FLOW SHEET OPTIMIZATION BY THE CONCEPT OF SUSTAINABLE DEVELOPMENT: ALUMINA INDUSTRY

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ABDURRAHMAN KURUCAK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
CHEMICAL ENGINEERING

FEBRUARY 2010

Approval of the thesis:

FLOW SHEET OPTIMIZATION BY THE CONCEPT OF SUSTAINABLE DEVELOPMENT: ALUMINA INDUSTRY

submitted by **ABDURRAHMAN KURUCAK** in partial fulfillment of the requirements for the degree of **Master of Science in Chemical Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Gürkan KARAKAŞ
Head of Department, **Chemical Engineering**

Prof. Dr. Güniz GÜRÜZ
Supervisor, **Chemical Engineering Dept., METU**

Prof. Dr. İsmail TOSUN
Co-Supervisor, **Chemical Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. Hayrettin YÜCEL
Chemical Engineering Dept., METU

Prof. Dr. Güniz GÜRÜZ
Chemical Engineering Dept., METU

Prof. Dr. Ülkü YETİŞ
Environmental Engineering Dept., METU

Assoc. Prof. Dr. Halil KALIPÇILAR
Chemical Engineering Dept., METU

Şükrü DİKKAYA
Head of Grup, Petroleum Market Dept., EMRA

Date : 05.02.2010

PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Abdurrahman KURUCAK

Signature :

ABSTRACT

FLOW SHEET OPTIMIZATION BY THE CONCEPT OF SUSTAINABLE DEVELOPMENT: ALUMINA INDUSTRY

Kurucak, Abdurrahman
M.Sc., Department of Chemical Engineering
Supervisor : Prof. Dr. Güniz Gürüz
Co-Supervisor: Prof. Dr. İsmail Tosun

February 2010, 93 Pages

In this study, effects of changes in various parameters of the Bayer process on the amount of “red mud”, which has many environmental drawbacks, were examined in accordance with the principles of “sustainable development”.

The production process of Seydişehir Aluminum Plant is modeled as a case study. First a steady-state mass balance calculation is carried out by incorporating sequential modular approach. Then a model of the Bayer process digester is programmed and several simulations are carried out using this model.

Results of the mass balance calculation revealed that changes in the extent of the digestion reaction, which is a function of temperature and caustic concentration, and washing efficiency may have a 2.07% decrease on the amount of red mud produced, which implies nearly 10,000 tons of decline per annum, while amount of hydrate produced is increased by 4.52%. A 7.40 % decrease on the amount of red mud produced on dry basis per kg of hydrate was found to be achievable. Optimum operating temperature for the digester was calculated as 277.3 °C.

Keywords: Alumina, Bayer Process, Sustainable Development, Red Mud, Environment Protection, Process Modeling.

ÖZ

SÜRDÜRÜLEBİLİR KALKINMA MODELİ İLE AKIM ŞEMASI OPTİMİZASYONU: ALUMİNYUM ENDÜSTRİSİ

Kurucak, Abdurrahman
Yüksek Lisans, Kimya Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Güniz Gürüz
Ortak Tez Yöneticisi: Prof. Dr. İsmail Tosun

Şubat 2010, 93 sayfa

Önerilen çalışmada çeşitli parametrelerde yapılan değişikliklerin çevre açısından birçok olumsuz etkisi olan "kırmızı çamur"un miktarına olan etkileri sürdürülebilir kalkınma kavramı çerçevesinde incelenmiştir.

Seydişehir Alüminyum Fabrikasının üretim prosesi örnek olay olarak modellenmiştir. İki farklı simülasyon yaklaşımı kullanılmıştır. İlk olarak ardışık modüler yaklaşımla yatışkın hal kütle denkliği hesabı yapılmıştır. Daha sonra erime reaktörü modeli programlanarak, bu model ile çeşitli simülasyonlar yapılmıştır.

Simülasyon sonuçları, reaksiyon sıcaklığı ve kostik konsantrasyonunun bir fonksiyonu olan erime reaksiyonunun ilerleme miktarı ve yıkama verimindeki değişikliklerin, kırmızı çamur miktarında % 2.07'ye kadar bir azalma sağlayabileceğini ortaya koymaktadır ki, bu da yıllık olarak yaklaşık 10,000 ton civarında bir azalma anlamına gelmektedir. Aynı şartlarda alüminyum hidroksit üretimi de % 4,52 oranında artmaktadır. Alüminyum hidroksitin beher kilogramı için, kırmızı çamur miktarında kuru bazda % 7,40 oranında bir azalma elde edilebileceği sonucuna varılmıştır. Erime reaktörünün optimum çalışma sıcaklığı 277,3 °C olarak hesaplanmıştır.

Anahtar sözcükler: Alumina, Bayer Prosesi, Sürdürülebilir Kalkınma, Kırmızı Çamur, Çevre Koruması, Proses Modelleme.

to *My Mother and Father*

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor, Prof. Dr. Güniz Gürüz and my co-supervisor Prof. Dr. İsmail Tosun for their enthusiastic supervision, insightful conversations during the development of this thesis, and helpful comments on the text.

My very special thanks go to my family for their great support, encouragement and unshakable faith in me.

And finally, I am forever indebted to Gülin, for her endless patience and compassion but above all, for giving me Yiğit Ali.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

NOMENCLATURE

Letters

a_i	Molal activity of component *i*
E	Number of moles of element
$\hat{f}_i$	Fugacity of component *i* in the solution
f_i^*	Fugacity of component *i* in its standard state
F	Faradays constant
G	Gibbs energy
$\bar{G}$	Partial molar Gibbs energy
H	Enthalpy
I	Ionic strength
k_{dg}	Bauxite digestion kinetic constant
K	Equilibrium constant
m_i	Molality of ion *I*
mu	Ratio of molecular weights of Al_2O_3 to Na_2O
n_i	Number of moles of component *I* in the solution
N_A	Avogadros number,
P	Pressure
P_i^{sat}	Vapor pressure of pure component *i*
R	Gas constant
RP	Ponderal ratio
s	Stream
S	Entropy
T	Temperature

V	Volume
w	Weight
x_i	Mole fraction of component *I* in liquid phase
y_i	Mole fraction of component *I* in gas phase
z_i	Charge of ion *i*,

Greek Symbols

β_{ji}	Coefficient for species by element matrix
γ_i	Molal activity coefficient of component *i*
$\gamma_\pm$	Mean molal activity coefficient
ε_0	Vacuum permittivity
ε_r	Relative permittivity (dielectric constant)
λ	Lagrangian multiplier
μ_i	Chemical potential of component *i*
μ_i^*	Chemical potential of component *i* in its standard state
ν	Stoichiometric coefficient
ξ_i	Extent of reaction *i*
ρ	Density
φ	Any thermodynamic property
ϕ	Osmotic coefficient
Ω	Number of moles of solvent in a kilogram

Superscripts

ex	Excess property
g	Gibbs energy
IS	Ideal solution property
ϕ	Osmotic coefficient
0	Standard-state

Subscripts

Eq	Equilibrium
i	Species i
g	Gibbs energy
T	Theoretical

w	Water
ϕ	Osmotic coefficient
γ	Activity coefficient
0	Standard-state

CHAPTER 1

INTRODUCTION

Bayer process is known to be the most economical means of obtaining alumina from bauxite. The process involves numerous process units which interact with mixed streams of dissolved and undissolved solids, water and steam. Bayer process consists of three main steps: digestion (extraction), precipitation, and calcination.

In Bayer process, bauxite and strong sodium hydroxide solution are added to the liquor recycled from the process. Alumina in bauxite is extracted at elevated temperatures and pressures by dissolving in concentrated sodium hydroxide solution. After digestion, remaining solids and impurities are washed in counter current decanters to recover valuable process chemicals and solid residue which is widely called “red mud”, is dumped into red mud lake. The solid-free supersaturated green liquor is cooled to precipitate gibbsite. After solid and liquid phases are separated, spent liquor is recycled for reuse while solid gibbsite is calcined in rotary kilns to produce alumina.

Sustainable development for production processes is defined as eliminating toxic processing materials and reducing quantity and toxicity of all emissions and wastes before they leave a production process, by conserving raw materials and energy (***1***).

For every ton of alumina produced, between 1 and 2 tons (dry wt.) of red mud residue is produced. This residue is highly alkaline, of relatively high surface area

($13\text{-}22\ m^2\ g^{-1}$), contains a number of chemical and mineralogical species and if not adequately controlled, can have a dramatic impact on the environment. The disposal of this residue thus constitutes a significant proportion of the overall production cost of alumina.

The methods used for decreasing the amount of effluents from the Bayer process can be grouped as (***2***):

- Pre-process modifications (increasing the quality of bauxite)
- Process modifications (process parameter optimization)
- Post-process modifications (waste disposal modifications)

Among the listed methods, process parameter optimization is the most rational method as the other methods are either very expensive because of the requirement of new installments, or impossible for an already installed plant.

The objective of this study is to model the important steps in the flow sheet of alumina production process and investigate the effects of critical parameters as extent of digestion reaction; which is a function of temperature and caustic concentration, and the washing efficiency of counter current decanters, on decreasing the amount of red mud and minimizing this amount.

With the purpose of achieving this objective, the study is separated into two parts. The first part is the steady-state modeling of the Bayer process plant and investigating the effects of process parameters by feeding different parameters to the simulation. Seydişehir aluminum plant is chosen as the model system. In the second part of the study, a model of the Bayer process digester is programmed using Mathematica® computational software program and several simulations are carried out using this model.

The concept of sustainable development is investigated in Chapter 2.

CHAPTER 2

SUSTAINABLE DEVELOPMENT

Sustainable development is the concept used to describe the economic growth model that prescribed its intended purpose as improving the level of welfare of the community by meeting short, medium, and long term needs. This concept is based on the idea: meeting today's requirements does not prevent the possibility of future generations to meet their own needs. In practice, sustainable development provides a long-term economic growth while giving importance to environment protection (***3***).

"Sustainability" concept, as we know today was first discussed in a series of meetings and used in various reports in the 1970’s (***4***). The UN Stockholm Conference, held in 1972, is known for being the first major international meeting regarding the negative effects of human activities on environment and human health.

In World Conservation Strategy of 1980, which was prepared by the World Conservation Union in cooperation with the World Wide Fund for Nature and the United Nations Environment Program, the idea of protecting environment for the self-interest of mankind was developed.

The Brundtland Commission, which was working with the support of the United Nations, released the "Our Common Future" report in 1987. The report considered underdevelopment together with environmental issues and it has been concluded that economic growth cannot be stopped, but its route needs to be changed to be compatible with the world's ecological balance (***5***).

The Brundtland Report also made the most widely accepted definition of sustainable development:

"*Sustainable development means, meeting the needs of the present, without compromising the ability of future generations to meet their own needs*."

World's interest in the concept of sustainability peaked after the United Nations Conference on Environment and Development in Rio de Janeiro in 1992. Sustainable development concept which had been defined by the Brundtland Report for the first time was approved by all 179 countries participating in the conference.

Review of the decade following the Rio de Janeiro meeting was carried out and new targets were identified in the World Summit on Sustainable Development (WSSD) in Johannesburg in 2002. During the summit, a global decision was taken by all participating states and international multilateral organizations, which included the commitment to comply with the targets of "Summit Implementation Plan" (***6***).

Summit Implementation Plan proposes the establishment of national strategies for sustainable development. Sustainable development strategies are expected to integrate economic, social, and environmental aspects and be monitored in accordance with national priorities of each country.

With the Amsterdam Treaty signed in 1997, sustainable development was also included in the European Union's objectives (***7***).

Sustainable development has the following three basic conceptual dimensions:

- Economic growth,
- Social benefits,
- Environmental protection.

Economic growth means elimination of poverty by using natural resources in the most appropriate and most efficient manner to increase society's welfare to the highest level.

Social pillar of sustainable development includes the fulfillment of basic objectives such as protecting human health, facilitating access to health and education services, security and respect for human rights.

The concept of social benefit requires the economic and social development to be evenly distributed in every segment of society.

Environmental protection, which is perhaps the most important component of sustainability, is related with protection and development of environment through the effective use of physical and biological resources.

To implement sustainable development, the complex relationship between the economic, social, and environmental objectives must be evaluated and managed properly.

In the next chapter, some information on the principles of alumina production by the Bayer process is given.

CHAPTER 3

PRODUCTION OF ALUMINA BY THE BAYER PROCESS

3.1 Introduction

In 1855 the French mining engineer Louis Le Chatelier obtained alumina from bauxite by sintering with sodium carbonate at 1200°C and leaching the sodium aluminate with water. Aluminum hydroxide was then precipitated from the sodium aluminate solution by carbon dioxide. Early use of aluminum hydroxide was chiefly as a mordant in the textile dyeing industry. The demand for pure alumina increased rapidly when it became the raw material for aluminum production upon development of the Hall-Héroult cell. The Austrian chemist Karl Joseph Bayer received German patent 43977 in August, 1888 for a new, improved process for production of aluminum hydroxide from bauxite, and the process became known as the Bayer process in his honor (***8***).

Since then, plants have been built in more than 25 countries, and the present world alumina capacity is over 40×10^6 t per annum. The important features exploited in the Bayer process are that boehmite, gibbsite, and diaspore can be dissolved in NaOH solutions under moderate hydrothermal conditions; the solubility of Al_2O_3 in NaOH is temperature dependent; most other components of the bauxite are quite inert in the process; and the silica that does dissolve subsequently forms a nearly insoluble compound. Due to these features, formation of a sodium aluminate solution, physical separation of the impurities, and precipitation of pure $Al(OH)_3$ from the cooled solution are possible.

The main features of the Bayer process have remained virtually unchanged, although the scale of operations has been enlarged considerably due to chemical engineering developments. A simple flow sheet of the Bayer process is shown in Figure 3-1 (***9***).

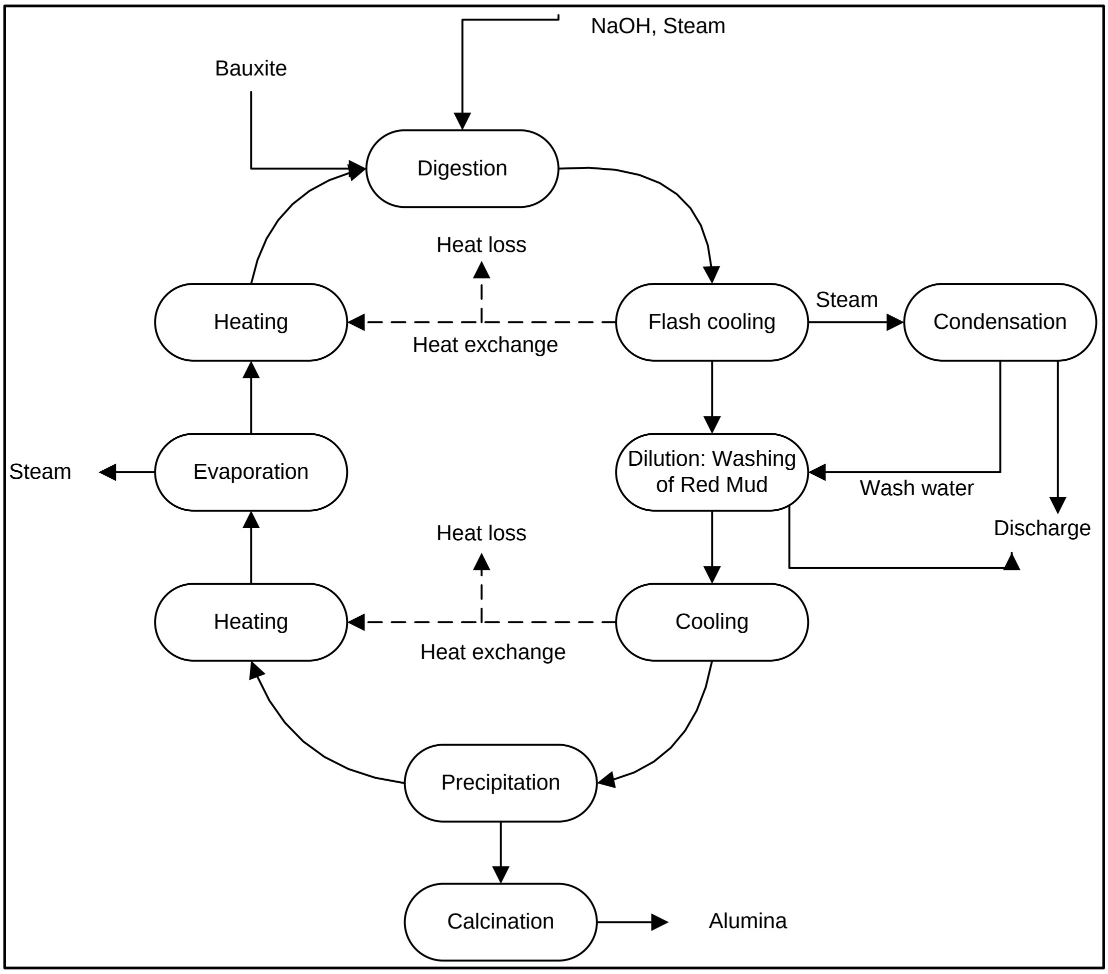

Figure 3-1 - A simplified flow sheet of an alumina plant

It is clearly seen in Figure 3-1 that the Bayer process is of cyclic nature and has quite a number of recycle streams. A brief explanation of the important stages shown is given in the proceeding section. A more detailed explanation, in particular, of the production in Seydişehir alumina plant, is given in Chapter 5.

3.2 Important Stages of the Bayer Process

The Bayer process begins with preparation of bauxite by blending for uniform composition followed by grinding. In most plants the bauxite is ground while suspended in a portion of the process solution. This slurry is mixed with the balance of the heated sodium hydroxide solution.

The prepared slurry is treated in a digester vessel with NaOH at well above atmospheric pressure up to 40 atm. Three parallel reactions proceed in the digester

$$Al_2O_3 \cdot H_2O + 2\ NaOH \leftrightarrow 2\ NaAlO_2 + 2\ H_2O \tag{3.2-1}$$

$$\begin{aligned} &SiO_2 + 2n_2\ NaAlO_2 + (n_2 + n_3 - n_1)\ H_2O \\ &\qquad \leftrightarrow n_1Na_2O \cdot n_2Al_2O_3 \cdot SiO_2 \cdot n_3H_2O + 3H_2O \end{aligned} \tag{3.2-2}$$

$$CO_2 + 2NaOH \leftrightarrow Na_2CO_3 + H_2O \tag{3.2-3}$$

The slurry leaves the digester at a temperature above its normal boiling point and is cooled by flashing off steam as the pressure is reduced in several stages. The flashed steam is used to heat the slurry and the solution entering the digester.

The bauxite residue solids are separated from the sodium aluminate solution in two steps so that the coarse fraction is processed separately from the fine. Both residue fractions are washed and discarded.

The solution, being free of solids, is cooled and seeded with fine crystals of $Al(OH)_3$. This causes the AlO_2^- ions to decompose to $Al(OH)_3$, thereby reversing the reaction that previously had taken place in the digester. Again, the heat removed in cooling the solution is used to heat a colder stream in the process. After the precipitation reaction has proceeded to the point that about half of the Al_2O_3 in the solution has been removed, the mixture of solids and solution is sent to classifiers. The fine $Al(OH)_3$ particles are returned to the process to serve as seed.

The coarse particles from classifiers are washed and calcined to Al_2O_3. Excess solution introduced in washing the product and the residue must be removed by evaporation. In some cases the solution is treated to remove both organic and inorganic impurities before the solution is recycled through the plant.(***10***)

3.3 Thermodynamics of Electrolyte Solutions

3.3.1 Excess Properties

Ideal solution assumes identical intermolecular interactions between like and unlike molecules and exactly the same volume be occupied by the molecules of each species. These assumptions are generally not valid for electrolyte solutions, even at low concentrations.

Real solutions properties deviate from ideal solution properties and excess properties are used to demonstrate deviation from ideal solution property. One may define excess property φ^{ex},

$$\varphi^{ex} = \varphi - \varphi^{IS} \tag{3.3-1}$$

Partial excess property of species *i* analogously becomes

$$\bar{\varphi}_i^{ex} = \bar{\varphi}_i - \bar{\varphi}_i^{IS} \tag{3.3-2}$$

For a single phase, multicomponent open system the differential change in the excess Gibbs energy is given by

$$dG^{ex} = V^{ex}dP - S^{ex}dT + \sum_{i=1}^{k} \bar{G}_i^{ex} dn_i \tag{3.3-3}$$

The use of identity

$$d\left(\frac{G^{ex}}{RT}\right) = \frac{1}{RT} dG^{ex} - \frac{G^{ex}}{RT^2} dT \tag{3.3-4}$$

in Eq. (3.3-3) after dividing each term by RT yields

$$d\left(\frac{G^{ex}}{RT}\right) = \frac{V^{ex}}{RT} dP - \frac{H^{ex}}{RT^2} dT + \sum_{i=1}^{k} \frac{\bar{G}_i^{ex}}{RT} dn_i \tag{3.3-5}$$

which is known as the fundamental excess-property relation.

The Gibbs energy can be regarded as a generating function to estimate other thermodynamic quantities as shown in Figure 3-2 (***11***).

3.3.2 Activity and Activity Coefficient

Activity of component *i*, in a multicomponent mixture with pure component is taken as standard state, is defined as

$$\hat{a}_i(T, P, x_i) = \frac{\hat{f}_i(T, P, x_i)}{f_i(T, P)} \tag{3.3-6}$$

In an ideal solution, making use of the Lewis-Randall rule, activity of component *i*, $\hat{a}_i^{IS}$ becomes

$$\hat{a}_i^{IS} = x_i \tag{3.3-7}$$

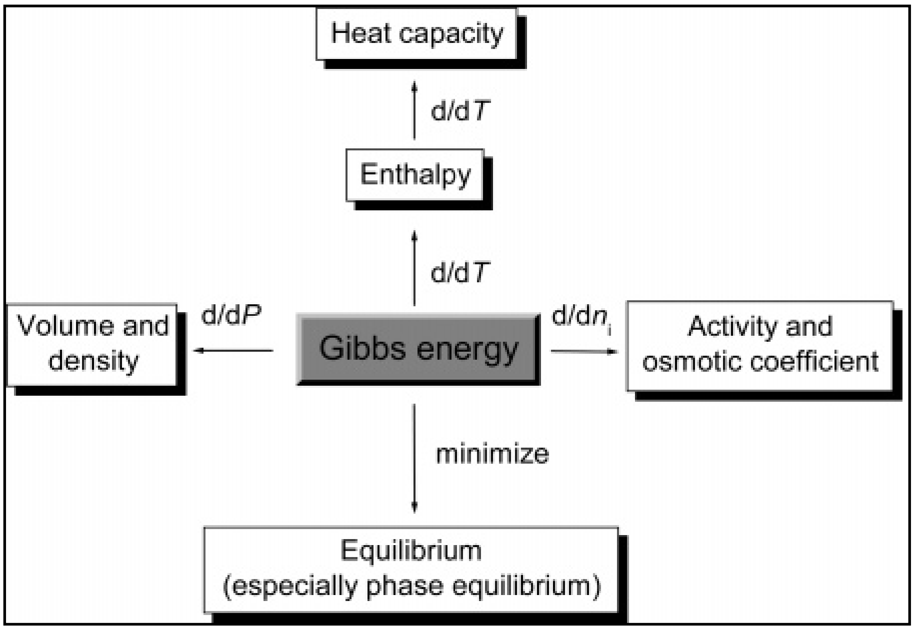

Figure 3-2 - Thermodynamic relationships between Gibbs energy and other measurable quantities.

In order to account the deviation from ideal solution behavior, activity coefficient, γ_i, is defined, i.e.,

$$\hat{a_i} = \gamma_i x_i \tag{3.3-8}$$

Comparison of Eqs. (3.3-6) and (3.3-8) yields

$$\gamma_i(T, P, x_i) = \frac{\hat{f}_i(T, P, x_i)}{x_i f_i(T, P)} \tag{3.3-9}$$

As molarity (mol/L) is temperature dependent and therefore not practical in a thermodynamic model, molality (mol (kg H_2O)$^{-1}$) and mole fraction are frequently used concentration scales in electrolyte thermodynamics (***12***).

On the molality scale, activity of component *i* is defined

$$a_i = m_i \gamma_i \tag{3.3-10}$$

In an electrolyte solution, it is not possible to measure activity coefficients of ions separately. Therefore a measurable quantity, mean molal activity coefficient is introduced and for an ionic solute *MX* it is represented by

$$\gamma_{\mp} = \gamma_M^{\nu_M} + \gamma_X^{\nu_X} \tag{3.3-11}$$

3.3.3 Derivation of Activity Coefficients from Excess Gibbs Energy Functions

Using Eq. (3.3-2), the partial molar excess Gibbs energy is expressed as

$$\bar{G}_i^{ex} = \bar{G}_i - \bar{G}_i^{IS} = RT \ln\left(\frac{\hat{f}_i}{x_i f_i}\right) \tag{3.3-12}$$

Substituting Eq. (3.3-9) into Eq. (3.3-12) gives

$$\bar{G}_i^{ex} = RT \ln \gamma_i \tag{3.3-13}$$

3.3.4 Osmotic Coefficient

As the water activity and the water activity coefficient are very close to unity in dilute aqueous solutions, to be able to report water activities without a large number of significant digits, the osmotic coefficient is commonly used.

The osmotic coefficient is represented by the symbol ϕ and is defined by

$$\phi = -\left(\frac{\Omega}{\sum m_i}\right)\ln a_w \tag{3.3-14}$$

where Ω is the number of moles of solvent in a kilogram (55.51 for water), and summation is over all solute species.

3.3.5 Debye-Hückel Limiting Law

In electrolyte solutions, strong interactions occur both among the ions and between ions and solvent through their electrical charges. Therefore such solutions significantly deviate from ideal conditions, even at low concentrations.

In 1923, the Debye-Hückel limiting law for electrolyte solutions was formulated by Peter Debye and Erich Hückel, by assuming that each ion is surrounded by an ionic atmosphere of ions of opposite charge. The Debye-Hückel limiting law describes the nonideal behavior of extremely dilute electrolyte solutions, caused by electrostatic forces (***13***).

For a salt with the stoichiometric coefficients v_i and the sum of stoichiometric coefficients v, the limiting law is expressed as

$$\ln \gamma_{\pm} = -\frac{1}{\nu}\sum_i \nu_i z_i^2 A I^{\frac{1}{2}} + \ln x_w \tag{3.3-15}$$

where

$$A = \frac{F^3}{4\pi N_A}\left[\frac{\rho}{2(\varepsilon_0\varepsilon_r RT)^3}\right]^{\frac{1}{2}} \tag{3.3-16}$$

and

$$I = \frac{1}{2}\sum_{i} m_i z_i^2 \tag{3.3-17}$$

As the Debye-Hückel limiting law in Eq. (3.3-15) is derived from Debye-Hückel theory by neglecting terms that only play a role at concentrations higher than 0.01 mol (kg H_2O)$^{-1}$, it provides an accurate representation of the limiting behavior of the activity coefficients in dilute ionic solutions which have ionic strengths less than 0.01 mol (kg H_2O)$^{-1}$.

The Debye-Hückel limiting law may be extended to include some of these terms

$$\ln \gamma_{\pm} = -\frac{1}{\nu}\sum_{i} \nu_i z_i^2 \frac{A I^{\frac{1}{2}}}{1 + b I^{\frac{1}{2}}} + \ln x_w \tag{3.3-18}$$

where b depends on the size of the involved ions, but is usually taken constant. Extended Debye-Hückel law is valid at ionic strengths up to 0.1 mol (kg H_2O)$^{-1}$.

Although being accurate for describing the activity coefficients of very dilute ionic solutions, Debye-Hückel limiting law and the extended Debye-Hückel law are not convenient for describing the properties of real electrolyte solutions. These equations are therefore often used for extrapolating the properties of electrolyte solutions to infinite dilution. Other equations such as Pitzer ion interaction model are required for describing real electrolyte solutions.

3.3.6 Pitzer Ion Interaction Model for Concentrated Electrolyte Solutions

Pitzer (1973) developed a set of equations that represents excess properties of electrolyte solutions as a function of temperature, pressure and composition in the form (***14***).

$$G^{ex} = w_w \nu m R T (1 - \phi + \ln \gamma_{\pm}) \tag{3.3-19}$$

In the Pitzer model, G^{ex} contains a Debye-Hückel term, a representation of the ionic strength dependence of the second virial coefficient B_g as well as a third virial coefficient C_g that is independent of ionic strength. For a salt *MX*;

$$\frac{G^{ex}}{w_w RT} = f_g + m^2(2\nu_M \nu_X)\left[B_g + m(\nu_M z_M)C_g\right] \tag{3.3-20}$$

For a 1-1 salt, all coefficients become unity.

Debye-Hückel term is expressed as

$$f_g = -2A_\phi I h_e^g \tag{3.3-21}$$

where

$$I = \frac{1}{2}\left(\frac{m_M z_M^2 + m_X z_X^2}{m_0}\right), m_0 = 1\frac{mol}{kg\ H_2O} \tag{3.3-22}$$

$$A_\phi = \frac{1}{3}\left(\frac{2\pi N_0 \rho_w}{1000}\right)^{\frac{1}{2}}\left(\frac{e^2}{4\pi\epsilon_0 \epsilon kT}\right)^{\frac{3}{2}} \tag{3.3-23}$$

$$h_e^g = \frac{2}{b}\ln(1 + bI^{1/2})$$

$$b = 1.2 \tag{3.3-24}$$

The second virial coefficient B_g is given by

$$B_g = \beta^{(0)} + \beta^{(1)}h_1^g + \beta^{(2)}h_2^g \tag{3.3-25}$$

where

$$h_i^g = \left(\frac{2}{\alpha_i^2 I}\right)\left(1 - \left(1 + \alpha_i I^{\frac{1}{2}}\right) exp\left(-\alpha_i I^{\frac{1}{2}}\right)\right) \qquad (3.3\text{-}26)$$

It is possible to obtain equations for ϕ and $\ln \gamma_{\pm}$, which are related to the partial molar excess Gibbs energies of solvent and solute, by appropriate differentiation.

Pitzer equation for osmotic coefficient is given by

$$\phi = 1 + |z_M z_X| f_\phi + m\left(\frac{2\nu_M \nu_X}{\nu}\right) B_\phi + m^2\left(\frac{(\nu_M \nu_X)^{\frac{3}{2}}}{\nu}\right) C_\phi \qquad (3.3\text{-}27)$$

with Debye-Hückel term

$$f_\phi = -A_\phi h_e^\phi \qquad (3.3\text{-}28)$$

$$h_e^\phi = \frac{I^{\frac{1}{2}}}{\left(1 + bI^{\frac{1}{2}}\right)} \qquad (3.3\text{-}29)$$

The second and the third virial coefficients for osmotic coefficient are given by

$$B_\phi = \beta^{(0)} + \beta^{(1)} h_1^\phi + \beta^{(2)} h_2^\phi \qquad (3.3\text{-}30)$$

$$h_i^\phi = exp(-\alpha_i I^{1/2}) \qquad (3.3\text{-}31)$$

$$C_{\phi} = w_1 + w_2 \frac{T^*}{T} + w_3 ln \frac{T}{T^*} + w_4 \frac{T}{T^*} + w_5 \frac{T^2}{T^{*2}} + \frac{w_6}{\left(\frac{T}{T^*}\right) - 210} + \frac{w_7}{647 - \frac{T}{T^*}} \tag{3.3-32}$$

$$T^* = 1\,K$$

Third virial expression for excess Gibbs energy is equals to

$$C_g = C_{\phi} h \left(2|z_M z_X|^{\frac{1}{2}}\right)^{-1} \tag{3.3-33}$$

Pitzer equation for mean activity coefficient is given by

$$\ln \gamma_{\mp} = |z_M z_X| f_{\gamma} + m\left(\frac{2\nu_M \nu_X}{\nu}\right) B_{\gamma} + m^2 \left(\frac{2(\nu_M \nu_X)^{\frac{3}{2}}}{\nu}\right) C_{\gamma} \tag{3.3-34}$$

with Debye-Hückel term

$$f_{\gamma} = -A_{\,\phi}\left(h_e^g + h_e^{\phi}\right) \tag{3.3-35}$$

The second and the third virial coefficients for activity coefficient are given by

$$B_{\gamma} = 2\beta^{(0)} + \beta^{(1)} * (h_1^g + h_1^{\phi}) + \beta^{(2)}(h_2^g + h_2^{\phi}) \tag{3.3-36}$$

$$C_{\gamma} = \frac{3}{2} C_{\phi} \tag{3.3-37}$$

$$C_{p,\phi}^{\infty} = q_1 + q_2 \frac{T}{T^*} + q_3 \left(\frac{T}{T^*}\right)^2 + q_4 \frac{T}{T^*}\left(\frac{T}{T^*} - 210\right)^3 + \frac{q_5 T}{T^*}\left(647 - \frac{T}{T^*}\right)^3 \quad (3.3\text{-}38)$$

These equations are used to model the electrolyte solution involved in the Bayer process.

CHAPTER 4

LITERATURE SURVEY

4.1 Process Modeling

Steady-state modeling of chemical processes is only a few decades old. Three methods are commonly used for simulating processes (***15***). These include sequential modular method, simultaneous modular approach and equation oriented approach.

Sequential modular approach is the oldest and most common approach among these. This method is similar to the conventional hand calculation. There are unit modules representing unit operations, which consist of necessary model equations for the unit operations. Once the flow rates and compositions, enthalpy, temperature, pressure of the input streams and parameters of the equipment are given, relevant unit module calculates the flow rates and compositions, enthalpy, temperature, pressure of output streams. Output of this module is fed as input for the next unit operation - module. Since the output streams are inputs for other unit operations, specifying all input streams equipment parameters and calculation sequence are sufficient to simulate the process.

Main problem of sequential modular approach is recycle streams. Recycle streams enter some unit modules before they are calculated as output of some other unit modules. This means iteration. One has to assume some value for the recycle stream and compare the calculated value with the assumed one. If the two values are not close enough, calculated values are replaced with the assumed ones. Procedure is repeated until a specified tolerance value is reached.

Iteration results in longer calculation times. This problem is considered by many investigators. Most simulation programs use Wegstein algorithm instead of direct substitution for faster convergence in iterations (***16***).

In simultaneous modular approach, after solving the balance equations of the system with sequential modular approach, a linear relationship is constructed between the first estimate of input and output values of each unit. This linear system is solved simultaneously together with already - linear interconnection equations. An iterative technique is followed until a specified tolerance is reached. As all equations are linear, solution time is reduced.

Advantage of equation oriented approach is that one is not obliged to supply all input parameters. It is sufficient to provide either input or output parameters that satisfy zero degrees of freedom. After specifying enough parameters to solve equations, all equations are solved simultaneously. This approach requires solution of a large set of nonlinear equations. Therefore an efficient equation solving algorithm is necessary.

The heart of any simulation program is the executive program that controls the execution sequence. There are input and output routines and possibly programs for optimization and economic analysis. The basic components of a simulator are shown in Figure 4-1 (***17***).

4.2 Bayer Process Simulation

There are a number of studies in the literature that address different approaches to simulate the Bayer process.

Sidrak (2001) reviewed simulation and control technologies in alumina refineries and the basis for the application of advanced control to various processes within a refinery (***18***).

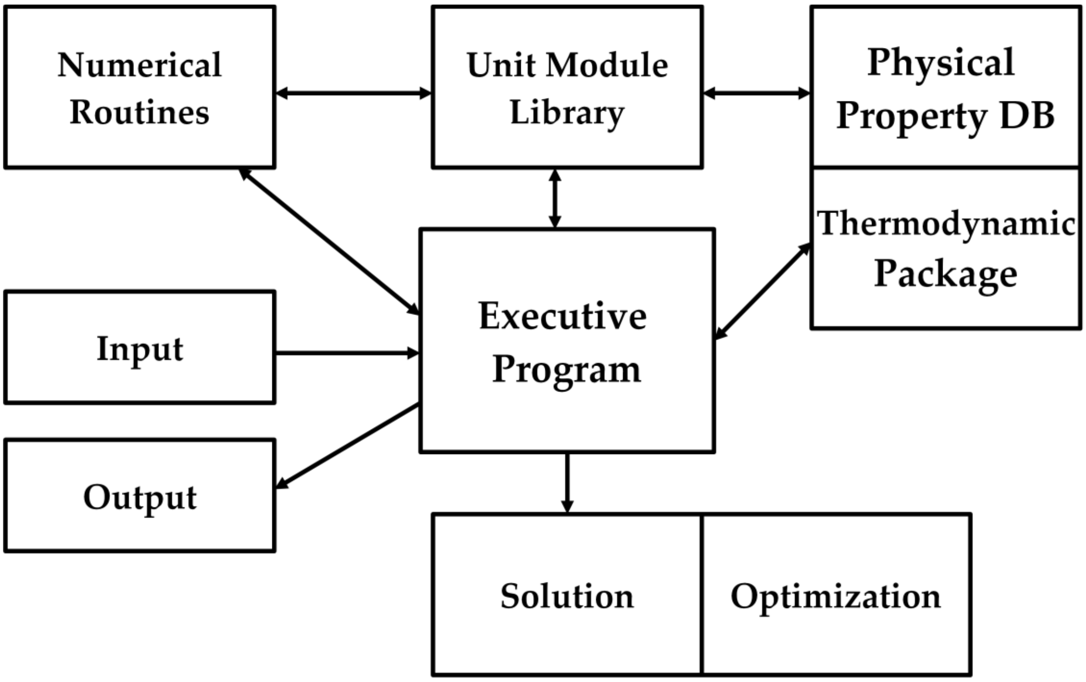

Figure 4-1 - Components of a simulation program

Clérin et al. (1988) described a general method for studying the digestibility of bauxites and proposed a practical model which allows a decrease of experiments (***19***).

Izurieta and Edwards (1980) discussed the successful application of the computer method General Energy and Material-Balance Simulator (GEMS) in hydrometallurgy (***20***).

Adjemian and Colombe (1984) developed a computer program package to obtain, for a given production rate, optimal values of the operating parameters which significantly affect the economy of the plant, taking into account all the process and technological constraints of the system (***21***).

Chinloy and Holzwarth (1984) described the general capabilities and limitations of FLOW I-a sequential modular flow sheet program designed to assist computing steady state heat and mass balances for processes involving Bayer liquor, steam and solids- and demonstrates its usage through a specific example. Convergence techniques to handle the large number of recycle streams typical of a Bayer plant were also discussed in the study (***22***).

Kiranoudis et al. (1997) presented a review of the modular steady-state simulation tool (Process Integrated Simulator for Metallurgical Applications (PRISMA)) structure, along with its operation. In the second part of their study, the steady-state process simulator developed has been used for the detailed simulation of the Bayer process flow sheet ***(23, 24).***

Königsberger (2008) applied the Gibbs energy minimiser incorporated in the thermodynamic subroutine library ChemApp and a thermodynamically consistent, ten-component Bayer liquor model to a simulation of the Bayer process (***25***).

4.3 Bayer Process Thermodynamics

Bayer process involves complex thermodynamic relations as it inquires a mixture of solid, liquid and gas phases. There are extensive number of studies on the thermodynamics of the Bayer process.

Wehrli et al. (1990) determined the dissolution reactivities of surface Al centers of several minerals and of comparable dissolved polynuclear Al complexes. It is found out that the comparison offers a second possibility for the indirect estimation of the concentration range of active surface sites and the coordination geometries of surface Al centers, which are given by the crystal lattice network, were considered to determine the reactivity in dissolution processes (***26***).

Palmer et al. (2001) measured the solubility of boehmite in a modified hydrogen-electrode concentration cell, which provided continuous in situ measurements of hydrogen ion molality over the range of ionic strengths from 0.1 to 5.0 mol kg^{-1} (NaCl) at temperatures from 100 to 290°C and also carried out a series of conventional solubility measurements in acidic solutions over the same temperature range (***27***).

Palmer et al. (2003) studied the kinetics of dissolution and the equilibrium solubility of gibbsite, boehmite, bayerite and various sodalites from 30 to 90°C as functions of caustic concentration and ionic strength in the absence and presence of triethanolamine (TEA) and other additives. The solubility constants were treated with consistent models that include appropriate experimental data taken from the literature for prediction of solubility over a wide range of state conditions. The solubility enhancing effect (kinetic and equilibrium) of TEA were also established and quantified (***28***).

Bennett et al. (2004) discussed the use of the Group Method of Data Handling (GMDH) polynomial neural network for developing a gibbsite equilibrium solubility model for Bayer process liquors and found out that the resulting predictive model appears to correctly incorporate the effects of liquor impurities and is found to offer a level of performance comparable to the most sophisticated phenomenological model presented to date (***29***).

Bénézeth et al. (2008) measured the dissolution and precipitation rates of boehmite, AlOOH, at 100.3 °C and limited precipitation kinetics of gibbsite, $Al(OH)_3$, at 50.0 °C in neutral to basic solutions at 0.1 molal ionic strength (***30***).

Li et al. obtained thermogravimetric data for the decomposition of diaspore under nonisothermal conditions by using a model-free isoconversional method to yield dependency of the activation energy on the extent of conversion for experiments (***31***).

Majzlan et al. (2000) derived enthalpies of formation of boehmite samples with variable surface area were using high-temperature oxide-melt calorimetry (***32***).

Königsberger et al. (2005) presented a Pitzer model representing the thermodynamic properties of synthetic Bayer liquor solutions, consisting of the components NaOH-$NaAl(OH)_4$-Na_2CO_3-Na_2SO_4-NaCl-NaF-$Na_2C_2O_4$ (sodium oxalate)-Na HCOO (sodium formate)-$NaCH_3COO$ (sodium acetate)-H_2O. The model is able to cvalculate, in a thermodynamically consistent manner, heat capacities, enthalpies, activity and osmotic coefficients, and densities of these solutions as well as the solubilities of gibbsite, $Al(OH)_3$, boehmite, AlOOH, sodium oxalate, sodium sulfate and other relevant solid phases in synthetic Bayer liquors over concentration and temperature ranges of industrial interest (***11***).

Königsberger and Eriksson (1999) applied the Pitzer model is to simulate various single- and multistage modifications of the industrial conversion process

$$Na_2SO_4 + 2\ KCI \rightarrow K_2SO_4 + 2\ NaCl$$

by repetitive thermochemical equilibrium calculations and observed good agreement between simulated values and operating data as the Pitzer model accurately predicts the solubility of the various salts involved in different stages of the process operating from 0 to 120°C (***33***).

Königsberger (2001) proposed a method that permits the proper extrapolation of thermodynamic quantities represented by the Pitzer equations from 25°C to other temperatures and found out that the new method, which assumes temperature independent heat capacities, was tested for the NaCl-H_2O system was very satisfactory. A new evaluation of the Na_2CO_3-H_2O system according to the CALPHAD method is also presented, solubilities for the quaternary Na_2CO_3-NaCl-NaOH-H_2O and its ternary subsystems were predicted and the limitations of these predictions and of the Pitzer model regarding extrapolations to high molalities are discussed in the study (***34***).

Königsberger (2002) studied thermodynamically consistent models that can simultaneously predict densities, heat capacities, and apparent molar enthalpies of multicomponent electrolyte mixtures together with activities of their constituents and discussed data optimization issues that included the CALPHAD (CALculation of PHAse Diagrams) method applied to electrolyte systems and the generation of robust models that extrapolate well outside the parameterization space. Recent development of software suitable for the simulation of industrial processes involving concentrated electrolyte solutions was also outlined (***35***).

Silvester and Pitzer (1977) obtained an improved equation for the dielectric constant of water over a wide range of temperature and density; which was used to calculate the Debye-Hückel parameters to 350°C for enthalpy and heat capacity as well as for osmotic and activity coefficients applied the equations to the extensive array of thermodynamic data for aqueous sodium chloride from 0 to 300°C. It is found out that the equations provide a convenient analytical representation of the thermodynamic properties of $NaCl_{(aq)}$ (***36***).

Bradley and Pitzer (1979) developed an equation for the representation of the dielectric constant of water and calculated and are presented the Debye-Hückel limiting law parameters for osmotic and activity coefficients, enthalpies, heat capacities, volumes, compressibilities and expansibilities with this equation and an equation of state for water (***37***).

Polya et al. (2001) measured molar enthalpies of dilution $\Delta_{dil}H_m$ of $Na_2CO_{3(aq)}$ from 1.45 molal to 0.008 molal at temperatures from T = 298 K to T = 523 K at the pressure p = 7 MPa, and at temperatures from T = 371 K to T = 523 K at the pressure p = 40 MPa. And facilitated a quantitative representation of apparent molar enthalpies, activity coefficients, and osmotic coefficients with the Pitzer ion-interaction treatment over the ranges of temperature, pressure, and molality of the experiments (***38***).

Simonson et al. (1989) measured the molar enthalpy of dilution $\Delta_{dil}H_m$ of $NaOH_{(aq)}$ from 6.3 to 0.008 molal at temperatures from 298 to 523 K and pressures from 7 to 40 MPa and apparent molar heat capacities from 4.0 to 0.1 molal from 323 to 523 K at 7 MPa. In the study results were correlated with available literature values as functions of temperature, pressure, and molality with the ion-interaction thermodynamic treatment. It is found out that standard heat capacities calculated from this work were consistent with literature values for the heat-capacity change on ionization of water (***39***).

Magalhães et al. (2002) measured heat capacities of aqueous solutions of Na_2SO_4 and Na_2CO_3 up to near saturation and of NaOH at 25 °C and established an experimental protocol suitable for the reliable determination of the heat capacities of concentrated electrolyte solutions by Picker calorimetry on the basis of these measurements. The heat capacities for $Na_2SO_{4(aq)}$, $Na_2CO_{3(aq)}$, and $NaOH_{(aq)}$ and literature data for the apparent molar volumes of $NaOH_{(aq)}$ at 25 °C were correlated using the Pitzer formalism in the study (***40***).

Simonson and Ryther (1989) measured densities of aqueous sodium hydroxide solutions from 0.06 to 25.2 molal at temperatures from 278 to 318 K and pressures from 0.7 to 34.4 MPa and compared the values of the apparent molar volumes calculated from the measured densities are compared with available literature results at similar experimental conditions. It is found out that standard-state partial molar volumes, expansivities, compressibilities, and heat capacities calculated from the treatment were consistent with the reaction thermodynamic properties of water and available values for $NaCl_{(aq)}$ and $HCl_{(aq)}$ (***41***).

Pabalan and Pitzer (1987) used apparent molar volumes and osmotic coefficients of NaOH solutions, calculated from critically evaluated literature data on volumes and vapor pressures to fit the Pitzer equations to yield ion-interaction parameters as a function of temperature and pressure which allow the calculation of osmotic and activity coefficients from 0-350°C, saturation pressure to about 400 bars and 0-10 molal NaOH, and volumetric properties at saturation pressure over the same temperature and concentration range. Specific volumes of NaOH solutions at saturation pressure and up to 350°C are also calculated and tabulated (***42***).

Holmes and Mesmer (1998) measured isopiestic molalities of aqueous solutions of the alkali metal hydroxides at temperatures of 383.25, 413.12, and 443.09 K. It was found that the ion-interaction model gave an excellent fit with the experimental osmotic coefficients and provided parameters for the calculation of activity coefficients and the model of Simonson et al. (***39***) is an excellent predictor of the experimental results for NaOH(aq). A simple model was also presented from which activity and osmotic coefficients (at p^{sat}) can be calculated as a function of molality for T = 273.15 K to T = 523.15 K (***43***).

Königsberger et al. (2005) measured the densities of NaOH + $NaAl(OH)_4$ solutions in isomolal series at ionic strengths over the range of $1 \leq I/mol\ kg^{-1} \leq 6$ and at total concentration ratios of $0 \leq [Al(III)]_T/[Na+]_T \leq 0.6$ in the temperature range of $50 \leq t/°C \leq 90$ and derived a Pitzer model representing the volumetric properties of hypothetical pure sodium aluminate solutions (***44***).

Wesolowski (1992) measured the solubility of gibbsite in a large number of solutions in the system Na-K-Cl-OH-$Al(OH)_4$ from 6.4 to 80°C and 0.01 to 5.0 molal ionic strength. Obtained results have been coupled with selected literature data and modeled using the Pitzer ion interaction model in the study (***45***).

Hovey et al. (1988) determined apparent molar heat capacities and volumes for aqueous solutions of $NaAl(OH)_4$ in an excess of NaOH at temperatures from 10 to 55°C and reported standard-state partial molar heat capacities and volumes for aqueous $NaAl(OH)_4$ at each temperature (***46***).

Caiani et al. (1989) measured the apparent molar heat capacities of alkaline aqueous solutions of aluminum ion in excess NaOH have been at temperatures between 50 and 250°C in the overall molality range 0.3-1.7 $mol\text{-}kg^{-1}$. Enthalpies of dilution, have also been determined at 99°C and apparent molar relative enthalpies were calculated starting from 2.16 $mol\text{-}kg^{-1}$ in the study. The thermodynamic data obtained and the corresponding quantities for aqueous NaOH previously determined have been fitted to the equations of the Pitzer ionic interaction model to obtain parameters relative to aqueous $NaAl(OH)_4$ which permit the calculation of apparent molar heat capacities and apparent molar relative enthalpies for this species over the examined range of temperatures and concentrations (***47***).

Magalhães et al. (2002) measured heat capacities of aqueous NaOH/$NaAl(OH)_4$ solutions at total stoichiometric ionic strengths of up to 6 mol kg^{-1} and with up to 60 mol% substitution of OH^- by $Al(OH)_4^-$ at 25°C. and found out that the results obtained were well described by a Pitzer model adapted from the literature, even up to high concentrations (***48***).

Park and Englezos (1999) obtained osmotic coefficient data for the aqueous solutions of NaOH-NaCl-$NaAl(OH)_4$ and found out that the experimental osmotic coefficient data were correlated well with Pitzer's model using the parameters obtained (***49***).

Li et al. (2005) discussed the feasibility of establishing a calculation model of activity coefficients of $NaAl(OH)_4$-NaOH-H_2O system according to Debye-Hückel theory, via the equilibrium solubility data of sodium aluminate solution. The results indicated that Debye-Hückel model is applicable to $NaAl(OH)_4$-NaOH-H_2O system (***50***).

Königsberger et al. (2006) presented a Pitzer model representing the thermodynamics of alkaline sodium aluminate solutions that calculates thermodynamic properties, such as heat capacities, osmotic coefficients, and densities, of these solutions as well as the solubilities of gibbsite and boehmite over temperature and concentration ranges of industrial interest (***51***).

Lwin (2000) described the use of Solver function, contained in the Microsoft Excel spreadsheet package, in computing the chemical equilibrium of known substances in either an ideal or non-ideal gas phase by the method of direct Gibbs energy minimization and found out that apart from some deficiency of Solver in applying the generalized reduced gradient method for this type of problem, the approach is simple to understand and apply and provides a direct solution procedure whereas the traditional approach often employs a high-level programming language (***52***).

4.4 Utilization of Red Mud

Studies on alternative utilization of red mud can be grouped in some subjects, such as disposal of red mud, use of red mud as catalyst, leaching precious material from red mud and use of red mud in cement and ceramics. Some literature on these subjects are listed below.

Brunori et al. (2005), Paramguru et al. (2005) and Pilurzu et al. (1999) studied the alternative techniques for the disposal of red mud ***(53-55).*** Cakici et al. (2004), Alvarez et al. (1999) and Yokoyama et al. (1993) studied use of red mud as a catalyst ***(56-58).*** Krgovic et al. (2004), Youssef et al.(2002), Rodriguez et al. (1999) and Mymrin et al. (2003) studied utilization of red mud in ceramic industry ***(59-62)***.

CHAPTER 5

SIMULATION AND OPTIMIZATION OF THE BAYER PROCESS FLOW SHEET

Chemical process simulation is a model-based representation of chemical processes and unit operations in computer code. Basic prerequisites are a thorough knowledge of chemical and physical properties of pure components and mixtures, of reactions, and of mathematical models which, in combination, allow the calculation of a process in computers.

Process simulation programs describe processes in flow diagrams where unit operations are positioned and connected by streams. The code has to solve the mass and energy balance to find a stable operating point. The goal of a process simulation is to find optimal conditions for an examined process. This is essentially an optimization problem which has to be solved in an iterative process.

Process simulation always uses models which introduce approximations and assumptions but allow the description of a property over a wide range of temperatures and pressures which might not be covered by real data. Models also allow interpolating and extrapolating within certain limits and enable the search for conditions outside the range of known properties (***63***).

In the current study, two different simulations techniques were used to simulate the production in Seydişehir alumina plant. First a steady-state mass balance calculation is carried out on a simplified flow sheet using conventional sequential modular approach which is described in detail in the next section. After that a model of the Bayer process digester is programmed and several simulations are carried out using this model.

5.1 Steady-State Simulation of Bayer Process

Sequential modular method is used in the calculation of steady-state mass balance of Seydişehir alumina plant. In this approach, after defining input streams and parameters of unit modules, other streams entering and leaving of the unit modules are calculated sequentially.

A simplified flow sheet of the alumina production process in Seydişehir Alumina Plant is shown in Figure 5.1. Production is simulated using this flow sheet. A compilation of plant data used in calculations is given in Table 5.1.

Table 5.1 A compilation of plant data used in calculations

Major streams	**Stream # in Figure 5-1**	**Flow rates (kg/h)**
Bauxite	S01	48,826
Strong liquor	S02	7,396
Wash water	S15	120,000
Red mud	S27	60,626
Hydrate ($Al(OH)_3$)	S40	36,686
Composition of bauxite		
Al_2O_3	55.63%	
SiO_2	6.86%	
CO_2	0.83%	
CaO	1.20%	
Fe_2O_3	17.39%	
Amorph	3.27%	
$H_2O_{(s)}$	12.36%	
$H_2O_{(l)}$	2.47%	

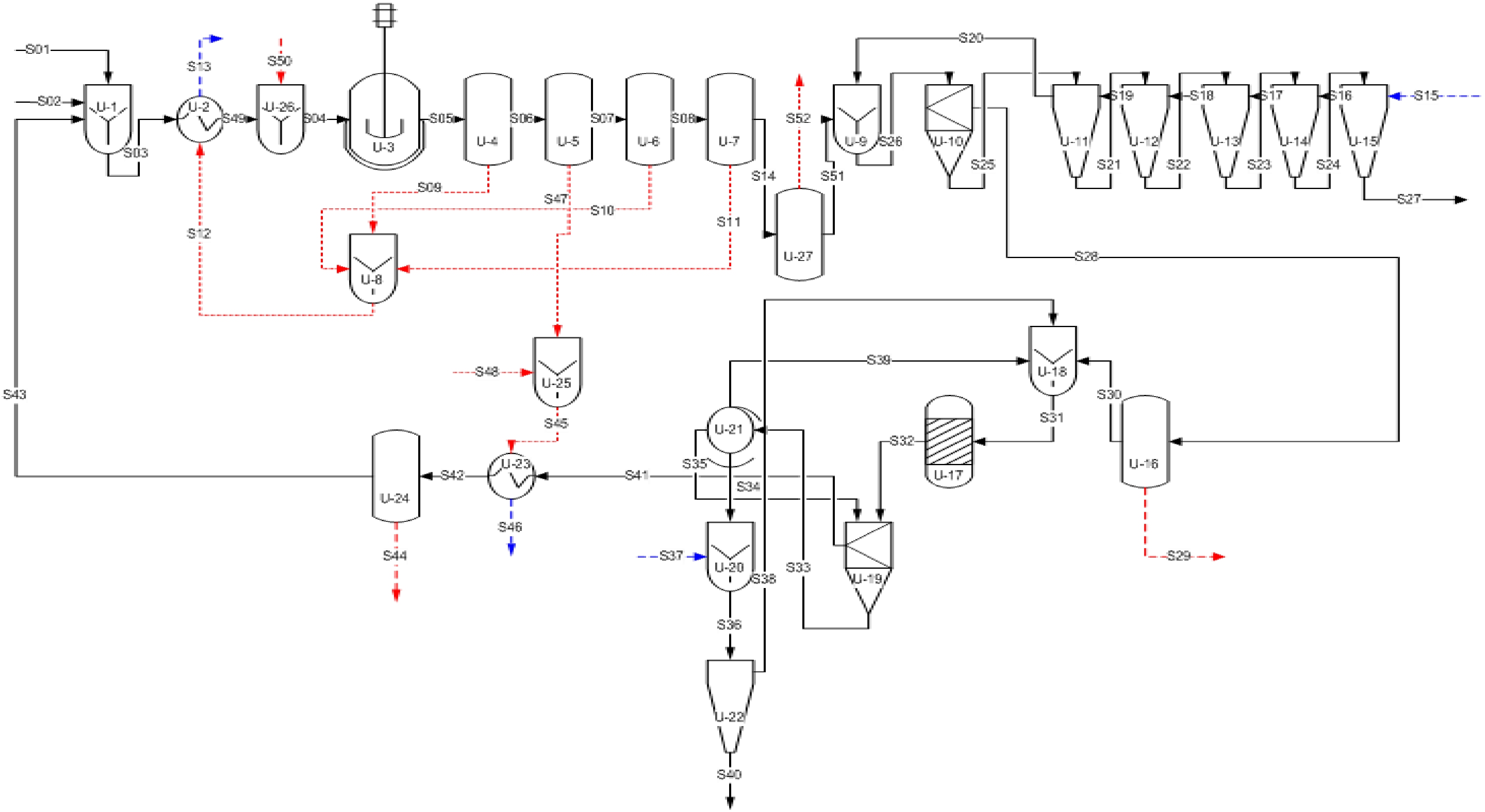

Figure 5-1- Flow sheet of Seydişehir alumina plant

5.1.1 Mathematical Models of Unit Operations

In the flow sheet there are seven mixers, seven flash drums, two reactors, five counter current decanters, two heat exchangers, three separators, one rotary filter and one cyclone. Each unit operation is represented by a module in simulation. Modules are subroutines defined to take parameters of input streams and equipment, process them and return results of output streams. A flow chart of the computer program is shown in Figure 5-2.

5.1.1.1 Mixer

Mixer is used to mix two or more streams to yield single output stream. General assumptions used in the model of the mixer are:

- Adiabatic
- No reaction takes place during mixing operation
- Perfect mixing
- Steady state

“Mixer2” is a subroutine representing a two inlet one outlet mixer, and called by

$$Mixer2(s1, s2, s3) \tag{5.1-1}$$

where s1 and s2 are vectors representing the flow rates, temperature and pressure of two inlet streams, and s3 represents the outlet stream. No other parameter is required due to the assumptions.

Material balance for component *i* around the unit is given by

$$s3_i = s1_i + s2_i \tag{5.1-2}$$

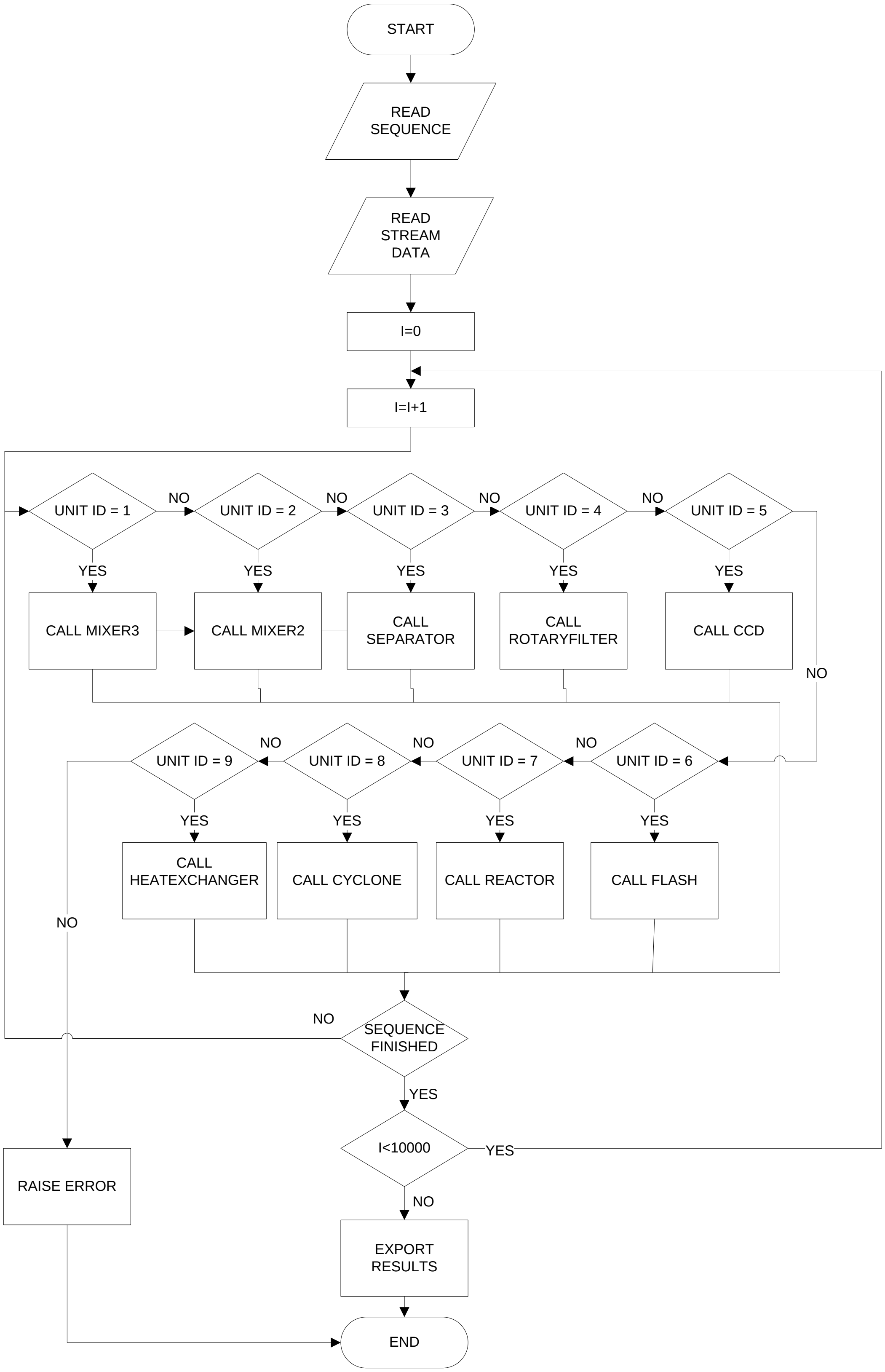

Figure 5-2 - Flow chart of mass balance program

5.1.1.2 Mixer3

Mixer3 functions exactly the same as Mixer2, except Mixer3 represents a three inlet one outlet mixer,and called by

$$Mixer3(s1, s2, s3, s4) \tag{5.1-3}$$

where s1, s2 and s3 are vectors representing the flow rates, temperature and pressure of three inlet streams and s4 represents the outlet stream.

Material balance for component *i* around the unit is given by

$$s4_i = s1_i + s2_i + s3_i \tag{5.1-4}$$

5.1.1.3 Heat Exchanger

Heat exchangers are used to heat up or cool down the streams. A heat exchanger has two inlets and two outlets. One is for the hot stream and the other is for the cold stream. Two streams do not contact directly with each other. Therefore amounts of hot and cold streams do not change at their outlets. Assumptions included are as follows:

- Adiabatic
- No reaction takes place during heat exchange
- Steady-state

“HeatExchanger” is a subroutine representing a heat exchanger, and called by

$$HeatExchanger(s1, s2, s3, s4) \tag{5.1-5}$$

where s1 and s2 are vectors representing the flow rates, temperature and pressure of two inlet streams, and s3 and s4 represent the outlet streams.

Material balance is determined by no mixing condition

$$s1 = s3$$
$$s2 = s4 \tag{5.1-6}$$

5.1.1.4 Reactor

Reactor is modeled as a stoichiometric reactor. Reactor has single inlet and single outlet streams. Components in the inlet stream are either generated or consumed due to reactions in the reactor and leave. For a stoichiometric reactor, it is necessary to know stoichiometric coefficients for all components in all reactions and extents of the reactions. Assumptions included are:

- Isothermal reactor
- Steady-state

"Reactor" is a subroutine representing an extent reactor, and called by

$$Reactor(s1, s2, extent, coeff, keycomponent) \tag{5.1-7}$$

where s1 is the vector representing the flow rate, temperature and pressure of inlet stream, and s2 represents the outlet stream. "extent" is the conversion ratio for the key component. "coeff" is the stoichiometric coefficient matrix of the components. "keycomponent" is a vector of components which conversion is based on. Multiplication of conversion with the initial moles of the key component gives the extent of that reaction.

Material balance for j^{th} component in is given by i^{th} reaction is given by

$$s2_j = extent_i \cdot coeff_{i,j} \cdot s1_{keycomponent_i} \tag{5.1-8}$$

5.1.1.5 Separator

Separator is a unit that splits single inlet stream into several streams of desired composition. Degree of separation, "*splitratio*" is a parameter of the unit and either supplied or calculated from other relationships.

"Separator" is a subroutine representing separator, and called by

$$Separator(s1, s2, s3, splitratio) \tag{5.1-9}$$

where s1 is the vector representing the flow rate, temperature and pressure of inlet stream, and s2 and s3 represent the outlet stream. "*splitratio*" is a vector for split ratio of each component. Adiabatic operation is assumed.

Material balance is given by

$$\begin{gathered} s2 \;=\; s1 \cdot splitratio \\ s3 \;=\; s1 \cdot (1 - splitratio) \end{gathered} \tag{5.1-10}$$

5.1.1.6 Cyclone

Cyclone take two inlet streams, mix them and then split them into two streams. Therefore, it would be convenient to model a cyclone as a sum of a "Mixer2" and "Separator". Assumptions of mixer and separator are valid for cyclone as well.

"Cyclone" is a subroutine representing a cyclone, and called by

$$Cyclone(s1, s2, s3, s4, splitratio) \tag{5.1-11}$$

where s1 and s2 are vectors representing the flow rates, temperature and pressure of two inlet streams and s3 and s4 represent the outlet streams. "splitratio" is required for separator.

Balances for mixer and separator are also valid for cyclone.

5.1.1.7 Flash Drum

Flash drums are some kind of separators to separate material by the concept of relative volatility. Applying heat cause the more volatile material to vaporize more than the less volatile one.

In alumina production process, flash drums are used to separate water vapor from the pulp in order to cool the pulp down. Resulting steam is also used in heating in other parts of production. As water is the only volatile species, flash drums can also be considered as boilers.

5.1.1.8 Counter Current Decanter

Counter current decanters (CCD) are devices to wash solids in order to regain valuable components. In CCD pulp and wash water is flowing counter currently to provide a better washing.

For mass balance purposes CCD is modeled just like a separator. All assumptions of separator are also valid for CCD.

“CCD” is a subroutine representing a counter current decanter, and called by

$$CCD(s1, s2, s3, s4) \tag{5.1-12}$$

where s1 is the vector representing the flow rate, temperature and pressure of inlet stream, and s2 and s3 represent the outlet streams. A split ratio which is considered as wash efficiency is coded inside the subroutine.

5.1.1.9 Rotary Filter

Rotary filter is used to separate solids from liquid. In the production of alumina in Seydişehir, it is operating like a separator that splits the stream into three. Therefore, it can be modeled as a combination of two separators. Assumptions of separator are valid for rotary filter too.

“RotaryFilter” is a subroutine representing a rotary filter, and called by

$$RotaryFilter(s1, s2, s3, s4, splitratio, splitratio2) \tag{5.1-13}$$

where s1 and s2 are vectors representing the flow rates, temperature and pressure of two inlet streams and s3 and s4 represent the outlet streams. “*splitratio*” and “*splitratio2*” are the split ratio vectors for the first and the second separators. Material balance is calculated like separator.

5.2 Simulation of Digester

Digestion is probably the most important stage of the Bayer process in terms of the amount of red mud. Therefore in the second part of this study, a detailed mathematical model of the digester is programmed using Mathematica® computational software program and effect of reaction temperature on the extents of reactions taking place in the digester is investigated.

5.2.1 Mathematical Model

Digestion is assumed to be carried out in a completely stirred tank reactor (CSTR). The principal reactions are the following

$$Al_2O_3 \cdot H_2O + 2\ NaOH \leftrightarrow 2\ NaAlO_2 + 2\ H_2O \tag{5.2-1}$$

$$\begin{aligned} &SiO_2 + 2n_2\ NaAlO_2 + (n_2 + n_3 - n_1)\ H_2O \\ &\qquad \leftrightarrow n_1Na_2O \cdot n_2Al_2O_3 \cdot SiO_2 \cdot n_3H_2O + 3H_2O \end{aligned} \tag{5.2-2}$$

$$CO_2 + 2NaOH \leftrightarrow Na_2CO_3 + H_2O \tag{5.2-3}$$

The rates of Reactions (5.2-1) through (5.2-3) are rigorously calculated through kinetic models describing the alumina dissolution, the ratio of silica oxide trapped in the argilopyritic product, and the dissolution of carbonates in the Bayer liquor, respectively. The kinetics of the alumina dissolution Reaction (5.2-1) have been simulated using the modeling expressions of Kiranoudis, et al (1997) based on the solubility equilibrium of diaspore bauxite and the dissolution kinetics. It involves the ponderal ratio (RP) of the alumina pulp, which expresses the weight ratio of alumina to soda (***24***). In particular, for a CSTR, reaction kinetics expressed on an appropriate RP basis may be formulated as follows

$$\mathrm{RP} - \mathrm{RP}_0 = k_{\mathrm{dg}}(\mathrm{RP}_T - \mathrm{RP})(\mathrm{RP}_{\mathrm{eq}} - \mathrm{RP}) \tag{5.2-4}$$

where k_{dg} is the bauxite digestion kinetic constant RP, RP_0, RP_T and $\mathrm{RP}_{\mathrm{eq}}$ are the ponderal ratio of alumina to soda leaving the digester, entering the digester, theoretical and equilibrium ponderal ratios, respectively.

The weight ratios of alumina to soda are given by the following equations

$$\mathrm{RP} = \frac{F_9/2}{(\mathrm{F0}_9 + \mathrm{F0}_{10})/2 - \xi_3 - n_1\xi_2}\mathrm{mu} \tag{5.2-5}$$

$$\mathrm{RP}_0 = \frac{\mathrm{F0}_9/2}{(\mathrm{F0}_9 + \mathrm{F0}_{10})/2 - \xi_3 - n_1\xi_2}\mathrm{mu} \tag{5.2-6}$$

$$\mathrm{RP}_T = \frac{(\mathrm{F0}_1 - F_3 n_2/n_1) + \mathrm{F0}_9/2}{(\mathrm{F0}_9 + \mathrm{F0}_{10})/2 - \xi_3 - n_1\xi_2}\mathrm{mu} \tag{5.2-7}$$

$$\mathrm{RP}_{\mathrm{eq}} = \frac{1.645}{1 + a_w\sqrt{\mathrm{K}}} \tag{5.2-8}$$

where $\mathrm{F_i}$ represents the molar flow rate of species in the order $Al_2O_{3(s)}$, $SiO_{2(s)}$, $Na_2O_{(s)}$, $CO_{2(s)}$, $CaO_{(s)}$, $Fe_2O_{3(s)}$, Amorph, $H_2O_{(s)}$, $NaAlO_{2(aq)}$, $NaOH_{(aq)}$, $Na_2CO_{3(aq)}$ and $H_2O_{(l)}$, $\mathrm{F0_i}$ represents the initial molar flow rate of species, a_w represents activity of water, and K is the equilibrium constant of Reaction (5.2-1). mu is the ratio of molecular weights of Al_2O_3 to Na_2O.

One may obtain the equilibrium constant by

$$K = e^{-\frac{\Delta_{\mathrm{r}}G^0}{RT}} \qquad (5.2\text{-}9)$$

where $\Delta_{\mathrm{r}}G^0$ is the reaction standard Gibbs energy, which is the sum of the standard Gibbs energies of the reaction products minus the sum of standard Gibbs energies of reactants. Standard state partial molal Gibbs free energies of the species in Reaction (5.2-1) is given in Table 5.2. Activity of water is calculated by Pitzer ion interaction model.

Table 5.2 - Standard state partial molal Gibbs free energies of species in Reaction (5.2-1)(*64*)

Temperature °C	$Al_2O_3.H_2O$ kJ/mol	NaOH kJ/mol	$NaAlO_2$ kJ/mol	H_2O kJ/mol
225.00	-1859.29	-427.73	-1106.84	-255.35
250.00	-1862.80	-430.58	-1110.06	-258.11
300.00	-1870.42	-437.65	-1117.04	-263.88
350.00	-1878.53	-445.05	-1125.16	-269.99

The extents of Reactions (5.2-2) and (5.2-3) are calculated by

$$\xi_2 = \mathrm{F0_2 f_{dsp}} \qquad (5.2\text{-}10)$$

$$\xi_3 = \mathrm{F0}_4 \tag{5.2-11}$$

where $\mathrm{f}_{\mathrm{dsp}}$ is the fraction of silicon oxide precipitated as desilication product.

The extent of Reaction (5.2-1) becomes

$$\xi_1 = \frac{1}{2}\left(\frac{\mathrm{RP}}{\mathrm{RP}_0} - 1\right)\mathrm{F0}_9 + n_2\xi_2 \tag{5.2-12}$$

Mass balances for species are given by

$$F_j = \mathrm{F0}_j + \sum_{i=1}^{3} \xi_i \nu_{i,j} \quad j = 1,2,\ldots,12 \tag{5.2-13}$$

where $\nu_{i,j}$, the stoichiometric matrix is given by

$$\nu_{\mathrm{i,j}} = \begin{bmatrix} -1. & 0. & 0. & 0. & 0. & 0. & 0. & -1. & 2. & -2. & 0. & 2. \\ 0.5 & 0. & 0.5 & 0. & 0. & 0. & 0. & 0.5 & -1. & 0. & 0. & -0.5 \\ 0. & 0. & 0. & -1. & 0. & 0. & 0. & 0. & 0. & -2. & 1. & 1. \end{bmatrix} \tag{5.2-14}$$

The nonlinear set of Eqs. (5.2-4 to 14) constitute the mathematical model of the bauxite digestion reactor.

CHAPTER 6

RESULTS AND DISCUSSION

In the current study, a steady-state mass balance calculation of Bayer process was carried out using the production data of Seydişehir alumina plant.

Results of overall mass balances around several subsections of the process are given below. Legends SXX denotes the streams and U-XX stands for units.

6.1 Results of Material Balance

6.1.1 Digestion

In digestion section of the Bayer process, alumina present in bauxite is extracted by dissolving in hot and concentrated sodium hydroxide solution. Flow sheet of the subsection representing the digestion section of Bayer process flow sheet is shown in Figure 6-1.

In this section, spent liquor recycled from process is mixed with ground bauxite and strong liquor (concentrated NaOH solution). The mixture is then heated and pressurized through a heat exchanger and mixer and then fed into a set of autoclaves to leach the alumina present.

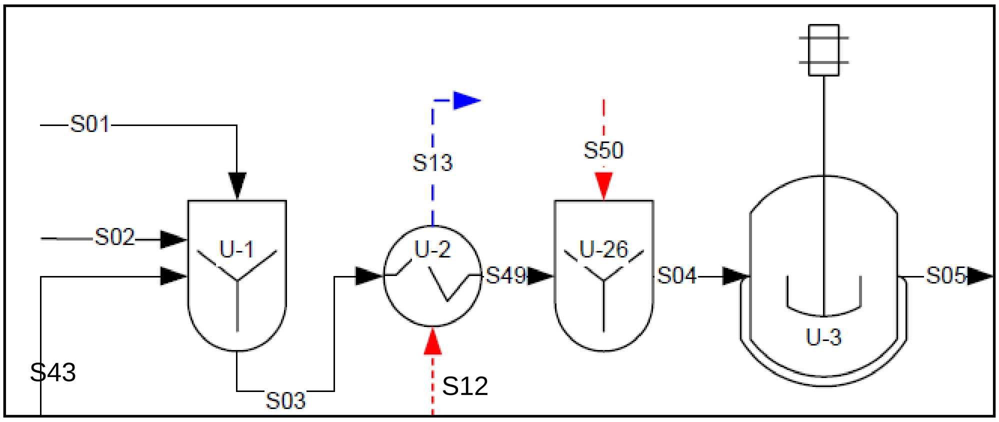

Figure 6-1 - Digestion

This process is represented by two mixers, one heat exchanger and a reactor in the current computer program. Descriptions of streams and units in Figure 6-1 and results of mass balance calculation of digestion section are shown in Table 6.1 and Table 6.2, respectively.

Table 6.1 - Descriptions of streams and units of digestion section.

STREAMS	Description	UNITS	Description
S01	Bauxite	**U-1**	Mixer
S02	Strong liquor	**U-2**	Heat Exchanger
S05	Pregnant liquor	**U-26**	Mixer
S12	Steam from flash cooling	**U-3**	Reactor
S13	Water		
S43	Spent liquor recycled from process		
S50	Steam		

Table 6.2 - Results of mass balance calculation of digestion section

	Inlet Streams (kg/h)						Outlet Streams (kg/h)		
	S01	**S02**	**S12**	**S43**	**S50**	**TOTAL**	**S05**	**S13**	**TOTAL**
Al_2O_3	27162	0	0	0	0	**27162**	3939	0	**3939**
SiO_2	3348	0	0	0	0	**3348**	3348	0	**3348**
Na_2O	0	0	0	0	0	**0**	1899	0	**1899**
CO_2	405	0	0	0	0	**405**	0	0	**0**
CaO	586	0	0	0	0	**586**	586	0	**586**
Fe_2O_3	8491	0	0	0	0	**8491**	8491	0	**8491**
Amorph	1595	0	0	0	0	**1595**	1595	0	**1595**
$H_2O_{(s)}$	6033	0	0	0	0	**6033**	1935	0	**1935**
$NaAlO_2$	0	0	0	25626	0	**25626**	62965	0	**62965**
NaOH	0	3550	0	25812	0	**29362**	7961	0	**7961**
Na_2CO_3	0	0	0	105574	0	**105574**	106549	0	**106549**
H_2O	1206	3846	31621	108793	37614	**183080**	160373	31621	**191994**
TOTAL	48826	7396	31621	265805	37614	**391262**	**359641**	**31621**	**391262**

6.1.2 Flash Cooling

Flash cooling is the section, where pregnant liquor is cooled down to temperatures near its boiling point to form a supersaturated solution. Flow sheet of the subsection representing the flash cooling section of Bayer process is shown in Figure 6-2.

In this section pregnant liquor coming from digestion is cooled by flashing through a set of flash drums. The cooled pregnant liquor, which now becomes a super saturated solution, is sent to washing section. Steam produced in this section is used for heating in the digestion section.

This process is represented by a mixer and five flash drums in current computer program. Descriptions of streams and units in Figure 6-2 and results of mass balance calculation of flash cooling section are shown in Table 6-3 and Table 6-4, respectively.

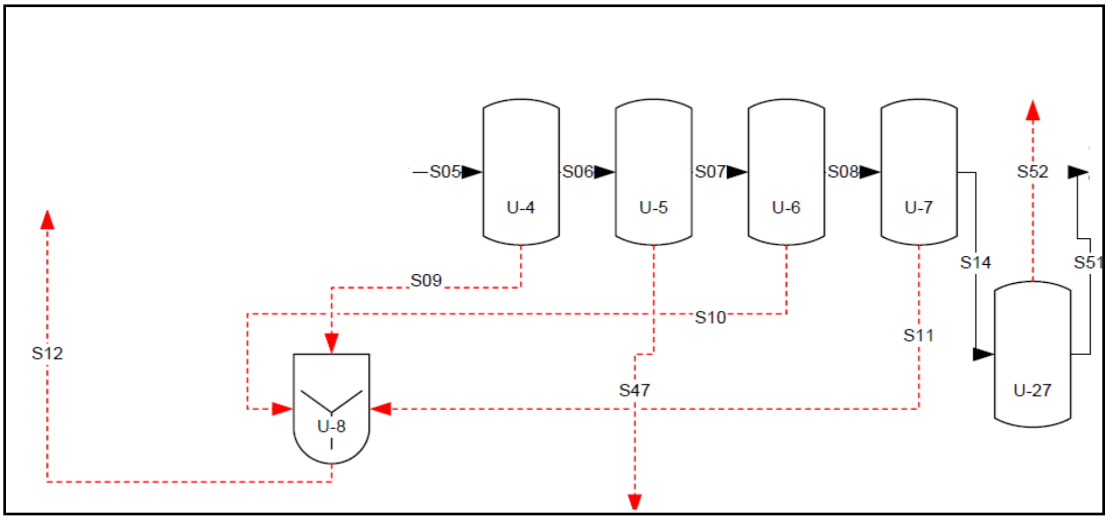

Figure 6-2 - Flash Cooling

6.1.3 Dilution: Washing of Red Mud

Solid and liquid phases in process pulp are separated and valuable process compounds are recovered by washing red mud in counter current decanters in dilution section. Flow sheet of the subsection representing the dilution section of Bayer process flow sheet is shown in Figure 6-3.

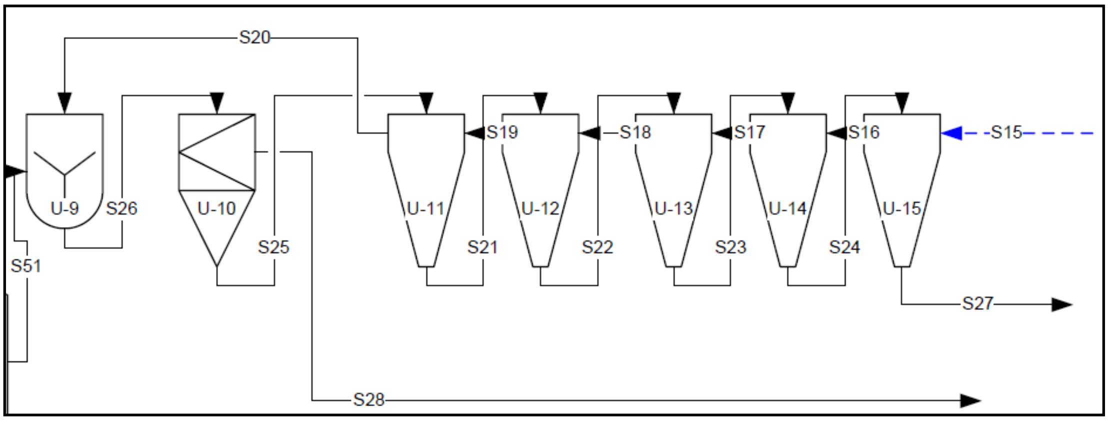

Figure 6-3 - Washing of red mud

Table 6.3 - Descriptions of streams and units of flash cooling section.

STREAMS	Description	UNITS	Description
S05	Pregnant liquor	**U-4**	Flash drum
S12	Steam to digestion	**U-5**	Flash drum
S47	Steam	**U-6**	Flash drum
S51	Cooled pregnant liquor	**U-7**	Flash drum
S52	Steam	**U-27**	Flash drum
		U-8	Mixer

Table 6.4 - Results of mass balance calculation of flash cooling section

	Inlet Streams (kg/h)		**Outlet Streams (kg/h)**				
	S05	**TOTAL**	**S12**	**S47**	**S51**	**S52**	**TOTAL**
Al_2O_3	3939	**3939**	0	0	3939	0	**3939**
SiO_2	3348	**3348**	0	0	3348	0	**3348**
Na_2O	1899	**1899**	0	0	1899	0	**1899**
CO_2	0	**0**	0	0	0	0	**0**
CaO	586	**586**	0	0	586	0	**586**
Fe_2O_3	8491	**8491**	0	0	8491	0	**8491**
Amorph	1595	**1595**	0	0	1595	0	**1595**
$H_2O_{(s)}$	1935	**1935**	0	0	1935	0	**1935**
$NaAlO_2$	62965	**62965**	0	0	62965	0	**62965**
NaOH	7961	**7961**	0	0	7961	0	**7961**
Na_2CO_3	106549	**106549**	0	0	106549	0	**106549**
H_2O	160373	**160373**	31621	10116	118517	119	**160373**
TOTAL	**359641**	**359641**	**31621**	**10116**	**317786**	**119**	**359641**

In this section process pulp from flash cooling is separated into solid and liquid phases through a hydrocyclone. Insoluble residue, red mud, is then washed in a series of counter current decanters in order to recover the valuable compounds. Pregnant liquor separated in hydro cyclone is sent to cooling and precipitation section.

This process is represented by a mixer, a hydrocyclone and five counter current decanters in the current computer program. Descriptions of streams and units in Figure 6-3 and results of mass balance calculation of dilution section are shown in Table 6.5 and Table 6-6, respectively.

6.1.4 Cooling and Precipitation

In cooling and precipitation section, pregnant liquor is further cooled in order to precipitate dissolved aluminum species as gibbsite. Flow sheet of the subsection representing the dilution section of the Bayer process flow sheet is shown in Figure 6-4.

In this section pregnant liquor separated from solids in dilution section is cooled under saturation temperature. Cooled liquor is fed into a series of reactors where precipitation reaction takes place. A hydrocyclone separates precipitated solids from spent liquor. Solid gibbsite passes through a rotary filter, and split into three streams. The coarse fraction, called primary product, is sent to calcination; the fine fractions, called secondary and tertiary seed, are recycled to be grown to product size.

This process is represented by a flash drum, two mixers, two hydrocyclones, a rotary filter and a reactor. Descriptions of streams and units in Figure 6-4 and results of mass balance calculation of dilution section are shown in Table 6.7 and Table 6.8, respectively.

Table 6.5 - Descriptions of streams and units of dilution section.

STREAMS	Description	UNITS	Description
S15	Wash water	**U-9**	Mixer
S27	Red mud	**U-10**	Hydrocyclone.
S28	Pregnant liquor to precipitation section	**U-11**	Counter current decanter
S51	Cooled pregnant liquor (including solids)	**U-12**	Counter current decanter
		U-13	Counter current decanter
		U-14	Counter current decanter
		U-15	Counter current decanter

Table 6.6 - Results of mass balance calculation of dilution section

	Inlet Streams (kg/h)			Outlet Streams (kg/h)		
	S15	**S51**	**TOTAL**	**S27**	**S28**	**TOTAL**
Al_2O_3	0	3939	**3939**	3939	0	**3939**
SiO_2	0	3348	**3348**	3348	0	**3348**
Na_2O	0	1899	**1899**	1899	0	**1899**
CO_2	0	0	**0**	0	0	**0**
CaO	0	586	**586**	586	0	**586**
Fe_2O_3	0	8491	**8491**	8491	0	**8491**
Amorph	0	1595	**1595**	1595	0	**1595**
$H_2O_{(s)}$	0	1935	**1935**	1935	0	**1935**
$NaAlO_2$	0	62965	**62965**	577	62389	**62965**
NaOH	0	7961	**7961**	73	7888	**7961**
Na_2CO_3	0	106549	**106549**	976	105574	**106549**
H_2O	120000	118517	**238517**	37209	201309	**238517**
TOTAL	**120000**	**317786**	**437786**	**60626**	**377160**	**437786**

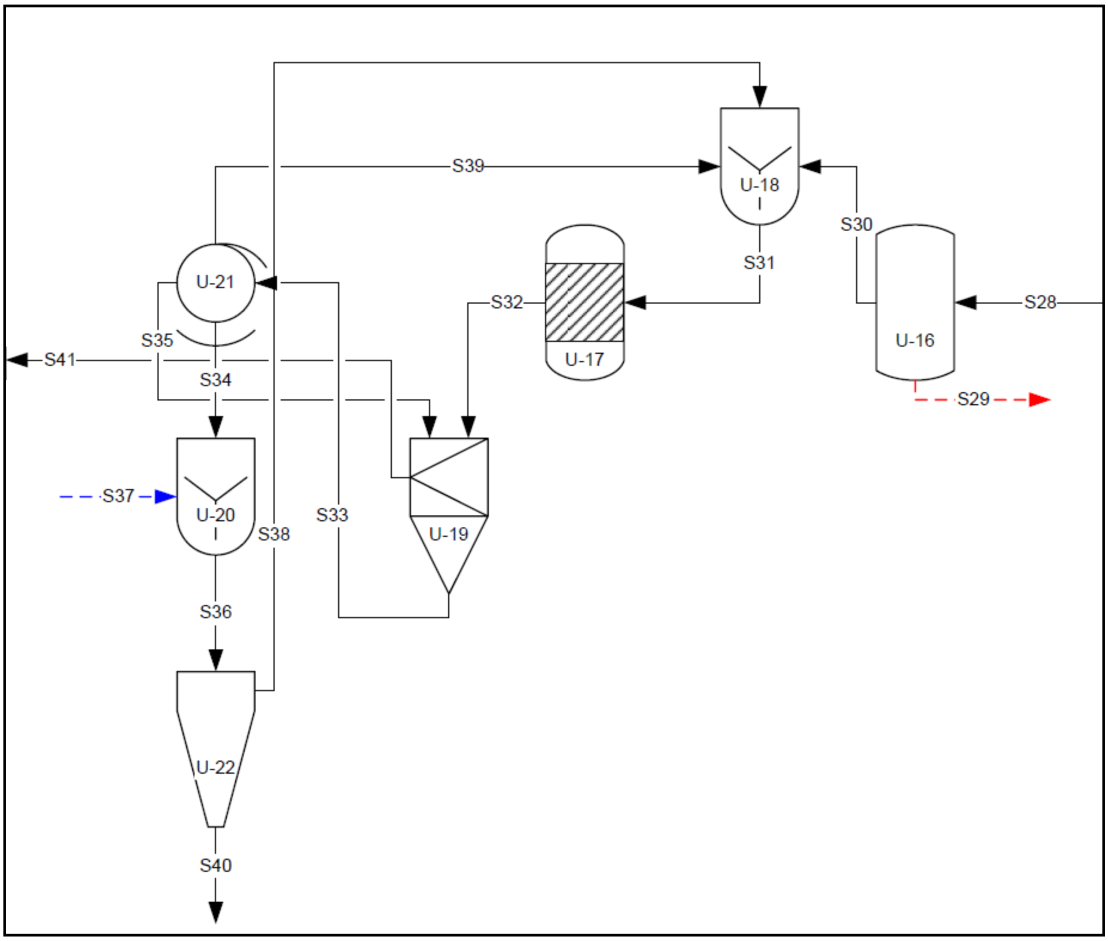

Figure 6-4 - Cooling and precipitation

Table 6.7 - Descriptions of streams and units of cooling and precipitation section.

STREAMS	Description	UNITS	Description
S28	Pregnant liquor	**U-16**	Flash drum
S29	Steam	**U-17**	Reactor
S37	Water	**U-18**	Mixer
S40	Gibbsite ($Al(OH)_3$) to calcination	**U-19**	Hydrocyclone
S41	Spent liquor	**U-20**	Mixer
		U-21	Rotary filter
		U-22	Hydrocyclone

Table 6.8 - Results of mass balance calculation of cooling and precipitation section

	Inlet Streams			Outlet Streams			
	S28	**S37**	**TOTAL**	**S29**	**S40**	**S41**	**TOTAL**
Al_2O_3	0	0	**0**	0	22861	0	**22861**
SiO_2	0	0	**0**	0	0	0	**0**
Na_2O	0	0	**0**	0	0	0	**0**
CO_2	0	0	**0**	0	0	0	**0**
CaO	0	0	**0**	0	0	0	**0**
Fe_2O_3	0	0	**0**	0	0	0	**0**
Amorph	0	0	**0**	0	0	0	**0**
$H_2O_{(s)}$	0	0	**0**	0	12103	0	**12103**
$NaAlO_2$	62389	0	**62389**	0	6	25626	**25632**
NaOH	7888	0	**7888**	0	7	25812	**25819**
Na_2CO_3	105574	0	**105574**	0	0	105574	**105574**
H_2O	201309	17051	**218359**	13689	2879	185654	**202222**
TOTAL	**377160**	**17051**	**394210**	**13689**	**37855**	**342666**	**394210**

6.1.5 Heating and Evaporation

Evaporation over and above that obtained in the cooling areas from flashed steam is usually required to maintain a water balance by removing dilution arising from residue and $Al(OH)_3$ washing, free moisture in the ore, injected steam, purge water, and uncontrolled dilutions. Evaporation also serves to concentrate impurities in the liquor making impurity removal easier. Flow sheet of the subsection representing the heating and evaporation section of Bayer process flow sheet is shown in Figure 6-5.

In this section spent liquor separated from solids in precipitation section is heated by and some water is evaporated to increase concentration. Product of this section is recycled back to digestion section.

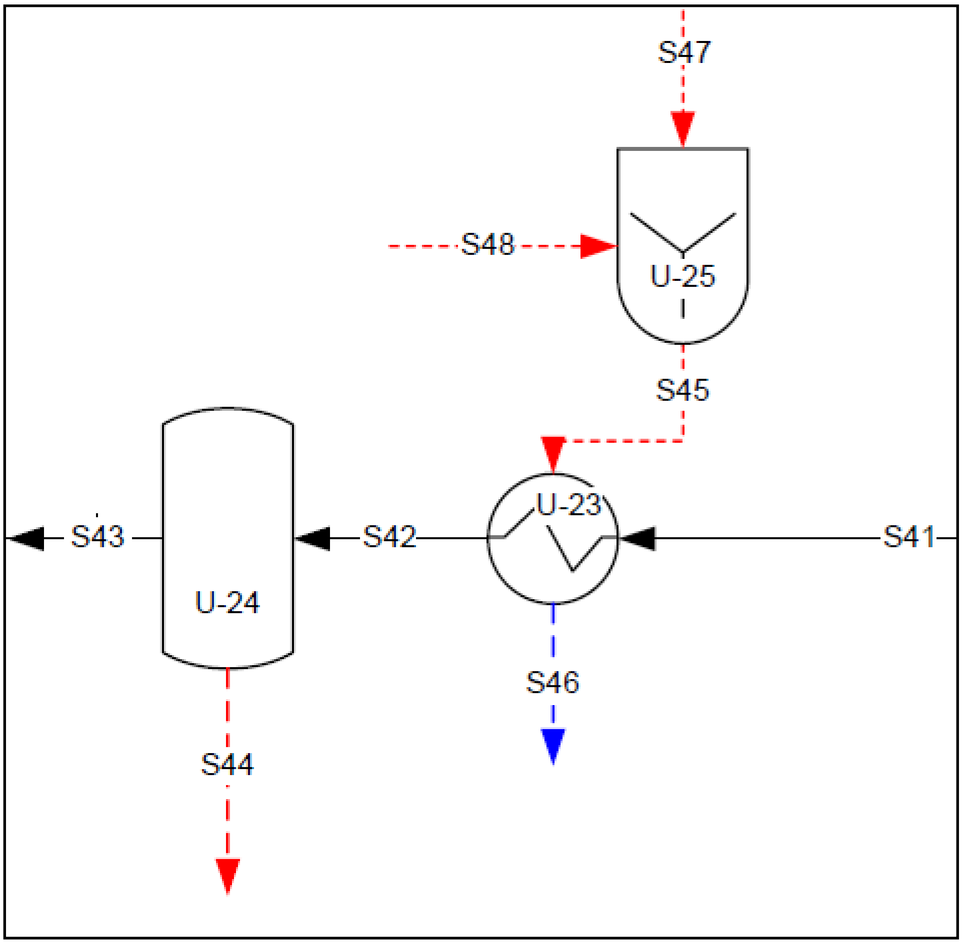

Figure 6-5 - Heating and evaporation

Table 6.9 - Descriptions of streams and units of cooling and precipitation section.

STREAMS	**Description**	**UNITS**	**Description**
S41	Spent liquor from precipitation	**U-23**	Heat exchanger
S43	Spent liquor to digestion	**U-24**	Flash drum
S44	Steam	**U-25**	Mixer
S46	Water		
S47	Steam		
S48	Steam		

Table 6.10 - Results of mass balance calculation of heating and evaporation section

	Inlet Streams (kg/h)				**Outlet Streams (kg/h)**			
	S41	**S47**	**S48**	**TOTAL**	**S43**	**S44**	**S46**	**TOTAL**
Al_2O_3	0	0	**0**	**0**	0	0	0	**0**
SiO_2	0	0	**0**	**0**	0	0	0	**0**
Na_2O	0	0	**0**	**0**	0	0	0	**0**
CO_2	0	0	**0**	**0**	0	0	0	**0**
CaO	0	0	**0**	**0**	0	0	0	**0**
Fe_2O_3	0	0	**0**	**0**	0	0	0	**0**
Amorph	0	0	**0**	**0**	0	0	0	**0**
$H_2O_{(s)}$	0	0	**0**	**0**	0	0	0	**0**
$NaAlO_2$	25626	0	**0**	**25626**	25626	0	0	**25626**
NaOH	25812	0	**0**	**25812**	25812	0	0	**25812**
Na_2CO_3	105574	0	**0**	**105574**	105574	0	0	**105574**
H_2O	185654	10116	**21854**	**217624**	108793	76861	31970	**217624**
TOTAL	**342666**	**10116**	**21854**	**374636**	**265805**	**76861**	**31970**	**374636**

This process is represented by a flash drum, a mixer and a heat exchanger. Descriptions of streams and units in Figure 6-5 and results of mass balance calculation of dilution section are shown in Table 6-9 and Table 6-10, respectively.

Good agreement between calculated values and plant data is obtained. Comparison of selected plant data with calculated values is given in Table 6.11.

Table 6.11 - Comparison of plant data with calculated values

	Plant Data (kg/h)	**Calculated Values (kg/h)**
Red Mud	60906	60626
Hydrate ($Al(OH)_3$)	36686	37855

Effect of varying process parameters on the amount of red mud is discussed in the proceeding section.

6.1.6 Effect of Parameters on the Amount of Red Mud

Effects of changes in alumina conversion in the digestion section and washing efficiency in the dilution section are investigated. Original process values for alumina conversion and washing efficiency are 0.855 and 0.95, respectively.

Figure 6-6 shows the effect of increased alumina conversion on red mud and hydrate amounts. It is found out that increasing alumina conversion from 0.855 to theoretically maximum 0.885 results in a 1.58% decrease in the amount of red mud while increasing the amount of produced hydrate ($Al(OH)_3$) by 3.24%.

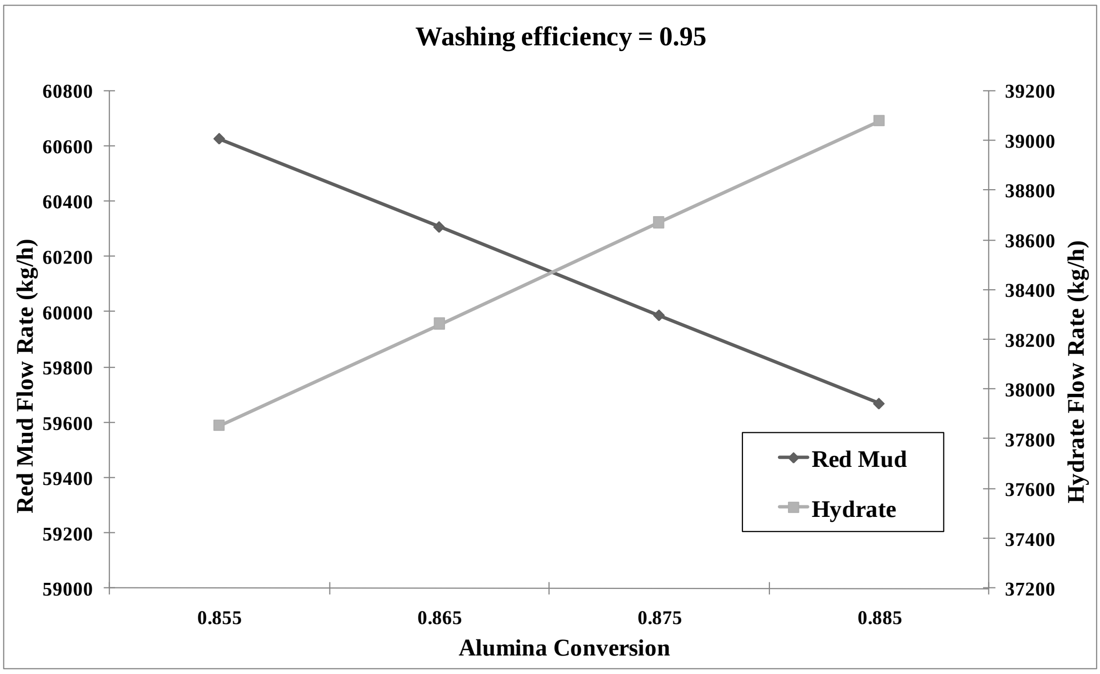

Figure 6-6 - Effect of increased alumina conversion on the amounts of red mud and hydrate

To investigate the effect of washing ratio on the amount of red mud, washing ratio is increased from 0.95 to 0.99 while keeping alumina conversion at 0.855. Results are shown in Figure 6-7.

Effect of increased washing efficiency is found out to be small with respect to increased alumina conversion: amount of red mud is reduced by 0.48% while increasing the amount of hydrate produced by 1.25%.

Increasing both alumina conversion to 0.885 and washing efficiency to 0.99, a reduction of 2.07% in the amount of red mud produced is obtained. Meanwhile, amount of hydrate produced is increased by 4.52%.

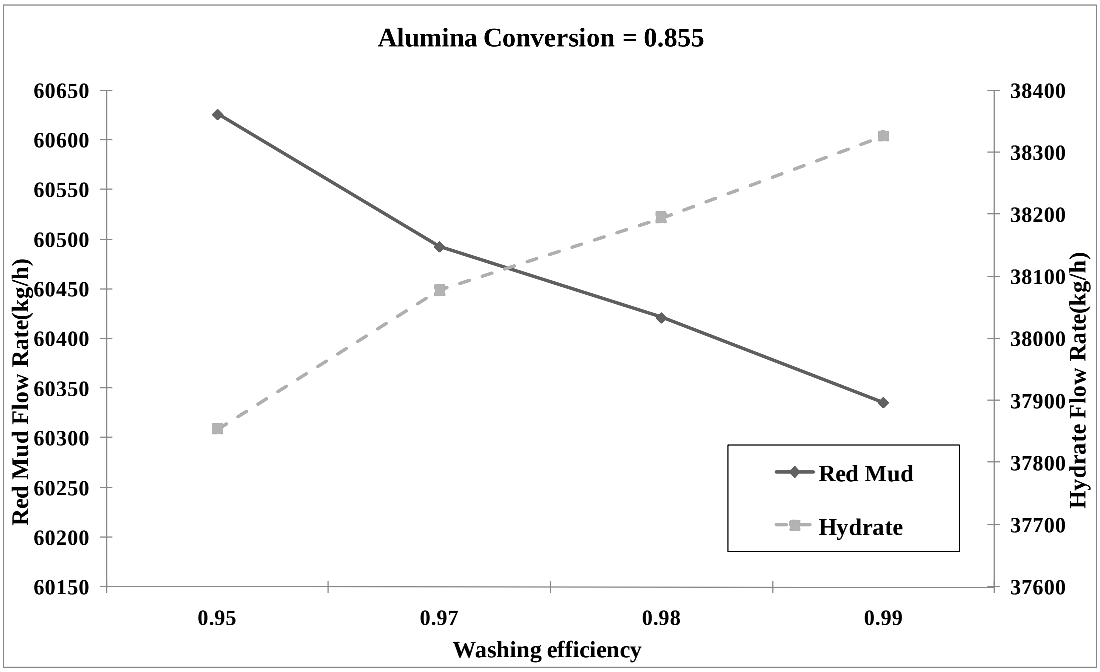

Figure 6-7 - Effect of increased washing efficiency on the amounts of red mud and hydrate

As the amount of red mud decreases and the amount of hydrate increases with increased alumina conversion, reduction in the amount of red mud becomes more significant if the amount of red mud produced on dry basis per kg of hydrate is analyzed. The amount is decreased from 0.576 $\frac{kg\ red\ mud}{kg\ hydrate}$ to 0.533 $\frac{kg\ red\ mud}{kg\ hydrate}$, which indicates a 7.40 % decrease. Results are shown in Figure 6-8.

Increasing both alumina conversion to 0.885 and washing efficiency to 0.99, a reduction of 8.54% in the amount of red mud produced on dry basis per kg of hydrate is obtained.

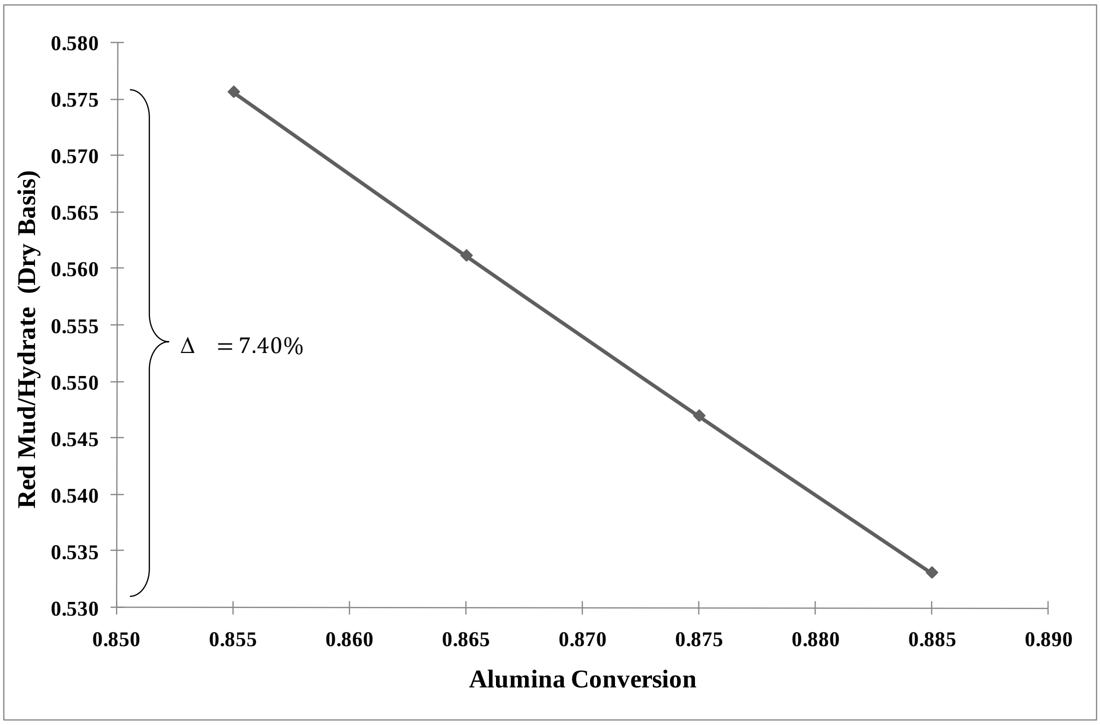

Figure 6-8 - Effect of alumina conversion on amount of red mud produced per kg of hydrate

6.2 Results of the Digester Model

In the second part of the study, a model of the Bayer process digester is programmed using Mathematica® computational software program and several simulations are carried out using this model.

Digestion reaction is taking place at 238 °C at Seydişehir alumina plant and an alumina conversion of 0.855 is achieved at that temperature. Simulation results indicate an alumina conversion of 0.852 at 238 °C, which is in very good agreement with the plant data.

Aluminum oxide extraction increases with increasing temperature. However rate of increase decreases as temperature increases. Change of extent of Reaction (5.2-1) with temperature is shown in Figure 6-9.

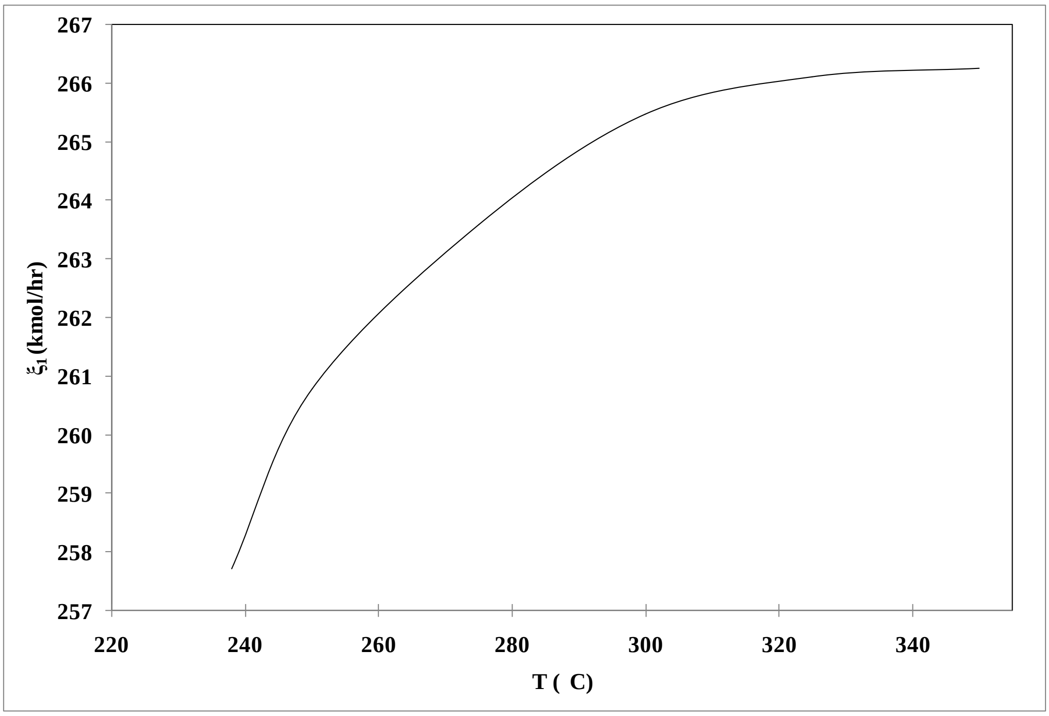

Figure 6-9 - Change of extent of Reaction (5.2-1) with temperature

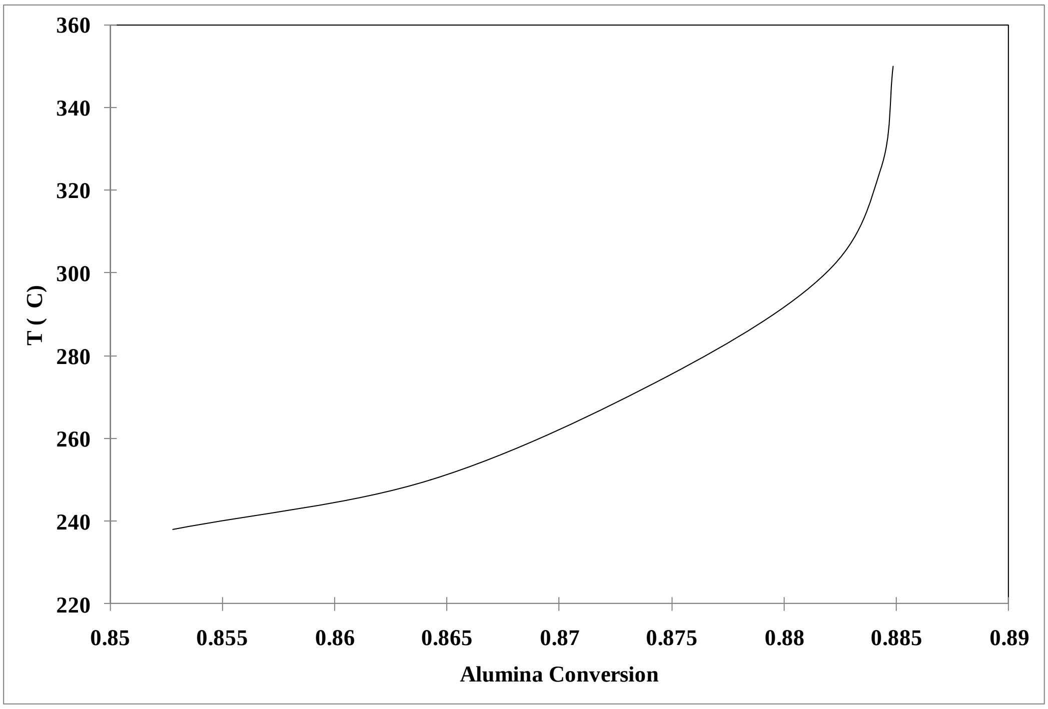

Figure 6-10 - Change of alumina conversion with temperature

The overall alumina conversion is also investigated at various temperatures. As it is found in the first part of the study alumina conversion is of crucial importance for the amount of red mud, this analysis is particularly critical. Figure 6-10 shows the change of alumina conversion with temperature. It is obvious from the figure that alumina conversion increases with increasing temperature and approaches to theoretical maximum conversion of 0.885 near 330 °C. Table 6.12 shows the change of flow rates of species leaving the digester with temperature.

Table 6.12 – Change of flow rates of species leaving the digester with temperature

Flow rates (kg/h)	**Temperature (°C)**					
	238	**250**	**275**	**300**	**325**	**350**
Al_2O_3	3998	3686	3401	3208	3143	3128
SiO_2	3348	3348	3348	3348	3348	3348
Na_2O	1899	1899	1899	1899	1899	1899
CO_2	0	0	0	0	0	0
CaO	586	586	586	586	586	586
Fe_2O_3	8491	8491	8491	8491	8491	8491
Amorph	1595	1595	1595	1595	1595	1595
$H2O_{(s)}$	1945	1890	1840	1806	1794	1792
$NaAlO_2$	62869	63371	63831	64140	64245	64268
NaOH	8008	7763	7539	7388	7337	7326
Na_2CO_3	106550	106550	106550	106550	106550	106550
H_2O	160351	160462	160562	160630	160653	160658

A trade-off analysis is carried out in order to find optimum operating temperature. The optimum operating temperature for the digester is calculated as 277.3 °C.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

The aim of this study is flow sheet optimization by the concept of sustainable development in alumina production plant. In order to achieve this objective, firstly steady-state mass balance calculation of an alumina plant was done. After that a model of the Bayer process digester was programmed and several simulations are carried out using this model.

The steady-state model of the alumina plant was assembled by linking the unit operation models sequentially, which is known as sequential modular approach. In order to link the unit operations in each stage and simulate the whole alumina plant, a python source code and subroutines were written. The steady-state model equations derived for mixer, reactor, heat exchanger, separator, counter current decanter, rotary filter, and flash drum were used in the subroutines.

In the second part, mathematical model representation of the Bayer process digester was programmed to investigate the effect of reaction temperature on the extent of digestion reaction, thus on the amount of red mud. Pitzer ion interaction model was used for calculating thermodynamic properties of electrolyte solutions involved in the Bayer process.

The applicability of alumina plant model was checked by Seydişehir alumina plant data. A good agreement was achieved between the simulated and plant data.

After simulating the whole alumina plant, different case studies were performed in order to decrease the amount of “red mud”, the insoluble residue of the Bayer process. As case studies, extent of digestion reaction, which is a function of temperature and caustic concentration, and washing ratio were optimized.

It is conducted from the case studies that the most effective process parameter was the extent of digestion reaction. It is found out that increasing alumina conversion from 0.855 to theoretically maximum 0.885 results in a 1.58% decrease in the amount of red mud while increasing the amount of hydrate produced by 3.24%. Effect of increased washing efficiency is found out to be small with respect to increased alumina conversion: amount of red mud is reduced by 0.48% while increasing the amount of hydrate produced by 1.25%. The overall process parameter optimization causes a reduction of 2.07% in the amount of red mud produced is obtained. Meanwhile, amount of hydrate produced is increased by 4.52%. Reduction of the amount of red mud becomes more significant if the amount of red mud produced on dry basis per kg of hydrate is analyzed. The amount is decreased from 0.576 $\frac{kg\ red\ mud}{kg\ hydrate}$ to 0.533 $\frac{kg\ red\ mud}{kg\ hydrate}$, which indicates a 7.40 % decrease.

Second part of the study yielded that alumina conversion increases with increasing temperature and approaches to theoretical maximum conversion of 0.885 near 330 °C. The optimum operating temperature for the digester was found to be 277.3 °C.

In conclusion, this study showed that a considerable decrease on the amount of the red mud produced is possible by overall process parameter optimization. In addition, the simulation can be used in modeling a general alumina production plant for different flow rates.

The major recommendation for future work is to develop dynamic simulation of this process on the basis of the present work and apply this to process control strategies. In addition, an economical analysis can be carried out by incorporating economical modules into simulation.

REFERENCES

1. Doğan, İ. *Flow Sheet Optimization by the Concept of Sustainable Development in Pulp and Paper Industry;* M.S. Thesis; Middle East Technical University: Ankara, 2000.

2. Hausberg, J.; Happel, U.; Meyer, F. M.; Mistry, M.; Röhrlich, M.; Koch, H.; Martens, P. N.; Schlimbach, J.; Rombach, G.; Krüger, J. Global Red Mud Reduction Potential Through Optimised Technologies And Ore Selection. *Mineral Resources Engineering (MRE)* **2000,** *9* (4), 407-420.

3. Economic Development Foundation. Sustainable Development. http://www.ikv.org.tr/sozluk2.php?ID=1237 (accessed July 19, 2008).

4. The Sustainability Report. A Brief History of Sustainable Development. http://www.sustreport.org/background/history.html (accessed Dec 5, 2009).

5. Brundtland, G. H. *Our Common Future;* Oxford University Press: Suffolk, 1987.

6. World Summit on Sustainable Development. Plan of Inplementation. http://www.johannesburgsummit.org/html/documents/summit_docs/2309_planfinal.htm (accessed June 11, 2009).

7. United Nations Development Programme. Environment and Sustainable Development. http://www.undp.org.tr/Gozlem2.aspx?WebSayfaNo=112 (accessed Sept 15, 2008).

8. Hudson, L. K.; Misra, C.; Perrotta, A. J.; Wefers, K.; Williams, F. S. *Ullman's Encyclopedia of Industrial Chemistry,* 5th ed.; American Chemical Society: New York United States of America, 1995; Vol. 120, pp 10792--10792.

9. Königsberger, E. Thermodynamic simulation of the Bayer process. *International Journal of Materials Research* **2008,** *99* (2), 197-202.

10. Sleppy, W. C. Aluminum Compounds, Survey. In *Kirk-Othmer Encyclopedia of Chemical Technology;* pp 344-360.

11. Königsberger, E.; Eriksson, G.; May, P. M.; Hefter. Comprehensive Model of Synthetic Bayer Liquors. Part 1. Overview. *Ind. Eng. Chem. Res.* **2005,** 5805-5814.

12. Pitzer, K. S. Ion Interaction Approach: Theory and Data Correlation. In Activity Coefficients in Electrolyte Solutions, 2nd. CRC Press: Boca Raton, Florida, USA, 1991; pp 75-153.

13. Debye, P.; Hückel, E. The theory of electrolytes. I. Lowering of freezing point and related phenomena. *Physikalische Zeitschrift* **1923,** *24,* 185–206.

14. Pitzer, K. S. Thermodynamics of electrolytes. I. Theoretical basis and general equations. *Journal of Physical Chemistry* **1973,** *7* (2), 268-277.

15. Kiranoudis, C.; Voros, N.; Kritikos, T.; Maroulis, Z.; Marinoskouris, D.; Papassiopi, N.; Dimitropoulou, O.; Paspaliaris, I.; Kontopoulos, A. Object-oriented simulation of hydrometallurgical processes: Part I. Requirements and implementation. *Metallurgical and Materials Transactions B* **1997,** *28,* 777-784(8).

16. Felder, R. M.; Rousseau, R. W. *Elementary principles of chemical processes;* J. Wiley & Sons: New York, 2000.

17. Raman, R. *Chemical Process Computations;* Elsevier Applied Science Publishers: London, 1985.

18. Sidrak, Y. L. Dynamic Simulation and Control of the Bayer Process. A Review. *Industrial & Engineering Chemistry Research* **2001,** *40* (4), 1146--1156.

19. Clerin, P.; Lepetit, J.; Mordini, J. Digestion of Diaspore Bauxites Utilization of Models. *Light Metals: Proceedings of Sessions, TMS Annual Meeting,* Warrendale, Pennsylvania, 1988; pp 111-117.

20. Izurieta, H.; Edwards, L. L. General energy and material-balance computer system (GEMS) applied to hydrometallurgical processes. *Journal of Metals* **1980,** *32* (12), 23-31.

21. Adjemian, A.; Colombe, P. Optimisation of a Bayer flowsheet by successive simulations automatically generated. *Light Metals: Proceedings of Sessions, AIME Annual Meeting,* Warrendale, PA, 1984; pp 3-11.

22. Chinloy, D. R.; Holzwarth, R. K. Steady state simulation of the Bayer process. *Light Metals: Proceedings of Sessions, AIME Annual Meeting,* Warrendale, PA, 1984; pp 13-26.

23. Kiranoudis, C. T.; Voros, N. G.; Kritikos, T.; Maroulis, Z. B.; Marinos-Kouris, D.; Papassiopi, N.; Dimitropoulou, O.; Paspaliaris, I.; Kontopoulos, A.; Kiranoudis, C. T.; Voros, N. G.; Kritikos, T. Object-oriented simulation of hydrometallurgical processes: Part I. requirements and implementation. *Metallurgical and Materials Transactions B: Process Metallurgy and Materials Processing Science* **1997,** *28* (5), 777-784.

24. Kiranoudis, C. T.; Voros, N. G.; Kritikos, T.; Maroulis, Z. B.; Marinos-Kouris, D.; Papassiopi, N.; Dimitropoulou, O.; Paspaliaris, I.; Kontopoulos, A. Object-oriented simulation of hydrometallurgical processes: Part II. application to the Bayer process. *Metallurgical and Materials Transactions B: Process Metallurgy and Materials Processing Science* **1997,** *28* (5), 785-793.

25. Koenigsberger, E. Thermodynamic simulation of the Bayer process. *International Journal of Materials Research* **2008,** *99,* 197-202.

26. Wehrli, B.; Wieland, E.; Furrer, G. Chemical mechanisms in the dissolution kinetics of minerals; the aspect of active sites - Dedicated to Werner Stumm for his 65th birthday. *Aquatic Sciences* **1990,** *52* (1), 3-31.

27. Palmer, D. A.; Bénézeth, P.; Wesolowski, D. J. Aqueous high-temperature solubility studies. I. The solubility of boehmite as functions of ionic strength (to 5 molal, NaCl), temperature (100-290°C), and pH as determined by in situ measurements. *Geochimica et Cosmochimica Acta* **2001,** *65* (13), 2081-2095.

28. Palmer, D. A.; Benezeth, P.; Wesolowski, D. J.; Hilic, S. Experimental Study of the Dissolution of Aluminum Phases as a Function of Temperature, Caustic Concentration and Additives. *Light Metals 2003,* Warrendale, PA, 2003.

29. Bennett, F. R.; Crew, P.; Muller, J. K. A GMDH Approach to Modelling Gibbsite Solubility in Bayer Process Liquors. *International Journal of Molecular Sciences* **2004,** *5,* 101-109.

30. Bénézeth, P.; Palmer, D. A.; Wesolowski, D. J. Dissolution/precipitation kinetics of boehmite and gibbsite: Application of a pH-relaxation technique to study near-equilibrium rates. *Geochimica et Cosmochimica Acta* **2008,** *72* (10), 2429 - 2453.

31. Li, H.; Shao, T.; Li, D.; Chen, D. Nonisothermal reaction kinetics of diasporic bauxite. *Thermochimica Acta* **2005,** *427* (1-2), 9 - 12.

32. Majzlan, J.; Navrotsky, A.; Casey, W. Surface enthalpy of boehmite. *Clays and Clay Minerals* **2000,** *48* (6), 699-707.

33. Königsberger, E.; Eriksson, G. Simulation of industrial processes involving concentrated aqueous solutions. *Journal of Solution Chemistry* **1999,** *28* (6), 721-730.

34. Königsberger, E. Prediction of Electrolyte Solubilities from Minimal Thermodynamic Information. *Monatshefte für Chemie / Chemical Monthly* **2001,** *132*, 1363--1386.

35. Königsberger, E. Solubility equilibria. From data optimization to process simulation. *Pure and Applied Chemistry* **2002,** *74* (10), 1831-1841.

36. Silvester, L. F.; Pitzer, K. S. Thermodynamics of electrolytes. 8. High-temperature properties, including enthalpy and heat capacity, with application to sodium chloride. *Journal of Physical Chemistry* **1977,** *81* (19), 1822-1828.

37. Bradley, D. J.; Pitzer, K. S. Thermodynamics of Electrolytes. 12. Dielectric Properties of Water and Debye-Hückel Parameters to 350 °C and 1 kbar. *The Journal of Physical Chemistry* **1979,** *83* (12), 1599-1603.

38. Polya, D. A.; Woolley, E. M.; Simonson, J. M.; Mesmer, R. E. The enthalpy of dilution and thermodynamics of Na2CO3(aq) and NaHCO3(aq) from T = 298 K to T = 523.15 K and pressure of 40 MPa. *Journal of Chemical Thermodynamics* **2001,** *33* (2), 205-243.

39. Simonson, J. M.; Mesmer, R. E.; Rogers, P. S. The enthalpy of dilution and apparent molar heat capacity of NaOH(aq) to 523 K and 40 MPa. *The Journal of Chemical Thermodynamics* **1989,** *21* (6), 561--584.

40. Magalhães, M. C. F.; Königsberger, E.; May, P. M.; Hefter, G. Heat capacities of concentrated aqueous solutions of sodium sulfate, sodium carbonate, and sodium hydroxide at 25 °C. *Journal of Chemical and Engineering Data* **2002,** *47* (3), 590-598.

41. Simonson, J. M.; Ryther, R. J. Volumetric properties of aqueous sodium hydroxide from 273.15 to 348.15 K. *Journal of Chemical and Engineering Data* **1989,** *34* (1), 57-63.

42. Pabalan, R. T.; Pitzer, K. S. Thermodynamics of NaOH(aq) in hydrothermal solutions. *Geochimica et Cosmochimica Acta* **1987,** *51* (4), 829-837.

43. Holmes, H. F.; Mesmer, R. E. Isopiestic molalities for aqueous solutions of the alkali metal hydroxides at elevated temperatures. *Journal of Chemical Thermodynamics* **1998,** *30* (3), 311-326.

44. Königsberger, E.; Bevis, S.; Hefter, G.; May, P. M. Comprehensive model of synthetic bayer liquors. Part 2. Densities of alkaline aluminate solutions to 90°C. *Journal of Chemical and Engineering Data* **2005,** *50* (4), 1270-1276.

45. Wesolowski, D. J. Aluminum speciation and equilibria in aqueous solution: I. The solubility of gibbsite in the system Na-K-Cl-OH-Al(OH)4 from 0 to 100°C. *Geochimica et Cosmochimica Acta* **1992,** *56* (3), 1065-1091.

46. Hovey, J. K.; Hepler, L. G.; Tremaine, P. R. Thermodynamics of aqueous aluminate ion: Standard partial molar heat capacities and volumes of Al(OH)4-(aq) from 10 to 55 °C. *Journal of Physical Chemistry* **1988,** *92* (5), 1323-1332.

47. Caiani, P.; Conti, G.; Gianni, P.; Matteoli, E. Apparent molar heat capacity and relative enthalpy of aqueous sodium hydroxoaluminate between 323 and 523 K. *Journal of Solution Chemistry* **1989,** *18* (5), 447-461.

48. Magalhães, M. C. F.; Königsberger, E.; May, P. M.; Hefter, G. Heat capacities of concentrated aqueous alkaline aluminate solutions at 25°C. *Journal of Chemical and Engineering Data* **2002,** *47* (4), 960-963.

49. Park, H.; Englezos, P. Osmotic coefficient data for NaOH-NaCl-NaAl(OH)4-H2O system measured by an isopiestic method and modeled using Pitzers model at 298.15 K. *Fluid Phase Equilibria* **1999,** *155* (2), 251-259.

50. Li, X.-b.; Wu, W.-j.; Feng, G.-t.; Liu, G.-h.; Peng, Z.-h.; Zhou, Q.-s.; Meng, Y. The Applicability of Debye-Hückel Model in NaAl(OH)4-NaOH-H2O System. *Chinese Journal of Process Engineering* **2005,** *5* (5), 525-528.

51. Königsberger, E.; May, P. M.; Hefter, G. Comprehensive model of synthetic bayer liquors. Part 3. Sodium aluminate solutions and the solubility of gibbsite and boehmite. *Monatshefte für Chemie* **2006,** *137* (9), 1139-1149.

52. Lwin, Y. Chemical Equilibrium by Gibbs Energy Minimization on Spreadsheets. *International Journal of Engineering Education* **2000,** *16* (4), 335-339.

53. Brunori, C.; Cremisini, C.; Massanisso, P.; Pinto, V.; Torricelli, L. Reuse of a treated red mud bauxite waste: studies on environmental compatibility. *Journal of Hazardous Materials* **2005,** *117,* 55-63.

54. Paramguru, R.; Rath, P.; Misra, V. Trends in red mud utilization - A review. *Mineral Processing and Extractive Metallurgy Review* **2005,** *26,* 1-29.

55. Pilurzu, S.; Cucca, L.; Tore, G.; Ullu, F. New research proposals for utilisation and disposal of bauxitic red mud from Bayer process., 1999; pp 471-480.

56. Cakici, A. I.; Yanik, J.; Ucar, S.; Karayildirim, T.; Anil, H. Utilization of red mud as catalyst in conversion of waste oil and waste plastics to fuel. *Journal of Material Cycles and Waste Management* **2004,** *6* (1), 20-26.

57. Alvarez, J.; Ordonez, S.; Rosal, R.; Sastre, H.; Diez, F. V. A new method for enhancing the performance of red mud as a hydrogenation catalyst. *Applied Catalysis, A: General* **1999,** *180* (1-2), 399-409.

58. Yokoyama, S.; Yamamoto, M.; Maekawa, Y.; Kotanigawa, T. Activities of various catalysts for hydrotreatment of Cold Lake vacuum bottoms. *Fuel* **1993,** *72* (4), 573-578.

59. Krgovic, M. M.; Blagojevic, N. Z.; Jacimovic, Z. K.; Zejak, R. Possibilities of using red mud as raw materials mixture component for production of bricks. *Research Journal of Chemistry and Environment* **2004,** *8* (4), 73-76.

60. Youssef, N. F.; Shater, M. O.; Abadir, M. F.; Ibrahim, O. A. Utilization of red mud in the manufacture of ceramic tiles. *Key Engineering Materials* **2002,** *206-213* (Pt. 3, Euro Ceramics VII), 1775-1778.

61. Rodriguez, G. A. P.; Rivera, F. G.; De Aza Pendas, S. Manufacture of ceramic materials from Bayer process red muds. *Boletin de la Sociedad Espanola de Ceramica y Vidrio* **1999,** *38* (3), 220-226.

62. Mymrin, V.; de Araujo Ponte, H.; Ferreira Lopes, O.; Vazquez Vaamonde, A. Environment-friendly method of high alkaline bauxite's Red Mud and Ferrous Slag utilization as an example of green chemistry. *Green Chemistry* **2003,** *5* (3), 357-360.

63. Process simulation. http://en.wikipedia.org/wiki/Process_simulation (accessed October 16, 2009).

64. Oelkers, E. H.; Helgeson, H. C.; Shock, E. L.; Sverjensky, D. A.; Johnson, J. W.; Pokrovskii, V. A. Summary of the apparent standard partial molal Gibbs free-energies of formation of aqueous species, minerals, and gases at pressures 1 to 5000 bars and temperatures 25 to 1000°C. *Journal of Physical and Chemical Reference Data* **1995,** *24* (4), 1401-1560.

APPENDICES

A. SOURCE CODE

A.1 MaterialBalance.py

```
#import code accelerator
try:
    import psyco
    psyco.profile()
except:
    print 'Psyco not found, ignoring it'

#import required modules
from math import *
from numpy import *
from su import *
from data import *

#Define a class representing a stream

class Stream:
    def __init__ (self,name='
',phases=[],temp=0.,pres=0.,components=[],fractions=[],flowrate=0):
        s=[]
#       self.Name=name
#       self.Phases=phases
#       self.Temp=temp
#       self.Pres=pres
#       self.Components=components
#       self.fractions=fractions
#       self.Flowrate=flowrate
#       s.append(name)
#       s.append(phases)
#       s.append(temp)
#       s.append(pres)
#       s.append(components)
#       s.append(fractions)
#       s.append(flowrate)

# Function to convert mass based flow rates to mol based flow rates
    def ConvertMol(self,stream):
        for i in xrange(stream.shape[0]-2):
```

```
            stream[i]=stream[i]/Component().MW[i]
        return stream

# Function to convert mol based flow rates to mass based flow rates

    def ConvertMass(self,stream):
        for i in xrange(stream.shape[0]-2):
            stream[i]=stream[i]*Component().MW[i]
        return stream

#Define a class representing a process unit

class ProcessUnit:
    def __init__(self,name,strin,strout):
        self.Name=name
        self.InputStreams=strin
        self.OutStreams=strout

#Define a class representing a stream

class Component:

# Initialize a component

    def __init__(self,temp=273.16):
        self.temp=temp
        self.ComponentName=componentnames
        self.MW=component_data[:,1]
        self.HeatofFormation=component_data[:,2]
        self.Cp=self.CalculateCp()
        self.h=self.CalculateEnthalpy()

# Function to calculate Cp of a component

    def CalculateCp(self):
        self.HeatCap=[]
        for i in xrange(component_data.shape[0]-1):
            temp=0
            temp+=component_data[i,3]
            temp+=component_data[i,4]*self.temp
            temp+=component_data[i,5]*self.temp**2.
            temp+=component_data[i,6]*self.temp**3.
            temp+=component_data[i,7]*self.temp**-.5
            temp+=component_data[i,8]*self.temp**-1.
            temp+=component_data[i,9]*self.temp**-2.
            temp+=component_data[i,10]*self.temp**-3.
            self.HeatCap.append(temp)
        self.HeatCap.append(Liquid(self.temp,0.101325).Cp())
        return self.HeatCap

# Function to calculate enthalpy of a component

    def CalculateEnthalpy(self):
        self.Enthalpy=[]
        p=Water(self.temp).setPressure()
        for i in xrange(component_data.shape[0]-1):
            temp=0. # component_data[i,2]*1000.
            temp+=component_data[i,3]*self.temp
```

```
            temp+=component_data[i,4]*self.temp**2./2.
            temp+=component_data[i,5]*self.temp**3./3.
            temp+=component_data[i,6]*self.temp**4./4.
            temp+=component_data[i,7]*self.temp**.5/.5
            temp+=component_data[i,8]*log(self.temp)
            temp+=component_data[i,9]*self.temp**-1./-1.
            temp+=component_data[i,10]*self.temp**-2./-2
            self.Enthalpy.append(temp)
        self.Enthalpy.append(Liquid(self.temp,p).h())
        return self.Enthalpy

#Define a class representing a unit operation model

class Model:

# Function representing a three inlet mixer

    def Mixer3(self,s1,s2,s3,s4):#ID=1
        s4=s1+s2+s3
        return s4

# Function representing a two inlet mixer

    def Mixer2(self,s1,s2,s3):#ID=2
        s3=s1+s2
        return s3

# Function representing a separator

    def Separator(self,s1,s2,s3,splitratio):#ID=3
        for i in xrange(streamdata.shape[1]-2):
            s2[i]=splitratio[i]*s1[i]
            s3[i]=(1.-splitratio[i])*s1[i]
        for i in xrange((streamdata.shape[1]-
2),(streamdata.shape[1])):
            s3[i],s2[i]=s1[i],s1[i]
        return s2,s3

# Function representing a rotary filter

    def RotaryFilter(self,s1,s2,s3,s4,splitratio,splitratio2):#ID=4
        for i in xrange(streamdata.shape[1]-2):
            s2[i]=splitratio[i]*s1[i]
            s3[i]=splitratio2[i]*s1[i]
            s4[i]=s1[i]-s2[i]-s3[i]
        return s2,s3,s4

# Function representing a counter current decanter

    def CCD(self,s1,s2,s3,s4):
        temp=s1+s2
        split=[0.,0.,0.,0.,0.,0.,0.,0.,0.95,0.95,0.95,s2[-3]/(s1[-
3]+s2[-3])]
        s3,s4=self.Separator(temp,s3,s4,split)
        return s3,s4

# Function representing a flash drum
```

```
    def Flash(self,s1,s2,s3,split):#ID=6
        s2=s1.copy()
        s2[-3]=s1[-3]*(1-split)
        s3[-3]=s1[-3]*(split)
        return s2,s3

# Function representing a stoichiometric reactor

    def Reactor(self,s1,s2,extent,coeff,keycomponent):#ID=7
        s1=Stream().ConvertMol(s1.copy())
        s2=s1.copy()
        extent=array(extent)
        coeff=array(coeff)
        keycomponent=array(keycomponent, int)
        for i in xrange(coeff.shape[0]):
            for j in xrange(streamdata.shape[1]-2):
                s2[j]+=extent[i]*s1[keycomponent[i]]*coeff[i,j]
        s2=Stream().ConvertMass(s2.copy())
        return s2

# Function representing a cyclone

    def Cyclone(self,s1,s2,s3,s4,splitratio):#ID=8
        temp=range(streamdata.shape[1])
        temp=self.Mixer2(s1,s2,temp)
        s3,s4=self.Separator(temp,s3,s4,splitratio)
        return s3,s4

# Function representing a heat exchanger

    def HeatExchanger(self,s1,s2,s3,s4):#ID=9
        s3=s1.copy()
        s4=s2.copy()
        return s3,s4


#Define a class representing the process

class Process(Model):

# Initialize the process

    def __init__(self):
        self.s=streamdata.copy()
        self.seq=sequence
        self.w=[]

# Function to simulate the process

    def Simulate(self,times):
            for order in self.seq:
                    self.current=processunitdata[order-1]

                    #Check ID's and call appropiate function for
that process unit...
                    if self.current[0]==1:
                        #print "Mixer3"
```

```
self.s[self.current[2][0]]=self.Mixer3(self.s[self.current[1][0]],se
lf.s[self.current[1][1]],self.s[self.current[1][2]],self.s[self.curr
ent[2][0]])
                    elif self.current[0]==2:
                        #print "Mixer2"

self.s[self.current[2][0]]=self.Mixer2(self.s[self.current[1][0]],se
lf.s[self.current[1][1]],self.s[self.current[2][0]])
                    elif self.current[0]==3:
                        #print "Separator"

self.s[self.current[2][0]],self.s[self.current[2][1]]=self.Separator
(self.s[self.current[1][0]],self.s[self.current[2][0]],self.s[self.c
urrent[2][1]],self.current[3])
                    elif self.current[0]==4:
                        #print "Rotary Filter"

self.s[self.current[2][0]],self.s[self.current[2][1]],self.s[self.cu
rrent[2][2]]=self.RotaryFilter(self.s[self.current[1][0]],self.s[sel
f.current[2][0]],self.s[self.current[2][1]],self.s[self.current[2][2
]],self.current[3],self.current[4])
                        #print
self.s[self.current[2][0]],self.s[self.current[2][1]],self.s[self.cu
rrent[2][2]]
                    elif self.current[0]==5:
                        #print "CCD"

self.s[self.current[2][0]],self.s[self.current[2][1]]=self.CCD(self.
s[self.current[1][0]],self.s[self.current[1][1]],self.s[self.current
[2][0]],self.s[self.current[2][1]])
                    elif self.current[0]==6:
                        #print "Flash"

self.s[self.current[2][0]],self.s[self.current[2][1]]=self.Flash(sel
f.s[self.current[1][0]],self.s[self.current[2][0]],self.s[self.curre
nt[2][1]],self.current[3])
                    elif self.current[0]==7:
                        #print "Reactor"

self.s[self.current[2][0]]=self.Reactor(self.s[self.current[1][0]],s
elf.s[self.current[2][0]],self.current[3],self.current[4],self.curre
nt[5])
                    elif self.current[0]==8:
                        #print "Cyclone"

self.s[self.current[2][0]],self.s[self.current[2][1]]=self.Cyclone(s
elf.s[self.current[1][0]],self.s[self.current[1][1]],self.s[self.cur
rent[2][0]],self.s[self.current[2][1]],self.current[3])
                    elif self.current[0]==9:
                        #print "Heat Exchanger"

self.s[self.current[2][0]],self.s[self.current[2][1]]=self.HeatExcha
nger(self.s[self.current[1][0]],self.s[self.current[1][1]],self.s[se
lf.current[2][0]],self.s[self.current[2][1]])
                    else:pass
                        #print "Error"

# Function to export simulation results to an excel file
```

```
    def Export(self):
        file=open("deneme1.xls","w")
        for i in xrange(12):
            file.write(Component().ComponentName[i]+"\t",)
        file.write("\n")
        for j in xrange(52):
            for k in xrange(12):
                file.write(str(p.s[j,k])+"\t",)
            file.write("\n")
        file.close()
        return
#-------------------------------------------------------------------
------

# Import module for timing the code execution
import time

# Set initial time
t0=time.clock()

# Initialize the process
p=Process()

# Simulate the process and loop 10000 times
p.Simulate(10000)

# Set final time
t1=time.clock()-t0

# Export results to file
p.Export()

# Print total execution time
print "execution time=",t1
```

A.2 data.py

```
# Import required modules

from scipy import *

#   Data for components:
#Al2O3,SiO2,Na2O,CO2,CaO,Fe2O3,Amorph,H2O(s),Al(OH)3,NaAlO2,NaOH,Na2
CO3,H2O

componentnames=["Al2O3","SiO2","Na2O","CO2","CaO","Fe2O3","Amorph","
H2O(s)","NaAlO2","NaOH","Na2CO3","H2O"]

component_data=[4.05,102,-1675.7,233.004,-0.0391826,9.4441E-06,0,-
2465.18,0,0,0,
2.66,44,-910.7,81.1447,0.0182834,5.4058E-06,0,-698.46,0,-180990,0,
3,62,-417.98,30.37908835,0.185869223,-0.000217596,1.00317E-
07,0,0,0,0,
```

```
3,44,-393.51,87.82,-0.0026442,0,0,-998.86,0,706410,0,
3,56,-638.659,71.6851,-0.00616496,2.23862E-06,0,-431.99,0,-255580,0,
5.3,160,-824.611,-5750.0704,1.1779302,0,0,257666.9,-
3702932,333873800,-20033214000,
3,100,-100,0,0,0,0,0,0,0,0,
3,18,-241.814,10.4381,0.025955,-4.46885E-06,0,299.19,0,-131080,0,
1,82,-1158.537,3848.972937,-13.74184785,0.014023614,-4.74772E-
06,0,0,0,0,
1,40,-470,1138.280971,-4.849048934,0.007205839,-4.46288E-06,0,0,-
24602000,0,
1,106,-1156.5806,-3874.543666,24.95051195,-0.054509005,3.70773E-
05,0,0,0,0,
1,18,-285.83,0,0,0,0,0,0,0,0]
component_data=array(component_data,float)
component_data.shape=(12,11)

# Data for streams
streamdata=[27162,3348,0,405,586,8491,1595,6033,0,0,0,1206,0.101325,
298,
0,0,0,0,0,0,0,0,0,3550,0,3846,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#2
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#5
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#10
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,120000,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#15
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#20
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#25
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#30
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#35
0,0,0,0,0,0,0,0,0,0,0,17050.5,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
```

```
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#40
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,26100,32528,1472,165361,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#45
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,21854,0.101325,444,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,
0,0,0,0,0,0,0,0,0,0,0,37614,0.101325,521,
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,#50
0,0,0,0,0,0,0,0,0,0,0,0,0.101325,298,]

streamdata=array(streamdata,float)
streamdata.shape=(52,14)


# Process Unit data

processunitdata=[[1,[0,1,42],[2],[]],
[9,[2,11],[48,12],3.8535],
[7,[3],[4],[0.97,0.805,1.],[[-1.0,0.0,0.0,0.0,0.0,0.0,0.0,-1.0,2.0,-
2.0,0.0,2.0],
[0.5,0.0,0.5,0.0,0.0,0.0,0.0,0.5,-1.0,0.0,0.0,-0.5],
[0.0,0.0,0.0,-1.0,0.0,0.0,0.0,0.0,0.0,-2.0,1.0,1.0]],[0,1,3]],#U3
[6,[4],[5,8],0.0885],
[6,[5],[6,46],0.0692],
[6,[6],[7,9],0.0579],
[6,[7],[13,10],0.0745],
[1,[8,9,10],[11],[]],
[2,[19,50],[25],[]],#U9
[3,[25],[24,27],[1.,1.,1.,1.,1.,1.,1.,1.,0.156,0.156,0.156,0.156]],
[5,[24,14],[19,26]],#U11
[0,[0],[0]],
[0,[0],[0]],
[0,[0],[0]],
[0,[0],[0]],
[6,[27],[29,28],0.068],
[7,[30],[31],[0.5557],[[0.5,0.,0.,0.,0.,0.,0.,1.5,-1.,1.,0.,-
2.]],[8]],
[1,[29,37,38],[30],[]],#U18
[8,[31,34],[32,40],[1.,1.,1.,1.,1.,1.,1.,1.,0.32,0.32,0.32,0.32]],
[2,[33,36],[35],[]],
[4,[32],[34,38,33],[0.,0.,0.,0.,0.,0.,0.,0.,0.688,0.688,0.688,0.672]
,[0.831,0.,0.,0.,0.,0.,0.,0.831,0.306,0.306,0.306,0.299]],
[3,[35],[37,39],[0.,0.,0.,0.,0.,0.,0.,0.,0.923,0.909,1.,0.853]],
[9,[40,44],[41,45],0.101325],
[6,[41],[42,43],0.414],
[2,[46,47],[44],[]],
[2,[48,49],[3],[]],#U26
[6,[13],[50,51],0.001]]

process_unit_names=[[["U1"],["MIXER"]],
[["U2"],["HEAT_EXC"]],
[["U3"],["REACTOR"]],
[["U4"],["FLASH_DRUM"]],
[["U5"],["FLASH_DRUM"]],
[["U6"],["FLASH_DRUM"]],
```

```
[["U7"],["FLASH_DRUM"]],
[["U8"],["MIXER"]],
[["U9"],["MIXER"]],
[["U10"],["SEPARATOR"]],
[["U11"],["CCD"]],
[["U12"],["CCD"]],
[["U13"],["CCD"]],
[["U14"],["CCD"]],
[["U15"],["CCD"]],
[["U16"],["FLASH_DRUM"]],
[["U17"],["REACTOR"]],
[["U18"],["MIXER"]],
[["U19"],["CYCLONE"]],
[["U20"],["MIXER"]],
[["U21"],["ROTARY_FILTER"]],
[["U22"],["SEPARATOR"]],
[["U23"],["HEAT_EXC"]],
[["U24"],["FLASH_DRUM"]],
[["U25"],["MIXER"]],
[["U26"],["MIXER"]],
[["U27"],["FLASH_DRUM"]]]

#   Unit sequence
sequence=[1,2,26,3,4,5,6,7,8,27,9,10,11,12,13,14,15,16,18,17,19,21,2
0,22,25,23,24]
```

A.3 su.py

```
# Program to calculate thermodynamic properties of water by IAPWS97
formulation

from constants import *

class Water:
    def __init__(self,T=298,p=0.101325):
        self.T=T
        self.p=p
        self.pyi=self.p/16.53
        self.tau=1386./self.T
        self.pyi2=self.p/1.
        self.tau2=540./self.T
        self.R=0.461526
    def setPressure(self):
        nu=self.T+ca4[8]/(self.T-ca4[9])
        A=nu**2.+ca4[0]*nu+ca4[1]
        B=ca4[2]*nu**2.+ca4[3]*nu+ca4[4]
        C=ca4[5]*nu**2.+ca4[6]*nu+ca4[7]
        ps=(2*C/(-1.*B+(B**2.-4*A*C)**.5))**4.
        return ps

class Liquid(Water):
    def gamma(self):
        gp=0.
        for i in range(34):
            gp+=ca1[i,2]*(7.1-self.pyi)**(ca1[i,0])*(self.tau-
```

```
1.222)**ca1[i,1]
        return gp

    def gammapi(self):
        gp=0.
        for i in range(34):
            gp+=(-1.*ca1[i,2]*ca1[i,0]*(7.1-self.pyi)**(ca1[i,0]-
1.))*((self.tau-1.222)**ca1[i,1])
        return gp

    def gammapipi(self):
        gp=0.
        for i in range(34):
            gp+=(ca1[i,2]*ca1[i,0]*(ca1[i,0]-1.)*(7.1-
self.pyi)**(ca1[i,0]-2.))*((self.tau-1.222)**ca1[i,1])
        return gp

    def gammatetha(self):
        gt=0.
        for i in range(34):
            gt+=(ca1[i,2]*(7.1-
self.pyi)**(ca1[i,0]))*(ca1[i,1]*(self.tau-1.222)**(ca1[i,1]-1.))
        return gt

    def gammatethatetha(self):
        gt=0.
        for i in range(34):
            gt+=(ca1[i,2]*(7.1-
self.pyi)**(ca1[i,0]))*(ca1[i,1]*(ca1[i,1]-1.)*(self.tau-
1.222)**(ca1[i,1]-2.))
        return gt

    def gammapitetha(self):
        gpt=0.
        for i in range(34):
            gpt+=(-1.*ca1[i,2]*ca1[i,0]*(7.1-self.pyi)**(ca1[i,0]-
1.))*(ca1[i,1]*(self.tau-1.222)**(ca1[i,1]-1.))
        return gpt

    def V(self):
        return self.gammapi()*self.pyi*self.R*self.T/self.p/1000.
    def ro(self):
        return 1./self.V()
    def h(self):
        return self.tau*self.gammatetha()*self.R*self.T
    def Cp(self):
        return -1.*self.tau**2.*self.gammatethatetha()*self.R
    def G(self):
        return self.gamma()*self.R*self.T

class Vapor(Water):
    def gammataui(self):
        gt=0.
        for i in range(9):
            gt+=ca2i[i,1]*ca2i[i,0]*self.tau2**(ca2i[i,0]-1.)
        return gt
    def gammataur(self):
        gt=0.
```

```
        for i in range(43):

gt+=ca2r[i,2]*ca2r[i,1]*self.pyi2**ca2r[i,0]*(self.tau2-
.5)**(ca2r[i,1]-1.)
        return gt
    def h(self):
        return
self.R*self.T*self.tau2*(self.gammataui()+self.gammataur())*18
```

A.4 constants.py

```
#Constants for IAPWS 97 formulation for thermodynamic properties of
steam and water

from numpy import *
constRegionOne=[0,-2,1.46329712131670E-01,
0,-1,-8.45481871691140E-01,
0,0,-3.75636036720400E+00,
0,1,3.38551691683850E+00,
0,2,-9.57919633878720E-01,
0,3,1.57720385132280E-01,
0,4,-1.66164171995010E-02,
0,5,8.12146299835680E-04,
1,-9,2.83190801238040E-04,
1,-7,-6.07063015658740E-04,
1,-1,-1.89900682184190E-02,
1,0,-3.25297487705050E-02,
1,1,-2.18417171754140E-02,
1,3,-5.28383579699300E-05,
2,-3,-4.71843210732670E-04,
2,0,-3.00017807930260E-04,
2,1,4.76613939069870E-05,
2,3,-4.41418453308460E-06,
2,17,-7.26949962975940E-16,
3,-4,-3.16796448450540E-05,
3,0,-2.82707979853120E-06,
3,6,-8.52051281201030E-10,
4,-5,-2.24252819080000E-06,
4,-2,-6.51712228956010E-07,
4,10,-1.43417299379240E-13,
5,-8,-4.05169968601170E-07,
8,-11,-1.27343017416410E-09,
8,-6,-1.74248712306340E-10,
21,-29,-6.87621312955310E-19,
23,-31,1.44783078285210E-20,
29,-38,2.63357816627950E-23,
30,-39,-1.19476226400710E-23,
31,-40,1.82280945814040E-24,
32,-41,-9.35370872924580E-26]
constRegionTwoIdeal=array([[ 0.00000000e+000, -9.69276865e+000],
       [ 1.00000000e+000,  1.00866560e+001],
       [-5.00000000e+000, -5.60879113e-003],
       [-4.00000000e+000,  7.14527381e-002],
       [-3.00000000e+000, -4.07104982e-001],
       [-2.00000000e+000,  1.42408192e+000],
       [-1.00000000e+000, -4.38395113e+000],
```

```
       [ 2.00000000e+000, -2.84086325e-001],
       [ 3.00000000e+000,  2.12684638e-002]])
constRegionTwoResidual=array([[ 1.00000000e+000,  0.00000000e+000,
-1.77317425e-003],
       [ 1.00000000e+000,  1.00000000e+000, -1.78348623e-002],
       [ 1.00000000e+000,  2.00000000e+000, -4.59960137e-002],
       [ 1.00000000e+000,  3.00000000e+000, -5.75812591e-002],
       [ 1.00000000e+000,  6.00000000e+000, -5.03252787e-002],
       [ 2.00000000e+000,  1.00000000e+000, -3.30326417e-005],
       [ 2.00000000e+000,  2.00000000e+000, -1.89489875e-004],
       [ 2.00000000e+000,  4.00000000e+000, -3.93927772e-003],
       [ 2.00000000e+000,  7.00000000e+000, -4.37972957e-002],
       [ 2.00000000e+000,  3.60000000e+001, -2.66745479e-005],
       [ 3.00000000e+000,  0.00000000e+000,  2.04817377e-008],
       [ 3.00000000e+000,  1.00000000e+000,  4.38706673e-007],
       [ 3.00000000e+000,  3.00000000e+000, -3.22776772e-005],
       [ 3.00000000e+000,  6.00000000e+000, -1.50339245e-003],
       [ 3.00000000e+000,  3.50000000e+001, -4.06682536e-002],
       [ 4.00000000e+000,  1.00000000e+000, -7.88473096e-010],
       [ 4.00000000e+000,  2.00000000e+000,  1.27907179e-008],
       [ 4.00000000e+000,  3.00000000e+000,  4.82253727e-007],
       [ 5.00000000e+000,  7.00000000e+000,  2.29220763e-006],
       [ 6.00000000e+000,  3.00000000e+000, -1.67147665e-011],
       [ 6.00000000e+000,  1.60000000e+001, -2.11714723e-003],
       [ 6.00000000e+000,  3.50000000e+001, -2.38957419e+001],
       [ 7.00000000e+000,  0.00000000e+000, -5.90595643e-018],
       [ 7.00000000e+000,  1.10000000e+001, -1.26218089e-006],
       [ 7.00000000e+000,  2.50000000e+001, -3.89468424e-002],
       [ 8.00000000e+000,  8.00000000e+000,  1.12562114e-011],
       [ 8.00000000e+000,  3.60000000e+001, -8.23113409e+000],
       [ 9.00000000e+000,  1.30000000e+001,  1.98097128e-008],
       [ 1.00000000e+001,  4.00000000e+000,  1.04069652e-019],
       [ 1.00000000e+001,  1.00000000e+001, -1.02347471e-013],
       [ 1.00000000e+001,  1.40000000e+001, -1.00181794e-009],
       [ 1.60000000e+001,  2.90000000e+001, -8.08829086e-011],
       [ 1.60000000e+001,  5.00000000e+001,  1.06930319e-001],
       [ 1.80000000e+001,  5.70000000e+001, -3.36622506e-001],
       [ 2.00000000e+001,  2.00000000e+001,  8.91858454e-025],
       [ 2.00000000e+001,  3.50000000e+001,  3.06293169e-013],
       [ 2.00000000e+001,  4.80000000e+001, -4.20024677e-006],
       [ 2.10000000e+001,  2.10000000e+001, -5.90560297e-026],
       [ 2.20000000e+001,  5.30000000e+001,  3.78269476e-006],
       [ 2.30000000e+001,  3.90000000e+001, -1.27686089e-015],
       [ 2.40000000e+001,  2.60000000e+001,  7.30876106e-029],
       [ 2.40000000e+001,  4.00000000e+001,  5.54147154e-017],
       [ 2.40000000e+001,  5.80000000e+001, -9.43697072e-007]])
constRegionFour=[0.11670521452767E4,
-0.72421316703206E6,
-0.17073846940092E2,
0.12020824702470E5,
-0.32325550322333E7,
0.14915108613530E2,
-0.48232657361591E4,
0.40511340542057E6,
-0.23855557567849,
0.65017534844798E3]
ca1=array(constRegionOne)
ca1.shape=(34,3)
```

```
ca2i=constRegionTwoIdeal
ca2r=constRegionTwoResidual
ca4=array(constRegionFour)
ca4.shape=(10,1)
```

A.5 Pitzer.py

```
# Program to calculate activity and osmotic coefficients by Pitzer
ion interaction approach
try:
    import psyco
    psyco.full()
except ImportError:
    pass

#Import required modules
from pitzer_params import *
from dh_constants import *

#Function that calculates activity and osmotic coefficients by
Pitzer ion interaction approach

def Pitzer(T,P,m,component_data,fl):
    R=8.31447215
    z_M=component_data['z_M']
    z_X=component_data['z_X']
    nu_M=component_data['nu_M']
    nu_X=component_data['nu_X']
    nu=nu_M+nu_X
    m_M=m*nu_M
    m_X=m*nu_X
    m0=1.0
    I=0.5*((m_M*z_M**2.0+m_X*z_X**2.0)/m0)
    alpha=component_data['alpha']
    h_phi=exp(-alpha*I**0.5)
    h_g=(2.0/(alpha**2.0*I))*(1.0-(1.0+alpha*I**0.5)*exp(-
1.0*alpha*I**0.5))
    q=component_data['coefficients']
    T_star=1.0
    beta_0 = sum(q[0][i]*fl(T)[component_data['beta_0'][i]] for i in
range(len(component_data['beta_0'])))
    beta_1 = sum(q[1][i]*fl(T)[component_data['beta_1'][i]] for i in
range(len(component_data['beta_1'])))
    C_phi = sum(q[2][i]*fl(T)[component_data['C_phi'][i]] for i in
range(len(component_data['C_phi'])))
    beta_2 =
q[4][0]*(sum(q[4][i+1]*fl(T)[component_data['beta_2'][i]] for i in
range(len(component_data['beta_2'])-1))+ fl(T)[6])
    C_g=C_phi/(2.0*abs(z_M*z_X)**0.5)
    C_gamma=3.0*C_phi/2.0

B_gamma=2.0*beta_0+beta_1*(h_g[0]+h_phi[0])+beta_2*(h_g[1]+h_phi[1])
    B_phi=beta_0+beta_1*h_phi[0]+beta_2*h_phi[1]
    B_g=beta_0+beta_1*h_g[0]+beta_2*h_g[1]
    b=1.2
    he_phi=I**0.5/(1.0+b*I**0.5)
```

```
    he_g=(2.0/b)*log(1.0+b*I**0.5)
    A_phi=Aphi(T)
    f_gamma=-A_phi*(he_g+he_phi)
    f_phi=-A_phi*he_phi
    f_g=-2.0*A_phi*I*he_g
    G_ex_over_RT=f_g
+(m/m0)**2.0*(2.0*nu_M*nu_X)*(B_g+(m/m0)*(nu_M*z_M)*C_g)

ln_gamma=abs(z_M*z_X)*f_gamma+m/m0*(2.0*nu_M*nu_X/nu)*B_gamma+(m/m0)
**2.0*(2.0*(nu_M*nu_X)**1.5/nu)*C_gamma

phi=1.0+abs(z_M*z_X)*f_phi+m/m0*(2.0*nu_M*nu_X/nu)*B_phi+(m/m0)**2.0
*(2.0*(nu_M*nu_X)**1.5/nu)*C_phi
    #Temperature and pressure effects
    beta_0_L = sum(q[0][i]*fl(T)[component_data['beta_0'][i]+11] for
i in range(len(component_data['beta_0'])))
    beta_1_L = sum(q[1][i]*fl(T)[component_data['beta_1'][i]+11] for
i in range(len(component_data['beta_1'])))
    #beta_2 =
q[4][0]*(sum(q[4][i+1]*fl(T)[component_data['beta_2'][i]] for i in
range(len(component_data['beta_2'])-1))+ fl(T)[6])
    #B_L = lambda T: beta_0

    #C_L=C_g.diff(T)
    #B_J=B_L.diff(T)+2.0*B_L/T
    #C_J=C_L.diff(T)+2.0*C_L/T
    #A_J=1.0
    #A_h=1.0
    #Cp_phi=Cp_phi_inf + A_J*he_g/2.0-
2.0*R*T**2.0*(B_J*I+C_J*I**2.0)
    #L_phi=A_h*he_g/2.0-2.0*R*T**2.0*(B_L*I+C_L*I**2.0)
    return [phi,exp(ln_gamma),G_ex_over_RT*R*T]

#Some test

Temps=arange(330.,565.25,25.0)
for temp in Temps:
    print temp, Pitzer(temp,0.1,1.0,NaOH,function_list)
```

A.6 pitzer_params.py

```
#Parameter database to use in Pitzer.py

from numpy import *
#function list
T_star=1.0 #K
function_list=lambda T:[1.0,
               T_star/T,
               log(T/T_star),
               T/T_star,
               T**2.0/T_star**2.0,
               1.0/((T/T_star)-210.0),
               1.0/(647.0-T/T_star),
               T/T_star*(T/T_star-210.0)**3.0,
               T/T_star*(647.0-T/T_star)**3.0,
```

```
               (T/T_star-443.0)**3.0/3.0,
               (T/T_star)**3.0,
               0.0,
                -T_star/T**2,
                1/T,
                1/T_star,
                2.0*T**1.0*T_star**-2.0,
                -1.0/(T_star*(210.0 - T/T_star)**2),
                1/(T_star*(647.0 - T/T_star)**2),
                (-210.0 + T/T_star)**3.0/T_star - 3.0*T*(-210.0 +
T/T_star)**3.0/(T_star**2*(210.0 - T/T_star)),
                -3.0*T*(647.0 - T/T_star)**2.0/T_star**2 + (647.0 -
T/T_star)**3.0/T_star,
                -1.0*(-443.0 + T/T_star)**3.0/(T_star*(443.0 -
T/T_star)),
                3.0*(T/T_star)**3.0/T,
                0.0,
                -T_star/T**2,
                1/T,
                1/T_star,
                2.0*T**1.0*T_star**-2.0,
                -1.0/(T_star*(210.0 - T/T_star)**2),
                1/(T_star*(647.0 - T/T_star)**2),
                (-210.0 + T/T_star)**3.0/T_star - 3.0*T*(-210.0 +
T/T_star)**3.0/(T_star**2*(210.0 - T/T_star)),
                -3.0*T*(647.0 - T/T_star)**2.0/T_star**2 + (647.0 -
T/T_star)**3.0/T_star,
                -1.0*(-443.0 + T/T_star)**3.0/(T_star*(443.0 -
T/T_star)),
                3.0*(T/T_star)**3.0/T
               ]

NaOH={
    'z_M':1.0,
    'z_X':-1.0,
    'nu_M':1.0,
    'nu_X':1.0,
    'coefficients':array([[12.6677,-389.770,-2.13287,3.42184e-3,-
11.6610e-7,-4.16833,1.68567], #beta_0
    [-60.5470,2708.27,9.31351,0.0,-94.2226e-7,-
43.0272,11.9610],#beta_1
    [-0.01687,2.23824,0.0,0.0,0.31094e-7,1.01063,0.0],#C_phi
    [-3.25646e2,2.26011,-3.76119e-3,-2.28323e5,-
1.19721e6,0.0,0.0],#Cp_phi_inf
    [-340.559,-0.19554,-7.4581e-5,4.534,3.505e-2,0.0,0.0] #beta_2
    ]),
    'alpha':array([2.0,8.0]),
    'beta_0':[0,1,2,3,4,5,6],
    'beta_1':[0,1,2,3,4,5,6],
    'C_phi':[0,1,2,3,4,5,6],
    'beta_2':[0,3,1,2,6]
    }

Na2CO3={
    'z_M':1.0,
    'z_X':-2.0,
    'nu_M':2.0,
    'nu_X':1.0,
```

```
    'coefficients':array([[5.153e-1,-5.991e-4,0.0,-25.81,-
2.659,0.0,8.75e-5,-2.66e-8],#beta_0
    [20.44e-1,-43.03e-4,0.0,-25.45,361.8,0.0,0.0,0.0], #beta_1
    [-0.914e-1,0.0,0.0,6.482,8.048,0.0,-2.89e-5,0.0],#C_phi
    [0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],#Cp_phi_inf
    [0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0]#beta_2
    ]),
    'alpha':array([2.0,1.0]),
    'beta_0':[0,3,1,5,6,9],
    'beta_1':[0,3,1,5,6,9],
    'C_phi':[0,3,1,5,6,9],
    'beta_2':[0,0,0,0,0]
    }

NaAlOH4={
    'z_M':1.0,
    'z_X':-1.0,
    'nu_M':1.0,
    'nu_X':1.0,
    'coefficients':array([[9.93607e-4,-4.81382e-6,5.87747e-9,0.0],
#beta_0
    [-1.21394e-2,7.88087e-5,-1.67861e-7,1.16436e-10],#beta_1
    [-4.97929e-4,2.41445e-6,-2.94923e-9,0.0],#C_phi
    [-3.16659e3,2.03730e1,-3.72120e-2,1.79165e-5],#Cp_phi_inf
    [0.0,0.0,0.0,0.0] #beta_2
    ]),
    'alpha':array([2.0,8.0]),
    'beta_0':[0,3,4],
    'beta_1':[0,3,4,10],
    'C_phi':[0,3,4],
    'beta_2':[0,0,0]
    }
```

A.7 dh_constants.py

```
# Program to calculate Debye-Huckel slopes to use in Pitzer.py
# Import required modules

from scipy.interpolate.interpolate import interp1d
from numpy import array

T=array([273.15,283.15,293.15,298.15,303.15,313.15,323.15,333.15,343
.15,
353.15,363.15,373.15,383.15,393.15,403.15,413.15,423.15,
433.15,443.15,453.15,463.15,473.15,483.15,493.15,503.15,
513.15,523.15,533.15,543.15,553.15,563.15,573.15,583.15,
593.15,603.15,613.15,623.15])


A_phi=array([3.77E-01,3.82E-01,3.88E-01,3.91E-01,
3.95E-01,4.02E-01,4.10E-01,4.19E-01,
4.28E-01,4.38E-01,4.49E-01,4.61E-01,
4.73E-01,4.86E-01,4.99E-01,5.14E-01,
5.30E-01,5.46E-01,5.63E-01,5.82E-01,
6.02E-01,6.23E-01,6.45E-01,6.69E-01,
```

```
6.95E-01,7.23E-01,7.54E-1,7.87E-1,
8.23E-1,8.63E-1,9.08E-1,9.60E-1,
1.02E-0,1.09E0,1.18e0,1.29e0,
1.44e0])

AH_over_RT=array([5.51E-01,6.50E-01,7.50E-01,8.02E-01,8.54E-01,
9.64E-01,1.08E+00,1.20E+00,1.33E+00,
1.47E+00,1.61E+00,1.76E+00,1.93E+00,
2.10E+00,2.29E+00,2.49E+00,2.71E+00,
2.95E+00,3.21E+00,3.50E+00,3.82E+00,
4.18E+00,4.58E+00,5.04E+00,5.56E+00,
6.17E+00,6.89E+00,7.75E+00,8.81E+00,
1.01E+01,1.18E+01,1.40E+01,1.71E+01,2.14E+01,
2.80E+01,3.86E+01,5.73E+01])

A_V=array([1.51E+00,1.63E+00,
1.78E+00,1.87E+00,1.96E+00,2.15E+00,
2.37E+00,2.62E+00,2.91E+00,3.23E+00,
3.61E+00,4.04E+00,4.53E+00,5.11E+00,
5.77E+00,6.54E+00,7.44E+00,8.49E+00,
9.73E+00,1.12E+01,1.29E+01,1.50E+01,
1.75E+01,2.05E+01,2.41E+01,2.86E+01,
3.42E+01,4.12E+01,5.02E+01,6.18E+01,
7.72E+01,9.80E+01,1.27E+02,1.69E+02,
2.31E+02,3.30E+02,4.93E+02])

Aphi=interp1d(T,A_phi)
AH=interp1d(T,AH_over_RT)
AV=interp1d(T,A_V)
```

B. DETAILED RESULTS OF MASS BALANCE CALCULATIONS

Table B.1 - Detailed results of mass balance calculations (continued)

	Al2O3	SiO2	Na2O	CO2	CaO	Fe2O3	Amorph	H2O(s)	NaAlO2	NaOH	Na2CO3	H2O	TOTAL
S01	27162	3348	0	405	586	8491	1595	6033	0	0	0	1206	**48826**
S02	0	0	0	0	0	0	0	0	0	3550	0	3846	**7396**
S03	27162	3348	0	405	586	8491	1595	6033	25626	29362	105574	113845	**322027**
S04	27162	3348	0	405	586	8491	1595	6033	25626	29362	105574	151459	**359641**
S05	3939	3348	1899	0	586	8491	1595	1935	62965	7961	106549	160373	**359641**
S06	3939	3348	1899	0	586	8491	1595	1935	62965	7961	106549	146180	**345448**
S07	3939	3348	1899	0	586	8491	1595	1935	62965	7961	106549	136064	**335332**
S08	3939	3348	1899	0	586	8491	1595	1935	62965	7961	106549	128186	**327454**
S09	0	0	0	0	0	0	0	0	0	0	0	14193	**14193**
S10	0	0	0	0	0	0	0	0	0	0	0	7878	**7878**
S11	0	0	0	0	0	0	0	0	0	0	0	9550	**9550**
S12	0	0	0	0	0	0	0	0	0	0	0	31621	**31621**
S13	0	0	0	0	0	0	0	0	0	0	0	31621	**31621**
S14	3939	3348	1899	0	586	8491	1595	1935	62965	7961	106549	118636	**317905**
S15	0	0	0	0	0	0	0	0	0	0	0	120000	**120000**

Table B.1 - Detailed results of mass balance calculations (continued)

	Al2O3	SiO2	Na2O	CO2	CaO	Fe2O3	Amorph	H2O(s)	NaAlO2	NaOH	Na2CO3	H2O	TOTAL
S16	0	0	0	0	0	0	0	0	0	0	0	0	**0**
S17	0	0	0	0	0	0	0	0	0	0	0	0	**0**
S18	0	0	0	0	0	0	0	0	0	0	0	0	**0**
S19	0	0	0	0	0	0	0	0	0	0	0	0	**0**
S20	0	0	0	0	0	0	0	0	10955	1385	18538	120000	**150878**
S21	0	0	0	0	0	0	0	0	0	0	0	0	**0**
S22	0	0	0	0	0	0	0	0	0	0	0	0	**0**
S23	0	0	0	0	0	0	0	0	0	0	0	0	**0**
S24	0	0	0	0	0	0	0	0	0	0	0	0	**0**
S25	3939	3348	1899	0	586	8491	1595	1935	11532	1458	19514	37209	**91504**
S26	3939	3348	1899	0	586	8491	1595	1935	73920	9346	125087	238517	**468664**
S27	3939	3348	1899	0	586	8491	1595	1935	577	73	976	37209	**60626**
S28	0	0	0	0	0	0	0	0	62389	7888	105574	201309	**377160**
S29	0	0	0	0	0	0	0	0	0	0	0	13689	**13689**
S30	0	0	0	0	0	0	0	0	62389	7888	105574	187620	**363471**
S31	112412	0	0	0	0	0	0	59512	66146	11671	121074	230448	**601263**
S32	135273	0	0	0	0	0	0	71615	29389	29602	121074	214310	**601263**

Table B.1 - Detailed results of mass balance calculations (continued)

	Al2O3	SiO2	Na2O	CO2	CaO	Fe2O3	Amorph	H2O(s)	NaAlO2	NaOH	Na2CO3	H2O	TOTAL
S33	135273	0	0	0	0	0	0	71615	12059	12147	49682	87367	**368143**
S34	22861	0	0	0	0	0	0	12103	72	73	298	2534	**37941**
S35	0	0	0	0	0	0	0	0	8297	8357	34181	58710	**109545**
S36	22861	0	0	0	0	0	0	12103	72	73	298	19584	**54992**
S37	0	0	0	0	0	0	0	0	0	0	0	17051	**17051**
S38	0	0	0	0	0	0	0	0	67	66	298	16705	**17136**
S39	112412	0	0	0	0	0	0	59512	3690	3717	15203	26123	**220657**
S40	22861	0	0	0	0	0	0	12103	6	7	0	2879	**37855**
S41	0	0	0	0	0	0	0	0	25626	25812	105574	185654	**342666**
S42	0	0	0	0	0	0	0	0	25626	25812	105574	185654	**342666**
S43	0	0	0	0	0	0	0	0	25626	25812	105574	108793	**265805**
S44	0	0	0	0	0	0	0	0	0	0	0	76861	**76861**
S45	0	0	0	0	0	0	0	0	0	0	0	31970	**31970**
S46	0	0	0	0	0	0	0	0	0	0	0	31970	**31970**
S47	0	0	0	0	0	0	0	0	0	0	0	10116	**10116**
S48	0	0	0	0	0	0	0	0	0	0	0	21854	**21854**
S49	27162	3348	0	405	586	8491	1595	6033	25626	29362	105574	113845	**322027**

Table B.1 - Detailed results of mass balance calculations (continued)

	Al2O3	**SiO2**	**Na2O**	**CO2**	**CaO**	**Fe2O3**	**Amorph**	**H2O(s)**	**NaAlO2**	**NaOH**	**Na2CO3**	**H2O**	**TOTAL**
S50	0	0	0	0	0	0	0	0	0	0	0	37614	**37614**
S51	3939	3348	1899	0	586	8491	1595	1935	62965	7961	106549	118517	**317786**
S52	0	0	0	0	0	0	0	0	0	0	0	119	**119**