygen carrier to form combustion products, ash, and the reduced metal oxide. In direct CLC systems, the solid fuel and oxygen carrier are fluidized by mixtures of CO\(_2\) and H\(_2\)O, recirculated from the exhaust of the fuel reactor. Because the solid fuel is directly fed to the fuel reactor, some of the sulfur and chlorine impurities in it can react with the metal oxide.

In indirect CLC systems, the solid fuel and oxygen carrier never mix because the solid fuel is first gasified to a syngas. The syngas is supplied to the fuel reactor where
it reacts with the oxygen carrier to form combustion products and the reduced metal oxide. Because the solid fuel is first gasified, ash and impurities common in solid fuels like sulfur and chlorine never mix with the oxygen carrier. The energy needed for gasification in indirect CLC systems will come from inside the system (i.e., from the exothermic reactions in the air reactor) and as a result indirect CLC system are expected to be difficult to implement and likely less efficient than a direct CLC system due to this extra heat transfer step [18, 20]. Despite this, indirect systems are still of interest because they avoid a number of engineering challenges that arise from the direct CLC system architecture, among them, ash separation from the metal oxide. Before direct CLC systems can be developed at a practical scale, issues related to ash accumulation, contamination from sulfur and chlorine impurities, and carbon burnout must be addressed.

All solid fuels contain some mineral matter and when the solid fuels are burned, the mineral matter is converted to ash. The ash content can be as high as 40% by weight of the initial fuel and if it is not removed from the air and fuel reactors, it will accumulate in the system and possibly even react with the oxygen carriers, resulting in the de-activation of the oxygen carrier [21, 24]. Although it has not been tested at scale, several research groups have hypothesized that since coal ash has a lower density than most oxygen carriers, it will be easy to separate the ash and oxygen carriers with a standard cyclone [22, 23].

If the ash reacts with the oxygen carrier, it might result in the oxygen carriers not transferring the full amount of oxygen between the air and fuel reactors or lead to oxygen carrier de-fluidization. Several studies have also been performed to investigate the potential of these side reactions. Siriwardane et al. [23] used the software package HSC Chemistry 6 to investigate the thermodynamic potential for the major constituents in ash (i.e., Al₂O₃, SiO₂) to react with various oxygen carriers including iron, nickel, manganese and copper. This study found that formation of aluminates and silicates is thermodynamically feasible when copper, nickel and manganese based oxygen carriers are used and is less likely to occur with iron based oxygen carriers. To further investigate the potential interaction between ash and the oxygen carriers, Siriwardane et al. [23] heated nickel, iron and manganese based oxygen carriers with
1.3. CHEMICAL LOOPING COMBUSTION

fly ash in a one-to-one weight ratio, to 1000 °C under dry air for 4 hours. Similar experiments were conducted with a copper based oxygen carrier and fly ash and the experiments with copper were conducted at 900 °C. No new phases were observed after in the copper or iron samples after this 4 hour thermal treatment indicating no major challenges with ash-oxygen carrier interactions [23]. Rubel et al. [25, 26] also investigated the potential for interactions between ash and the oxygen carrier by cyclically oxidizing and reducing mixtures of an iron based oxygen carrier and coal ash in a TGA at 950 °C. In their experiments, oxidation occurred with air and reduction with a syngas comprised of 10% H₂, 15% CO, 20% CO₂, the balance being argon. Cycling the mixture of iron and fly ash was found to have no detrimental impact on the oxygen transport capacity of the iron as an oxygen carrier [25, 26]. In fact, the presence of ash contributed to the net transport of oxygen during the cycling experiments. These experiments indicate that ash interactions with the fuel will be limited but most authors agree that larger scale tests may be necessary [21, 23, 25, 26].

The second challenge with direct CLC systems is that coal and biomass often contain species like sulfur and chlorine. The sulfur and chlorine can react with the metal oxide in the fuel reactor and lead to oxygen carrier deactivation [18, 27, 28]. Oxygen carrier deactivation from sulfur and chlorine will depend on the base metal being used in the system, and the rates of formation and decomposition of the deactivated oxygen carrier species. In the case of copper under fuel rich conditions (i.e., the fuel and oxygen carrier equivalence ratio is greater than 1) thermodynamic equilibrium predicts that the dominant sulfur species will be Cu₂S and CuSO₄, and the dominant chlorine species CuCl₂ [27]. If these species are formed in the fuel reactor, equilibrium calculations predict that these species will decompose in the highly oxidizing environment of the air reactor, releasing SO₂ or SO₃ and Cl₂ gas. To verify this experimentally, Forero et al. [28] performed experiments with methane seeded with H₂S as high as 1300 ppm, in a 500 W th CLC system with copper was the oxygen carrier. In this system, the fuel reactor was at 1073 K and complete combustion of the methane to CO₂ and H₂O was observed when the oxygen carrier-to-fuel mass ratio was greater than 1.5. Under these conditions, sulfur was present in the flue gas stream of both the air and fuel reactors in the form of SO₂. At higher oxygen carrier-to-fuel ratios,
more of the sulfur species was present in the exhaust of the fuel reactor. Only when
the oxygen carrier to fuel ratio was less than 1.5 did $\text{Cu}_2\text{S}$ form in the fuel reactor,
and when it formed it was regenerated in the air reactor as predicted by equilibrium.
This indicates that Cu-based CLC systems should not be limited by copper reacting
with sulfur forming de-activated oxygen carrier.

The final challenge considered in this work with developing direct CLC systems is
ensuring complete combustion of the fuel in the fuel reactor. If complete combustion
does not occur in the fuel reactor, the solid fuel will be transferred to the air reactor
where it will be burned and result in $\text{CO}_2$ emissions. While this will not prevent
the operation of CLC systems, it does lower their $\text{CO}_2$ capture efficiency. To ad-
dress this potential $\text{CO}_2$ emissions, extra process units including a char stripper have
been suggested to ensure complete combustion of the fuel \cite{18}. Ultimately, in order
to optimally design fuel reactors and ensure complete fuel combustion, the kinetics
occurring in the fuel reactor must be understood.

Since CLC became an active area of research, there have been many experimen-
tal studies that demonstrated the oxidation of gaseous fuels with oxygen carries is
sufficiently fast and will not limit the overall system operation \cite{18,20,29,31}. With
respect to solid fuels, it is expected that when coal or biomass is fed directly into
the fuel reactor, the volatile matter in solid fuel will react quickly with the oxygen
carrier to form combustion products. Unlike the volatile matter, the char component
of the fuel will not react directly with the oxygen carrier. While a solid-solid pathway
involving reactions between the oxygen carrier and the char is possible, it is unlikely
to be a significant contributor to char oxidation due to the small amount of external
surface area of the oxygen carrier and solid fuel \cite{22}. Instead, researchers expected
that the char will be gasified with the fluidization gases $\text{CO}_2$ and $\text{H}_2\text{O}$ present in the
fuel reactor to form a syngas and this syngas will react with the oxygen carriers to
form combustion products as illustrated in Fig. 1.6a \cite{18}. In this way, gasification is
expected to be the rate limiting step in fuel reactors of direct CLC systems \cite{18,32,33}.

The gasification rate of a char in the fuel reactor will depend on many different
variables, including the concentrations of $\text{CO}_2$, $\text{H}_2\text{O}$, which are gasifying agents, and
the concentrations of $\text{H}_2$ and $\text{CO}$, which are species that inhibit gasification (slow
1.3. CHEMICAL LOOPING COMBUSTION

(a) Gasification Pathway
(b) Oxygen Release Pathway

Figure 1.6: Kinetic Pathways for solid fuel combustion in a fuel reactors of CLC systems

gasification rates) \[18, 34\]. Ultimately gasification is characterized by heterogeneous surface chemistry and can physically be described by adsorption of gas phase species, surface reactions and desorption of surface species \[35\]. Char gasification, at relevant temperatures to CLC fuel reactors (800 to 1200 °C), is quite slow and can take minutes to an hour for complete burn out of a 100 µm diameter particle \[18, 33\].

More recently, Mattisson et al. \[29\] hypothesized a second pathway that applies to a limited subset of metal oxides, where oxygen in the metal oxide can react directly with the fuel. As a result, char conversion is limited by oxidation chemistry which is orders of magnitude faster than gasification chemistry. This pathway involves a metal oxide releasing its oxygen and the oxygen reacting directly with the fuel. A metal oxide can release its oxygen forming a lower oxidation-state oxide or metal to minimize the Gibbs free energy of the system and this decomposition occurs at elevated temperatures (approaching the melting point of the oxide) and at low oxygen activities. The conversion pathway describing oxygen release from the oxygen carrier is illustrated in Fig 1.6b and oxygen carriers that are capable of decomposing at temperatures relevant to CLC systems are known as Chemical Looping Oxygen Uncoupling (CLOU) oxygen carriers.

Many researchers \[36–42\] believe that higher char conversion rates can be achieved by using CLOU oxygen carriers. When a CLOU oxygen carrier is combined with gaseous and solid fuels at relevant temperatures for oxygen release, the free oxygen will
likely first react with the gaseous fuel species, which will increase the concentrations of CO$_2$ and H$_2$O (gasification agents) and decrease the concentration of CO and H$_2$ (gasification inhibitors). Consequently, the conversion rate of the char will increase. If the oxygen concentration is sufficiently high to completely oxidize the gaseous fuel species, gas phase O$_2$ can even react directly with the char $^{29,37}$. As a result, the char conversion rate will be faster than it normally would be if the char was gasified $^{21,29,37,39}$. Since char conversion to combustion products increases, the size and cost of fuel reactors are expected to be reduced for CLOU oxygen carriers.

The driving force for oxygen release from the oxygen carrier is controlled by the difference in chemical potential between surface oxygen and the effective oxygen activity at the metal oxide/sub-oxide (e.g., CuO/Cu$_2$O) or metal oxide/metal interface (e.g., Cu/Cu$_2$O). For a particular metal oxide (e.g., CuO, MnO, Fe$_2$O$_3$) at a specified temperature, the threshold ambient oxygen pressure required for oxygen release is given by the equilibrium gas-phase oxygen pressure between the metal oxide (Me$_a$O$_b$) and the next lowest oxidation state (Me$_a$O$_{b-1}$), defined by Eq. 1.3a. In this expression, $\Delta G$ is defined by Eq. 1.3b and is the free energy change for Eq. 1.1a. For the copper system, the equilibrium between CuO and Cu$_2$O will determine this oxygen pressure (e.g. 2·CuO ↔ Cu$_2$O + 1/2O$_2$). When the oxygen pressure in the fuel reactor is less than the effective equilibrium oxygen pressure for the metal oxide, oxygen release will occur until the actual oxygen pressure equals its equilibrium value.

$$\frac{1}{P_{O_2}/P_{O_2}^{ref}} = exp\left(-\frac{2 \cdot \Delta G_o}{RT}\right) \quad (1.3a)$$

$$\Delta G_i = \mu_{Me_aO_b} - \frac{1}{2} \cdot \mu_{O_2} - \mu_{Me_aO_{b-1}} \quad (1.3b)$$

The effective equilibrium oxygen pressure as a function of temperature for several oxides/suboxides is shown in Fig. 1.7. From Fig. 1.7 it can be seen that all metal oxides exhibit some oxygen release capacity, however the equilibrium oxygen mole fraction for most metal oxides only becomes appreciable at temperatures roughly greater than 80% of their melting point. The red region in Fig. 1.7 demarcates expected fuel reactor temperatures (i.e., 800 to 1200 °C) and shows that CuO and
MnO have the highest oxygen mole fraction at these temperatures. CuO, MnO along with CoO (not shown in the figure) are the most prominent CLOU oxygen carriers studied by the CLC community \[32, 36\]. Iron and nickel oxides have equilibrium oxygen concentrations less than \(10^{-4}\) in this temperature window and therefore are not considered CLOU materials. Under CLOU operating conditions, copper based oxygen carriers cycle between CuO and Cu\(_2\)O and manganese based oxygen carriers between Mn\(_2\)O\(_3\) and Mn\(_3\)O\(_4\). From Table 1.2 it can be seen that this limits their oxygen carrier capacity in half and as a result, CLOU systems require a larger oxygen carrier inventory.

![Equilibrium oxygen partial pressure vs temperature](image)

Figure 1.7: Equilibrium oxygen partial pressure vs temperature

Experimental evidence supporting the idea that CLOU materials convert char to combustion products faster than non-CLOU materials was first presented by Mattisson, Leion and Lyngfelt from Chalmers University \[33, 43, 44\] and since then, many other groups have also performed experiments with CLOU materials (mostly Cu and Mn based OCs) \[36, 42\]. In the initial studies performed at Chalmers, an iron and copper based oxygen carrier were fluidized in a batch reactor that was cycled between
oxidizing and reducing conditions. Oxidation occurred with air and reduction with a Mexican petroleum coke. The oxygen carriers used in the experiments were fluidized with either pure N\textsubscript{2} or mixtures of N\textsubscript{2} and steam, and a fixed amount of a petroleum coke was fed into the reactor during the reduction phase. The exhaust gases were measured as a function of time using a continuous gas analyzer. Once the fuel was completely converted to a syngas or combustion products, air was introduced into the reactor to re-oxidize the oxygen carrier to its oxidized state. The iron based oxygen carrier considered was composed of 60% Fe\textsubscript{2}O\textsubscript{3} by weight supported on MgAl\textsubscript{2}O\textsubscript{3} and the copper based oxygen carrier 40% CuO supported on ZrO\textsubscript{2}. A summary of the experiments and the materials used in them is listed in Table 1.3.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>60% Fe\textsubscript{2}O\textsubscript{3}/MgAl\textsubscript{2}O\textsubscript{3}</td>
<td>950 °C</td>
<td>20 g</td>
<td>90 to 125 µm</td>
<td>0.2 g</td>
<td>125 to 180 µm</td>
</tr>
<tr>
<td>44</td>
<td>40% CuO/ZrO\textsubscript{2}</td>
<td>880 and 985 °C</td>
<td>15 g</td>
<td>125 to 180 µm</td>
<td>0.1 g</td>
<td>180 to 250 µm</td>
</tr>
</tbody>
</table>

Table 1.3: Summary of experimental conditions from \cite{29,33,43,44}

At 950 °C, the iron based oxygen carrier required 49.1 minutes to convert 95% of the coke to a syngas with pure N\textsubscript{2} (zero steam) and 8.4 minutes with a mixture of 50% steam/50% N\textsubscript{2}. The dry exhaust gases measured during these experiments were a mixture of CO, CO\textsubscript{2}, and H\textsubscript{2}. While the exhaust of this reactor had CO and H\textsubscript{2}, it serves as an upper limit to the gasification rate expected in fuel reactors when solid fuels are used.

For the copper experiments, the oxygen carrier was fluidized in pure N\textsubscript{2}. At 850 and 985 °C. Under these two temperatures, the equilibrium oxygen mole fraction between Cu\textsubscript{2}O and CuO is 1% and 8%, respectively. Considering the 850 °C experiment, prior to the introduction of the char while the reactor was at temperature, the OC in these experiments released O\textsubscript{2} to maintain the O\textsubscript{2} mole fraction in the flue gas of the reactor at 1%. Once the char was fed into the reactor, the oxygen concentration
measured by the continuous analyzer dropped to around 0.9% and once the fuel was completely consumed (determined when no CO\textsubscript{2} was observed in the flue gas), the O\textsubscript{2} mole fraction returned to its equilibrium value of 1%. During this period, no CO was measured from the continuous analyzer. A carbon balance was also performed on the measured CO\textsubscript{2} and verified the complete conversion of the petroleum coke \cite{44}. At 880 °C, it took 2 minutes for 95% of the char to be converted to CO\textsubscript{2}. Similarly, at 980 °C, with an equilibrium oxygen mole fraction of 10%, it took 20 seconds to convert 95% of the char to CO\textsubscript{2}.

By using the CLOU approach, the conversion rate of the char was significantly faster than it would have been with a non-CLOU oxygen carrier. This is important because a higher conversion rate of the char can allow for a smaller fuel reactor. However, to achieve the increased rates associated with CLOU oxygen carriers, a CuO to char ratio of 60 was required. This is a factor of 4.9 times larger than the stoichiometric amount of CuO required to oxidize the char. Such a high oxygen carrier inventory was needed in these experiments because oxygen carrier conversion was limited between CuO and Cu\textsubscript{2}O. While it is apparent that char conversion rates with CLOU oxygen carriers are faster than with non-CLOU oxygen carriers at conditions relevant to CLC, CLOU-CLC system require a significantly larger oxygen carrier inventory. Consequently, there is likely an optimum oxygen carrier inventory that will minimize the size of the fuel reactor (e.g., minimize the oxygen carrier inventory and maximize the fuel conversion rate). Modeling the kinetics of CuO and solid fuels (syngas and char) will be necessary to understand this tradeoff and design an optimized CLC system.

1.4 Oxidation and Reduction Kinetics

One of the primary challenges in the advancement of CLC systems is designing oxygen carriers with sufficiently fast oxidation and reduction kinetics that will minimize the size of air and fuel reactors. There are many different variables that can influence observed kinetic rates of oxygen carriers including OC morphology, support material, diameter, porosity, and percent oxygen carrier. Several phenomenological models have
previously been proposed from within the CLC community to describe oxidation and reduction of oxygen carriers in CLC systems with the three most prominent being the shrinking-core model (SCM), the changing grain size model (CGSM), and the nucleation and nuclei growth model [18, 30, 45]. While these models can be used to characterize oxidation data over a limited set of experimental conditions, they are global in nature and have had limited success in being applied as predictive tools [18]. As a result, more fundamental models that are capable of predicting process variables like the system oxygen carrier loading, OC recirculation rate (residence time), and minimum air and fuel flowrate as a function of the aforementioned morphological parameters need to be developed.

Under the field of corrosion science, the high temperature oxidation of metals has been studied since the early 1900s. From these studies of metal oxidation, it was observed that the thickness of a metal oxide layer grows proportional to the square root of time. This is due to the fact that oxidation is inherently a diffusion limited process. This observation, denoted parabolic growth, was first explained by Carl Wagner in the 1930s as the result of the diffusion of lattice defects in the growing oxide layer [46–48].

In the case of Cu oxidation, globally described by Eq. 1.4a and 1.4b, separate, non-porous, Cu$_2$O and CuO phases form. The Cu$_2$O and CuO oxide layers form in the order of their oxygen activities: a CuO layer forms as the outer most layer and a Cu$_2$O layer forms on top of the Cu core. The Cu$_2$O growth rate is characterized by the diffusion of neutral or singly charged copper vacancies at high oxygen concentrations (e.g., conditions in the air reactor) and by copper interstitials and oxygen vacancies at low oxygen concentrations (at oxygen pressures when CuO does not form) [49–52]. The oxidation of Cu$_2$O to CuO (Eq. 1.4a) is less characterized than Cu conversion to Cu$_2$O by the metallurgic community; however, it is believed that conversion of Cu$_2$O to CuO is limited by the diffusion of doubly charged copper vacancies [53].

\[
\begin{align*}
\text{Cu} + \frac{1}{4} \cdot \text{O}_2 & \leftrightarrow \frac{1}{2} \cdot \text{Cu}_2\text{O} \hspace{2cm} (1.4a) \\
\text{Cu}_2\text{O} + \frac{1}{2} \cdot \text{O}_2 & \leftrightarrow 2 \cdot \text{CuO} \hspace{2cm} (1.4b)
\end{align*}
\]
1.4. OXIDATION AND REDUCTION KINETICS

The reduction of metal oxides is characterized by the removal of lattice oxygen via a reducing agent like fuels (*e.g.* CO, H₂, CH₄), or by the liberation of oxygen to the gas phase. While oxidation of metals has been well characterized, the atomistic processes occurring during the reduction of metal oxides are still not well understood [54–56]. Generally, reduction of a metal oxide particle with fuel gases requires the adsorption of the fuel species on the oxide surface, followed by the surface species reacting with oxygen resulting in the nucleation and growth of sub-oxide or metal. After the nuclei grow and cover the entire surface, the core metal oxide will shrink in size as the sub-oxide and metal layers grow until the particle is completely reduced.

In the case of CuO reduction, which has several possible oxidation states, there are two likely global pathways for reduction: a direct pathway given by Eq. (1.5a) and an indirect pathway given by Eqs. (1.5b) and (1.5c):

\[
\begin{align*}
\text{CuO} + \text{fuel} & \rightarrow \text{Cu} + \text{Combustion Products (i.e., CO}_2 + \text{H}_2\text{O)} \\
\text{CuO} + \text{fuel} & \rightarrow \frac{1}{2}\text{Cu}_2\text{O} + \text{Combustion Products (i.e., CO}_2 + \text{H}_2\text{O)} \\
\text{Cu}_2\text{O} + \text{fuel} & \rightarrow 2\text{Cu} + \text{Combustion Products (i.e., CO}_2 + \text{H}_2\text{O)}
\end{align*}
\]

(1.5a)

(1.5b)

(1.5c)

The actual rate of reduction and pathway for CuO reduction will depend on many factors including (1) the presence of surface defects, (2) grain boundaries, (3) orientation of the exposed surfaces, (4) presence of support or other metals, and (5) presence of other surface groupings like hydroxyl or carbonates [54].

There is concern from the CLC community that metallurgic theories are not applicable to describe oxidation of OCs with the reasons being that metallurgic theories were developed with samples of a different form, (plates vs powders) and the theories are developed with fresh materials, instead of cycled metals and metal oxides [18]. While OC samples are of a different physical form and are oxidized and reduced many times in CLC systems, these differences are not enough to change the physical processes occurring during oxidation. Much can be learned by understanding and applying models based in these fundamental processes and they can be used to guide the development of OC for use in CLC systems.
1.5 Thesis Objectives

The overall objective of this thesis is to apply a mechanistic understanding of Cu oxidation with air and CuO reduction with various fuels to predict reaction conversion rates of oxygen carriers in air and fuel reactors of CLC systems. Once these models are presented, the understanding from them will be used to estimate the size of air and fuel reactors and estimate the copper and air flow rates needed to oxidize a given amount of fuel and produce a specified amount of electricity.

Each chapter of this thesis will be written with this goal in mind. Chapter 2 will present a fundamental kinetic model that describes oxidation rates in high temperature oxygen rich environments. Chapter 3 will discuss the application of the kinetic model presented in Chapter 2 to predict oxidation rates of a supported oxygen carrier. Chapter 4 will review metal oxide reduction models and present experimental work on Cu$_2$O and CuO reduction in a flow reactor and TGA, and Chapter 5 will apply both kinetic models to estimate the size and flow rates of CLC systems. Finally, Chapter 6 will provide a summary of results from these modeling efforts and will provide suggestions for areas of future research.
Chapter 2

Copper Oxidation Chemistry

Chapter 2 presents a theoretical model describing Cu oxidation along with accompanying Cu oxidation data collected from experiments performed in a flow reactor and a thermogravimetric analyzer (TGA). The model is characterized by defect chemistry and Wagner’s Theory on defect diffusion. The experimental data is analyzed with the presented model and the model is used to extract defect diffusion coefficients from the flow reactor data. The experimental data and the extracted defect diffusion coefficients are then compared to previously published oxidation data and finally the sensitivity of the model output to given input parameters is discussed.

2.1 Literature Review on Cu Oxidation

The oxidation of metals has been studied since the early 1900s with the purpose of understanding how materials corrode in high-temperature, oxygen-rich environments. In these early investigations it was observed that the thickness of the metal oxide layer grew proportional to the square root of time. This phenomenon, known as parabolic growth, was first explained by Carl Wagner in the 1930s [46–48], and in the case of Cu oxidation to Cu$_2$O, understood to be a result of the outward diffusion of Cu atoms via a vacancy mechanism [46, 57–59]. Since then, the understanding of Cu oxidation to Cu$_2$O has grown significantly, indicating that Cu$_2$O is a p-type semiconductor that is dominated by neutral or singly charged copper vacancies.
high oxygen pressures, near or above the Cu$_2$O / CuO equilibrium oxygen pressure at a given temperature, and by copper interstitials or oxygen vacancies at low oxygen pressures, approximately near or equal to the Cu / Cu$_2$O equilibrium oxygen pressure at a given temperature \[49\]-\[52\].

Unlike the agreement in the literature describing defect chemistry in Cu$_2$O, little consensus exists on the defect structure of CuO and the oxidation mechanism of Cu$_2$O to CuO. Researchers have reported both an oxygen deficiency, CuO$_{1-x}$ with \(2 \cdot 10^{-5} < x < 0.009\) \[60\]-\[64\], as well as a copper deficiency, Cu$_{1-x}$O with \(x < 0.001\) \[53\]-\[62\]-\[65\]. The dominant defects in these two cases are copper interstitials and oxygen vacancies \[60\]-\[61\]-\[64\] or copper vacancies and oxygen interstitials \[53\]-\[62\]-\[63\]-\[65\]-\[66\], respectively.

In a review by Carel et al. \[60\], four data sets \[61\]-\[62\]-\[64\]-\[65\], which represent much of the published studies on CuO non-stoichiometry, were re-analyzed to reconcile the differences in the literature. From this re-examination, Carel et al. concluded that oxygen vacancies or copper interstitials, or clusters of oxygen vacancies and copper interstitials are the dominant defects in CuO, rejecting data supporting a copper vacancy dominated defect structure due to the high level of impurities present in the samples corroborating a copper deficiency.

More recently, Grzesik and Migdalska \[53\] investigated Cu$_2$O oxidation to CuO with thermogravimetric and marker experiments. In these experiments, Cu$_2$O samples were made from high purity Cu (99.9999 wt.%) under carefully selected conditions to ensure consistent initial defect state and grain structure (coarse grains approximately 50 µm) and then oxidized for over 100 hours. Grzesik and Migdalska observed non-parabolic growth for oxidation times less than 25 hours and parabolic growth for oxidation times greater than 25 hours. Their experimental findings from the long time oxidation experiments indicate that CuO is copper deficient, Cu$_{1-x}$O with \(x \leq 10^{-5}\), where the dominant defects are doubly ionized copper vacancies. Furthermore, by assuming the hole and electron concentrations in CuO to be equal, as was reported by Park and Natesan \[67\], Grzesik and Migdalska were also able to present a mechanism for Cu$_2$O oxidation to CuO as well as the self-diffusion coefficient for Cu atoms in CuO.
In addition to an oxidation mechanism by lattice diffusion, oxygen diffusion along grain boundaries can also contribute to the overall oxidation rate of copper. Grain boundary diffusion provides a parallel, lower resistance pathway for copper or oxygen atoms to diffuse through the Cu2O and CuO layers \[68-70\]. Zhu et al., \[71-73\] studied Cu2O oxidation to CuO and found that grain boundary diffusion influences oxidation at temperatures below 600 °C and is less important at higher temperatures. Since grain boundary diffusion is only expected to influence rates at temperatures less than 600 °C, it was not considered in the present analysis.

In this chapter, Cu oxidation data obtained in experiments performed in a flow reactor and in a thermogravimetric analyzer (TGA) are analyzed, a defect model is constructed, and defect-diffusion coefficients in Cu2O and CuO are extracted. The chapter is organized into four sections: the first section describes the theoretical framework in which the Cu/Cu2O/CuO/O2 system is modeled; the second section discusses the experimental procedure; the third section addresses model results and presents diffusion coefficients of defects in Cu2O and CuO; and the fourth section discusses results related to previously published diffusivity data and addresses sensitivity of the model output to the given input parameters.

### 2.2 Theoretical Approach

Three processes were considered when describing Cu oxidation: 1) chemisorption-desorption of oxygen on Cu surfaces, 2) nucleation and coalesce of oxide islands on the surface and 3) growth of Cu2O and CuO. Of these three processes, oxide layer growth is the most important to describe the conversion of Cu to Cu2O and CuO under the conditions relevant to CLC (6% to 21% O2 and T > 700 °C). At these conditions, the timescales for oxygen chemisorption-desorption and oxide nucleation are quite short relative to the timescale for complete oxidation of Cu to CuO, and oxide nucleation is essentially complete before significant oxide layer growth can occur. For these reasons, chemisorption-desorption and nucleation processes have been neglected in the theoretical modeling framework presented in this paper. Consequently, in order to begin the calculations, Cu2O and CuO oxide layers, each 10 nm thick (about the
copper oxide thickness observed in nucleation studies on single crystal Cu [74, 75],
are assumed to be initially present on the copper surface. This thickness in the model
is of inconsequential significance with respect to the oxidation of copper.

As in Wagner’s theory, the presented model assumes that crystalline defects 1) form an ideal solution in the oxide layer (i.e., non-interacting), 2) are in local chemical
equilibrium at a given phase interface, and 3) have fluxes proportional to their elec-
trochemical potential (\(\eta\)) gradients \[47, 76, 78\]. While the determination of dominant
lattice defects and their heats of reaction in Cu\(_2\)O and CuO is still an active area
of research, most researchers agree that defects in copper oxides exist in the cation
sub-lattice \[49, 53, 79, 81\].

The defects considered within the framework of the model in cuprous oxide, written
in their Kröger-Vink format and presented in Table 2.1, are neutral and singly
charged copper vacancies (\(V^x_{Cu}\) and \(V'_{Cu}\)), neutral and singly charged copper intersti-
tials (\(Cu^x_i\) and \(Cu'_i\)), electrons (\(e'\)), holes (\(h'\)), and singly charged positive impurity
atoms (\(I_{Cu_2O}\)). Negative impurity atoms were not considered because they are un-
likely to be present in Cu\(_2\)O. Oxygen vacancies in cuprous oxide were also neglected
in the model put forth because at the high oxygen activities of interest, the oxygen
vacancy concentration is expected to be quite small compared to the copper vacancy
concentration, and the oxygen vacancy diffusion coefficient is expected to be quite
low compared to those of copper vacancies \[49, 50\]. The defects considered to exist
in cupric oxide are doubly ionized copper vacancies (\(V'_{Cu}\)), doubly ionized copper
interstitials (\(Cu''_i\)), electrons (\(e'\)), holes (\(h'\)) and singly charged positive and negative
impurity atoms, (\(I_{CuO}\) and \(I'_{CuO}\)), respectively.

Table 2.1: Key defects in the Cu\(_2\)O and CuO layers during Cu oxidation

<table>
<thead>
<tr>
<th>Layer</th>
<th>Defects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cu(_2)O Layer</td>
<td>(V^x_{Cu}), (V'_{Cu}), (Cu^x_i), (Cu'<em>i), (e'), (h'), (I</em>{Cu_2O})</td>
</tr>
<tr>
<td>CuO Layer</td>
<td>(V''<em>{Cu}), (Cu''<em>i), (e'), (h'), (I</em>{CuO}), (I'</em>{CuO})</td>
</tr>
</tbody>
</table>

Impurity atoms can exist as substitutional or interstitial defects in Cu\(_2\)O and
CuO. Depending on the concentration, ionic radius, oxidation state and electronic
structure, the presence of impurity atoms can alter (1) the reaction energy to form defects, (2) defect concentrations, (3) the oxidation rates and (4) even change the band gap locally [82, 83]. The model presents a simplified account of impurity atoms in that only the net cumulative charge of impurity atoms with respect to regular copper atoms in Cu₂O or CuO is considered. In this way, the primary influence of impurities on oxidation rates is through their effect on defect concentrations, governed by a charge neutrality condition. Following the work of Stoklosa [49], cation impurities in Cu₂O and CuO layers are assumed to only influence the Cu sublattice, form an ideal solution with the other defects, not influence the reaction entropy or enthalpy of the other defects in Cu₂O and CuO, and be uniformly distributed and immobile in the Cu₂O and CuO layers.

2.2.1 Equilibrium Defect Concentrations

The equilibrium defect concentration at the Cu/Cu₂O, Cu₂O/CuO and CuO/O₂ interfaces are calculated using the pertinent defect reactions along with the corresponding values for standard enthalpy and entropy change presented in Table 2.2. These thermochemical properties were either obtained from the literature [49, 53] or estimated, as described below. The equilibrium constants for the vacancy and interstitial type defect reactions are calculated from the thermochemical properties: \( K_j = \exp(-\Delta G_j^o/RT) \), where \( \Delta G_j^o = \Delta H_j^o - T \cdot \Delta S_j^o \). The equilibrium constant for hole-electron equilibrium in Cu₂O (Rx. 5) is calculated based on the fundamentals of solid state physics:

\[
K_{e^-h^+} = N_V \cdot N_C \cdot e^{-E_g/k_B T} \quad (2.1)
\]

Here, the effective density of electrons and holes in the conduction \( (N_C) \) and valence \( (N_V) \) band, respectively, are given by:

\[
N_C = \left( \frac{2 \cdot \pi \cdot m_e^* \cdot k_B \cdot T}{\hbar^2} \right)^{3/2} \quad \text{and} \quad N_V = \left( \frac{2 \cdot \pi \cdot m_h^* \cdot k_B \cdot T}{\hbar^2} \right)^{3/2} [1/cm^3] \quad (2.2)
\]
CHAPTER 2. COPPER OXIDATION CHEMISTRY

In this approach, the effective mass of holes \( m_{h}^{*} \) and electrons \( m_{e}^{*} \) in cuprous oxide are taken to be 0.84 and 0.61 \[51\], respectively, and the band gap of cuprous oxide is taken to be temperature dependent, expressed in the form \( E_G(T) = E_G^0 - \frac{\alpha T^2}{\beta + T} \), where \( \alpha = 4.8 \times 10^{-4} \) eV/K and \( \beta = 275 \) K \[84\].

In this work, CuO is treated as a charge transfer insulator \[66, 85, 86\]. Within charge transfer insulators, holes (i.e., missing electrons) are localized on 2p orbitals of oxygen ions and electrons are localized on 3d orbitals of copper ions \[87, 88\]. This is unlike semiconductors (e.g., Cu_{2}O) where electrons are delocalized in the conduction band and holes, in the valence band. From semiconductor theory, the concentration of bound electrons on copper ions and bound holes on oxygen ions is given by Eq. \[2.3\], where \( N_{Cu} = N_{Oxy} = N_s \) are the atomic number density of copper and oxygen in CuO. The factor of 6 arises in the expression for the concentration of holes because there are six outer electrons on an oxygen ion that can be removed resulting in the creation of a hole, whereas each copper ion has only one open d-orbital for an electron and so the factor for electrons bound to Cu ions is 1.

\[
[e'] = N_{h} \cdot exp \left\{ -\frac{(E_d - E_F)}{k_B \cdot T} \right\}, \tag{2.3a}
\]

\[
[h'] = 6 \cdot N_s \cdot exp \left\{ -\frac{(E_F - E_p)}{k_B \cdot T} \right\} \tag{2.3b}
\]

The energy required to excite an electron from an oxygen p-orbital \( (E_p) \) to a copper d-orbital \( (E_d) \) is given by the band gap \( (E_g) \) of CuO, which has been measured to be between 1 and 1.9 eV \[86, 87, 89, 92\]; in the presented calculations a value of 1.7 eV is used. Knowing these parameters, the equilibrium constant for electron holes in CuO (Rx. 8) can be estimated using Eq. \[2.4\].

\[
K_{e'-h'} = 6 \cdot N_s^2 \cdot exp \left\{ -\frac{E_g}{k_B \cdot T} \right\} \tag{2.4}
\]
Table 2.2: Defect reactions, equilibrium constants and formation energies for defects in Cu₂O and CuO

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>V'ₐCu</td>
<td>(\frac{1}{4}O₂ \leftrightarrow \frac{1}{2}O_D' + V'ₐCu + h^+)</td>
<td>(K_{V'ₐCu} = \frac{[h^+][V'ₐCu]}{P_{O_2}})</td>
<td>94 ± 3</td>
<td>-41 ± 3</td>
<td>[49]</td>
</tr>
<tr>
<td>2</td>
<td>VₐCu</td>
<td>x</td>
<td>h^- + VₐCu ← VₐCu</td>
<td>x</td>
<td>(K_{VₐCu</td>
<td>x} = \frac{[VₐCu</td>
</tr>
<tr>
<td>3</td>
<td>Cu_i</td>
<td>(\frac{1}{2}O_D' + Cu + h^- \leftrightarrow Cu_i^+ + \frac{1}{4}O_2)</td>
<td>(K_{Cu_i} = \frac{P_{O_2}^{1/4}[Cu_i]^+}{[h^-][Cu_i]})</td>
<td>103 ± 3</td>
<td>57 ± 2</td>
<td>[49]</td>
</tr>
<tr>
<td>4</td>
<td>Cu_i</td>
<td>x</td>
<td>Cu_i^+ + e^- ← Cu_i</td>
<td>(K_{Cu_i^+} = \frac{[Cu_i]^+}{[Cu_i^+]^+[e^-]})</td>
<td>-132 ± 2</td>
<td>26 ± 1</td>
</tr>
<tr>
<td>5</td>
<td>e^-/h^-</td>
<td>null ← e^- + h^-</td>
<td>(K_{e^-/h^-} = [e^-] \cdot [h^-])</td>
<td>[-]</td>
<td>[-]</td>
<td>[84]</td>
</tr>
</tbody>
</table>

**CuO**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>V''ₐCu</td>
<td>(\frac{1}{2}O₂ \leftrightarrow O_D'' + V''ₐCu + 2h^-)</td>
<td>(K_{V''ₐCu} = \frac{[h^-][V''ₐCu]}{P_{O_2}})</td>
<td>250^+</td>
<td>-39^+</td>
<td>[-]</td>
</tr>
<tr>
<td>7</td>
<td>Cu_i^-</td>
<td>(O_D'' + Cu + 2h^- \leftrightarrow Cu_i^- + 1/2O_2)</td>
<td>(K_{Cu_i^-} = \frac{[Cu_i^-]^+P_{O_2}^{1/2}}{[h^-][Cu_i]})</td>
<td>260</td>
<td>75^+</td>
<td>[93]</td>
</tr>
<tr>
<td>8</td>
<td>e^-/h^-</td>
<td>null ← e^- + h^-</td>
<td>(K_{e^-/h^-} = [e^-] \cdot [h^-])</td>
<td>[-]</td>
<td>[-]</td>
<td>[-]</td>
</tr>
</tbody>
</table>

Defect concentrations are non-dimensionalized by the O atom concentration in either Cu₂O or CuO.

*e.g.,* \([V_{Cu}^{x}]^* = \frac{[V_{Cu}^{x}/Cu_{2O}]}{[O_{Cu_{2O}}]}\) where 
[Cu₂O] is the atom density (1/cm³) of oxygen in Cu₂O.

In the equilibrium constant relations, the activity coefficients of 
[Cu] and [O] are taken as unity and therefore not included.

^Estimated values

Uncertainty values were taken from publications.

Equilibrium defect concentrations were determined by combining the charge neutrality condition (Eq. 2.10) with the equilibrium relations listed in Table 2.2. The
standard enthalpy and entropy change associated with forming a copper vacancy \( (V'_{\text{Cu}}) \) and the entropy change associated with forming a copper interstitial \( (\text{Cu}^{\cdot \cdot}_i) \) in \( \text{CuO} \) are not available in the literature, so these thermochemical properties had to be estimated based on experimental observations and theory. The standard entropy change that occurs upon formation of a vacancy or interstitial is primarily associated with vibrational changes in the lattice. For many oxides (\( \text{Cu}_2\text{O}, \text{NiO}, \text{CoO}, \text{MnO} \)), this entropy change does not vary too significantly, where typical standard reaction entropies vary between \( 5 \times R \) (41 J/mole-K) to \( 13 \times R \) (108 J/mole-K) for interstitials, and between \( -2 \times R \) (-17 J/mole-K) and \( -7 \times R \) (-60 J/mole-K) for vacancies. While these materials have different lattices than \( \text{CuO} \), they provide a \( \Delta S_R \) that \( \text{CuO} \) is likely to conform. Therefore, average values from these ranges were used in the calculations.

To bound the range of the formation enthalpy of vacancies in \( \text{CuO} \), \( \Delta H_{R,V'_{\text{Cu}}} \), the equilibrium defect concentration was calculated using thermochemistry data from Table 2.2 as a function of \( \Delta H_{R,V'_{\text{Cu}}}^{\text{CuO}} \), and is plotted in Fig. 2.1. Vacancy atom fractions in \( \text{CuO} \) are thought to be low [53,60,64,72], and have reported atom fractions ranging from \( 10^{-8} \) to \( 10^{-5} \) [64,80] over the temperatures considered in this work. These defect concentrations roughly correspond to a reaction enthalpy of \( 240 < \Delta H_{R,V'_{\text{Cu}}}^{\text{CuO}} < 260 \) kJ/mole as indicated by the grayed area in Fig. 2.1. For the purpose of this model, a value of 250 kJ/mole is used for the reaction enthalpy.

The calculated equilibrium defect concentrations are shown in Fig. 2.2 for the \( \text{Cu}_2\text{O} \) layer and Fig. 2.3 for the \( \text{CuO} \) layer. In these calculations, the impurity concentrations were set to zero and their impact is assessed in Section 2.5.1, entitled Impact of Impurities. As indicated in Fig. 2.2, the dominant defects in \( \text{Cu}_2\text{O} \) are neutral and singly ionized vacancies and from Fig. 2.3, the dominant defects in \( \text{CuO} \) are doubly ionized vacancies. Since the interstitial concentration in \( \text{CuO} \) is many orders of magnitude smaller than the vacancy concentration, they are not included in the model. Note in Fig. 2.3 that when \( \text{CuO} \) forms, the ambient oxygen partial pressure only influences the defect concentration in \( \text{CuO} \) and not in \( \text{Cu}_2\text{O} \). As a result, when oxidizing \( \text{Cu} \) to \( \text{CuO} \), the initial oxidation rate will not be strongly influenced by the oxygen pressure. Also in Fig. 2.3, for a given oxygen partial pressure, the
2.2. THEORETICAL APPROACH

Figure 2.1: Number fraction of $V_{\text{Cu}}^\prime$, $e'$ and $h'$ as a function of $\Delta H_{\text{Cu}}^{V_{\text{Cu}}^\prime}$ in CuO at 500 °C (a) and 900 °C (b). The dark band represents the region of reaction enthalpies based on the expected defect concentrations in CuO.

dotted and solid lines converge where the CuO phase is no longer stable.

Figure 2.2: Lattice defect number fraction in the Cu$_2$O layer at the Cu/Cu$_2$O and Cu$_2$O/CuO interface (a) and electronic defects in the Cu$_2$O layer at the Cu/Cu$_2$O and Cu$_2$O/CuO interface (b). Dotted lines are the number fraction at the Cu$_2$O/CuO interface and solid lines for the Cu/Cu$_2$O interface.

2.2.2 Defect Transport Model

The flux of the defect species in the model across an oxide layer is described by Fickian diffusion, given by Eq. 2.5
CHAPTER 2. COPPER OXIDATION CHEMISTRY

Figure 2.3: a) Vacancy, hole and electron number fraction and b) copper interstitial number fraction in the CuO layer at the Cu$_2$O/CuO and CuO/O$_2$ interface. Dotted lines are number fraction of defects at the CuO/O$_2$ interface and solid lines for the Cu$_2$O/CuO interface.

\[
J_i = -\frac{[X_i] \cdot D_i}{k_B \cdot T} \cdot \frac{\partial \eta_i}{\partial r} = -D_i \cdot \frac{\partial [X_i]}{\partial r} + e \cdot N_a \cdot z_i \cdot \frac{[X_i] \cdot D_i}{k_B \cdot T} \cdot \frac{\partial \chi}{\partial r} \quad (2.5)
\]

When the electrochemical potential of species \( i \) (\( \eta_i \)) is expanded under the ideal solution assumption, it can be seen that the defect flux is comprised of two terms: the first is the flux contribution due to a concentration gradient and the second term is the flux contribution due to the gradient in the electrostatic potential (\( \chi \)). In this equation, \( D_i \) and \( [X_i] \) are the defect diffusion coefficient and defect concentration, respectively, of mobile defect species \( i \) listed in Table 2.1 (not including impurities which are immobile), \( e \) is the charge of an electron, \( z_i \) is the charge number, and \( k_B \) is Boltzmann’s constant.

The differential equations that govern the transport of the defect species across an oxide layer are given by Fick’s Second Law, and are written in Eq. 2.6a and 2.6b for diffusion across the Cu$_2$O and CuO oxide layers, respectively. For each layer, a second order partial differential equation in the defect species concentrations results when the divergence term is expanded. It should be noted that the timescales for the defect species concentrations to reach steady state are short compared to the timescales for oxide layer growth. So that in the calculations, the steady state defect concentration
profiles are used instead of solving defect concentration profiles concurrently as the oxide layers grow.

\[
\text{Cu}_2\text{O Oxide Layer: } \frac{\partial[X_i]}{\partial t} \nabla \cdot J_i = 0, \text{ for } i = V_{\text{Cu}}, V_{\text{Cu}^2}, V_{\text{Cu}^3}, h^\circ, e' \tag{2.6a}
\]

\[
\text{CuO Oxide Layer: } \frac{\partial[X_i]}{\partial t} \nabla \cdot J_i = 0, \text{ for } i = V_{\text{Cu}^2}, V_{\text{Cu}^3}, h^\circ, e' \tag{2.6b}
\]

### 2.2.3 Defect Concentration Profiles

The defect concentration profiles across each oxide layer as a function of time were calculated via integration of Eq. 2.6a and 2.6b. To permit the integration, defect diffusion coefficients determined from the data (discussed in \[\text{3.3 The Results and Data Analysis Section of this chapter}\]), were employed in the calculations. In this approach, the differential equations were non-dimensionalized. For defects in the \(\text{Cu}_2\text{O}\) layer, the normalization is \([X_i]^* = [X_i]/[O]_{\text{Cu}_2\text{O}}\) and in the \(\text{CuO}\) layer \([X_i]^* = [X_i]/[O]_{\text{CuO}}\), where \([O]_{\text{Cu}_2\text{O}}\) and \([O]_{\text{CuO}}\) are the atom densities \((1/\text{cm}^3)\) of oxygen in \(\text{Cu}_2\text{O}\) and \(\text{CuO}\), respectively. In this way, \([X_i]^*\) is a number fraction of defects. The way in which the oxide layers were normalized depends on the geometry, and the relative distance from the \(\text{Cu}/\text{Cu}_2\text{O}\) interface in \(\text{Cu}_2\text{O}\) or the \(\text{Cu}_2\text{O} /\text{CuO}\) interface in \(\text{CuO}\). Referring to Fig. 2.4, the outer interfaces in the \(\text{Cu}_2\text{O}\) and \(\text{CuO}\) layers are closer to \(O_2\) and the inner interfaces in the \(\text{Cu}_2\text{O}\) and \(\text{CuO}\) layers are closer to the \(\text{Cu}\) core. The spatial normalization for rectangular, cylindrical and spherical geometries are:

\[
\text{Rectangular Geometry: } \xi = \frac{x - x_{\text{inner}}}{x_{\text{outer}} - x_{\text{inner}}} \tag{2.7a}
\]

\[
\text{Cylindrical Geometry: } \xi = \ln \frac{x}{x_{\text{inner}}} / \ln \frac{r_{\text{outer}}}{r_{\text{inner}}} \tag{2.7b}
\]

\[
\text{Spherical Geometry: } \xi = \frac{1/x - 1/x_{\text{inner}}}{1/x_{\text{inner}} - 1/x_{\text{outer}}} \tag{2.7c}
\]

The steady-state defect concentration profiles are shown as a function of normalized
distance through the Cu$_2$O or CuO layers in Fig. 2.5 during oxidation of Cu in air at 500, 700 and 900 °C. From these plots, it is noted that the hole and vacancy concentrations are higher at the outer interfaces (i.e., at the Cu$_2$O/CuO interface for the Cu$_2$O layer and at the CuO/O$_2$ interface for the CuO layer) and lower at the inner interfaces (i.e., at the Cu/Cu$_2$O interface for the Cu$_2$O layer and at the Cu$_2$O/CuO interface for the CuO layer). This is because vacancies and holes are created at the outer interfaces and diffuse inward. Similarly, electrons and interstitials in the Cu$_2$O layer and electrons in CuO layer have a higher concentration at the inner interfaces and diffuse out of the particle.

Figure 2.4: Schematic of a copper film during oxidation, illustrating the Cu/Cu$_2$O interface, Cu$_2$O/CuO interface and the CuO/O$_2$ interface and the equilibrium concentration profiles through the Cu$_2$O and CuO layers at a given oxide layer thickness.

2.2.4 Oxide Layer Growth

A copper atom balance in the Cu$_2$O and CuO layers yield the following relations:

\[ J_{Cu_2O} = -2 \cdot J_{Cu} = -J_{V'_Cu} - J_{V''_Cu} + J_{V'_{Cu_i}} + J_{V''_{Cu_i}} \]  
\[ J_{CuO} = -J_{V''_Cu} + J_{V'_{Cu_i}} \]

By applying conservation of mass on each oxide layer, the following set of differential equations describes the rate of change in the moles of Cu, Cu$_2$O and CuO as oxidation occurs, from which the thickness of each oxide layer can be determined:
2.2. THEORETICAL APPROACH

Figure 2.5: Vacancy, hole, electron, and interstitial number fraction profiles of charged defects, across the Cu$_2$O (a) and CuO layer (b) in 21% O$_2$ for a spherical at 500, 600 and 900 °C as a function of normalized distance through the Cu$_2$O and CuO layers. Note that the electron concentration profile in Cu$_2$O is not shown.

\[
\text{Cu Core: } \frac{dN_{\text{Cu}}}{dt} = -J_{\text{Cu}} \cdot A_{\text{Cu/Cu}_2\text{O}} \quad (2.9a)
\]

\[
\text{Cu}_2\text{O Layer: } \frac{dN_{\text{Cu}_2\text{O}}}{dt} = J_{\text{Cu}_2\text{O}} \cdot A_{\text{Cu/Cu}_2\text{O}} - J_{\text{CuO}} \cdot A_{\text{Cu}_2\text{O/CuO}} \quad (2.9b)
\]

\[
\text{CuO Layer: } \frac{dN_{\text{CuO}}}{dt} = J_{\text{CuO}} \cdot A_{\text{Cu}_2\text{O/CuO}} \quad (2.9c)
\]
Here, the flux of species $i$ ($J_i$) and the area ($A_i$) are evaluated at the interfaces between phases. Finally, it is assumed that charge neutrality exists (Eq. 2.10) and that no net current flows through a given oxide layer (Eq. 2.11).

\[
\sum z_i \cdot [X_i] = 0 \quad (2.10)
\]

\[
\sum z_i \cdot J_i = 0 \quad (2.11)
\]

The diffusivities for electrons and holes are determined from their mobilities, $\nu_k$, using Einstein’s relation ($D = \nu_k \cdot k_B \cdot T$). In the Cu$_2$O layer, hole and electron mobilities published by Dieckmann and Xue [52] are used and in the CuO layer, mobilities are estimated from electrical conductivity data published by Park and Natesan [67] assuming $\nu_e' = \nu_h'$. While this assumption is used, varying hole and electron mobilities by $\pm 10\%$ had little impact on the final model results. Also in the model, it is assumed that neutral and singly ionized copper vacancies and neutral and singly charged copper interstitials in Cu$_2$O have the same diffusion coefficients (i.e., $D_{V^x_{Cu}}^{Cu_2O} = D_{V_{Cu}'}^{Cu_2O} = D_{V_{Cu}}^{Cu_2O}$) and $D_{Cu^x_{i}}^{Cu_2O} = D_{Cu_i}^{Cu_2O}$, in accordance with the arguments put forth by Peterson and Wiley [81].

The unknowns in the model are the defect diffusion coefficients for vacancies ($V^x_{Cu}$ and $V_{Cu}^{'x}$) and interstitials ($Cu^x_{i}$ and $Cu_i$) in Cu$_2$O and vacancies ($V^x_{Cu}$) in CuO. These diffusion coefficients were extracted from data obtained in experiments discussed below with high-purity, polycrystalline, copper wires. The model was then validated by comparing the model output to oxidation data obtained in TGA experiments with polycrystalline copper particles. The sensitivity of model predictions to the values selected for the diffusivities of the defects is considered in the discussion section of this chapter.

2.3 Materials and Methods

Experiments were undertaken to determine defect diffusion coefficients and to validate the predictive capabilities of the model. In these experiments, polycrystalline copper
2.3. MATERIALS AND METHODS

samples were exposed to environments containing specified levels of oxygen at selected temperatures in a flow reactor and in a thermogravimetric analyzer (TGA). The flow reactor experiments were conducted with copper wire with a diameter of 0.25 mm and the TGA experiments with spherical copper particles of two size distributions. The thicknesses of the Cu2O and CuO oxide layers obtained from the flow reactor experiments on the wire samples were used to determine the diffusion coefficients of vacancies and interstitials in Cu2O and vacancies in CuO. The data collected from the TGA experiments was then used to validate the model.

The copper wires used in the flow reactor experiments and the copper particles used in the TGA experiments were both purchased from Alfa Aesar and have purities of 99.9985 wt.% and 99.9 wt.%, respectively. The weight fractions of the impurity atoms in the wires and particles, listed in Table 2.3, were provided by Alfa Aesar, and atom fractions were calculated based on the impurity atom molecular weights. The impurity measurements for the wires were made by glow discharge mass spectroscopy (GDMS), and for the spheres by inductively coupled plasma analysis (ICPA). The highest concentration of impurity species in the copper wires is sulfur and in the copper spheres O2 implying some native oxide layer is present.

A Coulter multi-counter was used to measure the size distribution of the copper particles used in this study. A detailed procedure of how the multi-counter was run can be found in Appendix B.2. The measured particle size distributions are shown in Fig. 2.6 along with their log-normal fits. From these fits, it can be seen that the mean particle diameters are 10 and 35 µm, respectively. The particles were oxidized in the TGA from 500 to 800 °C in 6 and 21 % O2 at 1 atm. The porosity of the particles was measured via tap density tests, where the procedure for the tap density is described in Appendix B.3. From the tap density test, the porosity of the particles was found to be less than 0.1. Therefore, the copper particles were assumed to be non-porous, and the mass specific surface area, $S_g$, was calculated from geometry: $S_g = \frac{6}{\rho_{Cu} d}$, where $d$ is the particle diameter and $\rho_{Cu}$ the particle’s true density.
Table 2.3: Impurity atom weight and atom fractions measured in the wire and sphere samples; the analysis is provided by Alfa Aesar and measured using GDMS for the Cu wires or ICPA for the Cu spheres. * Atom fractions calculated from the impurity analysis provided by Alfa Aesar.

<table>
<thead>
<tr>
<th>Atom</th>
<th>Wires Concentration, wt. ppb</th>
<th>Wires Concentration at. ppb*</th>
<th>Spheres Concentration, wt. ppm</th>
<th>Spheres Concentration at. ppm*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ag</td>
<td>19</td>
<td>11.2</td>
<td>&lt;10</td>
<td>&lt;6</td>
</tr>
<tr>
<td>Al</td>
<td>3</td>
<td>7.1</td>
<td>&lt;20</td>
<td>&lt;47</td>
</tr>
<tr>
<td>Si</td>
<td>3</td>
<td>6.8</td>
<td>&lt;20</td>
<td>&lt;45</td>
</tr>
<tr>
<td>Fe</td>
<td>17</td>
<td>19</td>
<td>&lt;20</td>
<td>&lt;23</td>
</tr>
<tr>
<td>S</td>
<td>200</td>
<td>360</td>
<td>&lt;20</td>
<td>&lt;19</td>
</tr>
<tr>
<td>Mn</td>
<td>4</td>
<td>4.6</td>
<td>&lt;20</td>
<td>&lt;6</td>
</tr>
<tr>
<td>Ti</td>
<td>2</td>
<td>2.7</td>
<td>&lt;20</td>
<td>&lt;160</td>
</tr>
<tr>
<td>K</td>
<td>&lt;5</td>
<td>&lt;8</td>
<td>5100</td>
<td>&lt;9700</td>
</tr>
<tr>
<td>Na</td>
<td>&lt;1</td>
<td>&lt;2.8</td>
<td>&lt;20</td>
<td>&lt;11</td>
</tr>
<tr>
<td>Li</td>
<td>&lt;1</td>
<td>&lt;9.1</td>
<td>50</td>
<td>&lt;54</td>
</tr>
</tbody>
</table>

2.3.1 Flow Reactor Experiments

In the flow reactor experiments, copper wires were oxidized for a specified period of time at a given temperature between 600 °C and 900 °C, under flowing dry air (21% O₂ bal. N₂) at atmospheric pressure. In the experiments conducted at 600, 700, and 800 °C, the copper wires were exposed to air for 10 min, 1 hr, 2 hr, and 3 hr; and for the tests conducted at 900 °C, the wires were exposed to air for 1 min, 10 min, 30 min, and 60 min. The flow reactor, illustrated in Fig. 2.7, is a 91 cm long quartz tube with a 5 cm diameter, mounted horizontally inside a 51 cm long Thermodyne tube furnace. A section of the quartz tube (approximately 41 cm) was kept outside of the heater such that at the end of an oxidation experiment when the gases were switched to N₂, the wires could cool quickly without being exposed to ambient air.
2.3. MATERIALS AND METHODS

Figure 2.6: Mass distributions for the nominally 10 $\mu$m (a) and 35 $\mu$m spherical copper particle samples (b) used in the TGA experiments.

![Mass Distributions](image1)

(a) $\mu = 10.2 \, \mu m$, $\sigma = 2.2 \, \mu m$
(b) $\mu = 35.3 \, \mu m$, $\sigma = 3.3 \, \mu m$

(Figure 2.7: Flow reactor setup with sample in the cooling section.

Each flow reactor test was run with the following protocol: 1) the copper wires were cleaned with isopropyl alcohol to remove any oils present on the wire surfaces and then placed in a ceramic boat. 2) The furnace was heated to the desired test temperature while flowing 99.995% N$_2$ through the furnace. 3) The ceramic boat and wire were then inserted into the heating section of the tube furnace along with a K-type thermocouple to measure the temperature in the section of the furnace where the wires were located. 4) After a dwell time of 5 minutes at the test temperature, the gas flow to the flow reactor was switched from nitrogen to dry air. 5) After a desired reaction time in air, the flow reactor input gas was switched from dry air to 99.995% N$_2$ and the ceramic boat containing the oxidized copper sample was pulled into the section of the reactor outside of the heater, where it was allowed to remain until the sample reached room temperature, ending the test.

Shown in Fig. 2.8 are SEM images of cross-sectioned wires oxidized at selected times for a reaction temperature of 700 °C. Oxide layer thicknesses were measured at
20 to 40 random locations around the circumference of the wire and this was used to estimate the variation of growth in the Cu$_2$O and CuO layers. Cracks in the oxide layers were found at the interfaces between Cu/Cu$_2$O, Cu$_2$O/CuO as well as in the bulk of the Cu$_2$O layer. It is believed that these cracks occurred due to the molar density differences between Cu, Cu$_2$O and CuO. The thickness of the Cu$_2$O and CuO layers grown at 800 °C are shown in Fig. 2.9a and the total oxide thicknesses of the wires at all temperatures are shown in Fig. 2.9b. The dotted lines in the figure denote calculated model results and are discussed below, in Section 3.3: Results and Data Analysis. The kink in the model predicted line at 800 and 900 °C corresponds to the point when all of the copper in the wire is consumed. The plots indicate that oxidation rates increase with increasing temperature and that the CuO layer grows much slower than the Cu$_2$O layer.

### 2.3.2 TGA Experiments

In the TGA experiments, Cu particles were oxidized at temperatures between 500 °C and 800 °C in 6 or 21% O$_2$ and 1 atm. A description of the thermogravimetric analyzer (TGA) employed in this work has been described elsewhere [94]. The calibration procedure used for the TGA can be found in Appendix B.1. For these TGA experiments, the following protocol was used: 1) between 15 to 25 mg of copper particles was placed on the balance pan of the TGA. 2) The TGA reaction chamber was purged with 250 mL/min of 99.995% N$_2$ at room temperature for 90 minutes. 3) The sample was heated to the desired reaction temperature at a ramp rate of 20 °C/min in 99.995% N$_2$. 4) The sample was held at temperature in 250 mL/min N$_2$ for 1 hour. 5) The nitrogen gas was turned off and the sample is exposed to 250 mL/min of 6% or 21% oxygen balance N$_2$. Sample weight is recorded every 5 seconds until the end of the exposure time. 6) The TGA reaction gas was switched back to 250 mL/min of 99.995% N$_2$ and the sample was allowed to cool to room temperature, ending the test.

Each TGA test was conducted at least twice to assess repeatability of the measurements. Additionally, in order to account for buoyancy and drag forces that act
2.3. MATERIALS AND METHODS

Figure 2.8: SEM images of copper wires in cross section, at 700 °C exposed for 0, 1, 2 and 3 hours in panels a through d, respectively.

on the TGA balance pan during a test, each experiment was repeated with a blank pan. Subtraction of measurements from runs with and without the sample yields drag- and buoyancy-corrected mass measurements. In order to reduce the impact of inter-particle diffusion during each oxidation experiment, a small mass of copper particles (typically less than 25 mg) was spread in a monolayer over the TGA balance pan. To verify that observed oxidation rates were not limited by the external diffusion of oxygen, diffusion-limited oxidation rates were calculated and compared to the measured oxidation rates. The calculated diffusion-limited rates for the copper particle sizes used in the experiments were at least an order of magnitude larger than the measured oxidation rates indicating that the measured rates were negligibly
Figure 2.9: (a) Thicknesses of the Cu$_2$O, CuO oxide layers and total oxide thickness (Cu$_2$O+CuO layer) at 800 °C and 21% O$_2$, and (b) the total oxide thickness as a function of the square root of time. Dotted lines represent calculated model predicted results and triangles are data points measured from the flow reactor experiments.

affected by external mass transport phenomena. Therefore, the measurements reflect
2.3. MATERIALS AND METHODS

The oxygen diluent present in the 99.995% N\textsubscript{2} used in the experiments is above the equilibrium partial pressure for Cu\textsubscript{2}O at some of the temperatures considered. To determine if this would impact the observed oxidation rates, prior to admitting the 6% or 21% O\textsubscript{2} to the TGA, the copper particles were exposed to a forming gas consisting of 7% H\textsubscript{2} bal. N\textsubscript{2}. This was also done to assess the influence of native oxide layers on observed oxidation rates. The measured oxidation rates in these experiments were found to be within the experimental error of those performed without this pretreatment. Consequently the forming gas was not used to pre-treat the copper in all the experiments.

Typical experimental results from Cu oxidation tests in 21% O\textsubscript{2}, with the nominally 10 \(\mu\)m and 35 \(\mu\)m particles are shown in Fig. 2.10. In the panels of this figure, mass gain data are shown as extent of conversion of the oxygen carriers, which is calculated by scaling the mass gain by the total amount of oxygen that the initial mass of copper can consume:

\[
\xi = \frac{m(t) - m_o}{m_o \cdot MW_o/MW_{Cu}}
\]

In Eq. 2.12, \(m_0\) is the initial mass of the sample, and \(MW_\text{O}\) and \(MW_{Cu}\) are the molecular weights of oxygen and copper, respectively. The impact of temperature on conversion of the nominally 10 \(\mu\)m and 35 \(\mu\)m particles in 21% O\textsubscript{2} is illustrated in Fig. 2.10 and illustrated in Fig. 2.11 is the impact of size and gas concentration on conversion at 600 and 800 °C. From these plots, it can be seen that, Cu oxidation is characterized by a fast initial rate, which drops off quickly and decreases with increasing conversion. The high initial rate in each experiment is characterized with Cu conversion to Cu\textsubscript{2}O, and the decrease in the rate with higher conversion is associated with Cu\textsubscript{2}O conversion to CuO.

It should also be noted that the observed oxidation rates are strongly dependent on the particle diameter: smaller particles are converted to CuO faster than larger particles. This is because the Cu oxidation rate is limited by defects diffusing through the growing Cu\textsubscript{2}O and CuO layers and the diffusion length (\textit{i.e.}, oxide layer thickness)
Figure 2.10: Conversion vs time of the nominally 10 µm (a) and 35 µm (b) particles exposed to 21% O₂ obtained from the TGA. The lines are TGA data and the error bars are based on repeat experiments.

Figure 2.11: Conversion vs time of the nominally 10 µm and 35 µm Cu particles at 600 °C (a) and 800 °C (b) exposed to 6 or 21% O₂. The lines are TGA data and the error bars are based on repeat experiments.

is smaller for smaller diameter particles. In 21% O₂ and 800 °C, the nominally 10 µm diameter copper particles are fully oxidized in 200 minutes whereas the nominally 35 µm diameter particles require over 400 minutes for complete conversion. The data also indicate that there is a weak dependence on oxygen concentration: the higher the partial pressure of oxygen the faster the Cu oxidation rates. This pressure dependence is only seen in the second half of oxidation, which corresponds to the growth of the CuO layer. At 800 °C and under 21% O₂, a nominally 35 µm particle reaches 90%
conversion in 200 minutes, while the same particles exposed to 6% O₂ reach 86% conversion in the same period of time.

2.4 Results and Data Analysis

The thicknesses of the Cu₂O and CuO oxide layers, grown on the copper wires oxidized in the flow reactor (see Fig. 2.9), were used to determine diffusion coefficients for defects in Cu₂O and CuO. For each experimental condition, the model was used to predict the number of moles of copper in each layer (i.e., Cu, Cu₂O and CuO) as a function of time, from which the masses and the thickness of the different oxide layers were calculated assuming cylindrical coordinates and utilizing the true densities of Cu, Cu₂O and CuO ($\rho_{Cu} = 8.9$ g/cm$^3$, $\rho_{Cu_2O} = 6.0$ g/cm$^3$, and $\rho_{CuO} = 6.3$ g/cm$^3$, respectively).

As discussed in Section 2.5.1, the impurity concentration in the wires are sufficiently low to have little observable impact on oxidation rates. As such, impurity atoms were not considered in the calculations used to extract the defect diffusion coefficients for the wires. In the case of the spheres, the impurity concentration is sufficiently high to alter the intrinsic defect concentrations in CuO and to a much lesser degree in Cu₂O. However, reasonable agreement between the oxidation rates of the copper spheres obtained in the TGA and the model employing the defect diffusion coefficients obtained from the high purity wires is found to occur when the impurity atom fraction in the model is set below $10^{-6}$. As a result, when analyzing the sphere TGA data, impurities were not considered.

For each experimental condition, the values of the defect diffusion coefficients were iteratively adjusted until the predicted and measured oxide thicknesses agreed. The resulting vacancy diffusion coefficients were then correlated with temperature, as illustrated in Fig. 2.12 and fit to an Arrhenius form, given in Eq. 2.13. The dashed lines in Fig. 2.12 denote other measured values for the vacancy diffusion coefficient found in the literature. Error bounds in Fig. 2.12 for the fit diffusion coefficients were based on the variation in the measured oxide thickness for a given exposure time and temperature. A comparison of these diffusion coefficients to previously published
values can be found below, in Section 2.5. Little to no sensitivity was observed for the interstitial diffusion coefficient in \( \text{Cu}_2\text{O} \) because the interstitial defect concentration was too low over the temperatures and oxygen activities considered. Therefore no diffusion coefficients for interstitials could be extracted.

Figure 2.12: Vacancy diffusion coefficients in \( \text{Cu}_2\text{O} \) (a) and \( \text{CuO} \) (b) where the data points are from this study, red solid line is a best fit, and the dashed lines in (a) are: MSG – Mrowec et al. [95], GMM – Grzesik et al. [80] and MFP Maluenda et al. [96], dashed lines (b) GM – Grzesik and Migdalska. [53]
2.4. RESULTS AND DATA ANALYSIS

\[ D_{\text{Cu}_2\text{O}}^{\text{Cu}} = 3.7 \cdot 10^{-2} \cdot e^{-54kJ/\text{mole}/RT} \] (2.13a)

\[ D_{\text{CuO}}^{\text{Cu}} = 3.5 \cdot 10^{-2} \cdot e^{-32kJ/\text{mole}/RT} \] (2.13b)

In order to assess the validity of the model, it was then used to predict conversion of the spherical copper particles oxidized in the TGA. To fit the TGA data, the measured size distributions were discretized in 2 µm increments and the model was used to predict conversion for a given diameter particle. Then a weighted sum of the solutions for a given diameter particle was calculated and compared to the TGA conversion data. Comparisons between measured and calculated extents of Cu conversion to CuO are shown as a function of temperature, particle size and oxygen concentration in Fig. 2.13. The agreement between model predictions and measurements is good, strongly suggesting that the defect model presented in this chapter accurately predicts copper oxidation rates over a range of temperatures, oxygen partial pressures and copper particle sizes. As can be seen in Fig. 2.13, the model does a better job at predicting the Cu oxidation rate at 600, 700 and 800 °C than at 500 °C and has a better prediction of the data collected at 500 °C for the 10 µm particles than for the 35 µm particles. The kink in the model predictions, demarked by circles in Fig. 2.13a, indicate the point at which the Cu core is completely consumed. This point can be found in Fig. 2.13b through 2.13d as well. This type of kink does not occur in the TGA data where there is a smoother transition between Cu conversion to Cu₂O and Cu₂O conversion to CuO.
Figure 2.13: Measured (triangles) by TGA experiments and calculated (solid lines) extent of conversion profiles for the nominally 35 \( \mu \text{m} \) (a,b) and 10 \( \mu \text{m} \) (c,d) copper particles exposed to 6% (a,c) and 21% (b,d) oxygen at 500 (black), 600 (blue), 700 (green) and 800 (red) °C. The dark gray circle in (a) denotes the point in the model where Cu depletion occurs.

2.5 Discussion

The defect diffusion coefficients determined for vacancies in \( \text{Cu}_2\text{O} \) and in \( \text{CuO} \) are key variables in the model. Questions of interest are how the extracted defect diffusion coefficients compare with previously published defect diffusion coefficients (Fig. 2.12), as well as reported self-diffusion coefficients (Fig. 2.14), and how sensitive the model predictions are to key variables within the model. The latter is discussed in the subsection below, titled Sensitivity Analysis 2.5.2. The defect diffusion coefficients
extracted from the measurements (Eq. 2.13a and 2.13b) can be converted to self-diffusion coefficients of migrating vacancies (\(V_{x_{Cu}}^{z}\) and \(V_{x_{Cu}}^{'}\)), in \(Cu_2O\), and vacancies (\(V_{Cu_{i}}^{''}\)) in CuO at a specified temperature and oxygen pressure using Eq. 2.14a and 2.14b when employing defect concentrations calculated from the defect model.

\[
D_{Cu_{i}}^{V_{x_{Cu}}^{z}} = D_{V_{Cu_{i}}}^{Cu_2O} \cdot [V_{x_{Cu}}^{z}],
D_{Cu_{i}}^{V_{x_{Cu}}^{'}} = D_{V_{Cu_{i}}}^{Cu_2O} \cdot [V_{x_{Cu}}^{'}]
\] (2.14a)

\[
D_{Cu_{i}}^{V_{x_{Cu_{i}}^{''}}} = D_{V_{Cu_{i}}}^{CuO} \cdot [V_{Cu_{i}}^{''}]
\] (2.14b)

Assuming that the vacancy defects in \(Cu_2O\) are non-interacting, which should be true since they are in dilute concentration at the conditions considered, an overall self-diffusion coefficient for Cu atoms in \(Cu_2O\) with contributions from neutral and singly charged vacancies can be defined by Eq. 2.15.

\[
D_{Cu_{2O}}^{Cu_{i}} = D_{Cu_{i}}^{V_{x_{Cu}}^{z}} + D_{Cu_{i}}^{V_{x_{Cu}}^{'}}
\] (2.15)

Shown in Fig. 2.12a are reported vacancy diffusion coefficients in \(Cu_2O\) determined by Mrowec et al. [95], Grzesik et al. [80] using a TGA, and by Maluenda et al. [96] obtained from electrical conductivity measurements. The previously published vacancy diffusion coefficients are in good agreement with all three data sets. However, better agreement is obtained from the vacancy diffusivities extracted from the TGA results of Grzesik et al. [53] and Mrowec et al. [95]. The slightly higher value of the diffusion coefficient determined at 600 °C relative to the best fit line may be a consequence of grain boundary diffusion, which becomes more important at low temperatures. Therefore the 600 °C datapoint was not included when computing the best fit line.

Shown in Fig. 2.12b is the vacancy diffusion coefficient in CuO as a function of inverse temperature. The dashed line in the plot was determined using Eq. 2.14b employing the self-diffusion coefficient reported by Grzesik and Migdalska [53]. This calculated vacancy diffusion coefficient has an activation energy of 63 kJ/mole, a value that is somewhat larger than the value (32 kJ/mole) reported in Eq. 2.13b.
vacancy diffusion coefficient in CuO determined in this study is three orders of magnitude larger than the vacancy diffusion coefficient calculated with the self-diffusion coefficient presented by Grzesik and Migdalska [53] indicating that the measured observed rates are faster. It is likely that our observed oxidation rates are faster due to a higher defect concentration in our samples or from oxygen diffusion through cracks that developed during oxidation in the wire or spherical samples. Cracks were unlikely to form in the samples created by Grezesik and Migdalska because their samples were of planar geometry [53].

Shown in Fig. 2.14a are published self-diffusion coefficients for copper diffusing in Cu$_2$O, evaluated at an oxygen activity equal to the Cu/Cu$_2$O phase equilibrium pressure as a function of temperature. These published self-diffusion coefficients were determined in radiotracer studies by Wiley and Peterson [81], Moore and Castellan [97] and Moore and Selikson [98] and in TGA studies by Haugsrud and Norby [99] and Mrowec, Stoklosa and Godlewski [95]. Equations 2.14a and Eq. 2.15 were used to calculate an overall self-diffusion coefficient resulting from diffusion of V$_{\text{Cu}}$ and V$'_{\text{Cu}}$ in Cu$_2$O. The predicted self-diffusion coefficient has an activation energy of 161 kJ/mole, which is in very good agreement within the range of published activation energies [81, 95, 97, 99]. This indicates that the model captures the correct defect concentration, temperature and pressure dependence in the Cu$_2$O layer. As noted above for the fitting of the vacancy diffusion coefficient in Cu$_2$O at 600 °C, the self-diffusion coefficient determined at 600 °C was not included in the best fit because it is likely being influenced by grain boundary diffusion.

Figure 2.14b shows the self-diffusion coefficient for copper diffusion in CuO reported by Grzesik and Migdalska (measured in a TGA) [53] as well as the presented model results, both evaluated at 1 atm and 21% O$_2$. From Fig. 2.14b, it can be seen that the calculated self-diffusion coefficient has a similar activation energy to the one reported by Grzesik and Migdalska, 122 kJ/mole vs 147 kJ/mole, respectively. As with the vacancy diffusion coefficients in CuO, the self-diffusion coefficients also differ by three orders of magnitude for similar reasons as mentioned above.
2.5. DISCUSSION

Figure 2.14: Overall self-diffusion coefficients for defects in Cu$_2$O evaluated under $P_{O_2}$ at the Cu/Cu$_2$O phase boundary (a) and self-diffusion coefficient in CuO evaluated under 21% O$_2$ (b). Data points, this study; red solid line best fit; dashed lines in (a): WP– Wiley Peterson [81], MC– Moore Castellan [97], MSM– Moore and Selikson [98], HN– Haugsrud Norby [99], MSG– Mrowec, Stoklosa and Godlewska [95]; dashed lines in (b): GM Grzesik and Migdalska [53].
2.5.1 Impact Of Impurities

The presence of aliovalent impurity atoms can impact defect concentrations in the Cu$_2$O and CuO layers as well as the observed copper oxidation rates. While impurity atoms with similar and different valance states as copper in Cu$_2$O or CuO may also influence the band gap, electrical conductivity or the defect formation energies through structural distortions or interacting electronic states, their influence is not considered in this analysis [49,100,101]. In the subsequent analysis, only metallic impurity atoms with similar ionic radius, bonding structure and electronegativity were considered.

The concentration of impurity atoms found in the un-oxidized Cu wires and spheres, presented in Table 2.3, were used to estimate the concentration of impurity atoms in the Cu$_2$O and CuO layers, listed in Tables 2.4 and 2.5. The net charge of these aliovalent impurities in the Cu$_2$O and CuO layers was then estimated by taking the weighted sum of the relative charge number with respect to Cu ions in Cu$_2$O or CuO, and the atom fractions of impurity species listed in Tables 2.4 and 2.5. In this way, impurity atoms with the same charge as the lattice copper do not contribute to the charge neutrality condition and negatively charged impurity atoms can balance out positively charged impurity atoms. In the wires, the net charge of the impurity species is positive and the atom fraction in the Cu$_2$O and CuO layers are $2 \times 10^{-7}$ and $8.5 \times 10^{-10}$, respectively. In the spheres, the net charge of the impurity atoms is also positive and their atom fractions in Cu$_2$O and CuO are $8.3 \times 10^{-4}$ and $1.5 \times 10^{-4}$ respectively.

The calculated concentrations of aliovalent impurity atoms in Cu$_2$O and CuO were employed in the model to determine their impact on the equilibrium defect concentration. The results of this analysis, shown in Fig. 2.15, indicate that the impurity concentration in the wires is sufficiently low not to influence oxidation rates whereas the impurity concentration in the spheres is high enough to change the defect concentrations. Additionally from Fig. 2.15 it can be seen that increasing the concentration of positive impurity atoms decreases the concentration of positive defects (e.g., holes and interstitials) and increases the concentration of negative defects (vacancies and electrons (not shown)) in the Cu$_2$O and CuO layers. Defect concentration changes
occur in this way due to the charge neutrality condition, Eq. 2.15, a larger concentration of positively charge impurities must be balanced by a larger concentration of negatively charged defects or a lower concentration of positively charged defects. Introducing negative impurity atoms results in the opposite trend. The model also supports that impurities have a larger impact on defect concentrations at lower temperatures than higher temperatures. This is because defect concentrations tend to increase exponentially with temperature, while the impurity concentration is fixed by the impurity content of the starting material.

Table 2.4: Estimated Impurity concentrations in the Cu$_2$O layer. This table only considers the metallic species found in the base copper of the wires and spheres.

<table>
<thead>
<tr>
<th>Atoms</th>
<th>Ionic Charge Number</th>
<th>Impurity Concentration in Cu$_2$O, wt. ppb</th>
<th>Impurity Concentration in CuO, wt. ppb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ag</td>
<td>1</td>
<td>37.3</td>
<td>19.8</td>
</tr>
<tr>
<td>Fe</td>
<td>3</td>
<td>64.5</td>
<td>34.2</td>
</tr>
<tr>
<td>Al</td>
<td>3</td>
<td>23.6</td>
<td>12.5</td>
</tr>
<tr>
<td>Li</td>
<td>1</td>
<td>30.5</td>
<td>16.2</td>
</tr>
<tr>
<td>K</td>
<td>1</td>
<td>27.1</td>
<td>14.4</td>
</tr>
<tr>
<td>Mn</td>
<td>3</td>
<td>15.4</td>
<td>8.2</td>
</tr>
<tr>
<td>Ti</td>
<td>4</td>
<td>8.9</td>
<td>4.7</td>
</tr>
<tr>
<td>Na</td>
<td>1</td>
<td>9.2</td>
<td>4.9</td>
</tr>
</tbody>
</table>

The equilibrium defect concentrations in Cu$_2$O and CuO determined when accounting for the net positive impurity concentration in the spheres ($8.3 \times 10^{-4}$ in Cu$_2$O and $1.5 \times 10^{-4}$ in CuO) were then employed in the oxidation model to calculate particle conversion (where conversion is defined by Eq. 2.12) at 700 °C and under 21% O$_2$ and 1 atm. The conversion plot, shown in Fig. 2.16 illustrates that
Table 2.5: Estimated Impurity concentrations in the CuO layer. This table only considers the metallic species found in the base copper of the wires and spheres.

<table>
<thead>
<tr>
<th>Atoms</th>
<th>Ionic Charge Number</th>
<th>Impurity Concentration in Cu\textsubscript{2}O, wt. ppb</th>
<th>Impurity Concentration in CuO, wt. ppb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ag</td>
<td>1</td>
<td>19.5</td>
<td>10.3</td>
</tr>
<tr>
<td>Fe</td>
<td>3</td>
<td>75.2</td>
<td>39.8</td>
</tr>
<tr>
<td>Al</td>
<td>3</td>
<td>155.7</td>
<td>82.4</td>
</tr>
<tr>
<td>Pb</td>
<td>2</td>
<td>20.3</td>
<td>10.7</td>
</tr>
<tr>
<td>Zn</td>
<td>2</td>
<td>64.2</td>
<td>34</td>
</tr>
<tr>
<td>Ni</td>
<td>2</td>
<td>178.9</td>
<td>94.7</td>
</tr>
<tr>
<td>Sn</td>
<td>4</td>
<td>35.4</td>
<td>18.7</td>
</tr>
</tbody>
</table>

Figure 2.15: Equilibrium defect number fractions in a) Cu\textsubscript{2}O and b) CuO as a function of temperature for positive impurity loadings of pure Cu, —— wires (2 \times 10^{-7} in Cu\textsubscript{2}O and 8.5 \times 10^{-10} in CuO), and spheres (8.3 \times 10^{-4} in Cu\textsubscript{2}O and 1.5 \times 10^{-4} in CuO)

Impurity loadings less than roughly 10^{-6} are in agreement with the experimentally measured conversion. This implies either 1) the aliovalent impurity concentration in
2.5. DISCUSSION

Cu$_2$O and CuO is overestimated, or 2) that it is an accurate estimate and unaccounted secondary effects like changes in the band gap or the formation enthalpy of defects hinder the creation of extrinsic defects. Both are plausible. The actual concentration of impurity atoms in the Cu$_2$O and CuO layers could be less than this estimate for example, if the impurity atoms segregate to grain boundaries or surfaces as found by Zhu et al. [102], or become insoluble in the oxides as a result of having vastly different ionic radii, electronegativity or bonding structure.

![Figure 2.16: Predicted Cu-sphere conversion at 700 °C in 21% O$_2$ for impurity atom fraction of a positively charged defect between 0 and 10$^{-4}$](image)

2.5.2 Sensitivity Analysis

In order to determine how uncertainty in key variables within the model affects the predicted oxide layer growth, a sensitivity analysis was performed. The calculations were performed for a copper sphere of nominal diameter 50 µm at temperatures between 600 and 1000 °C under 21% O$_2$ and 1 atm. At a given temperature, each key variable was independently perturbed by ±10%, and the predicted oxide layer thicknesses ($\delta_{\text{Cu}_2\text{O}}$ or $\delta_{\text{CuO}}$) was determined at 50% conversion, where conversion is defined by Eq. 2.12. A sensitivity coefficient, defined by Eq. 2.16, was used to quantify the relative change of the predicted oxide layer thicknesses for a 10% change of variable, $X_j$, while holding all the other $X_j$’s constant.
\[
\sigma_{i,j} = x_{j|0\%} \cdot \frac{\partial \delta}{\partial x_{j}} \approx \frac{x_{j|0\%} \cdot \delta_{|+10\%} - \delta_{|-10\%}}{x_{j|+10\%} - x_{j|-10\%}}
\]

The variables, \(X_j\), that were considered in the study are: 1) \(\Delta H_{R}^{V_{Cu}^{X}}\) of neutral vacancies in \(\text{Cu}_2\text{O}\), 2) \(\Delta H_{R}^{V_{Cu}^{+}}\) of singly ionized vacancies in \(\text{Cu}_2\text{O}\), 3) \(\Delta H_{R}^{V_{Cu}^{0}}\) of vacancies in \(\text{CuO}\), 4) \(D_{V_{Cu}^{X}}^{\text{Cu}_2\text{O}}\) defect diffusivity of vacancies in \(\text{Cu}_2\text{O}\), 5) \(D_{Cu_i}^{\text{Cu}_2\text{O}}\) defect diffusivity of interstitials in \(\text{Cu}_2\text{O}\), and 6) \(D_{V_{Cu}^{0}}^{\text{CuO}}\) defect diffusivity of vacancies in \(\text{CuO}\).

The sensitivity coefficients for perturbations of the heats of reaction and diffusion coefficients are shown in Fig. 2.17 and Fig. 2.18 respectively. The sensitivity analysis indicates that the model predictions are more sensitive to changes in \(\Delta H_{R}^{V_{Cu}^{0}}\) than the defect diffusion coefficients, and more sensitive to changes in the \(\text{CuO}\) layer than the \(\text{Cu}_2\text{O}\) layer. From Fig. 2.17, it can be seen that for a 10% increase of \(\Delta H_{R}^{V_{Cu}^{0}}\) from its nominal value, the \(\text{CuO}\) layer thickness decreases and the \(\text{Cu}_2\text{O}\) layer thickness will increase. This is due to the fact that when the reaction enthalpy increases by 10%, the concentration of vacancies in \(\text{CuO}\) decreases. This decrease in defect concentrations leads to a slower growth rate of the \(\text{CuO}\) layer and a thicker \(\text{Cu}_2\text{O}\) layer. A similar trend is observed with an increase in the reaction enthalpy of \(V_{Cu}^{0}\) and \(V_{Cu}^{\pm}\) in \(\text{Cu}_2\text{O}\). Note that the defect reaction for \(V_{Cu}^{x}\) has a negative reaction enthalpy and as it becomes more negative more neutral vacancies in \(\text{Cu}_2\text{O}\) form, resulting in a faster growth rate of the \(\text{Cu}_2\text{O}\) layer and a slower growth rate of the \(\text{CuO}\) layer. The sensitivity of model predictions to the formation enthalpy decreases with increasing temperatures. This is due to the fact that there are fewer intrinsic defects at lower temperatures than there are at higher temperatures and changes to their concentration are more pronounced.

The model is much less sensitive to the defect diffusion coefficients than the reaction enthalpies, as can be seen when comparing the sensitivity coefficients in Fig. 2.18 to those presented in Fig. 2.17. As is expected, increasing the diffusion coefficient of defects in \(\text{Cu}_2\text{O}\) increases the thickness of the \(\text{Cu}_2\text{O}\) layer and increasing the diffusion coefficient of vacancies in \(\text{CuO}\) increases the thickness of the \(\text{CuO}\) layer. The sensitivity to the diffusion coefficients is also roughly the same at all the temperatures. In order to assess the sensitivity to the interstitial diffusion coefficient in
2.5. DISCUSSION

Figure 2.17: Sensitivity coefficient at 50% conversion for a ± 10% perturbation of $\Delta H_j$, for $j = V_{Cu}^x$, $V_{Cu}'$ in $Cu_2O$ and $V_{Cu}''$ in $CuO$ for $T = 600, 733, 866, \text{and} 1000 \, ^{\circ}C$

$Cu_2O$, it was set equal to its largest expected value, i.e., the vacancy diffusion coefficient $D_{Cu}^{Cu_2O}$, and then perturbed in the same manner. In this case, the sensitivity to interstitial diffusion is still quite small, less than 0.1. This indicates that interstitials are unimportant at the conditions being considered.
Figure 2.18: Sensitivity coefficient at 50% conversion for a ± 10% perturbation of $D_j$, for $j = V^\text{Cu}_2O$, $C^\text{Cu}_2O$ in $Cu_2O$ and $V^\text{CuO}$ in $CuO$ for $T = 600, 733, 866,$ and $1000 \degree C$
Chapter 3

Cermet Particle Model

In Chapter 3 a model that is capable of predicting the oxidation rate of supported particles in air reactors of CLC systems is presented. Recall, oxygen carriers in CLC systems will be supported to improve particle utilization, conversion rates, and particle robustness. The model developed is for a porous agglomerate particle composed of non-porous metal (Cu) and ceramic (ZrO$_2$) sub-particles. The agglomerate particles, also referred to as cermet particles, are porous as a result of the sub-particle packing. In order to characterize observed oxidation rates of particles with this geometry, a model is presented that describes both oxygen transport through the sub-particle matrix and the fundamental chemical processes occurring during metal oxidation. The modeling efforts describing oxygen transport will be discussed in this chapter and a description of the chemical model is presented in Chapter 2. The cermet particle model is validated by comparing the model predictions with mass gain data collected from oxidation experiments performed in a TGA. The model is then exercised to predict burning rates for particles with several different diameter copper sub-particles.

3.1 Theoretical Approach

The cermet particles modeled in this work, which are composed of tightly packed Cu and ZrO$_2$ spheres, are illustrated in Fig. 3.1. The copper spheres are assumed to be uniformly distributed in the cermet particle. Key variables in the model are the initial
mass fraction of Cu, $\eta_{Cu}$, the diameter of the cermet particle, $d_p$, and those of the sub-particles, $d_{Cu}$ and $d_{ZrO_2}$, respectively, the overall porosity, $\Phi$, and the particle mass specific surface area, $S_g$. For a given cermet particle diameter, porosity and copper loading, the number of Cu sub-particles within the cermet particle can be calculated as a function of the copper particle diameter. The largest copper particle occurs when there is one copper particle, surrounded by a shell of tightly packed ZrO$_2$ (shown on the left half of Fig. 3.1). The diameter of the largest possible particle is given by Eq. 3.1. As the copper diameter decreases from this value, the number of copper particles in the cermet will increase and are assumed to be distributed uniformly throughout the cermet particle.

$$d_{Cu,\text{max}} = \left( \frac{\eta_{Cu} \cdot \rho_A \cdot d_p^3}{\rho_{Cu}} \right)^{1/3}$$  (3.1)

Oxygen (O$_2$) can diffuse through the pores created by the packing of the Cu and ZrO$_2$ sub-particles. The flux of oxygen through these pores, $J_{O_2}$, is given by Fick’s First Law, Eq. 3.2. The first term in Eq. 3.2 is the contribution to the total flux due to an oxygen concentration gradient and the second term is the contribution related to Stefan flow. Stefan-flow arises in heterogeneous reacting systems when there is a net mole change upon reaction. In Eq. 3.2, $D_{eff}$ and $[O_2]$ are the effective O$_2$ diffusion coefficient [cm$^2$/s], and O$_2$ concentration [moles/cm$^3$], respectively, and $V$ the Stefan-velocity.

$$J_{O_2} = -D_{eff} \cdot \frac{\partial [O_2]}{\partial r} - V \cdot [O_2]$$  (3.2)
3.1. THEORETICAL APPROACH

The differential equation that governs oxygen transport through the agglomerate particle is given by Fick’s Second Law, Eq. 3.3a, and is subject to the conditions listed in Eqs. 3.3b through 3.3d. Eq. 3.3b specifies that the particle is initially void of O$_2$. Eq. 3.3c states that the flux of oxygen at the center-line of the particle is zero, and is applied when the Cu particles are uniformly distributed throughout the particle. When there is only one Cu particle located at the center of the cermet, the flux of oxygen at the Cu surface is taken to be zero. The final boundary condition, Eq. 3.3d, equates the overall flux of oxygen to the outer surface of the cermet particle through the boundary layer, $\delta_{bl}$, to the reaction rate, $\hat{R}_{O_2}$, inside the particle.

\[
\frac{\partial [O_2]}{\partial t} = -\frac{1}{r^2} \cdot \frac{\partial}{\partial r} r^2 \cdot J_{O_2} - \hat{R}_{O_2} \quad (3.3a)
\]

\[
\left[ O_2 \right](r, t = 0) = 0 \quad (3.3b)
\]

\[
\left. \frac{\partial [O_2]}{\partial r} \right|_{r=0} = 0 \quad \text{or} \quad \left. \frac{\partial [O_2]}{\partial r} \right|_{r=d_{Cu}/2} = 0 \quad \text{when one Cu particle} \quad (3.3c)
\]

\[
D_{O_2} \cdot \frac{([O_2]_{\infty} - [O_2]_{r=r_0})}{\delta_{bl}} = \hat{R}_{O_2} \quad (3.3d)
\]

The effective diffusion coefficient of oxygen, $D_{eff}$, in Eq. 3.2 reflects the combined influence of Knudsen and bulk diffusion through the porous Cu/ZrO$_2$ sub-matrix. The effective diffusion coefficient along with $D_{Knudsen}$ and $D_{Bulk}$ are defined by Eqs. 3.4a through 3.4c.

\[
\frac{1}{D_{eff}} = \frac{1}{D_{Knudsen}} + \frac{1}{D_{Bulk}} \quad (3.4a)
\]

\[
D_{Knudsen} = \frac{2 \cdot r_p}{3} \cdot \sqrt{\frac{8 \cdot k_B \cdot T \cdot N_a}{\pi \cdot M W_{O_2}} \cdot \frac{\Phi}{\tau}} \quad (3.4b)
\]

\[
D_{Bulk} = A \cdot T^{1.66} / P \quad (3.4c)
\]

The form of the bulk diffusion coefficient, Eq. 3.4c, is from diffusion theory of gases where the value of $A$ is $5.01 \times 10^{-4}$ cm$^2$. Pa/K$^{1.66}$s [104]. Also in these expressions, $k_B$ is Boltzmann constant, $N_a$ is Avogadro’s number, and $\tau$ the tortuosity for the
agglomerate particle. In this work, the tortuosity is taken to be 3, as is commonly
done with coal particles [105]. The bulk diffusion coefficient is based on the Chapman-
Enskog description of diffusion in binary mixtures of gases [104]. The mean pore
radius, \( r_p \), is based on the particle specific surface area, apparent density and the
surface roughness factor (\( SRF \); taken to be 2) and is approximated via by Eq. 3.5:

\[
r_p = \frac{2 \cdot SRF \cdot \Phi \cdot (1 - \Phi)}{\rho_{A,p} \cdot S_{g,p}}
\]  

(3.5)

The Stefan-velocity, \( V \) in Eq. 3.2, is defined by the relation given in Eq. 3.6
which is determined by summing Eq. 3.3a for all gas phase species and recognizing
that there is no net mole change in the system for an ideal gas at a fixed temperature
and pressure (i.e., \( \sum \frac{\partial[X_i]}{\partial t} = 0 \)). The net reaction term, \( \hat{R}_{O_2} \) represents the uptake of
oxygen (moles) by the particle per m\(^2\) of surface and is determined from the kinetic
model described in Chapter 2:

\[
\frac{1}{r^2} \cdot \frac{\partial}{\partial r} [r^2 \cdot V \cdot \frac{P}{RT}] = -\hat{R}_{O_2}
\]  

(3.6)

In the calculations presented, the cermet particle was discretized into \( N \), non-
uniformly spaced concentric shells. The shell distribution was set such that there is a
higher gradation of shells at the outer surface of the particle. In the calculations, the
mole fraction of Cu/Cu\(_2\)O/CuO and [O\(_2\)] were assumed to be uniform within each
shell. Then the set of differential equations describing copper oxidation chemistry
and oxygen transport were applied to each shell, allowing for the mole fractions of
Cu/Cu\(_2\)O/CuO and O\(_2\) to be determined in time. The presented calculations are for
\( N = 100 \) shells. Calculations with a larger number of shells yielded the same results.

### 3.2 Experimental Studies

To demonstrate the capabilities of the model, it was used to predict mass gain data
collected by oxidizing Cu/ZrO\(_2\) cermet particles in a TGA. The particles used in
the experiments were initially composed of a mixture of 20% weight CuO/80% wt.
ZrO\(_2\). During each experiment, these CuO particles were reduced with a forming
3.2. EXPERIMENTAL STUDIES

Gas of 7% H\textsubscript{2} in N\textsubscript{2}, producing cermet particles of 16.6% Cu/84.4% ZrO\textsubscript{2}. These resulting Cu/ZrO\textsubscript{2} cermet particles were then oxidized in the TGA with 1 atm air at temperatures between 600 and 900 °C. The weight loss and weight gain measured from these experiments also confirmed that the particles were initially 20% CuO.

3.2.1 The Cermet Particles

The cermet particles used in this work were manufactured using a spray-drying method, prepared by VITO NV, at the Flemish Institute for Technological Research and provided by Chalmers University. To manufacture these particles, a mixture of 5 to 20 µm diameter CuO and nominally 1 µm diameter ZrO\textsubscript{2} spheres were mixed together with water and dispersant, and ball milled for a specified period of time. Next, the slurry of particles was sprayed into a high temperature furnace where the droplets rapidly dried resulting in spherical particles. The particles were then collected and calcined in air at 1050 °C and sieved through standard screens to yield particles in the range of 90 to 125 µm.

A scanning electron microscope was used to characterize the external and internal structure of the oxidized (20% CuO/ZrO\textsubscript{2}) and reduced (16.6% Cu/ZrO\textsubscript{2}) form of the particles. The internal structure of the particles was assessed by mounting the particles in epoxy resin and cross sectioning them. Energy-Dispersive X-ray Spectroscopy (EDAX) was used to identify CuO, Cu, ZrO\textsubscript{2} and epoxy in the particle. Shown in Fig. 3.2 are images of the raw CuO/ZrO\textsubscript{2} particles and in Fig. 3.3 and 3.4 are cross section images of the raw and reduced particles, respectively. From Fig. 3.2a, it can be seen that the particles are of uniform size and from Fig. 3.2a and Fig. 3.2c it is evident that some portion of the particles are actually hollow spheres. This may have occurred from CuO particles coming out of the Zr\textsubscript{O2} shell. The EDAX system was used to determine that the external surface of the particles is primarily ZrO\textsubscript{2} and identify CuO on the external surfaces of the particles (labeled with Xs in Figs. 3.2c and 3.2d). This also verifies that the characteristic size of the ZrO\textsubscript{2} is roughly 1 µm.

The EDAX system of the SEM was also used to identify Cu or CuO, ZrO\textsubscript{2} and epoxy present in the cross section images of the particles shown in Fig. 3.3 and 3.4.
CHAPTER 3. CERMET PARTICLE MODEL

The granular material in Figs. 3.3 and 3.4 is the ZrO$_2$, the dark material is the epoxy, and the smooth material marked by Xs is CuO. Fig. 3.3a confirms that that a large fraction of the particles are hollow spheres. A close up of the shell material, Fig. 3.3b indicates that the ZrO$_2$ forms a porous shell in these particles. Figs. 3.3c, 3.3d and 3.3e show the variation of the internal structure of the particles in this sample. The particles can be classified based on the characteristic size of the Cu within them. The particles shown in Fig. 3.3c, 3.3d have a CuO core approximately 20 $\mu$m in diameter surrounded by a porous ZrO$_2$ outer shell; and the particle shown in Fig. 3.3e is primarily porous ZrO$_2$ with small CuO particles, less than 10 $\mu$m, distributed throughout it.

Figure 3.2: SEM of particles used in this work. Panel a) shows a far field image of the CuO/ZrO$_2$ particles, b) a close up of the grain structure of the CuO/ZrO$_2$ c) individual particles. x’s identify CuO sub-particles.
Figure 3.3: SEM of the cross-section of the particles used in this work.

Figure 3.4 illustrates what the particles in Fig. 3.3 look like after reduction: Figs. 3.4a and 3.4b likely started out with a large CuO core (e.g. Fig. 3.3c or 3.3d), and
Fig. 3.4d started out with small, distributed CuO particles (e.g., Fig. 3.3e). From Figs. 3.4a, 3.4b and 3.4d, it can be seen that after the CuO is reduced to Cu, the resulting copper occupies less volume. This is due to the molar density difference between CuO (0.079 mole/cm$^3$) and Cu (0.14 mole/cm$^3$). For example, a CuO sub-particle with a diameter of 20 to 30 µm will become a Cu particle of 16.5 to 24.8 µm upon reduction). This density difference can lead to cracks, evidence of which is found in Fig. 3.4a. Additionally, it is also possible for the Cu sub-particles to come out of the cermet particle. Because CuO has a lower molar density than Cu, the reduced form of the cermet particles (e.g. Cu/ZrO$_2$) will have a larger porosity than the oxidized form (e.g. CuO/ZrO$_2$). The Cu in the cermet particles shown in Figs. 3.4c and 3.4d (a cermet with small, distributed Cu particles) also appears to have undergone some densification from reduction, however less cracking in the Cu can be seen.

Physical properties of both the unreacted (20%CuO/ZrO$_2$) and reduced (16.6%Cu/ZrO$_2$) OC particles are listed in Table 3.1. The specific surface area of the unreacted particles (CuO/ZrO$_2$) was measured by N$_2$ BET by Moldenhauer et al. [40] and found to be 0.7 m$^2$/g. In order to determine the overall porosity of the and apparent density of the CuO/ZrO$_2$ particles, they were crushed with a mortar and pestle to remove the center void volume that can be seen in Figs. 3.3a and 3.4b. Tap-density measurements of crushed and non-crushed particles were made assuming a packing factor of 0.37, and true densities of Cu, CuO and ZrO$_2$ to be 8.9 g/cm$^3$, 6.3 g/cm$^3$ and 5.62 g/cm$^3$, respectively. The porosity of the unreacted, non-crushed and crushed particles was found to be 0.54 and 0.44, respectively. The measured apparent density and porosity of the unreacted, crushed particles is consistent with porosity of the unreacted, non-crushed particles when accounting for the center void volume as assessed from the SEM image in Fig 3.3a.

A second data point to describe the structure of the unreacted particles was obtained by removing the CuO present in the cermet particles and then measuring the apparent density of the remaining sample. The CuO was removed from the cermet particles by soaking a sample of particles in 1 molar hydrochloric acid. After the particles were soaked for several days, the acid soaked particles were washed with
3.2. EXPERIMENTAL STUDIES

dionized water, dried, crushed and the apparent density re-measured. The porosity of the acid soaked particles was measured to be 0.54. This value is higher than the porosity measured of the non-acid soaked crushed particles, found to be 0.43. A higher porosity value is expected because removal of the CuO should result in a larger pore volume.

The apparent density and porosity of the particles were also measured after reducing the initial sample to 16.6% Cu/ZrO\(_2\) with a forming gas of 7% H\(_2\)/N\(_2\). The porosity of the reduced, non-crushed and crushed samples was found to be 0.56 and 0.5, respectively. As expected, the crushed particles have a lower porosity than the non-crushed sample. The porosity of the reduced particles (crushed-0.5, whole-0.56) is higher than with the unreacted particles (crushed-0.44, whole-0.54). This indicates that as the CuO is reduced to Cu, it densifies and occupies less volume in the particle.

These measurements further support that the particle consists of a ZrO\(_2\) structure with pockets of CuO. In this way, oxygen must diffuse through a network of porous of ZrO\(_2\) before it can react with the copper. Using Eq. 3.5 with the apparent density and porosity determined from these measurements, the mean pore radius was calculated and is listed in Table 3.1. Its value, 0.4 \(\mu\)m, is consistent with pores in the ZrO\(_2\) matrix of the unreacted particles illustrated in Fig. 3.3b.

3.2.2 TGA Experiments

The TGA used to conduct the experiments described in Chapter 2 was also used for the experiments described in this chapter. Each TGA experiment was run with the following procedure: 1) a fresh sample of 20% CuO/ZrO\(_2\) cermet particles was loaded into the TGA; 2) the TGA was purged with 99.995% N\(_2\) for 120 minutes; 3) the sample was heated to the desired temperature at a rate of 15 °C/min; 4) the sample was held at temperature for 60 minutes; and 5) reactive gases were introduced to the reaction chamber. The cermet particles, initially 20% CuO/ZrO\(_2\), were then exposed to 7% H\(_2\)/N\(_2\) resulting in the CuO being reduced to Cu. Once reduction was complete (usually after 20 to 50 minutes depending on the TGA temperature), the TGA reactor was purged with N\(_2\) for 10 minutes. Then the cermet particles (now
composed of 16.6% Cu/ZrO$_2$) were exposed to 21% O$_2$/N$_2$. This process of reducing then oxidizing the sample was carried out for four reduction and three oxidation cycles. Three oxidation cycles was found to be sufficient to yield repeatable oxidation rates and this number of cycles is in agreement with other researchers investigating kinetics of Cu based oxygen carriers [19, 106]. These final two cycles are what is reported in the results section.

To reduce the impact of inter-particle diffusion in these experiments, the cermet particles were spread in a mono-layer over the TGA balance pan. To test the repeatability of an experiment at a given set of conditions, each experiment was conducted at least twice. In order to account for buoyancy and drag forces that act on the TGA
3.2. EXPERIMENTAL STUDIES

Table 3.1: Properties of Spray Granulated CuO/ZrO$_2$ particles

<table>
<thead>
<tr>
<th></th>
<th>Initial State</th>
<th>After Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass Fraction</td>
<td>20% CuO/ZrO$_2$</td>
<td>16.6 % Cu/ZrO$_2$</td>
</tr>
<tr>
<td>Calcination Temperature</td>
<td>1050 °C</td>
<td></td>
</tr>
<tr>
<td>True Density of particles $^*$</td>
<td>5.74 g/cm$^3$</td>
<td>True Density of particles $^*$</td>
</tr>
<tr>
<td>Apparent Density whole Particles</td>
<td>2.66 g/cm$^3$</td>
<td>Apparent Density whole Particles</td>
</tr>
<tr>
<td>Porosity Whole Particles</td>
<td>0.54</td>
<td>Porosity Whole Particles</td>
</tr>
<tr>
<td>Apparent Density crushed Particles</td>
<td>3.2 g/cm$^3$</td>
<td>Apparent Density crushed Particles</td>
</tr>
<tr>
<td>Porosity Crushed Particles</td>
<td>0.44</td>
<td>Porosity Crushed Particles</td>
</tr>
<tr>
<td>ZrO$_2$ Packing Density (acid soaked)</td>
<td>0.54</td>
<td>Mean Pore Size $^*$</td>
</tr>
<tr>
<td>Specific Surface Area</td>
<td>0.7 m$^2$/s</td>
<td>Mean Pore Size $^*$</td>
</tr>
</tbody>
</table>

$^*$ Denotes a calculated value. A packing factor of 0.37 was used to calculate the apparent density from the bulk density.

pan during an experiment, experiments were conducted with an empty pan. The results from these blank experiments were then subtracted from experiments with a sample to yield the drag- and buoyancy- corrected measurements.

The variation in conversion at 600 °C from cycle-to-cycle and from two different initial samples (denoted batch 1 and batch 2) are shown in Fig. 3.5a and the average particle conversion at 600, 700, 800 and 900 °C are presented in Fig. 3.5b. In these
plots, the mass gain data obtained from the TGA experiments is presented as an extent of conversion, defined by Eq. 2.12.

Fig. 3.5a illustrates that conversion rates are consistent from cycle-to-cycle and batch-to-batch. This agrees with the observed cycle-to-cycle variation observed from oxidizing and reducing a similar set of particles over a period of 45 hours in a CLC reactor [40]. The extent of conversion for the cermet particles at temperatures between 600 to 900 °C, shown in Fig. 3.5b, is characterized by a fast initial rate followed by a slower rate to full conversion. Overall, conversion is faster at higher temperatures and there is a diminishing increase in the rate with temperature.

Figure 3.5: In panel a) is the conversion of the OC at 600 °C at different cycles and in panel b) is the conversion of the oxygen carrier at temperatures between 600 and 900 °C and c) early conversion time.
Conversion within the first 25 seconds of the experiment, shown in Fig. 3.5c, is characterized by reaction delay and fast conversion rates. This reaction delay, which is not typical of oxidation reaction at these temperatures and oxygen concentrations, is likely due to limitations with the introduction of O\textsubscript{2} into the TGA volume and not a feature of Cu oxidation or characteristic with the sample consuming all of the incoming oxygen. During this period, the oxygen concentration in the TGA reactor is changing in time as O\textsubscript{2} begins to displace the N\textsubscript{2} already in the chamber. Given the dimensions of the TGA reactor, 2.54 cm diameter and 10 cm length, and the flow rates of gases in a given experiment, 250 mL/min, it takes roughly 10 to 20 seconds for the oxygen concentration in the reactor to reach 21\%. In order to account for this variation in the model, the oxygen mole fraction profile was estimated by Eq. 3.7. A function of this form was chosen to account for variation in oxygen mole fraction because it can map the asymptotic approach of the oxygen concentration from its value before chemical reactions occur to the inlet or free stream value.

$$X_{O_2} = \frac{X_{O_2}^\infty}{1 + B \cdot e^{-t/k_{O_2}}} \text{ where } B = \frac{X_{O_2}^\infty - X_{Cu/Cu_2O}^{Cu/Cu_2O}}{X_{Cu/Cu_2O}}$$ (3.7)

In Eq. 3.7, \(X_{O_2}^\infty\) is the inlet O\textsubscript{2} mole fraction for the experiment (i.e., 21\%), \(X_{O_2}^{Cu/Cu_2O}\) is the oxygen mole fraction below which no oxidation occurs (a thermodynamic limit) and \(k_{O_2}\) is the fitting parameter used to characterize the oxygen mole fraction in the TGA volume. The value for \(k_{O_2}\) represents a characteristic mixing time and was assumed to be proportional to temperature: \(k_{O_2} = k_{O_2}^0 \cdot T/T_o\).

### 3.3 Results and Data Analysis

The particle model was used to predict the weight gain data obtained from the TGA experiments. In order to fit the conversion data with the model, the characteristic copper diameter, and \(k_{O_2}^0\) from Eq. 3.7 were varied until reasonable agreement was obtained. The characteristic copper diameter was selected as a fitting variable because the cermet particles contain Cu sub-particles with diameter in an unknown size distribution between 5 and 20 µm. In this way, the characteristic diameter fit
with the model will reflect the mean copper size. It will also indicate if the copper is distributed as small spheres or as a single copper core. For the calculations, the overall particle diameter was set to 100 $\mu$m with an overall porosity of 40%.

The value of $k_{O_2}$ that best supports the observed conversion rate data is 3.1 [1/sec]. Using this value for $k_{O_2}$, the model prediction of the O$_2$ mole fraction is shown in Fig. 3.6. These predicted curves are consistent with the reaction delays observed in Fig. 3.5c. The highest temperature experiment has the shortest delay time and the lowest temperature experiment has the longest delay. The conversion data obtained from the TGA experiment at 600 °C is plotted with the model predictions for characteristic copper diameters of 5, 8, 10, 15 and 20 $\mu$m in Fig. 3.7. At 600 °C as well as at higher temperatures, a characteristic copper diameter of 8 $\mu$m was found to yield good agreement between the experimental data and the predicted conversion rates. This supports the finding that the particles are of character with distributed copper as opposed to having a large single copper core. The model predictions at all the temperatures and is shown in Fig. 3.8.

![Figure 3.6: Predicted oxygen mole fraction as a function of time at 600 to 900 °C](image)

From these plots, it can be seen that the model does a reasonable job at predicting the time required for full conversion at all the temperatures considered. The model
3.4. ESTIMATING OXIDATION RATES

Figure 3.7: The measured and modeled extent of conversion for oxidation of the 16 wt. % Cu cermet particle at 600 °C in 21% O₂ balance N₂ and 1 atm. The model predictions are for several characteristic copper diameters ranging from 5 µm to 20 µm.

slightly under predicts initial conversion at 700 °C and over predicts the initial conversion at 900 °C but these variations are well within the experimental error. Part of this difference can be attributed to the gas mixing model and better agreement could potentially be obtained if the O₂ mole fraction at the sample was known.

3.4 Estimating Oxidation Rates

Lastly, the chemical model presented in Chapter 2 and the cermet particle model presented above were used to estimate of the oxidation rates of Cu-based oxygen carriers. These calculated rates are then compared with the diffusion limited burning rate for a copper particle of the same diameter.

Based on the physical characteristics of the cermet particles studied in this chapter (d_p = 100 µm, d_{Cu} = 8 µm, Φ = 0.4, see Table 3.1) and the conditions established in these experiments described in Section 3.3, the average oxidation rate as a function of temperature was calculated and is presented in Fig. 3.9. In this figure, the average reactivity is defined using Eq. 3.8. Additionally, shown in this figure is the diffusion
limited oxidation rate for a 5, 50 and 100 μm diameter copper particle. From the data plotted in this figure, it can be seen that the diffusion limited oxidation rate is always at least three orders of magnitude larger than the observed oxidation rate measured in the TGA experiments presented above in the Results and Data Analysis Section (Section 3.3). This supports findings that the rates measured in the TGA experiments are chemically controlled, and not being influenced by oxygen diffusion.

\[ RR_{ave} = \int_{0}^{1} \xi \cdot RR_{O2} \cdot d\xi \]  

(3.8)

Also shown in Fig. 3.9 are the predicted oxidation rates for a 100 μm cermet particle with 40% porosity and characteristic copper diameters of 1 μm and 8 μm. As
3.4. ESTIMATING OXIDATION RATES

expected, the cermet particle burning rate increases as the characteristic Cu diameter decreases. This trend can also be seen in Figure 3.10, which shows the average reactivity of a 100 µm cermet particle with 40% porosity as a function of the characteristic copper diameter. The apparent oxidation rate for the cermet particle is expected to increase and approach the diffusion limited oxidation rate, the fastest rate at which these particles can react.

Figure 3.9: Average reactivity, as predicted by the cermet particle model and diffusion limited oxidation rate

Figure 3.10: Average reactivity as a function of the characteristic copper diameter
Chapter 4

Copper Oxide Reduction Chemistry

In Chapter 4, theory and experimental work related to metal oxide reduction is presented. Understanding reduction chemistry of metal oxides is important for the development of CLC systems because it will enable researchers to optimize the design and operation of fuel reactors. While reducing a metal oxide to a lower oxidation state or its base metal is commonly performed in industrial processes, there are few models that describe the atomistic steps that occur during reduction [54–56]. This is most notably because metal oxide reduction depends on many variables including but not limited to fuel type, surface structure/orientation, and even other species present on the surface [54].

Generally, metal oxide reduction occurs by the removal of oxygen from the lattice, where the driving force for reduction is minimizing the free energy of the metal oxide in its given environment [54]. There are many substances that can serve as a reducing agent and within fuel reactors of CLC systems the relevant reducing agents are gaseous fuels like CO, H₂, and CH₄ and solid fuels like coal and biomass. As was noted in Chapter 1 when a coal or a biomass is utilized in the fuel reactor, solid fuel conversion is often limited by gasification, which produces a syngas comprised of predominantly CO and H₂ [18, 22, 23, 32, 33]. In addition to metal oxides being reduced by a gaseous fuels in CLC systems, reduction can also occur by the release of oxygen
from the lattice \cite{21,29,36,41}. Both of these pathways are relevant when describing
the reduction of oxygen carriers in fuel reactors of CLC systems.

\section{Reduction Chemistry Background and Theory}

The reduction of a metal oxide with a gaseous fuel begins with the adsorption of the
fuel on the metal oxide surface. The observed adsorption rate and adsorption energy
will depend on a number of factors including the surface defect concentration, grain
size, impurity atom concentration, crystalline orientation, and coverage of other ad-
sorbed species \cite{54,107}. The most favorable adsorption sites are under-coordinated
metal ions and surface oxygen vacancies \cite{54,108}, where adsorption-desorption ki-
netics have been found to follow Langmuir-Hinshelwood-type kinetics \cite{108,109}. For
fuels like H$_2$ and CH$_4$, upon adsorption they can dissociate into radicals like H and
CH$_x$ and then migrate to a surface oxygen atom. A fuel like CO will not disassociate
and can react directly with surface oxygen \cite{54}. When the fuel species and radicals
react with surface oxygen they will form combustion products (CO$_2$ or H$_2$O) and
create surface oxygen vacancy \cite{54–56}.

When a critical concentration of oxygen vacancies is reached, it is believed that
the lattice rearranges nucleating the reduced (i.e. lower) metal oxide or metal \cite{54,
56,108,110,112}. This is often observed experimentally after an induction period
\cite{54,56,108,110,112}. At the edge interface between the sub-oxide/metal nuclei and
the metal oxide, the lattice will likely be strained due to a mismatch in the molar
densities of the oxide and sub-oxide/metal. As a result of this strain, the adsorption
and dissociation energies in this region can become reduced, leading to accelerated
growth \cite{54}. In this way, as the nuclei and this edge region grow, the rate of conversion
increases. This phenomena is described as an auto-catalytic effect and has been
observed for CuO reduction \cite{54,55,111,113}.

As the nuclei grow, they eventually cover the entire external surface. Once the
surface is completely covered with nuclei, the rate of reduction then begins to slow.
Then, the subsequent reduction of the metal oxide becomes limited by the diffusion of
lattice defects (e.g. oxygen vacancies) through the product sub-oxide/metal layer or
CHAPTER 4. COPPER OXIDE REDUCTION CHEMISTRY

via the diffusion of the reducing gas through cracks/grain boundaries in the thickening product layer [54, 55]. Experimentally, it has been observed that it is difficult to fully remove all of the oxygen from a metal oxide. This is explained by some researchers as oxygen becoming dissolved inside the product metal layer due to its finite solubility dictated by temperature and oxygen activity. [56, 110–112].

There are two models that are commonly used to describe metal oxide reduction depending on whether the rate controlling process is nucleation or reduction of the bulk metal oxide [54, 55, 110–112]. The models are the nucleation model or the contracting sphere model, respectively. In the nucleation model, it is assumed that the probability of forming a sub-oxide or metal nuclei is uniform over the entire metal oxide surface and therefore, the nuclei growth rate is proportional to its surface area [55, 114, 115]. The rate of reduction can then be expressed in terms of the fraction of reacted surface \( \alpha \), given by Eq. 4.1. In this expression, \( k_N \) is a rate constant for nucleation, \( A \) the area and \( \alpha \) the fraction of surface covered with nuclei. If the particle area is known, this expression can be integrated to yield the nuclei coverage as a function of time.

\[
\frac{d\alpha}{dt} = k_N \cdot A \cdot (1 - \alpha) \tag{4.1}
\]

In the shrinking sphere model, the rate of reduction is assumed to be proportional to the active surface area of the unreacted core, \( A' \), [55] and is given by Eq. 4.2. If \( A' \) is known, this expression can be integrated to yield conversion as a function of time.

\[
\frac{d\alpha}{dt} = k_N \cdot A' \tag{4.2}
\]

While these models can be used to fit experimental data over a limited range of conditions, the models are not explicit functions of morphological parameters or the system temperature, pressure and gas phase species. As a result, there use as predictive tools is limited due to their phenomenological nature.
4.1. REDUCTION CHEMISTRY BACKGROUND AND THEORY

4.1.1 Background on CuO Reduction

There are at least three stable oxidation states for copper in addition to the metal, and their lattice and structural parameters are listed in Table 4.1. Note that Cu$_2$O and Cu both have a cubic lattice whereas CuO has a monoclinic lattice. As oxygen is removed from CuO during reduction, it can be converted directly to Cu or first form an intermediate state of Cu$_2$O before being fully reduced to Cu \[54,56,110,111\]. While Cu$_3$O$_4$ is a stable intermediate oxidation state, Kim et al. \[56\] explain that it is unlikely to form because it would require specific oxygen atoms to be removed from CuO in addition to cause significant strain on the CuO lattice \[56,110\].

<table>
<thead>
<tr>
<th>Oxidation State</th>
<th>Structure</th>
<th>Lattice Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>CuO</td>
<td>Cu$^{2+}$</td>
<td>Monoclinic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a = 4.68 Å, b = 3.42 Å, c = 5.12 Å [116]</td>
</tr>
<tr>
<td>Cu$_3$O$_4$</td>
<td>Cu$^{1.5+}$</td>
<td>Tetragonal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a = b = 5.51 Å, c = 13.82 Å [117]</td>
</tr>
<tr>
<td>Cu$_2$O</td>
<td>Cu$^{1+}$</td>
<td>Cubic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a = b = c = 4.27 Å [116]</td>
</tr>
<tr>
<td>Cu</td>
<td>Cu$^{0+}$</td>
<td>Cubic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a = b = c = 3.62 Å</td>
</tr>
</tbody>
</table>

Table 4.1: Copper oxides and Cu structure

Many different experimental studies were performed to study how the CuO lattice evolves through these different oxidation states during reduction \[31,56,110,112,118,121\]. From these works, it is apparent that the actual pathway for reduction is complex and depends on many number of factors including: (i) the creation of active sites for fuel adsorption, (ii) particle morphology, (iii) surface diffusion of fuel species, (iv) embedding of fuel species in the lattice, (v) diffusion of oxygen vacancies from the surface to the bulk, (vi) growth and nucleation of sub-oxide or metal, and (vii) desorption of combustion products CO$_2$ and H$_2$O.

Based on the nucleation part of reduction, a model characterized by surface chemistry was proposed to describe CuO reduction with CO \[122\]. The details of the model can be found in Appendix A\[A\]. The rate coefficients in the model were extracted from thermogravimetric experiments that are discussed in the experimental section below. When this model was developed, it only considered surface processes and if it is to be used to characterize reduction rates in CLC systems, a pathway for bulk reduction...
4.2 Experimental Section

In order to study the reduction of unsupported CuO and Cu$_2$O, experiments were performed using a batch-fed, flowreactor and a thermogravimetric analyzer (TGA). The CuO sample studied in this work is of 99.99% purity and was purchased from Sigma Aldrich and the Cu$_2$O sample studied is of 99.9% purity and was purchased from Alpha Aesar. Unsupported CuO particles were used as opposed to supported particles in order to simplify the experimental setup and analysis; unsupported particles could be purchased instead of manufactured and pure CuO particles were thought to be easier to characterize than supported particles. The use of a support can change the apparent rate by creating more surface sites, however the mechanism for reduction should remain the same [55].

Prior to the experiments, the CuO was crushed with a mortar and pestle and dry-sieved with standard screens in the following three size ranges: 75 to 106, 106 to 125 and 125 to 150 µm. The as-received Cu$_2$O particles had diameters less than 30 µm and as such, no sieving was performed. Particles with diameters less than 150 µm were selected for these experiments in order to reduce the potential for diffusion limitations from reactant species. The size distributions for the particles were measured using a Coulter Multisizer and the results are shown in Fig. 4.1. These size distributions for the CuO particles have mean diameters of roughly 60, 85 and 110 µm for the 75 to 106, 106 to 125 and 125 to 150 µm sieved particles, respectively.

The particles were found to be smaller than the size ranges given by the sieves because the sieves did not break apart particles that were stuck together. However once the particles were placed in the solution of the multisizer, the particles that were stuck together came apart. This is perhaps why, that despite sieving, there is still a large number of particles having diameters less than 30 µm in each sample as indicated by Fig. 4.1. A size distribution for the Cu$_2$O particles could not be measured with the Coulter Multisizer due to difficulty dispersing the Cu$_2$O particles in the multisizer solution. Instead, a scanning electron microscope (SEM, FEI, XL30...
4.2. EXPERIMENTAL SECTION

Sirion SEM) was used to estimate the mean particle size. The mean size of the Cu$_2$O particles was found to be 12 $\mu$m with a standard deviation of 6 $\mu$m.

![Size distributions of the CuO particles used in the flow reactor and TGA](image)

Figure 4.1: Size distributions of the CuO particles used in the flow reactor and TGA

The unreacted and reacted particles were also examined with a scanning electron microscope (SEM) and qualitatively analyzed via x-ray diffraction spectroscopy. Representative SEM scans of the unreacted particles are shown in Fig. 4.2 and partially reduced CuO particles are shown in Fig. 4.3. It is apparent from the morphology seen in Fig. 4.2a and 4.2c that the raw CuO and Cu$_2$O particles have a lot of fines, supporting the size distributions measured by the Multisizer and measured on the SEM. A close up of the CuO particles presented in Fig. 4.2b shows that the particles have a dendritic form, which is likely a result of how the particles were manufactured. The raw Cu$_2$O particles shown in Fig. 4.2c have a much different morphology than the CuO particles shown in Fig. 4.2b; the surfaces of the Cu$_2$O particles are smooth and they appear to contain much less porosity than the CuO particles.

The morphological change of the CuO particle as a result of a reduction and subsequent oxidation is evident from Fig. 4.3. The external surface of the CuO particles becomes smoother upon reduction to Cu. Despite the physical change of the surface as a result of CuO reduction, measurements of the specific surface area before and after reduction remained about the same. Measurements of surface areas of the particles were performed in the TGA, in which the mass specific surface area
of both sets of particles was found to be about 1 to 5 m$^2$/g. The Brunauer-Emmett-Teller (BET) [123] approach was used in the analysis of gas adsorption weight data to determine the surface areas. Carbon dioxide was used as the adsorption gas, and adsorption tests were performed at 296 K and 10 atm. A value of 22.2 Å$^2$ was used for the molecular cross section of CO$_2$ [124].

![Figure 4.2: Raw Samples](image)

(a) 
(b) 
(c)

Figure 4.2: Raw Samples (a) CuO, (b) close up of the CuO particles, (c) Cu$_2$O

Based on the volume-to-external surface area ratio, the mean radius of pores inside both the CuO and Cu$_2$O particles was estimated to be 0.014 μm. Pores as large as 0.5 μm were noted in some of our SEM images of the particles (see Fig. 4.2b). Using a tap density technique, the apparent densities of the CuO and Cu$_2$O particles were determined to be 3.66 and 5.0 g/cm$^3$, respectively. Assuming true densities of 6.3 and 6.0 g/cm$^3$ for CuO and Cu$_2$O, respectively, the particle porosity was estimated to be 0.42 for CuO and 0.17 for Cu$_2$O.
4.2. EXPERIMENTAL SECTION

4.2.1 Oxygen Release Pathway

In order to estimate the temperature when the oxygen release pathway becomes activated, calculations were performed in HSC Simulation [125] to estimate the equilibrium temperature at which CuO begins to decompose. The calculation describes the thermal reduction of a CuO sample during heating from 25 to 1000 °C while under 1 atm of N\textsubscript{2}. The product species of CuO, Cu\textsubscript{2}O, Cu and O\textsubscript{2} were calculated as a function of temperature the results are shown in Fig. 4.4. As can be seen from this figure, CuO begins to decompose at temperatures above 600 °C. At temperatures below 600 °C, the influence of oxygen release should be limited. Also shown in this figure is weight loss data measured with a TGA from a thermal decomposition experiment as the temperature of a CuO sample was ramped from 25 to 1000 °C. The orange circles, which are the measured sample mass, appear to follow the equilibrium mass loss curve and this indicates that oxygen evolution is driven by thermodynamic equilibrium. Note that in this experiment, N\textsubscript{2} gas continuously sweeps away any O\textsubscript{2} that evolves from the sample, and this may result in faster decomposition rates than may occur in a fuel reactor.
4.2.2 Batch Reactor

Ramped temperature experiments reducing CuO particles with gaseous fuel (CO, H₂ or CH₄), and mixed CuO particles with solid fuels (char, as-received coal or as-received biomass) were performed in the batch-fed flow reactor shown in Fig. 4.5. The reactor is 30.5 cm long, has a 2.54 cm diameter and contains a porous quartz plate located 7.6 cm from the inlet to support the CuO particles being studied. In these experiments, the CuO sample was loaded into the reactor at room temperature, then the reactor was heated up under a linear temperature ramp in N₂. Gases enter the bottom of the reactor, percolate through the CuO or CuO + solid fuel particles, and then exit through a port at the top of the reactor. The reactor sits within a tubular heater capable of sustaining bed temperatures up to 1100 °C. The reactor and heater was also wrapped in fiberglass insulation to minimize thermal gradients during a given experiment. Two inches of the quartz reactor upstream of the porous support plate and three inch of the quartz reactor after the CuO particles are located within the domain of the heater. The section of tubing prior to the quartz plate serves as a pre-heat section to ensure reaction gases are at the desired temperature.

K-type thermocouples were used to measure the sample temperature, the heater temperature and the inlet gas during a given experiment, and their locations are
4.2. EXPERIMENTAL SECTION

Figure 4.5: (a) Drawing of the flow reactor and (b) flow reactor experiment encased in insulation

illustrated in Fig. 4.5a. The thermocouple used to measure the heater temperature was also used as the input for the PID temperature controller to maintain the desired temperature ramp-rate. Rotometers, which were calibrated prior to each experiment, were used to meter gas flow into the reactor. The flowrates of the gaseous fuels or inert purge was nominally 300 mL per minute. At this flowrate throughout the duration of a given experiment (room temperature to final temperature), the bed was slightly bubbling but never fully fluidized.

The composition of the reactor exhaust gas was measured using a Varian gas chromatograph (GC). The GC calibration procedure can be found in Appendix B.4. Gases were continuously flown through the GC, which was run at the interval of 3 to 7 minutes (depending on the gases that were being measured). The species monitored include CO, CO₂, H₂, and CH₄. In experiments with H₂O as a product and with the Wyodak coal, rice straw and corn stover, a cold trap was placed at the exit of the reactor to prevent water and tars from condensing in the tubing to the GC.
Over 20 CuO reduction experiments were performed in this reactor with a variety of fuels while cycling the bed material from reducing to oxidizing conditions up to three times. An additional 20 experiments were also performed in this reactor with Fe$_2$O$_3$ to determine similar information on iron based CLC systems. These experiments were documented in final report of California Energy Commission Grant 07-05 [126].

4.2.3 Gaseous Fuel Experimental Results

The gaseous fuels and their compositions used in the flow reactor experiments were 99.9% CO, 20% H$_2$ balance 80% Ar and 99% CH$_4$. A summary of the flowrates and bed masses used in these experiments are listed in Table 4.2. Typical exhaust gas measurements from a reduction experiment with CO, H$_2$ and CH$_4$ are shown in Figures 4.6, 4.7 and 4.8, respectively. From these figures, it is evident that the chemical reactions with gaseous fuels are initiated at temperatures well below 600 °C, the temperature at which CuO begins to thermally decompose. Chemical reactions with 99.9% CO begin at roughly 100 °C, with 20% H$_2$ at 125 °C and with 99% CH$_4$ at 400 °C. This is consistent with the other ramped temperature experiments performed by a group at Brookhaven National Lab [56][110], investigating CuO reduction with gas mixtures of 5% CO in He and 5% H$_2$ in helium. As expected, methane has a higher initiation temperature than either CO or H$_2$. This is likely due to the fact that four C-H bonds need to be broken before CH$_4$ can react to form combustion products.

Table 4.2: Initial conditions for gaseous fuel oxidation with copper oxide

<table>
<thead>
<tr>
<th>Fuel</th>
<th>CuO Size Distribution $[\mu m]$</th>
<th>Fuel Flow Rate $[mL/min]$</th>
<th>Initial CuO Loading $[g]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO</td>
<td>75 to 125</td>
<td>301</td>
<td>27.45</td>
</tr>
<tr>
<td>H$_2$</td>
<td>125 to 150</td>
<td>271</td>
<td>13.10</td>
</tr>
<tr>
<td>CH$_4$</td>
<td>125 to 150</td>
<td>296</td>
<td>6.67</td>
</tr>
</tbody>
</table>

The exothermic nature of these chemical reactions is evident from the positive
deviation of the bed temperature profile from linearity for CO and CH\textsubscript{4}. A larger temperature deviation is observed for CO than for CH\textsubscript{4}. No significant deviation is noted for H\textsubscript{2} and this is possibly due to its much larger mass specific heat capacity and that it is dilute in Ar.

As can be seen from Figs. 4.6 and 4.7, the CO and H\textsubscript{2} supplied to the reactor was completely consumed during the experiment whereas only a small portion (less than 5\%) of the CH\textsubscript{4} was converted to combustion products. At around 400 °C, there is a decrease in the CH\textsubscript{4} mole fraction, which tracks with an increase in the CO\textsubscript{2} mole fraction suggesting complete methane conversion. As the temperature increases above 500°C, H\textsubscript{2} and modest amounts of CO are observed. The appearance of H\textsubscript{2} without CO or CO\textsubscript{2} suggests the beginning of carbon formation. There are two peaks in the
H₂ mole fraction, one appearing at 600 °C and the other around 900 °C. The depletion of the first H₂ peak is believed to coincide with the release of oxygen from the CuO particles and the production of H₂O. Once all the oxygen atoms have been released, the H₂ mole fraction starts to increase as the methane thermally decomposes. The heaters were turned off at the time the temperature starts to drop just before 240 minutes into the test.

Figure 4.8: Methane oxidation via CuO, (a) major species, (b) minor species

In these experiments, the overall rate of conversion of the metal oxide was determined from the flow rates of the gases fed to the reactor and the measured mole fractions of CO, CO₂, H₂ and CH₄ in the gases leaving, employing conservation relations. These overall rates were then expressed as functions of temperature and pressure using the following expression:

\[ R_{\text{CuO},i} = -\left\{ \frac{1}{m} \cdot \frac{dm}{dt} \right\} = P_i^n \cdot A_i \cdot \exp\left(\frac{-E_i}{RT}\right) \] (4.3)

Here, \( R_{\text{MeO},i} \) is the specific reaction rate for the metal oxide when exposed to reactive gas \( i \), \( n \) is the order of reaction with respect to the partial pressure of the reactive gas, \( A_i \) is the pre-exponential factor, \( E_i \) is the activation energy, \( R \) is the universal gas constant, and \( T \) is temperature. Values determined for the kinetic parameters when CuO is the oxidizer are shown in Table 4.3. Since sufficient tests were not completed to determine the reaction order \( n \), a value of unity was assumed when fitting the data. A plot of these rates are given by the dotted lines in Figure 4.9. Also shown
in this figure are diffusion limited rates and measured oxidation rates performed in the TGA, which will be discussed in the subsection describing the TGA experiments. The flow reactor rates predicted by this model indicate that at low temperatures, CuO reduction with CO is faster than CuO reduction with H\textsubscript{2} however at higher temperatures the rates become commensurate. Reduction of CuO with methane is always slower than reduction with CO and H\textsubscript{2}.

Figure 4.9: Maximum reactivity from the flow reactor (dashed lines) and TGA reduction experiments (data points) along with the diffusion limited rates (solid lines)

<table>
<thead>
<tr>
<th>Fuel</th>
<th>( A_i ) [1/Pa-s]</th>
<th>( E_i ) [kJ/mole]</th>
<th>( R_i ), [1/s] at ( T=900 ) °C</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO</td>
<td>( 7.78 \times 10^{-5} )</td>
<td>16.0</td>
<td>1.53</td>
</tr>
<tr>
<td>( \text{H}_2 )</td>
<td>( 4.42 \times 10^{-4} )</td>
<td>29.6</td>
<td>2.13</td>
</tr>
<tr>
<td>( \text{CH}_4 )</td>
<td>( 2.28 \times 10^{-5} )</td>
<td>34.3</td>
<td>( 6.85 \times 10^{-2} )</td>
</tr>
</tbody>
</table>

Table 4.3: Kinetic parameters from CuO reduction with CO, \( \text{H}_2 \) and \( \text{CH}_4 \)

### 4.2.4 Solid Fuels

The solid fuels examined in this study are: an activated carbon from Fischer Scientific, Wyodak coal, a sub-bituminous coal from the Powder River Basin, rice straw, obtained from the Biomass Collaborative at the University of California at Davis,
and corn stover, obtained from the Center for Sustainable Environment Technologies at the University of Iowa. The Wyodak coal, corn stover and rice straw studied in this work were used in their as-received form (i.e. they contained volatile matter) whereas the Fischer char was devolatilized. Ultimate and proximate analysis for these fuels are given in Table 4.4. Note that the activated carbon contains only 8% volatile matter whereas the rice straw and corn stover have greater than 60% volatile matter by mass. Additionally, the corn stover and rice straw contain a significant amount of oxygen whereas the coal and activated carbon contain much less.

These fuel samples were sieved with screens in the size ranges of 75 to 106 µm or 106 to 125 µm. Only a size distribution of the coal could be measured (shown in Fig 4.10) and this was used to show that coal particles have a mean size of 95 µm. Size distributions of the rice straw and corn stover were difficult to obtain because these samples became very sticky when placed in the multisizer solution and resulted in the Coulter counter system clogging.

In the flow reactor experiments with solid fuels, the ramp rate and final set temperature of the experiments was 10 °C per minute and 1000 °C, respectively. At the beginning of the experiments with the Wyodak coal, rice straw and corn stover, the samples were heated to just above 100 °C for at least 10 minutes to dry the samples prior to the start of the temperature ramp. To maintain gas flow through the reactor, argon at a flow rate of 250 mL per min was used as a purge gas. A summary of the CuO and solid fuel bed sample masses used in each experiment is listed in Table 4.5. As can be seen from Table 4.5, the CuO-to-solid fuel mass ratio was kept greater than 10, corresponding to lean initial equivalence ratio, to ensure that there was sufficient oxygen to convert all the carbon and hydrogen in the fuel to CO$_2$ and H$_2$O, respectively.

Figures 4.11 through 4.14 contain examples of the measured exhaust gas concentration from the reduction experiments with the activated carbon, Wyodak coal, rice straw and corn stover, respectively. The exhaust gas measurements from the coal and two biomass experiments have regions labeled D or C.O. standing for the devolatilization or char oxidation portion of the reaction.

From the CuO reduction experiment with the activated carbon (Fig. 4.11), it is
Table 4.4: Ultimate, Elemental and Heating Value Analysis

<table>
<thead>
<tr>
<th></th>
<th>Activated Carbon</th>
<th>Wyodak Coal</th>
<th>Rice Straw*</th>
<th>Corn Stover**</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ultimate Analysis, dry(%)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ash</td>
<td>3.03</td>
<td>7.57</td>
<td>17.55</td>
<td>5.43</td>
</tr>
<tr>
<td>Carbon</td>
<td>94.39</td>
<td>69.77</td>
<td>41.02</td>
<td>43.9</td>
</tr>
<tr>
<td>Hydrogen</td>
<td>0.54</td>
<td>5.65</td>
<td>4.94</td>
<td>5.62</td>
</tr>
<tr>
<td>Nitrogen</td>
<td>0.35</td>
<td>0.94</td>
<td>0.52</td>
<td>0.36</td>
</tr>
<tr>
<td>Total Sulfur</td>
<td>0.13</td>
<td>0.43</td>
<td>0.06</td>
<td>0.07</td>
</tr>
<tr>
<td>Oxygen (diff.)</td>
<td>1.56</td>
<td>15.64</td>
<td>35.91</td>
<td>44.70</td>
</tr>
<tr>
<td><strong>Proximate Analysis, dry(%)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ash</td>
<td>3.03</td>
<td>7.57</td>
<td>17.80</td>
<td>5.43</td>
</tr>
<tr>
<td>Vol. Matter</td>
<td>8.05</td>
<td>44.86</td>
<td>66.70</td>
<td>73.30</td>
</tr>
<tr>
<td>Fixed Carbon</td>
<td>88.92</td>
<td>47.57</td>
<td>15.50</td>
<td>21.30</td>
</tr>
</tbody>
</table>

(*) Rice straw was provided by the Biomass Collaborative at UC Davis, (**) The corn stover was provided by Center for Sustainable Environment Technologies at University of Iowa

Table 4.5: Experimental Conditions

<table>
<thead>
<tr>
<th>Fuel</th>
<th>CuO Size Dist. [µm]</th>
<th>Fuel Size Dist. [µm]</th>
<th>Ar Flowrate [mL/min]</th>
<th>CuO Loading [g]</th>
<th>Fuel Loading [g]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carbon</td>
<td>≤ 150</td>
<td>106-125</td>
<td>262</td>
<td>9.03</td>
<td>0.62</td>
</tr>
<tr>
<td>Wyodak Coal</td>
<td>≤ 450</td>
<td>106 to 125</td>
<td>251</td>
<td>12.03</td>
<td>0.92</td>
</tr>
<tr>
<td>Rice Straw</td>
<td>125 to 150</td>
<td>75 to 106</td>
<td>262</td>
<td>9.06</td>
<td>0.73</td>
</tr>
<tr>
<td>Corn Stover</td>
<td>125 to 150</td>
<td>75 to 106</td>
<td>251</td>
<td>15.62</td>
<td>0.89</td>
</tr>
</tbody>
</table>

evident by the appearance of CO\textsubscript{2} in the exhaust stream that the CuO and carbon begin to react at temperatures above 600 °C. This temperature roughly corresponds with the predicted temperature for O\textsubscript{2} evolution from CuO shown in Fig 4.4. The CO
CHAPTER 4. COPPER OXIDE REDUCTION CHEMISTRY

Figure 4.10: Size distribution for coal sieved with the 75 to 106 μm screens

concentration also has a slight peak coinciding with this first CO$_2$ peak. A second peak in the CO$_2$ concentration occurs at higher temperatures and this is believed to be due to the onset of CO$_2$ gasification reactions or the slow reduction of Cu$_2$O to Cu. There is little CO measured in the exhaust at temperatures above 800 °C, indicating that copper oxides react readily with CO.

Figure 4.11: CuO and Activated Carbon
The CuO reduction experiments with Wyodak coal shown in Fig. 4.12, rice straw in Fig. 4.13, and the corn stover in Fig. 4.14 have more complicated exhaust gas concentration profiles due to the fact that these fuels contained significant amounts of volatile matter. Devolatilization of coal or biomass, which in these experiments is observed to start at temperatures around 250 °C, typically results in the production of a syngas of CO, H₂ and CH₄. However, due to the fact that CuO is also present with the solid fuel, much of the evolved syngas is converted to CO₂ and H₂O as indicated by the large peak in CO₂ and the appearance of little H₂ in the exhaust gas. There was a minor amount of CH₄ in the exhaust from reduction with Wyodak coal and this supports the finding that reactions between CH₄ and CuO are slow. The CO₂ peak associated with devolitization is larger for the rice straw than coal and this is consistent with the volatile matter measured from the proximate analysis, listed in Table 4.4. There was a smaller than expected volatile peak observed from the corn stover experiment.

After the devolatilization period, the exhaust gas profiles resemble those from the CuO and activated carbon experiments: there is a peak in CO₂ that occurs at 600 °C coinciding with the evolution of oxygen from CuO and a second peak in the CO₂ concentration at 800 °C likely related to the increased rate of the gasification reactions or reduction of Cu₂O to Cu.
4.2.5 TGA Experiments

To further characterize the reduction chemistry between CuO or Cu$_2$O with CO, additional experiments were performed in the same TGA described in Chapter 2 of this thesis. Carbon monoxide was selected as the reactant for these experiments because it will likely be the dominant reductant in coal fired CLC systems. In the reported experiments, the CuO or Cu$_2$O samples were reduced in mixtures of CO
in N$_2$ or CO$_2$ at temperatures between 100 and 500 °C. While these temperatures are low relative to applications in CLC (expected to be between 800 and 1100 °C), they are needed to be this low to ensure the observed rates are chemically controlled. Additionally, the TGA being used in this work monitors weight every 5 seconds and requires 10 to 25 seconds to fill the TGA volume with the reactant gas. If the reduction rates are too fast (e.g., at higher temperatures), the resolution might be insufficient to capture all of the mass loss data and the gas concentration in the reactor will be changing with conversion.

Each TGA experiment was run as follows: (1) the TGA balance pan was loaded with a monolayer of 10 to 20 mg of CuO or Cu$_2$O particles; (2) the reaction chamber was purged with 300 mL/min of nitrogen at 20 °C for 120 minutes; (3) the temperature of the TGA was increased to the specified reaction temperature; (4) the particles were then held at temperature in N$_2$ for 60 minutes; and (5) CO was introduced into the reaction chamber at 300 mL/min. The sample mass in the TGA experiments was limited to 20 mg to minimize the influence of external mass transfer. Each test was run multiple times to ensure repeatable results. Additionally, each test was run once without a sample in the TGA pan, and this was used to make drag and buoyancy corrections to the collected data, as discussed in Chapter 2. Each TGA test was very repeatable as shown in panels of Fig. 4.15. The panels in this figure are of
thermograms from CuO reduction experiments under 1.6% CO at 200 and 300 °C. Also seen from these figures is that as the sample weight increases above 20 mg, bed diffusion begins to slow down the observed oxidation rate. In order to compare data collected with different initial masses, the thermograms were converted to extent of conversion using Eq. 2.12.

Figure 4.15: Repeatability and bed diffusion (a) 200 °C, (b) 300 °C. Note with a bed mass larger than 20 mg, bed diffusion starts to influence the observed reactivity

Based on the physical characteristics of the particles used in the experiments, diffusion limited conversion rates for selected particle size were calculated for the
highest temperature experiments performed in the TGA. These diffusion limited conversion rates were found to be larger than the experimentally observed mass loss rates. This indicates that the observed rates are chemically controlled. Based on the measured and calculated diffusion limited conversion rates, the Thiele modulus was estimated. For all the experiments run in the TGA, the Thiele Modulus was of order one, further supporting that the particle reduction rates are kinetically controlled. Consequently, it is assumed that there are no significant mass transport effects impacting the CuO and Cu$_2$O conversion rates, and the CO concentration throughout the particle is uniform.

4.2.6 TGA Experimental Results

Shown in Figs. 4.16 and 4.17 are the extent of conversion for CuO reduction to Cu with 1.6% and 5% CO in N$_2$, respectively and in Fig. 4.18 the extent of conversion of Cu$_2$O reduction to Cu in 1.6% CO in N$_2$. As observed by other authors during reduction with gaseous fuels, an induction period is observed when reducing CuO or Cu$_2$O with CO. The induction period is longest at lower temperatures, lasting about 15 minutes at 100 °C and 5% CO, and decreases in duration as the temperature increases, becoming as short as 15 to 20 seconds at 500 °C and under 1.6% CO.

Figure 4.19 is an overlay of conversion data at 300 °C for CuO reduction in 1.6% and 5% CO and Cu$_2$O reduction in 1.6% CO. From this figure, it can be seen that both the induction time and the overall rate of reduction is faster under a higher concentration of CO. Additionally from this figure it can be seen that CuO reduction is faster than Cu$_2$O reduction.

In order to estimate the reduction rate of CuO particles, the reactivity ($1/m \times dm/dt$) of the unsupported Cu$_2$O or CuO was calculated by differentiating mass loss profiles obtained from the TGA experiments. The average reactivity from the TGA experiments was extracted and is plotted in Fig. 4.9 on page 93 as a function of the experimental temperature. Also shown in Fig. 4.9 are the model fit reactivities from the flow reactor experiments with gaseous fuels (Table 4.3 on page 93), and
the calculated diffusion limited burning rate for 5, 50 and 500 μm diameter particles reacting with 1.6% CO in N₂. As can be seen, the measured TGA rates are several orders of magnitude smaller than the diffusion limited rate. It is also evident that the rates measured in the flow reactor experiments are much faster than the rates in the TGA. This is due to the higher concentration of reactive gases used. The reduction rates measured in the flow reactor using 99.9% CO is roughly two orders of magnitude faster than the rate measured in the TGA using 1.6% CO in N₂.
4.2. EXPERIMENTAL SECTION

A surface chemistry based model was developed to predict the reduction rates of CuO in CO environments and the theoretical development of the model is presented in Appendix A. In this model, CO first absorbs on copper sites and then migrate on the surface until it reacts with oxygen to become CO$_2$ and form either Cu$_2$O or Cu. However, to obtain reasonable agreement between this model and the thermograms, the copper site density had to be increased by one or two orders of magnitude. This difference is attributed to 1) the fact that as CuO reacts to form Cu, the molar density...
differences between the oxide phases cause cracks to form allowing for bulk CuO to react with gaseous CO, 2) no defect pathway was accounted for to allow the bulk reduction of the CuO.
Chapter 5

Residence Time Analysis

In Chapter 5, the results of a thermodynamic and residence time analysis of a Chemical Looping Combustion (CLC) system are presented. The thermodynamic and residence time analysis were used to estimate the recirculation rate, residence time, and inventory of several different oxygen carriers (OC) in the air and fuel reactor of a CLC system. In this chapter, the oxygen carrier inventory refers to the accumulated oxygen carrier in either the fuel or air reactor. In the thermodynamic model, it was assumed that the exhaust of the air and fuel reactors are in thermodynamic equilibrium at the specified reactor temperature and pressure. For the residence time analysis, the air and fuel reactors are modeled as well-stirred reactors with the oxygen carrier composition in and out of the air reactor and fuel reactor calculated from the thermodynamic model. Three different fuels: natural gas, syngas, and coal; and four different oxygen carrier geometries: (1) 400 µm cermet particles with a 300 µm copper core, (2) a 200 µm cermet particle with a 100 µm copper core, (3) a 20% Cu cermet particles with 10 µm Cu sub-particles, and (4) a 20% Cu cermet particles with 1 µm Cu sub-particles; were considered in the analysis. These OC geometries were selected to assess the impact of Cu particle size and Cu mass fraction on OC inventory in a CLC system. Finally, the operating conditions for the minimum OC inventory for the three fuels is presented.
5.1 Thermodynamic Model

A schematic of the system modeled for this analysis is presented in Fig. 5.1. The system consists of four components: an air reactor (AR), fuel reactor (FR), and two solid-gas separators (SGS). The inlet streams to the AR are the reduced oxygen carrier (OC) and an air stream (21% O\textsubscript{2} / 79% N\textsubscript{2}) at ambient conditions. The inlet streams to the FR are the oxidized OC and the fuel at ambient conditions. In CLC systems, the AR and FR are typically fluidized bed reactors, where air is the fluidizing agent in the AR and the gaseous fuel is the fluidizing agent in the FR. In this model, the fluidisation conditions for these reactors are not analyzed, however, when the primary fuel is a solid, exhaust gases from the FR are utilized as the fluidising agent. This recirculated gas stream is un-cooled so that there are no additional heat loads associated with recirculation / fluidisation. The AR and FR will be modeled as well mixed (i.e., isothermal) and have exhaust streams that are in thermodynamic equilibrium at the specified reactor temperature and pressure.

The SGS components in the model split the exhaust of the AR or FR into two streams, one of predominately solids and the other gases. The separators are assumed to be isothermal, require no work input, and have a separation efficiency of 95%. The separation efficiency defines the fraction of the gas that is separated from the solids stream. It is assumed that the solids can be perfectly separated from the gas exhaust stream so no OC leaves in the AR or FR exhaust. The separation efficiency in full-scale CLC system is also expected to be less than 100% and as a result, CO\textsubscript{2} will be present in the AR exhaust. The separation efficiency chosen in this model yields an AR exhaust CO\textsubscript{2} concentration that is similar to other models and experimental results \cite{128,130}. When the fuel contains ash, ash separation at the exhaust of the FR is assumed to occur perfectly to avoid its accumulation in the system.

The thermodynamic model was implemented using Matlab, where equilibrium calculations and thermodynamic properties were evaluated using the software package Cantera \cite{131}. The thermodynamic properties were evaluated using the NASA dataset \cite{132}. The code for the system model can be found in Appendix C.3. Mass and energy balances were performed for both the AR and FR and the solid-gas separators during
The energy balance for a given component $k$ is given by Eq. \[5.1\]:

$$Q_{out,net}^k - W_{in,net}^k = \sum_{j=1}^{n} \left\{ N_{j,in}^k \cdot \dot{h}_{j,in}(T_{in}, P_{in}) - N_{j,out}^k \cdot \dot{h}_{j,out}(T_{out}, P_{out}) \right\}$$

In the energy balance, $N_{j}^k$ represent the molar flowrate of the $j^{th}$ species out of $n$ total species into or out of component $k$, $\dot{h}_{j}(T, P)$ is the molar enthalpy of a particular species into or out of component $k$, and $Q_{out,net}^k$ and $W_{in,net}^k$ the net heat out and work into component $k$, respectively. When moisture or mineral matter is present in the initial fuel, they are also included in the energy balance.

The mass balance for component $k$ in the system is given by Eq. \[5.2\]:

Mass Balance: $\sum_{l=1:a} \sum_{j=1:n} a_{j,l} \cdot (N_{j,in}^k - N_{j,out}^k) = 0$ \[5.2\]

Natural gas (N.G.), a syngas composed of 50% CO/50% H$_2$ (S.G.), and Wyodak coal (W.C.) are the fuels that were considered in this analysis. The heating values
and C:H ratio of the three fuels are given in Table 5.1, the composition of the natural gas used in these calculations is given in Table 5.2 and the proximate and ultimate analysis of the Wyodak coal given in Table 4.4. From Table 5.1, it can be seen that the W.C. has the lowest C:H ratio followed by the S.G. and then the N.G., and the S.G. has the lowest heating value followed by the W.C. and N.G. As the C:H ratio decreases from 3.8 for N.G. to 1 for the W.C., more CO$_2$ is released per energy in the fuel.

When coal is the primary fuel, a recirculation ratio $\Psi_{FR}$ is used to specify the flowrate of recirculated gas, where $\Psi_{FR} = \dot{m}_{FR,\text{recirc}}/\dot{m}_{FR,\text{exhaust}}$. In this expression, $\dot{m}_{FR,\text{recirc}}$ is the amount of recirculated exhaust gas and $\dot{m}_{FR,\text{exhaust}}$ is the flowrate of the exhaust gases leaving the FR. Following the work of Peltola et al., the recirculation ratio was set to 0.4 $^{[128]}$. The Wyodak coal is assumed to have a moisture content of 20 wt.% when entering the FR. This is approximately the weight percentage of water found in the as received W.C. sample studied in this thesis $^{[133]}$. Moisture is assumed to enter as a sub-cooled liquid and the mineral matter in the solid fuel is assumed to be converted to ash instantaneously. The sensible enthalpy of the coal ash is calculated through its heat capacity, which is taken to be $c_p = 754 + 0.586 \cdot T$, J/kg-K in accordance with Merrick $^{[134]}$.

The calculations were initialized with the un-oxidized oxygen carrier (e.g., Cu or Cu / ZrO$_2$) being supplied to the air reactor. The output from this first calculation was used as the input into the CLC system model, which was then iterated (e.g., fuel reactor output going to the air reactor input and air reactor output into to the fuel reactor) until steady-state was reached.

<table>
<thead>
<tr>
<th>Fuel</th>
<th>C to H ratio</th>
<th>LHV (MJ/kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural Gas</td>
<td>1:3.8</td>
<td>43</td>
</tr>
<tr>
<td>Syngas</td>
<td>1:2</td>
<td>17</td>
</tr>
<tr>
<td>Wyodak Coal</td>
<td>1:0.8</td>
<td>25</td>
</tr>
</tbody>
</table>
Table 5.2: Composition of natural gas used in this study

<table>
<thead>
<tr>
<th>Species</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>CH$_4$</td>
<td>90.7%</td>
</tr>
<tr>
<td>C$_2$H$_6$</td>
<td>3.6%</td>
</tr>
<tr>
<td>C$_3$H$_8$</td>
<td>1.9%</td>
</tr>
<tr>
<td>N$_2$</td>
<td>1.8%</td>
</tr>
<tr>
<td>CO$_2$</td>
<td>1%</td>
</tr>
<tr>
<td>O$_2$</td>
<td>1%</td>
</tr>
</tbody>
</table>

5.1.1 Parametric Study Results

To illustrate the capabilities of the model, it was run for a range of air and copper flowrates with the natural gas, syngas and Wyodak coal, which are described above, as the fuels. For the system, the temperature and pressure of both the air and fuel reactor were set to 800 °C and 1 atm, respectively. The temperature and pressure were fixed at these values in order to reduce the number of parameters being changed in the analysis. To ensure high fuel utilization within the system and minimal emissions of un-burnt hydrocarbons, only air and OC flowrates where the fuel species (e.g., CO, H$_2$, CH$_4$) concentration in the exhaust of the FR is less than 500 ppm are presented.

The results from this parametric study are shown in Figs. 5.2 through 5.6 as a function of the equivalence ratios in the AR and FR. The equivalence ratios shown in the plots are defined on an atom basis via Eq. 5.3a for the fuel reactor and Eq. 5.3b for the air reactor. Based on these definitions, a higher air flowrate corresponds to a lower equivalence ratio in the air reactor, and a higher OC flowrate corresponds to lower equivalence ratio in the fuel reactor.

The CO and H$_2$ gas concentration in the exhaust of the FR is below 500 ppm when $\phi_{FR} < 0.95$. When $\phi_{FR} < 0.95$, the fraction of the carbon exiting the FR as graphite, presented in Fig. 5.2b, is also always less than 0.1 ppm. The dotted line in Fig. 5.2b is the limit where the CO and H$_2$ concentration in the exhaust is greater
than 500 ppm.

\[
\phi_{FR} = \frac{\text{Fuel/Air}_{\text{actual}}}{\text{Fuel/Air}_{\text{stoic}}} = \frac{(2 \cdot \dot{N}_{C} + 1/2 \cdot \dot{N}_{H} - \dot{N}_{O})}{2 \cdot \dot{N}_{O^C}}
\]

\[
\phi_{AR} = \frac{\text{Fuel/Air}_{\text{actual}}}{\text{Fuel/Air}_{\text{stoic}}} = \frac{\dot{N}_{Cu} - \dot{N}_{O^C}}{\dot{N}_{O^C}^{Air}}
\]

Shown in Fig. 5.2 is a contour plot of the O\textsubscript{2} mole fraction in the AR exhaust. The O\textsubscript{2} concentration in the exhaust of the AR is needed to calculate the oxidation rate of the OC particles used in the residence time analysis in the next section. From this figure it can be seen that as \(\phi_{AR}\) decreases (corresponding to a higher air flowrate), the concentration of O\textsubscript{2} in its exhaust increases. When 0.4 < \(\phi_{AR}\) < 2, the O\textsubscript{2} in the exhaust is typically between 0.14 and 0.02, and this finding is consistent with the predictions of other models [128–130].

Figure 5.3 is an illustration of the excess energy in the air and fuel reactors as a function of the AR and FR equivalence ratio for the three different fuels considered in this study. It can be seen from Fig. 5.3a it can be noted that the AR has excess energy (\(Q_{in,AR} < 0\)) when \(\phi_{AR} > 0.5\), and when \(\phi_{AR} < 0.5\), the air reactor requires energy as an input (i.e., \(Q_{in,AR} > 0\)). Less energy is available as the equivalence ratio of the AR decreases. This is because as \(\phi_{AR}\) decreases, the flowrate of air to the AR increases and more energy is required to maintain the excess air at the AR temperature. For a fixed AR equivalence ratio (e.g., \(\phi_{AR} = 1\)), less energy is given off by the AR as the fuel equivalence ratio decreases. This occurs because there is more OC flowing through the system and less of it is reduced to Cu in the FR. As noted in the figure, the AR releases the most energy when N.G. is the fuel and releases the least amount of energy when syngas is the fuel. This is because the N.G. has the largest heat of reaction as noted in Table 5.3.

The FR always has excess energy (Fig. 5.3b, e.g., \(Q_{in,FR} < 0\)) for the conditions that were considered in this analysis. As the FR equivalence ratio decreases, by increasing the OC flowrate, more heat is released from the FR. This is because \(Cu_{2}O\) is the favored product over Cu from reduction in the FR and the reduction of CuO.
5.1. THERMODYNAMIC MODEL

Figure 5.2: a) the $O_2$ concentration in the exhaust of the AR, b) the carbon fraction in the exhaust of the FR

$Cu_2O$ is more exothermic than $CuO$ to $Cu$ or $Cu_2O$ to $Cu$. This can be seen by considering the heats of reduction listed in Table 5.3.

The circulation rate of the OC on a support-free-basis leaving the AR and FR per MW$_{th}$ of fuel is presented in Fig. 5.4. In this figure, lines of the same color are evaluated at the given equivalence ratio where the higher of the two lines is the flowrate leaving the AR and the lower line is the flowrate leaving the FR. The AR flowrate is always higher than the FR flowrate because it contains the oxidized form of the OC. As the equivalence ratio in the AR decreases, a smaller OC reticulation rate is required by the system because the OC reaches a higher degree of conversion.
leaving the AR. Also, it can be seen from this figure that N.G. requires the highest recirculation rate per MW\textsubscript{th} released by the fuel followed by S.G. and then W.C. This is because the N.G. requires the most oxygen to be converted to combustion products. The recirculation rates estimated from the thermodynamic model are within ±10% of similar to results presented in the literature \[22,39,128,135,136\].

The conversion of the OC through AR and FR, and the corresponding mass fractions of Cu, Cu\textsubscript{2}O and CuO of the OC in the AR and FR exhaust are shown in Fig. 5.5 and Fig. 5.6, respectively. As the OC is cycled, its conversion and composition vary according to these plots. The set of curves in the lower half of Fig. 5.5 and in
Table 5.3: heat of reaction for copper oxides with CH₄, H₂, CO and C_{gr}

<table>
<thead>
<tr>
<th>Reaction</th>
<th>CH₄</th>
<th>H₂</th>
<th>CO</th>
<th>C_{gr}</th>
</tr>
</thead>
<tbody>
<tr>
<td>CuO+fuel → Cu</td>
<td>-178</td>
<td>-85.8</td>
<td>126.9</td>
<td>-81.4</td>
</tr>
<tr>
<td>CuO+fuel → Cu₂O</td>
<td>-236.6</td>
<td>-100.4</td>
<td>-141.6</td>
<td>-110.7</td>
</tr>
<tr>
<td>Cu₂O+fuel → Cu</td>
<td>-119.5</td>
<td>-71.1</td>
<td>-112.3</td>
<td>-51.1</td>
</tr>
</tbody>
</table>

Figure 5.4: OC recirculation rate exiting the AR and FR per MWₜₜ of fuel (LHV·m_Fuel)

Fig. 5.6a correspond to the exhaust state of the FR and the upper set of curves in Fig. 5.5 and in Fig. 5.6b correspond to the exhaust state of the AR. In Fig. 5.5
the extent of conversion for the FR and AR is defined relative to CuO (Eq. 2.12), where full conversion of the OC exiting the AR corresponds to 1 and full conversion of the OC exiting the FR corresponds to 0. When the equivalence ratio in the FR is near 1, there is insufficient oxygen to convert all of the fuel to combustion products and the OC exiting the FR has a value very close to 0 (full conversion). As excess OC is introduced into the FR, \( \phi_{\text{FR}} < 1 \), the OC leaving the FR is mostly Cu and Cu\(_2\)O and contains minor amounts of CuO. Under all of the conditions considered, the fuel type has very little impact on the FR conversion or the composition leaving the AR or FR. This is because the exhaust states of the AR and FR are determined by thermodynamic equilibrium \((T, P\) and \(N_i)\).

![OC conversion from the exhaust of the fuel and air reactors with N.G., S.G. and W.C. as the fuels](image)

Conversion in the AR is not sensitive to \( \phi_{\text{FR}} \) when the AR is operated with excess oxygen \((\phi_{\text{AR}} < 1)\). Only when the AR is oxygen deficient \((\phi_{\text{AR}} > 1)\) is conversion in the AR sensitive to \( \phi_{\text{FR}} \). When \( \phi_{\text{AR}} < 1 \), corresponding to a scenario with excess air, equilibrium predicts the particles will reach 95% conversion at 800 °C. When \( \phi_{\text{AR}} > 1 \), conversion of the OC leaving the AR drops as low as low as 55%, and from Fig. 5.6b, the OC will be a mixture of Cu\(_2\)O and CuO.
5.2 Residence Time Analysis

Chemical reactor theory was used to estimate OC inventory ($m_{OC}$; accumulated oxygen carrier in either the fuel or air reactor) and residence time ($\tau_{OC}$) necessary to react the fuel and oxygen carrier in the FR, and regenerate the OC in the AR to equilibrium products. The results from this analysis were then compared to other models presented in the literature that also attempt to estimate the inventory of the AR and FR of CLC systems $[36, 39, 128, 130, 135, 137, 139]$. These models predict inventories for copper-based oxygen carriers to be between 100 and 600 kg per MW$_{th}$
in the fuel reactor and 10 and 100 kg per MW$_{th}$ in the air reactor. While many of these models have a high level of detail characterizing system components expected to be in a real CLC system, most of the models use kinetic data collected from lab scale experiments and assume a global reaction rate when estimating the AR or FR size. This differs from the present analysis because the conversion of the OC in the AR is estimated utilizing the oxidation model presented in Chapter 3. Consequently, the presented predictions of the AR inventory are sensitive to the air-to-OC recirculation rate.

In the presented calculations, the air and fuel reactors are modeled as a well stirred reactors. In accordance with the well stirred reactor model, the AR and FR are assumed to be well mixed, have an exit stream with concentration equal to that of the contents within the reactor, and be in steady state. The conversion of the oxygen carrier through the air and fuel reactors are related to the OC inventory, $m_{OC}$ through Eq. (5.4a). In this equation, $\dot{N}_{Cu}$ is the molar flowrate of Cu, $\xi_{out}$ and $\xi_{in}$ are the OC conversion in and out of the reactor, $R_{OC}^{ave}$ is the average area specific burning rate (g/m$^2$-s) defined by Eq 3.8, and $S_g$ the mass specific surface area of the OC. The residence time of the reactor is then determined by Eq. (5.4b). In this section, the system residence time and reactor inventory are calculated assuming kinetically limited oxidation and reduction rates. Kinetically limited rates were assumed because they reflect the fastest the particles can react under the given conditions.

$$\dot{m}_{OC}^in - \dot{m}_{OC}^in = (\xi_{out} - \xi_{in}) \cdot \dot{N}_{Cu} \cdot \frac{MW_O}{MW_{Cu}} = R_{OC}^{ave}(T, P_{O2}) \cdot S_g^OC \cdot m_{OC} \quad \quad \text{(5.4a)}$$

$$\tau_{OC} = \frac{m_{OC}}{\dot{m}_{OC}} \quad \quad \text{(5.4b)}$$

The OC oxidation rate in the AR was determined using the models presented in Chapters 2 and 3, and OC reduction rates in the FR were calculated using the models presented in Table 4.3 of Chapter 4. The OC oxidation rate was evaluated as a function of the air and fuel reactor equivalence ratios. At FR equivalence ratios less than 0.9, the different fuel types have little impact on the air reactor size. Consequently, in this range of FR equivalence ratios, the AR results will not vary
5.2. RESIDENCE TIME ANALYSIS

significantly for the natural gas, the syngas or coal. The inputs for the oxidation model are the initial OC composition, the equilibrium \( \text{O}_2 \) concentration exiting the AR, and the final conversion of the OC exiting the AR. These parameters were calculated from the thermodynamic model presented in the previous section and values for these parameters can be found in Fig. 5.6b, Fig. 5.2a and Fig. 5.5 respectively.

The OC reduction rate in the FR was evaluated for a range of fuel and air reactor equivalence ratios. The overall reduction rates for CuO with N.G. and S.G. were approximated using the kinetic parameters in Table 4.3. In this approach, it is assumed that the reactivity of CuO particles does not vary with the size. While this may not be necessarily true, there are no models that characterize CuO reduction with particle size. The CuO reduction rate with N.G. was approximated with the CuO/CH\textsubscript{4} rate parameters and the CuO reduction rate with syngas was determined with the CuO/CO and CuO/H\textsubscript{2} rate parameters, assuming the CuO-syngas reaction rate is the mole weighted average of H\textsubscript{2} and CO reduction: \[ R_{\text{syngas}} = X_{\text{CO}} \cdot \hat{R}_{\text{CO}} + X_{\text{H}_2} \cdot \hat{R}_{\text{H}_2}. \]

The actual reduction rate of a mixture of CO and H\textsubscript{2} will likely not be a simple linear combination of the two rates, however due to insufficient experimental data, this will be taken as the first approach. The reduction rates with CH\textsubscript{4}, CO and H\textsubscript{2} were evaluated assuming the fuel reactor inlet concentrations.

As was mentioned in Chapter 1, when a solid fuel is supplied directly to the FR, the OC conversion rate will be limited by the rate of char gasification. In this way, the FR should be sized to provide a sufficient residence time for the solid fuel to be completely gasified to a syngas. Once the syngas is created, it will react readily with the OC to form combustion products [140]. Typical gasifying agents are mixtures of CO\textsubscript{2} and H\textsubscript{2}O and this will likely be supplied by re-circulating FR exhaust gases. Carbon monoxide and H\textsubscript{2} can be strong inhibitors of gasification chemistry [140]. However, since CO and H\textsubscript{2} chemistry with Cu\textsubscript{2}O and CuO is quite fast, the expected concentration of CO and H\textsubscript{2} in the FR should be sufficiently low. As a result, the gasification rate in FRs should not be significantly inhibited and will predominantly depend on the concentrations of CO\textsubscript{2} and H\textsubscript{2}O [141].

The gasification rates that were used in this analysis were taken from gasification experiments performed in a TGA with mixtures of CO\textsubscript{2} in N\textsubscript{2}, H\textsubscript{2}O in N\textsubscript{2}, or mixtures
of H$_2$O, CO$_2$ and N$_2$. In these experiments, the intrinsic reactivity $R_{\text{char}}$ of the W.C. char at 800 °C and 1 atm was measured to be between $3.3 \times 10^{-7}$ to $3.3 \times 10^{-6}$ [g/m$^2$-s] depending on the gasifying mixture \cite{142}. This coal char has an apparent density of 1 g/cm$^3$ and a mass specific surface area of 300 m$^2$/g. Given the reactivity of this char, the corresponding reactivity of the OC can be determined via Eq. \ref{eq:5.5}, where in this expression the factor of 2 arises because every carbon atom requires 2 oxygen atoms to form CO$_2$.

$$R_{\text{OC}} = R_{\text{char}} \cdot \frac{MW_O}{MW_C} \cdot 2 \quad (5.5)$$

The residence time of the char particle was computed by integrating Eq. \ref{eq:5.6a}, assuming a constant mass loss rate ($R_{\text{char}} \times S_g$), using the reactivity and mass specific surface area given by Tilghman et al. \cite{142}. Upon integration, it can be shown that the residence time is given by Eq. \ref{eq:5.6b}. The residence time of the char particle can then be determined as a function of the equilibrium char conversion in the FR, which is an output from the thermodynamic model.

$$1/m_p \cdot \frac{dm_p}{dt} = -R_{\text{char}} \cdot S_{\text{char}}^g \quad (5.6a)$$

$$\tau_{\text{char}} = \frac{\ln(m_o/m_p)}{R_{\text{char}} \cdot S_{\text{char}}^g} \quad (5.6b)$$

Four oxygen carriers were considered in the residence time analysis, denoted OC1 through OC4, and they are: OC1– a 500 µm cermet particle with a 400 µm Cu core, OC2– a 200 µm cermet with a 100 µm Cu core, OC3– 200 µm cermet particle, 20% Cu / 80% ZrO$_2$ with 10 µm Cu sub-particles, and OC4– 200 µm cermet particle, 20% CuO / 80% ZrO$_2$ with 1 µm Cu sub-particles. Physical properties describing the particles can be found in Table \ref{table:5.4}. The copper in all of the particles is assumed to be non-porous and the mass specific surface area of the copper is assumed to be its geometric value ($S_{\text{Cu}}^g = 6/(\pi \cdot r \cdot \rho_A)$, where $r$ is the radius of the particle. The OC mass specific surface area, $S_{\text{OC}}^g$, is then defined as the quotient of the active surface area to the OC particle mass where the active surface area is based on the Cu or CuO
5.2. RESIDENCE TIME ANALYSIS

particles and the OC mass, which includes the Cu/Cu$_2$O/CuO and ZrO$_2$ support, determined from geometry. For the reduction calculations, the mass specific surface area during reduction is assumed to be constant. In actuality, the specific surface area relevant to the reactive material (Cu$_2$O or CuO) will increase as the particle is reduced similar to a shrinking sphere (see Chapter 4), however since this has been un-characterized for the particles, it is assumed to be constant in the present analysis.

Table 5.4: Properties of the OC particles, $^+$ denotes the fully reduced weight fraction and $^*$ denotes the CuO fully oxidized weight fraction

<table>
<thead>
<tr>
<th></th>
<th>OC1</th>
<th>OC2</th>
<th>OC3</th>
<th>OC4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Particle Diameter, µm</td>
<td>400</td>
<td>200</td>
<td>200</td>
<td>200</td>
</tr>
<tr>
<td>Copper Diameter, µm</td>
<td>300</td>
<td>100</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>Cermet Particle Form</td>
<td>Core</td>
<td>Core</td>
<td>Distributed</td>
<td>Distributed</td>
</tr>
<tr>
<td>% Cu / ZrO$_2^+$</td>
<td>89%</td>
<td>34%</td>
<td>20%</td>
<td>20%</td>
</tr>
<tr>
<td>% CuO / ZrO$_2^*$</td>
<td>91%</td>
<td>39%</td>
<td>24%</td>
<td>24%</td>
</tr>
<tr>
<td>Porosity, [-]</td>
<td></td>
<td></td>
<td>50%</td>
<td></td>
</tr>
<tr>
<td>$S_g$, m$^2$/g</td>
<td>1</td>
<td>0.1</td>
<td>$3.8 \times 10^{-3}$</td>
<td>$2.9 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

5.2.1 Air Reactor Results and Discussion

The average reaction rate, OC inventory per MW$_{th}$ heat released from the AR and residence time of the four OC particles are presented in Figs 5.7 to 5.9 respectively for $\phi_{FR} = 0.8$. As expected, the cermet particles with the smallest copper diameter react the fastest and require the smallest OC inventory. Considering Fig. 5.7, the average reactivity of the OC particles increases as the AR equivalence ratio increases above 1. This occurs despite there being a lower O$_2$ concentration (Fig. 5.2a) in the AR because the initial composition of the OC has a higher Cu mass fraction (Fig. 5.6b) and the Cu oxidation rate is greater than the Cu$_2$O oxidation rate.

The data presented in Fig. 5.8 is of the OC inventory per MW$_{th}$ energy released from the AR. The values presented in this figure include the weight of the support material. The percentages given in the labels are of the Cu content in the initial
particle. From this figure, it is apparent that the size of the Cu particle strongly impacts the OC inventory. Even though OC1 has a higher percentage of Cu than the other OCs, it requires a larger inventory because it reacts much slower. The data in this figure also indicates that the AR inventory decreases as the air reactor is operated at higher equivalence ratios (corresponding to lower air flowrates). This is because more heat is released from the reactor at higher equivalence ratios. As the air flowrate increases and the equivalence ratio approaches 0.4, the oxygen carrier inventory rapidly increasing. This occurs because the heat released from the air reactor approaches 0 (see Fig. 5.3a).

Figure 5.7: Average Reaction Rate in the AR, $\phi_{FR} = 0.8$

![Average Reaction Rate in the AR, $\phi_{FR} = 0.8$](image)

Figure 5.8: Oxygen Carrier Inventory of the AR, $\phi_{FR} = 0.8$

![Oxygen Carrier Inventory of the AR, $\phi_{FR} = 0.8$](image)
5.2. RESIDENCE TIME ANALYSIS

The average residence time of the particles in the AR is shown in Fig. 5.9. As expected, the OC with the smallest copper particles has the shortest residence time and the OC with the biggest copper diameter has the longest residence time. OC4 reacts approximately 100 times faster than OC3, and greater than $10^4$ times faster than OC1 or OC2. These predicted residence times are consistent with values published by Eyring et al. [39], who estimated the residence time of the AR to be 50 seconds for similarly sized particles, and Peltola et al. [128] who estimated the residence time of the AR to be between 10 and 50 seconds.

![Image](image.png)

Figure 5.9: Residence Time of the OC through the AR, For a fuel reactor equivalence ratio of 0.8

Given that OC4 has the lowest AR inventory, the model was used to predict the OC inventory over a range of AR and FR equivalence ratios to determine if there is a global minimum in the AR inventory. The results of these calculations are in Fig. 5.10a, 5.10b, 5.10c when N.G., S.G. and W.C., respectively, are the primary fuel. From these figures, it can be seen that the OC inventory in the AR does not vary significantly with the fuel type. The AR inventory is highest for low values of $\phi_{AR}$ and decreases as $\phi_{AR}$ and $\phi_{FR}$ increase. The smallest OC inventories occur when $\phi_{AR}$ is roughly between 1.2 and 2 and $\phi_{FR}$ is between 1 and 0.8. Under these reactor conditions, the overall OC reaction is characterized by: $\text{Cu} + \frac{1}{4} \text{O}_2 \rightarrow \frac{1}{2} \text{Cu}_2\text{O}$. This can be seen a little clearer when considering the composition of the OC between the inlet and the outlet of the AR, which is presented in Fig. 5.6a and 5.6b.
Fig. 5.6a, it can be seen that the composition of the OC initially into the AR is Cu with minor parts Cu₂O, and is oxidized to a particle of mostly Cu₂O with minor amounts of CuO.

![Figure 5.6a](image)

Figure 5.10: Air reactor inventory of for OC4 as a function of the AR and FR equivalence ratio when a) N.G., b) S.G. and c) W.C. are the primary fuel.

5.2.2 Fuel Reactor Results and Analysis

The average reaction rate of the OC or the Wyodak coal, the OC inventory per MWₜₕ heat released from the FR, and the residence time of the OC or the Wyodak coal through the FR are presented in Figs. 5.11 through 5.13 for $\phi_{AR} = 1$. Shown in Fig. 5.11 is an illustration of the average reaction rate of the four OC particles (see Table 5.4) with natural gas (N.G.) and syngas (S.G.), and the upper and lower limit for the gasification rate of the W.C. provided by Tilghman et al. [142]. Illustrated in Fig. 5.11 is the OC inventory per heat released from the FR reacting with the N.G.,
5.2. RESIDENCE TIME ANALYSIS

S.G., and W.C. Additionally, in Fig. 5.13 is the residence time of the OCs reacting with N.G. and S.G., and the residence time of the W.C. char particle reacting at its lower and upper gasification rate ($3.3 \times 10^{-7}$ to $3.3 \times 10^{-6} \text{[g/m}^2\text{-s]}$).

Considering Fig. 5.11 it can be seen that the OC reacts more readily with the S.G. than either the N.G. or W.C. At 800 °C, the syngas is expected to be approximately 10 times more reactive than the N.G. with the OC. Considering the OC reacting with N.G. or S.G., the average reaction rate increases as the OC particle diameter decreases. This is predicted due to an increase in the specific surface area of the OC, which occurs as the characteristic copper diameter decreases. The average reaction rate of the OC with the S.G., N.G. and the W.C. are independent of the FR equivalence ratio because the reactivity of the OCs are constant in the model.

![Figure 5.11: Average Reaction Rate in the FR, $\phi_{AR} = 1$](image)

The FR OC inventory per MW\textsubscript{th} of heat released in the FR when N.G. and S.G. are the fuels is presented in Fig. 5.12a and in Fig. 5.12b when W.C. is the fuel. The labels in these figures include the fraction of CuO in the initial particles. The model predicts that OC4 will require the smallest inventory followed by OC3, OC1 and OC2. Unexpectedly, OC1 is predicted to have a smaller inventory than OC2. The model predicts OC1 to have a smaller inventory than OC2 because OC1 has a smaller fraction of inert material. OC4 has the fastest rate because it has the largest mass specific surface area. From Fig 5.12a and 5.12b it is also apparent that the W.C. requires the largest OC inventory followed by the N.G. and the S.G. This
is primarily due to the fact that the OC is most reactive with the S.G. Using the kinetic parameters in Table 4.3 for N.G. and S.G. and the char reactivity mentioned above, the reactivity of the OC with S.G., N.G. and W.C. at 800 °C and 1 atm were estimated to be 0.4, 0.05 and .9 x 10^{-7} to 8.9 x 10^{-6} g/m^2-s, respectively. Similar solid inventories were estimated by Abad et al. [130] for Cu based oxygen carriers reacting with N.G. and S.G.

The overall residence time of the OC particles in the FR at 800 °C is presented in Fig. 5.13. The model predicts that OC4 will have the smallest residence time followed by OC3, OC1 and OC2. The residence time labeled W.C. is the time required to burn
5.2. RESIDENCE TIME ANALYSIS

a char particle out to equilibrium (Fig. 5.2b). The range of residence times predicted by the model for the W.C. char are two to three orders of magnitude higher than all of the OC considered in this analysis. This is consistent with other works that estimate FR inventory size when coal is the fuel and this is often the reason given to motivate researchers related to CLOU (an option where the solid fuel is burned in oxygen instead of gasified and as a result has a much shorter residence time) [37, 40, 135]. When assuming char conversion is limited by oxygen release from CuO, Adanez-Rubio et al. [36] have estimated OC inventory as low as 43 kg/MW$_{th}$ at 930 °C when using a Colombian Coal.

![Figure 5.13: Residence Time of the FR, $\phi_{AR} = 1$](image)

Of the OCs considered, OC4 also has the smallest inventory in the FR. In order to determine if there is a minimum FR inventory of OC4, the model was evaluated over a range of FR and AR equivalence ratios and results from these calculations are shown in the panels of Fig. 5.14 when N.G., S.G. and W.C. are the primary fuels. The calculation considering the W.C. use the faster of the two reduction rates considered (i.e., $3.3 \times 10^{-6}$ [g/m$^2$-s]). From the panels of Fig 5.14 it is evident that the OC inventory in the FR varies significantly with the fuel type, where the W.C. requires the largest inventory followed by N.G. and then the S.G. This is due to the different reduction rates with the three fuels mentioned above (S.G. 0.4 g/m$^2$-s, N.G. 0.05 g/m$^2$-s and W.C. 8.9 $\times 10^{-6}$ g/m$^2$-s). Based on these calculations, the FR
inventory decreases as \( \phi_{AR} \) and \( \phi_{FR} \) decrease.

![Graphs](image)

Figure 5.14: Fuel reactor inventory of for OC4 as a function of the AR and FR equivalence ratio when a) N.G., b) S.G. and c) W.C. are the primary fuel

### 5.2.3 CLC System Analysis Results and Discussion

The overall inventory of the CLC system (AR+FR) will strongly depend on the selected OC, fuel, air flowrate and the OC recirculation rate. The results from the presented model indicate that the OC inventory is smallest when the AR is operated at a high equivalence ratio (low air flowrate) and the FR operated at a low equivalence ratio (high oxygen carrier flowrate). To determine the AR and FR equivalence ratios that minimize the OC inventory in system, the AR inventory presented in Fig. 5.10 was added with the FR inventory presented in Fig. 5.14 and the total OC inventory of OC4 was determined. The system inventory when N.G., S.G. and W.C. are the fuel are shown in the panels of Fig. 5.15. From these figures, it is evident that the
fuel type has a significant impact on the OC inventory in the system. The S.G. has the smallest inventory, followed by the N.G. and then the W.C. When N.G. and W.C. are the primary fuel, the model indicates that smaller OC inventories can be achieved by operating the AR and FR at lean equivalence ratios (\( \phi_{\text{AR}} < 1 \) and \( \phi_{\text{FR}} < 0.6 \)). However, when the S.G. is the primary fuel, the smallest inventories occur when \( 1 < \phi_{\text{AR}} < 1.4 \) and \( \phi_{\text{FR}} > 0.6 \). The inventory maps when N.G. and W.C. are the primary fuel (Fig. 5.15a and 5.15c) have similar trends because the FR inventory dominates the character of the system.

Figure 5.15: Oxygen carrier inventory for the AR and FR when a) N.G., b) S.G. and c) W.C. are the fuel
Chapter 6

Summary of Major Results and Conclusions

Chemical looping combustion (CLC) is a novel combustion strategy that can enable the capture of CO₂ produced from burning fossil fuels to generate electricity. There are a number of advantages to using CLC technology over the alternative carbon capture strategies with the primary benefit being that CLC systems do not use additional energy to separate CO₂ from the flue gas exhaust. This is unlike alternative carbon capture strategies which typically require in excess of 20% of a power plant’s thermal energy to separate CO₂ from the flue gas exhaust. As a result, CLC systems are expected to be nearly as efficient as traditional power plants, however with the significant advantage of being able to capture CO₂. One obstacle hindering the development of CLC systems is that there are very few models capable of predicting oxygen carrier conversion through the air and fuel reactors as a function of particle morphology, air and fuel flowrate, and system temperature and pressure.

To support efforts in this area, this thesis is focused on the development and application of an oxidation model that describes the fundamental processes occurring during metal oxidation. This thesis also documents the development of a flow reactor test facility that was used to obtain reduction rate data of unsupported copper oxide particles reacting with a variety of gaseous and solid fuels. The reduction rate data were used to determine parameters in the reduction model. Using both the oxidation
and reduction models, the oxygen carrier inventory of a 10 MW<sub>th</sub> CLC system was estimated as a function of different oxygen carrier geometries and conditions in the air and fuel reactors.

Based on the work performed in this thesis, it can be concluded that copper is an acceptable choice for the base metal of oxygen carriers used in CLC systems. Additionally, from this work it can also be concluded that CLC systems can be optimized to further reduce the sizes of the air and fuel reactors. The results also demonstrated that in the case of copper-based oxygen carriers, by designing a system that cycles the oxygen carrier between Cu and Cu<sub>2</sub>O as opposed to Cu and CuO (or Cu<sub>2</sub>O and CuO), faster conversion rates can be achieved. By enabling the air and fuel reactors in CLC systems to be smaller, this will reduce the initial cost of building a CLC system, thereby improving the likelihood that CLC systems are built in the future.

6.1 Oxidation

In Chapter 2 a model is presented that is capable of predicting Cu and Cu<sub>2</sub>O oxidation rates in high-temperature, oxygen-rich, environments. The model is based on the principals of defect diffusion and Wagner Theory, and relates the oxidation rate to the flux of defects through a growing oxide layer. Using this framework with fundamental parameters like defect reaction enthalpies, the Cu<sub>2</sub>O and CuO band gaps, and vacancy diffusion coefficients, it was possible to predict oxidation rates of copper particles in a TGA at temperatures between 500 and 800 °C and 6% and 21% O<sub>2</sub> at 1 atm. The model utilizes defect diffusion coefficient that were extracted from wire oxidation experiments performed in a flow reactor at temperatures between 600 and 900 °C, in 21% O<sub>2</sub>, under 1 atm. The extracted vacancy- and self- diffusion coefficients in Cu<sub>2</sub>O were found to be in good agreement with previously published diffusion coefficient values. The model presented in Chapter 2 is also one of the first defect chemistry models of cupric oxide. Using estimates of the defect concentration in CuO, a reaction enthalpy for the creation of doubly ionized vacancies V<sup>′′</sup> Cu<sub>a</sub> was estimated to be 250 kJ/mole. The extracted vacancy diffusion coefficient in CuO
was found to have an activation energy of 32 kJ/mole and using the presented defect chemistry model, a self-diffusion coefficient was calculated and found to have a similar activation energy to the self-diffusion coefficient extracted by other researchers.

The oxidation model presented in Chapter 2 supports findings that Cu oxidation to Cu$_2$O and CuO can be modeled by Wagner Theory and is characterized by parabolic growth. Additionally, under the conditions considered, Cu oxidation to CuO progresses through Cu$_2$O: Cu $\rightarrow$ Cu$_2$O $\rightarrow$ CuO. From this modeling effort, it was shown that the dominant defects in Cu$_2$O are neutral and singly ionized vacancies and that interstitials are less important at the temperatures between 600 and 900 °C in 21% O$_2$. Additionally, the interstitial concentration and interstitial diffusion coefficients in CuO were determined to be too low to contribute to the overall oxidation rate. The findings from Chapter 2 also support the conclusion that CuO is a charge transfer insulator, where holes and electrons are bound to oxygen and copper ions, respectively, instead of being delocalized in the conduction and valance band. Finally, the model was used to demonstrate that Cu oxidation to Cu$_2$O is significantly faster than Cu$_2$O oxidation to CuO. This is because the defect concentration and defect diffusion coefficients in Cu$_2$O are higher than they are in CuO.

The fundamental oxidation model presented in Chapter 2 was then applied to predict the oxidation rates of Cu particles supported with ZrO$_2$. The cermet model, presented in Chapter 3, is one of the first applications of defect chemistry to predict oxidation rates of oxygen carrier particles designed for use in CLC systems. The model was used to show that faster oxidation rates are achievable by reducing the characteristic size of the copper particles.

### 6.2 Reduction

In Chapter 4, theoretical and experimental work related to CuO reduction chemistry in high temperature environments is presented. While the processes occurring during reduction have been studied with great detail, actual reduction rates are difficult to predict because they are highly sensitive to the oxide surface conditions. Additionally, the surface structure of CuO can also vary as the particle reacts. This is a result of
the fact that CuO densifies as it reacts to form Cu, and this can lead to cracks in the Cu product layer which are a pathway for bulk CuO to be exposed to the fuel.

In order to estimate the rate at which oxygen is removed from CuO, reduction experiments with various gaseous and solid fuels were conducted in a flow reactor and in a thermogravimetric analyzer. From these experiments, it was observed that there are two pathways for CuO reduction: a direct pathway observed at higher temperatures, where CuO is reduced to Cu, and an indirect pathway observed at lower temperatures, which has Cu₂O as an intermediate. The experimentally observed rates support the conclusion that CuO reacts more readily with CO and H₂ than with CH₄, and the oxidation of solid fuels with CuO likely occurs via a gasification process.

6.3 System Analysis

The oxidation model and reduction rate data was used in conjunction with results from a thermodynamic model to estimate the OC inventory and residence time of a 10 MWth CLC system. From this analysis, it was shown that oxygen carrier particles with the smallest characteristic copper size require the smallest oxygen carrier inventory because smaller copper particles have a higher mass specific surface area and react faster than larger diameter copper particles. The results from the residence time analysis show that the AR reactor inventory is minimized when the Cu in the OC cycles between Cu and Cu₂O in the air and fuel reactors. This is mainly because Cu oxidation to Cu₂O is much faster than Cu or Cu₂O oxidation to CuO. The OC inventory in the fuel reactor depends very strongly on the kinetic rate between the OC and the fuel. The faster the OC reacts with the fuel, the smaller the OC inventory will be. The FR OC inventory is smallest when a syngas is the fuel followed by natural gas and then the Wyodak coal. Based on the residence time analysis, the FR inventory is smallest when the OC-to-fuel flowrate is highest.

The results from the AR and FR were combined to show that there is a minimum OC inventory that occurs when varying the air and oxygen carrier flowrates for a given fuel flowrate, reactor temperature and pressure. Based on the model results, it can be concluded that when natural gas and Wyodak coal are the fuel, the fuel
and air reactor should be run under oxygen rich conditions with excess air to the air reactor and excess oxygen carrier to the fuel reactor. This corresponds air and fuel reactor equivalence ratios of $\Phi_{AR} < 0.8$ and $\Phi_{FR} < 0.6$. Additionally, it can also be concluded that when syngas is the fuel, the system inventory will be minimized when the air reactor is run under stoichiometric conditions and the fuel reactor run with excess oxygen carrier. This occurs for air and fuel reactor equivalence ratios of $0.8 < \Phi_{AR} < 1.3$ and $0.7 < \Phi_{FR} < 0.1$.

6.4 Future Work

While the oxidation model is an advancement from the phenomenological models used by the CLC community, much more work is still needed to improve the design of oxygen carrier particles for use in CLC systems. One refinement to the cermet model that could further improve its predictive capability would be to account for the variation of porosity as the copper in the particle reacts to copper oxide. This was not important in this work because the cermet particles that were studied were quite porous, and not limited by gas phase oxygen diffusion. However, by accounting for the size variation of the copper particles during oxidation, a better estimates of the pore size limit that results in diffusion controlled burning could be made.

A second recommendation for future work would be to apply the oxidation model to characterize reaction rates of natural ore- or alloy- based oxygen carriers. Many different oxygen carrier particles with Cu, Ni, Fe or Mn as the active species have been tested by the CLC community (for a list see review articles written by Adanez [18] or Hossain [30]). More recently, researchers have begun to evaluate the oxygen transport capability of natural ores like iliminite, an iron-based ore, or alloys of Cu, Ni, Fe and Mn in the form of perovskites, a metal oxide of the form $ABO_3$, where A atoms are typically alkaline earth metals, B atoms are transition metals and O oxygen [21, 36, 143]. The primary benefit to use a natural ore as the oxygen carrier is that natural ores are expected to be less expensive than an oxygen carrier that needs to be manufactured [18, 21]. A benefit to use a perovskite-based oxygen carrier is that the defect structure of the perovskite can be tuned to increase defect species
concentrations, which should result in enhanced oxidation and reduction rates. A model capable of evaluating the oxidation rate of ores or perovskite based oxygen carriers could improve how quickly oxygen carriers can be evaluated in the lab.

Further work is also needed to develop a more comprehensive reduction model capable of characterizing reaction rates of metal oxides with CO, H₂ and CH₄. Recall that metal oxide reduction depends on both surface and bulk defect chemistry. While a model describing CuO reduction with CO was developed during the course of this thesis and is presented in Appendix A [122], it only accounts for surface chemistry. As a result, it may not be able to capture reduction rates over a wide range of conditions. In order to improve this model, the a pathway accounting for defect chemistry should be included.
Appendix A

Surface Chemistry Reduction Model

In this Appendix, a model that is capable of characterizing the nucleation stage of reduction is presented. This model was initially proposed to describe the TGA reduction experiments presented in Chapter 4 however, it was not used since it only accounts for surface chemistry, and not defect chemistry. For the model to be applicable, a pathway for bulk reduction of CuO (i.e., defect chemistry) needs to be included.

In this surface chemistry model, reduction is described by the following 9 reaction
In these reactions, CuO(CO), Cu$_2$O(CO) and Cu(CO) are adsorbed surface species (copper sites having adsorbed CO), formed via reactions [R.1] [R.4] and [R.7] when CuO, Cu$_2$O and Cu are exposed to CO, respectively. Reactions [R.8] and [R.9] are representative of migration reactions, where loosely bound CO can diffuse on the surface between copper sites. These reactions are assumed to be reversible, as is reaction [R.6].

As a consequence of chemical reactions, the rate of change in the moles of surface species $i$ (e.g., CuO, Cu$_2$O, Cu, CuO(CO), Cu$_2$O(CO), Cu(CO)) is given by Eq. A.2.

In this expression, $A$ is the total surface area available for adsorption, $[X_i]$ is the surface molar concentration of species $i$ (in mol/m$^2$), and $\hat{R}_i''$ is the overall reaction rate of species $i$ per unit surface area (in mol/m$^2$-s). The surface molar concentrations $[X_i]$ (mole/m$^2$) are defined as a function of the species site fractions, $\theta_i$, and given by Eq. A.3. In this expression, $H_i$ is the number of copper sites that comprises species $i$ (e.g., Cu$_2$O comprises two Cu sites), $S_i$ is the total site density (the total number of copper sites per m$^2$ surface) and $N_a$ is Avogadro’s number.
\[
\frac{dN_i}{dt} = A \cdot [X_i] = [X_i] \cdot \frac{dA}{dt} + A \cdot [X_i] = A \cdot \dot{R}_{i}''
\] (A.2)

\[
[X_i] = \frac{\theta_i \cdot (S_d/N_a)}{H_i}
\] (A.3)

Equation A.2 can be divided by the total surface area and combined with Eq. A.3 to yield expressions for the rate of change of the site fractions during the course of reaction. The following rate equations result for the site fractions of the six surface species (i.e., CuO, Cu₂O, Cu, CuO(CO), Cu₂O(CO), Cu(CO)) when the overall reaction rates are written in terms of the net rates of the individual reactions:

\[
\frac{d\theta_{\text{CuO}}}{dt} = \frac{H_{\text{CuO}}}{S_d/N_a} \left( -\dot{R}_1'' - \dot{R}_3'' + \dot{R}_6'' + \dot{R}_8'' \right) - \frac{\theta_{\text{CuO}}}{S} \cdot \frac{dS}{dt} (A.4a)
\]

\[
\frac{d\theta_{\text{Cu}_2\text{O}}}{dt} = \frac{H_{\text{Cu}_2\text{O}}}{S_d/N_a} \left( \dot{R}_3'' - \dot{R}_4'' - \dot{R}_6'' + \dot{R}_9'' \right) - \frac{\theta_{\text{Cu}_2\text{O}}}{S} \cdot \frac{dS}{dt} (A.4b)
\]

\[
\frac{d\theta_{\text{Cu}}}{dt} = \frac{H_{\text{Cu}}}{S_d/N_a} \left( \dot{R}_2'' + 2\dot{R}_5'' + \dot{R}_6'' - \dot{R}_7'' - \dot{R}_8'' - \dot{R}_9'' \right) - \frac{\theta_{\text{Cu}}}{S} \cdot \frac{dS}{dt} (A.4c)
\]

\[
\frac{d\theta_{\text{CuO(CO)}}}{dt} = \frac{H_{\text{CuO(CO)}}}{S_d/N_a} \left( \dot{R}_1'' - \dot{R}_2'' - \dot{R}_3'' - \dot{R}_8'' \right) - \frac{\theta_{\text{CuO(CO)}}}{S} \cdot \frac{dS}{dt} (A.4d)
\]

\[
\frac{d\theta_{\text{Cu}_2\text{O(CO)}}}{dt} = \frac{H_{\text{Cu}_2\text{O(CO)}}}{S_d/N_a} \left( \dot{R}_4'' - \dot{R}_5'' - \dot{R}_9'' \right) - \frac{\theta_{\text{Cu}_2\text{O(CO)}}}{S} \cdot \frac{dS}{dt} (A.4e)
\]

\[
\frac{d\theta_{\text{Cu(CO)}}}{dt} = \frac{H_{\text{Cu(CO)}}}{S_d/N_a} \left( \dot{R}_7'' + \dot{R}_8'' + \dot{R}_9'' \right) - \frac{\theta_{\text{Cu(CO)}}}{S} \cdot \frac{dS}{dt} (A.4f)
\]

In the above expressions, the net rate of reaction \( j \) is given by Eq. A.5 where \( \nu_{i,j} \) and \( \nu'_{i,j} \) are the stoichiometric coefficients for species \( i \) in the forward and reverse directions, respectively, of reaction \( j \), and \( k_{f,i} \) and \( k_{r,i} \) are the forward and reverse rate coefficients, respectively, of reaction \( j \). For an irreversible reaction, \( k_{r,j} = 0 \).

\[
\dot{R}_j'' = k_{f,j} \prod_{i=1}^{i}[X_i]^\nu_{i,j} - k_{r,j} \prod_{i=1}^{i}[X_i]^\nu'_{i,j} (A.5)
\]

In this model, it was assumed that the total surface area of a copper oxide particle depended on the relative amounts of the different species constituting the particle.
mass, given by Eq. [A.6]. Here, \( m_i \) and \( S_{g,i} \) are the mass and specific surface area, respectively, of species \( i \) in the particle. The specific surface areas are assumed to be constant whereas the mass of species \( i \) is given by Eq. [A.7]

\[
S = \sum_i m_i \cdot S_{g,i} \tag{A.6}
\]

\[
m_i = [X_i] \cdot S \cdot MW_i = \frac{\theta_i \cdot S_d / N_a}{H_i} \cdot S \cdot MW_i \tag{A.7}
\]

Equations [A.6] and [A.7] can be differentiated with respect to time and combined to yield an expression for \( dS/dt \) in terms of site fractions. When employed in Eqs. [A.4a] through [A.4f] the resulting equations can be integrated simultaneously to yield the species site fractions as a function of time, for specified rate coefficients and initial conditions. Instead of integrating Eq. [A.4a] for the CuO site fraction, it is determined by difference (\( \theta_{\text{CuO}} = 1 - \theta_{\text{Cu}_2\text{O}} - \theta_{\text{Cu}} - \theta_{\text{CuO(CO)}} - \theta_{\text{Cu}_2\text{O(CO)}} - \theta_{\text{Cu(CO)}} \)) when the initial mass is composed of CuO particles. If the initial particle is composed of \( \text{Cu}_2\text{O} \), instead of integrating Eq. [A.4b] the \( \text{Cu}_2\text{O} \) site fraction is determined by difference. Having determined the site fractions, the surface concentrations and masses of each component in the particle are readily calculable.

Based on the reaction mechanism presented above, the quantity \( 1/m \cdot dm/dt \), an expression that depends on the rates of the chemical reactions and can be calculated from thermogravimetric data, is given by Eq. [A.8]. In this expression, \( S_g \) is the specific surface area of the particle calculated in the model as: \( S_g = S_{\text{total}} / m_{\text{total}} \), and the molar reaction rates (\( \hat{R}_i \)) are written in terms of rate coefficients \( k_i \) and the concentrations of the species involved in the reaction, as described above. Using a least squares procedure, \( k_i \) for each reaction can be determined by minimizing the sum of the differences in the measured and calculated values of \( 1/m \cdot dm/dt \) at each measurement time for each test.

\[
\frac{1}{m} \frac{dm}{dt}_{\text{calc}} = S_g \cdot (MW_{\text{CO}} \cdot (\hat{R}_1'' - \hat{R}_4'' - \hat{R}_7'') + MW_{\text{CO}_2} \cdot (\hat{R}_2'' + \hat{R}_3'' + \hat{R}_5'')) \tag{A.8}
\]
Parameters that describe the molar reaction rates in Eq. A.8 were extracted by fitting the thermogravimetric data presented in Chapter 4. However, this surface chemistry model was not used in this thesis because it was impossible to accurately predict the TGA data when employing the known porosity, diameter, and specific surface area of the CuO particles unless an unrealistic value of the Cu site density was used (10 to 100 times the expected site density). For this reason, the power-law approach described in Chapter 4, which includes the combined effects of surface and bulk chemistry, was employed.
Appendix B

Experimental Procedures

A number of experimental systems were used throughout the course of this thesis. A brief description of each system along with the procedure used to operate the system are given below.

B.1 Thermogravimetric Analyzer

A Thermogravimetric Analyzer (TGA) is a sensitive scale that can be used to measure weight change in time (thermogram) of a sample as a function of time in specified gaseous environments. The thermograms were used to derive the mass specific reaction rate, $1/m \cdot dm/dt$ used in this thesis.

The TGA was calibrated by performing a ramped temperature experiment with a sample of calcium oxalate ($H_2O\cdot CaC_2O_4$). The calcium oxalate was heated up in an inert environment and the weight change of the calcium oxalate as it decomposes was recorded as a function of time and temperature. As the calcium oxalate is heated from ambient to approximately 900 °C, it decomposes into $H_2O$, $CO$ and $CO_2$. $H_2O$ evolves from the sample at low temperatures, around 100 °C, $CO$ at temperatures between 400 and 500 °C, and $CO_2$ at temperatures between 500 and 700 °C. An example thermogram from this calibration procedure is presented in B.1.
B.2 Coulter Counter Procedure

A Coulter Counter, a particle sizing system based on the Coulter Principle (described elsewhere, [144]), was used to measure the size distribution of the copper, activated carbon, and wyodak coal particles studied in this thesis. The Coulter counter contains a tube with a small aperture that is immersed in an electrolyte solution [144]. In the Coulter system, particles are suspended in the electrolyte solution and are pumped through the orifice. As the particles go through the orifice, the particles displace some of the electrolyte solution and this is volume is measured by the system. The electrolyte solution used in the Coulter Counter was comprised of 20% glycerol and 80% deionized water. For every 100 mL of solution, 1 gram of high purity NaCl was added to the mixture. The glycerol, while not necessary for the Coulter system to work, was added to increase the viscosity of the electrolyte solution, helping to suspend the particles making it easier for them to go through the orifice.

B.3 Tap Density Procedure

A tap density test was used to determine the bulk density of powder samples studied in this thesis. In a tap density measurement, a mass of the powder is placed in a small graduated tube. The tube is then taped until the sample is compressed at which point the volume and mass of the sample is measured. This is repeated with several
different masses and the slope of the mass and volume data can be used to estimate the bulk density of the contents in the tube. There are voids in between the particles, and depending on the shape of the particles a packing factor can be assigned and used to extract the apparent density of the sample.

### B.4 Gas Chromatography

A Varian 3400 Gas Chromatograph (GC) was used to measure the different gas phase species present in the exhaust of the flow reactor. A detailed description of gas chromatography can be found elsewhere [145]. The gas chromatograph used in this thesis had a thermal conductivity detector and was setup with a packed column and a molecular sieve, where the carrier gas was argon. The GC was calibrated after every two to three experiments with high purity gases supplied by Praxair. In order to calibrate the GC, mixtures of known gas were supplied through the system and both the peak height and area of the signal produced from the thermal conductivity detector were correlated with gas mole fraction. The standard gases from Praxair were diluted with a known amount of argon to make mixtures at lower concentration. An example of the signal produced from the GC is given in Fig. B.2. Similar calibrations measurements were made with H$_2$ and CH$_4$.

![Signal measured by GC from calibration run. Gas mixtures used: (1) 50% CO, 50% CO$_2$, (2) 21% CO, 21% CO$_2$, (3) 14% CO, 14% CO$_2$](image)

Figure B.2: Signal measured by GC from calibration run. Gas mixtures used: (1) 50% CO, 50% CO$_2$, (2) 21% CO, 21% CO$_2$, (3) 14% CO, 14% CO$_2$
Appendix C

Matlab Codes

In this appendix is the code that was developed for the oxidation model, the cermet particle model and the system analysis.

C.1 Chemical Model

The oxidation model is run with a driver file. This script file first calls the initialization file which contains input parameters for the model then the function which describes the oxidation processes.

C.1.1 Driver Code

```matlab
% Mode description
% This is the driver file for calculations modeling the oxidation of Cu in oxygen; oxidation is described by transport of crystalline and electronic defects in copper cuprous and cupric oxide lattices

% Creation Date: 1/30/12

clear all
tic

InitFileparticleCM_nuc
```
C.1. CHEMICAL MODEL

![Chemical Model Code](image)

C.1.2 Initialization Function

This is the initiation code for the copper oxidation model.
APPENDIX C. MATLAB CODES

```
aSphere = 3; % index, indicates a spherical geometry
aCylinder = 2; % index, indicates a cylindrical geometry
aPlane = 1; % index, indicates a planer geometry

shapeFactor = aSphere; % input shape to model
solveMoles = true; % Only copper sub-lattice defects

allCu = true; % initially the particle is only Cu
% define thickness of each layer initially
% initial Cu or Cu2O...
MtVect = 0; % Initial impurity fraction
diamDist = 35; % particle diameter, micron
sysPresO2 = 0.21*101325; % oxygen partial pressure, Pa
sysTemp = 700; % temp, deg C;

Dx(1) = 0.003677*exp(-53.9*1000/(8.314*(sysTemp+273.15))); % Vac. Diff. in Cu2O \( \rightarrow \) cm\(^2\)/s
Dx(2) = 1.47e-3*exp(-84.8*1000/(8.314*(sysTemp+273.15))); % Cu. Int. Diff. in \( \rightarrow \) Cu2O \( \rightarrow \) cm\(^2\)/s
Dx(3) = 3.47e-4*exp(-32*1000/(8.314*(sysTemp+273.15))); % Vac. Diff. in CuO \( \rightarrow \) cm\(^2\)/s
Dx(4) = 1e-10; % Cu. Int. Diff. in CuO \( \rightarrow \) cm\(^2\)/s

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * * temperature / pressure Option * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
sys.presTot = 101325; %0.1e6; % Pa
try
    sys.presO2 = sysPresO2;
catch
    sys.presO2 = 0.21 * sys.presTot; % Pa
end
sys.presN2 = sys.presTot - sys.presO2;
sys.presO2_atm = sys.presO2/101325; % Atm
```

```
sys.presO2_atmG = sys.presO2/101325; % Atm % what is this defined for?

try
    sys.temp = sysTemp+273.15;
catch
    sys.temp = 700+273.15; % Kelvin
end

tRange.rampTime = 0*60; % end of ramp, seconds

tRange.endTime = 450*60; % end time of experiment, seconds

tRange.timeLG = -1; % define fast step and slow step... seconds, if -1
    ↩ no time steps...

tRange.tRangeM = tRange.endTime - tRange.timeLG; % seconds

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * * * particle characteristics * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

Ns0.Nuclei = 1e-9; % nuclei per nm^2, 25 Ang^2 = 0.25 nm^2

Ns.Nuclei = Ns0.Nuclei * exp(0.3/(Cn.kB_eV * sys.temp)); % nm^-2

Ld.Nuclei = Ns.Nuclei^-0.5/(1000); % micron, length scale, for oxygen capture
    ↩ around each Cu2O island

k.Nuclei = 5; % (micron^-2-s)^-1 proportionality constant, probability for Cu
    ↩ and O to form Cu2O... should be exponential...

K_s = 1; % [-], sticking coefficient of oxygen to the Cu2O island perimeter

% f_s = 8*10^9; % [atoms/micon-s] diffusive flux of oxygen, how should this
    ↩ compare to nc_bar/4

switch round(sys.temp-273.15)
case 500
    f_s = 6*10^-9; % [atoms/micon-s] 600 C

case 600
    f_s = 12*10^-9; % [atoms/micon-s] 600 C

case 700
    f_s = 25*10^-9; % [atoms/micon-s] 700 C

case 800
    f_s = 40*10^-9; % [atoms/micon-s] 800 C

case 900
    f_s = 75*10^-9; % [atoms/micon-s] 900 C
otherwise
    f_s = 75*10^-9; % [atoms/micon-s] 900 C
end

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * rho characteristics * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

rho.Cu0 = 6.31; % g/cm^-3 * 100^-3 cm^-3/m^-3 [=] g/m^-3

rho.Cu2O = 6.0; % g/cm^-3 * 100^-3 cm^-3/m^-3

rho.Cu = 8.94; % g/cm^-3 * 100^-3 cm^-3/m^-3
switch shapeFactor
    case aSphere
        try
            particle.diam = diamDist*10^-6*100; % cm
        catch
            particle.diam = 50e-4; % cm
        end
        particle.vol = 4/3 * pi * (particle.diam/2)^3; % cm^3
        if allCu
            particle.Sg = 6./(particle.diam * rho.Cu); % diam in cm, rho.cu in cm, cm^2/g
        else
            particle.Sg = 6./(particle.diam * rho.Cu2O); % diam in cm, rho.cu in cm, cm^2/g
        end
        surfArea = particle.Sg * particle.vol * rho.Cu; % cm^2
    case aCylinder
        particle.wireLength = 100; % cm
        particle.wireDiam = 0.025; % cm
        surfArea = 2*pi*(particle.wireDiam/2)*particle.wireLength; % cm^2
    case aPlane
        initThick = 5000e-6*100; % cm
        surfArea = 1*1; % cm^2
        initVol = surfArea*initThick; % cm^3
end

if isempty(Dx)
    Dx(1) = 1.48e-4*exp(-24*1000/(8.314*sys.temp)); % if only vacancies
    Dx(2) = 1.47e-3*exp(-84.8*1000/(8.314*sys.temp)); % if only interstitials
    Dx(3) = 3.47e-4*exp(-32*1000/(8.314*sys.temp));
    Dx(4) = 1e-10;
end

% time step selection
% tEnd = x*Large +y*Small
% small time steps at the end,
% large time steps at the beginning
if tRange.timeLG>0
    tempTimeLrgStep = 40; % duration to apply large step...
    tempLrgStep = tempTimeLrgStep/tRange.timeLG;
    tempSmlStep = round(max((tRange.endTime - tempTimeLrgStep)/
                           tRange.timeLG,0));
C.1. CHEMICAL MODEL

```
fullSteps = tempnLrgStep+tempnSmlStep;
finalTimeStep = rem(tempTimeLrgStep,tRange.timeLG)+rem(tRange.endTime-
→ tempTimeLrgStep,tRange.tRange);
if finalTimeStep ~= 0
  % timeStep is a vector of the dt
  % ttimeStepS is a vector of the cumulative time steps
  tRange.timeStep = zeros(fullSteps+2,1);
  tRange.timeStepS = zeros(fullSteps+2,1);
else
  tRange.timeStep = zeros(fullSteps+1,1);
  tRange.timeStepS = zeros(fullSteps+1,1);
end
for jj = 2:(length(tRange.timeStep))
  if jj <= tempnLrgStep+1
    tRange.timeStep(jj) = tRange.timeLG;
    tRange.timeStepS(jj) = tRange.timeLG + tRange.timeStepS(jj-1);
  elseif jj <= tempnSmlStep+tempnLrgStep+1
    tRange.timeStep(jj) = tRange.tRangeM;
    tRange.timeStepS(jj) = tRange.tRangeM + tRange.timeStepS(jj-1);
  end
end
clear tempTimeLrgStep tempnLrgStep tempnSmlStep fullSteps finalTimeStep;
else
  tRange.timeStep = [0 tRange.endTime];
  tRange.timeStepS = [0 tRange.endTime];
end

% equilibrium interface pressure....
sys.presEqu = phasePressure(sys.temp,sys.presTot);
sys.presCuCu2O_atm = sys.presEqu(1)/101325;
sys.presCu2OCuO_atm = sys.presEqu(2)/101325;

% determine number of phases....
if sys.presO2 > sys.presEqu(1) && sys.presO2 < sys.presEqu(2)
  nPhase = 2; % for only Cu & Cu20..
isCu0 = 0;
elseif sys.presO2 > sys.presEqu(2)
  isCu0 = 1;
  nPhase = 3; % for only Cu & Cu20..
else
  % pressure too low, no oxidation...
isCu0 = 0;
nPhase = 1;
```
```matlab
end

nDefect = 2; % number of defect types/ lattice; vacancy
% hole vacancy equilibrium....
% define number of variables to track / solve for in each domain.
% - vacancy number density
% - interstitial number denstiy
% - Cu / Cu2O / CuO shell thickness
nVar = nPhase + nDefect*nPhase;

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * Particle Shell Radius Distribution * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

tStep = 0:.5:20;
nStep = length(tStep);
NNuclei = @( time ) 1/ Ld.Nuclei ^2 *(1 - exp (-k.Nuclei*Ld.Nuclei ^2* time)); % number density of Cu2O nuclei , num / micron ^2
tau99p99 = -log (0.001 )/( k.Nuclei * Ld.Nuclei ^2) ; % seconds -- time to reach saturation nucleation density
AreaNuclei_t = 0;
if allCu
  if nPhase == 3
    fracCu2O = 0.5; % mass fraction of Cu2O in the nuclei layer
  else
    fracCu2O = 1; % only Cu2O formed...
  end
else
  fracCu2O = 0; % mass fraction of Cu2O in the nuclei layer
end
rhoNuclei = ( fracCu2O / rho.Cu2O + (1 - fracCu2O )/ rho.CuO )^ -1; %g/cc
MWNuclei = 1/( fracCu2O / MW.Cu2O + (1 - fracCu2O )/ MW.CuO ); % g/mole
volDensO = 1/ rhoNucleiA ; % volume occupied by one oxygen atom , micron ^3 /
rat = sqrt (1 -(1 - aspectRatio ) ^2) ;
for kk = 2: nStep
  dN = (NNuclei(tStep(2:kk))-NNuclei(tStep(1:(kk-1))))*surfArea*(10^-4)^2; % new nuclie created at time interval dt
```

Note: The code snippet provided is a MATLAB code for simulating the growth of Cu2O nuclei during oxidation processes, focusing on the nucleation and growth dynamics. The code segments include definitions of variables, calculations for density, mass fractions, and nucleation rates, among other aspects of the simulation.
C.1. CHEMICAL MODEL

Area = A_3d(fliplr(tStep(2:kk)))/(10^-4)^2; % cm^2, area of nuclei created at tStep(i) discritized time...

AreaNuclei_t(kk) =sum(dN.* Area); % total nuclei area, cm^2

aDim = sqrt(Area/(2 * pi * (1 + (1-rat^-2)/rat*atanh(rat))))*10^-4; % micron, width

bDim = aspectRatio * aDim; % micron, height

volumeTotal = 4/3*pi * aDim.^2.* bDim/(10^-4)^3; % cm^3 per nuclei

volumeNuclei.tot(kk) = sum(dN.* volumeTotal); % cm^3

end

indexFullCoverage = find(AreaNuclei_t>surfArea,1);

if isempty(indexFullCoverage)
    figure(10)
    plot(tStep,NNuclei(tStep))
    figure(11)
    plot(tStep,AreaNuclei_t)
    hold on
    plot(tStep,surfArea*ones(length(tStep),1), 'r')

    error('nucleation rate not fast enough')
else
    % disp('Time for Nuclie Coverage is...')
    % disp([num2str(tStep(indexFullCoverage)) ' minutes'])
end

AreaNuclei_t(indexFullCoverage) = surfArea;
AreaNuclei_t = AreaNuclei_t(1:indexFullCoverage);

volumeNuclei.tot = volumeNuclei.tot(1:indexFullCoverage);
weightNuclei.tot = volumeNuclei.tot(1:indexFullCoverage) * rhoNuclei/10; % grams

timeNuclei = tStep(1:indexFullCoverage)'; % seconds..

indexFullCoverage = 1;
volumeNuclei.tot = surfArea * (10e-9*100); %cm^-3
weightNuclei.tot = volumeNuclei.tot * rhoNuclei; % grams

timeNuclei = 0; % seconds...

% once fully covered, replace spherids with equilivalent shell, constant thickness...
weightNuclei.Cu2O = weightNuclei.tot*fracCu2O;
weightNuclei.CuO = weightNuclei.tot*(1-fracCu2O);

volumeNuclei.Cu2O = weightNuclei.Cu2O/rho.Cu2O; % cc

switch shapeFactor
```matlab
% are we starting with Cu or Cu2O
if allCu
    molesSys = 4/3*pi*(particle.diam/2)^3 * rho.Cu/MW.Cu; % moles Cu per particle
    molesCore = molesSys - weightNuclei.Cu0(end)/MW.CuO - weightNuclei.Cu2O(end)/MW.Cu2O*2;
    particle.Cu.thick = (3/(4*pi)*(molesCore/MW.Cu2O/rho.Cu2O))^(1/3); % cm
    particle.Cu2O.thick = ((3/(4*pi)*(weightNuclei.Cu2O(end)/rho.Cu)) - particle.Cu.thick^3)^(1/3) - particle.Cu.thick; % cm
else
    molesSys = 4/3*pi*(particle.diam/2)^3 * rho.Cu2O/MW.Cu2O; % moles Cu per particle
    molesCore = molesSys*2 - weightNuclei.Cu0(end)/MW.CuO;
    particleNucliMass.Cu = 0;
    particle.Cu.thick = 0; % cm
    particle.Cu2O.thick = ((3/(4*pi)*(molesCore/2*MW.Cu2O/rho.Cu2O)))^(1/3); % cm
end

% done to prevent double counting, start of code and end of nucleation are the same weight.
particle.volume.Cu = 4/3 * pi*(particle.Cu.thick^3 - 0^3); % cm^3
```
C.1. CHEMICAL MODEL

\[
\text{particle.volume.CuO} = \frac{4}{3} \pi \left( (\text{particle.CuO.thick} + \text{particle.Cu2O.thick} + \text{particle.Cu.thick})^3 - (\text{particle.Cu2O.thick} + \text{particle.Cu.thick})^3 \right); \ \text{cm}^3
\]

\[
\text{case aCylinder}
\]

\[
\text{if allCu}
\]

\[
\text{molesSys} = \text{particle.wireLength} \times \pi \left( \frac{\text{particle.wireDiam}}{2} \right)^2 \times \rho_{\text{Cu}}; \ \% \text{moles Cu per wire}
\]

\[
\text{molesCore} = \text{molesSys} - \frac{\text{weightNuclei.CuO}(\text{end})}{\text{MW.CuO}} - \frac{\text{weightNuclei.Cu2O}(\text{end})}{\text{MW.Cu2O}} \times 2;
\]

\[
\text{particleNucliMass.Cu} = \frac{\text{molesSys} \times \text{MW.Cu} - \text{weightNuclei.CuO}(\text{end})}{\text{MW.CuO}} - \frac{\text{weightNuclei.Cu2O}(\text{end})}{\text{MW.Cu2O}} \times \text{MW.Cu} - \text{weightNuclei.Cu2O} \times 2;
\]

\[
\text{particleNucliMass.Cu2O} = \text{weightNuclei.Cu2O};
\]

\[
\text{particleNucliMass.CuO} = \text{weightNuclei.CuO};
\]

\[
\]

\[
\text{particle.Cu.thick} = \left( \frac{\text{particleNucliMass.Cu}(\text{end})}{\rho_{\text{Cu}} \times \text{particle.wireLength} \times \pi} \right)^{0.5}; \ \% \text{cm, wire radius}
\]

\[
\text{particle.Cu2O.thick} = \left( \frac{\text{molesCore} \times \text{MW.Cu2O}}{\rho_{\text{Cu2O}} \times \text{particle.wireLength} \times \pi} + \text{particle.Cu.thick}^2 \right)^{0.5} - \text{particle.Cu.thick}; \ \% \text{cm}
\]

\[
\text{particle.CuO.thick} = \left( \frac{\text{weightNuclei.CuO}(\text{end})}{\rho_{\text{CuO}} \times \text{particle.wireLength} + \text{particle.Cu2O.thick}^2} \right)^{0.5} - \text{particle.Cu2O.thick}; \ \% \text{cm}
\]

\[
\text{else}
\]

\[
\text{Cu2O} + \frac{1}{4} \text{O2} \rightarrow 2\text{CuO}
\]

\[
\text{molesSys} = \text{particle.wireLength} \times \pi \left( \frac{\text{particle.wireDiam}}{2} \right)^2 \times \rho_{\text{Cu2O}}; \ \% \text{moles Cu2O per wire}
\]

\[
\text{molesCore} = \text{molesSys} - \frac{\text{weightNuclei.CuO}(\text{end})}{\text{MW.CuO}} - \frac{\text{weightNuclei.Cu2O}(\text{end})}{\text{MW.Cu2O}} \times 2;
\]

\[
\text{particleNucliMass.Cu} = 0;
\]

\[
\text{particleNucliMass.Cu2O} = \frac{\text{molesSys} \times \text{MW.Cu2O} - \text{weightNuclei.CuO}(\text{end})}{\text{MW.CuO}} - \frac{\text{weightNuclei.Cu2O}(\text{end})}{\text{MW.Cu2O}} \times \text{MW.Cu2O} - \text{weightNuclei.CuO} \times 2;
\]

\[
\text{particleNucliMass.CuO} = \text{weightNuclei.CuO};
\]

\[
\]

\[
\text{particle.Cu.thick} = 0; \ \% \text{cm}
\]

\[
\text{particle.Cu2O.thick} = \left( \frac{\text{molesCore} \times \text{MW.Cu2O}}{\rho_{\text{Cu2O}} \times \text{particle.wireLength} \times \pi} \right)^{0.5}; \ \% \text{cm}
\]

\[
\text{particle.CuO.thick} = \left( \frac{\text{weightNuclei.CuO}(\text{end})}{\rho_{\text{CuO}} \times \text{particle.wireLength} + \text{particle.Cu.thick} \times 2} \right)^{0.5} - \text{particle.Cu.thick}; \ \% \text{cm}
\]

\[
\text{end}
\]
337  particle.volume.Cu = pi*particle.Cu.thick^2 * particle.wireLength;  % cm^3
338  particle.volume.Cu2O = particle.wireLength*pi*(particle.Cu2O.thick +  % cm^3
339  particle.Cu.thick)^2 - particle.Cu.thick^2);  % cm^3
340  particle.volume.CuO = particle.wireLength*pi*((particle.Cu0.thick +  % cm^3
341  particle.Cu2O.thick + particle.Cu.thick)^2 - ...  
342  (particle.Cu2O.thick + particle.Cu.thick)^2));  % cm^3
343  % cm^3
344  case aPlane  % if a plane...
345  if allCu  % moles Cu per plate
346  molesSys = initVol * rho.Cu/MW.Cu;  % moles Cu per plate
347  molesCore = molesSys - weightNuclei.Cu0(end)/MW.Cu -  % moles Cu per plate
348  particleNucliMass.Cu = molesSys*MW.Cu - weightNuclei.Cu0/MW.Cu*  % moles Cu per plate
349  MW.Cu - weightNuclei.Cu20/MW.Cu20*2;
353  + particleNucliMass.CuO;
354  particle.Cu.thick = particleNucliMass.Cu(end)/rho.Cu/surfArea;  % cm  % moles Cu per plate
355  particle.Cu2O.thick = particleNucliMass.Cu20(end)/rho.Cu20/surfArea;  % cm
356  particle.CuO.thick = particleNucliMass.CuO(end)/rho.Cu0/surfArea;  % cm
357  % moles Cu per plate
358  else  % moles Cu2O per plate
359  molesSys = initVol * rho.Cu20/MW.Cu20;  % moles Cu2O per plate
360  particleNucliMass.Cu = weightNuclei.Cu0*0;
361  particleNucliMass.Cu20 = molesSys*MW.Cu20 - weightNuclei.Cu0/MW.Cu0*  % moles Cu2O per plate
362  MW.Cu20/2;
365  + particleNucliMass.CuO;
366  particle.Cu.thick = 0;  % cm, thickness of Cu layer
367  particle.Cu20.thick = particleNucliMass.Cu20(end)/rho.Cu20/surfArea;  % cm
368  particle.CuO.thick = particleNucliMass.CuO(end)/rho.Cu0/surfArea;  % cm
369  end
370  particle.volume.Cu = surfArea*particle.Cu.thick;  % cm^3
371  particle.volume.Cu20 = surfArea*particle.Cu20.thick;  % cm^3
372  particle.volume.CuO = surfArea*particle.Cu0.thick;  % cm^3
C.1. CHEMICAL MODEL

end

particle.mass.Cu  = rho.Cu * particle.volume.Cu;  % grams
particle.mass.Cu2O = rho.Cu2O * particle.volume.Cu2O;  % grams
particle.mass.CuO  = rho.CuO  * particle.volume.CuO;  % grams

% Other Copper Properties
% 
% [regSite Keq]= defectEquilibriumFunc(sys);
% define impurity concentrations
impur.Cu2O.pos = MtVect;
impur.Cu2O.neg = 0;
impur.CuO.pos  = MtVect;
impur.CuO.neg  = 0;

deltaGCuO.vac = (250*1000 - ( -39*sys.temp )); % J/mol
deltaGCuO.int = (260*1000 - (75*sys.temp )); % J/mol

[regSite Keq]= defectEquilibriumFuncDH(sus, deltaGCuO, impur); % temp, pres, presO2)

% Define heat and mass transport properties
% 
% particle.SRF   = 1;  % surface roughness factor
% particle.tortuosity = 3;  % particle tortuosity
% particle.ShNum  = 2;  % sherwood number
% particle.Nu    = 2;  % h*D/k

particle.gbWidth = 10e-9;  % cm
particle.gbNum  = 10;  % boundaries per m^-2
% 
% TestName = ['TGAModel_' num2str(sys.temp) 'degC' num2str(cputime)];
% 
% if solveMoles
% 
% if isCuO

APPENDIX C. MATLAB CODES

```matlab
        (particle.mass.Cu/WM.Cu) / (particle.mass.Cu/WM.Cu + 2*particle.mass.Cu2O / MW.Cu2O + particle.mass.CuO / MW.CuO); % moles of each shell per total Cu moles in the system...
else
    y_init = [particle.mass.Cu / MW.Cu, particle.mass.Cu2O / MW.Cu2O] / (particle.mass.Cu/WM.Cu + 2*particle.mass.Cu2O / MW.Cu2O); % moles of each shell per total Cu moles in the system...
end

% DENSITY CONSTANT, O and Cu lattice...
if isCuO
    % moles of each shell per total Cu moles in the system...
    y_init = [particle.mass.Cu / MW.Cu, 0, particle.mass.Cu2O / MW.Cu2O * [2 1], [1]
else
    y_init = [particle.mass.Cu / MW.Cu, 0, particle.mass.Cu2O / MW.Cu2O * [2 1]] / (particle.mass.Cu/WM.Cu + 2*particle.mass.Cu2O / MW.Cu2O); % moles of each shell per total Cu moles in the system...
end

% input flux for a given thickness...
% recall that for a plane, cylinder or sphere J, J x r, and J x r^2 will be a constant as the layer grows

nVols = 400;
ri = 5*10^-4; % cm, inner radius
ro = 20*10^-4; % cm, outer radius
deltaStep = (ro-ri)/(nVols-2);
rCenter = [ri,((deltaStep/2):deltaStep:deltaStep*(nVols-2))+ri,ro]; % not incluede end points ri or ro
```
C.1. CHEMICAL MODEL

if allCu

% what is c_h, c_e, c_Vac and c_int
% are self-diffusion coeffs for neutral species
D_eff.Cu2O.vacCuX = Dx(1); % cm^2/s -- vac diffusion coef for vacCuX
D_eff.Cu2O.intCuX = Dx(2); % 100*DeffFuncDefect('Cu2O','intCuX'); % cm^2/s

% x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x
switch shapeFactor
  case aPlane
    C1 = D_eff.Cu2O.vacCuX*(regSite.vacCuX.Cu2OCuO -
      \leftarrow regSite.vacCuX.CuCu2O)/(ri - ro);
    C2 = regSite.vacCuX.CuCu2O + C1*ri/D_eff.Cu2O.vacCuX;
    conc.Cu2O.vacCuX = -C1/D_eff.Cu2O.vacCuX * rCenter + C2;

  case aCylinder
    C1 = D_eff.Cu2O.vacCuX*(regSite.vacCuX.Cu2OCuO -
      \leftarrow regSite.vacCuX.CuCu2O)/log(ri/ro);
    C2 = regSite.vacCuX.CuCu2O + C1*log(ri)/D_eff.Cu2O.vacCuX;
    conc.Cu2O.vacCuX = -C1/D_eff.Cu2O.vacCuX * log(rCenter) + C2;

  case aSphere
    C1 = D_eff.Cu2O.vacCuX*(regSite.vacCuX.Cu2OCuO -
      \leftarrow regSite.vacCuX.CuCu2O)/(1/ri - 1/ro);
    C2 = regSite.vacCuX.CuCu2O + C1/(ri*D_eff.Cu2O.vacCuX);
    conc.Cu2O.vacCuX = -C1/(D_eff.Cu2O.vacCuX * rCenter) + C2;

% x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x
% NEUTRAL species in Cu2O
% x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x
switch shapeFactor
  case aPlane
    C1 = D_eff.Cu2O.intCuX*(regSite.intCuX.Cu2OCuO -
      \leftarrow regSite.intCuX.CuCu2O)/(ri - ro);
    C2 = regSite.intCuX.CuCu2O + C1*ri/D_eff.Cu2O.intCuX +
      \leftarrow regSite.intCuX.Cu2OCuO; % 100*DeffFuncDefect('Cu2O','intCuX'); % cm^2/s
    C2 = regSite.intCuX.CuCu2O + C1*ri/D_eff.Cu2O.intCuX;
    conc.Cu2O.intCuX = -C1/D_eff.Cu2O.intCuX * rCenter + C2;
  case aCylinder
    C1 = D_eff.Cu2O.intCuX*(regSite.intCuX.Cu2OCuO -
      \leftarrow regSite.intCuX.CuCu2O)/log(ri/ro);
    C2 = regSite.intCuX.CuCu2O + C1*log(ri)/D_eff.Cu2O.intCuX;
    conc.Cu2O.intCuX = -C1/D_eff.Cu2O.intCuX * log(rCenter) + C2;
  case aSphere
    C1 = D_eff.Cu2O.intCuX*(regSite.intCuX.Cu2OCuO -
      \leftarrow regSite.intCuX.CuCu2O)/(1/ri - 1/ro);
    C2 = regSite.intCuX.CuCu2O + C1/(ri*D_eff.Cu2O.intCuX);
    conc.Cu2O.vacCuX = -C1/(D_eff.Cu2O.vacCuX * rCenter) + C2;
% C1 = \log \left( \frac{\text{regSite.vacCuX.CuCu2O}}{\text{regSite.vacCuX.Cu2OCuO}} \right) \times \frac{\text{D}_{\text{eff}}.\text{Cu2O}.\text{vacCuX}}{1/\text{ri} - 1/\text{ro}};
% C2 = \frac{\text{regSite.vacCuX.Cu2OCuO}}{\text{exp}( -C1 / \left( \text{D}_{\text{eff}}.\text{Cu2O}.\text{vacCuX} \times \text{ro} \right) )};
% \text{conc.Cu2O.vacCuX} = C2 \times \exp(C1 / (\text{rCenter} \times \text{D}_{\text{eff}}.\text{Cu2O}.\text{vacCuX}));

C1 = \text{D}_{\text{eff}}.\text{Cu20}.\text{intCuX} \times (\text{regSite.intCuX.Cu2OCuO} - \text{regSite.intCuX.CuCu2O}) / (1/\text{ri} - 1/\text{ro});
C2 = \frac{\text{regSite.intCuX.CuCu2O}}{\text{ri} \times \text{D}_{\text{eff}}.\text{Cu20}.\text{intCuX}} + C1;
\text{conc.Cu20.intCuX} = \frac{\text{C1}}{\text{D}_{\text{eff}}.\text{Cu20}.\text{intCuX} \times \text{rCenter}} + C2;

end

if \text{sum(\text{isnan(\text{conc.Cu20.intCuX})})} > 0
\text{conc.Cu20.intCuX} = \text{zeros(nVols,1)};
end

% CHARGED SPECIES

y_{\text{Init Cu2O}}((1:nVols)) = \frac{(\text{regSite.h.Cu2OCuO} - \text{regSite.h.CuCu2O}) / (\text{ro} - \text{ri}) \times (\text{rCenter} - \text{ri}) {+ \text{regSite.h.CuCu2O}}}{\text{regSite.h.Cu2OCuO} - \text{regSite.h.CuCu2O}}; \ % holes
y_{\text{Init Cu20}}(nVols+1:(1:nVols)) = \frac{(\text{regSite.vacCuM.Cu2OCuO} - \text{regSite.vacCuM.CuCu2O}) / (\text{ro} - \text{ri}) \times (\text{rCenter} - \text{ri}) {+ \text{regSite.vacCuM.CuCu2O}}}{\text{regSite.vacCuM.Cu2OCuO} - \text{regSite.vacCuM.CuCu2O}}; \ % vacs
y_{\text{Init Cu20}}(nVols+2:(1:nVols)) = \frac{(\text{regSite.intCuP.Cu2OCuO} - \text{regSite.intCuP.CuCu2O}) / (\text{ro} - \text{ri}) \times (\text{rCenter} - \text{ri}) {+ \text{regSite.intCuP.CuCu2O}}}{\text{regSite.intCuP.Cu2OCuO} - \text{regSite.intCuP.CuCu2O}}; \ % interstials

% it shouldnt take this long to converge... right?
timeStep = [0 1e^{-5} 1e^{-4} 1e^{-3} 1e^{-2} 1e^{-1} 1 10 100 1000 2000]; \% seconds..
relTolVal = 1e^{-13};
absTolVal = 1e^{-6};
options = \text{odeset}('\text{RelTol}',\text{relTolVal},'\text{AbsTol}',\text{absTolVal},'\text{jpattern}',\text{JpatDiff}({\rightarrow y_{\text{Init Cu20}}});'\%',\text{NonNegative}',1:nVols+3);
tic
[\text{time}, \text{outPut}] = \text{ode15s}(@\text{defCu2OProfile_PE2_NBL},\text{timeStep},y_{\text{Init Cu20}},\text{options},\rightarrow \text{sys},\text{Dx}(1:2),\text{regSite},\text{Keq},\text{rCenter},\text{nVols},\text{impur}); \% same as \rightarrow \text{Cu20/CuO Calc}.
toc
for ii = 1:length(time)
C.1. CHEMICAL MODEL

Conc. Cu2O.h (ii,:) = outPut(ii,1:nVols)*(regSite.h.Cu2OCuO - regSite.h.CuCu2O) + regSite.h.CuCu2O; % moles in Cu2O

Conc. Cu2O.vacCuM (ii,:) = outPut(ii,1:nVols)*((regSite.vacCuM.Cu2OCuO - regSite.vacCuM.CuCu2O) + regSite.vacCuM.CuCu2O); % moles in Cu2O

Conc. Cu2O.intCuP (ii,:) = outPut(ii,2:nVols)*((regSite.intCuP.Cu2OCuO - regSite.intCuP.CuCu2O) + regSite.intCuP.CuCu2O); % moles in Cu2O


end

clear outPut

LastCurve = length(time);

% determine defect diffusivities Cu2O

D_eff.Cu2O.vacCuM = Dx(1)./conc.Cu2O.vacCuM(lastCurve,:); % cm^2/s
D_eff.Cu2O.vacCuM = Dx(1); % cm^2/s
D_eff.Cu2O.intCuP = Dx(2); % cm^2/s
D_eff.Cu2O.h = DeffFuncDefect('Cu2O','hole'); % cm^2/s
D_eff.Cu2O.e = DeffFuncDefect('Cu2O','elec'); % cm^2/s

% determine d[C]/dr ... and calculate c's

dVCuM_r = dxCalc(conc.Cu2O.vacCuM(lastCurve,:),rCenter)*rho.Cu2O/MW.Cu2O;
dVCuX_r = dxCalc(conc.Cu2O.vacCuX,rCenter)*rho.Cu2O/MW.Cu2O;
dintCuX_r = dxCalc(conc.Cu2O.intCuX,rCenter)*rho.Cu2O/MW.Cu2O;
dintCuP_r = dxCalc(conc.Cu2O.intCuP(lastCurve,:),rCenter)*rho.Cu2O/MW.Cu2O;
dh = dxCalc(conc.Cu2O.h(lastCurve,:),rCenter)*rho.Cu2O/MW.Cu2O;
de = dxCalc(conc.Cu2O.e(lastCurve,:),rCenter)*rho.Cu2O/MW.Cu2O;

B.Cu2O.vacCuM = D_eff.Cu2O.vacCuM. * conc.Cu2O.vacCuM(lastCurve,:)*rho.Cu20/MW.Cu20/(Cn.kB*sys.temp*10^7); %
B.Cu20.intCuP = D_eff.Cu20.intCuP. * conc.Cu20.intCuP(lastCurve,:)*rho.Cu20/MW.Cu20/(Cn.kB*sys.temp*10^7); %
B.Cu20.h = D_eff.Cu20.h. * conc.Cu20.h(lastCurve,:)*rho.Cu20/MW.Cu20/(Cn.kB*sys.temp*10^7); %
B.Cu20.e = D_eff.Cu20.e. * conc.Cu20.e(lastCurve,:)*rho.Cu20/MW.Cu20/(Cn.kB*sys.temp*10^7); %
B.Cu20.total = B.Cu20.vacCuM+B.Cu20.h+B.Cu20.e+B.Cu20.intCuP;

dphidr = -(D_eff.Cu20.vacCuM.*dVCuM_r + D_eff.Cu20.e.*de - D_eff.Cu20.h.*dh - D_eff.Cu20.intCuP.*dintCuP_r)./(Cn.e*Cn.Na.*B.Cu20.total);
JStruct.Cu20.h = (-D_eff.Cu20.h.*dh + B.Cu20.h.*Cn.e*Cn.Na.*dphidr).*rCenter.(shapeFactor-1);
JStruct.Cu20.e = (-D_eff.Cu20.e.*de - B.Cu20.e.*Cn.e*Cn.Na.*dphidr).*rCenter.(shapeFactor-1);
JStruct.Cu20.vacCuM = (-D_eff.Cu20.vacCuM.*dVCuM_r - B.Cu20.vacCuM.*Cn.e*Cn.Na.*dphidr).*rCenter.(shapeFactor-1);
JStruct.Cu20.intCuP = (-D_eff.Cu20.intCuP.*dintCuP_r + B.Cu20.intCuP.*Cn.e*Cn.Na.*dphidr).*rCenter.(shapeFactor-1);
JStruct.Cu20.vacCuX = -D_eff.Cu20.vacCuX.*dVCuX_r.*rCenter.(shapeFactor-1);
JStruct.Cu20.intCuX = -D_eff.Cu20.intCuX.*dintCuX_r.*rCenter.(shapeFactor-1);
JStruct.Cu20.vacOPP = 0;

switch shapeFactor
case aPlane
    dcdr.Cu20.h = dh(1)*(ro-ri);
dcdr.Cu20.e = de(1)*(ro-ri);
dcdr.Cu20.vacCuM = dVCuM_r(1)*(ro-ri);
dcdr.Cu20.vacCuX = dVCuX_r(1)*(ro-ri);
dcdr.Cu20.intCuP = dintCuP_r(1)*(ro-ri);
dcdr.Cu20.intCuX = dintCuX_r(1)*(ro-ri);
dcdr.Cu20.vacOPP = 0;
end

case aCylinder
    dcdr.Cu20.h = dh(1)*((log(ro/ri)*ri);
dcdr.Cu20.e = de(1)*((log(ro/ri)*ri);
dcdr.Cu20.vacCuM = dVCuM_r(1)*((log(ro/ri)*ri);
dcdr.Cu20.vacCuX = dVCuX_r(1)*((log(ro/ri)*ri);
dcdr.Cu20.intCuP = dintCuP_r(1)*((log(ro/ri)*ri);
dcdr.Cu20.intCuX = dintCuX_r(1)*((log(ro/ri)*ri);
dcdr.Cu20.vacOPP = 0;

case aSphere
    dcdr.Cu20.h = dh(1)*(((1/ri - 1/ro)*ri^2);
dcdr.Cu20.e = de(1)*(((1/ri - 1/ro)*ri^2);
dcdr.Cu20.vacCuM = dVCuM_r(1)*(((1/ri - 1/ro)*ri^2);
dcdr.Cu20.vacCuX = dVCuX_r(1)*(((1/ri - 1/ro)*ri^2);
dcdr.Cu20.intCuP = dintCuP_r(1)*(((1/ri - 1/ro)*ri^2);
dcdr.Cu20.intCuX = dintCuX_r(1)*(((1/ri - 1/ro)*ri^2);
dcdr.Cu20.vacOPP = 0;
end

else
    dcdr.Cu20.h = 0;
dcdr.Cu20.e = 0;
dcdr.Cu20.vacCuM = 0;
dcdr.Cu20.vacCuX = 0;
dcdr.Cu20.intCuP = 0;
C.1. CHEMICAL MODEL

dcdr.Cu20.intCuX = 0;
dcdr.Cu20.vacOPP = 0;
end

switch shapeFactor
  case aPlane
    xi.Cu2O = (rCenter - ri)/(ro - ri);
  case aCylinder
    xi.Cu2O = log(rCenter/ri)/log(ro/ri);
  case aSphere
    xi.Cu2O = (1/ri - 1./rCenter)/(1/ri - 1/ro);
end

% figure; hold on
% plot(rCenter, JStruct.Cu20.h.*rCenter.^2, 'c', 'DisplayName', 'Holes')
% plot(rCenter, JStruct.Cu20.e.*rCenter.^2, 'b', 'DisplayName', 'Electrons')
% plot(rCenter, JStruct.Cu20.vacCuM.*rCenter.^2, 'g', 'DisplayName', 'VacsM')
% plot(rCenter, JStruct.Cu20.vacCuX.*rCenter.^2, 'r', 'DisplayName', 'VacsX')
% plot(rCenter, JStruct.Cu20.intCuP.*rCenter.^2, 'r', 'DisplayName', 'IntP')
% plot(rCenter, JStruct.Cu20.intCuX.*rCenter.^2, 'r', 'DisplayName', 'IntX')

% figure(3)
% hold on
% plot(time, (conc.Cu20.h(:,2) - regSite.h.Cu2OCuO)/(regSite.h.Cu2OCuO - regSite.h.CuCu2O), '-c', 'DisplayName', 'ri-h')
% plot(time, (conc.Cu20.e(:,2) - regSite.e.Cu2OCuO)/(regSite.e.Cu2OCuO - regSite.e.CuCu2O), '-b', 'DisplayName', 'ri-e')
% plot(time, (conc.Cu20.vacCuM(:,2) - regSite.vacCuM.Cu2OCuO)/(regSite.vacCuM.Cu2OCuO - regSite.vacCuM.CuCu2O), '-g', 'DisplayName', 'ri-V')
% plot(time, (conc.Cu20.intCuP(:,2) - regSite.intCuP.Cu2OCuO)/(regSite.intCuP.Cu2OCuO - regSite.intCuP.CuCu2O), '-r', 'DisplayName', 'ri-i')

% legend toggle
% plot(time, (conc.Cu20.h(:,nVols-1) - regSite.h.Cu2OCuO)/(regSite.h.Cu2OCuO - regSite.h.CuCu2O), '-c', 'DisplayName', 'ro-h')
% plot(time, (conc.Cu20.e(:,nVols-1) - regSite.e.Cu2OCuO)/(regSite.e.Cu2OCuO - regSite.e.CuCu2O), '-b', 'DisplayName', 'ro-e')
% plot(time, (conc.Cu20.vacCuM(:,nVols-1) - regSite.vacCuM.Cu2OCuO)/(regSite.vacCuM.Cu2OCuO - regSite.vacCuM.CuCu2O), '-g', 'DisplayName', 'ro-V')
% plot(time, (conc.Cu20.intCuP(:,nVols-1) - regSite.intCuP.Cu2OCuO)/(regSite.intCuP.Cu2OCuO - regSite.intCuP.CuCu2O), '-r', 'DisplayName', 'ro-i')
% ylabel('defectConcs, [-])
% xlabel('time, [sec]')
if isCuO
  nVols = 200;
  deltaStep = (ro-ri)/(nVols-2);
  rCenter = [ri ,((deltaStep/2):deltaStep*(nVols-2))+ ri ,ro]'; % not
  include end points ri or ro
  hVect = (regSite.h.CuOO2 - regSite.h.CuOCu2O)/(ro - ri)*(rCenter-ri)+
  regSite.h.CuOCu2O; % interstitials
  iVect = ((regSite.intCuPP.CuOO2 - regSite.intCuPP.CuOCu2O)/(ro - ri)*(rCenter
  -ri)+regSite.intCuPP.CuOCu2O); % interstitials
  eVect = Keq.CuO.eh. / hVect;
  vVect = ( hVect + 2*iVect - eVect ) /2;
  y_Init_CuO (1: nVols ) = ( hVect - regSite.h.CuOCu2O )/( regSite.h.CuOO2 -
  regSite.h.CuOCu2O) ; % holes
  y_Init_CuO (nVols+(1: nVols )) = ( vVect - regSite.vacCuMM.CuOCu2O )/(
  regSite.vacCuMM.CuOO2 - regSite.vacCuMM.CuOCu2O); % vacs
  y_Init_CuO (nVols*2+(1: nVols )) = ( iVect - regSite.intCuPP.CuOCu2O )/(
  regSite.intCuPP.CuOO2 - regSite.intCuPP.CuOCu2O); % interstitials
  timeStep = [0 1e-9 1e-8 1e-7 1e-6 1e-5 1e-4 1e-3 1e-2 1e-1 1e-0 1e-1 1e-0 1e-2 1e-3 1e-4 1e-5 1e-6 1e-7 1e-8 1e-9 0]; % seconds, 1e
  relTolVal = 1e-13;
  absTolVal = 1e-5;
  options = odeset ('RelTol',relTolVal,'AbsTol',absTolVal); % , 'jpattern',
  JpatDiffCuO(y_Init_CuO));%'NonNegative',1:nVols*3);
C.1. CHEMICAL MODEL

\[
\text{conc.CuO.h}(i,:) = \text{outPut}(i,(1:nVols)) \times (\text{regSite.h.CuO02} - \text{regSite.h.CuO0Cu20}) + \text{regSite.h.CuO0Cu20}; \quad \% \text{ moles in Cu2O}
\]

\[
\text{conc.CuO.vacCuMM}(i,:) = \text{outPut}(i,nVols+(1:nVols)) \times (\text{regSite.vacCuMM.CuO02} - \text{regSite.vacCuMM.CuOCu20}) + \text{regSite.vacCuMM.CuOCu20}; \quad \% \text{ moles in Cu2O}
\]

\[
\text{conc.CuO.intCuPP}(i,:) = \text{outPut}(i,nVols*2+(1:nVols)) \times (\text{regSite.intCuPP.CuO02} - \text{regSite.intCuPP.CuOCu20}) + \text{regSite.intCuPP.CuOCu20}; \quad \% \text{ moles in Cu2O}
\]

\[
\text{conc.CuO.e}(i,:) = \text{conc.CuO.h}(i,:) - 2 \times \text{conc.CuO.vacCuMM}(i,:) + 2 \times \text{conc.CuO.intCuPP}(i,:) + (\text{impur.CuO.pos} - \text{impur.CuO.neg});
\]

end

clear outPut

switch shapeFactor
  case aPlane
    \[x_i.CuO = (rCenter - ri)/(ro - ri)\]
  case aCylinder
    \[x_i.CuO = log(rCenter/ri)/log(ro/ri)\]
  case aSphere
    \[x_i.CuO = (1/ri - 1/rCenter)/(1/ri - 1/ro)\]
end

nEndTime = length(time);

dVCuMM_r = dxCalc(conc.CuO.vacCuMM(nEndTime,:),rCenter)*rho.CuO/MW.CuO;
dintCuPP_r = dxCalc(conc.CuO.intCuPP(nEndTime,:),rCenter)*rho.CuO/MW.CuO;
dh = dxCalc(conc.CuO.h(nEndTime,:),rCenter)*rho.CuO/MW.CuO;
de = dxCalc(conc.CuO.e(nEndTime,:),rCenter)*rho.CuO/MW.CuO;

D_eff.vacCuMM = Dx(3); \% cm^2/s
D_eff.intCuPP = Dx(4); \% cm^2/s
DRatio = 1; \% hole to electron diffusivity...
D_eff.h = DeffFuncDefect('CuO', 'hole').*(conc.Cu0.h.(nEndTime,:)).*(rho.CuO/ MW.CuO)*DRatio*(conc.Cu0.e.(nEndTime,:)).*(rho.CuO/MW.CuO)); \% cm^-2/s
D_eff.e = D_eff.h.*DRatio; \% cm^-2/s

B.vacCuMM = D_eff.vacCuMM.*conc.Cu0.vacCuMM(nEndTime,:).*rho.Cu0/MW.Cu0/( Cn.kB*sys.temp*10^7); \%
B.intCuPP = D_eff.intCuPP.*conc.Cu0.intCuPP(nEndTime,:).*rho.Cu0/MW.Cu0/( Cn.kB*sys.temp*10^7); \%
B.h = D_eff.h.*conc.Cu0.h(nEndTime,:).*rho.Cu0/MW.Cu0/(Cn.kB*sys.temp *10^7); \%
B.e = D_eff.e.*conc.Cu0.e(nEndTime,:).*rho.Cu0/MW.Cu0/(Cn.kB*sys.temp *10^7); \%
B.total = 4*B.vacCuMM + B.h + B.e + 4*B.intCuPP;
dphidr = -(2*D_eff.vacCuMM.*dVCuMM_r + D_eff.e.*de - D_eff.h.*dh - 2* 
D_eff.intCuPP.*dintCuPP_r)./(Cn.e*Cn.Na.*B.total);

JStruct.CuO.h = (-D_eff.h.*dh + B.h.*Cn.e*Cn.Na.*dphidr').*rCenter.^(shapeFactor-1);
JStruct.CuO.e = (-D_eff.e.*de - B.e.*Cn.e*Cn.Na.*dphidr').*rCenter.^(shapeFactor-1);
JStruct.CuO.vacCuMM = (-D_eff.vacCuMM.*dVCuMM_r - 2*B.vacCuMM.*Cn.e*Cn.Na.* 
- dphidr').*rCenter.^(shapeFactor-1);
JStruct.CuO.intCuPP = (-D_eff.intCuPP.*dintCuPP_r + 2*B.intCuPP.*Cn.e*Cn.Na.* 
- dphidr').*rCenter.^(shapeFactor-1);
JStruct.CuO.vacOPP = 0;

switch shapeFactor
  case aPlane
    dcdr.CuO.h = dh(1)*(ro-ri);
    dcdr.CuO.e = de(1)*(ro-ri);
    dcdr.CuO.vacCuMM = dVCuMM_r(1)*(ro-ri);
    dcdr.CuO.intCuPP = dintCuPP_r(1)*(ro-ri);
    dcdr.CuO.vacOPP = 0;
  case aCylinder
    dcdr.CuO.h = dh(1)*(log(ro/ri)*ri);
    dcdr.CuO.e = de(1)*(log(ro/ri)*ri);
    dcdr.CuO.vacCuMM = dVCuMM_r(1)*(log(ro/ri)*ri);
    dcdr.CuO.intCuPP = dintCuPP_r(1)*(log(ro/ri)*ri);
    dcdr.CuO.vacOPP = 0;
  case aSphere
    dcdr.CuO.h = dh(1)*((1/ri - 1/ro)*ri^-2);
    dcdr.CuO.e = de(1)*((1/ri - 1/ro)*ri^-2);
    dcdr.CuO.vacCuMM = dVCuMM_r(1)*((1/ri - 1/ro)*ri^-2);
    dcdr.CuO.intCuPP = dintCuPP_r(1)*((1/ri - 1/ro)*ri^-2);
    dcdr.CuO.vacOPP = 0;
  end
else
    dcdr.CuO.h = 0;
    dcdr.CuO.e = 0;
    dcdr.CuO.vacCuMM = 0;
    d cdr.CuO.intCuPP = 0;
    dcdr.CuO.vacOPP = 0;
end

C.1.3 Oxidation Model

This function runs the oxidation model (SpecsCalcOxSdndt.m).
function [outPutVar, outPutTime]=ParticleOxModel_nuc(y_Init, particle, regSite, Keq, dcdr)

% single particle model... with nucleation component

global sys tRange solveMoles pres02Vect

global Dx

% =========================================================================
% This function solves (nShells-1) x nVar spieces equations and unknowns

% initialize....
outPutTime = 0;
% =========================================================================
for jj = 1:(length(tRange.timeStep)-1)
% =========================================================================
% timeStep = [0 tRange.timeStep(jj+1)]; % time range for calculation
jacobScriptDefect % will select the correct jacobian pattern based on
areaModel, gasOn and bulkSpec variables...

if length(tRange.timeStep) == 2
% [time, outPut] = ode15s(@SpecsCalcOxStupidSimple, timeStep, y_Init,
% options,particle,regSite); % surface calc internal
if solveMoles
[t ime, outPut] = ode15s(@SpecsCalcOxSdndt, timeStep, y_Init, options,
% particle,regSite,Keq, dcdr); % surface calc internal
end
else
% solve moles w O2
[t ime, outPut] = ode15s(@SpecsCalcOxSdndt_vs2, timeStep, y_Init, options
% ,particle,regSite,Keq,JStruct); % surface calc internal
end
else
% ConcO2_S = sys.pres02/(Cn.gasconst*sys.temp); % moles / m^-3
sys.pres02_atm = pres02Vect(jj);
% regSite = defectEquilibriumFunc(sys);
[regSite Keq]= defectEquilibriumFuncDHi(sys, deltaGCuO, impur); % temp,pres
% ,pres02)
[t ime, outPut] = ode15s(@SpecsCalcOx_dt, timeStep, y_Init, options,particle,
% regSite); % surface calc external
end
% toc
% =========================================================================
% Store Data
% =========================================================================
nPoints = round(length(time)/3); % nPoints is the number of data points that are saved

if nPoints == 1
    nPoints = 2;
end
nLoc = round(linspace(1,length(time),nPoints));

outPutVar(nPointsP:(nPoints+nPointsP-1),:) = outPut(nLoc,:);
outPutTime(nPointsP:(nPoints+nPointsP-1)) = time(nLoc)+tRange.timeStepS(jj);

% =========================================================================
% redefine input matrix
% =========================================================================
y_Init = outPut(end,:)

% =========================================================================
% calculate extent of for the whole particle...
% =========================================================================
if length(tRange.timeStep) == 2
    % no need to run, were done.
    disp(['fraction of copper in Cu2O' num2str(outPut(end,2)/(outPut(end,1)+2*outPut(end,2))) ' %'])
    break
end

end

C.1.4 Oxidation Model

This function contains the differential equations that describe Cu oxidation.

function dy = SpecsCalcOxSdndt(t,y,particle,regSite,Keq,dcdrN)
% oxidation model, simple...
% this file models the oxidation of the Cu particle to Cu0
% JUST DEFECTS IN Copper

global sys Cn rho MW Dx

global nPhase surfArea

global shapeFactor

% assume linear defect profiles...
% DEFINE driving Variables....
C.1. CHEMICAL MODEL

13 % initial mass...
14 % total copper moles...
16 molesCu = sum(particle.mass.Cu)/MW.Cu + 2 * sum(particle.mass.Cu2O) / MW.Cu2O...
17 moles.Cu = y(1)*molesCu;
18 moles.Cu2O = y(2)*molesCu;
19
20 isCu2O = sys.presO2_atm > sys.presCuCu2O_atm;
21 isCuO = sys.presO2_atm > sys.presCu2OCuO_atm;
22 if isCuO
23 moles.CuO = y(3)*molesCu;
24 end
25
26 if y(1) < 1e-5
27 moles.Cu = 0;
28 end
29
30 % based on the number of moles, determine shell volumes...
31 switch shapeFactor
32 case 3 % sphere
33 thick.Cu = (3/(4*pi)*(moles.Cu*MW.Cu)/rho.Cu)^(1/3); % cm
34 thick.Cu2O = (((3/(4*pi)*moles.Cu2O*MW.Cu2O)/(rho.Cu2O*1-0) + thick.Cu
[^] -> 3)^(1/3)-thick.Cu); % cm
35 AoR.Cu2O = 4*pi*(thick.Cu)^2; % area at Cu/Cu2O interface...
36 if isCuO
37 thick.Cu0 = (((3/(4*pi)*moles.Cu0*MW.Cu0)/(rho.Cu0*1-0)) + (thick.Cu2O+[^] -> thick.Cu)^3)^(1/3)-thick.Cu2O-thick.Cu); % cm
38 AoR.Cu0 = 4*pi*(thick.Cu+thick.Cu2O)^2; % area at Cu2O/CuO
[^] -> interface...
39 end
40 case 2 % cylinder
41 thick.Cu = ((moles.Cu*MW.Cu)/(rho.Cu*pi*particle.wireLength))^0.5; % cm
42 thick.Cu2O = sqrt((moles.Cu2O*MW.Cu2O/(rho.Cu2O*particle.wireLength*pi))
[^] -> + thick.Cu^2) - thick.Cu; % cm
43 AoR.Cu2O = 2*pi*(thick.Cu)*particle.wireLength; % area at Cu/Cu2O
[^] -> interface...
44 if isCuO
45 thick.Cu0 = ((moles.Cu0*MW.Cu0)/(rho.Cu0*particle.wireLength*pi)) + ( thick.Cu2O+thick.Cu)^2)^(1/2)-thick.Cu2O-thick.Cu); % cm
46 AoR.Cu0 = 2*pi*particle.wireLength*(thick.Cu + thick.Cu2O); % area at
[^] -> Cu2O/CuO interface...
case 1 % plane
    AoR.Cu2O = surfArea; % area at Cu/Cu2O interface...
    % g - cm^-2/s^-2
    if isCuO
        AoR.CuO = surfArea; % area at Cu/Cu2O interface...
        % B [=] (mole of x) * (s)
        / (g-cm^-3)
    end
end

% ---
% diffusivity of defects...
% ---

% Dxx = [1 1 1 1];
if ~isempty(Dx)
    Dxx = Dx; % these is the self-diffusion coeff. for V_Cu'
end

% VACANCY DIFFUSION or defect diffusion coefficients
D_eff.Cu2O.vacCuX = Dxx(1); % cm^-2/s -- vac diffusion coef for vac x and vac '
D_eff.Cu2O.vacCuM = Dxx(1); % cm^-2/s
D_eff.Cu2O.intCuX = Dxx(2); % cm^-2/s
D_eff.Cu2O.intCuP = Dxx(2); % cm^-2/s
D_eff.Cu2O.vacOPP = DeffFuncDefect('Cu2O','vacOPP'); % cm^-2/s
D_eff.Cu2O.h = DeffFuncDefect('Cu2O','hole'); % cm^-2/s
D_eff.Cu2O.e = DeffFuncDefect('Cu2O','elec'); % cm^-2/s

if isinf(D_eff.Cu2O.vacOPP) || isnan(D_eff.Cu2O.vacOPP)
    D_eff.Cu2O.vacOPP = 0;
end

if isCuO
    % self diffusion coef...
    D_eff.CuO.vacCuMM = Dxx(3); % cm^-2/s, make a vacancy diffusivity from the
    % self diffusion coefficient...
    D_eff.CuO.intCuPP = Dxx(4); % cm^-2/s
    D_eff.CuO.vacOPP = 0*DeffFuncDefect('CuO','vacOPP'); % cm^-2/s
    % cm
    %^2/s
    % cm
    DRatio = 1; % hole to electron diffusivity...
    D_eff.CuO.h = DeffFuncDefect('CuO','hole')/((regSite.h.CuOCu2O)*(rho.CuO/
    % MW.CuO)+DRatio*(regSite.e.CuOCu20)*(rho.CuO/MW.CuO)); % cm
    %^2/s -- should be 'diffusivity' in terms of an electrical
    % conductivity or mobility....

C.1. CHEMICAL MODEL

D_eff.CuO.e = D_eff.CuO.h*DRatio;  % cm^-2/s

if isinf(D_eff.CuO.vacOPP) || isnan(D_eff.CuO.vacOPP)
  D_eff.CuO.vacOPP =0;
end

% =========================================================================
% Defect Profiles...
% =========================================================================
ri = thick.Cu;
ro = thick.Cu+thick.Cu2O;
switch shapeFactor
  case 1
    dcdr.Cu2O.h = dcdrN.Cu2O.h/thick.Cu2O;
    dcdr.Cu2O.e = dcdrN.Cu2O.e/thick.Cu2O;
    dcdr.Cu2O.vacCuM = dcdrN.Cu2O.vacCuM/thick.Cu2O;
    dcdr.Cu2O.vacCuX = dcdrN.Cu2O.vacCuX/thick.Cu2O;
    dcdr.Cu2O.intCuP = dcdrN.Cu2O.intCuP/thick.Cu2O;
    dcdr.Cu2O.intCuX = dcdrN.Cu2O.intCuX/thick.Cu20;
    dcdr.Cu2O.vacOPP = dcdrN.Cu2O.vacOPP/thick.Cu20;
  case 2
    % dVectN = dh(1)/(log(roOld/riOld)*riOld)*(log(ro/ri)*ri)
    dcdr.Cu20.h = dcdrN.Cu20.h/(log(ro/ri)*ri);
    dcdr.Cu20.e = dcdrN.Cu20.e/(log(ro/ri)*ri);
    dcdr.Cu20.vacCuM = dcdrN.Cu20.vacCuM/(log(ro/ri)*ri);
    dcdr.Cu20.vacCuX = dcdrN.Cu20.vacCuX/(log(ro/ri)*ri);
    dcdr.Cu20.intCuP = dcdrN.Cu20.intCuP/(log(ro/ri)*ri);
    dcdr.Cu20.intCuX = dcdrN.Cu20.intCuX/(log(ro/ri)*ri);
    dcdr.Cu20.vacOPP = dcdrN.Cu20.vacOPP/(log(ro/ri)*ri);
  case 3
    dcdr.Cu20.h = dcdrN.Cu20.h/((1/ri - 1/ro)*ri^2);
    dcdr.Cu20.e = dcdrN.Cu20.e/((1/ri - 1/ro)*ri^2);
    dcdr.Cu20.vacCuM = dcdrN.Cu20.vacCuM/((1/ri - 1/ro)*ri^2);
    dcdr.Cu20.vacCuX = dcdrN.Cu20.vacCuX/((1/ri - 1/ro)*ri^2);
    dcdr.Cu20.intCuP = dcdrN.Cu20.intCuP/((1/ri - 1/ro)*ri^2);
    dcdr.Cu20.intCuX = dcdrN.Cu20.intCuX/((1/ri - 1/ro)*ri^2);
    dcdr.Cu20.vacOPP = dcdrN.Cu20.vacOPP/((1/ri - 1/ro)*ri^2);
end

if isCuO
  ri = thick.Cu+thick.Cu20;
  ro = thick.Cu+thick.Cu20+thick.CuO;
switch shapeFactor
  case 1
    dcdr.Cu0.h = dcdrN.Cu0.h/thick.Cu0;
APPENDIX C. MATLAB CODES

```matlab
dcdr.CuO.e = dcdrN.CuO.e/thick.CuO;
dcdr.CuO.vacCuMM = dcdrN.CuO.vacCuMM/thick.CuO;
dcdr.CuO.vacOPP = 0;
dcdr.CuO.intCuPP = dcdrN.CuO.intCuPP/thick.CuO;

case 2

dcdr.CuO.h = dcdrN.CuO.h/(log(ro/ri)*ri);
dcdr.CuO.e = dcdrN.CuO.e/(log(ro/ri)*ri);
dcdr.CuO.vacCuMM = dcdrN.CuO.vacCuMM/(log(ro/ri)*ri);
dcdr.CuO.vacOPP = 0;
dcdr.CuO.intCuPP = dcdrN.CuO.intCuPP/(log(ro/ri)*ri);

case 3

dcdr.CuO.h = dcdrN.CuO.h/((1/ ri - 1/ro)*ri^2);
dcdr.CuO.e = dcdrN.CuO.e/((1/ ri - 1/ro)*ri^2);
dcdr.CuO.vacCuMM = dcdrN.CuO.vacCuMM/((1/ ri - 1/ro)*ri^2);
dcdr.CuO.vacOPP = 0;
dcdr.CuO.intCuPP = dcdrN.CuO.intCuPP/((1/ ri - 1/ro)*ri^2);
end
end

% site fractions at the Cu/Cu2O interface-- ave volume approach

numDens.Cu2O.vacCuX = regSite.vacCuX.CuCu2O * rho.Cu2O / MW.Cu2O;
numDens.Cu2O.vacCuM = regSite.vacCuM.CuCu2O * rho.Cu2O / MW.Cu2O;
numDens.Cu2O.h = regSite.h.CuCu2O * rho.Cu2O / MW.Cu2O;
numDens.Cu2O.e = regSite.e.CuCu2O * rho.Cu2O / MW.Cu2O;
numDens.Cu2O.intCuX = regSite.intCuX.CuCu2O * rho.Cu2O / MW.Cu2O;
numDens.Cu2O.intCuP = regSite.intCuP.CuCu2O * rho.Cu2O / MW.Cu2O;
numDens.Cu2O.vacOPP = regSite.vacOPP.CuCu2O * rho.Cu2O / MW.Cu2O;

if isCuO
    % Cu2O/CuO interface
    numDens.CuO.vacCuMM = regSite.vacCuMM.CuOCu2O * rho.Cu0/MW.Cu0;  % moles
    numDens.CuO.vacOPP = regSite.vacOPP.CuOCu2O * rho.Cu0/MW.Cu0;  % moles
    numDens.CuO.intCuPP = regSite.intCuPP.CuOCu2O * rho.Cu0/MW.Cu0;  % moles
    numDens.CuO.h = regSite.h.CuOCu2O * rho.Cu0/MW.Cu0;  % moles of i
    numDens.CuO.e = regSite.e.CuOCu2O * rho.Cu0/MW.Cu0;  % moles of i
end

% define B-variable; B_i [*] D_i * C_i / (kB*T),  J [*] kg-m^2/s^2 = 10^-7 g-m^-2/s^2
```
C.1. CHEMICAL MODEL

\[ B.\text{Cu}_2\text{O}.\text{vacCuX} = D_{\text{eff}.\text{Cu}_2\text{O}.\text{vacCuX}} \cdot \text{numDens}.\text{Cu}_2\text{O}.\text{vacCuX}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{Cu}_2\text{O}.\text{vacCuM} = D_{\text{eff}.\text{Cu}_2\text{O}.\text{vacCuM}} \cdot \text{numDens}.\text{Cu}_2\text{O}.\text{vacCuM}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{Cu}_2\text{O}.\text{vacOPP} = D_{\text{eff}.\text{Cu}_2\text{O}.\text{vacOPP}} \cdot \text{numDens}.\text{Cu}_2\text{O}.\text{vacOPP}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{Cu}_2\text{O}.\text{intCuX} = D_{\text{eff}.\text{Cu}_2\text{O}.\text{intCuX}} \cdot \text{numDens}.\text{Cu}_2\text{O}.\text{intCuX}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{Cu}_2\text{O}.\text{intCuP} = D_{\text{eff}.\text{Cu}_2\text{O}.\text{intCuP}} \cdot \text{numDens}.\text{Cu}_2\text{O}.\text{intCuP}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{Cu}_2\text{O}.\text{h} = D_{\text{eff}.\text{Cu}_2\text{O}.\text{h}} \cdot \text{numDens}.\text{Cu}_2\text{O}.\text{h}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{Cu}_2\text{O}.\text{e} = D_{\text{eff}.\text{Cu}_2\text{O}.\text{e}} \cdot \text{numDens}.\text{Cu}_2\text{O}.\text{e}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  

\[ \text{if isCuO} \]
\[ B.\text{CuO}.\text{vacCuMM} = D_{\text{eff}.\text{CuO}.\text{vacCuMM}} \cdot \text{numDens}.\text{CuO}.\text{vacCuMM}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{CuO}.\text{vacOPP} = D_{\text{eff}.\text{CuO}.\text{vacOPP}} \cdot \text{numDens}.\text{CuO}.\text{vacOPP}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{CuO}.\text{intCuPP} = D_{\text{eff}.\text{CuO}.\text{intCuPP}} \cdot \text{numDens}.\text{CuO}.\text{intCuPP}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{CuO}.\text{h} = D_{\text{eff}.\text{CuO}.\text{h}} \cdot \text{numDens}.\text{CuO}.\text{h}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ B.\text{CuO}.\text{e} = D_{\text{eff}.\text{CuO}.\text{e}} \cdot \text{numDens}.\text{CuO}.\text{e}/(Cn.\text{b} \cdot \text{sys.temp} \cdot 10^{-7}) \]  
\[ \text{end} \]

% Vacancies, ions and interstitials...

\[ \text{dy} = \text{zeros}(\text{nPhase}, 1); \]

% calculate deravitives of mid-points: dc/dr [*] moles/m, d2c/dr^2 = moles/m^2

if ~isCuO % i.e if this only Cu2O

\[ \phi.\text{Cu}_2\text{O} = 1/(Cn.\text{e} \cdot Cn.\text{Na}) \cdot ((D_{\text{eff}.\text{Cu}_2\text{O}.\text{vacCuM}} \cdot \text{dcdr}.\text{Cu}_2\text{O}.\text{vacCuM} + D_{\text{eff}.\text{Cu}_2\text{O}.\text{e}} \cdot \text{dcdr}.\text{Cu}_2\text{O}.\text{e}) - \ldots) \]
\[ (D_{\text{eff}.\text{Cu}_2\text{O}.\text{intCuP}} \cdot \text{dcdr}.\text{Cu}_2\text{O}.\text{intCuP} + D_{\text{eff}.\text{Cu}_2\text{O}.\text{h}} \cdot \text{dcdr}.\text{Cu}_2\text{O}.\text{h} + \ldots) 2 \cdot D_{\text{eff}.\text{Cu}_2\text{O}.\text{vacOPP}} \cdot \text{dcdr}.\text{Cu}_2\text{O}.\text{vacOPP})/\ldots \]
\[ (B.\text{Cu}_2\text{O}.\text{intCuP} + B.\text{Cu}_2\text{O}.\text{vacCuM} + B.\text{Cu}_2\text{O}.\text{e} + B.\text{Cu}_2\text{O}.\text{h} + 4 \cdot \ldots) \]
\[ B.\text{Cu}_2\text{O}.\text{vacOPP}); \]

% define at Cu/Cu2O interface

J.\text{Cu}_2\text{O}.\text{vacCuM} = -D_{\text{eff}.\text{Cu}_2\text{O}.\text{vacCuM}} \cdot \text{dcdr}.\text{Cu}_2\text{O}.\text{vacCuM} + B.\text{Cu}_2\text{O}.\text{vacCuM} \cdot Cn.\text{e} \leftrightarrow Cn.\text{Na} \cdot \phi.\text{Cu}_2\text{O} ;

J.\text{Cu}_2\text{O}.\text{vacCuX} = -D_{\text{eff}.\text{Cu}_2\text{O}.\text{vacCuX}} \cdot \text{dcdr}.\text{Cu}_2\text{O}.\text{vacCuX};

J.\text{Cu}_2\text{O}.\text{intCuP} = -D_{\text{eff}.\text{Cu}_2\text{O}.\text{intCuP}} \cdot \text{dcdr}.\text{Cu}_2\text{O}.\text{intCuP} - B.\text{Cu}_2\text{O}.\text{intCuP} \cdot Cn.\text{e} \leftrightarrow Cn.\text{Na} \cdot \phi.\text{Cu}_2\text{O} ;

\]
J.Cu20.intCuX = -D_eff.Cu20.intCuX*dcdr.Cu20.intCuX;
    ⇔ Cn.Na*phi.Cu20;
else
% -----------------------------------------------------------------------
% Cu/Cu20 interface
% -----------------------------------------------------------------------
    ⇔ *dcdr.Cu20.e) - ...)
    (D_eff.Cu20.intCuP*dcdr.Cu20.intCuP + D_eff.Cu20.h*dcdr.Cu20.h +
    ⇔ 2*D_eff.Cu20.vacOPP*dcdr.Cu20.vacOPP))/...
    (B.Cu20.intCuP + B.Cu20.vacCuM + B.Cu20.e + B.Cu20.h + 4*
    ⇔ B.Cu20.vacOPP); % (cm^-2/s) * (mole of x/cm^3) / ((
    ⇔ mole of x) * (s) / (g-cm^3))
%
    ⇔ Cn.Na*phi.Cu20; % 1/cm x (mole of x * s / (g-cm^-3)) x J
J.Cu20.vacCuX = -D_eff.Cu20.vacCuX*dcdr.Cu20.vacCuX; % mole of x / cm^-2 - s
    ⇔ 1/cm x (mole of x * s / (g-cm^-3)) x g-cm^-2/s^-2
J.Cu20.intCuP = -D_eff.Cu20.intCuP*dcdr.Cu20.intCuP - B.Cu20.intCuP * Cn.e*
    ⇔ Cn.Na*phi.Cu20; % mole of x / cm^-2 - s
J.Cu20.intCuX = -D_eff.Cu20.intCuX*dcdr.Cu20.intCuX;
    ⇔ Cn.Na*phi.Cu20;
% x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+
C.1. CHEMICAL MODEL

```
232 \phi_{\text{CuO}} = 1/(C_{\text{e}} \cdot C_{\text{Na}}) \times \left( 2 \times D_{\text{eff}} \cdot \text{CuO} \times \text{vacCuMM} \times dcdr \cdot \text{CuO} \cdot \text{vacCuMM} + D_{\text{eff}} \cdot \text{CuO} \cdot e \right. 
\left. \rightarrow \times dcdr \cdot \text{CuO} \cdot e \right) - ... 
\left( 2 \times D_{\text{eff}} \cdot \text{CuO} \cdot \text{intCuPP} \times dcdr \cdot \text{CuO} \cdot \text{intCuPP} + D_{\text{eff}} \cdot \text{CuO} \cdot h \times dcdr \cdot \text{CuO} \cdot h \right. 
\left. \rightarrow \times 2 \times D_{\text{eff}} \cdot \text{CuO} \cdot \text{vacOPP} \times dcdr \cdot \text{CuO} \cdot \text{vacOPP} \right) / ... 
\left( 4 \times B \cdot \text{CuO} \cdot \text{intCuPP} + 4 \times B \cdot \text{CuO} \cdot \text{vacCuMM} + B \cdot \text{CuO} \cdot e + B \cdot \text{CuO} \cdot h + 4 \right. 
\left. \rightarrow \times B \cdot \text{CuO} \cdot \text{vacOPP} \right); 
235
236 J_{\text{CuO} \cdot \text{vacCuMM}} = - D_{\text{eff}} \cdot \text{CuO} \cdot \text{vacCuMM} \times dcdr \cdot \text{CuO} \cdot \text{vacCuMM} + 2 \times B \cdot \text{CuO} \cdot \text{vacCuMM} \times C_{\text{e}} \times C_{\text{Na}} \times \phi_{\text{CuO}}; 
237 J_{\text{CuO} \cdot \text{intCuPP}} = - D_{\text{eff}} \cdot \text{CuO} \cdot \text{intCuPP} \times dcdr \cdot \text{CuO} \cdot \text{intCuPP} - 2 \times B \cdot \text{CuO} \cdot \text{intCuPP} \times C_{\text{e}} \times C_{\text{Na}} \times \phi_{\text{CuO}}; 
238 J_{\text{CuO} \cdot \text{vacOPP}} = - D_{\text{eff}} \cdot \text{CuO} \cdot \text{vacOPP} \times dcdr \cdot \text{CuO} \cdot \text{vacOPP} - 2 \times B \cdot \text{CuO} \cdot \text{vacOPP} \times C_{\text{e}} \times C_{\text{Na}} \times \phi_{\text{CuO}}; 
239 J_{\text{CuO} \cdot h} = - D_{\text{eff}} \cdot \text{CuO} \cdot h \times dcdr \cdot \text{CuO} \cdot h - B \cdot \text{CuO} \cdot h \times C_{\text{e}} \times C_{\text{Na}} \times \phi_{\text{CuO}}; 
240 J_{\text{CuO} \cdot e} = - D_{\text{eff}} \cdot \text{CuO} \cdot e \times dcdr \cdot \text{CuO} \cdot e + B \cdot \text{CuO} \cdot e \times C_{\text{e}} \times C_{\text{Na}} \times \phi_{\text{CuO}}; 
241
242 \% \phi \left[=\right] \text{cm}^{-2} \times \text{g} / (\text{A-s}^{-3}) 
243
244 \% B \left[=\right] \text{mole} \times \text{s} / (\text{g-cm}^{-3}) 
245
246 \% \text{------------------------------------------} 
247
248 \% no more Cu 
249 if y(1) <= 1e-5 
250 J_{\text{Cu2O} \cdot \text{vacCuM}} = 0; 
251 J_{\text{Cu2O} \cdot \text{vacCuX}} = 0; 
252 J_{\text{Cu2O} \cdot \text{intCuP}} = 0; 
253 J_{\text{Cu2O} \cdot \text{intCuX}} = 0; 
254 J_{\text{Cu2O} \cdot \text{vacOPP}} = 0; 
255 J_{\text{Cu2O} \cdot h} = 0; 
256 J_{\text{Cu2O} \cdot e} = 0; 
257 end 
258
259 \% \text{------------------------------------------} 
260 \% ----- CALCULATE FUXES OF EACH SPECIES ----- 
261 \% \text{------------------------------------------} 
262
263 \% Flux equations \left[=\right] \text{moles} / \text{cm}^{-2} \times \text{s} 
264 neutralityCond.Cu2O = J_{\text{Cu2O} \cdot \text{intCuP}} + 2 \times J_{\text{Cu2O} \cdot \text{vacOPP}} + J_{\text{Cu2O} \cdot h} - ( J_{\text{Cu2O} \cdot \text{vacCuM}} + J_{\text{Cu2O} \cdot e}); 
265 neutralityCond.CuO = 0; 
266 if isCuO 
267 neutralityCond.CuO = 2 \times J_{\text{CuO} \cdot \text{intCuPP}} + 2 \times J_{\text{CuO} \cdot \text{vacOPP}} + J_{\text{CuO} \cdot h} - (2 \times J_{\text{CuO} \cdot \text{vacCuMM}} + J_{\text{CuO} \cdot e}); 
268 end 
269
```
if sum(abs(neutralityCond.Cu2O))>1e-8 || sum(abs(neutralityCond.CuO))>1e-8
    disp('neutrality not met...')
end

% [units]
% D [=] m^2/s
% dcdr [=] atoms/m^4
% B [=] CD/kbT [=] (atoms/m^3 * m^2/s) / (J/atom)
% e*z*V/m [=] columbs/atom * V/m = J/(atom-m)

% net flux.... contributing to Cu2O growth...

% should dN_Cu = - dN_Cu2O.... when only two species
% yes , because if Cu decreases dN_Cu2O must increase... where else where the copper go? the concentration profile will not be linear in a spherical particle...

% sign of fluxes...
J.Cu20.e > 0
J.Cu20.h < 0
J.Cu20.vacOPP > 0
J.Cu20.vacCuMM < 0
J.Cu20.intCuPP > 0

if shapeFactor == 3 || shapeFactor == 2
    dnCu = (J.Cu20.intCuX + J.Cu20.intCuP)/molesCu*AoR.Cu20...
    - (J.Cu20.vacCuX + J.Cu20.vacCuM)/molesCu*AoR.Cu20...
    + 2*J.Cu20.vacOPP/molesCu*AoR.Cu20;
    dnCu20 = dnCu/2;
else
    dy(1) = -dnCu;
if ~isCuO
dy(2) = dnCu2O;

else

% I define the change of CuO as any copper coming from Cu2O
% I define the change of Cu2O as requiring a Cu -O pair from CuO

% this term represents copper atoms leaving Cu2O and entering CuO
dnCuO = J.CuO.intCuPP/molesCu*AoR.CuO-J.CuO.vacCuMM/molesCu*AoR.CuO ... 
+J.CuO.vacOOP/molesCu*AoR.CuO;
% NOTE: Copper atoms only move outward
% Oxygen atoms only move inward
% when there is a Cu Core
% --> Cu -O atom pairs go from CuO to Cu2O
% --> Cu atoms jump from Cu2O to CuO
% dy(2) = dnCu2O - dnCuO/2; % growth of CuO leads to destruction of
% dy(3) = dnCuO; % Growth of Cu2O leads to a decrease of CuO

% if y(1) <= 1e-8 % when there is not a Cu core
% dy(2) = -dnCuO; % Cu2O
% dy(3) = dnCuO*2; % CuO
% else % when there is a Cu Core
% dy(2) = dnCu20 - dnCu0/2; % growth of CuO leads to destruction of
% dy(3) = dnCu0; % Growth of Cu20 leads to a decrease of CuO
% end
% end
% =========================================================================
% =========================================================================

elseif shapeFactor == 1

dnCu = (J.Cu20.intCuX + J.Cu20.intCuP)/molesCu*surfArea ... 
-(J.Cu20.vacCuX + J.Cu20.vacCuM)/molesCu*surfArea;
dnCu2O = dnCu/2;
dy(1) = -dnCu; % Cu20 thickness
if ~isCuO

dy(2) = dnCu2O; % Cu20 thickness
else

dnCuO = J.CuO.intCuPP/molesCu*surfArea-J.CuO.vacCuMM/molesCu*surfArea;
% dnCuO = J.CuO.intCuPP/molesCu*surfArea-J.CuO.vacCuMM* 
% regSite.vacCuMM.CuO/Cu20/relSite.Cu0/Cu/molesCu*surfArea; %reg site ratio
% not used b/c Deff_V"' is actually for a Cu Ion...
% if y(1) <= 1e-8 % when there is not a Cu core
% dy(2) = -dnCuO; % Cu2O
% dy(3) = dnCuO*2; % CuO
% else % when there is a Cu Core
% dy(2) = dnCu20 - dnCu0/2; % growth of CuO leads to destruction of
% dy(3) = dnCu0; % Growth of Cu20 leads to a decrease of CuO
% end
% end
else
error('wronggg option')
end
% =========================================================================
% =========================================================================

C.1. CHEMICAL MODEL
if y(2) <=0
    dy(:) =0;
end

if isCu2O == 0 && isCuO == 0
    % no reaction ...
    dy(:) = 0;
end

if isCuO == 0 && particle.CuO.thick >0;
    dy(3) = 0;
end

% atom balance
% if abs ( moles.Cu + 2*moles.Cu2O + moles.CuO - molesCu )/ molesCu > 1e-6
% a=1;
% end

% =========================================================================
% =========================================================================
% [ units ]
% J [=} atoms/m^2-s
% rho [=} grams/m^3
% MW/Na [=} grams/mole*moles/atom [=} grams/atom
% dy [=} m^3/atom * atom/m^2-s = m / s

C.2 Cermet Particle Model

This section contains code related to the cermet particle model. The code is setup with the same structure as the oxidation model. It starts out with a driver function. This driver function calls an initialization function with inputs to the model. The particle model is then run which contains the partial differential equations that describe oxygen diffusion and copper oxidation.

C.2.1 Driver Code

% Mode description
% This is the driver file for calculations modeling the oxidation of Cu in
C.2. CERMET PARTICLE MODEL

% oxygen; oxidation is described by transport of crystalline and electronic
% defects in copper cuprous and cupric oxide lattices
% OD [=] oxygen diffusion through shells...

% Creation Date: 1/30/12
% Update: 10/20/14 -- add in option to turn off rxs if CL not reached--
% don't use though bc it will oscillate from one solution to another

clear all
tic

InitFileparticleOD

% profile on
[output, time] = ParticleOxModel_OD(y_init, particle, particleV, shell, impur, 
    deltaGCuO, dETAdrCu2O, dETAdrCuO, D_eff); % gas noGas and bulk noBulk

% profile viewer
% output = [outputVar; output(2:end,:)];
% time = [outputTime time(2:end)+outputTime(end)];
% output = [output; outputVar(2:end,:)];
% time = [time, outputTime(2:end)+time(end)];
% output = outputVar;
% time = outputTime;
% save('900Cdata', 'time', 'output')
% output = [output; outputVar(2:end,:)];
% time = [time (outputTime(2:end)+time(end))];
% output = outputVar;
% time = outputTime;
% outputVar = output;
% outputTime = time;
toc

% profile off
% CuPerParticle = particleV.Cu.mass/MW.Cu+2*paregSite.h.Cu0O2rticleV.Cu2O.mass/
    MW.Cu2O+particleV.CuO.mass/MW.CuO;

if length(particleV.tot.CuMoles) == 1
    outputmoles.Cu = zeros(length(time),nVols);
    outputgas.O2 = output(:,1:nVols); %/(sys.presTot/(8.314*sys.temp*100^3));
    outputmoles.Cu = output(:,nVols+1).*particleV.tot.CuMoles; % moles in Cu2O
    outputmoles.Cu2O = output(:,nVols+2).*particleV.tot.CuMoles; % moles in Cu2O
    outputmoles.Cu0 = particleV.tot.CuMoles - outputmoles.Cu2O*2 - 
        outputmoles.Cu;

41
42
43
44
45
46
APPENDIX C. MATLAB CODES

```matlab
xi_time = (sum(repmat(shell.Me.numb',length(time),1).*(outPutmoles.Cu*MW.Cu+
  outPutmoles.Cu2O*MW.Cu2O+outPutmoles.CuO*MW.CuO),2) - ...
  sum(shell.Me.CuMoles*MW.Cu))./sum(shell.Me.CuMoles*MW.O);
else
  outPutmoles.Cu = zeros(length(time),nVols);
  outPutgas.O2 = outPut(:,1:nVols); %/(sys.presTot/(8.314*sys.temp*100^3));
  outPutmoles.Cu = outPut(:,nVols+(1:nVols)).*particleV.tot.CuMoles(1); %
  moles in Cu2O
  outPutmoles.Cu2O = outPut(:,2*nVols+(1:nVols)).*particleV.tot.CuMoles(1); %
  moles in Cu2O
  outPutmoles.CuO = outPut(:,3*nVols+(1:nVols)).*particleV.tot.CuMoles(1);
  xi_time = (sum(repmat(shell.Me.numb',length(time),1).*(outPutmoles.Cu*MW.Cu+
  outPutmoles.Cu2O*MW.Cu2O+outPutmoles.CuO*MW.CuO)),2) - ...
  sum(shell.Me.CuMoles*MW.Cu))./sum(shell.Me.CuMoles*MW.O);
end

% take output and post process
% for i = 1:length(time)
%  outPutgas.O2(i,:) = (outPutAA(i,(1:nVols))'); %/(sys.presTot/(8.314*
  sys.temp*100^3));
%  outPutmoles.Cu(i,:) = outPutAA(i,nVols+(1:nVols))'*CuPerParticle(1:nVols
  <->);
%  outPutmoles.Cu2O(i,:) = outPutAA(i,2*nVols+(1:nVols))'*CuPerParticle(1:
  nVols);。
%  outPutmoles.CuO(i,:) = CuPerParticle(1:nVols) - outPutmoles.Cu2O(i,:);%
%  xi_time(i) = (sum(shell.Me.numb'*(outPutmoles.Cu(i,:)*MW.Cu+
    outPutmoles.Cu2O(1,:))*MW.Cu2O+outPutmoles.CuO(1,:)*MW.CuO)) - ...
%  sum(shell.Me.CuMoles*MW.Cu))./sum(shell.Me.CuMoles*MW.O);
% end

% SMOOTH OUT Time...
clear outPutgas_S outPutmoles_S thickOut XO2_s
% splitPoint = find(time >=.1,1);
% newTime = [logspace(log10(time(2)),log10(time(splitPoint)),round(length(time(1:
  <-> splitPoint)))))/10 ...]
%  linspace(time(splitPoint+1),time(end),(length(time)-splitPoint)
%  ->3)];
newTime = time;

if length(particleV.tot.CuMoles) ==1
  for jj = 1:nVols
    outPutgas_S.O2(:,jj) = spline(time,outPutgas.O2(:,jj)',newTime);
  end
  outPutmoles_S.Cu = spline(time,outPut(:,nVols+1)*particleV.tot.CuMoles, %
  newTime); % moles in Cu2O
```
C.2. CERMET PARTICLE MODEL

```matlab
outPutmoles_S.Cu2O = spline(time, outPut(:, nVols+2)*particleV.tot.CuMoles, newTime);  % moles in Cu2O
outPutmoles_S.CuO = particleV.tot.CuMoles - outPutmoles_S.Cu - outPutmoles_S.Cu2O*2;

thickOut.Cu = (3/(4*pi)* outPutmoles_S.Cu/rho.Cu*MW.Cu).^(1/3);

else
    for jj = 1:nVols
        outPutgas_S.O2(:, jj) = spline(time, outPutgas.O2(:, jj)', newTime);
        outPutmoles_S.Cu(:, jj) = spline(time, outPut(:, nVols+jj)', particleV.tot.CuMoles(1), newTime);  % moles in Cu
        outPutmoles_S.Cu2O(:, jj) = spline(time, outPut(:, 2*nVols+jj)', particleV.tot.CuMoles(1), newTime);  % moles in Cu2O
        outPutmoles_S.CuO(:, jj) = spline(time, outPut(:, 3*nVols+jj)', particleV.tot.CuMoles(1), newTime);  % moles in CuO
        % outPutmoles_S.CuO(:, jj) = particleV.tot.CuMoles(1) - outPutmoles_S.Cu(:, jj) - outPutmoles_S.Cu2O(:, jj) * 2;

        thickOut.Cu(:, jj) = (3/(4*pi)* outPutmoles_S.Cu(:, jj)/rho.Cu*MW.Cu).^(1/3);
    end
end

figure(2)
hold on
plot(time, xi_time, 'displayName', [num2str(particle.Me.diam*10^-4) ' mum']);
plot([17.99 30], .995*[1 1], 'r')
plot([0 length(time)]/60, 0.21*[1 1])
plot(tRange.timeStepS/60, presO2Func(tRange.timeStepS), 'r')
plot(time/60, presO2Func(time), 'r')
xlabel('Time, Seconds')
ylabel('Conversion, [\cdot]')
grid on

% hold on
% plot([1e-7 30],[1 1]*sys.presCuCu2O_atm*1.1,'g')
% plot([1e-7 30],[1 1]*sys.presCu2OCuO_atm,'g')
```
% ylim ([0 8e-4])

if 0
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_12.3121_finWeight_12.574mg_in_21pcntO_date_030314.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_18.8935_finWeight_0mg_in_21pcntO_date_032614.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_19.329_finWeight_0mg_in_21pcntO_date_032714.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_19.6676_finWeight_22.32mg_in_21pcntO_date_040114.txt';
files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_13.7189_finWeight_14.58mg_in_21pcntO_date_030514.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_14.6255_finWeight_14.484mg_in_21pcntO_date_030714.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_13.0899_finWeight_0mg_in_21pcntO_date_051414.txt';
files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_13.7189_finWeight_14.58mg_in_21pcntO_date_030514.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_14.6255_finWeight_14.484mg_in_21pcntO_date_030714.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_13.0899_finWeight_0mg_in_21pcntO_date_051414.txt';
files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_13.8641_finWeight_15.193mg_in_21pcntO_date_022514.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_8.9983_finWeight_0mg_in_21pcntO_date_051914.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_8.6083_finWeight_0mg_in_21pcntO_date_021814.txt';
files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_8.4823_finWeight_10.27mg_in_21pcntO_date_020614.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_10.1386_finWeight_0mg_in_21pcntO_date_021714.txt';
  Files = '.../TGA_ANalysis/fitData/Cycles/
          init_weight_10.5456_finWeight_10.17mg_in_21pcntO_date_021814.txt';
% Files = '../TGA_ANalysis/fitData/Cycles/
% Time_mTotal_CuOZrO2_20PcntCu_0.16647_Reactivity_T =800
% init_weight_12.502_finWeight_12.637mg_in_21pcntO_date_031114.txt'

fid = fopen (Files , 'r');
A = fscanf (fid , '%e', [3 inf]);
A(:,4) = (A(:,2)-A(1,2))/A(1,2)*MW.Cu/MW.O/0.16647; % extent of conversion
endTime = A(end,1)/60-61;
timeExp = A(:,find(A(:,1)/60>=endTime,1))/60; % minutes
mass_t = interp1(A(:,1)/60,A(:,2),timeExp);
RR_t = interp1(A(:,1)/60,A(:,3),timeExp); % reactivity... 1/m*dm/ dt....
xi_t = interp1(A(:,1)/60,A(:,4),timeExp);
hold on
plot(timeExp*60,xi_t)
dmdt = dxCalc(mass_t,timeExp*60);
hold on
plot(xi_t,dmdt./mass_t)
end

% figure(16)
% hold on
% timeSearch = [logspace(-8,-1,40)];
% for i = 1:length(timeSearch)
% try
% timeIndex = find(timeSearch(i) < time,1);
% catch
% timeIndex = length(time);
% end
% if newTime(i)>.01
% semilogx(shell.radius(1:nVols)*10^-4,outPutgas.O2(timeIndex,:),'k','
% displayName',[num2str(time(timeIndex),'%2.5g\n'))
% else
% semilogx(shell.radius(1:nVols)*10^-4,outPutgas.O2(timeIndex,:),'k','
% displayName',[num2str(time(timeIndex),'%2.2e\n'))
% end
% end
% ylabel ('O2 Mole Fraction')
figure;
plot (time, outPutgas.O2 (:,1), 'DisplayName', 'Shell 1')
hold on
plot (time, outPutgas.O2 (:,3), 'DisplayName', 'Shell 3')
plot (time, outPutgas.O2 (:,8), 'DisplayName', 'Shell 8')
plot (time, outPutgas.O2 (:,10), 'DisplayName', 'Shell 10')
plot (time, outPutgas.O2 (:,nVols-10), 'DisplayName', 'Shell nVols-10')
plot (time, outPutgas.O2 (:,nVols), 'DisplayName', 'Shell nVols')
xlabel ('Time, seconds')
ylabel ('O2 Mole Fraction, [-]')

vidObj = VideoWriter ('peaks.avi');
open (vidObj);
Z = vidObj;
figure (Z)
set (gca, 'nextplot', 'replacechildren');
set (gcf, 'Renderer', 'zbuffer');
axis tight
for i = 1:length (newTime (1:20))
    Ntotal = (outPutmoles_S.Cu(i,:) + outPutmoles_S.Cu2O(i,:)+ outPutmoles_S.CuO(i,:));
    plot (shell.xi (1:nVols)*particle.diam/2*10^-4, outPutmoles_S.Cu(i,:)./Ntotal, '-ob', 'DisplayName', 'Cu(t)', 'lineWidth', 3)
    hold on
    plot (shell.xi (1:nVols)*particle.diam/2*10^-4, outPutmoles_S.Cu2O(i,:)./Ntotal, '-or', 'DisplayName', 'Cu_2O(t)', 'lineWidth', 3)
    plot (shell.xi (1:nVols)*particle.diam/2*10^-4, outPutmoles_S.CuO(i,:)./Ntotal, '-og', 'DisplayName', 'CuO(t)', 'lineWidth', 3)
    % plot (shell.xi (1:nVols)*particle.diam/2*10^-4, outPutgas_S.O2 (i,:), '-ok',
    %       DisplayName, 'O_2(t)', 'lineWidth', 3)
    hold off
    xlim ([0 particle.diam/2*10^-4+1])
    ylim ([0 1.05])
    text (15, .4, ['time = ' num2str (newTime (i), 10)], 'BackgroundColor', [.7 .9 .7])
    legend toggle
    grid on
    ylabel ('O2 mole fraction, [-]')
    xlabel ('Radius, [ m]')
    if newTime (i)>0
        pause
    end
    drawnow
    currFrameMF (i) = getframe;
% end
% writeVideo(vidObj,currFrameMF);
% close(vidObj);

% figure(5)
% for i = 1:length(time)
% Ntotal = (outPutmoles.Cu(i,:) + outPutmoles.Cu2O(i,:) + outPutmoles.CuO(i,:))';
% plot(shell.xi(1:nVols)*particle.diam/2*10^4, outPutmoles.Cu(i,:)'./Ntotal,'-ob','DisplayName','Cu(t)','lineWidth',3)
% hold on
% plot(shell.xi(1:nVols)*particle.diam/2*10^4, outPutmoles.Cu2O(i,:)'./Ntotal,'-or','DisplayName','Cu_2O (t)','lineWidth',3)
% plot(shell.xi(1:nVols)*particle.diam/2*10^4, outPutmoles.CuO(i,:)'./Ntotal,'-og','DisplayName','CuO (t)','lineWidth',3)
% hold off
% xlim([0 particle.diam/2*10^4+1])
% ylim([0 1 .05])
% text(15,.4,['time = ' num2str(time(i),10)],'BackgroundColor',[.7 .9 .7])
% legend toggle
% grid on
% ylabel('O2 mole fraction, [-]')
% xlabel('Radius, [ m]')
% if newTime(i)>0
% pause
% end
% drawnow
% currFrameMF(i) = getframe;
% end

figure(6)
hold on
if strcmp(manufactMethod,'CORE')
nTot = (outPutmoles_S.Cu*MW.Cu+outPutmoles_S.Cu2O*MW.Cu2O+outPutmoles_S.CuO*MW.CuO)*sum(shell.Me.numb);
plot(newTime/60,(outPutmoles_S.Cu*MW.Cu)*sum(shell.Me.numb)/nTot,'r','DisplayName','Cu')
plot(newTime/60,(outPutmoles_S.Cu2O*MW.Cu2O)*sum(shell.Me.numb)/nTot,'g','DisplayName','Cu2O')
plot(newTime/60,(outPutmoles_S.CuO*MW.CuO)*sum(shell.Me.numb)/nTot,'b','DisplayName','CuO')
end
hold off
```matlab
else
    nTot = sum(outPutmoles_S.Cu*MW.Cu+outPutmoles_S.Cu2O*MW.Cu2O+
               outPutmoles_S.CuO*MW.CuO,2)*sum(shell.Me.numb);       
    plot(newTime/60,sum(outPutmoles_S.Cu*MW.Cu,2)*sum(shell.Me.numb)./nTot,'r',
         'DisplayName','Cu')
    plot(newTime/60,sum(outPutmoles_S.Cu2O*MW.Cu2O,2)*sum(shell.Me.numb)./nTot,'g'
         'DisplayName','Cu2O')
    plot(newTime/60,sum(outPutmoles_S.CuO*MW.CuO,2)*sum(shell.Me.numb)./nTot,'b',
         'DisplayName','CuO')
end
xlabel('Time, [min]')
ylabel('Mass Fraction, [-]')

% copper atom balance...
figure(14); hold on
xlabel('time, sec')
ylabel('copper atoms')
grid
plot(time,1-((sum(outPutmoles.Cu,2)+sum(outPutmoles.Cu2O,2)*2+
             sum(outPutmoles.CuO,2))/sum(particleV.tot.CuMoles))

% calculate 1/m x dm/dt
mTotal = sum(outPutmoles.Cu,2)*MW.Cu+sum(outPutmoles.Cu2O,2)*MW.Cu2O+
         sum(outPutmoles.CuO,2)*MW.CuO; % mass of single particle, grams
xiTot = (mTotal-mTotal(1))/(mTotal(1)*MW.CuO/MW.Cu*particle.pcntCu);
dmdt = dxCalc(mTotal,time);
OCreactivity = dmdt./mTotal;
figure(3);
hold on
plot(xi_time,OCreactivity,'displayName',['T = ' num2str(sys.temp-273.15) setstr
         '(176) ' ' C'])
xlabel('Conversion, [-]')
ylabel('Reactivity, 1/s')

% average reactivity
% spline(xi_time,OCreactivity,0.5)
aveReact = trapz(xi_time,OCreactivity.*xi_time)

% time for 95% conversion
```
spline(xi_time,time,0.95) % seconds
plot(0.5,aveReact,'x')
hold on

% dat = [600 0.00015436 0.0003655
700 0.00072498 0.001988
800 0.0027056 0.00788
900 0.0060333 0.02249];

% dat=[600 0.0103
700 0.0552
800 0.2198
900 0.63];
hold on;
plot(dat(:,1),dat(:,2),'x')
figure;plot(time,dmdt)
plot(time,xiTot)
xlabel('Conversion')
ylabel('Reactivity, 1/m x dm/dt')
grid on

% PlotFixer
break

xoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxoxox
figure(18)
hold on
timePlots = [0 1e-7 1e-6 1e-5 1e-4 1]; % seconds
% timePlots = [0 1e-6 1e-5 1e-4]; % seconds
for i = 1:length(timePlots)
    nPoints = find(timePlots(i) <= newTime,1);
    Ntotal = (outPutmoles_S.Cu(nPoints,:)+outPutmoles_S.Cu2O(nPoints,:)+
    outPutmoles_S.CuO(nPoints,:));
    plot(shell.xi(1:nVols)*particle.diam/2*10^-4,outPutmoles_S.Cu(nPoints,:)./Ntotal,'-ob','DisplayName',['Cu(t) num2str(i)'],'lineWidth',3)
    plot(shell.xi(1:nVols)*particle.diam/2*10^-4,outPutmoles_S.Cu2O(nPoints,:)./Ntotal,'-or','DisplayName',['Cu_2O(t) num2str(i)'],'lineWidth',3)
    plot(shell.xi(1:nVols)*particle.diam/2*10^-4,outPutmoles_S.CuO(nPoints,:)./Ntotal,'-og','DisplayName',['CuO(t) num2str(i)'],'lineWidth',3)
plot(shell.xi(1:nVols)*particle.diam/2*10^-4,outPutgas_S.O2(nPoints,:),'-ok','DisplayName','[O_2(t) t=' num2str(newTime(nPoints))],'lineWidth',3)

if i == 1
    legend toggle
end
xlim([0 particle.diam/2*10^-4+1])
ylim([0 1])

grid on
ylabel('Mole Fraction, [-]')
xlabel('Radius, [m]')
currFrameMF(i) = getframe;
end
hold off

% check solution to the diffusion equation...
for i = 1:length(newTime)
    vars = outPut(i,:);
    vars = [outPutgas_S.O2(i,:),outPutmoles_S.Cu(i,:)/particleV.tot.CuMoles(i),
            outPutmoles_S.Cu2O(i,:)/particleV.tot.CuMoles(i),...
            outPutmoles_S.CuO(i,:)/particleV.tot.CuMoles(i)];
    dy = SpecsCalcOxSdndt_ODSFT(time(i),vars,particleV,particle,shell,sys,D_eff,
            impur,deltaGCuO,dETAdrCuO,dETAdrCu2O);
    dO2dt(i,:) = dy(1:nVols)*(sys.presTot/(8.314*sys.temp*100^3));
    dCudt(i,:) = dy(nVols+(1:nVols)).*particleV.tot.CuMoles;
    dCu2Odt(i,:) = dy(nVols*2+(1:nVols)).*particleV.tot.CuMoles;
end

dCuOdt = -dCudt - 2*dCu2Odt;
RR_O2 = (dCuOdt + dCu2Odt)/2;
Vel = zeros(length(newTime),nVols);
for i = 1:length(newTime)
    RR_Term(i,:) = RR_O2(i,:).*shell.Me.numb./(shell.vol)*1.6702; % moles/s x num / cm^3 void space, moles/cm^3-s
    % RR_Term(i,:) = RR_O2(i,:).*shell.Me.numb./(shell.vol.*shell.porosity); % moles/s x num / cm^3 void space, moles/cm^3-s
    % RR_Term(i,:) = RR_O2(i,:).*shell.Me.numb./(shell.vol); % moles/s x num / cm
    % ~3 void space, moles/cm^3-s
% calculate stefan velocity profile...
integralTerm = -flipud(cumtrapz(flipud(shell.radCm),flipud(shell.radCm.^2.*RR_Term(i,:')))); % define at tcenter of each volume

Vel(i,:) = 8.314*sys.temp*100^3./(sys.presTot*shell.radCm.^2).*integralTerm; % cm/s
end

% determine flux of oxygen into the particle.
% how much oxygen do all particles need?
D_eff.O2 = DeffFuncOx(particle,shell); % cm^2/s
DBulk = DBulkOx(sys.temp,sys.presO2) * 100^2; % cm^2/s
externalArea = 4*pi*(particle.diam/2)^2; % cm^2
ConcO2_inf = sys.presO2_atm*sys.presTot/(Cn.gasconst*sys.temp*100^3);

gamma = 1; % change in volume from reaction (e.g. change of moles in gas
  phase if T/P fixed) per reactant
SHNum = 2; % limit of small particles, convective mass transport to
  diffusive transport, k_eff = L / D_bulk
betaTerm = SHNum * DBulk / particle.diam * sys.presTot/(gamma*8.314*sys.temp
  *100^3); % moles/cm^2-s

XO2_s = zeros(length(newTime),1);
for i = 1:length(newTime)
  XO2_s(i) = (1 - exp(sum(RR_O2(i,:)*shell.Me.numb)./(pi*particle.diam^2*
    betaTerm)))/(1 - gamma*sys.presO2_atm))/gamma;
end

ConcO2_s = XO2_s*sys.presTot/(8.314*sys.temp*100^3); % moles/cm^3

JBL = betaTerm * log((1 - gamma*XO2_s)/(1 - gamma*sys.presO2_atm)); % moles/m^2-s, flux of oxygen to surface of the particle

for ii = 1:length(newTime)
  dXO2dr = dxCalc(outPutgas_S.O2(ii,:),shell.radius(1:nVols))'; % dXO2/dr
  JSurf2(ii,:) = (-D_eff.O2.*dXO2dr + Vel(ii,:)'*outPutgas_S.O2(ii,:)'*/
    sys.presTot/(Cn.gasconst*sys.temp*100^3));
  nDotSF(ii,:) = Vel(ii,:)'*outPutgas_S.O2(ii,:)'*sys.presTot/(Cn.gasconst
    *sys.temp*100^3)*externalArea*2;
  nDotFX(ii,:) = -D_eff.O2.*dXO2dr*sys.presTot/(Cn.gasconst*sys.temp
    *100^3)*externalArea*2;
  nDotBL(ii,1) = JBL(ii)*externalArea*2; % moles of oxygen/s
  nDotO2Part(ii,:) = JSurf2(ii,:)'*4*pi.*shell.radCm.^2*2; % moles of
    oxygen/s vs moles of O2
moles_particles(ii) = ((outPutmoles_S.Cu2O(ii,:)+outPutmoles_S.CuO(ii,:)) - (outPutmoles_S.Cu2O(1,:)+outPutmoles_S.CuO(1,:))*shell.Me.numb);
% moles of oxygen in the system
poreOxy(ii) = sum(shell.vol.*shell.porosity*outPutgas_S.O2(ii,:)*sys.presTot/(Cn.gasconst*sys.temp*100^3)); % moles in pore volume as func of time
end

% firstShell = ((outPutmoles_S.Cu2O(ii,1)+outPutmoles_S.CuO(ii,1) - (outPutmoles_S.Cu2O(1,1)+outPutmoles_S.CuO(1,1)))*shell.Me.numb(1)
% shell.vol.*shell.porosity*outPutgas_S.O2(ii,:)*sys.presTot/(8.314*sys.temp*100^3)
figure(4)
hold on
plot(newTime,cumtrapz(newTime,nDotBL)/sum(shell.Me.CuMoles),'r','DisplayName','BL') % moles
% plot(newTime,cumtrapz(newTime,-nDotSF(:,1))/sum(particleV.tot.CuMoles),'k','DisplayName','SF term') % moles
% plot(newTime,cumtrapz(newTime,-nDotFX(:,1))/sum(particleV.tot.CuMoles),'k','DisplayName','dC/dr term') % moles
% plot(newTime,cumtrapz(newTime,-nDotO2Part(:,1))/sum(shell.Me.CuMoles),'k','DisplayName','Flux: SF+dC/dr term') % moles
% plot(newTime,cumtrapz(newTime,-nDotSF(:,1)-nDotFX(:,1))/sum(shell.Me.CuMoles)','k','DisplayName','Flux: SF+dC/dr term') % moles
plot(newTime,moles_particles/sum(shell.Me.CuMoles),'DisplayName','CuO + Cu2O Moles')
plot(newTime,poreOxy/sum(shell.Me.CuMoles),'DisplayName','Pore Oxy')
xlabel('Time, [seconds]')
ylabel('Moles of Oxygen to total moles of Oxygen')
grid on

% ratios...
[moles_particles./cumtrapz(newTime,nDotBL) moles_particles./cumtrapz(newTime,-nDotSF(:,1)-nDotFX(:,1)) cumtrapz(newTime,-nDotSF(:,1)-nDotFX(:,1))./cumtrapz(newTime,nDotBL)]
figure(5)
hold on
plot(newTime,JBL,'DisplayName','BL')
plot(newTime,-JSurf2(:,1),'r','DisplayName','Flux O2')
xlabel('Time, [seconds]')
ylabel('Flux O2')
grid on
figure(6)
hold on
plot(newTime,cumtrapz(newTime,nDotBL),'r',DisplayName,'BL') % moles
plot(newTime,cumtrapz(newTime,-nDotSF(:,i))/sum(particleV.tot.CuMoles),'k ' --> ' DisplayName', 'SF term') % moles
plot(newTime,cumtrapz(newTime,-nDotFX(:,i))/sum(particleV.tot.CuMoles),'k ' --> ' DisplayName', 'dCdr term') % moles
plot(newTime,cumtrapz(newTime,-nDotSF(:,i)-nDotFX(:,i)),'k ',DisplayName,' Flux: SF+dCdr term') % moles
plot(newTime,moles_particles,DisplayName,'CuO+Cu2O Moles')
xlabel('Time, [seconds]')
ylabel('Moles of Oxygen to total moles of Oxygen')
grid on

break

% plot oxide layer thickness vs time...
figure(8)
for i = 1:length(newTime)
    plot(shell.xi(1:nVols)*particle.diam/2*10^-4,thickOut.Cu(i,:)*10^-4,'*b','
    lineWidth',3,'DisplayName','Cu thickness [\mum]')
hold on
    plot(shell.xi(1:nVols)*particle.diam/2*10^-4,thickOut.Cu2O(i,:)*10^-4,'*r','
    lineWidth',3,'DisplayName','Cu2O thickness [\mum]')
    plot(shell.xi(1:nVols)*particle.diam/2*10^-4,thickOut.CuO(i,:)*10^-4,'*g','
    lineWidth',3,'DisplayName','CuO thickness [\mum]')
    plot(shell.xi(1:nVols)*particle.diam/2*10^-4,outPutgas_S.O2(i,:),'k','
    lineWidth',3)
hold off
    xlim([0 particle.diam/2*10^-4])
ylim([0 particle.Me.diam*10^-4])
xlabel('time, min')
ylabel('thickness \mum')
grid
text(20,particle.diam*10^-4*0.5,['time = ' num2str(newTime(i))])
legend toggle
legend('Location','NorthWest ')
xlabel('time, min')
ylabel('thickness \mum')
if newTime(i)>0
    pause
end
drawnow
end
% conversion of individual particle vs time...
figure(8)
xlabel('time, min')
ylabel('Conversion, [-]')
grid

for i = 1:length(newTime)
    xi = (outPutmoles_S.Cu2O(i,:)+outPutmoles_S.CuO(i,:))*MW.O/(CuPerParticle
    plot(shell.xi(1:nVols)*particle.diam/2*10^-4,xi,'*-g','lineWidth',3')

xlim([0 particle.diam/2*10^-4])
ylim([0 1.06])
text(20,0.4,['time = ' num2str(newTime(i))])
drawnow
end

figure
plot(shell.radius(1:nVols)*10^-4,shell.Me.numb)
hold on
for i = 1:nVols
    plot(shell.radius(i)*10^-4*[1 1],[0 1500],'k')
end

C.2.2 Initialization Function

This is the initiation code for the cermet particle model.

% date created: 1/30/13
%
% initialization for particle model
% This m-file loads molecular weights and counter for solving kinetics
% integration
% Oxygen diffusion through pores...
% ************************************
% Define global variables, used in SpeciesSolverCalculations
% ************************************
global tRange nPhase Cn
C.2. CERMET PARTICLE MODEL

```matlab
% global MW rho nVols manufactMethod
% global sys
% global Dx shapeFactor
% global presO2Func
% global gasDiffOff reactOff

gasDiffOff = true; % gas diffusion off
reactOff  = false;
% manufactMethod = 'granulation';
manufactMethod = 'CORE';
% manufactMethod = 'wet';
shapeFactor =3; % always a sphere... don't change

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * initialize variables * * * * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
MWs % -- molecular weight structured array
Cn.gasconst = 8.314; % J/mole-K
Cn.Na = 6.022e23; % atoms/mole
Cn.kB = Cn.gasconst/Cn.Na; % J/atom-K
Cn.kB_eV = 8.617332478*10^-5; % eV/K
Cn.hPlank_eV = 4.13566751691*10^-15; % eV-s
Cn.hPlank_JS = 6.6260695729*10^-34; % J-s
Cn.e = 1.60217646e-19; % coulumb/electron...

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% temperature / pressure Option * * * * * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
sys.presTot = 101325; % Pa
sys.presO2 = .15*101325; % Pa
sys.presN2 = sys.presTot - sys.presO2;
sys.presO2_atm = sys.presO2 /101325; % Atm
sys.temp = 900+273.15; % Kelvin

tRange.endTime = 30*60; % end time of experiment, seconds
% end time of experiment,
% duration of early time, -1 if no time stepping
% duration of long time step
% time step selection
% small time steps at the end,
% large time steps at the beginning
```
if tRange.timeEARL > 0
  numEarlyTimeSteps = 20;
  numLateTimeSteps = 20;
  fullSteps = numEarlyTimeSteps + numLateTimeSteps; % number of steps
  tRange.timeStep = zeros(fullSteps + 1, 1);
  tRange.timeStepS = zeros(fullSteps + 1, 1);
  for jj = 2:(length(tRange.timeStep))
    if jj <= numEarlyTimeSteps + 1
      tRange.timeStep(jj) = tRange.timeEARL / numEarlyTimeSteps;
      tRange.timeStepS(jj) = tRange.timeEARL / numEarlyTimeSteps +
      tRange.timeStepS(jj - 1);
    elseif jj <= numLateTimeSteps + numEarlyTimeSteps + 1
      tRange.timeStep(jj) = tRange.timeLATE / numLateTimeSteps;
      tRange.timeStepS(jj) = tRange.timeLATE / numLateTimeSteps +
      tRange.timeStepS(jj - 1);
    end
  end
  clear tempTimeLrgStep tempnLrgStep tempnSmlStep fullSteps finalTimeStep;
else
  tRange.timeStep = [0 tRange.endTime];
  tRange.timeStepS = [0 tRange.endTime];
end

  tRange.timeStep = [0; 1e-6; tRange.timeStep(2:end)];
  tRange.timeStepS = [0; 1e-6; (tRange.timeStepS(2:end)+1e-6)];
% --------------------------------------------------------------------------
% ---------------- particle characteristics ---------------------------------
% --------------------------------------------------------------------------
  rho.Cu0 = 6.31; % g/cm^3 * 100^3 cm^3/m^3 [=] g/m^3
  rho.Cu2O = 6.0; % g/cm^3 * 100^3 cm^3/m^3
  rho.Cu = 8.94; % g/cm^3 * 100^3 cm^3/m^3
  rho.sub = 5.62; % g/cm^3 ... true density of ZrO2
% --------------------------------------------------------------------------
% equilibrium interface pressure....
% --------------------------------------------------------------------------
sys.presEqu = phasePressure(sys.temp, sys.presTot);
sys.presCuCu2O_atm = sys.presEqu(1)/101325;
sys.presCu2OCuO_atm = sys.presEqu(2)/101325;
% determine number of phases....
if sys.presO2 > sys.presEqu(1) && sys.presO2 < sys.presEqu(2)
  nPhase = 2; % for only Cu & Cu2O..
  isCuO = 0;
elseif sys.presO2 > sys.presEqu(2)
  isCuO = 1;
  nPhase = 3; % for only Cu & Cu2O..
else

% pressure too low, no oxidation...

isCuO = 0;
nPhase = 1;

% presO2Vect = [logspace(log10(sys.presCu2O_atm*1.01),log10(sys.presO2_atm),115) ones(1,205)*sys.presO2_atm];
presO2Func = @(t) sys.presO2_atm*ones(length(t),1);

% presO2Func = @(t) -0.21 * (-1+exp((-t/20)) + sys.presCu2O_atm*1.1;
% presO2Func = @(t) 0.21./(1+9522.8*exp(-t/2)) + sys.presCu2O_atm*1.1;
% presO2Func = @(t) sys.presO2_atm./(1+(sys.presO2_atm - sys.presCu2O_atm*1.1)).*exp(-t*3.1*sys.temp/1073));

% figure(4); hold on;

% plot(tRange.timeStepS,presO2Func(tRange.timeStepS),'.r')
% plot([1e-5 1],sys.presCu2O_atm*[1 1],'g')
% plot([3 6],sys.presCu2O_atm*[1 1],'g')
% plot([1 10],sys.presO2_atm*[1 1],'g')
% ------------------------------------------------------------------------
% ------------- Overall Particle Properties -------------------------------
% ------------------------------------------------------------------------

% particle.diam = 100e-6*100; % cm (size dist. 90 to 125)
% particle.pcntCuO = 0.2; % weight percent of Cu in particle... incudes porosity of grains...
% particle.pcntCu = particle.pcntCuO * MW.Cu / MW.CuO /( particle.pcntCuO * MW.Cu / (1 - particle.pcntCu0));
% particle.vol = 1/6 * pi * particle.diam^3;

% particle.porosity = 0.4; % V_void/V_actual... from packing of Cu & ZrO2 particles
% particle.Sg = 0.7*100^-2; % cm^-2 / g
% particle.rhoT = (particle.pcntCu/rho.Cu + (1-particle.pcntCu)/rho.sub)^-1;
% particle.rhoA = particle.rhoT*(1-particle.porosity);
% particle.SRF = 1; % surface roughness factor
% particle.tortuosity = 3; % particle tortuosity, randomly orineted spheres
% particle.ShNum = 2; % sherwood number..
% particle.Nu = 2; % h*D/k

% ------------------------------------------------------------------------
% Particle Type-- Wet Impregnation or Spray Granulation
% ------------------------------------------------------------------------

if strcmp(manufactMethod,'granulation') || strcmp(manufactMethod,'CORE')
    particle.sub.diam = 1e-6*100; % cm, characteristic diameter
    particle.sub.Sg = 6/(rho.sub*particle.sub.diam); % cm^-2/g... just for the substrate
% particle.Me.diam = 10e^-6*100; % cm, characteristic diameter
% particle.Me.diam = (6/pi*((1/6*pi*particle.diam^3 * particle.rhoA)*
  particle.pcntCu / rho.Cu))^(1/3); % cm
particle.Me.Sg = 6/(rho.Cu*particle.Me.diam); % cm^-2/g... just for the Cu

% overlap of area...
particle.SgOverlap = (particle.Me.Sg*particle.pcntCu + (1-particle.pcntCu) *
  particle.sub.Sg) - particle.Sg;

impur.Cu2O.pos = 0;
impur.Cu2O.neg = 0;
impur.CuO.pos = 0;
impur.CuO.neg = 0;
deltaGCuO.vac = (250*1000 - (-39*sys.temp)); % J/mol **
deltaGCuO.int = (260*1000 - (75*sys.temp)); % J/mol **
aSphere =1; aPlane = 0;

elseif strcmp (manufactMethod, 'wet')
  % % % particle.sub.diam = particle.diam; % cm, characteristic diameter of
  % % % substrate
  % % % particle.sub.Sg = 30*100^2; % cm^-2/g... just for the substrate
  % % % particle.sub.porosity = 0.5; % V_void/V_actual... of substrate --> NOT
  % % % SURE IF THIS IS NEEDED
  % % % particle.sub.Area = particle.sub.mass*particle.sub.Sg; % cm^-@
  % %
  % % % particle.fracOfCoverage = 0.3; % 0 to 1 -- how much of the substrate
  % % % surf. is covered w Cu
  % % % particle.DtoT = 1; % diam of Cu on surf to thickness ratio
  % % % particle.Me.Area = particle.sub.Area * particle.fracOfCoverage; % cm^-2
  % % % particle.Me.mass = particle.sub.mass / (1-particle.pcntCu)*
  % % % particle.pcntCu;
  % % % particle.Me.vol = particle.Me.mass/rho.Cu;
  % % % particle.Me.diam = particle.Me.vol/particle.Me.Area; % cm, thickness
  % of Cu layer on Al2O3
  % particle.gbWidth = 10e^-9; % cm, grain boundary
  % particle.gbNum = 10; % boundaries per m^-2
  aSphere =0 ; aPlane = 1;
end

nVols = 100; % -- works for 800 C
delta = 1/(nVols);
smallEnd = [(delta/2):delta:(delta*(nVols))]*particle.diam/2;
C.2. CERMET PARTICLE MODEL

```matlab
shell.radCm = flipud(smallerEnd(1:(nVols))); % loc of shell centers
shell.radius = [particle.diam/2; ([shell.radCm(1:nVols-1)] + [shell.radCm(2:
    nVols)])/2;0];
shell.vol = 4/3*pi*(shell.radius(1:nVols).^3-shell.radius(2:(nVols+1)).^3); %
    cm^-3
shell.xi = shell.radCm/(particle.diam/2); % location of center node...

% figure
% plot(shell.radius*10^4, ones(nVols+1,1), 'o-')
% hold on%
% plot(shell.radCm*10^4, ones(nVols,1), 'ko-')
% plot(particle.diam/2*10^4,1, 'o-r')
% plot(shell.radius(nVols+1)/2*10^4,1, 'o-r')
% xlabel('Grid Location, \mu m')
% grid on

if strcmp(manufactMethod,'granulation')
    % volume of Cu to Cu and ZrO2 volume...
    volPcntCu = (particle.pcntCu / rho.Cu)/(particle.pcntCu / rho.Cu + (1-
        particle.pcntCu)/ rho.sub); % x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-

    % particleV structure represents indivdiual Cu particle in a given shell
    % radius of individual metal particles
    particleV.Cu.thick = particle.Me.diam/2 * ones(nVols,1); % cm, initial
        thickness
    particleV.CuO.thick = particle.Me.diam/2/100 * ones(nVols,1); % cm,
        thickness of CuO layer
    particleV.Cu2O.thick = particle.Me.diam/2/100 * ones(nVols,1); % cm,
        thickness of Cu2O layer

    % mass of an indivudal spehre at radius r
    particleV.sub.mass = 4/3 * pi * (particle.sub.diam/2)^3 * rho.sub*ones(
        nVols,1);
    particleV.Cu.mass = 4/3 * pi * particleV.Cu.thick.^3 * rho.Cu; % initial
        mass of Cu particle
    particleV.tot.CuMoles2 = particleV.Cu.mass / MW.Cu;

    % what is the new thickness of Cu with this little reaction?
    f = @(rT) rho.Cu/MW.Cu*rT^3 + 2*rho.Cu2O/MW.Cu2O*(((rT+...
        particleV.Cu2O.thick(1)).^3 - rT.^3) + ...
        rho.Cu0/MW.Cu0 * (((rT+particleV.Cu2O.thick(1)+
    rT)*particleV.Cu2O.thick(1)).^3 - (rT+particleV.Cu2O.thick(1)).
        ^3)) - ...
```

180  shell.radCm = flipud(smallerEnd(1:(nVols))); % loc of shell centers
181  shell.radius = [particle.diam/2; ([shell.radCm(1:nVols-1)] + [shell.radCm(2:
182      nVols)])/2;0];
183  shell.vol = 4/3*pi*(shell.radius(1:nVols).^3-shell.radius(2:(nVols+1)).^3); %
184      cm^-3
185  shell.xi = shell.radCm/(particle.diam/2); % location of center node...
186  % figure
187  % plot(shell.radius*10^4, ones(nVols+1,1), 'o-')
188  % hold on
189  % plot(shell.radCm*10^4, ones(nVols,1), 'ko-')
190  % plot(particle.diam/2*10^4,1, 'o-r')
191  % xlabel('Grid Location, \mu m')
192  % grid on
193  
194  if strcmp(manufactMethod,'granulation')
195    % volume of Cu to Cu and ZrO2 volume...
196    volPcntCu = (particle.pcntCu / rho.Cu)/(particle.pcntCu / rho.Cu + (1-
197        particle.pcntCu)/ rho.sub); % x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-
198    
199    % particleV structure represents indivdiual Cu particle in a given shell
200    % radius of individual metal particles
201    particleV.Cu.thick = particle.Me.diam/2 * ones(nVols,1); % cm, initial
202        thickness
203    particleV.CuO.thick = particle.Me.diam/2/100 * ones(nVols,1); % cm,
204        thickness of CuO layer
205    particleV.Cu2O.thick = particle.Me.diam/2/100 * ones(nVols,1); % cm,
206        thickness of Cu2O layer
207    
208    % mass of an indivudal spehre at radius r
209    particleV.sub.mass = 4/3 * pi * (particle.sub.diam/2)^3 * rho.sub*ones(
210        nVols,1);
211    particleV.Cu.mass = 4/3 * pi * particleV.Cu.thick.^3 * rho.Cu; % initial
212        mass of Cu particle
213    particleV.tot.CuMoles2 = particleV.Cu.mass / MW.Cu;
214    
215    % what is the new thickness of Cu with this little reaction?
216    f = @(rT) rho.Cu/MW.Cu*rT^3 + 2*rho.Cu2O/MW.Cu2O*(((rT+...
217        particleV.Cu2O.thick(1)).^3 - rT.^3) + ...
218        rho.Cu0/MW.Cu0 * (((rT+particleV.Cu2O.thick(1)+
219            rT)*particleV.Cu2O.thick(1)).^3 - (rT+particleV.Cu2O.thick(1)).
220            ^3)) - ...
APPENDIX C. MATLAB CODES

particleV.Cu.mass(1) / MW.Cu * 3/(4*pi);  
particleV.Cu.thick = fzero(f, particle.Me.diam/2)*ones(nVols,1);  
particleV.Cu.mass = 4/3 * pi * particleV.Cu.thick.^3 * rho.Cu; % UPDATE with new diameter...

% add oxide layer, take Cu atoms from Core  
particleV.Cu2O.mass = 4/3 * pi * (((particleV.Cu.thick +  
particleV.Cu2O.thick).^3 - particleV.Cu.thick.^3) * rho.Cu2O;  
particleV.CuO.mass = 4/3 * pi * (((particleV.Cu.thick +  
particleV.Cu2O.thick + particleV.CuO.thick).^3 - (particleV.Cu.thick  
+ particleV.Cu2O.thick).^3) * rho.CuO;

% particleV.sub.vol = particleV.sub.mass. /(rho.sub *(1-  
particle.sub.porosity));  
particleV.Cu.vol = (particleV.Cu.mass. / rho.Cu);  
particleV.Cu2O.vol = (particleV.Cu2O.mass. / rho.Cu2O);  
particleV.CuO.vol = (particleV.CuO.mass. / rho.CuO);

% tot [=] individul Cu/Cu2O/CuO particle  
particleV.tot.mass = particleV.Cu.mass+particleV.Cu2O.mass+  
particleV.CuO.mass; % individual Cu/Cu2O/CuO particle mass  
particleV.tot.diam = (particleV.Cu.thick+particleV.Cu2O.thick+  
particleV.CuO.thick)*2; % cm, diameter of an individul particle\  
particleV.tot.vol = particleV.Cu.vol + particleV.Cu2O.vol +  
particleV.CuO.vol; % cm^3, volume of an individual Cu particle  
particleV.tot.rho = particleV.tot.mass./particleV.tot.vol; % g/cc,  
density of individul particle

particleV.tot.Sg = 6./(particleV.tot.rho.*particleV.tot.diam); % m^{-2}/g  
particleV.tot.CuMoles = particleV.Cu.mass/MW.Cu+2*particleV.Cu2O.mass/  
MW.Cu2O+particleV.CuO.mass/MW.CuO;

% initConv =-sum(particleV.tot.CuMoles*MW.Cu-particleV.tot.mass)./sum(  
particleV.tot.CuMoles*MW.O)  

% DEFINE effective properties of the shell  
% just consider external area of the two particles since internal area in the  
shell.Me.numb = ((shell.vol*(1-particle.porosity)*volPcntCu)./(4/3 * pi  
* (particle.Me.diam/2).^3)); % number of Cu particles per shell  
shell.Me.mass = shell.Me.numb.* particleV.tot.CuMoles*MW.Cu; % grams,  
collective particles in shell i  
shell.Me.CuMoles= shell.Me.numb.* particleV.tot.CuMoles; % total moles of  
Cu per shell
C.2. CERMET PARTICLE MODEL

\[
\text{shell.sub.mass} = \text{shell.Me.mass} / \text{particle.pcntCu} \times (1 - \text{particle.pcntCu}); \quad \text{grams, collective particles in shell i}
\]

\[
\text{shell.Sg} = \frac{\text{shell.sub.mass} \times \text{particle.sub.Sg} + \text{particleV.tot.Sg} \times \text{shell.Me.mass}}{\text{shell.sub.mass} + \text{shell.Me.mass}}; \quad \% \text{g/cm}^2 \text{ of total A/g of particle... measured for the particle...}
\]

\[
\text{shell.Sg2} = \frac{\text{shell.sub.mass} \times \text{particle.sub.SgEx} + \text{particleV.tot.Sg} \times \text{shell.Me.mass}}{\text{shell.sub.mass} + \text{shell.Me.mass}}; \quad \% \text{g/cm}^2 \text{ of total A/g of particle... measured for the particle...}
\]

\[
\text{shell.porosity} = \frac{\text{shell.vol} - \text{shell.Me.mass} / \rho.\text{Cu} - \text{shell.sub.mass} / \rho.\text{sub}}{\text{shell.vol}}; \quad \% \text{include porosity of substrate}
\]

\[
\text{shell.rhoT} = \frac{\text{shell.Me.mass} + \text{shell.sub.mass}}{\text{shell.vol} \times (1 - \text{shell.porosity})}; \quad \% \text{g/cc, true density for the overall particle}
\]

\[
\text{shell.rhoA} = \frac{\text{shell.Me.mass} + \text{shell.sub.mass}}{\text{shell.vol}}; \quad \% \text{g/cc, apparent density for the overall particle}
\]

\[
\text{elseif} \ \text{strcmp(manufactMethod, 'CORE')}
\]

\[
\delta = 1 / (\text{nVols})
\]

\[
\text{smallEnd} = ((\delta / 2):\delta:.\delta \times \text{nVols}) \times (\text{particle.diam} / 2 - \text{particle.Me.diam} / 2) + \text{particle.Me.diam} / 2;
\]

\[
\text{shell.radCm} = \text{flipud}(\text{smallEnd}(1:(\text{nVols}))); \quad \% \text{loc of shell centers}
\]

\[
\text{shell.radius} = [\text{particle.diam} / 2; ([\text{shell.radCm}(1: \text{nVols} - 1)] + [\text{shell.radCm}(2: \text{nVols})] / 2; \text{particle.Me.diam} / 2);
\]

\[
\text{shell.vol} = 4 / 3 \times \pi \times (\text{shell.radius}(1: \text{nVols})^3 - \text{shell.radius}(2: (\text{nVols} + 1)) - 3); \quad \% \text{cm}^3
\]

\[
\text{shell.xi} = \text{shell.radCm} / (\text{particle.diam} / 2); \quad \% \text{location of center node...}
\]

\[
\text{particleV.Cu.thick} = \text{particle.Me.diam} / 2; \quad \% \text{cm, initial thickness}
\]

\[
\text{particleV.CuO.thick} = \text{particle.Me.diam} / 2 / 1000; \quad \% \text{cm, thickness of CuO layer}
\]

\[
\text{particleV.Cu2O.thick} = \text{particle.Me.diam} / 2 / 1000; \quad \% \text{cm, thickness of Cu}_2\text{O layer}
\]

\[
\text{particleV.sub.mass} = 4 / 3 \times \pi \times (\text{particle.sub.diam} / 2)^3 \times \rho.\text{sub};
\]

\[
\text{particleV.Cu.mass} = 4 / 3 \times \pi \times \text{particleV.Cu.thick}^3 \times \rho.\text{Cu}; \quad \% \text{initial mass of Cu particle}
\]

\[
\text{particleV.tot.CuMoles2} = \text{particleV.Cu.mass} / \text{MW.Cu};
\]

\[
\% \text{what is the new thickness of Cu with this little reaction?}
\]

\[
f = \Phi(rT) \times \rho.\text{Cu} / \text{MW.Cu} \times \Phi(rT) + 2 \times \rho.\text{Cu2O} / \text{MW.Cu2O} \times ((rT) + \text{particleV.Cu2O.thick}(1))^3 - rT^3) + ...
\]
rho.CuO / MW.Cu * ((( rT+ particleV.Cu2O.thick(1) + 
particleV.Cu.thick(1) ).^3 - (rT+ particleV.Cu2O.thick(1)).^3) - ... 
particleV.Cu.mass(1)/MW.Cu * 3/(4*pi);

particleV.Cu.thick = fzero(f, particle.Me.diam/2);

particleV.Cu2O.mass = 4/3 * pi * particleV.Cu.thick.^3 * rho.Cu;

particleV.Cu2O.mass = 4/3 * pi * ((particleV.Cu.thick +
particleV.Cu2O.thick).^3 - particleV.Cu.thick.^3) * rho.Cu2O;

particleV.CuO.mass = 4/3 * pi * ((particleV.Cu.thick +
particleV.Cu2O.thick + particleV.CuO.thick).^3 - (particleV.Cu.thick +
particleV.Cu2O.thick + particleV.CuO.thick).^3) * rho.CuO;

% particleV.sub.vol = particleV.sub.mass. /( rho.sub *(1 -
particle.sub.porosity));

particleV.Cu.vol = ( particleV.Cu.mass. / rho.Cu );

particleV.Cu2O.vol = ( particleV.Cu2O.mass. / rho.Cu2O );

particleV.CuO.vol = ( particleV.CuO.mass. / rho.CuO );

% tot [] indivudal Cu/Cu2O/CuO particle

particleV.tot.mass = particleV.Cu.mass + particleV.Cu2O.mass +
particleV.CuO.mass; % individual Cu/Cu2O/CuO particle mass

particleV.tot.diam = (particleV.Cu.thick+particleV.Cu2O.thick+ 
particleV.CuO.thick)*2; % cm, diameter of an indivudal particle

particleV.tot.vol = particleV.Cu.vol + particleV.Cu2O.vol +
particleV.CuO.vol; % cm^3, volume of an individual Cu particle

particleV.tot.rho = particleV.tot.mass./particleV.tot.vol; % g/cc, 
density of indivudal particle

particleV.tot.Sg = 6./(particleV.tot.rho.*particleV.tot.diam); % m^-2/g

particleV.tot.CuMoles = particleV.Cu.mass / MW.Cu +2* particleV.Cu2O.mass /
MW.Cu2O + particleV.CuO.mass / MW.CuO;

% initConv =-sum( particleV.tot.CuMoles*MW.Cu - particleV.tot.mass )./sum( 
particleV.tot.CuMoles*MW.0)

% DEFINE effective properties of the shell
% just consider external area of the two particles since internal area in the
% ZrO2 is not being considered..

shell.Me.numb = 1; % number of Cu particles per shell

shell.Me.mass = shell.Me.numb.* particleV.tot.CuMoles*MW.Cu; % grams,
collective particles in shell i

shell.Me.CuMoles= shell.Me.numb.* particleV.tot.CuMoles; % total moles of 
Cu per shell

shell.sub.mass = shell.vol * rho.sub*(1-particle.porosity); % grams,
collective particles in shell i
C.2. CERMET PARTICLE MODEL

% calculate properties of particle

\[
\text{particle.pcntCu} = \frac{\text{shell.Me.mass}}{\text{shell.Me.mass} + \text{sum(shell.sub.mass)}};
\]

\[
\text{shell.Sg} = \frac{\text{shell.sub.mass} \times \text{particle.sub.Sg}}{\text{shell.sub.mass}}; \quad \text{g/cm}^2 \quad \text{of total A/g of particle... measured for the particle...}
\]

\[
\text{shell.Sg} = \frac{\text{shell.sub.mass} \times \text{particle.sub.Sg} + \text{particleV.tot.Sg} \times \text{shell.Me.mass}}{\text{shell.sub.mass} + \text{shell.Me.mass}}; \quad \text{g/cm}^2 \quad \text{of total A/g of particle... measured for the particle...}
\]

\[
\text{shell.porosity} = \frac{\text{shell.vol} - \text{shell.sub.mass} / \rho_{\text{sub}}}{\text{shell.vol}}; \quad \% \quad \text{include porosity of substrate}
\]

\[
\text{shell.rhoT} = \frac{\text{shell.sub.mass}}{\text{shell.vol} \times (1 - \text{shell.porosity})}; \quad \text{g/cc, true density for the overall particle}
\]

\[
\text{shell.rhoA} = \frac{\text{shell.sub.mass}}{\text{shell.vol}}; \quad \% \quad \text{g/cc, apparent density for the overall particle}
\]

\[
\text{particle.rhoT} = (\text{particle.pcntCu} / \rho_{\text{Cu}} + (1 - \text{particle.pcntCu}) / \rho_{\text{sub}})^{-1};
\]

\[
\text{particle.rhoA} = \text{particle.rhoT} \times (1 - \text{particle.porosity});
\]

\[
\text{rPore} = \frac{2 \times \text{particle.SRF} \times \text{particle.porosity} \times (1 - \text{particle.porosity})}{\text{shell.rhoA} \times \text{particle.Sg} \times 10^4}; \quad \% \quad \text{cm}^3/\text{cm}^2 \quad \times \text{10}^4 \quad \text{micron/cm} = \text{micron}
\]

\[
\text{elseif strcmp(manufactMethod,'wet')}
\]

\[
\text{error('wrong manufacturing method... ')}
\]

% ------------------------------- DDE solver initial conditions -------------------------------
% ---

% gas, Cu moles, Cu2O moles, CuO moles

\[
\text{y_init} = \left[ \begin{array}{c}
\text{ones(nVols,1) \times (sys.presCuCu2O_atm \times 0.99)}; [\text{particleV.Cu.mass} / \text{MW.Cu}]; \\
\text{particleV.Cu2O.mass} / \text{MW.Cu2O}; \\
\text{particleV.CuO.mass} / \text{MW.CuO}
\end{array} \right];
\]

\[
\% \quad \text{y_init} = \left[ \begin{array}{c}
\text{ones(nVols,1) \times (sys.presCuCu2O_atm \times 0.99)}; [\text{particleV.Cu.mass} / \text{MW.Cu}]; \\
\text{particleV.Cu2O.mass} / \text{MW.Cu2O}; \\
\text{particleV.CuO.mass} / \text{MW.CuO}
\end{array} \right];
\]

% CuPerParticle = [particleV.tot.CuMoles; particleV.tot.CuMoles];

\[
\text{y_init} = \left[ \begin{array}{c}
\text{zeros(nVols,1)}; [\text{particleV.Cu.mass} / \text{MW.Cu}; \\
\text{particleV.Cu2O.mass} / \text{MW.Cu2O}]
\end{array} \right]; / \text{CuPerParticle};
\]

end
pRangeCu2O = logspace(log10(sys.presCu20CuO_atm*.01),log10(sys.presCu20CuO_atm*.999),7);

pRangeCuO = (logspace(log10(sys.presCu20CuO_atm*1.05),log10(sys.presO2_atm),7))

% Turn off gas diffusion
if gasDiffOff
  % only works for P02 > sys.presCu20CuO_atm...
y_init(1:nVols) = sys.presO2_atm;
pRangeCu2O = sys.presCu20CuO_atm;
pRangeCuO = sys.presO2_atm;
end

% define defect diffusivities
Dx(1) = 1.48e-4*exp(-24*1000/(8.314*sys.temp)); % if only vacancies
Dx(2) = 1.47e-3*exp(-84.8*1000/(8.314*sys.temp)); % if only interstitials
Dx(3) = 3.47e-4*exp(-32*1000/(8.314*sys.temp));
Dx(4) = 1e-10;

% calculate d[ETA]/dr in the particles given the initial O2 partial pressure...
regSiteCu2O = defectEquilibriumFunc_Y(sys, pRangeCu2O, deltaGCuO, impur); % site ratios when all shells are 21 pcnt. oxygen
regSiteCuO = defectEquilibriumFunc_Y(sys, pRangeCuO, deltaGCuO, impur); % site ratios when all shells are 21 pcnt. oxygen

regSiteCu2O.pRange = pRangeCu2O;
regSiteCuO.pRange = pRangeCuO;

% VACANCY DIFFUSION or defect diffusion coefficients
D_eff.Cu20.vacCuX = Dx(1); % cm^2/s -- vac diffusion coef for vac x and vac
D_eff.Cu20.vacCuM = Dx(1); % cm^2/s
D_eff.Cu20.intCuX = Dx(2); % cm^2/s
D_eff.Cu20.intCuP = Dx(2); % cm^2/s
D_eff.Cu20.vacOPP = 0; % cm^2/s
D_eff.Cu20.h = DeffFuncDefect('Cu20','hole'); % cm^2/s
D_eff.Cu20.e = DeffFuncDefect('Cu20','elec'); % cm^2/s

if isinf(D_eff.Cu20.vacOPP) || isnan(D_eff.Cu20.vacOPP)
  D_eff.Cu20.vacOPP=0;
end

% self diffusion coef...
D_eff.Cu0.vacCuMM = Dx(3); % cm^2/s, make a vacancy diffusivity from the self diffusion coefficient...
D_{eff}.CuO.intCuPP = D_{x}(4); \% cm^{-2}/s
D_{eff}.CuO.vacOPP = 0; \% cm^{-2}/s; \% cm^{-2}/s
mueRat = 1; \% hole to electron mobility...
D_{eff}.CuO.h = DeffFuncDefect('CuO','hole')./(\text{regSiteCuO.h.CuOCu2O})*(\text{rho.CuO}/\rightarrow MW.CuO)*mueRat*(\text{regSiteCuO.e.CuOCu2O})*(\text{rho.CuO/MW.CuO})); \% cm^{-2}/s -- should be 'diffusivity' in terms of an electrical conductivity or \rightarrow mobility....
D_{eff}.CuO.e = D_{eff}.CuO.h*mueRat; \% cm^{-2}/s

if isinf(D_{eff}.CuO.vacOPP) || isnan(D_{eff}.CuO.vacOPP)
D_{eff}.CuO.vacOPP =0;
end
tic
if \sim reactOff
\text{dETAdrCu2O} = dcdrCalcV(\text{pRangeCu2O}, D_{eff}, \text{regSiteCu2O}, \text{impur}); \% code calculates \rightarrow dimensionless defect concentration gradients as a function of \rightarrow different oxygen pressures
\text{dETAdrCuO} = dcdrCalcV(\text{pRangeCuO}, D_{eff}, \text{regSiteCuO}, \text{impur});
\text{dETAdrCu2O}.\text{pres} = \text{pRangeCu20};
\text{dETAdrCuO}.\text{pres} = \text{pRangeCuO};
else
\text{dETAdrCu2O} = 0;
\text{dETAdrCuO} = 0;
end
disp(['d[X] calc time: ' num2str(toc) 'seconds'])

C.2.3 Oxidation Model
This function runs the oxidation model (SpecsCalcOxSndt.m).

% single particle model...
% OD [=] oxygen diffusion...
% This function solves (nShells-1) x nVar species equations and unknowns
APPENDIX C. MATLAB CODES

% initialize....
outPutTime = 0;

% =========================================================================
for jj = 1:(length(tRange.timeStep)-1)

% =========================================================================
if tRange.timeStep(jj+1) < 1 && jj == 1
    timeStep = [0 linspace((1e-8),(tRange.timeStep(jj+1)),10)]; % time range for calculation
else
    timeStep = [0 logspace(log10(1e-8),log10(tRange.timeStep(jj+1)),30)]; % time range for calculation
end

timeStep = [0 tRange.timeStep(jj+1)]; % time range for calculation
jacbScript % will select the correct jacobian pattern based on areaModel, gasOn and bulkSpec variables...

if length(tRange.timeStep) == 2

else
    if strcmp(manufactMethod,'CORE')
        sys.presO2_atm = presO2Func(tRange.timeStepS(jj));
globTime = tRange.timeStepS(jj);
        % SpecsCalcOxSdndt_OD2SFT_CORE
        [time, outPut] = ode15s(@SpecsCalcOxSdndt_ODSFT_CORE,timeStep,y_Init, options,particleV,particle,shell,sys,D_eff,impur,deltaGCuO, dETAdrCuO,dETAdrCu2O); % OD2 -- solve Cu/Cu2O and get CuO by difference
    else

        sys.presO2_atm = presO2Func(tRange.timeStepS(jj));
globTime = tRange.timeStepS(jj);
        % reactON = y_Init(nVols) > sys.presCuCu2O_atm;
        [time, outPut] = ode15s(@SpecsCalcOxSdndt_OD2SFT_D,timeStep,y_Init, options,particleV,particle,shell,sys,D_eff,impur,deltaGCuO, dETAdrCuO,dETAdrCu2O,reactON); % OD2 -- solve Cu/Cu2O and get CuO by difference
        [time, outPut] = ode15s(@SpecsCalcOxSdndt_OD2SFT,timeStep,y_Init, options,particleV,particle,shell,sys,D_eff,impur,deltaGCuO,dETAdrCuO, dETAdrCu2O); % OD2 -- solve Cu/Cu2O and get CuO by difference
        [time, outPut] = ode15s(@SpecsCalcOxSdndt_ODSFT,timeStep,y_Init, options,particleV,particle,shell,sys,D_eff,impur,deltaGCuO, dETAdrCuO,dETAdrCu2O); % OD2 -- solve Cu/Cu2O and get CuO by difference

        else

        end

end

end
C.2. CERMET PARTICLE MODEL

if jj > 30
  a = 1;
end

if jj > 100
  a = 1;
end

if jj > 104
  a = 1;
end

% =========================================================================
% Store Data
% =========================================================================
nPoints = round(length(time)/1);  % nPoints is the number of data points that → are saved

if nPoints == 1
  nPoints = 2;
end

nLoc = round(linspace(1, length(time), nPoints));
nPointsP = length(outPutTime);

outPutVar(nPointsP:(nPoints+nPointsP-1),:) = outPut(nLoc,:);
outPutTime(nPointsP:(nPoints+nPointsP-1)) = time(nLoc)+tRange.timeStepS(jj);

save('testRun3.mat','outPutTime','outPutVar');

outPutgas.O2 = (outPut(length(time),(1:nVols)));

if length(particleV.tot.CuMoles) == 1
  outPutmoles.Cu = outPut(length(time),nVols+1).*particleV.tot.CuMoles;  % moles in Cu
  outPutmoles.Cu2O = outPut(length(time),nVols+2).*particleV.tot.CuMoles;  % moles in Cu2O
  outPutmoles.CuO(i,:) = outPut(i,nVols+2*nVols+(1:nVols)).*particleV.tot.CuMoles;  % moles in CuO
else
  outPutmoles.Cu = outPut(length(time),nVols+(1:nVols)).*particleV.tot.CuMoles;  % moles in Cu
outPutmoles.Cu2O = outPut(length(time),nVols+nVols+(1:nVols))';
⇒ particleV.tot.CuMoles; % moles in Cu2O
⇒ outPutmoles.Cu2O);
%
outPutmoles.Cu0(i,:) = outPut(i,nVols+2*nVols+(1:nVols))';
⇒ particleV.tot.CuMoles; % moles in Cu2O
end

xi_time = (sum(shell.Me.numb.*(outPutmoles.Cu*MW.Cu+outPutmoles.Cu2O*MW.Cu2O+
⇒ outPutmoles.CuO*MW.CuO)) - ...
sum(shell.Me.CuMoles*MW.Cu))/sum(shell.Me.CuMoles*MW.O);

fprintf('%4g %5.5g %5.5g %5.5g %5.8g 
⇒ outPutgas.O2(nVols) toc)
end

% =========================================================================
% PlotData
% =========================================================================
if false
  clear outPutgas outPutmoles
  for i = 1:length(time)
    outPutgas.O2(i,:) = (outPut(i,(1:nVols))');
    outPutmoles.Cu(i,:) = outPut(i,nVols+(1:nVols))';
⇒ particleV.tot.CuMoles
%
    outPutmoles.Cu2O(i,:) = outPut(i,nVols+2*nVols+(1:nVols))';
⇒ particleV.tot.CuMoles; % moles in Cu2O
%
    outPutmoles.CuO(i,:) = particleV.tot.CuMoles -(outPutmoles.Cu(i,:) +
⇒ 2*outPutmoles.Cu2O(i,:));
    outPutmoles.CuO(i,:) = outPut(i,nVols+2*nVols+(1:nVols))';
⇒ particleV.tot.CuMoles; % moles in Cu2O
    xi_time(i) = (sum(shell.Me.numb'.*(outPutmoles.Cu(i,:)*MW.Cu+
⇒ outPutmoles.Cu2O(i,:)*MW.Cu2O+outPutmoles.CuO(i,:)*MW.CuO)) - ...
sum(shell.Me.CuMoles*MW.Cu))/sum(shell.Me.CuMoles*MW.O);
  end

figure(15)
for i = 1:length(time)
  Ntotal = (outPutmoles.Cu(i,:)+outPutmoles.Cu2O(i,:)+outPutmoles.Cu0(i,:))
⇒ ;
  plot(shell.xi(1:nVols)*particle.diam/2*10^-4,outPutmoles.Cu(i,:))'/Ntotal,
⇒ '-ob','DisplayName','Cu(t)','LineWidth',3)
  hold on
  plot(shell.xi(1:nVols)*particle.diam/2*10^-4,outPutmoles.Cu2O(i,:))'/
⇒ Ntotal,'-or','DisplayName','Cu_2O(t)','LineWidth',3)
  plot(shell.xi(1:nVols)*particle.diam/2*10^-4,outPutmoles.CuO(i,:))'/Ntotal
⇒ '-og','DisplayName','CuO(t)','LineWidth',3)
C.2. CERMET PARTICLE MODEL

108 plot(shell.xi(1:nVols)*particle.diam/2*10^-4, outPutgas.O2(i,:), 'ok', 'DisplayName', 'O_2(t)', 'LineWidth', 3)

119 hold off
120 xlim([0 particle.diam/2*10^-4+1])
121 ylim([0 1.01])
122
text(15,.4,[\text{time = num2str(time(i),10)},'BackgroundColor', [.7 .9 .7])
123 grid on
124 ylabel('O2 mole fraction, [-]')
125 xlabel('Radius, [m]')
126 drawnow
127
display(

131 end
132
display;
133
display
134
\begin{verbatim}
% =========================================================================
% redefine input matrix
% =========================================================================

y_Init = outPut(end,:);
138
if sum(y_Init(1:nVols) >= sys.presO2_atm) > 1
139 y_Init(y_Init(1:nVols) >= sys.presO2_atm) = sys.presO2_atm;
140 end
141
if y_Init(nVols) >= sys.presO2_atm * 0.9999
144 y_Init(1:nVols) = sys.presO2_atm;
145 end
146
% =========================================================================
% calculate extent of for the whole particle...
% =========================================================================

if length(tRange.timeStep) == 2
152 % no need to run, were done.
153 % disp(['fraction of copper in Cu2O' num2str(outPut(end,2)/(outPut(end,1)+2*outPut(end,2))) ' %'])
155 break
156 end
157
if tRange.timeStepS(jj) > 8500 || xi_time > 0.995
158 break
159 end
160
% if xi_time > 0.90
163
\end{verbatim}
% % if the inner shell is 85% converted... stop the calc
165 % break
166 % end
167
168 % if y_Init(nVols) > sys.presO2_atm*0.9
169 % % if the PO2 at the CL equals 90 percent of the surface pressure
170 % break
171 % end
172 % end
173
174 % disp(['iteration: ' num2str(jj) ', ' num2str(length(time)) ' calculated up
175 % to minute ' num2str(tRange.timeStepS(jj)/60) '. Percent Conversion '
176 % num2str(particle.extent*100)])
177 % disp(['outer diameter ' num2str(particle.rad*10^-6) 'microns'])
178 % toc
179
C.2.4 Cermet particle model- distributed copper sub-particles

This function contains the code that describes the cermet particle when the cermet particle has Cu distributed throughout its core.

function dy = SpecsCalcOxSdndt_ODSFT(t,y,particleV,particle,shell,sys,D_effSolid,
    impur,deltaGCuO,dETAdrCuO,dETAdrCu2O)

% oxidation model, simple...
% this file models the oxidation of the Cu particle to CuO
% solve for copper by difference...
% SF [=] Stefan Flow
% OD [=] oxygen diffusion
% T [=] total calc--> regsite calc is internal

% Stefan flow says that since moles are fixed (TP = const) as O2 is
% consumed at the particle surface, new oxygen must enter the shell to
% replace consumed oxygen... otherwise the total pressure would change...

% future version: addd regSite calc inside

global rho MW
global nVols tSS
global reactOff gasDiffOff
C.2. CERMET PARTICLE MODEL

20 \[ t_{SS} = t; \]
21
22 \% CATCH STATEMENTS
23 \% ---------------------------------------------
24 \% ---------------------------------------------
25 if \( t > 1e^{-8} \)
26 \( a = 1; \)
27 end
28 if \( t > 1e^{-7} \)
29 \( a = 1; \)
30 end
31 if sum(\( y(1:nVols) > \text{sys.presO2_atm} \))
32 \( a = 1; \)
33 end
34
35 \% ---------------------------------------------
36 \% ---------------------------------------------
37 PO2Atm = \( y(1:nVols); \% \left( \text{sys.presTot}/(8.314*\text{sys.temp}*100^3) \right) \) \% ATM
38 ConcO2 = \( \text{PO2Atm} \times \text{sys.presTot}/(8.314*\text{sys.temp}*100^3); \)
39 \% \text{moles / cm}^3
40 \% ---------------------------------------------
41 \% ---------------------------------------------
42 moles.Cu = \( y(nVols+(1:nVols)) \times \text{particleV.tot.CuMoles}; \)
43 moles.Cu2O = \( y(nVols*2+(1:nVols)) \times \text{particleV.tot.CuMoles}; \% \text{Cu2O moles...} \)
44
45 \% if sum(moles.Cu<0)
46 \text{moles.Cu(moles.Cu<0)=0};
47 end
48
49 \% if sum(moles.Cu2O <0)
50 \text{moles.Cu2O(moles.Cu2O<0)=0};
51 end
52
53 volume.Cu = \( \text{moles.Cu}/\rho.Cu \times \text{MW.Cu}; \)
54 volume.Cu2O= \( \text{moles.Cu2O}/\rho.Cu2O \times \text{MW.Cu2O}; \)
55
56 thick.Cu = \( (3/(4*\pi) \times \text{volume.Cu})^{(1/3)}; \)
57 thick.Cu2O = \( ((3/(4*\pi) \times \text{volume.Cu2O + thick.Cu}^{3}))^{(1/3)}-\text{thick.Cu}; \% \text{cm} \)
58
59 moles.CuO = 0;
60 volume.CuO = 0;
61 thick.CuO = 0;
62 if \( \text{sys.presO2_atm} \geq \text{sys.presCu2OCuO_atm} \) \% if the potential for Cu0 formation
63 \% exists...
64 \text{moles.CuO} = \( y(3*nVols+(1:nVols)) \times \text{particleV.tot.CuMoles}; \)
65 \text{volume.CuO} = \text{moles.CuO}/\rho.CuO \times \text{MW.CuO};
\[
\text{thick.CuO} = \left(\frac{3}{4\pi}\right) \frac{\text{volume.CuO} + (\text{thick.Cu2O} + \text{thick.Cu})^3}{\text{thick.Cu2O} - \text{thick.Cu}} \quad \text{cm}
\]

% if sum(abs(1 - (moles.Cu+moles.Cu2O*2+moles.CuO)./particleV.tot.CuMoles))>1e-4
% a=1;
% end

% ---------------------------------------------------------------------
% initialize dy vector
% ---------------------------------------------------------------------
\text{dy} = \text{zeros}(nVols + 3*nVols,1); % O2, Cu, Cu2O, CuO
% ---------------------------------------------------------------------

noMoreCu = (y(nVols+(1:nVols)) < 1e-6); % Where Cu is depleted
noMoreCu2O = (y(nVols*2+(1:nVols)) < 1e-6); % Where Cu2O is depleted
% ---------------------------------------------------------------------
% ============ DEFINE Where ChemRx can occur ==============================
\text{Cu2OStable} = \text{find}(\text{PO2Atm} \leq \text{sys.presCu2OCuO_atm} \& \text{PO2Atm} > \text{sys.presCuCu2O_atm}); % shells where only Cu2O can form
\text{CuOStable} = \text{find}(\text{PO2Atm} < \text{sys.presCu2OCuO_atm}); % shells where CuO Can form
% ---------------------------------------------------------------------

if reactOff
\text{Cu2OStable} = []; \\
\text{CuOStable} = []; \\
end
% ---------------------------------------------------------------------
% Initialize Reactivity Vectors
% ---------------------------------------------------------------------
\text{RR.CuX} = \text{zeros}(nVols,1); \\
\text{RR.Cu2OX} = \text{zeros}(nVols,1); \\
\text{RR.CuOX} = \text{zeros}(nVols,1); \\
% ---------------------------------------------------------------------

nPoints = nVols; % if ~isempty(Cu2OStable)
nPoints = length(Cu2OStable);
% ---------------------------------------------------------------------
\text{dcdrNCu2O.Cu2O.h} = \text{zeros}(nPoints,1); \\
\text{dcdrNCu2O.Cu2O.e} = \text{zeros}(nPoints,1); \\
\text{dcdrNCu2O.Cu2O.vacCuM} = \text{zeros}(nPoints,1); \\
\text{dcdrNCu2O.Cu2O.vacCuX} = \text{zeros}(nPoints,1); \\
\text{dcdrNCu2O.Cu2O.intCuP} = \text{zeros}(nPoints,1); \\
\text{dcdrNCu2O.Cu2O.intCuX} = \text{zeros}(nPoints,1); \\
\text{dcdrNCu2O.Cu2O.vacOPP} = \text{zeros}(nPoints,1); \\
\text{dcdrNCu2O.Cu2O.CuO.h} = 0; \\
\text{dcdrNCu2O.Cu2O.CuO.e} = 0;
\[ \text{dcdrNCu2O.Cu0.vacCuM} = 0; \]
\[ \text{dcdrNCu2O.Cu0.vacCuX} = 0; \]
\[ \text{dcdrNCu2O.Cu0.intCuP} = 0; \]
\[ \text{dcdrNCu2O.Cu0.intCuX} = 0; \]
\[ \text{dcdrNCu2O.Cu0.vacOPP} = 0; \]

% ========================================================================
\[ \text{nPoints} = \text{nVols}; \]
\[ \text{if} \quad \text{isempty} ( \text{CuOStable} ) \]
\[ \quad \text{nPoints} = \text{length} ( \text{CuOStable} ); \]
\[ \text{end} \]
\[ \text{dcdrNCu0.Cu2O.h} = 0; \]
\[ \text{dcdrNCu0.Cu2O.e} = 0; \]
\[ \text{dcdrNCu0.Cu2O.vacCuM} = 0; \]
\[ \text{dcdrNCu0.Cu2O.vacCuX} = 0; \]
\[ \text{dcdrNCu0.Cu2O.intCuP} = 0; \]
\[ \text{dcdrNCu0.Cu2O.intCuX} = 0; \]
\[ \text{dcdrNCu0.Cu2O.vacOPP} = 0; \]

% ========================================================================
\[ \text{nPoints} = \text{nVols}; \]
\[ \text{if} \quad \text{isempty} ( \text{Cu2OStable} ) \]
\[ \quad \text{regSiteCu2O} = \text{defectEquilibriumFunc_V} ( \text{sys}, \text{PO2Atm} ( \text{Cu2OStable}), \text{deltaGCuO} , \text{impur}); \quad \% \text{site ratios when all shells are 21 pcnt. oxygen} \]
\[ \quad \text{regSiteCu2O.Cu2OStable} = \text{Cu2OStable}; \]
\[ \quad \text{dcdrNCu2O.Cu2O.h} = \text{spline} ( \log10 ( \text{dETAdrCu2O.pres} ), \text{dETAdrCu2O.Cu2O.h} , \log10 ( \text{PO2Atm} ( \text{Cu2OStable} ) )); \quad \% \text{site ratios when all shells are 21 pcnt. oxygen} \]
\[ \quad \text{regSiteCu20} = \text{defectEquilibriumFunc_V} ( \text{sys}, \text{P02Atm} ( \text{Cu20Stable} ), \text{deltaGCu0} , \text{impur}); \quad \% \text{site ratios when all shells are 21 pcnt. oxygen} \]
\[ \quad \text{regSiteCu20.Cu20Stable} = \text{Cu20Stable}; \]
\[ \text{dcdrNCu2O.Cu20.h} = \text{spline} ( \log10 ( \text{dETAdrCu20.pres} ), \text{dETAdrCu20.Cu20.h} , \log10 ( \text{PO2Atm} ( \text{Cu20Stable} ) )); \quad \% \text{site ratios when all shells are 21 pcnt. oxygen} \]
\[ \text{dcdrNCu2O.Cu20.e} = \text{spline} ( \log10 ( \text{dETAdrCu20.pres} ), \text{dETAdrCu20.Cu20.e} , \log10 ( \text{PO2Atm} ( \text{Cu20Stable} ) )); \quad \% \text{site ratios when all shells are 21 pcnt. oxygen} \]
\[ \text{dcdrNCu2O.Cu20.vacCuM} = \text{spline} ( \log10 ( \text{dETAdrCu20.pres} ), \text{dETAdrCu20.vacCuM}, \log10 ( \text{P02Atm} ( \text{Cu20Stable} ) )); \quad \% \text{site ratios when all shells are 21 pcnt. oxygen} \]
\[ \text{dcdrNCu2O.Cu20.vacCuX} = \text{spline} ( \log10 ( \text{dETAdrCu20.pres} ), \text{dETAdrCu20.vacCuX}, \log10 ( \text{P02Atm} ( \text{Cu20Stable} ) )); \quad \% \text{site ratios when all shells are 21 pcnt. oxygen} \]
dcdrNCu2O.Cu2O.intCuP = spline(log10(dETAdrCu2O.pres),
    dETAdrCu2O.Cu2O.intCuP,log10(P02Atm(Cu2OStable))).*(
    regSiteCu2O.intCuP.Cu2O2 - regSiteCu2O.intCuP.CuCu2O)*rho.Cu2O/
    MW.Cu2O;

dcdrNCu2O.Cu2O.intCuX = spline(log10(dETAdrCu2O.pres),
    dETAdrCu2O.Cu2O.intCuX,log10(P02Atm(Cu2OStable))).*(
    regSiteCu2O.intCuX.Cu2O2 - regSiteCu2O.intCuX.CuCu2O)*rho.Cu2O/
    MW.Cu2O;

dcdrNCu2O.Cu2O.vacOPP = 0;

% only Cu2O forms here so no CuO...

dcdrNCu2O.Cu2O.h = 0;

dcdrNCu2O.Cu2O.e = 0;

dcdrNCu2O.Cu2O.vacCuMM = 0;

dcdrNCu2O.Cu2O.intCuPP = 0;

dcdrNCu2O.Cu2O.vacOPP = 0;
end

% -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -= -
C.2. CERMET PARTICLE MODEL

```plaintext
170  dcdrNCuO.CuO.h = spline(log10(dETAdrCuO.pres),dETAdrCuO.CuO.h, 
  log10(PO2Atm(CuOStable))).*(regSiteCuO.h.CuOO2 - 
  regSiteCuO.h.CuOCu2O)*rho.CuO/MW.CuO;
171  dcdrNCuO.CuO.e = spline(log10(dETAdrCuO.pres),dETAdrCuO.CuO.e, 
  log10(PO2Atm(CuOStable))).*(regSiteCuO.e.CuOO2 - 
  regSiteCuO.e.CuOCu2O)*rho.CuO/MW.CuO;
172  dcdrNCuO.CuO.vacCuMM = spline(log10(dETAdrCuO.pres), 
  dETAdrCuO.CuO.vacCuMM,log10(PO2Atm(CuOStable))).*( 
  regSiteCuO.vacCuMM.CuOO2 - regSiteCuO.vacCuMM.CuOCu2O)*rho.CuO 
  /MW.CuO;
173  dcdrNCuO.CuO.intCuPP = spline(log10(dETAdrCuO.pres), 
  dETAdrCuO.CuO.intCuPP,log10(PO2Atm(CuOStable))).*( 
  regSiteCuO.intCuPP.CuOO2 - regSiteCuO.intCuPP.CuOCu2O)*rho.CuO 
  /MW.CuO;
174  dcdrNCuO.CuO.vacOPP = 0;
175  else
176    dcdrNCuO.CuO.h = dETAdrCuO.CuO.h.*(regSiteCuO.h.CuOO2 - 
      regSiteCuO.h.CuOCu2O)*rho.CuO/MW.CuO;
177    dcdrNCuO.CuO.e = dETAdrCuO.CuO.e.*(regSiteCuO.e.CuOO2 - 
      regSiteCuO.e.CuOCu2O)*rho.CuO/MW.CuO;
178    dcdrNCuO.CuO.vacCuMM = dETAdrCuO.CuO.vacCuMM.*( 
      regSiteCuO.vacCuMM.CuOO2 - regSiteCuO.vacCuMM.CuOCu2O)*rho.CuO 
      /MW.CuO;
179    dcdrNCuO.CuO.intCuPP = dETAdrCuO.CuO.intCuPP.*( 
      regSiteCuO.intCuPP.CuOO2 - regSiteCuO.intCuPP.CuOCu2O)*rho.CuO 
      /MW.CuO;
180    dcdrNCuO.CuO.vacOPP = 0;
181  end
182
183  \% reactions should stop if P_O2 goes below P_O2/Cu2O/CuO
184  if ~isempty(Cu2OStable)
185    RR_Cu2O = RRParticleFuncDefect(thick,Cu2OStable,sys,y(nVols+(1:nVols)),y( 
      nVols+2+(1:nVols)),D_effSolid,regSiteCu2O,dcdrNCu2O); % [moles/m^2-s 
      \rightarrow ], Cu/Cu2O/CuO
186  end
187
188  if ~isempty(CuOStable)
189    RR_CuO = RRParticleFuncDefect(thick,CuOStable,sys,y(nVols+(1:nVols)),y(nVols 
      +2+(1:nVols)),D_effSolid,regSiteCuO,dcdrNCuO); % [moles/m^2-s], Cu/ 
      \rightarrow Cu2O/CuO
190  end
191
192  if ~isempty(Cu2OStable)
193    RR_CuX(Cu2OStable) = RR_Cu2O.Cu; \% mole/m^2 of Cu particle in shell i-
194  \rightarrow second
```
RR_Cu2OX(Cu2OStable) = RR_Cu2O.Cu2O; % mole/m^2 of Cu particle in shell i-→ second

if ~isempty(CuOStable)
    RR_CuX(CuOStable) = RR_CuO.Cu; % mole/m^2 of Cu particle in shell i-→ second
    RR_Cu2OX(CuOStable) = RR_CuO.Cu2O; % mole/m^2 of Cu particle in shell i-→ second
    RR_CuOX(CuOStable) = RR_CuO.CuO; % mole/m^2 of Cu particle in shell i-→ second
end

dy(nVols+(1:nVols)) = RR_CuX./particleV.tot.CuMoles; % Cu, mole of i /total moles - seconds

dy(nVols*2+(1:nVols)) = RR_Cu2OX./particleV.tot.CuMoles; % Cu2O

if sys.presO2_atm > sys.presCu2OCuO_atm
    dy(nVols*3+(1:nVols)) = RR_CuOX./particleV.tot.CuMoles; % CuO
end

% stop critera for Cu
if sum(noMoreCu)>0
    dy(nVols+ noMoreCu) = 0;
end

% stop critera for no Cu2O
if sum(noMoreCu2O)>0
    dy(nVols + noMoreCu2O) = 0;
    dy(nVols*2+noMoreCu2O) = 0;
    dy(nVols*3+noMoreCu2O) = 0; % if Cu2O is done in a given shell then so is Cu0
end

RR_O2 = (RR_Cu2OX + RR_CuOX)/2; % moles/s of O2

% =========================================================================
% =================== Reactivity TERM =====================================
% RR_Term = RR_O2.*shell.Me.numb./(shell.vol.*shell.porosity); % moles/s x num / cm^3 void space, moles/cm^3-s
% RR_Term = RR_O2.*shell.Me.numb./(shell.vol); % moles/s x num / cm^3, moles/cm^3-s
% =========================================================================

% Oxygen Diffusivity
D_eff.O2 = DeffFuncOx(particle,shell); % cm^2/s, defined at node center
DBulk = DBulkOx(sys.temp,sys.presO2) * 100^2; % cm^2/s
C.2. CERMET PARTICLE MODEL

\% ============= START ==============
\% J = -D x d[O2]/dr + v x [O2] [=] moles/m^2-s, where v x [O2] = RR x XO2
\% RR_O2 * shell.Me.numb = A * betaTerm * log((1-gamma*XO2_S)/(1-gamma*XO2_inf))
\% 2Cu+1/2 O2 -> Cu2O gamma = 1
\% Cu+1/2 O2 -> CuO gamma = 1
\% gamma = 1; % change in volume from reaction (e.g. change of moles in gas
\% phase if T/P fixed) per reactant
\% SHNum = 2; % limit of small particles, convective mass transport to
\% diffusive transport, k_eff * L / D_bulk
\% betaTerm = SHNum * DBulk/particle.diam * sys.presTot/(gamma*8.314*sys.temp
\% *100^-3); % moles/cm^2-s
\% XO2_s = (1 - exp(sum(RR_O2.*shell.Me.numb)/(pi*particle.diam^2*betaTerm)))*(1-
\% gamma*sys.presO2_atm))/gamma;
\% ConcO2_s = XO2_s * sys.presTot/(8.314*sys.temp*100^-3); % moles/cm^3
\% JBL = betaTerm * log((1 - gamma*XO2_s)/(1 - gamma*sys.presO2_atm)); % moles/m^2-s, flux of oxygen to surface of the particle
\% calculate stefan velocity profile...
\% integralTerm = -flipud(cumtrapz(flipud(shell.radCm),flipud(shell.radCm.^2.*
\% RR_Term))); % define at tcenter of each volume
\% Vel = 8.314*sys.temp*100^-3./((sys.presTot*shell.radCm.^2).*integralTerm); % cm/s
\% if false
\% % MolesO2PSec = JBL * 4*pi* (particle.diam/2)^2; % moles/sec
\% % MolesO2Cons = sum(RR_O2.*shell.Me.numb); % moles/sec
\% % MolesO2SFlux = Vel.*(4*pi*shell.radCm.^2).* PO2Atm/(8.314*sys.temp*100^-3);
\% % nDotO2 = (J02 + 0*J02SF).*4*pi.*shell.radCm.^2;
\% % J02 = D_eff.O2.* sys.presTot/(8.314 * sys.temp*100^-3).*dxCalc(PO2Atm,
\% shell.radCm);'
\% % J02SF = -Vel.* PO2Atm* sys.presTot / (8.314 * sys.temp*100^-3);
\% % nDotO2Eox = RR_O2 * shell.Me.numb; % moles / second consumed per shell
\% % stefan velocity = RR/(P/RT)/Area * gamma
% Shell 1
% minus 1/2 and plus 1/2 point
rhm = shell.radius(1);
rph = shell.radius(2);

% flux in through boundary layer
fluxInTerms = (rhm).^2.*D_eff.O2(1).* ...
(CoC02_s - CoC02(1))./(shell.radius(1) - shell.radCm(1))

fluxOutTerms = (rph).^2.*D_eff.O2(2).* ...
(CoC02(1) - CoC02(2))./(shell.radCm(1) - shell.radCm(2));

diffTerms(1,1) = 1./(shell.radCm(1)).^2.*(fluxInTerms - fluxOutTerms)./(rhm - rph);
stefTerms(1,1) = RR_Term(1) * PO2Atm(1) - Vel(1) * (CoC02_s - CoC02(1)) /
(shell.radius(1) - shell.radCm(1));

dy(1) = -RR_Term(1) + diffTerms(1,1) + stefTerms(1,1); % mole/m^-3-s

% Middle: Node 2 to N-1
% if not the first or the last shell....
% Apply Conservative Differencing Scheme
midRange = (2:(nVols-1))';

% minus 1/2 point, plus 1/2 point
rhm = shell.radius(midRange); % minus half
rph = shell.radius(midRange+1); % plus half

fluxInTerms = rmh.^2.*D_eff.O2(midRange-1).* (CoC02(midRange-1) - CoC02(midRange))./(shell.radCm(midRange-1) - shell.radCm(midRange)); % mole/s

fluxOutTerms = rph.^2.*D_eff.O2(midRange+1).* (CoC02(midRange) - CoC02(midRange+1))./(shell.radCm(midRange) - shell.radCm(midRange+1)); % mole/s

diffTerms(midRange,1) = 2./(shell.radCm(midRange)).^2.*(fluxInTerms - fluxOutTerms)./(rhm - rph);
stefTerms(midRange,1) = RR_Term(midRange).*PO2Atm(midRange) - (Vel(midRange-1) + Vel(midRange))/2.* (CoC02(midRange-1) - CoC02(midRange)) / (shell.radCm(midRange-1) - shell.radCm(midRange));

dy(midRange) = -RR_Term(midRange).* (1 - PO2Atm(midRange)) * diffTerms(midRange)
C.2. CERMET PARTICLE MODEL

\[ - (\text{Vel} (\text{midRange} - 1) + \text{Vel} (\text{midRange})) / 2 \times (\text{ConcO2} (\text{midRange} - 1) - \text{ConcO2} (\text{midRange}))/ (\text{shell.radCm}(\text{midRange} - 1) - \text{shell.radCm}(\text{midRange})); \] 

\[ \text{dy}(\text{midRange}) = -\text{RR_Term}(\text{midRange}) + \text{diffTerms}(\text{midRange}) + \text{stefTerms}(\text{midRange}); \]

\[ \text{END} \text{ (N} \text{ shell)} \]

\[ \% \text{ flux at center line is zero...} \]

\[ \% \ c(r) = a + b \times r + c \times r^2 \text{ where } \text{dc/dr} = 0 \quad \text{N N-1 N-2} \]

\[ \% \ \text{nVol is the cell next to the center line} \]

\[ \text{coefEND.c} = (\text{ConcO2(nVol}-1) - \text{ConcO2(nVols)})/(\text{shell.radCm(nVols}-1)^2 - \text{shell.radCm(nVols)}^2); \%
\]

\[ \% \text{ minus 1/2 point, plus 1/2 point} \]

\[ \text{rmh} = \text{shell.radius(nVol);} \]

\[ \text{rph} = \text{shell.radius(nVol+1);} \]

\[ \text{fluxInTerms} = \text{rmh}^2 \times \text{D_eff.O2(nVol}-1) \times (2 \times \text{coefEND.c} \times \text{rmh}); \]

\[ \text{fluxOutTerms} = \text{rph}^2 \times \text{D_eff.O2(nVols)} \times (2 \times \text{coefEND.c} \times \text{rph}); \]

\[ \text{diffTerms(nVols)} = 2 / (\text{shell.radCm(nVols)}^2 \times (\text{fluxInTerms} - \text{fluxOutTerms}) / (\text{rmh} - \text{rph}); \]

\[ \text{stefTerms(nVols,1)} = \text{RR_Term(nVols)} \times \text{PO2Atm(nVols)} - (\text{Vel(nVol}-1) + \text{Vel(nVols)}))/2 \times (\text{ConcO2(nVols}-1) - \text{ConcO2(nVols)})/ (\text{shell.radCm(nVols}-1) - \text{shell.radCm(nVols)}); \%
\]

\[ \% \text{ dy(nVols)} = -(\text{RR_Term(nVols)} \times (1 - \text{PO2Atm(nVols)}) + \text{diffTerms(nVols)}... \]

\[ \% \text{ - (Vel(nVol}-1) + \text{Vel(nVol))}/2 \times (\text{ConcO2(nVols}-1) - \text{ConcO2(nVols)}); \%
\]

\[ \% \text{ dy(nVols)} = -(\text{RR_Term(nVols)} \times \text{diffTerms(nVols)} + \text{stefTerms(nVols)}; \%
\]

\[ \% \text{ dy(1:nVols)} = \text{dy(1:nVols)} * 1 / (\text{sys.presTot} / (8.314 \times \text{sys.temp} \times 100^3)); \%
\]

\[ \% \text{ imagDY = find(abs(img(dy))>0);} \]

\[ \% \text{ toc} \]

\[ \% \text{ disp('imag number...')} \]

\[ \% \text{ pause} \]

\[ \% \text{ a =1;} \]

\[ \text{dy(imgDY)} = 0; \]

\[ \text{end} \]

\[ \% \text{ if PO2 is greater than ambient... stop reactions in that cell} \]

\[ \% \text{ P02Atm(nVols) >= sys.presO2_atm} \]
APPENDIX C. MATLAB CODES

C.2.5 Cermet particle model- core copper

This function contains the code that describes the cermet particle when there is a core copper particle.

```matlab
function dy = SpecsCalcOxSdndt_ODSFT_CORE(t,y,particleV,particle,shell,sys,
            D_effSolid,impur,deltaGCuO,dETAdrCuO,dETAdrCu2O)
% oxidation model, simple...
% this file models the oxidation of the Cu particle to CuO
% solve for copper by difference...
% SF [=] Stefan Flow
% OD [=] oxygen diffusion
% T [=] total calc--> regsite calc is internal
% Stefan flow says that since moles are fixed (TP = const) as O2 is
% consumed at the particle surface, new oxygen must enter the shell to
% replace consumed oxygen... otherwise the total pressure would change...
% future version: addd regSite calc inside

global rho MW
global nVols tSS
global reactOff gasDiffOff

% Stefan flow

tSS = t;

% catch statements

if t > 1e-8
    a=1;
end
if t>1e-7
    a=1;
end
```
if sum(y(1:nVols) > sys.presO2_atm) 
a = 1;
end

% =========================================================================
% =========================================================================
PO2Atm = y(1:nVols); % /(sys.presTot/(8.314*sys.temp*100^-3)); % ATM
ConcO2 = PO2Atm*sys.presTot/(8.314*sys.temp*100^-3);%
% moles / cm^-3
% =========================================================================
% =========================================================================
% Define particle properties
% moles.Cu = y(nVols+1).*particleV.tot.CuMoles;
% moles.Cu2O = y(nVols+2).*particleV.tot.CuMoles; % Cu2O moles...
% noMoreCu = (y(1+(nVols))< 1e^-6); % Where Cu is depleted
% noMoreCu2O = (y(2+(nVols))< 1e^-6); % Where Cu2O is depleted
% moles.Cu = (1-noMoreCu) * moles.Cu;
% moles.Cu2O = (1-noMoreCu2O) * moles.Cu2O;
% if length(noMoreCu)==nVols && length(noMoreCu2O)==nVols
% code is done...
% dy(1:(nVols+3),1) = 0;
% return
% end
% if sum(isnan(y(1:nVols))) == nVols
% code is done...
% dy(1:(nVols+3),1) = 0;
% return
% end
% =========================================================================

thick.Cu = (3/(4*pi)*volume.Cu).^(-1/3);
thick.Cu2O = ((3/(4*pi)*volume.Cu2O + thick.Cu.^3)).^-1/3-thick.Cu; % cm

moles.Cu0 = 0;
volume.Cu0 = 0;

if sys.presO2_atm >= sys.presCu2OCuO_atm % if the potential for CuO formation
% exists...
moles.Cu0 = y(nVols+3).*particleV.tot.CuMoles;
volume.Cu0 = moles.Cu0/rho.Cu0*MW.CuO;
thick.CuO = ((3/(4*pi)*volume.CuO + (thick.Cu2O+thick.Cu)^3))^(1/3) - thick.Cu2O-thick.Cu; % cm

% Initialize dy vector
dy = zeros(nVols+3,1); % O2, Cu, Cu2O

% DEFINE Where ChemRx can occur
Cu2OStable = find(PO2Atm(nVols) <= sys.presCu2OCuO_atm & PO2Atm(nVols) > sys.presCuCu2O_atm); % shells where only Cu2O can form
CuOStable = find(PO2Atm(nVols) > sys.presCu2OCuO_atm); % shells where CuO Can form

if reactOff
    Cu2OStable = [];
    CuOStable = [];
end

% Initialize Reactivity Vectors
RR_CuX = 0;
RR_Cu2OX = 0;
RR_CuOX = 0;

% Initialize dy vectors
nPoints = 1;
if ~isempty(Cu2OStable)
    nPoints = length(Cu2OStable);
end
dcdrNCu2O.Cu2O.h = zeros(nPoints,1);
dcdrNCu2O.Cu2O.e = zeros(nPoints,1);
dcdrNCu2O.Cu2O.vacCuM = zeros(nPoints,1);
dcdrNCu2O.Cu2O.vacCuX = zeros(nPoints,1);
dcdrNCu2O.Cu2O.intCuP = zeros(nPoints,1);
dcdrNCu2O.Cu2O.intCuX = zeros(nPoints,1);
dcdrNCu2O.Cu2O.vacOPP = zeros(nPoints,1);
dcdrNCu2O.Cu0.h = 0;
dcdrNCu2O.Cu0.e = 0;
dcdrNCu2O.Cu0.vacCuM = 0;
dcdrNCu2O.Cu0.vacCuX = 0;
dcdrNCu2O.Cu0.intCuP = 0;
dcdrNCu2O.Cu0.intCuX = 0;
dcdrNCu2O.Cu0.vacOPP = 0;
C.2. CERMET PARTICLE MODEL

120 \% --------------------------------------------------------------------------------------------------------
121 nPoints = 1;
122 if ~isempty(CuOStable)
123     nPoints = length(CuOStable);
124 end
125 dcdrNCuO.Cu2O.h = 0;
126 dcdrNCuO.Cu2O.e = 0;
127 dcdrNCuO.Cu2O.vacCuM = 0;
128 dcdrNCuO.Cu2O.vacCuX = 0;
129 dcdrNCuO.Cu2O.intCuP = 0;
130 dcdrNCuO.Cu2O.intCuX = 0;
131 dcdrNCuO.Cu2O.vacOPP = 0;
132 dcdrNCuO.Cu0.h = zeros(nPoints,1);
133 dcdrNCuO.Cu0.e = zeros(nPoints,1);
134 dcdrNCuO.Cu0.vacCuM = zeros(nPoints,1);
135 dcdrNCuO.Cu0.intCuP = zeros(nPoints,1);
136 dcdrNCuO.Cu0.vacOPP = zeros(nPoints,1);
137 \% --------------------------------------------------------------------------------------------------------
138 if ~isempty(Cu2OStable)
139 \% WHEN ONLY CU2O FORMS
140     regSiteCu2O = defectEquilibriumFunc_V(sys, PO2Atm(nVols),deltaGCuO, impur);
141 \% site ratios when all shells are 21 pcnt. oxygen
142     regSiteCu2O.Cu2OStable = 1;
143     dcdrNCu2O.Cu2O.h = spline(log10(dETAdrCu2O.pres), dETAdrCu2O.Cu2O.h,
144 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.h.Cu2OO2 -
145 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.h.CuCu2O)*rho.Cu2O/MW.Cu2O;
146     dcdrNCu2O.Cu2O.e = spline(log10(dETAdrCu2O.pres), dETAdrCu2O.Cu2O.e,
147 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.e.Cu2OO2 -
148 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.e.CuCu2O)*rho.Cu2O/MW.Cu2O;
149     dcdrNCu2O.Cu2O.vacCuM = spline(log10(dETAdrCu2O.pres),
150 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.vacCuM.Cu2OO2 -
151 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.vacCuM.CuCu2O)*rho.Cu2O/MW.Cu2O;
152     dcdrNCu2O.Cu2O.vacCuX = spline(log10(dETAdrCu2O.pres),
153 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.vacCuX.Cu2OO2 -
154 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.vacCuX.CuCu2O)*rho.Cu2O/MW.Cu2O;
155     dcdrNCu2O.Cu2O.intCuP = spline(log10(dETAdrCu2O.pres),
156 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.intCuP.Cu2OO2 -
157 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.intCuP.CuCu2O)*rho.Cu2O/MW.Cu2O;
158     dcdrNCu2O.Cu2O.intCuX = spline(log10(dETAdrCu2O.pres),
159 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.intCuX.Cu2OO2 -
160 \% \log10\(P_{O2}\text{Atm}(\text{nVol})\)).*(regSiteCu2O.intCuX.CuCu2O)*rho.Cu2O/MW.Cu2O;
APPENDIX C. MATLAB CODES

dcdrNCu20.Cu20.vacOPP = 0;

% only Cu2O forms here so no CuO...
dcdrNCu20.Cu0.h = 0;
dcdrNCu20.Cu0.e = 0;
dcdrNCu20.Cu0.vacCuMM = 0;
dcdrNCu20.Cu0.intCuPP = 0;
dcdrNCu20.Cu0.vacOPP = 0;
end

% -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
% -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
% -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

% -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
% -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
% -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

if ~isempty(CuOStable)
regSiteCuO = defectEquilibriumFunc_V(sys,PO2Atm(nVols),deltaGCuO,impur);

regSiteCuO.CuOStable = 1;

dcdrNCu0.Cu0.h = dETAdrCu0.Cu0.h*(regSiteCuO.h.Cu0Cu0 -
regSiteCuO.h.CuCu2O)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.e = dETAdrCu0.Cu0.e*(regSiteCuO.e.Cu0Cu0 -
regSiteCuO.e.CuCu2O)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.vacCuM = dETAdrCu0.Cu0.vacCuM*(regSiteCu0.vacCuM.Cu0Cu0 -
regSiteCu0.vacCuM.CuCu2O)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.vacCuX = dETAdrCu0.Cu0.vacCuX*(regSiteCu0.vacCuX.Cu0Cu20 -
regSiteCu0.vacCuX.CuCu20)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.intCuP = dETAdrCu0.Cu0.intCuP*(regSiteCu0.intCuP.Cu0Cu20 -
regSiteCu0.intCuP.CuCu20)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.intCuX = dETAdrCu0.Cu0.intCuX*(regSiteCu0.intCuX.Cu0Cu20 -
regSiteCu0.intCuX.CuCu20)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.vacOPP = 0;

else if length(dETAdrCu0.pres) == 1

dcdrNCu0.Cu0.h = dETAdrCu0.Cu0.h*(regSiteCu0.h.Cu0Cu0 -
regSiteCu0.h.CuOCu20)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.e = dETAdrCu0.Cu0.e*(regSiteCu0.e.Cu0Cu0 -
regSiteCu0.e.CuOCu20)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.vacCuMM = dETAdrCu0.Cu0.vacCuMM*(regSiteCu0.vacCuMM.Cu0Cu0 -
regSiteCu0.vacCuMM.CuOCu20)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.intCuPP = dETAdrCu0.Cu0.intCuPP*(regSiteCu0.intCuPP.Cu0Cu0 -
regSiteCu0.intCuPP.CuOCu20)*rho.Cu0/MW.Cu0;
dcdrNCu0.Cu0.vacOPP = 0;
C.2. CERMET PARTICLE MODEL

```plaintext
else
  dcdrNCuO.CuO.h = spline(log10(dETAdrCuO.pres), dETAdrCuO.CuO.h, 
  log10(PO2Atm(nVols))).*(regSiteCuO.h.CuOO2 -
  regSiteCuO.h.CuOCu2O)*rho.CuO/MW.CuO;
  dcdrNCuO.CuO.e = spline(log10(dETAdrCuO.pres), dETAdrCuO.CuO.e, 
  log10(PO2Atm(nVols))).*(regSiteCuO.e.CuOO2 -
  regSiteCuO.e.CuOCu2O)*rho.CuO/MW.CuO;
  dcdrNCuO.CuO.vacMM = spline(log10(dETAdrCuO.pres),
  dETAdrCuO.CuO.vacMM, log10(PO2Atm(nVols))).*(
  regSiteCuO.vacMM.CuOO2 - regSiteCuO.vacMM.CuOCu2O)*rho.CuO 
  /MW.CuO;
  dcdrNCuO.CuO.intCuPP = spline(log10(dETAdrCuO.pres),
  dETAdrCuO.CuO.intCuPP, log10(PO2Atm(nVols))).*(
  regSiteCuO.intCuPP.CuOO2 - regSiteCuO.intCuPP.CuOCu2O)*rho.CuO 
  /MW.CuO;
  dcdrNCuO.CuO.vacOPP = 0;
end

% reactions should stop if PO2 goes below P_O2_Cu2O/CuO
if ~isempty(Cu2OStable)
  RR_Cu2O = RRParticleFuncDefect(thick, Cu2OStable, sys, y(1+(nVols)), y(2+(nVols)),
  D_effSolid, regSiteCu2O, dcdrNCu2O); % [moles/m^2-s], Cu/Cu2O/CuO
end
if ~isempty(CuOStable)
  RR_CuO = RRParticleFuncDefect(thick, CuOStable, sys, y(1+(nVols)), y(2+(nVols)),
  D_effSolid, regSiteCuO, dcdrNCuO); % [moles/m^2-s], Cu/Cu2O/CuO
end
if ~isempty(Cu2OStable)
  RR_CuX(Cu2OStable) = RR_Cu2O.Cu; % mole/cm^2 of Cu particle in shell i
    -second
  RR_Cu2OX(Cu2OStable) = RR_Cu2O.Cu2O; % mole/cm^2 of Cu particle in shell i
    -second
end
if ~isempty(CuOStable)
  RR_CuX(CuOStable) = RR_CuO.Cu; % mole/cm^2 of Cu particle in shell i
    -second
  RR_Cu2OX(CuOStable) = RR_CuO.Cu2O; % mole/cm^2 of Cu particle in shell i
    -second
  RR_CuOX(CuOStable) = RR_CuO.CuO; % mole/cm^2 of Cu particle in shell i
    -second
end
```
APPENDIX C. MATLAB CODES

dy(nVols+1) = RR_CuX./particleV.tot.CuMoles;  % Cu, mole of i / total moles -
\rightarrow \text{seconds}
dy(nVols+2) = RR_Cu2OX./particleV.tot.CuMoles;  % Cu2O
dy(nVols+3) = RR_CuOX./particleV.tot.CuMoles;  % Cu2O

% stop criteria for Cu
if sum(noMoreCu)
dy(nVols+ noMoreCu) = 0;
end

if sum(noMoreCu2O)>0
dy(nVols + noMoreCu2O) = 0;
dy(nVols+1+ noMoreCu2O) = 0;
end

RR_O2 = ( RR_Cu2OX + RR_CuOX ) /2;  % moles /s of O2

% =========================================================================
% =================== Reactivity TERM =====================================
RR_Term(1:nVols,1) = 0;
RR_Term(nVols) = RR_O2./(shell.vol(nVols));  % moles/s x num / cm^3 void space
\rightarrow , moles/cm^-3-s
% =========================================================================
% =================== Oxygen Diffusivity ====================================
D_eff.O2 = DeffFuncOx(particle, shell);  % cm^-2/s, defined at node center
DBulk = DBulkOx(sys.temp, sys.presO2) * 100^-2;  % cm^-2/s
% =========================================================================
% ============= START ====================================================
% ========================================================================

J = -D x d[O2]/ dr + v x [O2]  % moles/m^2-s, where v x [O2] = RR x XO2
RR_O2 * shell.Me.numb = A * betaTerm * log((1 - gamma * XO2_S )/(1 - gamma * XO2_inf ))
2Cu +1/2 O2 -> Cu2O gamma = 1
Cu +1/2 O2 -> CuO gamma = 1

gamma = 1;  % change in volume from reaction (e.g. change of moles in gas
\rightarrow \text{phase if T/P fixed} per reactant
SHNum = 2;  % limit of small particles, convective mass transport to
\rightarrow \text{diffusive transport}, k_eff \ast L / D_bulk
betaTerm = SHNum \ast DBulk/particle.diam \ast sys.presTot/(gamma*8.314*sys.temp
\rightarrow *100^-3);  % moles/cm^-2-s

XO2_s = (1 - exp(sum(RR_O2.*shell.Me.numb)/(pi*particle.diam^-2*betaTerm)))*(1-
\rightarrow gamma*sys.pres2_atm))/gamma;
ConcO2_s = XO2_s*sys.presTot/(8.314*sys.temp*100^-3);  % moles / cm^-3

JBL = betaTerm \ast log((1 - gamma*XO2_s)/(1 - gamma*sys.pres2_atm));  % moles/
\rightarrow m^-2-s, flux of oxygen to surface of the particle
% calculate stefan velocity profile...
integralTerm = -flipud(cumtrapz(flipud(shell.radCm),flipud(shell.radCm.^2.*RR_Term))); % define at tcenter of each volume
Vel = 8.314*sys.temp*100^3./(sys.presTot*shell.radCm.^2).*integralTerm; % cm/s

% stefan velocity = RR/(P/RT)/Area * gamma
% Shell 1
% minus 1/2 and plus 1/2 point
rmh = shell.radius(1);
rph = shell.radius(2);

% flux in through boundary layer
 fluxInTerms = (rmh).^2.*D_eff.O2(1).*... % D_eff(1) b/c this loc is not defined...
          (Conc02_s - Conc02(1))./(shell.radius(1) - shell.radCm(1))
          % mole/s
fluxOutTerms = (rph).^2.*D_eff.O2(2).*...
          (Conc02(1) - Conc02(2))./(shell.radCm(1) - shell.radCm(2));
          % mole/s

diffTerms(1,1) = 1./(shell.radCm(1)).^2.*(fluxInTerms - fluxOutTerms)./(rmh - rph);
stefTerms(1,1) = RR_Term(1)*PO2Atm(1) - Vel(1)*((Conc02_s - Conc02(1))/>
                   shell.radius(1) - shell.radCm(1));

dy(1) = -RR_Term(1) + diffTerms(1,1) + stefTerms(1,1); % mole/m^3-s

% Middle: Node 2 to N-1
% if not the first or the last shell....
% Apply Conservative Differencing Scheme
midRange = (2:(nVols-1))';
% minus 1/2 point, plus 1/2 point
rmh = shell.radius(midRange); % minus half
rph = shell.radius(midRange+1); % plus half

fluxInTerms = rmh.*2.*D_eff.O2(midRange-1).*((Conc02(midRange-1) - Conc02(midRange))./(shell.radCm(midRange-1) - shell.radCm(midRange)));
          % mole/s
fluxOutTerms = rph.*2.*D_eff.O2(midRange+1).*((Conc02(midRange) - Conc02(midRange+1))./(shell.radCm(midRange) - shell.radCm(midRange+1)));
          % mole/s
diffTerms(midRange,1) = 2./(shell.radCm(midRange)).^2.*(fluxInTerms - fluxOutTerms)./(rmh - rph);
APPENDIX C. MATLAB CODES

```matlab
    stefTerms(midRange,1) = RR_Term(midRange).*PO2Atm(midRange) - (Vel(midRange-1) + 
        Vel(midRange))/2.*ConcO2(midRange-1) - ConcO2(midRange))./(shell.radCm(midRange-1) - shell.radCm(midRange));

    % dy(midRange) = -RR_Term(midRange).*(1 - PO2Atm(midRange)) + diffTerms(midRange) 
        -(Vel(midRange-1) + Vel(midRange))/2.*ConcO2(midRange-1) - ConcO2(midRange))./(shell.radCm(midRange-1) - shell.radCm(midRange)); % mole/m^3-s

    % dy(midRange) = -RR_Term(midRange) + diffTerms(midRange) + stefTerms(midRange) % mole/m^3-s

    % flux at center line is zero...
    % c(r) = a + b*r + c * r^2 where dc/dr = 0 N N-1 N-2
    % nVols is the cell next to the center line in.
    %
    coefEND.c = (ConcO2(nVols-1)-ConcO2(nVols))/(shell.radCm(nVols-1)^2 - shell.radCm(nVols)^2); % [mole/cm^5]

    % minus 1/2 point, plus 1/2 point
    rmh = shell.radius(nVols);
    rph = shell.radius(nVols+1);

    fluxInTerms = rmh.^2.*D_eff.O2(nVols-1).*(2 * coefEND.c * rmh);
    fluxOutTerms = rph.^2.*D_eff.O2(nVols).*(2 * coefEND.c * rph);
    diffTerms(nVols) = 2./(shell.radCm(nVols)).^2.*(fluxInTerms - fluxOutTerms) 
        -(Vel(midRange-1) + Vel(midRange))/2.*ConcO2(midRange-1) - ConcO2(midRange))./(shell.radCm(midRange-1) - shell.radCm(midRange)); % mole/m^3-s

    % dy(nVols) = -RR_Term(nVols).*PO2Atm(nVols) + diffTerms(nVols)... 
        -shell.radCm(nVols-1) - shell.radCm(nVols)); % mole/m^3-s

    % dy(nVols) = -RR_Term(nVols) + diffTerms(nVols) + stefTerms(nVols); % mole/m^3-s

    % ========================================================================
    % dy(1:nVols) = dy(1:nVols)*1/(sys.presTot/(8.314*sys.temp*100^3));%.* 1./ 
        shell.porosity;
    % ========================================================================

    imagDY = find(abs(imag(dy))>0);
    if ~isempty(imagDY)
        toc
        disp('imag number...')
        pause
    ```
C.2. CERMET PARTICLE MODEL

C.2.6 Copper oxidation subroutine

This code calculates the reactivity of the copper for the cermet particle model.

```matlab
function outPut = RRParticleFuncDefect(thick, index, sys, CuAtomFrac, Cu2OAtomFrac,
                                         D_eff, regSite, dcdrN)
    global Cn rho MW
    isCuO = 0;
    try
        isCuO = ~isempty(regSite.CuOStable);
    end

    AoR.Cu2O = 4*pi*(thick.Cu(index)).^2; % area at Cu/Cu2O interface...
    AoR.CuO = 4*pi*(thick.Cu(index)+thick.Cu2O(index)).^2; % area at Cu2O/CuO interface...

dcdr.Cu2O.h = dcdrN.Cu2O.h./((1./ri - 1./ro).*ri.^2);
```

dcdr.Cu2O.e = dcdrN.Cu2O.e. /((1 ./ri - 1 ./ro).*ri.^2);  
dcdr.Cu2O.vacCuM = dcdrN.Cu2O.vacCuM. /((1 ./ri - 1 ./ro).*ri.^2);  
dcdr.Cu2O.vacCuX = dcdrN.Cu2O.vacCuX. /((1 ./ri - 1 ./ro).*ri.^2);  
dcdr.Cu2O.intCuP = dcdrN.Cu2O.intCuP. /((1 ./ri - 1 ./ro).*ri.^2);  
dcdr.Cu2O.intCuX = dcdrN.Cu2O.intCuX. /((1 ./ri - 1 ./ro).*ri.^2);  
dcdr.Cu2O.vacOPP = dcdrN.Cu2O.vacOPP. /((1 ./ri - 1 ./ro).*ri.^2);  

if isCuO  
    ri = thick.Cu(index)+thick.Cu2O(index);  
    ro = thick.Cu(index)+thick.Cu2O(index)+thick.CuO(index);  
dcdr.CuO.h = dcdrN.CuO.h. /((1 ./ri - 1 ./ro).*ri.^2);  
dcdr.CuO.e = dcdrN.CuO.e. /((1 ./ri - 1 ./ro).*ri.^2);  
dcdr.CuO.vacCuMM = dcdrN.CuO.vacCuMM. /((1 ./ri - 1 ./ro).*ri.^2);  
dcdr.CuO.vacOPP = 0;  
dcdr.CuO.intCuPP = 0;  
end

% site fractions at the Cu/Cu2O interface -- ave volume approach  
numDens.Cu2O.vacCuX = regSite.vacCuX.CuCu2O * rho.Cu2O / MW.Cu2O;  
numDens.Cu2O.vacCuM = regSite.vacCuM.CuCu2O * rho.Cu2O / MW.Cu2O;  
numDens.Cu2O.h = regSite.h.CuCu2O * rho.Cu2O / MW.Cu2O;  
numDens.Cu2O.e = regSite.e.CuCu2O * rho.Cu2O / MW.Cu2O;  
numDens.Cu2O.intCuX = regSite.intCuX.CuCu2O * rho.Cu2O / MW.Cu2O;  
numDens.Cu2O.intCuP = regSite.intCuP.CuCu2O * rho.Cu2O / MW.Cu2O;  
numDens.Cu2O.vacOPP = regSite.vacOPP.CuCu2O * rho.Cu2O / MW.Cu2O;  

if isCuO  
    % Cu2O/CuO interface  
    numDens.CuO.vacCuMM = regSite.vacCuMM.Cu0Cu2O*rho.CuO/MW.CuO; % moles  
    numDens.CuO.vacOPP = regSite.vacOPP.Cu0Cu2O*rho.CuO/MW.CuO; % moles  
    numDens.CuO.intCuPP = regSite.intCuPP.Cu0Cu2O*rho.CuO/MW.CuO; % moles  
    numDens.CuO.h = regSite.h.Cu0Cu2O*rho.CuO/MW.CuO; % moles of i  
    numDens.CuO.e = regSite.e.Cu0Cu2O*rho.CuO/MW.CuO; % moles of i  
end

% =========================================================================
% =========================================================================

% define B- variable; B_i [=] D_i * C_i / (kB*T), J [=] kg-m^2/s^2 = 10^-7 g-m^-2/s^2
% =========================================================================
% =========================================================================
%================================================================================================================================================================
C.2. CERMET PARTICLE MODEL

67 \[ B_{\text{Cu20}.\text{vacCuX}} = D_{\text{eff}.\text{Cu20}.\text{vacCuX}} \cdot \text{numDens.Cu20.vacCuX} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) \]
\[ \rightarrow; \] \%
\[ B_{\text{Cu20}.\text{vacCuM}} = D_{\text{eff}.\text{Cu20}.\text{vacCuM}} \cdot \text{numDens.Cu20.vacCuM} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) \]
\[ \rightarrow; \] \%
\[ B_{\text{Cu20}.\text{vacOPP}} = D_{\text{eff}.\text{Cu20}.\text{vacOPP}} \cdot \text{numDens.Cu20.vacOPP} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) \]
\[ \rightarrow; \] \%
\[ B_{\text{Cu20}.\text{intCuX}} = D_{\text{eff}.\text{Cu20}.\text{intCuX}} \cdot \text{numDens.Cu20.intCuX} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) \]
\[ \rightarrow; \] \%
\[ B_{\text{Cu20}.\text{intCuP}} = D_{\text{eff}.\text{Cu20}.\text{intCuP}} \cdot \text{numDens.Cu20.intCuP} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) \]
\[ \rightarrow; \] \%
\[ B_{\text{Cu20}.h} = D_{\text{eff}.\text{Cu20}.h} \cdot \text{numDens.Cu20.h} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) ; \]
\[ B_{\text{Cu20}.e} = D_{\text{eff}.\text{Cu20}.e} \cdot \text{numDens.Cu20.e} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) ; \]

72 if isCuO
\[ B_{\text{CuO}.\text{vacCuMM}} = D_{\text{eff}.\text{CuO}.\text{vacCuMM}} \cdot \text{numDens.CuO.vacCuMM} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) ; \]
\[ B_{\text{CuO}.\text{vacOPP}} = D_{\text{eff}.\text{CuO}.\text{vacOPP}} \cdot \text{numDens.CuO.vacOPP} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) ; \]
\[ B_{\text{CuO}.\text{intCuPP}} = D_{\text{eff}.\text{CuO}.\text{intCuPP}} \cdot \text{numDens.CuO.intCuPP} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) ; \]
\[ B_{\text{CuO}.h} = D_{\text{eff}.\text{CuO}.h} \cdot \text{numDens.CuO.h} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) ; \]
\[ B_{\text{CuO}.e} = D_{\text{eff}.\text{CuO}.e} \cdot \text{numDens.CuO.e} / (C_n.k_B*\text{sys.temp} \cdot 10^{-7}) ; \]
end

82 \%
83 % Vacancies, ions and intersitals...
84 \%
85 if ~isCuO \% i.e if this only Cu20
86 \%
87 \% define at Cu/Cu20 interface
88 J.Cu20.vacCuM = -D_{\text{eff}.\text{Cu20}.\text{vacCuM}} \cdot \text{dcdr.Cu20.vacCuM} + D_{\text{eff}.\text{Cu20}.e} \cdot \text{dcdr.Cu20.e} - ...
89 \% (kg-m^{-2}/s^{2} \cdot 1000 \ g/kg \cdot 100^{-3} ( \rightarrow cm^{-3}))
90 \% 2*D_{\text{eff}.\text{Cu20}.vacOPP} \cdot \text{dcdr.Cu20.vacOPP})/... ;
91 \% (B.Cu20.intCuP \cdot B.Cu20.vacCuM + B.Cu20.e \cdot B.Cu20.h + 4*...
92 \% \rightarrow B.Cu20.vacOPP);
93 \%
94 \% define at Cu/Cu20 interface
95 J.Cu20.vacCuX = -D_{\text{eff}.\text{Cu20}.\text{vacCuX}} \cdot \text{dcdr.Cu20.vacCuX} ;
96 J.Cu20.intCuP = -D_{\text{eff}.\text{Cu20}.intCuP} \cdot \text{dcdr.Cu20.intCuP} - B.Cu20.intCuP \cdot C_n.e* \rightarrow Cn.Na*\text{phi.Cu20} ;
97 J.Cu20.intCuX = -D_{\text{eff}.\text{Cu20}.intCuX} \cdot \text{dcdr.Cu20.intCuX} ;
98 J.Cu20.vacOPP = -D_{\text{eff}.\text{Cu20}.vacOPP} \cdot \text{dcdr.Cu20.vacOPP} - 2*B.Cu20.vacOPP \cdot C_n.e* \rightarrow Cn.Na*\text{phi.Cu20} ;
99 J.Cu20.h = -D_{\text{eff}.\text{Cu20}.h} \cdot \text{dcdr.Cu20.h} - B.Cu20.h \cdot C_n.e* Cn.Na*\text{phi.Cu20} ;
100 J.Cu20.e = -D_{\text{eff}.\text{Cu20}.e} \cdot \text{dcdr.Cu20.e} + B.Cu20.e \cdot C_n.e* Cn.Na*\text{phi.Cu20} ;

APPENDIX C. MATLAB CODES

else

% Cu/Cu2O interface

\[
\phi_{\text{Cu2O}} = \frac{1}{(C_{n,e} \cdot C_{n,Na}) \cdot ((D_{\text{eff,Cu2O, vacCuM}} \cdot \text{dcdr.Cu2O, vacCuM} + D_{\text{eff,Cu20,e}} \cdot \text{dcdr.Cu2O,e}) - \ldots \\
(D_{\text{eff,Cu20, intCuP}} \cdot \text{dcdr.Cu20, intCuP} + D_{\text{eff,Cu20, h}} \cdot \text{dcdr.Cu20,h}) + \\
2 \cdot D_{\text{eff,Cu20, vacOPP}} \cdot \text{dcdr.Cu20, vacOPP})/ \ldots \\
(B_{\text{Cu20, intCuP}} + B_{\text{Cu20, vacCuM}} + B_{\text{Cu20,e}} + B_{\text{Cu20,h}} + 4*) \\
\rightarrow B_{\text{Cu20, vacOPP}}; \ \% \ (\text{cm}^{-2}/\text{s}) \times (\text{mole of x/cm}^{-3}) / \ (( \\
\rightarrow \text{mole of x}) \times (\text{s}) / (\text{g-cm}^{-3}))
\]

\[
J_{\text{Cu20, vacCuM}} = -D_{\text{eff,Cu20, vacCuM}} \cdot \text{dcdr.Cu20, vacCuM} + B_{\text{Cu20, vacCuM}} \cdot C_{n,e} \times \\
C_{n,Na} \cdot \phi_{\text{Cu20}}; \ \% \ 1/\text{cm} \times (\text{mole of x} \times \text{s} / (\text{g-cm}^{-3})) \times \text{J}
\]

\[
J_{\text{Cu20, vacCuX}} = -D_{\text{eff,Cu20, vacCuX}} \cdot \text{dcdr.Cu20, vacCuX}; \ \% \ \text{mole of x} / \text{cm}^{-2} \cdot \text{s}
\]

\[
J_{\text{Cu20, intCuP}} = -D_{\text{eff,Cu20, intCuP}} \cdot \text{dcdr.Cu20, intCuP} - B_{\text{Cu20, intCuP}} \cdot C_{n,e} \times \\
C_{n,Na} \cdot \phi_{\text{Cu20}}; \ \% \ \text{mole of x} / \text{cm}^{-2} \cdot \text{s}
\]

\[
J_{\text{Cu20, intCuX}} = -D_{\text{eff,Cu20, intCuX}} \cdot \text{dcdr.Cu20, intCuX};
\]

\[
J_{\text{Cu20, vacOPP}} = -D_{\text{eff,Cu20, vacOPP}} \cdot \text{dcdr.Cu20, vacOPP} - 2 \cdot B_{\text{Cu20, vacOPP}} \cdot C_{n,e} \times \\
C_{n,Na} \cdot \phi_{\text{Cu20}};
\]

\[
J_{\text{Cu20, e}} = -D_{\text{eff,Cu20, e}} \cdot \text{dcdr.Cu20, e} + B_{\text{Cu20,e}} \cdot C_{n,e} \times C_{n,Na} \cdot \phi_{\text{Cu20}};
\]

% CuO / O2 interface

\[
\phi_{\text{CuO}} = \frac{1}{(C_{n,e} \cdot C_{n,Na}) \cdot ((2 \cdot D_{\text{eff,CuO, vacCuMM}} \cdot \text{dcdr.Cu0, vacCuMM} + D_{\text{eff,Cu0, e}} \cdot \text{dcdr.Cu0,e}) - \ldots \\
(2 \cdot D_{\text{eff,Cu0, intCuPP}} \cdot \text{dcdr.Cu0, intCuPP} + D_{\text{eff,Cu0, h}} \cdot \text{dcdr.Cu0,h}) + \\
2 \cdot D_{\text{eff,Cu0, vacOPP}} \cdot \text{dcdr.Cu0, vacOPP})/ \ldots
\]
(4* B.CuO.intCuPP + 4* B.CuO.vacCuMM + B.CuO.e + B.CuO.h + 4* 
\rightarrow B.CuO.vacOPP);

J.CuO.vacCuMM = - D_eff.CuO.vacCuMM * dcdr.CuO.vacCuMM + 2* B.CuO.vacCuMM * Cn.e * 
\rightarrow Cn.Na*phi.CuO;

J.CuO.intCuPP = - D_eff.CuO.intCuPP * dcdr.CuO.intCuPP + 2* B.CuO.intCuPP * Cn.e * 
\rightarrow Cn.Na*phi.CuO;

J.CuO.vacOPP = - D_eff.CuO.vacOPP * dcdr.CuO.vacOPP - 2* B.CuO.vacOPP * Cn.e * 
\rightarrow Cn.Na*phi.CuO;


\% phi [\text{cm}^{-2} * g / (A-s^{-3})
\% B [\text{mole} * s / (g-cm^{-3})
\% J [\text{moles/cm}^{-2-s}
\% \text{no more Cu}
\% CuIsDone = find(real(CuCheck(index))<1e-3);
noCuIndex = index(CuAtomFrac(index)<=2e-6);
if ~isempty(noCuIndex)
  \% define J.Cu2O relative to index not in the absolute vector sense so won't 
  \rightarrow always be
  \% find where CuCheck overlaps with index
  J.Cu2O.vacCuM(noCuIndex) = 0;
  J.Cu2O.vacCuX(noCuIndex) = 0;
  J.Cu2O.intCuP(noCuIndex) = 0;
  J.Cu2O.intCuX(noCuIndex) = 0;
  J.Cu2O.vacOPP(noCuIndex) = 0;
  J.Cu2O.h(noCuIndex) = 0;
  J.Cu2O.e(noCuIndex) = 0;
end

\% reactions done if Cu2O thickness is 0 or less than 0...
\% Cu2OIsDone = find(abs(Cu2OCheck(index))<1e-3);
noCu2OIndex = index(Cu2OAtomFrac(index)<=2e-6); \% if the mole frac of Cu2O is less 
\rightarrow than 1e-3 than stop all rx in that shell
if ~isempty(noCu2OIndex)
  \% noCu2OIndex =[];
  \% jn = 0;
  \% for i = 1:length(Cu2OCheck)
  \%  tempIndex = find(Cu2OCheck(i)==index);
  \%  if ~isempty(tempIndex)
  \%    noCu2OIndex(jn+1) = tempIndex;
  \%  jn = jn+1;
APPENDIX C. MATLAB CODES

```matlab
% end

J.CuO.vacCuMM (noCu2OIndex) =0;
J.CuO.intCuPP (noCu2OIndex) =0;
J.CuO.vacOPP (noCu2OIndex) =0;
J.CuO.e (noCu2OIndex) =0;
J.CuO.h (noCu2OIndex) =0;

J.Cu2O.vacCuM (noCu2OIndex) = 0;
J.Cu2O.vacCuX (noCu2OIndex) = 0;
J.Cu2O.intCuP (noCu2OIndex) = 0;
J.Cu2O.intCuX (noCu2OIndex) = 0;
J.Cu2O.vacOPP (noCu2OIndex) = 0;
J.Cu2O.h (noCu2OIndex) = 0;
J.Cu2O.e (noCu2OIndex) = 0;

end

%========================================================================
% ===== CALCULATE FULXES OF EACH SPECIES =================================
%========================================================================


if isCuO
    neutralityCond.CuO = 2* J.Cu0.intCuPP + 2* J.Cu0.vacOPP + J.Cu0.h -(2* J.Cu0.vacCuMM + J.Cu0.e);
end

if sum ( abs ( neutralityCond.CuO )) >1e-8
    disp ('neutrality not met...')
    zeroV = find ( abs ( neutralityCond.CuO ))>1e-8);

    J.Cu0.vacCuMM (zeroV) =0;
    J.Cu0.intCuPP (zeroV) =0;
    J.Cu0.vacOPP (zeroV) =0;
    J.Cu0.e (zeroV) =0;
    J.Cu0.h (zeroV) =0;
    J.Cu20.vacCuM (zeroV) = 0;
    J.Cu20.vacCuX (zeroV) = 0;
    J.Cu20.intCuP (zeroV) = 0;
    J.Cu20.intCuX (zeroV) = 0;
    J.Cu20.vacOPP (zeroV) = 0;
```

C.2. CERMET PARTICLE MODEL

207 J.Cu2O.h(zeroV) = 0;
208 J.Cu2O.e(zeroV) = 0;
209 end

210 if sum(abs(neutralityCond.Cu2O))>1e-8
211     disp('neutrality not met...')
212     % if not met set to zero...
213     zeroV = find(abs(neutralityCond.Cu2O)>1e-8);
214     J.Cu20.vacCuM(zeroV) = 0;
215     J.Cu20.vacCuX(zeroV) = 0;
216     J.Cu20.intCuP(zeroV) = 0;
217     J.Cu20.intCuX(zeroV) = 0;
218     J.Cu20.vacOPP(zeroV) = 0;
219     J.Cu20.h(zeroV) = 0;
220     J.Cu20.e(zeroV) = 0;
221 end

222 % [ units ]
223 % D [ = ] m^-2/s
224 % dcdr [ = ] atoms/m^-4
225 % B [ = ] CD/kbT [ = ] (atoms/m^-3 * m^-2/s) / (J/atom)
226 % B [ = ] CD/kbT [ = ] (atoms^-2/m^-s-J)
227 % e*z*V/m [ = ] columbs/atom * V/m = J/(atom*m
228 % ========================================================================
229 % ========================================================================
230 % net flux.... contributing to Cu2O growth...
231 % J.Cu.net = J.Cu20.intCuX + J.Cu20.intCuP - (J.Cu20.vacCuX + J.Cu20.vacCuM
233 % J.Cu2O.net = J.Cu20.intCuX + J.Cu20.intCuP - (J.Cu20.vacCuX + J.Cu20.vacCuM
235 % should dN_Cu = - dN_Cu2O.... when only two species
236 % yes, because if Cu decreases dN_Cu2O must increase... where else where the
237 % copper go? the concentration profile will not be linear in a spherical
238 % particle...
239 % sign of fluxes...
240 % J.Cu20.e > 0
241 % J.Cu20.h < 0
242 % J.Cu20.vacOPP > 0
243 % J.Cu20.vacCuX < 0
244 % J.Cu20.intCuX > 0
245 % J.Cu20.intCuP > 0
246 % J.Cu0.e > 0


```matlab
% J.Cu0.h < 0
% J.Cu0.vacOPP > 0
% J.Cu0.vacCuMM < 0
% J.Cu0.intCuPP > 0
try
    dnCu = ( J.Cu2O.intCuX + J.Cu2O.intCuP ) .* AoR.Cu2O ... 
    - ( J.Cu2O.vacCuX + J.Cu2O.vacCuM ) .* AoR.Cu2O ... 
    + 2 * J.Cu2O.vacOPP .* AoR.Cu2O;
catch
    a = 1;
    dnCu = 0;
end

dnCu2O = dnCu / 2;

outPut.Cu = -dnCu;
if ~isCuO
    outPut.Cu2O = dnCu2O;
else
    % I define the change of CuO as any copper coming from Cu2O
    % I define the change of Cu2O as requiring a Cu-O pair from CuO
    % this term represents copper atoms leaving Cu2O and entering CuO
    dnCu0 = J.Cu0.intCuPP .* AoR.Cu0 - J.Cu0.vacCuMM .* AoR.Cu0 ... 
    + J.Cu0.vacOPP .* AoR.Cu0;
    % NOTE: Copper atoms only move outward
    % Oxygen atoms only move inward
    % when there is a Cu Core
    % --> Cu-O atom pairs go from CuO to Cu2O
    % --> Cu atoms jump from Cu2O to CuO
    outPut.Cu2O = dnCu2O - dnCu0 / 2; % growth of CuO leads to destruction of 
                                      % 1/2 a Cu2O, 1/s
    outPut.CuO = dnCu0; % growth of Cu2O leads to a decrease of Cu0
end

% =========================================================================
% NoRx = find ( PO2Atm ( index ) < sys.presCu2OCuO_atm ); % if true, all below this limit 
% and all shells should be zero
% if ~isempty ( NoRx )
%    outPut.Cu0 ( NoRx ) = 0;
%    outPut.Cu2O ( NoRx ) = 0;
%    outPut.Cu ( NoRx ) = 0;
% end
```

---

```matlab`
% NoRx = find ( PO2Atm ( index ) < sys.presCu2OCu0_atm ); % if true, all below this limit 
% and all shells should be zero
% if ~isempty ( NoRx )
%    outPut.Cu0 ( NoRx ) = 0;
%    outPut.Cu2O ( NoRx ) = 0;
%    outPut.Cu ( NoRx ) = 0;
% end
```
% numDens.Cu2O.vacCuX = (regSite.vacCuX.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles of i per cc
% numDens.Cu2O.vacCuM = (regSite.vacCuM.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles of i per cc
% numDens.Cu2O.vacOPP = (regSite.vacOPP.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles of i per cc
% numDens.Cu2O.intCuX = (regSite.intCuX.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles of i per cc
% numDens.Cu2O.intCuP = (regSite.intCuP.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles of i per cc
% numDens.Cu2O.h = (regSite.h.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles of i per cc
% numDens.Cu2O.e = (regSite.e.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles of i per cc

% if isCuO
% Cu2O/CuO interface
% numDens.CuO.vacCuMM = (regSite.vacCuMM.CuOCu2O)*(rho.CuO/MW.CuO);
% moles of i per cc
% numDens.CuO.vacOPP = (regSite.vacOPP.CuOCu2O)*(rho.CuO/MW.CuO);
% moles of i per cc
% numDens.CuO.intCuPP = (regSite.intCuPP.CuOCu2O)*(rho.CuO/MW.CuO);
% moles of i per cc
% numDens.CuO.h = (regSite.h.CuOCu2O)*(rho.CuO/MW.CuO);
% moles of i per cc
% numDens.CuO.e = (regSite.e.CuOCu2O)*(rho.CuO/MW.CuO);
% moles of i per cc
% end

% % define B-variable; B_i [=] D_i * C_i / (kB*T), J [=} kg-m^2/s^2
% % B.Cu2O.vacCuX = D_eff.Cu2O.vacCuX.*numDens.Cu2O.vacCuX/(Cn.kB*sys.temp*10^-7); % (cm^-2/s) * (mole of x/cm^-3) / (g-cm^-2/s^-2)
C.2. CERMET PARTICLE MODEL

\[ B.Cu20.vacCuM = D_{\text{eff}}.Cu20.vacCuM \times \text{numDens}.Cu20.vacCuM/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.Cu20.vacOPP = D_{\text{eff}}.Cu20.vacOPP \times \text{numDens}.Cu20.vacOPP/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.Cu20.intCuX = D_{\text{eff}}.Cu20.intCuX \times \text{numDens}.Cu20.intCuX/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.Cu20.intCuP = D_{\text{eff}}.Cu20.intCuP \times \text{numDens}.Cu20.intCuP/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.Cu2O.vacCuMM = D_{\text{eff}}.CuO.vacCuMM \times \text{numDens}.CuO.vacCuMM/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.CuO.vacOPP = D_{\text{eff}}.CuO.vacOPP \times \text{numDens}.CuO.vacOPP/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.CuO.intCuPP = D_{\text{eff}}.CuO.intCuPP \times \text{numDens}.CuO.intCuPP/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.CuO.h = D_{\text{eff}}.CuO.h \times \text{numDens}.CuO.h/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.CuO.e = D_{\text{eff}}.CuO.e \times \text{numDens}.CuO.e/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ \text{if isCuO} \]

\[ B.Cu0.vacCuMM = D_{\text{eff}}.CuO.vacCuMM \times \text{numDens}.CuO.vacCuMM/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.CuO.vacOPP = D_{\text{eff}}.CuO.vacOPP \times \text{numDens}.CuO.vacOPP/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.CuO.intCuPP = D_{\text{eff}}.CuO.intCuPP \times \text{numDens}.CuO.intCuPP/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.CuO.h = D_{\text{eff}}.CuO.h \times \text{numDens}.CuO.h/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ B.CuO.e = D_{\text{eff}}.CuO.e \times \text{numDens}.CuO.e/(Cn.kB \times \text{sys}.temp \times 10^{-7}); \%
\]

\[ \text{end} \%
\]

% Vacancies, ions and interstitials...

% % % calculate derivatives of mid-points: dc/dr [=] moles/m, d2c/dr^2 = moles/m^2

% % if ~aSphere

% % if ~isCuO

% % dcdr.Cu20.vacCuX = (regSite.vacCuX.Cu202(index)-regSite.vacCuX.CuCu20)/(thick.Cu20(index))*(rho.Cu20/MW.Cu20); % moles/cm

% % dcdr.Cu20.vacCuM = (regSite.vacCuM.Cu202(index)-regSite.vacCuM.CuCu20)/(thick.Cu20(index))*(rho.Cu20/MW.Cu20); % moles/cm

% % dcdr.Cu20.vacOPP = (regSite.vacOPP.Cu202(index)-regSite.vacOPP.CuCu20)/(thick.Cu20(index))*(rho.Cu20/MW.Cu20); % moles/cm

% % dcdr.Cu20.intCuX = (regSite.intCuX.Cu202(index)-regSite.intCuX.CuCu20)/(thick.Cu20(index))*(rho.Cu20/MW.Cu20); % moles/cm
% dcdr.Cu2O.intCuP = (regSite.intCuP.Cu2OO2(index) - regSite.intCuP.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3

% dcdr.Cu2O.h = (regSite.h.Cu2OO2(index) - regSite.h.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3

% dcdr.Cu2O.e = (regSite.e.Cu2OO2(index) - regSite.e.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3


% J.Cu2O.vacCuM = -(D_eff.Cu2O.vacCuM*dcdr.Cu2O.vacCuM - B.Cu2O.vacCuM*Cn.e*Cn.Na*phi.Cu2O);

% J.Cu2O.vacCuX = -D_eff.Cu2O.vacCuX*dcdr.Cu2O.vacCuX;

% J.Cu2O.intCuP = -(D_eff.Cu2O.intCuP*dcdr.Cu2O.intCuP + B.Cu2O.intCuP*Cn.e*Cn.Na*phi.Cu2O);

% J.Cu2O.intCuX = -D_eff.Cu2O.intCuX*dcdr.Cu2O.intCuX;

% J.Cu2O.vacOPP = -(D_eff.Cu2O.vacOPP*dcdr.Cu2O.vacOPP + 2*B.Cu2O.vacOPP*Cn.e*Cn.Na*phi.Cu2O);

% J.Cu2O.h = -(D_eff.Cu2O.h*dcdr.Cu2O.h + B.Cu2O.h*Cn.e*CNa*phi.Cu2O);

% J.Cu2O.e = -(D_eff.Cu2O.e*dcdr.Cu2O.e + B.Cu2O.e*Cn.e*Cn.Na*phi.Cu2O);

% else

% deltaC.Cu2O.vacCuX = (regSite.vacCuX.Cu2OCuO - regSite.vacCuX.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3

% deltaC.Cu2O.vacCuM = (regSite.vacCuM.Cu2OCuO - regSite.vacCuM.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3

% deltaC.Cu2O.vacOPP = (regSite.vacOPP.Cu2OCuO - regSite.vacOPP.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3

% deltaC.Cu2O.intCuX = (regSite.intCuX.Cu2OCuO - regSite.intCuX.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3

% deltaC.Cu2O.intCuP = (regSite.intCuP.Cu2OCuO - regSite.intCuP.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3

% deltaC.Cu2O.h = (regSite.h.Cu2OCuO - regSite.h.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3

% deltaC.Cu2O.e = (regSite.e.Cu2OCuO - regSite.e.CuCu2O)/(thick.Cu2O(index))*(rho.Cu2O/MW.Cu2O); % moles/cm^-3
C2. CERMET PARTICLE MODEL

\[ \phi_{Cu2O} = \frac{1}{(Cn.e \cdot Cn.Na)} \cdot \left( \frac{D_{eff}.Cu2O.vacCuM \cdot \Delta C.Cu2O.vacCuM + D_{eff}.Cu2O.e \cdot \Delta C.Cu2O.e - \ldots}{(B.Cu2O.intCuP + B.Cu2O.vacCuM + B.Cu2O.e + \ldots) \cdot (B.Cu20.h + 4 \cdot B.Cu20.vacOPP)} \right) \]

\[ J_{Cu2O.vacCuM} = -(D_{eff}.Cu2O.vacCuM \cdot \Delta C.Cu2O.vacCuM + B.Cu20.vacCuM \cdot Cn.e \cdot Cn.Na \cdot \phi_{Cu2O}) \]

\[ J_{Cu20.vacCuX} = -D_{eff}.Cu20.vacCuX \cdot \Delta C.Cu20.vacCuX; \]

\[ J_{Cu20.intCuP} = -(D_{eff}.Cu20.intCuP \cdot \Delta C.Cu20.intCuP - \ldots) \cdot (B.Cu20.vacOPP + Cn.e \cdot Cn.Na \cdot \phi_{Cu20}) \]

\[ J_{Cu20.h} = -(D_{eff}.Cu20.h \cdot \Delta C.Cu20.h + B.Cu20.h \cdot \ldots) \cdot (Cn.e \cdot Cn.Na \cdot \phi_{Cu20}) \]

\[ J_{Cu20.e} = -(D_{eff}.Cu20.e \cdot \Delta C.Cu20.e - B.Cu20.e \cdot \ldots) \cdot (Cn.e \cdot Cn.Na \cdot \phi_{Cu20}) \]

\[ \delta C.CuO.vacCuMM = \frac{\text{regSite.vacCuMM.CuOO2(index)} - \text{regSite.vacCuMM.CuOCu2O}}{\text{thick.CuO(index)} \cdot \rho.CuO / MW.CuO}; \text{ moles/cm}^3 \]

\[ \delta C.CuO.vacOPP = \frac{\text{regSite.vacOPP.CuOO2(index)} - \text{regSite.vacOPP.CuOCu2O}}{\text{thick.CuO(index)} \cdot \rho.CuO / MW.CuO}; \text{ moles/cm}^3 \]

\[ \delta C.CuO.intCuPP = \frac{\text{regSite.intCuPP.CuOO2(index)} - \text{regSite.intCuPP.CuOCu2O}}{\text{thick.CuO(index)} \cdot \rho.CuO / MW.CuO}; \text{ moles/cm}^3 \]

\[ \delta C.CuO.h = \frac{\text{regSite.h.CuOO2(index) - regSite.h.CuOCu2O}}{\text{thick.CuO(index)} \cdot \rho.CuO / MW.CuO}; \text{ moles/cm}^3 \]

\[ \delta C.CuO.e = \frac{\text{regSite.e.CuOO2(index) - regSite.e.CuOCu2O}}{\text{thick.CuO(index)} \cdot \rho.CuO / MW.CuO}; \text{ moles/cm}^3 \]

\[ \phi_{CuO} = \frac{1}{(Cn.e \cdot Cn.Na)} \cdot \left( \frac{2 \cdot D_{eff}.CuO.vacCuMM \cdot \Delta C.CuO.vacCuMM + D_{eff}.CuO.e \cdot \Delta C.CuO.e - \ldots}{(2 \cdot B.CuO.intCuPP + 4 \cdot B.CuO.vacOPP + \ldots) \cdot (4 \cdot B.CuO.intCuPP + 4 \cdot B.CuO.vacCuMM + B.CuO.e + \ldots) \cdot (B.CuO.h + 4 \cdot B.CuO.vacOPP)} \right) \]

\[ J_{CuO.vacCuMM} = -(D_{eff}.CuO.vacCuMM \cdot \Delta C.CuO.vacCuMM + 2 \cdot \ldots) \cdot (B.CuO.vacCuMM + Cn.e \cdot Cn.Na \cdot \phi_{CuO}) \]
% J.CuO.intCuPP = -(D_eff.CuO.intCuPP*deltaC.CuO.intCuPP - 2*B.CuO.intCuPP*Cn.e*Cn.Na*phi.CuO);
% J.CuO.vacOPP = -(D_eff.CuO.vacOPP*deltaC.CuO.vacOPP - 2*B.CuO.vacOPP*Cn.e*Cn.Na*phi.CuO);
% J.CuO.h = -(D_eff.CuO.h*deltaC.CuO.h - B.CuO.h*Cn.e*Cn.Na*phi.CuO);
% J.CuO.e = -(D_eff.CuO.e*deltaC.CuO.e + B.CuO.e*Cn.e*Cn.Na*phi.CuO);
% end

elseif aSphere

% Cu/Cu2O interface

deltaC.Cu2O.vacCuX = (regSite.vacCuX.Cu2OCuO - regSite.vacCuX.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles/cm^3

deltaC.Cu2O.vacCuM = (regSite.vacCuM.Cu2OCuO - regSite.vacCuM.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles/cm^3

deltaC.Cu2O.vacOPP = (regSite.vacOPP.Cu2OCuO - regSite.vacOPP.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles/cm^3

deltaC.Cu2O.intCuX = (regSite.intCuX.Cu2OCuO - regSite.intCuX.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles/cm^3

deltaC.Cu2O.intCuP = (regSite.intCuP.Cu2OCuO - regSite.intCuP.CuCu2O)*(rho.Cu2O/MW.Cu2O);
% moles/cm^3

deltaC.Cu20.h = (regSite.h.Cu20CuO - regSite.h.CuCu2O)*(
  rho.Cu20/MW.Cu20);
% moles/cm^3

deltaC.Cu20.e = (regSite.e.Cu20CuO - regSite.e.CuCu2O)*(
  rho.Cu20/MW.Cu20);
% moles/cm^3

phi.Cu20 = 1/(Cn.e*Cn.Na)*((D_eff.Cu20.vacCuM*deltaC.Cu20.vacCuM
  + D_eff.Cu20.e*deltaC.Cu20.e) - ...)

% (B.Cu20.intCuP + B.Cu20.vacCuM + B.Cu20.e + B.Cu20.h
  + 4*B.Cu20.vacOPP); % (cm^-2/s) * (mole of x/cm^3) / ((mole of x) * (s
  -> ) / (g-cm^3))
g = cm^-2/s^2 [=] J
\[ \text{AoR.Cu2O} = 4\pi \cdot (\text{thick.Cu(index)})^2; \quad \% \text{area at Cu/Cu2O} \]

\[ \text{J.Cu2O.vacCuM} = -\text{const} \cdot (\text{D_eff.Cu2O.vacCuM} \cdot \text{deltaC.Cu2O.vacCuM} - \text{B.Cu2O.vacCuM} \cdot \text{Cn.e} \cdot \text{Cn.Na} \cdot \text{phi.Cu2O}); \quad \% \text{cm}^{-2} \text{ s}^{-1} \text{ mole of x} / (\text{g-cm}^{-3}) \]

\[ \text{J.Cu2O.vacCuX} = -\text{const} \cdot \text{D_eff.Cu2O.vacCuX} \cdot \text{deltaC.Cu2O.vacCuX}; \quad \% \text{mole of x} / \text{cm}^2 \text{ s}^{-1} \text{ mole of x} / (\text{g-cm}^{-3}) \]

\[ \text{J.Cu2O.intCuP} = -\text{const} \cdot (\text{D_eff.Cu2O.intCuP} \cdot \text{deltaC.Cu2O.intCuP} + \text{B.Cu2O.intCuP} \cdot \text{Cn.e} \cdot \text{Cn.Na} \cdot \text{phi.Cu2O}); \quad \% \text{mole of x} / \text{cm}^2 \text{ s}^{-1} \text{ mole of x} / (\text{g-cm}^{-3}) \]

\[ \text{J.Cu2O.h} = -\text{const} \cdot (\text{D_eff.Cu2O.h} \cdot \text{deltaC.Cu2O.h} + \text{B.Cu2O.h} \cdot \text{Cn.e} \cdot \text{Cn.Na} \cdot \text{phi.Cu2O}); \quad \% \text{mole of x} / \text{cm}^2 \text{ s}^{-1} \text{ mole of x} / (\text{g-cm}^{-3}) \]

\[ \text{J.Cu2O.e} = -\text{const} \cdot (\text{D_eff.Cu2O.e} \cdot \text{deltaC.Cu2O.e} - \text{B.Cu2O.e} \cdot \text{Cn.e} \cdot \text{Cn.Na} \cdot \text{phi.Cu2O}); \quad \% \text{mole of x} / \text{cm}^2 \text{ s}^{-1} \text{ mole of x} / (\text{g-cm}^{-3}) \]

\[ \text{J.Cu2O.vacOPP} = -\text{const} \cdot (\text{D_eff.Cu2O.vacOPP} \cdot \text{deltaC.Cu2O.vacOPP} + 2 \cdot \text{B.Cu2O.vacOPP} \cdot \text{Cn.e} \cdot \text{Cn.Na} \cdot \text{phi.Cu2O}); \quad \% \text{mole of x} / \text{cm}^2 \text{ s}^{-1} \text{ mole of x} / (\text{g-cm}^{-3}) \]

\[ \text{phi.CuO} = 1/(\text{Cn.e} \cdot \text{Cn.Na}) \cdot ((2 \cdot \text{D_eff.CuO.vacCuMM} \cdot \text{deltaC.CuO.vacCuMM} + \text{D_eff.CuO.e} \cdot \text{deltaC.CuO.e}) - ...) \]

\[ \text{deltaC.CuO.vacCuMM} = \text{regSite.vacCuMM.CuOCu20} \cdot (\text{rho.CuO} / \text{MW.CuO}); \quad \% \text{mole of x} / \text{cm}^{-3} \]

\[ \text{deltaC.CuO.vacOPP} = \text{regSite.vacOPP.CuOCu20} \cdot (\text{rho.CuO} / \text{MW.CuO}); \quad \% \text{mole of x} / \text{cm}^{-3} \]

\[ \text{deltaC.CuO.intCuPP} = \text{regSite.intCuPP.CuOCu20} \cdot (\text{rho.CuO} / \text{MW.CuO}); \quad \% \text{mole of x} / \text{cm}^{-3} \]

\[ \text{deltaC.CuO.h} = \text{regSite.h.CuOCu20} \cdot (\text{rho.CuO} / \text{MW.CuO}); \quad \% \text{mole of x} / \text{cm}^{-3} \]

\[ \text{deltaC.CuO.e} = \text{regSite.e.CuOCu20} \cdot (\text{rho.CuO} / \text{MW.CuO}); \quad \% \text{mole of x} / \text{cm}^{-3} \]

\[ \text{phi.CuO} = 1/(\text{Cn.e} \cdot \text{Cn.Na}) \cdot ((2 \cdot \text{D_eff.CuO.vacCuMM} \cdot \text{deltaC.CuO.vacCuMM} + \text{D_eff.CuO.e} \cdot \text{deltaC.CuO.e}) - ...) \]

\[ \text{J.CuO.vacCuMM} = -\text{const} \cdot (\text{D_eff.CuO.vacCuMM} \cdot \text{deltaC.CuO.vacCuMM} - 2 \cdot \text{B.CuO.vacCuMM} \cdot \text{Cn.e} \cdot \text{Cn.Na} \cdot \text{phi.Cu20}); \quad \% \text{cm}^{-2} \text{ s}^{-1} \text{ mole of x} / (\text{g-cm}^{-3}) \]
% % J.Cu0.intCuPP = - const.* (D_eff.Cu0.intCuPP*deltaC.Cu0.intCuPP + 2*B.Cu0.intCuPP * Cn.e*Cn.Na*phi.Cu0);% % J.Cu0.vacOPP = - const.* (D_eff.Cu0.vacOPP*deltaC.Cu0.vacOPP + 2*B.Cu0.vacOPP * Cn.e*Cn.Na*phi.Cu0);% % J.Cu0.h = - const.* (D_eff.Cu0.h*deltaC.Cu0.h + B.Cu0.h * Cn.e*Cn.Na*phi.Cu0);% % else% % deltaC.Cu2O.vacCuX = ( regSite.vacCuX.Cu2OO2 ( index )-regSite.vacCuX.CuCu2O )*( rho.Cu2O / MW.Cu2O ); % moles /cm ^3% % deltaC.Cu2O.vacCuM = ( regSite.vacCuM.Cu2OO2 ( index )-regSite.vacCuM.CuCu2O )*( rho.Cu2O / MW.Cu2O ); % moles /cm ^3% % deltaC.Cu2O.vacOPP = ( regSite.vacOPP.Cu2OO2 ( index )-regSite.vacOPP.CuCu2O )*( rho.Cu2O / MW.Cu2O ); % moles /cm ^3% % deltaC.Cu2O.intCuX = ( regSite.intCuX.Cu2OO2 ( index )-regSite.intCuX.CuCu2O )*( rho.Cu2O / MW.Cu2O ); % moles /cm ^3% % deltaC.Cu2O.intCuP = ( regSite.intCuP.Cu2OO2 ( index )-regSite.intCuP.CuCu2O )*( rho.Cu2O / MW.Cu2O ); % moles /cm ^3% % deltaC.Cu2O.h = ( regSite.h.Cu2OO2 ( index )-regSite.h.CuCu2O )* ( rho.Cu2O / MW.Cu2O ); % moles /cm ^3% % deltaC.Cu2O.e = ( regSite.e.Cu2OO2 ( index )-regSite.e.CuCu2O )* ( rho.Cu2O / MW.Cu2O ); % moles /cm ^3% % phi.Cu2O = 1/( Cn.e * Cn.Na ) * (( D_eff.Cu2O.vacCuM * deltaC.Cu2O.vacCuM + D_eff.Cu2O.vacCuX * deltaC.Cu2O.vacCuX + D_eff.Cu2O.intCuP * deltaC.Cu2O.intCuP + D_eff.Cu2O.h * deltaC.Cu2O.h + 2*D_eff.Cu2O.vacOPP*deltaC.Cu2O.vacOPP )/ ...% % ( B.Cu2O.intCuP + B.Cu2O.vacCuM + B.Cu2O.e + B.Cu2O.h ) + 4*B.Cu2O.vacOPP);% %% % # define at Cu/Cu2O interface% % AoR.Cu2O = 4*pi*( thick.Cu(index) ).^-2; % area at Cu/Cu2O% % interface...% % const = 4*pi./AoR.Cu20.*((thick.Cu(index)).*(thick.Cu(index)+thick.Cu20(index)))/(thick.Cu20(index));% % J.Cu20.vacCuM = - const.* (D_eff.Cu20.vacCuM*deltaC.Cu20.vacCuM - B.Cu20.vacCuM * Cn.e*Cn.Na*phi.Cu20);% % J.Cu20.vacCuX = - const.* (D_eff.Cu20.vacCuX*deltaC.Cu20.vacCuX);% % J.Cu20.intCuP = - const.* (D_eff.Cu20.intCuP*deltaC.Cu20.intCuP + B.Cu20.intCuP * Cn.e*Cn.Na*phi.Cu20);% % J.Cu20.intCuX = - const.* (D_eff.Cu20.intCuX*deltaC.Cu20.intCuX);% % J.Cu20.vacOPP = - const.* (D_eff.Cu20.vacOPP*deltaC.Cu20.vacOPP + 2*B.Cu20.vacOPP * Cn.e*Cn.Na*phi.Cu20);% % J.Cu20.h = - const.* (D_eff.Cu20.h*deltaC.Cu20.h + B.Cu20.h * Cn.e*Cn.Na*phi.Cu20);% % J.Cu20.e = - const.* (D_eff.Cu20.e*deltaC.Cu20.e - B.Cu20.e * Cn.e*Cn.Na*phi.Cu20);
C.2. CERMET PARTICLE MODEL

\[ J_{\text{Cu2O}.e} = -\text{const.} \times (D_{\text{eff}.\text{Cu2O}.e} \delta C_{\text{Cu2O}.e} - B_{\text{Cu2O}.e} C_{n.e} C_{n.Na} \phi_{\text{Cu2O}}); \]

\[ \text{CuDone} = \text{find}(\text{real} \left( \text{CuCheck}(\text{index}) \right) < 1e^{-3}); \]

\[ \text{if} \; \neg \text{isempty} \left( \text{CuDone} \right) \]
\[ J_{\text{Cu20}.\text{vacCuM}(\text{CuDone})} = 0; \]
\[ J_{\text{Cu20}.\text{vacCuX}(\text{CuDone})} = 0; \]
\[ J_{\text{Cu20}.\text{intCuP}(\text{CuDone})} = 0; \]
\[ J_{\text{Cu20}.\text{intCuX}(\text{CuDone})} = 0; \]
\[ J_{\text{Cu20}.\text{vacOPP}(\text{CuDone})} = 0; \]
\[ J_{\text{Cu20}.h(\text{CuDone})} = 0; \]
\[ J_{\text{Cu20}.e(\text{CuDone})} = 0; \]
\[ \text{disp('CuDone')} \]

\[ \text{Cu20IsDone} = \text{find}(\text{abs} \left( \text{Cu20Check}(\text{index}) \right) < 1e^{-3}); \]

\[ \text{if} \; \neg \text{isempty} \left( \text{Cu20Done} \right) \]
\[ J_{\text{Cu0}.\text{vacCuMM}(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu0}.\text{intCuPP}(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu0}.\text{vacOPP}(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu0}.e(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu0}.h(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu20}.\text{vacCuM}(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu20}.\text{vacCuX}(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu20}.\text{intCuP}(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu20}.\text{intCuX}(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu20}.\text{vacOPP}(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu20}.h(\text{Cu20IsDone})} = 0; \]
\[ J_{\text{Cu20}.e(\text{Cu20IsDone})} = 0; \]

\[ \text{end} \]

\% ===== CALCULATE FULXES OF EACH SPECIES

\[ \text{---} \]

\[ \text{---} \]

\[ \text{---} \]
Flux equations \([=] \text{moles/cm}^2\text{-s}

\[
\text{neutralityCond.Cu2O} = \text{J.Cu2O.intCuP} + 2\times \text{J.Cu2O.vacOPP} + \text{J.Cu2O.h} - \left( \text{J.Cu2O.vacCuM} + \text{J.Cu2O.e} \right)
\]

\[
\text{neutralityCond.CuO} = 0;
\]

\[
\text{if} \ \text{isCuO}
\]

\[
\text{neutralityCond.CuO} = 2\times \text{J.CuO.intCuPP} + 2\times \text{J.CuO.vacOPP} + \text{J.CuO.h} - (2\times \left( \text{J.CuO.vacCuMM} + \text{J.CuO.e} \right)
\]

\[
\text{end}
\]

\[
\text{if} \ \sum (\abs{\text{neutralityCond.Cu2O}}) > 1\times 10^{-8} || \sum (\abs{\text{neutralityCond.CuO}}) > 1\times 10^{-8}
\]

\[
\text{disp('neutrality not met...')}
\]

\[
\text{end}
\]

\[
\text{[units]}
\]

\[
\% D \ [=] \text{m}^{-2}/\text{s}
\]

\[
\% \text{dcr} \ [=] \text{atoms/m}^{-4}
\]

\[
\% \text{B} \ [=] \text{CD/}k_B T \ [=] (\text{atoms/m}^{-3} \times \text{m}^{-2}/\text{s}) / (\text{J/atom})
\]

\[
\% \text{B} \ [=] \text{CD/}k_B T \ [=] (\text{atoms}^{-2}/\text{m-s-J})
\]

\[
\% e/z*V/m \ [=] \text{coulombs/atom} \times V/m = \text{J/(atom-m}
\]

\[
\% \text{sign of fluxes...}
\]

\[
\% \text{J.Cu2O.e} > 0
\]

\[
\% \text{J.Cu2O.h} < 0
\]

\[
\% \text{J.Cu2O.vacOPP} > 0
\]

\[
\% \text{J.Cu2O.vacCuX} < 0
\]

\[
\% \text{J.Cu2O.vacCuM} < 0
\]
% J.Cu2O.intCuX > 0
% J.Cu2O.intCuM > 0
% J.CuO.e > 0
% J.CuO.h < 0
% J.CuO.vacOPP > 0
% J.CuO.vacCuMM < 0
% J.CuO.intCuPP > 0

% externalArea = 4*pi*(thick.Cu(index) + thick.Cu2O(index) + thick.CuO(index)).^2; % area at Cu2O/CuO interface...

% if aSphere

% dnCu = (J.Cu2O.intCuX + J.Cu2O.intCuP).*AoR.Cu2O ...
% -(J.Cu2O.vacCuX + J.Cu2O.vacCuM).*AoR.Cu2O...
% + 2*J.Cu2O.vacOPP.*AoR.Cu2O;

% dnCu2O = dnCu/2;

% outPut.Cu = -dnCu;
% if ~isCuO
%   outPut.Cu2O = dnCu2O;
% else
%   % I define the change of CuO as any copper coming from Cu2O
%   % I define the change of Cu2O as requireing a Cu-O pair from CuO
%   % this term represents copper atoms leaving Cu2O and entering CuO
%   dnCuO = J.CuO.intCuPP.*AoR.CuO-J.CuO.vacCuMM.*AoR.CuO ...
%   + J.CuO.vacOPP.*AoR.CuO;
%   outPut.Cu2O = (dnCu2O - dnCuO/2); % growth of CuO leads to destruction of 1/2 a Cu2O
%   outPut.CuO = dnCuO; % Growth of Cu20 leads to a decrease of CuO
% end
% elseif ~aSphere % aka a plane

% surfArea = -1;
% error('check area')
% dnCu = (J.Cu2O.intCuX + J.Cu2O.intCuP).*surfArea ...
% -(J.Cu2O.vacCuX + J.Cu2O.vacCuM).*surfArea;
% dnCu2O = dnCu/2;
% outPut.Cu = -dnCu; % Cu moles
% if ~isCuO
%   outPut.Cu2O = dnCu2O; % Cu2O thickness
% else

C.2. CERMET PARTICLE MODEL
C.2.7 Defect Equilibrium

This code calculates the equilibrium defect concentrations for the cermet particle model. It is used to calculate the defect concentrations in each shell of the cermet particle model and it calls the defect equilibrium calculation used in the copper oxidation model.
function [regSite,Keq] = defectEquilibriumFunc_V(sys,PO2Atm,deltaGCuO,impur) %
% calculate for each cell based on the pressure; need to do it this way
% because each cell can have a different pressure and thus a different
% defect population
% global rho Cn MW nVols
if isempty(PO2Atm)
    % initialize
    regSite.h.CuCu2O = zeros(nVols,1); % moles of holes /mole of oxygen
    regSite.h.Cu2OCuO = zeros(nVols,1); % moles of holes /mole of oxygen
    regSite.h.CuOCu2O = zeros(nVols,1); % moles of holes /mole of oxygen
    regSite.h.Cu2OO2 = zeros(nVols,1); % moles of holes /mole of oxygen
    regSite.e.CuCu2O = zeros(nVols,1); % moles of e /mole of oxygen
    regSite.e.Cu2OCuO = zeros(nVols,1); % moles of e /mole of oxygen
    regSite.e.CuOCu2O = zeros(nVols,1); % moles of e /mole of oxygen
    regSite.e.Cu2OO2 = zeros(nVols,1); % moles of e /mole of oxygen
    regSite.vacCuM.CuCu2O = zeros(nVols,1);
    regSite.vacCuM.Cu2OCuO = zeros(nVols,1);
    regSite.vacCuM.Cu2OO2 = zeros(nVols,1);
    regSite.vacOPP.CuCu2O = zeros(nVols,1); % moles of vacO /mole of oxygen
    regSite.vacOPP.Cu2OCuO = zeros(nVols,1); % moles of vacO /mole of oxygen
    regSite.vacOPP.CuOCu2O = zeros(nVols,1); % moles of vacO /mole of oxygen
    regSite.vacOPP.Cu2OO2 = zeros(nVols,1); % moles of vacO /mole of oxygen
    regSite.intCuP.CuCu2O = zeros(nVols,1);
    regSite.intCuP.Cu2OCuO = zeros(nVols,1);
    regSite.intCuP.Cu2OO2 = zeros(nVols,1);
    regSite.vacDPP.CuCu2O = zeros(nVols,1); % moles of vacO /mole of oxygen
    regSite.vacDPP.Cu2OCuO = zeros(nVols,1); % moles of vacO /mole of oxygen
    regSite.vacDPP.CuOCu2O = zeros(nVols,1); % moles of vacO /mole of oxygen
    regSite.vacDPP.Cu2OO2 = zeros(nVols,1); % moles of vacO /mole of oxygen
    regSite.intCuP.CuCu2O = zeros(nVols,1);
    regSite.intCuP.Cu2OCuO = zeros(nVols,1);
    regSite.intCuP.Cu2OO2 = zeros(nVols,1);
APPENDIX C. MATLAB CODES

regSite.intCuX.CuCu2O = zeros(nVols,1);
regSite.intCuX.Cu2OCuO = zeros(nVols,1);
regSite.intCuX.Cu2OO2 = zeros(nVols,1);

regSite.vacCuMM.CuOCu2O = zeros(nVols,1);
regSite.vacCuMM.CuOO2 = zeros(nVols,1);

regSite.intCuPP.CuOCu2O = zeros(nVols,1);
regSite.intCuPP.CuOO2 = zeros(nVols,1);

return

% determine number of phases....

CuOStable = find(PO2Atm >= sys.presCu2OCu0_atm,1);
Cu2OStable = find(PO2Atm < sys.presCu2OCu0_atm & PO2Atm >= sys.presCuCu2O_atm);

if ~isempty(Cu2OStable) || ~isempty(CuOStable)

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * numDens.phase.defectCharge * * * * * * * * * * * * * * * *
% * * * * * * * regular site concentration * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
regSite.Cu.int = regSite.Cu.Cu *2; % sites/cm^3 , interstitial site density.... <--- check...

regSite.Cu2O.int = regSite.Cu2O.Cu *8; % moles of inter / moles of Cu2O

regSite.CuO.int = regSite.CuO.Cu *4; % sites/cm^3 , interstitial site density....

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * Other Copper Properties * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
bgAlpha.Cu2O = 4.8e-4; % eV/K
bgBeta.Cu2O = 275; % K
e0.Cu2O = 2.173; % eV, band gap at 0 K... from italy paper, 129-- Defects
and doping in Cu2O
prop.Cu2O.elecMass = 0.61*9.10938188*10^-31; % kg...
C.2. CERMET PARTICLE MODEL

245

p r o p . C u 2 O . h o l e M a s s = 0 .84 *9 .10938188 *10^ -31; % kg...

82
83

bgAlpha.CuO = 0 * 4 .8e -4; % eV / K

84

< -- - - - - - CHECK VALUE

,→ - - - 85

bgBeta.CuO

86

eG0.CuO = 1 .7 ; % eV , band gap at 0 K...assume same slope as Cu2O... not true

= 275;

% K

< - - - - - - - CHECK VALUE - - - -

87

% wikipeida says 1 .2 at room tmp

88

p r o p . C u O . e l e c M a s s = 8*9 .10938188 *10^ -31; % kg...

< - - - - - - - CHECK VALUE - - - -

89

p r o p . C u O . h o l e M a s s = 8*9 .10938188 *10^ -31; % kg...

< - - - - - - - CHECK VALUE - - - -

,→ probs...

90
91

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

92

% * * * * * * * * Rate Constants and Equilibrium Constants

93

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

94

%

95

% Rx1 : 1/2 O2 <= > Ox + 2 VCu ' + 2 h.

96

%

** [=] from Stoklosa. Solid state chemistry , 2012

deltaG.R2Cu2O = ( -28*1000 - (53* sys.temp ) ) ; % J / mol

100

%

101

% Rx3 : Ox +2 Cu +2 h <= > 2 Cu_i. + 1/2 O2

**

deltaG.R2Cu2O = -6*1000 - ( -9 .6 * sys.temp ) ; % J / mol

**

deltaG.R3Cu2O = 2*((103) *1000 - ((57) * sys.temp ) ) ; % J / mol

102

**

% Rx4 : Cu_i. + e ' <= > Cu_ix
deltaG.R4Cu2O = ( -132*1000 - (26* sys.temp ) ) ; % J / mol

104
105

**
**

% Rx2 : V '+ h. <= > Vx

99

103

**

deltaG.R1Cu2O = 2*(94*1000 - ( -41* sys.temp ) ) ; % J / mol
deltaG.R1Cu2O = 2*(94*1000 - ( -41* sys.temp ) ) ; % J / mol

97
98

* * * * * * * *

% Rx5 :

** , method A

2 h + Ox <= > 1/2 O2 + V_O..

deltaG.R5Cu2O = ((294 .4 ) *1000 - ((106) * sys.temp ) ) ; % J / mol , from Xu &

106

,→ Diekmann , 169 ....for non - ionized - - - adjusted to account for based on
,→ rx2
107

% Frenkel Equilibrium , cation --> x -> Cu_i. + V_Cu '
delta G.RFECu 2O = 201*1000 - (19* sys.temp ) ; % J / mol

108
109
110

% x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - - x

111

% x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - - x

112

% Rx1 : 1/2 O2 <= > Ox + VCu ' ' + 2 h.

114

% Rx2 : Ox + Cu +2 h <= > Cu_i.. + 1/2 O2

115

%

116
117
118

**

deltaG.R1CuO = deltaGCuO.vac ; % J / mol

113

**

deltaG.R2CuO = (103/2*1000 - (62* sys.temp ) ) ; % J / mol

**

above....

deltaG.R2CuO = deltaGCuO.int ;
% Rx3 :

2 h + Ox <= > 1/2 O2 + V_O..

deltaG.R3CuO = ((294 .4 +28*3/2) *1000 - ((106 -53*3/2) * sys.temp ) ) ; % J / mol , from
,→

Xu & Diekmann , 169 ....for non - ionized - - - adjusted to account for

,→ based on rx2
119
120

% Frenkel Equilibrium , cation --> Cu - > Cu_i.. + V_Cu ' '
deltaG.RFECuO = 0; % J / mol

121
122

% x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - -x - - x


APPENDIX C. MATLAB CODES

```matlab
% x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x
prop.Cu2O.bandGap = eG0.Cu2O - bgAlpha.Cu2O * sys.temp ^2/(sys.temp + bgBeta.Cu2O) % eV
prop.Cu2O.valDOS = 2*((2* pi* prop.Cu2O.holeMass * Cn.kB * sys.temp )/ Cn.hPlank_JS ^2)^(3/2) /100^3; % atoms/cm ^3 ...
prop.Cu2O.conDOS = 2*((2* pi* prop.Cu2O.elecMass * Cn.kB * sys.temp )/ Cn.hPlank_JS ^2)^(3/2) /100^3; % atoms/cm ^3 ...

prop.CuO.bandGap = eG0.CuO - bgAlpha.CuO * sys.temp ^2/(sys.temp + bgBeta.CuO); % eV
prop.CuO.valDOS = 2*((2* pi* prop.CuO.holeMass * Cn.kB * sys.temp )/ Cn.hPlank_JS ^2)^(3/2) /100^3; % atoms/cm ^3 ...
prop.CuO.conDOS = 2*((2* pi* prop.CuO.elecMass * Cn.kB * sys.temp )/ Cn.hPlank_JS ^2)^(3/2) /100^3; % atoms/cm ^3 ...

Keq.Cu2O.Rx1 = exp (- deltaG.R1Cu2O /( Cn.gasconst * sys.temp )); % Cu vac. -
Keq.Cu2O.Rx2 = exp (- deltaG.R2Cu2O /( Cn.gasconst * sys.temp )); % Cu vac.
Keq.Cu2O.Rx3 = exp (- deltaG.R3Cu2O /( Cn.gasconst * sys.temp )); % Cu int.
Keq.Cu2O.Rx4 = exp (- deltaG.R4Cu2O /( Cn.gasconst * sys.temp )); % Cu int. +
Keq.Cu2O.Rx5 = 0*exp (- deltaG.R5Cu2O /( Cn.gasconst * sys.temp )); % Ox vac. ++
Keq.Cu2O.frenkel = 0*exp (- deltaG.RFECu2O /( Cn.gasconst * sys.temp ));

Keq.CuO.Rx1 = exp (- deltaG.R1CuO /( Cn.gasconst * sys.temp )); % Cu vac --
Keq.CuO.Rx2 = exp (- deltaG.R2CuO /( Cn.gasconst * sys.temp )); % Cu int ++
Keq.CuO.Rx3 = 0*exp (- deltaG.R3CuO /( Cn.gasconst * sys.temp )); % Ox vac ++
Keq.CuO.frenkel = 0*exp (- deltaG.RFECuO /( Cn.gasconst * sys.temp ));

% When an oxide forms, Cu/Cu2O interface
% x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x
coef.a = 2 * Keq.Cu2O.Rx5 / sys.presCuCu2O_atm ^(1/2); % h^-3
coef.b = 1 + Keq.Cu2O.Rx3 ^(1/2) / sys.presCuCu2O_atm ^(1/4); % h^-2
coef.c = Keq.Cu2O.eh + Keq.Cu2O.Rx1 ^(1/2) * sys.presCuCu2O_atm ^(1/4); % h^-0
```

% When an oxide forms, Cu/Cu2O interface
% x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x--x
% calculate site fractions for the given oxygen concentrations...
% HOLE concentration ->: solve charge neutrality equation
coef.a = 2 * Keq.Cu2O.Rx5 / sys.presCuCu2O_atm ^(1/2); % h^-3
coef.b = 1 + Keq.Cu2O.Rx3 ^(1/2) / sys.presCuCu2O_atm ^(1/4); % h^-2
coef.c = Keq.Cu2O.eh + Keq.Cu2O.Rx1 ^(1/2) * sys.presCuCu2O_atm ^(1/4); % h^-0
if coef.a == 0
    hdot = (coef.c/coef.b)^0.5;
else
    sol.roots = roots([coef.a,coef.b,0,-coef.c]);
    hdot = sol.roots(((imag(sol.roots)==0 & sol.roots>0));
    if isempty(hdot)
        sol.roots = roots([coef.b,0,-coef.c]);
        hdot = sol.roots(((imag(sol.roots)==0 & sol.roots>0));
    end
end
regSite.h.CuCu2O = hdot;
regSite.e.CuCu2O = Keq.Cu2O.eh./hdot; % moles/mole of oxygen
regSite.vacCuM.CuCu2O = Keq.Cu2O.Rx1.^(1/2) * sys.presCuCu2O_atm^(1/4);
regSite.vacCuX.CuCu2O = Keq.Cu2O.Rx2.*regSite.vacCuM.CuCu2O.*hdot; % moles/mole of oxygen
regSite.vacOPP.CuCu2O = Keq.Cu2O.Rx5.*hdot.^2./sys.presCuCu2O_atm.
regSite.intCuP.CuCu2O = Keq.Cu2O.Rx3.^(1/2).*regSite.intCuP.CuCu2O.*regSite.e.CuCu2O; % moles/mole of oxygen
regSite.intCuX.CuCu2O = Keq.Cu2O.Rx4.*regSite.intCuP.CuCu2O.*regSite.e.CuCu2O; % moles/mole of oxygen

if isempty(Cu2OStable) && isempty(CuOStable)
% from charge neutrality...
coeff.a = 2 * Keq.Cu2O.Rx5./P02Atm.^(1/2); % h^3
coeff.b = 1 + Keq.Cu2O.Rx3.^(1/2)./P02Atm.^(1/4); % h^2
coeff.c = impur.Cu2O.pos - impur.Cu2O.neg; % h^1
coeff.d = Keq.Cu2O.eh + Keq.Cu2O.Rx1.^(1/2).*P02Atm.^(1/4); % h^-0
if Keq.Cu2O.Rx5 == 0 && coeff.c ==0
    hdot = (coeff.d./coeff.b).^0.5;
else
    for ii = 1:length(P02Atm)
        sol.roots = roots([coeff.a(ii),coeff.b(ii),coeff.c,-coeff.d(ii)]);
        hdot(ii,1) = sol.roots(((imag(sol.roots)==0 & sol.roots>0));
        if isempty(hdot(ii,1))
            sol.roots = roots([coeff.b(ii),coeff.c,-coeff.d(ii)]);
            hdot(ii,1) = sol.roots(((imag(sol.roots)==0 & sol.roots>0));
        end
    end
end
APPENDIX C. MATLAB CODES

regSite.h.Cu2OO2 = hdot; % moles of holes /mole of oxygen
regSite.e.Cu2OO2 = Keq.Cu2O.eh./regSite.h.Cu2OO2; % moles/mole of oxygen

regSite.vacCuM.Cu2OO2 = Keq.Cu2O.Rx1.^(1/2) .* PO2Atm.^(1/4) ./ regSite.h.Cu2OO2; % moles/mole of oxygen
regSite.vacCuX.Cu2OO2 = Keq.Cu2O.Rx2.*regSite.vacCuM.Cu2OO2. * regSite.h.Cu2OO2; % moles/mole of oxygen
regSite.vacOOPP.Cu2OO2 = Keq.Cu2O.Rx5. * regSite.h.Cu2OO2. ^2./ PO2Atm.^(1/2); % moles /mole of oxygen
regSite.intCuP.Cu2OO2 = Keq.Cu2O.Rx3.^(1/2) .* regSite.h.Cu2OO2. / PO2Atm.^(1/4); % moles /mole of oxygen
regSite.intCuX.Cu2OO2 = Keq.Cu2O.Rx4. * regSite.intCuP.Cu2OO2. * regSite.e.Cu2OO2; % moles /mole of oxygen

elseif ~isempty ( CuOStable )

% from charge neutrality...
coef.a = 2 * Keq.Cu2O.Rx5 / sys.presCu2OCuO_atm ^(1/2); % h^-3
coef.b = 1 + Keq.Cu2O.Rx3 ^(1/2) / sys.presCu2OCuO_atm ^(1/4); % h^-2
coef.c = impur.Cu2O.pos - impur.Cu2O.neg; % h^-1
coef.d = Keq.Cu2O.eh + Keq.Cu2O.Rx1.* sys.presCu2OCuO_atm.*(1/4); % h^0

if coef.a == 0 && coef.c == 0
hdot = (coef.d/coef.b)^0.5;
if ~isreal (hdot)
a=1;
error (’holes negative’)
end
else
sol.roots = roots([coef.a,coef.b,coef.c,-coef.d]);
if isempty (sol.roots)
sol.roots = roots([coef.b,coef.c,-coef.d]);
end
end

regSite.h.Cu2OCuO = hdot; % moles of holes /mole of oxygen
regSite.e.Cu2OCuO = Keq.Cu2O.eh./hdot; % moles/mole of oxygen
regSite.vacCuM.Cu2OCuO = Keq.Cu2O.Rx1.* sys.presCu2OCuO_atm.*(1/4)/ hdot; % moles/mole of oxygen
regSite.vacCuX.Cu2OCuO = Keq.Cu2O.Rx2.*regSite.vacCuM.Cu2OCuO*hdot; % moles/mole of oxygen
C.2. CERMET PARTICLE MODEL

```plaintext
regSite.vacOPP.Cu2OCu0 = Keq.Cu2O.Rx5 * hdot^2 / sys.presCu2OCu0_atm
(1/2); % moles/mole of oxygen
regSite.intCuP.Cu2OCu0 = Keq.Cu2O.Rx3 * hdot / sys.presCu2OCu0_atm
(1/4); % moles/mole of oxygen
regSite.intCuX.Cu2OCu0 = Keq.Cu2O.Rx4 * regSite.intCuP.Cu2OCu0;
→ regSite.e.Cu2OCu0; % moles/mole of oxygen

% from charge neutrality...
coef.a = 2 * Keq.CuO.Rx3 * sys.presCu2OCu0_atm^((1/2)); % h^4
coef.b = 1; % h^3
coef.c = impur.CuO.pos - impur.CuO.neg; % h^-2
coef.d = -Keq.CuO.eh; % h^-1
coef.e = -2*Keq.CuO.Rx1 * sys.presCu2OCu0_atm^(1/2); % h^-0

% if coef.a == 0
% d0 = -3*coef.b*coef.c;
% d1 = 2?*coef.b^2*coef.d;
% C = ((d1+sqrt(d1^-2-4*d0^-3))/2)^(-1/3);
% uk = (-1+i*sqrt(3))/2;
% % uk = 1;
% % % %
% hdot = real((-1/3*coef.b) + (0+uk*C+d0/(uk*C)));
% else % r^-4 r^-3 r^-2 r 0
% sol.roots = roots([coef.a,coef.b,coef.c,coef.d,coef.e]);
% hdot = sol.roots((imag(sol.roots)==0 & sol.roots>0));
% end

regSite.h.CuOCu2O = hdot; % moles of holes / mole of oxygen
regSite.e.CuOCu2O = Keq.CuO.eh/hdot; % moles/mole of oxygen
regSite.vacCuMM.CuOCu2O = Keq.CuO.Rx1 * sys.presCu2OCu0_atm^(-1/2)/hdot^2;
→ % moles/mole of oxygen
regSite.vacOPP.CuOCu2O = Keq.CuO.Rx3 * hdot^-2 / sys.presCu2OCu0_atm
(1/2); % moles/mole of oxygen
regSite.intCuPP.CuOCu2O = Keq.CuO.Rx2 * hdot^-2 / sys.presCu2OCu0_atm
(1/2); % moles/mole of oxygen

% from charge neutrality...
coef.a = 2 * Keq.CuO.Rx3 / sys.presCu2OCu0_atm^((1/2)); % h^-4
coef.b = 1; % h^-3
coef.c = impur.CuO.pos - impur.CuO.neg; % h^-2
coef.d = -Keq.CuO.eh; % h^-1a
coef.e = -2*Keq.CuO.Rx1 * sys.presCu2OCu0_atm^((1/2)); % h^-0

% if Keq.CuO.Rx3 == 0 & Keq.CuO.Rx2 == 0
% d0 = -3*coef.b*coef.c;
```
C.3 System Model

Below is code that was used to model the CLC system. The code is setup with the following structure: the driver model calls the CLC Plant model. The CLC Plant Model has sub functions for the air reactor, fuel reactor and solid gas separators. Additional codes were developed to calculate the heat of formation of a solid fuel given its ultimate analysis and heating value, calculate the enthalpy of the streams leaving the air and fuel reactors, and calculate the sensible enthalpy of the ash and coal char.
C.3. SYSTEM MODEL

C.3.1 CLC Plant Driver

This function calls the function of the CLC system.

```matlab
clear all

% CLC System Model
% CREATOR : E.A. Goldstein
% DATE Created : 6-26-11

% System Description:
% - Air Reactor
%   INPUTS: Solid/Gas State, nDotVect, Air Flow Rate, Qloss
%   OUTPUTS: Solid/Gas State, nDotVect
% - Fuel Reactor
%   INPUTS: Solid/Gas State, nDotVect, Fuel Flow Rate, Qloss
%   OUTPUTS: Solid/Gas State, nDotVect
% - Solid Gas Separator
%   INPUTS: Solid/Gas State, nDotVect, Fraction of gas transferred with solid
%   OUTPUTS: Solid/Gas State, nDotVect
% - Gas Turbine
%   INPUTS: Gas State, nDotVect, Polytropic efficiency
%   OUTPUTS: Gas State, nDotVect, work out
% - Heat Exchanger + Steam Turbine
%   INPUTS: Gas State, nDotVect, water inlet state, water flow rate,
%   Heat loss, pinch temp, Polytropic Efficiency
%   OUTPUTS: Gas State, nDotVect, work out

% Exergy Calculations
% Solid Dead State: Inlet state of solid at 298 K
% Gas Dead State: Environment @ 298 K
% Air composition, Cu

% what parameter space will the system be investigated
% -- fuel to OC ratio
% -- air to OC ratio
```
% function returns --
% how much O2 taken up by Cu
% how much C -> CO2
% how much O -> CO2

var1 = [2 3 4 5 6 7]; % moles of Cu
var2 = [0.4 0.5 0.6 0.7 0.8 0.9]; % m3/s of air

tic
for i = 1:length(var1)
    for j = 1:length(var2)
        inputStrut.molesCuInit = var1(i); % moles of Cu initially
        inputStrut.molesCH4 = 1; % moles of CH4
        inputStrut.mDotCoal = 0; % kg/s of coal
        inputStrut.QAir = var2(j); % m3/s of air at STP

        inputStrut.TempAR = 800+273.15;
        inputStrut.TempFR = 800+273.15;
        output = clcPlantFunc(inputStrut);

        ARQ(i,j) = output.QReactors.AR; % heat from air reactor
        FRQ(i,j) = output.QReactors.FR; % heat from fuel reactor
        O2ExhaustFR(i,j) = output.COExhaust; % PPM of CO in the exhaust to FR
        CO2ExhaustAR(i,j) = output.MFCO2OutAir; % PPM of CO in the exhaust to AR
        disp(j)
    end
end
toc

% global nEnd nGas nSolid
C.3. SYSTEM MODEL

```matlab
global MW sys
figure
hold on
for j = 1:length(var2)
    plot(var1,ARQ(:,j)/1000,'b',DisplayName,[Heat_A_R, Q_a_i_r = ' num2str(var2 (j)) ' m^3/s'],LineWidth',2)
    plot(var1,FRQ(:,j)/1000,'k',DisplayName,[Heat_F_R, Q_a_i_r = ' num2str(var2 (j)) ' m^3/s'],LineWidth',2)
end
xlabel('Cu Loading, moles')
ylabel('Heat Release, kW')

figure
hold on
for i = 1:length(var1)
    plot(var2,OCOxyToAirOxy(i,:),'DisplayName',[OC Capacity to O supplied, Cu Loading = ' num2str(var1(i)) ' moles'],LineWidth',2)
end
xlabel('Air Flow, m^3/s')
ylabel('O in air to max O in OC')
grid on

figure
hold on
for i = 1:length(var1)
    plot(var2,MFO2OutAR(i,:),'DisplayName',[OC Capacity to O supplied, Cu Loading = ' num2str(var1(i)) ' moles'],LineWidth',2)
end
xlabel('Air Flow, m^3/s')
ylabel('O2 MF, Exhaust Air Reactor, [-]')
grid on

figure
hold on
for j = 1:length(var2)
    plot(var1*MW.Cu/(inputStrut.molesCH4*sys.P0/(8.314*sys.T0)),COExhaustFR(:,j),'b',DisplayName,[CO PPM, Q_a_i_r = ' num2str(var2(j)) ' m^3/s'],LineWidth',2)
end
xlabel('Cu to CH4 Ratio, g Cu per L/s of CH4')
ylabel('CO FR Exhaust, PPM')
grid on

% Questions:
% -how does the temp of each reactor change as the air flow rate and fuel flow rate change?
```
% -if the fuel and air reactor are locked in at specified temperatures, how
% does this impact system energetics...

% what is the temperature of the air/fuel reactor
% what are the composition of the streams?

% exergy.MeOutFr = flowExergyStream(MeOFuelIn,deadState,gasObject,solidObject);
% exergy.gasOutFR = flowExergyStream(MeOFuelIn,deadState,gasObject,solidObject);

% *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
% Fuel Reactor Exergy Calc and compute power extracted from exhaust
% *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
% set ( gasObject , 'T' , tempStructure.fuelReactor , 'P' , Pres , 'X' , nVectFuelExhaust
  (1:nEnd.gas));
% exergy.combProd = flowExergyMass(gasObject , 'Temp' , T0 , 'Pres' , P0 , 'X' ,
  xDeadState.gas); % per unit mass
% 
% set (gasObject , 'T' , tempStructure.fuelReactor , 'P' , Pres , 'X' , nVect(1:nEnd.gas))
  -> ;
% exergy.metal = flowExergyMass(gasObject , 'Temp' , T0 , 'Pres' , P0 , 'X' ,
  xDeadState , 'solidID' , solidObject , 'moleVect' , nVect); % per unit mass

% *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
% determine exergy lost from separation process
% *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
% set (gasObject , 'T' , tempStructure.fuelReactor , 'P' , Pres , 'X' , nVect(1:nEnd.gas
  )+nVectFuelExhaust(1:nEnd.gas));
% exergy.metalCombProd = flowExergyMass(gasObject , 'Temp' , T0 , 'Pres' , P0 , 'X' ,
  xDeadState , 'solidID' , solidObject , 'moleVect' , nVect+nVectFuelExhaust); % per
  unit mass
% 
% if sum(nVect(1:nEnd.gas)) ==0
C.3. SYSTEM MODEL

This script initializes the CLC plant model.

C.3.2 Initialization function for the CLC plant model

This script initializes the CLC plant model.
% for plotting purposes
if length(Temp) ==1
  tempVect = Temp;
else
  tempVect = linspace(Temp(1),Temp(2),10);
end

% Calculate the Elem. Matrix for the total system...
% define constraint matrix...
% ***********************************
% Constraints for atoms...
% ***********************************
Elem.gas = zeros(nEnd.gas,nAtom.gas);

for k = 1: nEnd.gas
  for j = 1: nAtom.gas
    Elem.gas(k,j) = nAtoms(gasObject,k,j);
  end
end

Elem.solid = zeros(nEnd.solid,nAtom.solid);
for k = 1: nEnd.solid
  for j = 1: nAtom.solid
    Elem.solid(k,j) = nAtoms(solidObject,k,j);
  end
end

% align the atoms!!
% row is gas...
% column solid...
% also ask, how many unique atoms are there?
alingmentVect = zeros(nAtom.solid,1);
nAtom.total = nAtom.gas;           % this will
nAtom.Names = nAtom.gasNames;
for i = 1: nAtom.solid
  for j = 1: nAtom.gas
    alingmentVect(i) = alingmentVect(i) + j * strcmp(nAtom.gasNames(j),
          nAtom.solidNames(i));
  end
  if(alingmentVect(i) ==0)
    nAtom.total = nAtom.total+1;
    alingmentVect(i) = nAtom.total;
    nAtom.Names(nAtom.total) = elementName(solidObject,i);
  end
end
C.3. SYSTEM MODEL

C.3.3 CLC Plant Function

This function is the model of the combine CLC system.

```matlab
function outPut = clcPlantFunc(inputStrut)

% initFileForFunc
% start with the air reactor...
% Run Calcs in Air Reactor & separator

[outPutState_FR,Qnet.AR] = airReactor(solidObject,gasObject,airIn,MeAirIn,
    ↪QLoss.airReactor);
[gasOutPut,solidOutPut] = solidGasSeparator(outPutState_FR,separationConst.air,
    ↪gasObject,solidObject);
%
% eBal & aBal
Hin = airIn.enthalpy*sum(airIn.nVect) + MeAirIn.enthalpy*sum(MeAirIn.nVect);
Hout = gasOutPut.enthalpy*sum(gasOutPut.nVect) + solidOutPut.enthalpy*sum(
    ↪solidOutPut.nVect) + solidOutPut.mDotAsh*solidOutPut.ashEnthalpy;
eBal.airReact = Hin - Hout + Qnet.AR;
aBal.airReact = Elem.total'*(airIn.nVect + MeAirIn.nVect - (gasOutPut.nVect
    ↪+solidOutPut.nVect));
```
APPENDIX C. MATLAB CODES

```
MeOFuelIn = solidOutPut;  \% solids leaving the air reactor enter into the fuel reactor...

notConverged = true;
iterCounter = 0;
deltaFR.nVect = -1;
deltaAR.nVect = -1;

while notConverged
    % *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=
    % Run calcs in Fuel Reactor & separator
    % *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=
    if iterCounter > 0
        gasFuel_p.nVect = outPutState_FR.nVect;
        gasFuel_p.Temp = outPutState_FR.Temp;
    end

    [outPutState_FR, Qnet.FR] = fuelReactor(solidObject, gasObject, fuelInAsGas,
                                           MeOFuelIn, QLoss.fuelReactor);
    [gasOutPut, solidOutPut] = solidGasSeparator(outPutState_FR,
                                              separationConst.fuel, gasObject, solidObject);
    Hin = (fuelInAsGas.enthalpy*sum(fuelInAsGas.nVect)+fuelInAsGas.mDotAsh*
           fuelInAsGas.ashEnthalpy)+MeOFuelIn.enthalpy*sum(MeOFuelIn.nVect)
         +MeOFuelIn.mDotAsh*MeOFuelIn.ashEnthalpy;
    Hout = gasOutPut.enthalpy*sum(gasOutPut.nVect) + solidOutPut.enthalpy*sum
         (solidOutPut.nVect) + solidOutPut.mDotAsh*solidOutPut.ashEnthalpy;
    eBal.fuelReactor = Hin - Hout + Qnet.FR;
    aBal.fuelReactor = Elem.total*(fuelInAsGas.nVect + MeOFuelIn.nVect - (g
         outPut.nVect+solidOutPut.nVect));

    MeAirIn = solidOutPut;
    % *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=
    % Exhaust gases and send solids to air reactor..
    % *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=
    if iterCounter > 0
        gasAir_p.nVect = outPutState_AR.nVect;
        gasAir_p.Temp = outPutState_AR.Temp;
    end

    [outPutState_AR, Qnet.AR] = airReactor(solidObject, gasObject, airIn, MeAirIn,
                                           QLoss.airReactor);
    [gasOutPut, solidOutPut] = solidGasSeparator(outPutState_AR,
                                              separationConst.air, gasObject, solidObject);
    Hin = airIn.enthalpy*sum(airIn.nVect) + MeAirIn.enthalpy*sum
        (MeAirIn.nVect) + MeAirIn.mDotAsh*MeAirIn.ashEnthalpy;
    Hout = gasOutPut.enthalpy*sum(gasOutPut.nVect) + solidOutPut.enthalpy*sum
        (solidOutPut.nVect)+solidOutPut.mDotAsh*solidOutPut.ashEnthalpy;
```

C.3. SYSTEM MODEL

\[ e_{\text{Bal.airReact}} = H_{\text{in}} - H_{\text{out}} + Q_{\text{net.AR}}; \]
\[ a_{\text{Bal.airReactor}} = E_{\text{lem.total}} \cdot (\text{airIn.nVect} + \text{MeAirIn.nVect} - (\text{gasOutPut.nVect} + \text{solidOutPut.nVect})); \]

% ash separation at outlet of air reactor...
if separationConst.ashAir
    solidOutPut.mdotAsh = 0;
end

% Update inlet to fuel reactor...
MeOFuelIn = solidOutPut; % solids leaving the air reactor enter into the fuel reactor...
if norm(aBal.fuelReactor) > 1e-6
    error('atom balance error, fuel reactor')
end
if norm(eBal.fuelReactor/Qnet.FR) > 1
    error('energy balance error, fuel reactor')
end

% *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
% convergence critera....
% *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
% check if gas changes after each iteration...
if iterCounter > 0
    deltaFR.T(iterCounter) = gasFuel_p.Temp - outPutState_FR.Temp;
    deltaAR.T(iterCounter) = gasAir_p.Temp - outPutState_AR.Temp;
    deltaFR.nVect(iterCounter) = norm(gasFuel_p.nVect - outPutState_FR.nVect);
    deltaAR.nVect(iterCounter) = norm(gasAir_p.nVect - outPutState_AR.nVect);
% *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
    if abs(deltaFR.nVect(iterCounter)) < 1e-3 && abs(deltaAR.nVect(iterCounter)) < 1e-3
        notConverged = false;
    end
end

% update counter
iterCounter = iterCounter + 1;
end

% *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
% run one more time to define finals steady state states
% *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
APPENDIX C. MATLAB CODES

MeOFuelIn = solidOutPut; % solids leaving the air reactor enter into the fuel reactor...

[outPutState_FR,Qnet.FR] = fuelReactor(solidObject,gasObject,fuelInAsGas,
MeOFuelIn,QLoss.fuelReactor);

gasOutPutFuel,solidOutPutFuel = solidGasSeparator(outPutState_FR,
separationConst.fuel,gasObject,solidObject);

Hin = (fuelInAsGas.enthalpy*sum(fuelInAsGas.nVect)+fuelInAsGas.mDotAsh*
fuelInAsGas.ashEnthalpy) + MeOFuelIn.enthalpy*sum(MeOFuelIn.nVect)+
MeOFuelIn.mDotAsh*MeOFuelIn.ashEnthalpy;

*sum(solidOutPutFuel.nVect) + solidOutPutFuel.mDotAsh*
solidOutPutFuel.ashEnthalpy;

eBal.fuelReactor = Hin - Hout + Qnet.FR;

aBal.fuelReactor = Elem.total'*((fuelInAsGas.nVect + MeOFuelIn.nVect -
gasOutPutFuel.nVect+solidOutPutFuel.nVect));

MeAirIn = solidOutPutFuel;

% *************************************************************
% Exhaust gases and send solids to air reactor...
% *************************************************************

[outPutState_AR,Qnet.AR] = airReactor(solidObject,gasObject,airIn,MeAirIn,
QLoss.airReactor);

[gasOutPutAir,solidOutPutAir] = solidGasSeparator(outPutState_AR,
separationConst.air,gasObject,solidObject);

Hin = airIn.enthalpy*sum(airIn.nVect) + MeAirIn.enthalpy*sum(MeAirIn.nVect)+
MeAirIn.mDotAsh*MeAirIn.ashEnthalpy;

Hout = gasOutPutAir.enthalpy*sum(gasOutPutAir.nVect) + solidOutPutAir.enthalpy*
sum(solidOutPutAir.nVect) + solidOutPutAir.mDotAsh*
solidOutPutAir.ashEnthalpy;

eBal.airReact = Hin - Hout + Qnet.AR;

aBal.airReactor = Elem.total'*((airIn.nVect + MeAirIn.nVect -
gasOutPutAir.nVect+
solidOutPutAir.nVect));

% *************************************************************
% *************************************************************

carbonInFuel = Elem.total(:,nAtom.Names.C)'*fuelInAsGas.nVect; % moles carbon in
the fuel

hydrogenInFuel = Elem.total(:,nAtom.Names.H)'*fuelInAsGas.nVect; % moles hydrogen
in the fuel

outPut.PcntCarbInCO2Fuel = (gasOutPutFuel.nVect(nGas.CO2))/carbonInFuel*100; %
percent of C in CO2 in the exhaust

outPut.COExhaust = gasOutPutFuel.nVect(nGas.CO)/sum(gasOutPutFuel.nVect)*10^-6; %
CO concentration in ppm
C.3. SYSTEM MODEL

OxyInAirGas = Elem.total(:,aAtom.Names.O)*airIn.nVect; % oxygen moles entering the air reactor

OxyInAirSolids = Elem.total(:,aAtom.Names.O)*MeAirIn.nVect;
% note not all O accounted for b/c O leaves as other species as well (e.g., CuSO4 and Cu2O)

OxyOutAirGas = gasOutPutAir.nVect(nGas.O2)*2; % oxygen moles leaving the air reactor as gas

OxyOutAirSolids = solidOutPutAir.nVect(nSolid.Cu2O)*solidOutPutAir.nVect(nSolid.CuO);
% oxygen moles leaving the air reactor as gas

deltaOxyFuel = solidOutPutAir.nVect(nSolid.Cu)+solidOutPutAir.nVect(nSolid.Cu2O)+solidOutPutAir.nVect(nSolid.CuO);

outPut.PcntO2Cu2OCuO = OxyOutAirSolids / (OxyInAirGas + OxyInAirSolids); % fraction of oxygen in that leaves air reactor as Cu2O or CuO

outPut.OxyUptake = (OxyInAirGas - OxyOutAirGas)/OxyInAirGas; % fraction of O2 in that is taken up by solids...

outPut.MFOxyOutAir = gasOutPutAir.nVect(nGas.O2)/sum(gasOutPutAir.nVect); % mole fraction

outPut.MFCO2OutAir = gasOutPutAir.nVect(nGas.CO2)/sum(gasOutPutAir.nVect); % mole fraction

outPut.stoic = (carbonInFuel*2+hydrogenInFuel/2)/deltaOxyFuel;


EOut = Qnet.AR+Qnet.FR;
EIn = sum(gasOutPutFuel.nVect)*gasOutPutFuel.enthalpy+sum(gasOutPutAir.nVect)*gasOutPutAir.enthalpy-
(sum(fuelInAsGas.nVect)*fuelInAsGas.enthalpy+sum(aIn.nVect)*aIn.enthalpy);

outPut.FracEnergyAR = Qnet.AR/EIn;
outPut.FracEnergyFR = Qnet.FR/EIn;

outPut.eBal = (EOut-EIn)/EIn;

outPut.OCtoFuel = sum(MeAirIn.nVect)/(carbonInFuel*2+hydrogenInFuel/2); % max moles O in OC to moles of O needed by fuel
outPut.OCOxytoAirOxy = (airIn.nVect(nGas.O2)*2)/sum(MeAirIn0.nVect);
% moles of O in air to max moles O in OC --> should always be > 1

cellPut.AROutState = outPutState_AR;
cellPut.FROutState = outPutState_FR;
cellPut.QReactors = Qnet;

C.3.4 Air Reactor Function

This function is the model of the air reactor.

```matlab
function [outPutState,QNet] = airReactor(solidObject,gasObject,airIn,MeOx,QLoss)
%
% Air reactor Model
% CREATOR: E.A. Goldstein
% Date: 7-2-13
% - Air Reactor
% INPUTS: Solid/Gas State, nDotVect, Air Flow Rate, Qloss
% OUTPUTS: Solid/Gas State, nDotVect
%
% UPDATE: 7/2/13
% Adiabatic Air reactor metal and air in
% QLoss functionality
% assume ash enthalpy not large compared to gases and other solids; so not
% included in equilibration...
%
global nEnd sys
%
% check if the temp of the solid is the same as the gas (it should be...)
% if not, break calcs
if (airIn.Pres - MeOx.Pres)/(airIn.Pres + MeOx.Pres)*2 > 1e-4
    error('gas pressure not equal in air reactor...')
end
%
if QLoss > 1
    error('Q-Loss should be a decimal...i.e. less then 1')
end
%
% MIX REACTING STREAMS
preReactState = gasSolidMixingChamberT(airIn,MeOx,gasObject,solidObject);
%
% energy bal. of gas-solid-mixing chamber
HStart = airIn.enthalpy*sum(airIn.nVect)+ MeOx.enthalpy*sum(MeOx.nVect) +
% MeOx.mDotAsh*MeOx.ashEnthalpy;
```
C.3. SYSTEM MODEL

HIn = (preReactState.enthalpy + sum(preReactState.nVect) +
preReactState.mDotAsh*preReactState.ashEnthalpy);

if sys.adbA % Is the air reactor adiabatic or @ a fixed T?
% Adiabatic
  equilibScaleFactor = 1;
  if sum(preReactState.nVect)<0.5
    equilibScaleFactor = 100;
    preReactState.nVect = preReactState.nVect*equilibScaleFactor;
  end

% HP Equilibrium...
  set (gasObject,'T',preReactState.Temp,'P',preReactState.Pres,'X',
   preReactState.nVect(1:nEnd.gas));
  if sum(preReactState.nVect(nEnd.gas+1:nEnd.total)) == 0
    % no solids...
    set (solidObject,'T',preReactState.Temp,'P',preReactState.Pres,'X',1);
  else
    set (solidObject,'T',preReactState.Temp,'P',preReactState.Pres,'X',
     preReactState.nVect(nEnd.gas+1:nEnd.total));
  end
mixSolidGas = Mixture({gasObject, sum(preReactState.nVect(1:nEnd.gas))};
   solidObject, sum(preReactState.nVect(nEnd.gas+1:nEnd.total))});

% -----------------------------------------
% react mixture
% -----------------------------------------
equilbrate(mixSolidGas,'HP');

% -----------------------------------------
% extract out temp, pres, moles and species distributions
% -----------------------------------------
nMolesPhase = phaseMoles(mixSolidGas)/equilibScaleFactor; % number of
    moles present in each phase
nDotGas = nMolesPhase(1) * moleFractions(gasObject);
nDotSolid = nMolesPhase(2) * moleFractions(solidObject);
nVect = [nDotGas;nDotSolid]; % kmols/s

% determine enthalpy change due to chemical reactions and ect...
outPutState.Temp = temperature(gasObject);
outPutState.Pres = pressure(gasObject);
outPutState.nVect = nVect;
outPutState.mDotAsh = preReactState.mDotAsh;
outPutState.ashEnthalpy = (ashSensEnthalpy(outPutState.Temp) -
   ashSensEnthalpy(sys.T0));
```matlab
outPutState.enthalpy = enthalpyStream(outPutState, gasObject, solidObject);
HOut = outPutState.enthalpy * sum(outPutState.nVect) +
       outPutState.mDotAsh*outPutState.ashEnthalpy;
QNet = HOut - HIn;

if QLoss ~= 0
    notConverged = 1;
    QLoss_c = QLoss;
    QLoss_cp = 0;
    QLossCalc_cp = QNet/HIn;

    set(gasObject,'T',outPutState.Temp,'P',outPutState.Pres,'X',
        outPutState.nVect(1:nEnd.gas));
    hGas = enthalpy_mass(gasObject); % J/kg

    while notConverged
        outPutState_temp = outPutState;

        equilibScaleFactor = 1;
        if sum(outPutState_temp.nVect)<0.5
            equilibScaleFactor = 100;
            outPutState_temp.nVect = outPutState_temp.nVect*
                equilibScaleFactor;
        end

        % reduce initial enthalpy by reducing the temp of the mixture...
        set(gasObject,'H',hGas*(1-QLoss_c),'P',outPutState_temp.Pres,'X',
            outPutState_temp.nVect(1:nEnd.gas));

        if sum(outPutState_temp.nVect(nEnd.gas+1:nEnd.total)) == 0
            % no solids... not sure what should be here...
            set(solidObject,'T',temperature(gasObject),'P',
                outPutState_temp.Pres,'X','Cu:1');
        else
            % set to temp of gas object b/c a reduction in gas enthalpy by
            % QLoss is not @ the same temp as the solid; the Mixture()
            % method sets the objects to the same temperature.... over
            % prediciting the answer...
            set(solidObject,'T',temperature(gasObject),'P',
                outPutState_temp.Pres,'X',outPutState_temp.nVect( 
                    nEnd.gas+1:nEnd.total));
        end

        mixSolidGas = Mixture({gasObject,sum(outPutState_temp.nVect(1:
            nEnd.gas));solidObject,sum(outPutState_temp.nVect(nEnd.gas 
            +1:nEnd.total))});
        equilibrate(mixSolidGas,'HP');
```
% re-calc final mole spec... and temp...

\[ n_{\text{MolesPhase}} = \frac{\text{phaseMoles(\text{mixSolidGas})}}{\text{equilibScaleFactor}}; \]
\[ \text{number of moles present in each phase} \]

\[ n_{\text{DotGas}} = n_{\text{MolesPhase}}(1) \times \text{moleFractions(gasObject)}; \]
\[ n_{\text{DotSolid}} = n_{\text{MolesPhase}}(2) \times \text{moleFractions(solidObject)}; \]
\[ n_{\text{Vect}} = [n_{\text{DotGas}}; n_{\text{DotSolid}}]; \text{ kmols/s} \]

% determine enthalpy change due to chemical reactions and ect...
\[ \text{outPutState_temp.Temp} = \text{temperature(gasObject)}; \]
\[ \text{outPutState_temp.Pres} = \text{pressure(gasObject)}; \]
\[ \text{outPutState_temp.nVect} = n_{\text{Vect}}; \]

\[ \text{outPutState_temp.enthalpy} = \text{enthalpyStream(outPutState_temp,} \]
\[ \text{gasObject, solidObject}); \]
\[ \text{HOut} = \text{outPutState_temp.enthalpy} \times \text{sum(outPutState_temp.nVect)}; \]
\[ \text{QNet} = \text{HOut} - \text{HIn}; \]
\[ \text{QLossCalc_c} = (\text{HOut} - \text{HIn})/\text{HIn}; \]

\[ \text{if abs(QLoss - QLossCalc_c)} < 1e^{-3} \]
% converged
\[ \text{notConverged} = 0; \]
\[ \text{end} \]

\[ \text{QLoss_New} = QLoss \times (QLoss_c - QLoss_cp)/(QLossCalc_c - QLossCalc_cp) \]
% update...
\[ \text{QLoss_cp} = QLoss_c; \]
\[ \text{QLossCalc_cp} = QLossCalc_c; \]
\[ \text{QLoss_c} = QLoss_New; \]
\[ \text{end} \]

\[ \text{outPutState.Temp} = \text{temperature(gasObject)}; \]
\[ \text{outPutState.Pres} = \text{pressure(gasObject)}; \]
\[ \text{outPutState.nVect} = n_{\text{Vect}}; \]

\[ \text{outPutState.enthalpy} = \text{enthalpyStream(outPutState, gasObject,} \]
\[ \text{solidObject}); \]
\[ \text{HOut} = \text{outPutState.enthalpy} \times \text{sum(outPutState.nVect)}; \]
\[ \text{QNet} = \text{HOut} - \text{HIn}; \]
\[ \text{end} \]
\[ \text{else} \]
% -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
% \text{Fixed Temperature} -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
% -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
% preReactState = gasSolidMixingChamber(airIn,MeOx,gasObject,solidObject)
HIn = airIn.enthalpy*sum(airIn.nVect)+MeOx.enthalpy*sum(MeOx.nVect) + MeOx.mDotAsh*MeOx.ashEnthalpy;
equilScaleFactor = 1;
if sum(preReactState.nVect)<0.5
    equilScaleFactor = 100;
    preReactState.nVect = preReactState.nVect*equilibScaleFactor;
end

% new code -- if num of moles is too large scale it bck for the equilibrium calculator
if sum(preReactState.nVect)>500
    equilibScaleFactor = 100/sum(preReactState.nVect);
    preReactState.nVect = preReactState.nVect*equilibScaleFactor;
end

% -----------------------------------------
% react mixture
% -----------------------------------------
try
equilbrate(mixSolidGas,'TP');
catch
    try
equilbrate(mixSolidGas,'TP'); %disp('try loop 1')
catch
equilbrate(mixSolidGas,'TP'); disp('try loop 2')
end
C.3.5 Fuel Reactor Function

This function is the model of the fuel reactor.

```matlab
function [outPutState,QNet] = fuelReactor(solidObject,gasObject,fuelIn,MeRed,QLoss)

% Fuel Reactor Model
% CREATOR: E.A. Goldstein
```
% Date: 7-2-13
% Fuel Reactor
% INPUTS: Solid/Gas State, nDotVect, Air Flow Rate, Qloss
% OUTPUTS: Solid/Gas State, nDotVect

% UPDATE: 7-2-13
% Adiabatic fuel reactor metal oxide and fuel in

global nEnd sys

% check if the temp of the solid is the same as the gas (it should be...)
% if not, break calcs
if (fuelIn.Pres - MeRed.Pres)/(fuelIn.Pres + MeRed.Pres) > 1e-4
    error('gas pressure not equal in fuel reactor...')
end

if QLoss > 1
    error('Q-loss should be a decimal...i.e. less then 1')
end

% MIX REACTING STREAMS
preReactState = gasSolidMixingChamberT(fuelIn, MeRed, gasObject, solidObject);

% energy bal. of gas-solid-mixing chamber
HStart = fuelIn.enthalpy * sum(fuelIn.nVect) + fuelIn.mDotAsh * fuelIn.ashEnthalpy + MeRed.enthalpy * sum(MeRed.nVect) + MeRed.mDotAsh * MeRed.ashEnthalpy;
HIn = (preReactState.enthalpy * sum(preReactState.nVect) + preReactState.mDotAsh * preReactState.ashEnthalpy);

ebal.GSM = HStart - HIn;

% aBal.SGC = Elem.total *(fuelInAsGas.nVect - fuelIn.nVect);

if sys.adbF
    % Adiabatic System
    equilibScaleFactor = 1;
    if sum(preReactState.nVect) < 0.5
        equilibScaleFactor = 100;
        preReactState.nVect = preReactState.nVect * equilibScaleFactor;
    end

    % HP Equilibrium...
        preReactState.nVect(1:nEnd.gas));
    if preReactState.nVect(1:nEnd.gas) == 0
            ' ');
    end
if sum(preReactState.nVect(nEnd.gas+1:nEnd.total)) == 0
    % no solids...
    set(solidObject,'T',preReactState.Temp,'P',preReactState.Pres,'X',1);
else
    set(solidObject,'T',preReactState.Temp,'P',preReactState.Pres,'X',
        preReactState.nVect(nEnd.gas+1:nEnd.total));
end
mixSolidGas = Mixture({gasObject,sum(preReactState.nVect(1:nEnd.gas));
                       solidObject,sum(preReactState.nVect(nEnd.gas+1:nEnd.total))});

% -----------------------------------------
% react mixture
% -----------------------------------------
equilbrate(mixSolidGas , 'HP');
% extract out temp, pres, moles and species distributions
% ------------------------------------------
nMolesPhase = phaseMoles(mixSolidGas)/equilibScaleFactor; % number of
    moles present in each phase
nDotGas = nMolesPhase(1) * moleFractions(gasObject);
end
nDotSolid = nMolesPhase(2) * moleFractions(solidObject);
end
nVect = [nDotGas;nDotSolid]; % kmols/s

% determine enthalpy change due to chemical reactions and ect...
outPutState.Temp = temperature(gasObject);
outPutState.Pres = pressure(gasObject);
outPutState.nVect = nVect;

outPutState.enthalpy = enthalpyStream(outPutState,gasObject,solidObject);
outPutState.mDotAsh = preReactState.mDotAsh;
outPutState.ashEnthalpy = preReactState.ashEnthalpy;
HOut = outPutState.enthalpy * sum(outPutState.nVect) +
    outPutState.mDotAsh*outPutState.ashEnthalpy;
QNet = HOut - HIn;
if QLoss ~= 0
    notConverged = 1;
    QLoss_c = QLoss;
    QLoss_cp = 0;
    QLossCalc_cp = QNet/HIn;
    set(gasObject,'T',outPutState.Temp,'P',outPutState.Pres,'X',
        outPutState.nVect(1:nEnd.gas));
hGas = enthalpy_mass(gasObject); % J/kg
while notConverged
% reduce initial enthalpy by reducing the temp of the mixture...
set(gasObject, 'H', hGas *(1 - QLoss_c), 'P', outPutState_temp.Pres, 'X',
    outPutState_temp.nVect (1: nEnd.gas));

if sum(outPutState_temp.nVect (nEnd.gas+1: nEnd.total)) == 0
    % no solids... not sure what should be here...
    set(solidObject, 'T', temperature(gasObject), 'P',
    outPutState_temp.Pres, 'X', 'Cu:1');
else
    % set to temp of gas object b/c a reduction in gas enthalpy by
    % Qloss is not @ the same temp as the solid; the Mixture()
    % method sets the objects to the same temperature.... over
    % prediciting the answer...
    set(solidObject, 'T', temperature(gasObject), 'P',
    outPutState_temp.Pres, 'X', outPutState_temp.nVect (nEnd.gas+1: nEnd.total));
end

mixSolidGas = Mixture([gasObject, sum(outPutState_temp.nVect (1:
    nEnd.gas)); solidObject, sum(outPutState_temp.nVect (nEnd.gas
    +1: nEnd.total))]);
equilbrate(mixSolidGas, 'HP');

% re-calc final mole spec... and temp...
nMolesPhase = phaseMoles(mixSolidGas)/equilibScaleFactor; \%
    number of moles present in each phase

nDotGas = nMolesPhase (1) * moleFractions(gasObject);
nDotSolid = nMolesPhase (2) * moleFractions(solidObject);

nVect = [nDotGas; nDotSolid]; \% kmol/s

% determine enthalpy change due to chemical reactions and ect...
outPutState_temp.Temp = temperature(gasObject);
outPutState_temp.Pres = pressure(gasObject);
outPutState_temp.nVect = nVect;

outPutState_temp.enthalpy = enthalpyStream(outPutState_temp,  
    gasObject, solidObject);

HOut = outPutState_temp.enthalpy * sum(outPutState_temp.nVect);
C.3. SYSTEM MODEL

\[ Q_{\text{Net}} = H_{\text{Out}} - H_{\text{In}}; \]
\[ Q_{\text{LossCalc}_c} = \frac{(H_{\text{Out}} - H_{\text{In}})}{H_{\text{In}}}; \]

\[
\text{if abs}(Q_{\text{Loss}} - Q_{\text{LossCalc}_c}) < 1e^{-3}
\]
\[ \% \text{ converged} \]
\[ \text{notConverged} = 0; \]
\[
Q_{\text{Loss}_\text{New}} = Q_{\text{Loss}} \times \frac{(Q_{\text{Loss}_c} - Q_{\text{Loss}_cp})}{(Q_{\text{LossCalc}_c} - Q_{\text{LossCalc}_cp})};
\]
\[ \% \text{ update...} \]
\[ Q_{\text{Loss}_cp} = Q_{\text{Loss}_c}; \]
\[ Q_{\text{LossCalc}_cp} = Q_{\text{LossCalc}_c}; \]
\[ Q_{\text{Loss}_c} = Q_{\text{Loss}_\text{New}}; \]

\[ \text{outPutState.Temp} = \text{temperature}(\text{gasObject}); \]
\[ \text{outPutState.Pres} = \text{pressure}(\text{gasObject}); \]
\[ \text{outPutState.nVect} = \text{nVect}; \]

\[ \text{outPutState.enthality} = \text{enthalpyStream}(\text{outPutState}, \text{gasObject}, \qquad \rightarrow \text{solidObject}); \]
\[ H_{\text{Out}} = \text{outPutState.enthality} \times \text{sum(outPutState.nVect)}; \]
\[ Q_{\text{Net}} = H_{\text{Out}} - H_{\text{In}}; \]

\[ \text{else} \]
\[ \% \text{ Fixed Temperature} \]
\[ \text{equilibScaleFactor} = 1; \]
\[ \text{if sum(preReactState.nVect) < 0.5} \]
\[ \text{equilibScaleFactor} = 100; \]
\[ \text{preReactState.nVect} = \text{preReactState.nVect} \times \text{equilibScaleFactor}; \]
\[ \% \text{ new code-- if num of moles is too large scale it bck for the} \]
\[ \% \text{ equilibrium calculator} \]
\[ \text{if sum(preReactState.nVect) > 1050} \]
\[ \text{equilibScaleFactor} = 150 / \text{sum(preReactState.nVect)}; \]
\[ \text{preReactState.nVect} = \text{preReactState.nVect} \times \text{equilibScaleFactor}; \]
\[ \% \text{ TP Equilibrium...} \]
\[ \text{set(gasObject,'T',sys.TfuelReactor,'P',preReactState.Pres,'X',} \]
\[ \rightarrow \text{preReactState.nVect(1:nEnd.gas)}); \]
\[ \text{if preReactState.nVect(1:nEnd.gas) == 0} \]
\[ \text{set(gasObject,'T',sys.TairReactor,'P',preReactState.Pres,'X','N2:1');} \]
\[ \text{end} \]
if sum(preReactState.nVect(nEnd.gas+1:nEnd.total)) == 0
    % no solids...
    set(solidObject,'T',sys.TfuelReactor,'P',preReactState.Pres,'X',1);
else
    set(solidObject,'T',sys.TfuelReactor,'P',preReactState.Pres,'X',
        preReactState.nVect(nEnd.gas+1:nEnd.total));
end
mixSolidGas = Mixture([gasObject,sum(preReactState.nVect(1:nEnd.gas));
    solidObject,sum(preReactState.nVect(nEnd.gas+1:nEnd.total))]);

% -----------------------------------------
% react mixture
% -----------------------------------------
try
    equilibrate(mixSolidGas,'TP');
catch
    try
        equilibrate(mixSolidGas,'TP');
catch
        try
            equilibrate(mixSolidGas,'TP');
catch
            equilibrate(mixSolidGas,'TP');
        end
    end
end
% -------------------------------
% extract out temp, pres, moles and species distributions
% -------------------------------
nMolesPhase = phaseMoles(mixSolidGas)/equilibScaleFactor; % number of
    moles present in each phase
nDotGas = nMolesPhase(1) * moleFractions(gasObject);
nDotSolid = nMolesPhase(2) * moleFractions(solidObject);
nVect = [nDotGas;nDotSolid]; % kmols/s

% determine enthalpy change due to chemical reactions and ect...
outPutState.Temp = temperature(gasObject);
outPutState.Pres = pressure(gasObject);
outPutState.nVect = nVect;

outPutState.enthalpy = enthalpyStream(outPutState,gasObject,solidObject);
outPutState.mDotAsh = preReactState.mDotAsh;
outPutState.ashEnthalpy = preReactState.ashEnthalpy;
HOut = outPutState.enthalpy * sum(outPutState.nVect) +
    outPutState.mDotAsh*outPutState.ashEnthalpy;
C.3. SYSTEM MODEL

C.3.6 Solid Gas Separator

This function is the model of the solid gas separator.

```matlab
function [gasExhaust, solidExhaust] = solidGasSeparator(state, separationConst, 
    gasObject, solidObject)
% when the variable separationConst is 1, all the gas is separated from the solid;
% when it is zero all the gas goes with the solids;
% when it is some fraction; that fraction goes out as gas and the rest w
% assume solid separation is ideal...

global nEnd

% initialize the output variables...
solidExhaust = state;
solidExhaust.nVect = zeros(nEnd.total,1);
gasExhaust.Temp = state.Temp;
gasExhaust.Pres = state.Pres;
gasExhaust.nVect = zeros(nEnd.total,1);
gasExhaust.enthalpy = 0;

% solids
solidExhaust.nVect((nEnd.gas+1):nEnd.total) = state.nVect((nEnd.gas+1):nEnd.total);

% All solids go here
solidExhaust.nVect(1:nEnd.gas) = state.nVect(1:nEnd.gas) * (1 - separationConst);

% With some gas
solidExhaust.nVect(1:nEnd.gas) = state.nVect(1:nEnd.gas) * separationConst;

% remaining gas goes here...

% check if ash or carbon...

try
    solidExhaust.mDotAsh = state.mDotAsh;
solidExhaust.ashEnthalpy = state.ashEnthalpy;
catch
    solidExhaust.mDotAsh = 0;
solidExhaust.ashEnthalpy = 0;
end
```
% update enthalpy of streams...
solidExhaust.enthalpy = enthalpyStream(solidExhaust, gasObject, solidObject);
gasExhaust.enthalpy = enthalpyStream(gasExhaust, gasObject, solidObject);

C.3.7 Solid Fuel Heat of Formation Function

This function calculates the heat of formation of a solid fuel given its lower or higher heating value and its ultimate analysis.

function outPut = solidFuelHofF(solidFuel, gasObject, waterObject)

% check if initial composition sums to 1.. if not, force it to sum to 1
if abs(1 - sum(solidFuel.Composition)) >1e-15
    solidFuel.Composition = solidFuel.Composition/sum(solidFuel.Composition);
end

if solidFuel.mDot == 0
    outPut = 0;
    return
end

% DEFINES FUEL COMPOSITION IN
fuelCompIn(nAtom.Names.C) = solidFuel.Composition(nAtom.Names.C)*(1 - solidFuel.Sw/100);
fuelCompIn(nAtom.Names.H) = solidFuel.Composition(nAtom.Names.H)*(1 - solidFuel.Sw/100);
fuelCompIn(nAtom.Names.O) = solidFuel.Composition(nAtom.Names.O)*(1 - solidFuel.Sw/100);
fuelCompIn(nAtom.Names.S) = solidFuel.Composition(nAtom.Names.S)*(1 - solidFuel.Sw/100);
fuelCompIn(nAtom.Names.Cl) = solidFuel.Composition(nAtom.Names.Cl)*(1 - solidFuel.Sw/100);
fuelCompIn(nAtom.Names.Ar) = solidFuel.Composition(nAtom.Names.Ar)*(1 - solidFuel.Sw/100);

% DEFINE GAS COMPOSITION IN, kg/s
mDot.Carbon = fuelCompIn(nAtom.Names.C)*solidFuel.mDot;
C.3. SYSTEM MODEL

mDot.Hydrogen = fuelCompIn(nAtom.Names.H)*solidFuel.mDot;
mDot.Oxygen = fuelCompIn(nAtom.Names.O)*solidFuel.mDot; %NA*solidFuel.mDot;
mDot.Nitrogen = fuelCompIn(nAtom.Names.N)*solidFuel.mDot;
mDot.Sulfur = fuelCompIn(nAtom.Names.S)*solidFuel.mDot; %NA*solidFuel.mDot;
mDot.Chlorine = fuelCompIn(nAtom.Names.Cl)*solidFuel.mDot; %
mDot.Argon = fuelCompIn(nAtom.Names.Ar)*solidFuel.mDot; %
mDot.Ash = fuelCompIn(nAtom.Names.Ash)*solidFuel.mDot;
mDot.H2O = (solidFuel.Sw/100)*solidFuel.mDot; % divide by 100 to make

% Covert mass flow to molar flow rate kg/s to kmoles/s

% note: Heat of formation for dry ash free coal....

nDot.Carbon = mDot.Carbon*(1/MW.C_gr);
nDot.Hydrogen = mDot.Hydrogen / MW.H;
nDot.Oxygen = mDot.Oxygen / MW.O;
nDot.Sulfur = mDot.Sulfur / MW.S;
nDot.Argon = mDot.Argon / MW.AR;
nDot.Chlorine = mDot.Chlorine / MW.CL;

% add in heat of formation...
% determine enthalpy of formation of the fuel...necessary for enthalpy
%
cal's w solid fuels

% CxHyOzNw..ect...Ash + A*O2 -> xCO2 + y/2* H2O + w/2* N2 ... Ash
% all carbon -> CO2
% all hydrogen -> H2O
% all chlorine -> Cl2
% all sulfur -> SO2
% all Nitrogen -> N2

nDotHxCalc = zeros(nEnd.gas,1);
nDotHxCalc(nGas.CO2) = nDot.Carbon;
nDotHxCalc(nGas.SO2) = nDot.Sulfur;
nDotHxCalc(nGas.CL2) = nDot.Chlorine/2;
nDotHxCalc(nGas.N2) = nDot.Nitrogen/2;
nDotHxCalc(nGas.AR) = nDot.Argon;
nDotHxCalc(nGas.CL2) = nDot.Chlorine/2;

% if the higher heating value is given, say all H goes to liquid H2O
if solidFuel.HHV < 0 % not higher heating value
  nDotLiquidH2O = 0;
  nDotVaporH2O = nDot.Hydrogen/2; % all H in fuel becomes H2O(l)
else % if a HHV (i.e. liquid water....)
  nDotLiquidH2O = nDot.Hydrogen/2;
  nDotVaporH2O = 0; % all H in fuel becomes H2O(g)
end

nDotO2Added = -nDot.Oxygen/2 + (nDotHxCalc(nGas.CO2)+(nDotLiquidH2O+

nDotHxCalc(nGas.SO2)));
C.3.8 Solid Gas Caster Function

This function is used to initialize the equilibrium calculations. It is used to convert a solid stream into a gas and it adjusts the enthalpy based on the heat of formation of the solid fuel.

```matlab
function gasStruct = solidGasCasterHC(solidFuel, gasObject, solidObject, fracCharAsh)

% SOLIDGASCASTER Summary of this function goes here
% Detailed explanation goes here
```
%DESCRIPTION:
% a function that holds the information for all of the simulants it
% converts the mass flow rate given by the user to the individual elemental
% molar flow rates for a given human waste simulant type

% Type 1: Human Waste found in 2002-01-2402 Table 1
% Type 2: Poultry litter found in 2002-01-2402 Table 1
% Type 3:

% Function Inputs and Outputs

% INPUTS:
% - Mass flow rate (kg/s)
% - Poop Type (numerical code)
% - Sw, Moisture Content (given in percentage of mass flow ie 20% = 20)
% - Tin, Temperature (Kelvin)
% - Pin, Pressure (Pascals)

% OUTPUTS:
% - Ndot, Molar Flow Rate by chemical species
% - mDot ash
% - enthalpy out

global MW nAtom nEnd nSolid nGas

% check if initial composition sums to 1.. if not, force it to sum to 1
if abs(1 - sum(solidFuel.Composition)) > 1e-14
    solidFuel.Composition = solidFuel.Composition / sum(solidFuel.Composition);
end

% DEFINES FUEL COMPOSITION IN
fuelCompIn(nAtom.Names.C) = solidFuel.Composition(nAtom.Names.C)*(1-
    solidFuel.Sw/100); % atom percent dry x frac dry mass/total mass [=]
atom percent wet mass
fuelCompIn(nAtom.Names.H) = solidFuel.Composition(nAtom.Names.H)*(1-
    solidFuel.Sw/100);
fuelCompIn(nAtom.Names.O) = solidFuel.Composition(nAtom.Names.O)*(1-
    solidFuel.Sw/100);
fuelCompIn(nAtom.Names.N) = solidFuel.Composition(nAtom.Names.N)*(1-
    solidFuel.Sw/100);
fuelCompIn(nAtom.Names.S) = solidFuel.Composition(nAtom.Names.S)*(1-
    solidFuel.Sw/100);
fuelCompIn(nAtom.Names.Cl) = solidFuel.Composition(nAtom.Names.Cl)*(1-
    solidFuel.Sw/100);
fuelCompIn(nAtom.Names.Ar) = solidFuel.Composition(nAtom.Names.Ar)*(1-
    solidFuel.Sw/100);
APPENDIX C. MATLAB CODES

```matlab
% DEFINE GAS COMPOSITION IN, kg/s
mDot.Carbon = fuelCompIn(nAtom.Names.C)*solidFuel.mDot;  % kg/s
mDot.Hydrogen = fuelCompIn(nAtom.Names.H)*solidFuel.mDot;
% mDot.Oxygen = fuelCompIn(nAtom.Names.O)*solidFuel.mDot;  %NA*solidFuel.mDot;
% mDot.Nitrogen = fuelCompIn(nAtom.Names.N)*solidFuel.mDot;
% mDot.Sulfur = fuelCompIn(nAtom.Names.S)*solidFuel.mDot; %NA*solidFuel.mDot;
% mDot.Chlorine = fuelCompIn(nAtom.Names.Cl)*solidFuel.mDot; %
% mDot.Argon = fuelCompIn(nAtom.Names.Ar)*solidFuel.mDot; %
% mDot.Ash = fuelCompIn(nAtom.Names.Ash)*solidFuel.mDot;
% mDot.H2O = (solidFuel.Sw/100)*solidFuel.mDot;  % divide by 100 to make percentage

% % % mDot.charCarbon = 0;
% % % if fracCharAsh == 0
% % % mDot.charCarbon = fracCharAsh * mDot.Carbon - mDot.Ash;
% % % what fraction of the initial coal will remain as char after devolitization & oxidaton...
% % % mDot.charCarbon = mDot.Ash/(1 - fracCharAsh) * fracCharAsh;
% % % if mDot.charCarbon > mDot.Carbon
% % % error('this model is wrong')
% % % end
% % %
% % mDot.Carbon = mDot.Carbon - mDot.charCarbon;

% Covert mass flow to molar flow rate kg/s to kmoles /s
% note: H20 must be absorbed into nDot.Hydrogen and nDot.Oxygen
nDot.Carbon = mDot.Carbon*(1/MW.C_gr);  % kmoles/s of each atom type...
% mDot.Oxygen = mDot.Hydrogen/MW.H + mDot.H2O*2/MW.H2O;
% nDot.Oxygen = mDot.Oxygen/MW.O + mDot.H2O/MW.H2O;
% mDot.Nitrogen = mDot.Nitrogen/MW.N;
% mDot.Sulfur = mDot.Sulfur/MW.S;
% mDot.Argon = mDot.Argon/MW.AR;
% mDot.Chlorine = mDot.Chlorine/MW.CL;

% OUTPUT-> cast inlet species as molecular species in our database...
% nDot.Carbon < nDot.Oxygen
mDotVect(nGas.CO) = nDot.Carbon*MW.CO;
mDotVect(nGas.O2) = (nDot.Oxygen-nDot.Carbon)/2*MW.O2;
else
mDotVect(nSolid.C_gr) = nDot.Carbon*MW.C_gr;
mDotVect(nGas.O2) = nDot.Oxygen/2*MW.O2;
end
```
C.3. SYSTEM MODEL

mDotVect(nSolid.C_gr) = nDot.Carbon * MW.C_gr;
mDotVect(nGas.H2) = (nDot.Hydrogen - nDot.Sulfur * 2) / 2 * MW.H2;
mDotVect(nGas.O2) = nDot.Oxygen / 2 * MW.O2;
mDotVect(nGas.N2) = nDot.Nitrogen / 2 * MW.N2;
mDotVect(nGas.H2S) = nDot.Nitrogen * MW.H2S;
mDotVect(nGas.AR) = nDot.Argon * MW.AR;
mDotVect(nGas.CL2) = nDot.Cl2 / 2 * MW.CL2;

mDotVect(nSolid.Ash) = mDot.Ash;

% define molar flow rate
nDotVect = mDotVect(1:nEnd.total) / MW.vect; % kmols /s

% determine enthalpy of formation of the fuel...necessary for enthalpy
% cal's w solid fuels
% CxHyOzNw...ect...Ash + A*O2 -> xCO2 + y/2* H2O + w/2* N2 ... Ash
% all carbon -> CO2
% all hydrogen -> H2O
% all chlorine -> Cl2
% all sulfur -> SO2
% all Nitrogen -> N2

% add in gas above the solid...
% NOTE: Only carbon solid is allowed here...
gasStruct.nVect = (nDotVect + solidFuel.nVect); % kmoles /s
gasStruct.mDotAsh = mDot.Ash; % kg/s
gasStruct.ashEnthalpy = solidFuel.ashEnthalpy;
gasStruct.Temp = solidFuel.Temp; % temp not defined for the given
← enthalpy...
gasStruct.Pres = solidFuel.Pres;

% % gasStruct.charCarbon = mDot.charCarbon / MW.C_gr; % carbon remainin in
← ash, moles/sec
% % set(solidObject,'T',gasStruct.Temp,'P',gasStruct.Pres,'X','C_gr:1');
% % gasStruct.charCarbonEnthalpy = enthalpy_mole(solidObject);

gasStruct.enthalpy = (solidFuel.solidEnthalpy * solidFuel.mDotDAF + ... 
sum(solidFuel.nVect) * solidFuel.enthalpy + mDot.H2O* 
← solidFuel.liquidEnthalpy) / ... % -gasStruct.charCarbon 
← *gasStruct.charCarbonEnthalpy 
(sum(gasStruct.nVect)); % J/kmol

if isnan(gasStruct.enthalpy)
gasStruct.enthalpy = 0;
end
% Recall... @ 298 K the heating value is the heat of reaction for the fuel

% -LHV*mdot = mdot(hProd - hReact)
% = mDot(hProd - deltaHF - hO2)

end

C.3.9 Ash Sensible Enthalpy Function

This function calculates the sensible enthalpy of the ash in the fuel.

function outPut = ashSensEnthalpy(Tin)

% sensible enthalpy of ash; Merrick, "Mathematica decomposition models of
% the thermal decomposition of coal"

outPut = 754 + 0.586*Tin; % J/kg-K

C.3.10 Flow Exergy Function

This function calculates the flow exergy of a stream.

function outPut = flowExergyStream(activeState,deadState,gasObject,solidObject)

% this function takes in a given state and a reference (dead state); using
% this information, along with gas and solid objects, the flow exergy
% (kJ/kg) is calculated. The flow exergy is defined as Phi = (h-T0*s) - (h0-T0*s0
% -> ) = (h-T0*s) - g0

% global nEnd nGas nSolid sys

set(gasObject,'T',activeState.Temp,'P',activeState.Pres,'X',activeState.nVect(1:
% -> nEnd.gas));

xInit.gas = moleFractions(gasObject);
hInit.gas = enthalpy_mole(gasObject);
sInit.gas = entropy_mole(gasObject);
if isnan(hInit.gas)
    xInit.gas = zeros(nSpecies(gasObject),1);
hInit.gas = 0;
sInit.gas = 0;
end
C.3. SYSTEM MODEL

20  xInit.solid = moleFractions(solidObject);
21  hInit.solid = enthalpy_mole(solidObject);
22  sInit.solid = entropy_mole(solidObject);
23  if isnan(hInit.solid)
24      xInit.solid = zeros(nSpecies(solidObject),1);
25      hInit.solid = 0;
26      sInit.solid = 0;
27  end
28
29  n.Total = nSpecies(solidObject) + nSpecies(gasObject);
30  a.Total = nElements(gasObject) + 1; % total number of gases + Cu...
31  Elem.Total = zeros(n.Total, a.Total);
32  ElemDeadState.Total = zeros(sum(deadState.nVect((nEnd.gas+1):nEnd.total)>0)+sum(deadState.nVect(1:nEnd.gas)>0), a.Total);
33
34  for i = 1:nEnd.solid
35      nVect(i) = 1;
36      set(solidObject, 'T', deadState.Temp, 'P', deadState.Pres, 'X', nVect);
37      chemPotVect.solid(i) = gibbs_mole(solidObject); % all separate phases... so
38          % should be pure component values...
39      nVect(i) = 0;
40  end
41  n.Vector = zeros(nEnd.solid,1);
42  for i = 1:nEnd.solid
43      nVect(i) = 1;
44      set(solidObject, 'T', deadState.Temp, 'P', deadState.Pres, 'X', nVect);
45      chemPotVect.solid(i) = gibbs_mole(solidObject); % all separate phases... not
46          % sure what value to use right now...
47  end
48
49  set(solidObject, 'T', deadState.Temp, 'P', deadState.Pres, 'X', activeState.nVect((nEnd.gas+1):nEnd.total));
50  % chemPotVect.solid = chemPotentials(solidObject); % all separate phases... not
51  % sure what value to use right now...
53  % chemPotVect.gas = chemPotentials(gasObject);
% set (gasObject,'T',deadState.Temp,'P',deadState.Pres,'X',deadState.nVect(1:nEnd.gas));

% * * * * * * * * * * * * * * * * * * * * * * * * * * * *

kk = 1;
for k = 1:nSpecies(solidObject)
    if deadState.nVect(k+nEnd.gas) ==0
        for j = 1:nElements(solidObject)
            ElemDeadState.solid(kk,j)= nAtoms(solidObject,k,j);
        end
        chemPotDeadState.solid(kk,1) = chemPotVect.solid(k);
        kk = kk+1;
    end
    for j = 1:nElements(solidObject)
        Elem.solid(k,j)= nAtoms(solidObject,k,j);
    end
end

% take extracted data and cast atoms into dead state species...
% * * * * * * * * * * * * * * * * * * * * * * * * * * * *

% pattern Elem.total
Elem.Total(1:nSpecies(gasObject),2:nElements(gasObject)+1) = Elem.gas;
Elem.Total(nSpecies(gasObject)+1:end,1:nElements(solidObject)) = Elem.solid;

numSpecDead.gas = sum((deadState.nVect(1:nEnd.gas)>0));
numSpecDead.solid = sum((deadState.nVect(1)>0))-numSpecDead.gas;

ElemDeadState.Total(1:1<numSpecDead.gas,2:nElements(gasObject)+1) =
    ElemDeadState.gas;
ElemDeadState.Total(1:(numSpecDead.gas+1):numSpecDead.gas+numSpecDead.solid,1:
    nElements(solidObject)) = ElemDeadState.solid;

% solves Ax = b where b is the atoms available and A is the
C.4 Auxiliary Functions

Below is auxiliary code that was used in two or more of the models given above.
C.4.1 Defect Equilibrium Concentrations

This code calculates the defect equilibrium concentrations as a function of temperature, oxygen partial pressure, impurity concentration and formation energies for the defect species. A charge neutrality constraint is imposed on the calculations and is used as a relation to solve the defect concentrations.

```matlab
function [regSite Keq] = defectEquilibriumFuncDHi(sys, deltaGCuO, impur) % temp,pres
% plot equilibrium defect populations as a function of T,PO2
% concentrations given in moles per mole of oxygen in Cu2O...
% pressures in ATM
% account for impurities in Cu20 and CuO... assume fully charge compensated...

global rho Cn MW

% define active defects:
% Cu2O CuO
% VCuX VCu', e', h.,Cu1,Cu. VCu'' , e', h.

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * numDens.phase.defectCharge * * * * * * * * * * * * * * * *
% * * * * * * * regular site concentration * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
regSite.Cu.int = regSite.Cu.Cu *2; % sites/cm^3 , interstial site density.... <--- check...

regSite.Cu2O.int = regSite.Cu2O.Cu *8; % moles of inter / moles of Cu2O

regSite.CuO.int = regSite.CuO.Cu *4; % sites/cm^3, interstial site density....

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% * * * * * * * Other Copper Properties * * * * * * * * * * * * * * * *
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

C.4. AUXILIARY FUNCTIONS

bgAlpha.Cu2O = 4.8e-4; % eV/K
bgBeta.Cu2O = 275; % K
eG0.Cu2O = 2.173; % eV, band gap at 0 K... from Italy paper, 129---Defects and doping in Cu2O
prop.Cu2O.elecMass = 0.61 * 9.10938188 * 10^-31; % kg...
prop.Cu2O.holeMass = 0.84 * 9.10938188 * 10^-31; % kg...
bgAlpha.CuO = 0 * 4.8e-4; % eV/K <----- CHECK VALUE
bgBeta.CuO = 275; % K <----- CHECK VALUE ----
eG0.CuO = 1.7; % eV, band gap at 0 K... assume same slope as Cu2O... not true
% wikipeida says 1.2 at room temp
prop.CuO.elecMass = 8 * 9.10938188 * 10^-31; % kg... <----- CHECK VALUE ----
prop.CuO.holeMass = 8 * 9.10938188 * 10^-31; % kg... <----- CHECK VALUE ----

% Rate Constants and Equilibrium Constants

% Rx1: 1/2 O2 <=> Ox + 2VCu + 2h.
\[
\Delta G_{R1Cu2O} = 2 \times (94 \times 1000 - (-41 \times \text{sys.temp})) \text{ J/mol}
\]
% Rx2: V' + h. \rightleftharpoons Vx
\[
\Delta G_{R2Cu2O} = (-28 \times 1000 - (53 \times \text{sys.temp})) \text{ J/mol}
\]
% Rx3: 0x + 2Cu + 2h \rightleftharpoons 2Cu_i. + 1/2 O2
\[
\Delta G_{R3Cu2O} = 2 \times ((103) \times 1000 - ((57) \times \text{sys.temp})) \text{ J/mol}
\]
% Rx4: Cu_i. + e' \rightleftharpoons Cu_{ix}
\[
\Delta G_{R4Cu2O} = (-132 \times 1000 - (26 \times \text{sys.temp})) \text{ J/mol}
\]
% Rx5: 2h + Ox \rightleftharpoons 1/2 O2 + V_O..
\[
\Delta G_{R5Cu2O} = ((294.4) \times 1000 - ((106) \times \text{sys.temp})) \text{ J/mol, from Xu & Diekmann, 169... for non-ionized---adjusted to account for based on rx2}
\]

% Frenkel Equilibrium, cation \rightleftharpoons x \rightarrow Cu_i. + V_Cu'
\[
\Delta G_{RFECu2O} = 201 \times 1000 - (19 \times \text{sys.temp}) \text{ J/mol}
\]
deltaG.R3CuO = ((294.4+28*3/2) *1000 - ((106 -53*3/2) * sys.temp )); % J/mol , from Xu & Diekmann, 169...for non-ionized---adjusted to account for based on rx2
% Frenkel Equilibrium, cation --> Cu->Cu_i.. + V_Cu''
deltaG.RFECuO = 0; % J/mol

prop.Cu2O.bandGap = eG0.Cu2O - bgAlpha.Cu2O * sys.temp ^2/( sys.temp + bgBeta.Cu2O ) ; % eV
prop.Cu2O.valDOS = 2*((2* pi* prop.Cu2O.holeMass * Cn.kB * sys.temp )/ Cn.hPlank_JS ^2) ^(3/2) /100^3; % atoms /cm ^3
prop.Cu2O.conDOS = 2*((2* pi* prop.Cu2O.elecMass * Cn.kB * sys.temp )/ Cn.hPlank_JS ^2)^(3/2) /100^3; % atoms /cm ^3

Keq.Cu2O.Rx1 = exp (- deltaG.R1Cu2O /( Cn.gasconst * sys.temp )); % Cu vac. -
Keq.Cu2O.Rx2 = exp (- deltaG.R2Cu2O /( Cn.gasconst * sys.temp )); % Cu vac.
Keq.Cu2O.Rx3 = exp (- deltaG.R3Cu2O /( Cn.gasconst * sys.temp )); % Cu int.
Keq.Cu2O.Rx4 = exp (- deltaG.R4Cu2O /( Cn.gasconst * sys.temp )); % Cu int. +
Keq.Cu2O.Rx5 = exp (- deltaG.R5Cu2O /( Cn.gasconst * sys.temp )); % Ox vac. ++
Keq.Cu2O.frenkel = exp (- deltaG.RFECu2O /( Cn.gasconst * sys.temp ));

Keq.CuO.Rx1 = exp (- deltaG.R1CuO /( Cn.gasconst * sys.temp )); % Cu vac --
Keq.CuO.Rx2 = exp (- deltaG.R2CuO /( Cn.gasconst * sys.temp )); % Cu int ++
Keq.CuO.Rx3 = 0*exp (- deltaG.R3CuO /( Cn.gasconst * sys.temp )); % Ox vac ++
Keq.CuO.frenkel = 0*exp (- deltaG.RFECuO /( Cn.gasconst * sys.temp ));
Keq.CuO.eh = 6*(( rho.CuO/MW.CuO ))^2 * exp (- prop.CuO.bandGap /( Cn.kB_eV * sys.temp )); % ( moles/mole of oxygen )^2

% sys.presEqu = phasePressure(sys.temp,sys.presTot);
% sys.presCuCu2O_atm = sys.presEqu(1)/101325;
% sys.presCu2OCuO_atm = sys.presEqu(2)/101325;
% determine number of phases....
if sys.presO2_atm >= sys.presCuCu2O_atm && sys.presO2_atm <
    sys.presCu2OCuO_atm
nPhase = 2; % for only Cu & Cu2O...
isCu0 = 0;
elseif sys.presO2_atm >= sys.presCu2OCuO_atm
isCu0 = 1;
nPhase = 3; % for only Cu & Cu2O..
else
    % pressure too low, no oxidation...
isCu0 = 0;
nPhase = 1;
end

% HOLE concentration ->: solve charge neutrality equation
coef.a = 2 * Keq.Cu2O.Rx5 / sys.presCuCu2O_atm^(1/2); % h^-3
coef.b = 1 + Keq.Cu2O.Rx3^(1/2) / sys.presCuCu2O_atm^(1/4); % h^-2
coef.c = impur.Cu2O.pos - impur.Cu2O.neg; % h^-1
coef.d = Keq.Cu2O.eh + Keq.Cu2O.Rx1^(1/2) * sys.presCuCu2O_atm^(1/4); % h^0
sol.roots = roots([coef.a,coef.b,coef.c,-coef.d]);
hdot = sol.roots(find(imag(sol.roots)==0 & sol.roots>0));
if isempty(hdot)
    sol.roots = roots([coef.b,0,-coef.c]);
    hdot = sol.roots(find(imag(sol.roots)==0 & sol.roots>0));
end
regSite.h.CuCu2O = hdot;
regSite.e.CuCu2O = Keq.Cu2O.eh/regSite.h.CuCu2O; % moles/mole of oxygen
regSite.vacCuM.CuCu2O = Keq.Cu2O.Rx1^(1/2) * sys.presCuCu2O_atm^(1/4) /
    regSite.h.CuCu2O; % moles/mole of oxygen
regSite.vacCuX.CuCu2O = Keq.Cu2O.Rx2 * regSite.vacCuM.CuCu2O * regSite.h.CuCu2O;
    % moles/mole of oxygen
regSite.vacOPP.CuCu2O = regSite.h.CuCu2O^2 * Keq.Cu2O.Rx5 / sys.presCuCu2O_atm^(1/2); % moles/mole of oxygen
regSite.intCuP.CuCu2O = Keq.Cu2O.Rx3^(1/2) * regSite.h.CuCu2O /
    sys.presCuCu2O_atm^(1/4); % moles/mole of oxygen
regSite.intCuX.CuCu2O = Keq.Cu2O.Rx4 * regSite.intCuP.CuCu2O *
    regSite.e.CuCu2O; % moles/mole of oxygen
if ~isCu0
    % from charge neutrality...
    coef.a = 2 * Keq.Cu2O.Rx5 / sys.presO2_atm^(1/2); % h^-3
    coef.b = 1 + Keq.Cu2O.Rx3^(1/2) / sys.presO2_atm^(1/4); % h^-2
    coef.c = impur.Cu2O.pos - impur.Cu2O.neg; % h^-1
    coef.d = Keq.Cu2O.eh + Keq.Cu2O.Rx1^(1/2) * sys.presO2_atm^(1/4); % h^0
    sol.roots = roots([coef.a,coef.b,coef.c,-coef.d]);
hdot = sol.roots(find(imag(sol.roots)==0 & sol.roots>0));

if isempty(hdot)
    sol.roots = roots([coef.b,0,-coef.c]);
    hdot = sol.roots(find(imag(sol.roots)==0 & sol.roots>0));
end

regSite.h.Cu2OO2 = hdot; % moles of holes / mole of oxygen
regSite.e.Cu2OO2 = Keq.Cu2O.eh/regSite.h.Cu2OO2; % moles/mole of oxygen
regSite.vacCuM.Cu2OO2 = Keq.Cu2O.Rx1^(1/2) * sys.presO2_atm^(1/4) / regSite.h.Cu2OO2; % moles/mole of oxygen
regSite.vacCuX.Cu2OO2 = Keq.Cu2O.Rx2 * regSite.vacCuM.Cu2OO2 * regSite.h.Cu2OO2 / sys.presO2_atm^(1/4); % moles/mole of oxygen
regSite.vacOPP.Cu2OO2 = Keq.Cu2O.Rx5 * regSite.h.Cu2OO2^2 / sys.presO2_atm^(1/2); % moles/mole of oxygen
regSite.intCuP.Cu2OO2 = Keq.Cu2O.Rx3^(1/2) * regSite.h.Cu2OO2 / sys.presO2_atm^(1/4); % moles/mole of oxygen
regSite.intCuX.Cu2OO2 = Keq.Cu2O.Rx4 * regSite.intCuP.Cu2OO2 * regSite.e.Cu2OO2; % moles/mole of oxygen
else
    % from charge neutrality...
    coef.a = 2 * Keq.Cu2O.Rx5 / sys.presCu2OCuO_atm^(1/2); % h^-3
    coef.b = 1 + Keq.Cu2O.Rx3^(1/2) / sys.presCu2OCuO_atm^(1/4); % h^-2
    coef.c = impur.Cu2O.pos - impur.Cu2O.neg; % h^-1
    coef.d = Keq.Cu2O.eh + Keq.Cu2O.Rx1^(1/2) * sys.presCu2OCuO_atm^(1/4); % h^0
    sol.roots = roots([coef.a,coef.b,coef.c,-coef.d]);
    hdot = sol.roots(find(imag(sol.roots)==0 & sol.roots>0));
if isempty(hdot)
    sol.roots = roots([coef.b,0,-coef.c]);
    hdot = sol.roots(find(imag(sol.roots)==0 & sol.roots>0));
end

regSite.h.Cu2OCuO = hdot; % moles of holes / mole of oxygen
regSite.e.Cu2OCuO = Keq.Cu2O.eh/regSite.h.Cu2OCuO; % moles/mole of oxygen
regSite.vacCuM.Cu2OCuO = Keq.Cu2O.Rx1^(1/2) * sys.presCu2OCuO_atm^(1/4) / regSite.h.Cu2OCuO; % moles/mole of oxygen
regSite.vacCuX.Cu2OCuO = Keq.Cu2O.Rx2 * regSite.vacCuM.Cu2OCuO / sys.presCu2OCuO_atm^(1/2); % moles/mole of oxygen
regSite.vacOPP.Cu2OCuO = Keq.Cu2O.Rx5 * regSite.h.Cu2OCuO^2 / sys.presCu2OCuO_atm^(1/4); % moles/mole of oxygen
regSite.intCuP.Cu2OCuO = Keq.Cu2O.Rx3^(1/2) * regSite.h.Cu2OCuO / sys.presCu2OCuO_atm^(1/4); % moles/mole of oxygen
regSite.intCuX.Cu2OCuO = Keq.Cu2O.Rx4 * regSite.intCuP.Cu2OCuO / regSite.e.Cu2OCuO; % moles/mole of oxygen
% from charge neutrality...

\[
\text{coef.a} = 2 \times \frac{\text{Keq.CuO.Rx3}}{\text{sys.presCu2OCuO\_atm}^{(1/2)}} + 2 \times \frac{\text{Keq.CuO.Rx2}}{\text{sys.presCu2OCuO\_atm}^{(1/2)}}; \quad \% \ \text{h}^{-4}
\]

\[
\text{coef.b} = 1; \quad \% \ \text{h}^{-3}
\]

\[
\text{coef.c} = \text{impur.CuO.pos} - \text{impur.CuO.neg}; \quad \% \ \text{h}^{-2}
\]

\[
\text{coef.d} = -\frac{\text{Keq.CuO.eh}}{\text{sys.presCu2OCuO\_atm}^{(1/2)}}; \quad \% \ \text{h}^{-1}
\]

\[
\text{coef.e} = -2 \times \frac{\text{Keq.CuO.Rx1} \times \text{sys.presCu2OCuO\_atm}^{(1/2)}}{}; \quad \% \ \text{h}^{0}
\]

\[
\text{sol.roots} = \text{roots}([\text{coef.a}, \text{coef.b}, \text{coef.c}, \text{coef.d}, \text{coef.e}]);
\]

\[
\text{hdot} = \text{sol.roots}(\text{find(\text{imag(\text{sol.roots})=0 & sol.roots>0}))};
\]

\[
\text{regSite.h.CuOCu2O} = \text{hdot}; \quad \% \ \text{moles of holes / mole of oxygen}
\]

\[
\text{regSite.e.CuOCu2O} = \frac{\text{Keq.CuO.eh}}{\text{regSite.h.CuOCu2O}}; \quad \% \ \text{moles/mole of oxygen}
\]

\[
\text{regSite.vacCuMM.CuOCu2O} = \frac{\text{Keq.CuO.Rx1} \times \text{sys.presCu2OCuO\_atm}^{(1/2)}}{}; \quad \% \ \text{moles/mole of oxygen}
\]

\[
\text{regSite.vacOPP.CuOCu2O} = \frac{\text{Keq.CuO.Rx3} \times \text{regSite.h.CuOCu2O}^2}{\text{sys.presCu2OCuO\_atm}^{(1/2)}}; \quad \% \ \text{moles/mole of oxygen}
\]

\[
\text{regSite.intCuPP.CuOCu2O} = \frac{\text{Keq.CuO.Rx2} \times \text{regSite.h.CuOCu2O}^2}{\text{sys.presCu2OCuO\_atm}^{(1/2)}}; \quad \% \ \text{moles/mole of oxygen}
\]

end
C.4.2 Derivative Code

Code was developed to calculate the derivative of experimental data or output from the model. To calculate the derivative of a data set, a 5\textsuperscript{th} order Padé scheme was used.

```matlab
function df = dxCalc(f,t)
% function takes the derivative of a input vector
% input f(x),x

N = length(f); % number of data points
% steps = N;

% spline data to make uniform grid...
if N > 1000
  reSamp = 1000;
else
  reSamp = N;
end

tRange = linspace(t(1),t(N),reSamp);
fSamp = spline(t,f,tRange);
dt = (tRange(reSamp)-tRange(1))/(reSamp-1);
% tRange = t;
% fSamp = f;
% dt = t(3)-t(4);

% dt = (t(N)-t(1))/(N-1);
% use unidirectional pade scheme at end points; 2nd order
% df(1) = (-3*f(1)+4*f(2)-f(3))/(2*dt);
% df(N) = (-3*f(N)+4*f(N-1)-f(N-2))/(2*dt);

% Define pade matrix
% preallocate...
aii = 4*ones(reSamp,1);
aii(1) = 1;
aii(reSamp) = 1;

aijp = ones(reSamp-1,1);
aijm = ones(reSamp-1,1);
aijp(1)=2;
aijm(reSamp-1)=2;

A = diag(aii) + diag(aijp,1) + diag(aijm,-1);```

C.4. AUXILIARY FUNCTIONS

C.4.3 Data Smoothing Code

This code takes an input data and calculates a moving average.

```matlab
function [yResult xResult] = DataSmother(v,p)

% program takes in period and vector of data and averages data points...

% v = vector of data to be "smoothed"
% p = the number of data points averaged per point in final data set

n = round(length(v)/p); % number of data points in final set
yResult = zeros(n,1); % condensed vector of y points
xResult = yResult; % condensed vector of x points

for i = 1:n
    startP = (i-1)*p+1;
    endP = startP + p;
    range = startP:endP;
    yResult(i) = sum(v(range))/p;
end
```

C.4.4 Equilibrium Oxygen Activity

This code was used to calculate the equilibrium oxygen activity at the Cu/Cu$_2$O and Cu$_2$O/CuO interface. Thermodynamic data from Cantera is used for the Cu/Cu$_2$O/CuO/O$_2$ species.

```matlab
function outPut = phasePressure(temp,pTot)
% number of phases present at a given T,P...

global Cn

% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
% initialize phase information
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
gasObject = importPhase('coalMix.xml'); %('GRI30.xml');
set(gasObject,'X','N2:1.0','T',temp,'P',pTot);
solidObject = importPhase('solids.xml');
set(solidObject,'X','Cu:1','T',temp,'P',pTot);
% * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

% xVect = zeros(nSpecies(gasObject),1);
% nO2 = speciesIndex(gasObject,'O2');
% nN2 = speciesIndex(gasObject,'N2');
% xVect(nO2) = pO2/pTot; % if 1 then pure O2 if < 1 then mixed with N2...
% xVect(nN2) = (pTot-pO2)/pTot;

set(gasObject,'T',temp,'P',pTot,'X','O2:1');
G.O2 = gibbs_mole(gasObject);

set(solidObject,'T',temp,'P',pTot,'X','Cu:1');
G.Cu = gibbs_mole(solidObject);
```
C.4. AUXILIARY FUNCTIONS

C.4.5 Diffusion Coefficient Calculations

This code calculates the defect diffusion coefficients given the temperature and oxygen activity of the system.

```matlab
function outPut = DeffFuncDefect(lattice, defect)
global sys Cn
if strcmp(lattice, 'Cu2O')
    if strcmp(defect, 'vacCuX')
        outPut = 3 * 10^-3 * 7e-6 * exp((-52.4-40.38)*10^-3/(Cn.gasconst*sys.temp)); % cm^-2/s, diffusivity of vacancies... cation vacancy diffusion coef.
    elseif strcmp(defect, 'intCuX')
        outPut = 1e-9; % cm^-2/s
    elseif strcmp(defect, 'vacCuM')
        outPut = 3 * 10^-3 * 7e-6 * exp((-52.4+40.38)*10^-3/(Cn.gasconst*sys.temp)); % cm^-2/s, diffusivity of vacancies... cation vacancy diffusion coef.
    elseif strcmp(defect, 'intCuM')
        outPut = 3 * 10^-3 * 7e-6 * exp((-52.4+40.38)*10^-3/(Cn.gasconst*sys.temp)); % cm^-2/s, diffusivity of vacancies... cation vacancy diffusion coef.
    elseif strcmp(defect, 'vacCuX')
        outPut = 3 * 10^-3 * 7e-6 * exp((-52.4-40.38)*10^-3/(Cn.gasconst*sys.temp)); % cm^-2/s, diffusivity of vacancies... cation vacancy diffusion coef.
    elseif strcmp(defect, 'intCuX')
        outPut = 1e-9; % cm^-2/s
    end
elseif strcmp(lattice, 'CuO')
    if strcmp(defect, 'vacCuX')
        outPut = 3 * 10^-3 * 7e-6 * exp((-52.4-40.38)*10^-3/(Cn.gasconst*sys.temp)); % cm^-2/s, diffusivity of vacancies... cation vacancy diffusion coef.
    elseif strcmp(defect, 'intCuX')
        outPut = 1e-9; % cm^-2/s
    elseif strcmp(defect, 'vacCuM')
        outPut = 3 * 10^-3 * 7e-6 * exp((-52.4+40.38)*10^-3/(Cn.gasconst*sys.temp)); % cm^-2/s, diffusivity of vacancies... cation vacancy diffusion coef.
    elseif strcmp(defect, 'intCuM')
        outPut = 3 * 10^-3 * 7e-6 * exp((-52.4+40.38)*10^-3/(Cn.gasconst*sys.temp)); % cm^-2/s, diffusivity of vacancies... cation vacancy diffusion coef.
    elseif strcmp(defect, 'vacCuX')
        outPut = 3 * 10^-3 * 7e-6 * exp((-52.4-40.38)*10^-3/(Cn.gasconst*sys.temp)); % cm^-2/s, diffusivity of vacancies... cation vacancy diffusion coef.
    elseif strcmp(defect, 'intCuX')
        outPut = 1e-9; % cm^-2/s
    end
end
```

C.4. AUXILIARY FUNCTIONS

293

set(solidObject,'T',temp,'P',pTot,'X','Cu2O:1');
G.Cu20 = gibbs_mole(solidObject);
set(solidObject,'T',temp,'P',pTot,'X','CuO:1');
G.CuO = gibbs_mole(solidObject);

% 4Cu + O2 -> 2Cu2O
deltaG.ra = 2*(G.Cu2O - 2*G.Cu - 1/2*G.O2)/1000; % J/mole

% 2Cu2O + O2 -> 4*CuO
deltaG.rb = 4*(G.CuO - 1/2*G.Cu2O - 1/4*G.O2)/1000;

outPut = 101325*[exp(deltaG.ra/(Cn.gasconst*temp)); exp(deltaG.rb/(Cn.gasconst*temp))];

% chemical diffusion
% In the above equations, the standard states for Cu, CuO, and Cu2O are defined as follows:
% Cu (s) = Cu (0.5 mol/L)
% CuO (s) = CuO (0.5 mol/L)
% Cu2O (s) = Cu2O (0.5 mol/L)
% The activities of these species are calculated using the following equations:
% a(Cu) = (Cu) / (Cu (0.5 mol/L))
% a(CuO) = (CuO) / (CuO (0.5 mol/L))
% a(Cu2O) = (Cu2O) / (Cu2O (0.5 mol/L))
% The diffusion coefficients of Cu vacancies and Cu interstitials in Cu2O and CuO are calculated using the following equations:
% D_vac = 3 * 10^-3 * exp(-52.4 + 40.38 * temp / Cn.gasconst * sys.temp); % cm^2/s, cation vacancy diffusion coef.
% D_int = 1e^-9; % cm^2/s
% The diffusion coefficients of Cu vacancies and Cu interstitials in CuO are calculated using the following equations:
% D_vac = 3 * 10^-3 * exp(-52.4 + 40.38 * temp / Cn.gasconst * sys.temp); % cm^2/s, cation vacancy diffusion coef.
% D_int = 1e^-9; % cm^2/s
% For Cu vacancies and Cu interstitials in Cu2O, the diffusivities are calculated as:
% D_vac = 3 * 10^-3 * 7e^-6 * exp((-52.4 + 40.38) * temp / Cn.gasconst * sys.temp); % cm^-2/s
% D_int = 1e^-9; % cm^-2/s

C.4.5 Diffusion Coefficient Calculations

This code calculates the defect diffusion coefficients given the temperature and oxygen activity of the system.
% outPut = 3 * 10^-3 * exp(-52.4*10^-3/(Cn.gasconst*sys.temp)); % cm^2/s, diffusivity of vacancies... cation vacancy diffusion coef.

elseif strcmp(defect,'intCuP')
    outPut = 1e-9; % cm^2/s

elseif strcmp(defect,'vacOPP')
    outPut = 3*10^-3 * sys.presO2_atm^0.4*exp(-35.700*4.184*10^3/(Cn.gasconst*sys.temp)); % cm^2/s --- Paper 217, Argone -- self diffusion coeff

elseif strcmp(defect,'elec')
    % outPut = 1e-2; % cm^2/s
    % mueElec = 175; % cm^2/V-sec , mobility between 1000 K and 1250 K roughly constant...
    % mueElec = 1.5*10^23*sys.temp^(-6.766); % cm^2/V-sec , mobility between 1000 K and 1250 K roughly constant...
    outPut = mueElec * Cn.kB * sys.temp / (abs(-1)*Cn.e); % cm^2/s

elseif strcmp(lattice,'CuO')
    if strcmp(defect,'vacCuMM')
        % for a Cu atom, traveling through Cu vacancies..
        % outPut = 1.5 * 10^-10 *10289*sys.presO2_atm^(-1/2)*exp(-147+20.4) /
        % 10^-3/(Cn.gasconst*sys.temp)*100^-2; % cm^2/s, copper ion... need to multiply by [Cu]/[V].... copper vacancies should move faster than ions..
        outPut = 1.5 * 10^-10 *sys.presO2_atm^(-1/2)*exp(-147*10^-3/(Cn.gasconst* sys.temp))*100^-2; % cm^2/s, copper ion... need to multiply by [Cu]/[V].... copper vacancies should move faster than ions..
    end

else
    outPut = 1e-9; % cm^2/s
end
slopeTerm = -0.509142903; % log10(conductivity) = 10^-4/T * slope + intercept

interceptTerm = 5.130385206; % from paper 154, "Oxidation of Cu and electronic transport in copper oxides"

sigsTotCuO = 10^(slopeTerm * 10^-4/sys.temp + interceptTerm);

outPut = sigsTotCuO * Cn.kB * sys.temp/(Cn.e*1)^2/Cn.Na; % cm^-2/s * moles/cm^3

elseif strcmp(defect,'elec')
  % outPut = 1e-2; % cm^-2/s
  % bRatio = 5;
  end

else
  error('wrong input== diffusivity...')
end
Bibliography


[125] Antti Roine. HSC Simulation.


