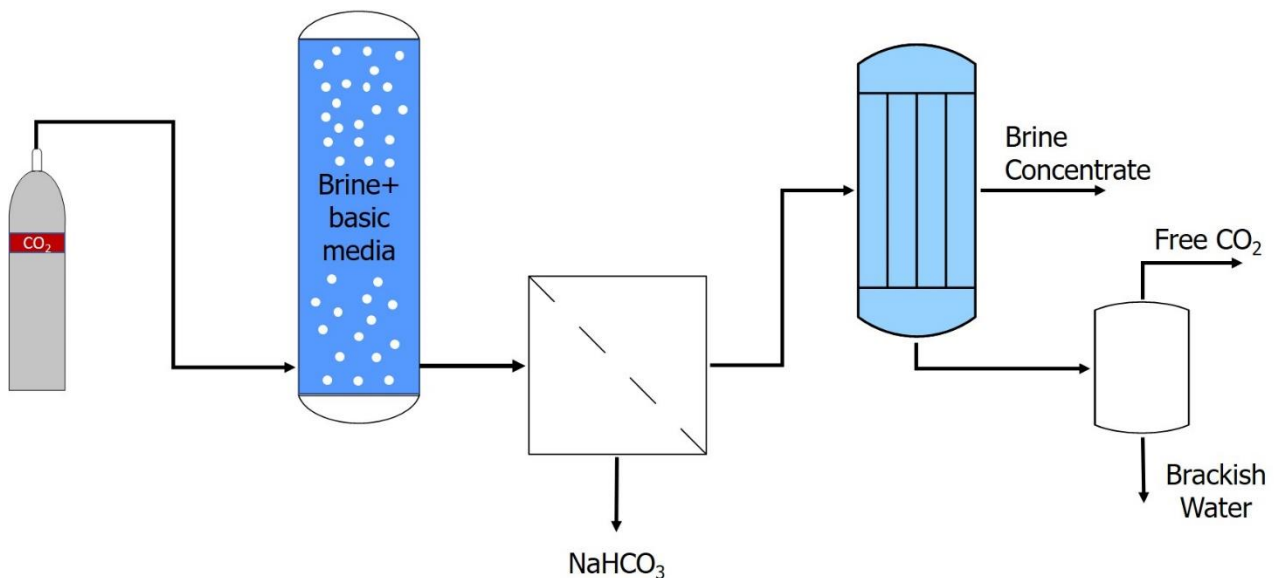


MASTER THESIS NO. 2022: 86

College of Engineering

Department of Chemical and Petroleum Engineering

**DESALINATION WASTE REVITALIZATION USING
CARBON CAPTURE: PROCESS OPTIMIZATION USING
MACHINE LEARNING***Ahmed Mohamed Nasereldin Mohamed Elsayed**May 2022*

United Arab Emirates University

College of Engineering

Department of Chemical and Petroleum Engineering

DESALINATION WASTE REVITALIZATION USING CARBON
CAPTURE: PROCESS OPTIMIZATION USING MACHINE
LEARNING

Ahmed Mohamed Nasereldin Mohamed Elsayed

This thesis is submitted in partial fulfilment of the requirements for the degree of Master
of Sciences in Chemical Engineering

May 2022

**United Arab Emirates University Master Thesis
2022: 86**

Cover: Process overview of the combined sodium and chloride removal from brine solution

(Photo: By Ahmed Mohamed Nasereldin Mohamed Elsayed)

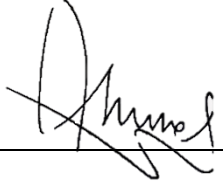
© 2022 Ahmed Mohamed Nasereldin Mohamed Elsayed, Al Ain, UAE

All Rights Reserved

Print: University Print Service, UAEU 2022

Declaration of Original Work

I, Ahmed Mohamed Nasereldin Mohamed Elsayed, the undersigned, a graduate student at the United Arab Emirates University (UAEU), and the author of this thesis entitled “*Desalination Waste Revitalization using Carbon Capture: Process Optimization using Machine Learning*”, hereby, solemnly declare that this is the original research work done by me under the supervision of Prof. Ali H. Al-Marzouqi, in the College of Engineering at UAEU. This work has not previously formed the basis for the award of any academic degree, diploma or a similar title at this or any other university. Any materials borrowed from other sources (whether published or unpublished) and relied upon or included in my thesis have been properly cited and acknowledged in accordance with appropriate academic conventions. I further declare that there is no potential conflict of interest with respect to the research, data collection, authorship, presentation and/or publication of this thesis.

Student's Signature: _____ 

Date: ___20/12/2022_____

Advisory Committee

1) Advisor: Prof. Ali H. Al-Marzouqi

Title: Professor and Dean of the College of Graduate Studies

Department of Chemical and Petroleum Engineering

College of Engineering

2) Co-advisor: Prof. Nazar Zaki

Title: Professor of Computer Science (Machine Learning)

Department of Computer Science and Software Engineering

College of Information Technology

Approval of the Master Thesis

This Master Thesis is approved by the following Examining Committee Members:

- 1) Advisor (Committee Chair): Prof. Ali H. Al-Marzouqi

Title: Professor

Department of Chemical & Petroleum Engineering

College of Engineering

Signature Ali Hassan

Date 11/10/2022

- 2) Member: Mohammednoor Altarawneh

Title: Associate Professor

Department of Chemical & Petroleum Engineering

College of Engineering

Signature Mohammednoor

Date 11/10/2022

- 3) Member (External Examiner): Cristian Picioreanu

Title: Associate Professor

Department of Environmental Science and Engineering

Institution: King Abdullah University of Science and Technology, Saudi Arabia

Signature Cristian Picioreanu

Date 11/10/2022

This Master Thesis is accepted by:

Acting Dean of the College of Engineering: Professor Mohamed H. Al-Marzouqi

Signature Mohamed AlMarzouqi

Date Jan. 15, 2023

Dean of the College of Graduate Studies: Professor Ali Al-Marzouqi

Signature Ali Hassan

Date 16/01/2023

Abstract

Over the next five to ten years, desalination will play an ever-important role in our society. Developing nations will look at desalination from the point of water scarcity, while developed nations will consider the perspective of the Water-Energy-Food nexus. With current technology, for every liter of freshwater produced, one and a half liters is thrown away as waste. This body of research proposes to employ chemical precipitation in ammoniated brine solutions, that have reasonably high ionic concentrations. CO₂ absorption in Bubble Column Reactors can reduce sodium ions by precipitating NaHCO₃. Also, the absorbed CO₂ can reduce chloride ions and induce hydrate formation, by operating the column at a lower temperature (10 – 15°C).

The Bubble Column Reactor model includes CO₂ absorption, aqueous reaction, and salt precipitation. The kinetic/equilibrium reaction system is simulated and optimized using non-linear programming. The hydrate formation process is modeled and optimized using Machine Learning. The results from the optimization show that brine with 23400 ppm of Na⁺ ions and 44000 ppm of Cl⁻ ions can be reduced by 72.5% and 54.2%, respectively. The absorption process will use 1.15 mol of CO₂ absorbed per mole of NH₃ dissolved with a total of 8.702 mol of CO₂ absorbed.

This work develops a new process for Zero Liquid Discharge. The proposed process utilizes CO₂ absorption in Bubble Column Reactor to remove Na⁺ and Cl⁻ ions. This study provides all the necessary equations to determine the appropriate operating conditions. These results can be taken into the lab and used as an initial point for optimization using statistical methods.

Keywords: Desalination revitalization, chemical precipitation, carbon capture, Bubble Column Reactor, hydrate formation, Machine Learning Application

Title and Abstract (in Arabic)

معالجة المياه المالحة بالتقاط ثاني أكسيد الكربون: تحسين العملية باستخدام التعلم الآلي

الملخص

على مدى السنوات الخمس إلى العشر القادمة ، ستلعب تحلية المياه دورًا مهمًا دائمًا في مجتمعنا. سنتنظر الدول النامية إلى تحلية المياه من وجهة ندرة المياه ، بينما سنتنظر الدول المتقدمة إلى منظور العلاقة بين المياه والطاقة والغذاء. باستخدام التكنولوجيا الحالية ، مع إنتاج كل لتر من المياه العذبة ، يتم إنتاج لتر ونصف كنفائات. تقترح الأطروحة هذه استخدام الترسيب الكيميائي في المحاليل الملحية المذوبة بالأمونيا، التي تحتوي على تركيزات أيونية عالية بشكل معقول. امتصاص ثاني أكسيد الكربون في العمود الفقاعي يمكنه أن يقلل من أيونات الصوديوم عن طريق ترسيب كربونات الصوديوم (NaHCO_3). أيضًا ، يمكن لثاني أكسيد الكربون الممتص تقليل أيونات الكلوريد والحث على تكوين الهيدرات ، عن طريق تشغيل العمود عند درجة حرارة منخفضة (10 - 15 درجة مئوية).

نموذج مفاعل العمود الفقاعي يحتوي على نماذج رياضية لامتناس ثاني أكسيد الكربون وللتفاعل المائي وترسيب الملح. يتم محاكاة نظام التفاعلات الحركية و التوازنية وتحسينه باستخدام البرمجة غير الخطية. تم تصميم عملية تكوين الهيدرات وتحسينها باستخدام التعلم الآلي. تظهر النتائج أنه يمكن تقليل محلول ملحي الذي يحتوي على 23400 جزء في المليون من الأيونات الصوديوم و 44000 جزء في المليون من الأيونات الكلوريد بنسبة 72.5% و 54.2% على التوالي. ستستخدم عملية الامتناس 1.15 مول من ثاني أكسيد الكربون لكل مول من الأمونيا الذائبة بإجمالي 8.702 مول من ثاني أكسيد الكربون الممتصة.

يطور هذا العمل عملية جديدة لتصريف السوائل الصفريّة. تستخدم العملية المقترحة امتصاص ثاني أكسيد الكربون في مفاعل العمود الفقاعي لإزالة أيونات الصوديوم والكلوريد. توفر هذه الدراسة جميع المعادلات اللازمة لتحديد ظروف التشغيل المناسبة للمفاعل العمود الفقاعي. يمكن أخذ هذه النتائج إلى المختبر واستخدامها كنقطة مبدئية للتحسين العملية باستخدام الأساليب الإحصائية.

مفاهيم البحث الرئيسية: تنشيط تحلية المياه، ترسيب الكيميائي، التقاط الكربون ، مفاعل عمود الفقاعي، التعلم الآلي.

Author Profile

Ahmed Elsayed is a graduate student in the Chemical Engineering department at United Arab Emirates University. His research interests are mainly resource and waste management, and their integration with the broader circular economy. He obtained his bachelor's degree in chemical engineering from Texas A&M University. His undergraduate research focused on process systems engineering by developing AI tools that could assist in waste management. Ahmed's current work studies the overall desalination process to deliver a solution that integrates with the circular economy. He wants to expand his work by engineering solutions for direct air capture, wastewater treatment, and lithium extraction for high-volume battery production. In his free time, he loves playing video games, reading about macroeconomics, and tracking new advancements in technology.

Acknowledgements

Before all, praise to God for guiding me throughout this journey, and for giving me the power to finish my dissertation. Firstly, I am very thankful to my advisor, Dr. Ali Al-Marzouqi, for introducing me to the desalination field and for his constant guidance along the way. I am very thankful for my co-advisor, Dr. Nazar Zaki, and his support in Machine Learning Optimization. I would like to thank the professors from the Chemical Engineering Department at UAE University. I would like to thank Dr. Mohamednoor for his assistance in modeling absorption with kinetic reactions, and Dr. Basim Abu-Jdayil for building my academic skills. I would like to thank the chemistry department at UAE University for their assistance in building the aqueous inorganic chemistry model. Also, I would like to thank Dr. Allen Leal from ETH – Zurich for his assistance in combining the thermodynamic and kinetic models.

Secondly, special thanks are extended to my colleagues and research partners. I would like to thank my colleagues from our research group. I would like to personally thank Dr. Ameera F. Mohammad for her assistance in CO₂ bubbling, Jawad Mustafa for his constant insights into the research work, and Aya Mourad for exchanging research ideas. Many thanks to my M.Sc. and Ph.D. colleagues from the chemical engineering department. Special thanks go out to Thair Alawadh, Ahmad Abu Abeid, Labeed Ali, Hyder Abdallah, Sidra Ahmed, Noran Mousa, Sabeera Harris, Anusha Mairpady, and Amal Mlhem.

Finally, I am very thankful to my parents for their lifelong love and support. Also, I am thankful to the many brothers, and sisters who helped me along the way. I cherish all the motivation and encouragement I have received during my research studies. I am grateful for all the assistance and friendship I have built along the way.

Dedication

To a future generation with abundance in water resources

Table of Contents

Title.....	i
Declaration of Original Work.....	iii
Advisory Committee.....	iv
Approval of the Master Thesis	v
Abstract.....	vii
Title and Abstract (in Arabic).....	viii
Author Profile	ix
Acknowledgements.....	x
Dedication.....	xi
Table of Contents.....	xii
List of Tables	xv
List of Figures.....	xvi
List of Abbreviations	xviii
Chapter 1: Introduction.....	1
1.1 Overview	1
1.2 Current State of Research.....	2
1.3 Scope of Thesis	3
Chapter 2: Literature Review	6
2.1 Brine treatment technologies.....	6
2.1.1 Conventional Method	7
2.1.2 Emerging technologies	9
2.1.3 Commercial Technologies	13
2.1.4 Summary	14
2.2 Aqueous CO ₂ Absorption.....	15
2.2.1 Chemical Absorbents	15
2.2.2 Column Operating Conditions	18
2.2.3 Summary	18
Chapter 3: Research Methodology	20
3.1 Reaction System Model	20
3.1.1 Kinetic Reactions	20
3.1.2 Thermodynamic Reactions	22

3.1.3 Fugacity and Activity.....	31
3.2 Absorption System Model.....	36
3.2.1 Lumped Method.....	36
3.2.2 Distributed Method.....	38
3.3 Numerical Solutions.....	41
3.3.1 Reaction System.....	41
3.3.2 Reaction and Absorption System.....	43
3.3.3 Model Simplification.....	43
3.4 Gas Hydrate Model.....	44
3.4.1 Data Collection.....	45
3.4.2 Machine Learning Algorithms.....	45
Chapter 4: Results and Discussion.....	52
4.1 Model Validation.....	52
4.1.1 CO ₂ Absorption.....	52
4.1.2 CO ₂ Reaction.....	53
4.1.3 Solubility Diagrams.....	55
4.1.4 Validation of CO ₂ -hydrate data.....	56
4.2 Simulation Results.....	69
4.2.1 CO ₂ Bubbling in Basic Solution.....	69
4.2.2 Operating Condition Effects.....	78
4.3 Optimization of Solid Precipitation.....	80
4.3.1 Base Comparison.....	81
4.3.2 Simulation of Optimum Conditions.....	82
4.3.3 CO ₂ -Hydrate Optimization.....	82
Chapter 5: Conclusion.....	84
References.....	86
Appendix.....	106
A.1 Pitzer Interactions.....	106
A.1.1 Two-Ion Interaction.....	108
A.1.2 Three-Ion Interaction.....	114
A.2 CO ₂ -Hydrate Experimental Data.....	118
A.3 Modeling Codebase.....	126
A.3.1 IAPWS_model.py.....	126

A.3.2 HKFT_model.py	127
A.3.3 gases_model.py	129
A.3.4 solids_model.py	129
A.3.5 peng_robinson.py	130
A.3.6 pitzer_model.py.....	130
A.3.7 simulation.py.....	138
A.3.8 optimization.py	140
A.3.9 CO2_hydrate.py	142

List of Tables

Table 1: List of salts and their respective eutectic point.	12
Table 2: List of absorbents and their enthalpy of absorption	17
Table 3: List of absorbents and their enthalpies of desorption.....	18
Table 4: Parameters (I_i, J_i, n) used in the Gibbs free energy formulation of water, Equation 4.	24
Table 5: Parameters ($ag, i, bg, , fg, i$) used in the $g(\rho_w, T)$ function	26
Table 6: List of species and their corresponding parameters for the HKFT model.....	27
Table 7: Parameters (N_i, i, j) used in the $g(\rho, T)$ function.....	28
Table 8: List of solids and gases, and their respective parameters for Gibbs free energy calculation	30
Table 9: List of specific groups used to calculate Gibbs free energy for complex solids	31
Table 10: Parameters used in calculating the density of saline water.	38
Table 11: List of the eight ML algorithms along with their respective results.	64
Table 12: Comparison of all three regression trees using MAE score.	65
Table 13: Scoring metric comparison for decision tree regression model, with and without PCA.	68
Table 14: Initial concentration of the ions considered in the CO ₂ bubbling process.	81
Table 15: Conclusions made from the optimization work.	81
Table 16: List of two-ion interactions used in the Pitzer Model	108
Table 17: List of three-ion interactions used in Pitzer Model	114
Table 18: CO ₂ -hydrate data used in the machine learning process.	118

List of Figures

Figure 1: Main dissolved ions (wt%) in brine solutions obtained from a Multi-Stage Flash desalination unit in Abu Dhabi, UAE	7
Figure 2: The desalination technology favorable at each salinity concentration	8
Figure 3: Zwitterion form of carbamic acid.	22
Figure 4: Simulation flowchart of modeling reactions.	44
Figure 5: Multi-Layer Perceptron architecture with an input layer composed of n inputs, n hidden layers, and an output layer with n responses.....	47
Figure 6: A typical structure of a decision tree.	48
Figure 7: A typical Support Vector Machine used for Regression with the hyperplane being the bold, solid line in the middle.....	50
Figure 8: CO ₂ solubility in water at various temperatures and pressures, experimental data (dots), and our model (lines).	53
Figure 9: Simulation of 3.8 mM CO ₂ (aq.) and 4.0 mM NH ₃ (aq.) reaction a) for 10 seconds [190], and b) for 40 seconds (this study).	54
Figure 10: Simulation of 3.8 mM CO ₂ (aq.) and 4.0 mM NH ₃ (aq.) reaction a) for 120 seconds [190], and b) for 10 minutes (this study).	55
Figure 11: Solubility plots of the six salts used in the reaction model.....	56
Figure 12: Scatter plots of the three output responses vs the five principal components used for cation data.	59
Figure 13: Scatter plots of the three output responses vs the five principal components used for anion data.	61
Figure 14: Scatter plots of the three output responses vs the five principal components used for the operating temperature and pressure data.	63
Figure 15: Feature importance for the top three models selected (decision tree, Random Forest, and XGBoost).....	65
Figure 16: Predicted vs actual scatter plot for decision tree regression model.	66
Figure 17: Predicted vs actual scatter plot for Random Forest regression model.	66
Figure 18: Predicted vs Actual scatter plot for XGBoost regression model.	67
Figure 19: Predicted vs actual scatter plot for decision tree regression model without PCA as a preprocessing step.	68
Figure 20: Predicted vs actual scatter plot for decision tree regression model with PCA as a preprocessing step.....	68
Figure 21: Final Machine Learning system used for Chloride removal modeling.	69
Figure 22: The measure of pH level in CO ₂ bubbling in a) hydroxide brine solution and b) ammonia brine solution, operating at temperature of 25°C and pressure of 1 atm.	71

Figure 23: Reaction mechanism of bubbling CO ₂ in 1 M ammonia solution at a temperature of 25°C and pressure of 1 atm.	72
Figure 24: CO ₂ absorption rate in 5M NaOH, at a temperature of 25°C, and different pressures.....	73
Figure 25: CO ₂ absorption capacity in 5M NaOH, at a pressure of 1 atm, and different temperatures.....	74
Figure 26: CO ₂ absorption capacity with the absorbent as a parameter, at a temperature of 25°C and pressure of 1 atm.	75
Figure 27: 5 molal of dissolved CO ₂ and its speciation in water as a function of pH level, at a temperature of 25°C and pressure of 1 atm.....	76
Figure 28: Salt precipitation in hydroxides, at a temperature of 25°C and pressure of 1 atm, in a) 2.5M of Ca(OH) ₂ and b) 5M of NaOH.....	77
Figure 29: Key factors and their effects on temperature.	78
Figure 30: CO ₂ absorption rate in 5M NaOH at 1 bar and different temperatures.	79
Figure 31: CO ₂ absorption capacity with NaOH concentration as a parameter, operating at a temperature of 25°C and pressure of 1 atm.	80
Figure 32: Simulation of salt precipitation in bubble column reactor, operating at 22.2°C and 68.7 bars.....	82

List of Abbreviations

Acronyms

ARROW	Advanced Reject Recovery Of Water
CFD	Computational Fluid Dynamics
DBM	Discrete Bubble Model
DCMD	Direct contact Membrane Distillation
D-H	Debye-Hückel equation
DNS	Direct Numerical Simulation
EOS	Equation Of State
FO	Forward Osmosis
HKFT	Helgeson-Kirkham-Flowers-Tanger electrolyte model
IAPWS	International Association for Properties of Water and Steam
LES	Large Eddy Simulation
MD	Membrane Distillation
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mixed Solvent Electrolyte
MSF	Multi-Stage Flash
OARO	Osmotically Assisted Reverse Osmosis
PCA	Principal Component Analysis
ppm	parts per million
ppt	precipitate
PR	Peng-Robinson
RANS	Reynolds Averaged Navier Stokes
RO	Reverse Osmosis

SIT	Specific-ion Interaction Theory
SPARRO	Slurry Precipitation And Recycle Reverse Osmosis
SRK	Soave-Redlich-Kwong
SVM	Support Vector Machine
TDS	Total Dissolved Solids
UNIQUAC	Universal Quasi-Chemical model
VSEP	Vibratory Shear Enhanced Process
WAIV	Wind-Aided Intensified eVaporation
XGBoost	eXtreme Gradient Boost
ZLD	Zero Liquid Discharge

Symbols

P	Pressure of the system (bar), unless otherwise specified
T	Temperature of the system (K), unless otherwise specified
ρ	The density of substance (kg/m^3), unless otherwise stated
G_f°	Standard molar Gibbs free energy of formation (J/mol), unless otherwise specified
S°	Standard molar entropy of substance ($\text{J K}^{-1} \text{mol}^{-1}$)
ΔH	The change in molar enthalpy from reference temperature (J/mol),
C_p	Heat capacity of a substance ($\text{J K}^{-1} \text{mol}^{-1}$)
V	The molar volume of a substance (m^3/mol)
ϕ	The fugacity coefficient of a gas species (unitless) The osmotic coefficient of aqueous solution (unitless)
γ	The activity coefficient of an aqueous species (unitless)
R	Universal gas constant ($8.304 \text{ J K}^{-1} \text{mol}^{-1}$), unless otherwise specified

r	Rate of reaction ($\text{mol L}^{-1} \text{s}^{-1}$)
k	Reaction rate constant ($1/\text{s}$)
C	Thermal conductivity of the phase, Equation 19-20
n_i, I_i, J_i	Concentration of species (mol/L)
P^*	Parameter list for Equation 4, listed in Table 4
T^*	Constant pressure term for Equation 4, with a value of 16.53 MPa
a_1, a_2	Constant temperature term for Equation 4, with a value of 1386 K
a_3, a_4	Constants for the pressure term in the non-solvation part of Equation 6
c_1, c_2	Constants for the temperature/pressure terms in the non-solvation part of Equation 6
w_{ref}	Constants for the temperature term in the non-solvation part of Equation 6
Ψ	Constant for the solvation part of Equation 6
Θ	Pressure constant for Equation 6, with value of 2600 bar
w_j	Temperature constant for Equation 6, with value of 228 K
ϵ	The born coefficient function of species j (J/mol)
Y_{ref}	The dielectric constant of water function (unitless)
Z_j	The derivative of the inverted dielectric constant with respect to temperature ($1/\text{K}$)
$r_{e,j}$	The electric charge of species j
N_A	The effective electrostatic radius of the aqueous species j (\AA)
e	The Avogadro's number, with a constant value (6.02×10^{23} atoms/mol)
g	The absolute electric charge ($4.803 \times 10^{-10} \text{ cm}^{3/2} \text{ g}^{1/2} \text{ s}^{-1}$)
	g function for Equation 6, the HKFT EoS (\AA)
	g -factor function for Equation 7, the dielectric constant (unitless)

	g function for cation-anion interactions in Equations 14-17
a_g, b_g, f	Parameter list associated with g function for Equation 6, listed in Table 5
ϵ_0	The permittivity of free space ($8.85\text{e-}12 \text{ C}^2 \text{ J}^{-1} \text{ m}^{-1}$)
k	The Boltzmann constant ($1.38\text{e-}23 \text{ J/K}$), Equation 7 and 14 Constant characteristic of each gas, Equation 13
α	The mean molecular polarizability of water ($1.64\text{e-}40 \text{ C}^2 \text{ J}^{-1} \text{ m}^{-2}$), Equation 7 Dimensionless function equal to unity at critical temperature, Equation 13
μ	The dipole moment of water ($6.14\text{e-}30 \text{ C m}$) Neutral-neutral-cation interaction term for Pitzer model, Equation 14-17 Dynamic viscosity of fluid ($\text{kg m}^{-1} \text{ s}^{-1}$)
MW_w	Molecular weight of water ($18.02\text{e-}3 \text{ mol/kg}$)
MW_{CO_2}	Molecular weight of CO_2 gas ($44.01\text{e-}3 \text{ mol/kg}$)
a, b, c, d	Parameters/constants associated with various functions
Z	Compressibility factor, Equation 13 and 18 Absolute charge strength (mol/kg), Equation 14
w	The acentric factor for the fugacity coefficient, Equation 13
A^ϕ	The temperature-dependent constant from the Debye-Hückel equation.
b	Constant term in the Debye-Hückel equation ($1.2 \text{ kg}^{1/2} \text{ mol}^{-1/2}$)
m	Molal mass of an aqueous species (mol/kg)
z	The electric charge of aqueous species
I	Ionic strength of the aqueous solution (mol/kg)
F	The Debye-Hückel contributing factor in pitzer model, Equation 15 and 16
B	Cation-anion interaction term for Pitzer model

C	Charge-weighted cation-anion interaction term for Pitzer model
θ	Cation-cation/anion-anion interaction term for Pitzer model
λ	Neutral-cation/neutral-anion interaction term for Pitzer model
ψ	Cation-cation-anion/anion-anion-cation interaction term for Pitzer model
ξ	Neutral-neutral-anion interaction term for Pitzer model
Φ	Long-range interaction function for θ in Pitzer model
x_{ij}	Lumped ionic-strength factor for J_0 and J_1 functions
J_0	Cluster-integral approximation for long-range interactions
J_1	Derivative of J_0 with respect to the ionic-strength factor (x_{ij})
$k_l a$	Volumetric mass transfer coefficient for CO ₂ absorption (1/s)
G	Gravitational acceleration constant (9.81 m/s ²)
U_g	The average gas bubble velocity in a bubble column reactor (m/s)
S	Salinity of aqueous brine solution (g TDS / kg of water), Equation 18
σ	Surface tension of aqueous solution (N/m)
D	The diffusion coefficient of the gas in an aqueous water solution (m ² /s)
a_0, a_1, a_2, a_3, a_4	Constants used in calculating the density of an aqueous solution for equation 18
b_0, b_1, b_2, b_3, b_4	Salinity-associated constants used in calculating the density of an aqueous solution for equation 18
t	Time variable (s)
ϵ	The dielectric constant of water function (unitless) Void fraction, Equation 19-22
∇	Del operator for radial and height z-direction
\cdot	Dot product operator between vectors

k	Thermal conductivity of the phase, Equation 19-20
m	Molar flowrate of gas transferred between phases
τ	Shear stress matrix for r- and z- direction
I	Identity matrix
Re	Reynolds number
C_D	Coefficient of drag
A_p	Interfacial area of the bubble
d_p	Diameter of bubble gas
M_D	Momentum of aqueous phase drag on the bubble

Superscripts

ϕ	Term associated with water equation of pitzer model, Equation 14
'	Derivative of the term attached to Another ion when attached to an anion (a) or a cation (c)
E	Accounts for unsymmetrical mixing
T	Transpose operator on a vector

Subscripts

i, j, k	Iterators for summation signs and product signs
$w, water$	Physical property of water
ref	Reference value
c	The critical value of the property List of all cations in Pitzer model, Equations 14-17
r	The reduced value of the property
$(aq.)$	Aqueous/dissolved species in water
non $- solvation$	Non-solvation portion of the Gibbs free energy in Equation 6

<i>solvation</i>	Solvation portion of the Gibbs free energy in Equation 6
<i>maier</i> – <i>kelley</i>	The Maier Kelley formulation of the heat capacity function
<i>multi – salt</i>	A grouping method for evaluating Gibbs free energy of complex solids
<i>n</i>	List of all neutral ions in Pitzer model, Equations 14-17
<i>a</i>	List of all anions in Pitzer model, Equations 14-17
<i>M</i>	Metal-ion/cation being calculated in Equation 15
<i>X</i>	Halide/anion being calculated in Equation 16
<i>sol</i>	Aqueous brine solution
<i>abs</i>	Absorption rate
<i>T</i>	Turbulence term
<i>eff</i>	The overall effective viscosity

Chapter 1: Introduction

1.1 Overview

Access to freshwater is one of the main necessities in life. Today, one percent of freshwater produced comes from desalination treatment plants. These plants supply 300 million people in 150 countries across the world [1]. Around 237 million m³ of water is desalinated in 16000 plants daily [2]. Desalinated water is used in many applications. Mostly, it is used in municipalities, in the form of potable water, for household applications, commercial uses, and public demands. The rest is mostly used industrially in cooling/heating utilities, energy generation, and washing products [3]. There are other uses such as irrigation, military use, and other miscellaneous activities. Water generation allows communities to grow as it provides the necessary resources needed to live a comfortable lifestyle.

Nations are pushed towards desalination because of the minimum amount of freshwater available. In fact, less than three percent of water bodies are considered freshwater, and only one percent is drinkable [4]. These bodies come in the form of lakes, rivers, groundwater, or even glaciers. Over the following couple of decades, 50%-60% of the global population will face water scarcity [5]. Reasons, such as climate change, population growth, and urbanization, are amplifying the water shortage problem. Among the many, California in USA, Cape Town in South Africa, and Egypt have all had their water shortage problems in the past decade [6]–[8]. The recent launch of water trading as a commodity [9], highlights the fears of water scarcity. The rising amount of global water shortages should lead to water-trade diplomacy [10], [11].

Another factor that will play an important role over the next couple of decades is the water-energy nexus. Energy is needed to produce purified/desalinated water, and water is needed to produce energy when utilizing steam turbines [12]. There are five important aspects to the nature of the nexus. The main aspect is the economic one, as it highlights that the water/electricity price structure will ultimately affect local/national economies [13]. In addition, technological advancements in any or both fields will lead to better outcomes for society [13]. Therefore, policies dictated at local and national levels will have to be integrated to ensure better outcomes. This will affect social

behavior and public perception of the matter. Finally, the nexus will allow governments to value their environment for what its worth of water and energy resources [13].

In the gulf region, United Arab Emirates depends on desalinating seawater to provide freshwater [14]. Geographically, the gulf region is abundant with petroleum and natural gas, yet they lack the natural resources needed to provide fresh drinkable water. UAE uses combined power and desalination plants to secure its energy and freshwater needs. There are two main types of desalination methods: (1) processes that depend on pressure, mainly reverse osmosis (RO); (2) processes that depend on temperature, mainly thermal distillation [15]. Although RO requires lower energy needs, thermal distillation is mainly used as it is integrated with energy production [14]. Combined power and desalination stations burn natural gas to evaporate seawater. The steam produced powers turbines to generate electricity, and the water exiting the turbines is used as potable water [16].

UAE has been working on national energy and economic strategies to secure its water and energy needs [14]. To be able to meet their ever-growing population demand, they will need to provide reliable electricity and freshwater resources. However, as the demand keeps rising, the environmental burden will keep persisting and increasing. UAE has been working on developing innovative solutions to this existing problem [17]. El-Naas et al. [18] have been proposing the management of desalination reject brine with simultaneous carbon dioxide capture.

1.2 Current State of Research

To fully make desalination affordable, one must solve the problem from its roots. The problem lies in two-fold: (1) wastewater recycling is less than 10% [19], and (2) desalination production is not as cheap as freshwater production [20]. Wastewater treatment is complex as it involves various processes to purify water streams. A typical treatment scheme includes physical/mechanical treatment, followed by chemical and biological treatments [21]. The final stage involves sludge disposal according to local regulations [21]. The issue lies in attempting to remove all impurities dissolved/suspended. The most difficult item to remove is Pharmaceuticals and Personal Care Products [22], followed by organics and microplastics [23]. The aeration process,

which degrades biological and organic matter, consumes most of the energy in wastewater treatment plants [24]. While wastewater is a huge problem, most research is focused on developing more affordable desalination systems [25].

Currently, desalinated water production costs 2-3 times the price of producing freshwater [26]. Reverse Osmosis (RO) and Multi-Stage Flash (MSF) are the most prominent technologies used in large-scale water production [2]. Most of the cost for producing desalinated water comes from energy costs (mechanical pressure or thermal energy) and amortized capital costs [27]. Seawater is the most water source used, followed by brackish water. RO plants can recover 42% to 65%, while MSF can only recover 22% to 33% [2]. Virtually, all newly built desalination plants use RO technology. Out of the 237 million m³ processed, only 97 million m³ is considered freshwater, as 141 million m³ is regarded as waste [2]. There is a lot of room for improvement.

The current trend in desalination technologies is to develop better membranes for RO applications [28]. There has been a focus on high-pressure RO systems. Certain membranes have been developed to handle high pressures and recover more water [29]. Another focus has been shifted in the direction of utilizing a new-generation of antiscalants in membranes [30]. Scaling can cause chemical fouling, which allows membranes to recover less water than optimal. This is one of the main limitations with current membrane technologies. New advancements should lower operating costs along with membrane replacement costs. Another trend is to use electrodialysis in brackish water desalination plants. At a Total Dissolved Solids (TDS) level less than 30 g/L, electrodialysis can compete with the price of RO [31]. Electrodialysis has a high energy efficiency with 90% water recovery, when compared to 65% recovery from RO technology [2].

1.3 Scope of Thesis

Research objectives describe what you expect to achieve by your research study. It should be closely related to the statement of the problem. For example, if the problem identified is low utilization of a health care service, the general objective of the study could be to identify the reasons for this low uptake, in order to find ways of improving it.

Research advancements in desalination technology require building state-of-the-art technology. This drives current research to focus on using cheaper material membranes that will require less energy in production. However, sometimes ignored is the amount of water discharged as desalination waste. For each cubic meter of desalinated water produced, one and a half cubic meter is rejected as waste [2]. This body of research aims to minimize the amount of brine water rejected. This should allow to reclaim more freshwater and monetize the salt produced, which should theoretically lower desalination costs. The research will focus on reclaiming desalination waste by dissolving pure CO₂, and by achieving so through its objectives. The five main objectives are:

1. Maximize Na⁺ removal. The main objective of this research is to study the absorption/reaction system inside a Bubble Column Reactor. This study will determine the best operating conditions that maximize sodium removal.
2. Maximize CO₂ absorption. The second objective is to determine the best operating conditions that maximize the amount of HCO₃⁻ formed. The amount of CO₂ dissolved affects the NaHCO₃ salt equilibrium and the amount of Na⁺ precipitated. This objective is considered a secondary objective, as it aids in achieving the first objective indirectly.
3. Maximize Cl⁻ removal. The third objective is to investigate Cl⁻ removal capacity using a CO₂-based hydrate system. The CO₂-hydrate system will be modeled using Machine Learning. The study will determine the factors important in achieving maximum Cl⁻ removal.
4. Model the Bubble Column Reactor. The fourth objective is to build a mathematical model that represents the important factors affecting the first three objectives. These factors are temperature, pressure, ionic interactions, and absorbent type and amount. This mathematical model will be important for understanding the process through simulation.
5. Optimizing operating conditions. The fifth and last objective of this study is to utilize the models built to achieve the first three objectives. The operating conditions of the Bubble Column Reactor are tuned using Non-Linear Programming and Machine Learning.

The novelty of this research relies on the mathematical framework used to develop an optimized solution. First, a mathematical model is developed to represent CO₂ absorption and salt precipitation in a Bubble Column Reactor. The multiphase phenomena modeled are gas absorption, aqueous reaction, and salt precipitation. These phenomena require modeling kinetic and thermodynamic reactions, and non-ideal behavior in gas and aqueous phases. Second, these models are optimized using Non-Linear Programming to maximize the amount of CO₂ absorbed and minimize the amount of material consumed. In addition, the process is optimized using Machine Learning to configure the best operating conditions for maximum chloride removal.

Desalination waste revitalization will aid in lowering the cost of desalination using ion removal by reclaiming the rejected water and utilizing salts precipitated. In the second chapter, a survey of the literature reviews current trends in treating reject brine, and in CO₂ capture using absorption. The third chapter covers the theoretical framework used to model CO₂ absorption and reaction in brine solutions. In addition, the third chapter includes the Machine Learning modeling framework. The fourth chapter discusses the results from model validation, simulation, and optimization. Finally, the last chapter concludes the research with a summary and three recommendations for future works.

Chapter 2: Literature Review

Zero Liquid Discharge (ZLD) is important not only for the environmental impact of brine, but also for the economic value it could add [32]. The technology aims to maximize the amount of freshwater reclaimed and minimize the amount of brine waste disposed of. It achieves so through a sustainable desalination solution [33]. Such a solution should aim to reuse any brine waste produced. If not feasible, it should aim to recycle brine and reduce waste disposed of. Beyond the three R's, a solution should recover any material from waste streams [34]. In addition, a solution should consider a balance between all the costs associated with water treatment, and all the benefits of minimizing the water supplied. Although current technologies are not economically viable for large-scale brine treatment, emerging solutions will aid in achieving so. In this chapter, brine treatment solutions are reviewed, along with carbon capture technologies through absorption.

2.1 Brine treatment technologies

There are various brine treatment technologies being developed. The conventional method of using thermal evaporation is very energy intensive; however, emerging technologies should make the process cheaper. Brine is usually characterized by having greater than 70 grams of Total Dissolved Solids (TDS). It mostly consists of sodium ions (Na^+) and chloride ions (Cl^-). Other major ions would be magnesium (Mg^{2+}), and sulfate (SO_4^{2-}), as shown in Figure 1. Some treatment technologies aim to remove certain ions, while other technologies work on reducing the total salinity of brine.

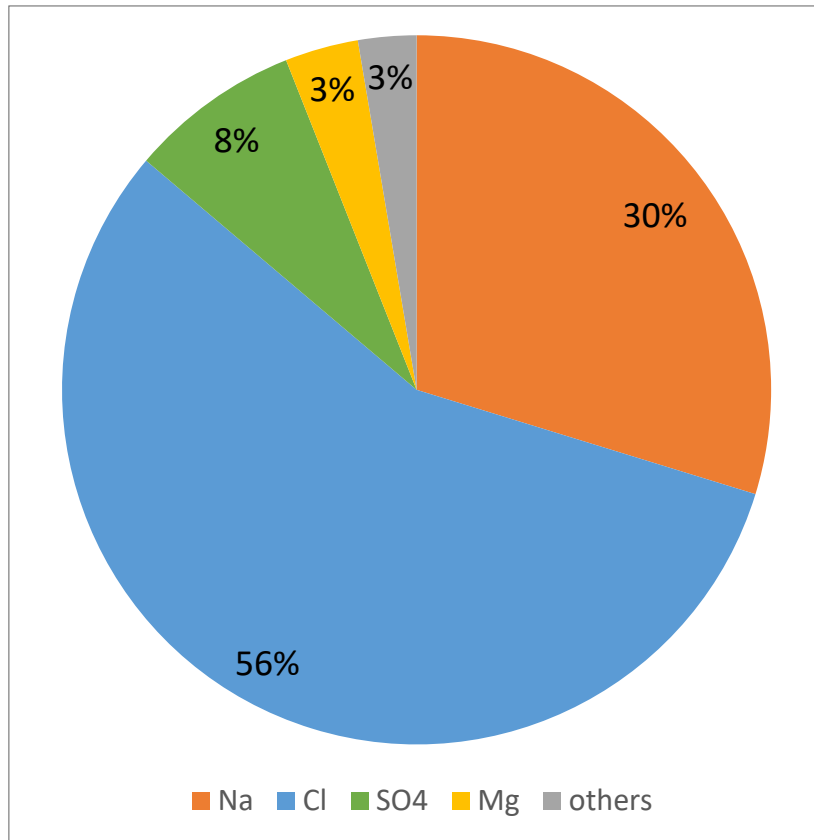


Figure 1: Main dissolved ions (wt%) in brine solutions obtained from a Multi-Stage Flash desalination unit in Abu Dhabi, UAE [35].

2.1.1 Conventional Method

The current conventional method for ZLD starts with a pretreatment unit before sending the effluent to a concentrator and a crystallizer [34]. Brines retentate coming from RO membranes contains about 70,000 ppm (mg/L) of TDS, as seen in Figure 2. The retentate, concentrated water is sent to a brine concentrator in which the brine is concentrated to 300,000 mg/L. The brine crystallizer evaporates the concentrate even further to recover most of the solids, and the final concentrate is sent to evaporation ponds for solid disposal [34]. The conventional method is only applied in certain applications, and to limited volumes of brine waste. This is due to the huge amounts of capital and operational costs associated with it [32].

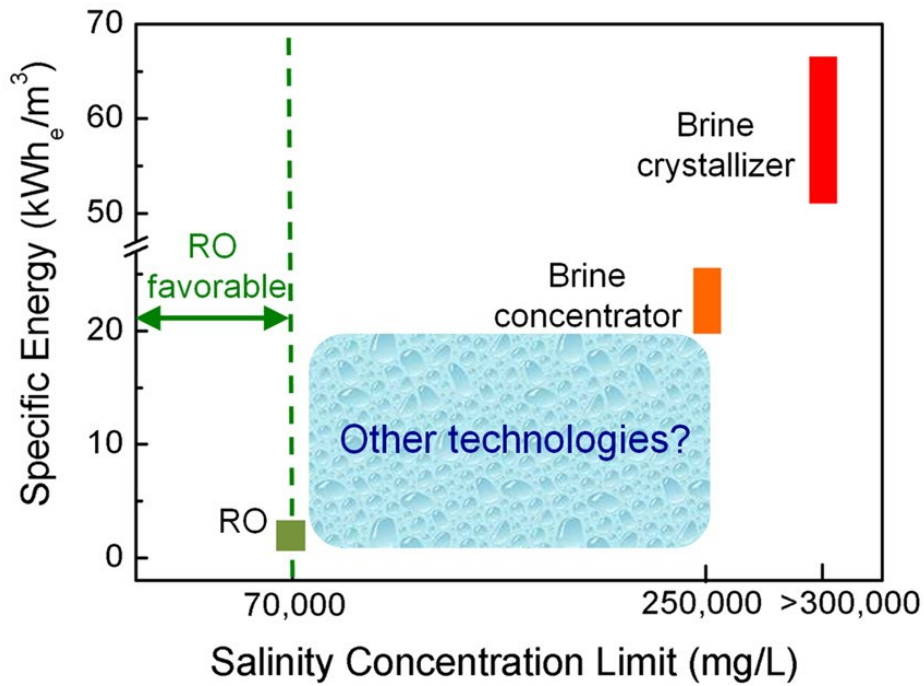


Figure 2: The desalination technology favorable at each salinity concentration [32].

2.1.1.1 Pretreatment

Pretreatment processes cover the chemical removal of scaling/fouling ions, such as calcium ions, sulfate ions, and silicon dioxide. The most common technique is to precipitate chemicals using a dosage of alkaline chemicals, such as NaOH, Na₂CO₃, or Ca(OH)₂ [36]. Another method is to use nucleation seeds, or pellets/inert-particles to induce precipitation. In addition, this allows salts to precipitate on pellets instead of producing a sludge that will be harder to separate. Chemical precipitation usually removes calcium ions, magnesium ions, and silicon dioxide. Carbonates can be removed through gas strippers using fluidized bed reactors by precipitating calcium carbonate (CaCO₃) [37]. Sulfates can precipitate, or be removed using ion-exchange resins [37], but new research focuses on using sulfate-reducing bacteria [38]. Sal-Proc was a product of all the research in this area, as it involves various steps in which ions precipitate one after another [39]. Other small-scale techniques are ion exchange units and electrocoagulation which have high operational and maintenance costs [40], [41]. Generally, pretreatment techniques are not used for their high capital and operational costs. These costs scale with the amount of water and TDS required to remove.

2.1.1.2 Brine Concentrator

Brine concentrators are evaporators that concentrate water without scaling or fouling the heat exchangers. Brine effluent coming out of Reverse Osmosis (RO) membrane units has a TDS of 70-90 g/L. These brine streams are heated to near 100°C, which allows the brine to be saturated with steam and start evaporating most of the water [42]. The final molality of such brine solutions reaches up to 300 g/L which is just below the solubility limit of NaCl [32]. Improvements to such exchangers allow them to operate at reduced temperatures and lower operating costs. A common type of brine concentrator is a falling film Brine Concentrator. It utilizes the characteristic of vertical falling films to saturate the solutions at around 50-70°C [43]. Another factor to consider is the turbidity occurring inside the heat exchanger, as salts can precipitate/suspend in the concentrator. Some of these common salts are Na₂SO₄, CaCO₃, and CaSO₄ [43].

2.1.1.3 Brine Crystallizer

Brine crystallizers are evaporators that operate at the NaCl solubility limit, 360 g/L TDS. Brine streams coming out of brine concentrators lose most of their water content. However, there is still some water left that needs to be evaporated in order to concentrate the NaCl solution. This can get tricky as scaling becomes an issue at these concentrations. The most common brine crystallizers are Multi-Stage-Flash Crystallizers and Membrane Distillation Crystallizers [32], [44]. The most important factor for crystallization effectiveness is the temperature of the crystallizer unit and the circulate flowrate. Energy intensification can be achieved by preheating the input flow and recovering valuable salts [44]. However, the operating costs hinder scaling such process for mass-scale.

2.1.2 Emerging technologies

Conventional Zero Liquid Discharge using thermal evaporation is an important step towards affordable ZLD technology. With current systems, RO is the best technology for desalination up to 70 g/L TDS, and brine concentrators are important for brines with TDS up to 250-300 g/L. Brine crystallizers are the only method for TDS above 300 g/L. Emerging technologies are fighting for the spot between 70 to 300 g/L

TDS range[32]. The best contenders in this space so far are Forward Osmosis [45] and Membrane Distillation [46]. The runners-ups are electrodialysis [47] and freeze distillation [48]. On that note, electrodialysis is associated with brackish water desalination more than it is associated with ZLD because of the energy costs of technology. Lately, electrodialysis has only been considered for ZLD, as the technology has become cheaper than Multi-Stage Flash for brine feeds [47].

2.1.2.1 Advanced Osmosis Systems

By far, the best emerging technology is osmotic systems, and Forward Osmosis (FO) is the most developed in that area, technology-wise. FO utilizes the osmotic pressure difference, or the concentration difference, to drive water across a semipermeable membrane, from the high-concentration side to the low-concentration side [49]. The emphasis is on the draw solution, or the concentrated solution. The most important characteristic of the draw solution is the ability to recover fresh water with low energy requirements [50]. The most widespread draw solution is an NH_3/CO_2 mixture, as it generates high osmotic pressures. It is easily recoverable because the NH_3 and CO_2 are easily volatile and evaporate at 60°C [50]. Another type of draw solution is using precipitable inorganics, such as sulfates. The approach is to saturate a solution with sulfates and draw fresh water in, so that small addition of these sulfates will precipitate them, and fresh water will be left [50]. Sulfuric acid is then added, and the draw solution is circulated back to the FO membrane. Currently, China's coal-to-chemicals plant in Zhejiang province uses lots of freshwater. FO units were used in their wastewater facilities to recover more than 90% of the water [32].

Reverse Osmosis for ZLD is in constant development alongside RO systems for desalination. RO for desalination focuses on developing membranes with better materials, or membranes that can withstand higher pressures. Reverse Osmosis for ZLD focuses on adding a secondary RO unit that can concentrate brine to even higher TDS values of 130 g/L [51]. To achieve so, RO membranes are usually accompanied with a pretreatment unit. These pretreatment units could be chemical-softening units or ion exchange units, as mentioned earlier. Low-Salt-Rejection Reverse Osmosis heightens the process, of a pretreatment followed by an RO unit, by having N stages [52]. The

retentate from the first RO unit is sent to the following RO unit in which the retentate keeps going until the final stage. The permeate from that RO unit is sent to the prior RO unit. This is possible when the total ΔP across all membranes is 70-80 bar. With all these advances, Reverse Osmosis will always be essential for desalination and ZLD systems.

Another approach to advancing osmosis systems is combining both RO and FO in what is called Osmotically Assisted Reverse Osmosis (OARO). In the unit, the brine-waste feed is pressurized through the OARO membrane. In addition, a draw/sweep solution is used to assist in driving water across the membrane, thus decreasing the pressure required [51]. Through a series of OARO stages, freshwater is transferred from a medium of high TDS to a medium of lower TDS (i.e., a series of dilutions). This explains why such systems can concentrate brines, while only recovering less than 50% of freshwater [51]. The technology is available commercially under different names: Cascading Osmotically Mediated Reverse Osmosis or Counter-Flow Reverse Osmosis [46]. The technology is still in its infancy with high energy/treatment costs compared to FO and RO costs.

2.1.2.2 Thermal Membrane Systems

Thermal membrane systems are another emerging technology aiming to drive freshwater across membranes using heat. Membrane Distillation (MD) is the most advanced technology in emerging technologies for ZLD, as it can treat brines with up to 350 g/L TDS [51]. It uses a temperature gradient to produce vapor pressure across a hydrophobic membrane. This allows only volatile components to pass with high recovery and more concentrate. There are various configurations, but Direct Contact Membrane Distillation (DCMD) is the most suitable for brine treatment. In DCMD, brine is heated to 40-80°C, and the overhead vapor transfers across the membrane, allowing for high rejection [51]. The cold side of the membrane condenses the vapor, allowing for high recovery. In practice, MD systems have low permeate flux compared to RO units [32], [51]. Although MD consumes less energy than Multi-Stage Flash, the technology is still in its infancy with research focusing on improving mass transfer rates and flux and controlling membrane wetting.

Alternatives to MD are Wind-Aided Intensified eVaporation (WAIV) and humidification and dehumidification systems. They are classified as thermal-based systems that are having some adoption. WAIV involves vertical wetted columns that use wind to evaporate brines [51]. WAIV systems are usually attached to RO units to handle concentrates, and column packing is used to enhance evaporation rates. These characteristics make WAIV a viable solution, as it has low energy requirements [53]. However, it has a high footprint, but still a lower footprint than evaporation ponds. Furthermore, humidification & dehumidification systems use air to carry water vapor and provide freshwater [54]. The humidifier heats the solution as air pass over to carry the water vapor. The dehumidifier condenses this saturated air to provide freshwater. Even with the drawbacks of high area requirements and low-scale application [55], it is used in treating highly contaminated wastewater in coal-fired power plants, in Shanghai [34]. These thermal systems provide an alternative to membrane distillation in specific applications where salinity is high, or where water recovery and water quality could be low.

2.1.2.3 Freeze Distillation

Another thermal-based approach that gained attention is Eutectic Freeze Crystallization. Instead of evaporating brines, Fernández-Torres et al. proposed freezing solutions as it requires 6-7 times less energy [48]. The technology freezes brine at various eutectic points to achieve freshwater with pure salts. A list of various salts and their eutectic points can be found in Table 1. In the process, brines are cooled down until the first eutectic point in which the salt precipitates and ice floats. The salt and ice are recovered as pure substances, and the remaining solution is cooled to the next eutectic point [48]. There is a large-scale plant in South Africa, reporting a treatment cost of \$1.42/m³ freshwater produced [56].

Table 1: List of salts and their respective eutectic point.

Salt	Eutectic point (°C)
Na₂SO₄	-1.27
NaHCO₃	-3.9
NaCl	-21.2

Hydrate-based desalination is another approach to freeze distillation. The technique traps water inside hydrated gases in what is called a clathrate [57]. The clathrate hydrate forms complex crystalline structures in water. The process usually operates at high pressures and low temperatures, but at temperatures much higher than the eutectic approach. CO₂ forms clathrate hydrates at 10°C and a pressure of 50 bars [58]. The higher the pressure, the better CO₂ is at capturing water into its clathrate hydrates. Attempts were made to combine CO₂ with other gases, such as CH₄ or other alkanes, so that the pressure required is lower. The research ended up focusing on cycloalkanes, specifically cyclopentane, as its clathrate has a lower density than water [59]. CO₂ hydrates had a higher density which made it hard to separate from precipitated salts [58]. Cyclopentane-hydrate provides the state of the art in hydrate distillation technology, even though the technology is still in its early phases.

2.1.3 Commercial Technologies

Commercial technologies refer to patented, sellable technologies. All the technologies mentioned below are just used on a small scale. There has been no attempt to use such in any other order. By far, the most widely used technology is the conventional method. Even the conventional method is used in limited cases with a limited scale of water. These technologies provide context on how the industry is tackling the problem at hand.

2.1.3.1 ARROW

Advanced Reject Recovery Of Water (ARROW) is a brine treatment method that can recover up to 95% of water [56]. ARROW has a configuration of a primary RO unit, followed by a secondary RO unit. It has a chemical softening step, but instead of having it between the membranes, it is the last step. In addition, the treated-water reject is recycled back to both RO units. The advantage of this method is a lower capital cost. The amount of treated water is reduced, allowing for smaller softening units. The disadvantage of such technology is that the process is very sensitive to the chemical precipitation step. Any changes to the number of hardness ions removed will increase the feed salinity and cause scaling [56]. There has been no report on energy requirements.

2.1.3.2 SPARRO

Slurry Precipitation And Recycle Reverse Osmosis (SPARRO) is a brine treatment technology used to recover up to 95% of water [60]. The technology was developed by the Chamber of Mines of South Africa Research Organization to treat scaling mine water [56]. It uses a tubular RO unit in which seed crystals are introduced to precipitate scaling compounds. In this case, these crystals serve as nucleation sites to precipitate CaSO_4 , instead of scaling the membrane [60]. The seeds are recoverable through a cyclone separator. The benefit of such a configuration is that the concentration factor can be higher than with regular RO membranes. The drawback is that seed crystals can clog membrane channels. There has been no report on energy requirements.

2.1.3.3 VESP

Vibratory Shear Enhanced Process (VSEP) is a brine treatment technology used to recover 75% to 92% of water [56]. It uses a plate-and-frame membrane in which a torsional oscillation is applied to the membrane's surface. These vibrations are sinusoidal and dampen within micrometers of the membrane surface. This produces a shear stress, in the fouling boundary layer, 10 times more than normal membranes [56]. The advantage of this vibratory motion is that it reduces scaling/fouling while using high operational flux. However, experimental results showed that scaling still occurred, yet acidic/basic chemical washing removed all scaling on membranes [56]. There has been no report on energy requirements.

2.1.4 Summary

The conventional method of using a concentrator followed by a crystallizer is the most common technology for achieving Zero Liquid Discharge. There has been some commercial application, yet nothing can replace the conventional method. Emerging technologies should provide a pathway toward ZLD. Whether through thermal-based solutions, or membrane-based technologies, ZLD will probably rely on the latter more than the former. Most research has been focused on the various types of osmosis systems, leaving behind older technologies such as chemical precipitation. This thesis

will focus on chemical precipitation, as more salt removal could be realized by using such technology.

2.2 Aqueous CO₂ Absorption

CO₂ absorption is important to examine as it is the second objective of this study, and it affects the first objective indirectly. Usually, CO₂ absorption is employed to remove acid gases for natural gas sweetening [61], or to remove CO₂ from flue gas streams for carbon capture [62]. In this study, CO₂ absorption is utilized to precipitate NaHCO₃. Generally, there are two important characteristics of an absorption column: the absorbent chemical [63] and the operating conditions of the column [64]. The chemical used can affect the absorption rate and the column capacity. In addition, energy requirements for absorption and desorption cycles will help in determining the best chemical to use. Studying the operating conditions of a column will aid in determining the optimal conditions for maximizing column capacity.

2.2.1 Chemical Absorbents

The most important characteristic of an absorption column is the absorbent used. Amines are the most surveyed chemical absorbents in literature [65], for CO₂ absorption. Lately, there has been a shift towards inorganic hydroxides and carbonates for Carbon Capture and Storage [66], [67]. The three absorbents surveyed in this study are: sodium hydroxide (NaOH), calcium hydroxide (Ca(OH)₂), and ammonia (NH₃), the simplest form from which all amines are derived. One important factor to examine is the mechanism of absorption, as this will aid in understanding the efficiency of the absorption. Another important factor to study is the column loading capacity, as this will be studied through the concentration of bicarbonate in the solution. Finally, the energy requirements of chemicals are studied through their absorption and regeneration costs.

2.2.1.1 Mechanism

It is important to examine CO₂ absorption mechanism, as it aids in understanding CO₂ absorption rate. The mechanism of CO₂ absorption depends on mass transfer resistance in the gas bubble, gas-liquid film, and bulk concentration of the base. For CO₂ absorption, there is no resistance in the gas bubble as the liquid film has most of the

resistance [68]. Resistance in the bulk liquid depends on the type of base used and its concentration. Depending on the pH region, CO₂ reaction with water is dominated by one of two reaction pathways: either to react with water or react with hydroxide ions. In the case of ammonia or amines, there is another pathway added in which an amine reacts with CO₂ through the zwitterion or tertiary mechanism [69]. This reaction decreases resistance in the bulk base at the expense of releasing bicarbonate later.

Hatta number [70] is what connects the resistance in the mass transfer with resistance in reaction rates. The optimal Hatta number should be around unity. For high Hatta numbers, mass transfer resistance can be decreased by using packed columns or fluidized beds, instead of a bubble column [71]. Both setups increase interfacial area either by increasing gas flowrate in fluidized beds, or by increasing gas-liquid contact in packed columns. Furthermore, the bulk concentration of the base increases by increasing the concentration of the base, or by increasing the strength of the base.

2.2.1.2 Column Loading

Column loading comes from either physical absorption or chemical absorption. Physical absorption is the transfer of CO₂ from the gas phase into the liquid phase. On the other hand, chemical absorption is the reaction of CO₂ with a base to form bicarbonate. Chemical absorption makes up most of the loading capacity because of the base used. In this study, bicarbonate loading is of major interest because any other carbonic form will hinder salt removal, specifically the sodium ions. Hydroxides should generate the most amount of bicarbonate; however, calcium hydroxide tends to precipitate calcium carbonate in low quantities [72]. Amines are very good at chemically absorbing CO₂, yet they cannot produce high amounts of bicarbonate ions. Amines react with CO₂ to form carbamate ions [73], an undesirable by-product, which minimizes the concentration of bicarbonates present. Hydroxide absorbents would do a better job because they do not involve the same side reactions as amines.

2.2.1.3 Absorption Costs

The energy costs of CO₂ absorption consist of the energy required for CO₂ hydration, and the energy used in side reactions with CO₂. The energy required for CO₂

hydration will be the same regardless of the chemical absorbents used. On the other hand, the energy consumed in the CO₂ reaction with the base will be different for each absorbent. Sodium reaction with CO₂ should not be considered because that is the intended consequence of this study, i.e. salt removal. However, the side reaction will be considered because some of the salt precipitated is from the base used and not from precipitating the salt in the brine solution. From Table 2, CO₂ reaction with hydroxides releases heat, while CO₂ reaction with amines requires heat. Sodium hydroxide is the best candidate in terms of absorption costs, but it would probably require the most amount of energy for separation from CO₂.

Table 2: List of absorbents and their enthalpy of absorption

Base	Energy Requirements for Absorption (kJ/kg of CO₂)	reference
NaOH	-2485	[74]
Ca(OH)₂	-2477	[74]
MEA	1930	[75]
NH₃ (2.5 wt.%)	1590	[75]
DEA	1590	[76]

2.2.1.4 Regeneration costs

Regeneration of absorbent is important for attaining a feasible absorption process. There are three possible ways for absorbent recovery: thermal regeneration, chemical substitution, or electrochemical regeneration. Thermal regeneration is quite common in amine-based absorption as it is the easiest method. According to one study, experts believe that amine regeneration costs for large-scale plants would be in the range of 3500-6000 kJ/kg of CO₂ absorbed [77]. The dual-alkali method, through the Solvay process [78], the hot potassium process [79], [80], and the modified Solvay process [35], [81], [82], is another form of regeneration through chemical substitution. The method utilizes two alkalis with different dissociation constants (pK_b) to absorb CO₂ and regenerate the more expensive alkali. The operating costs for such a process are mainly dependent on the cheap absorbent cost. The other possible way to regenerate absorbents is through electrochemical cells. One method [83] absorbs CO₂ using NaOH and regenerates the solution using a cell with a membrane that only passes positive ions (Na⁺). As the cell replaces Na⁺ with H⁺, the solution acidifies, and CO₂ can be collected

in a stripping unit. The regeneration costs for such an infant process are very high around 8500 kJ/kg-CO₂, but theoretically, the costs could be cut in half to 3730 kJ/kg-CO₂ [83]. All the absorbents discussed are summarized in Table 3.

Table 3: List of absorbents and their enthalpies of desorption

Base	Energy Requirements for Regeneration (kJ/kg of CO ₂)	reference
Amine	3500–6000	[77]
MEA (20 wt.%)	2955	[84]
NH₃	4000-4200	[85]
DEA	1630	[86]
NaOH	8500	[83]

2.2.2 Column Operating Conditions

Another important characteristic of an absorption column is absorption thermodynamics, and the thermodynamics can be studied through the operating conditions of the column, such as temperature, pressure, and electrochemistry. It is quite common for absorption columns to operate at high pressures and low temperatures, and vice versa for regeneration/stripping columns [87]. The reason being is that increased pressure enhances gas solubility, while increased temperature reduces the solubility [87]. In addition, high pressures and low temperatures favor forward reactions of CO₂ with absorbent, and vice versa for CO₂ dissociation reaction [88]. The heat of absorption is not affected by temperature and pressure changes [89]. However, it is important to operate at temperatures higher than and pressures lower than the condensation point [73]. It was reported that absorption can be enhanced by the electrolytic ions dissolved, and there are mainly two pathways where electrolysis can be applied. One application is seawater electrodialysis, in which Na⁺ and Cl⁻ ions are separated to obtain HCl and NaOH, where NaOH can then be used in carbon sequestration [90]. The other application is absorbent regeneration as mentioned above.

2.2.3 Summary

CO₂ absorption was studied to maximize NaHCO₃ precipitation by examining the effects of the absorbents and operating conditions of the column. The CO₂ absorbents studied were NH₃, NaOH, and Ca(OH)₂. Mechanism-wise, CO₂ absorption resistance is

mostly found in the gas-liquid film, and this resistance can be improved by either changing the column structure or the absorbent used. In addition, hydroxide absorbents have low absorption costs, while amines have lower regeneration costs. The dual-alkali process through Solvay or modified Solvay could be a better substitute than thermal regeneration. Finally, absorption columns should be operated at high pressures and low temperatures, but temperatures should be higher than and pressures should be lower than the condensation point. These conditions enhance solubility and favor forward reactions of CO₂ with absorbent.

Chapter 3: Research Methodology

This research focuses on answering the research objectives outlined in the introduction, maximizing sodium removal and maximizing the amount of CO₂ capture, along with maximizing chloride removal using CO₂-hydrate. The research aims to achieve so, through the modeling of a Bubble Column Reactor and optimizing the reactor's operating conditions. The modeling involves brine treatment using chemicals and gas absorption in aqueous media. Two phenomena occurring in the aqueous phase are gas absorption and various reactions. Gas absorption can be modeled using a mass transfer coefficient, or expanded through Computational Fluid Dynamics (CFD). Reactions modeled include aqueous complexation, acid-base reactions, and solid precipitation. In addition, the effective concentrations are accounted for through fugacity calculations for each phase. All these equations are compiled into what is known as reaction-diffusion-advection models. Finally, the chapter ends with the Machine Learning models used in the CO₂-hydrate system.

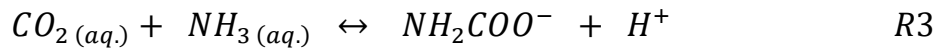
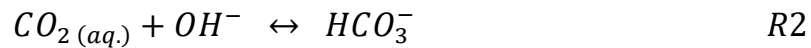
3.1 Reaction System Model

A complex reaction system will involve slow and fast reactions with non-ideal chemical interactions. Reaction modeling of complex systems requires solving equilibrium with kinetic equations simultaneously. In addition, chemical fugacity calculations should be included to account for the non-ideal behavior of aqueous species and gas mixtures. The following system of equations are compiled, and then, solved using Non-Linear Programming.

3.1.1 Kinetic Reactions

All chemical reactions are dynamic in nature and need to be modeled through chemical kinetics. The reaction rate constant is the determinant factor for modeling reaction dynamics. Slow reactions have a relatively small rate constant compared to fast reactions that have a higher rate constant [91]. In fact, the rates of fast and slow reactions can have orders of magnitude differences which can make reaction systems stiff, or harder to solve [92]. To address this issue, fast and slow reactions are often decoupled to speed up simulation time. In the case of a Bubble Column Reactor, there are only three

kinetics reactions: CO₂ absorption into aqueous media, CO₂ reaction with hydroxide ion, and CO₂ reaction with ammonia. There is various research focused on obtaining reaction rates for each reaction [93]. The first reaction is a phase transition equation that will be discussed in the Gas Absorption section of this chapter.



CO₂ reaction in basic media was reported by Pinsent et al. in 1956 along with ammonia kinetics [93], [94]. Over the years, there have been more supporting research to confirm the kinetic constant obtained from the original paper [95]. Research suggests that the hydroxide reaction is important only at high pH (pH>8.5) [96]. Carbon dioxide would then react with water to form carbonic acid which will then dissociate and form bicarbonate. Mechanism-wise, it is important to consider all pathways; however, bicarbonate production is the final and more important species to study. So, in the reaction system considered, only the hydroxide reaction is considered along with water dissociation thermodynamics to account for pH reaction dynamics. In addition, the backward reaction is accounted for through an equilibrium constant, which is a topic that will be discussed in the next section. The reaction rate constant [93] for the hydroxide reaction is listed in Equation 1.

$$\log k_{OH^-} = 13.635 - \left(\frac{2895}{T} \right) \quad (1)$$

As mentioned above, CO₂ reaction with ammonia was first reported by Pinsent et al, yet it was missing a reaction mechanism. Research suggests two pathways, either through zwitterion formation or by tertiary mechanism [69]. One theory suggests that a zwitterion forms when aqueous NH₃ combines with aqueous CO₂ [97]. A zwitterion is a molecule with a positive charge around the nitrogen atom and a negative charge around the oxygen atom. The zwitterion molecule of Carbamic acid is shown in Figure 3. The zwitterion formed can be easily attacked by a base to form a carbamate ion (NH₂COO⁻) [69]. The other theory suggests that a base deprotonates NH₃, while NH₃ is attacking

CO₂ in a three-way interaction. This instantly forms carbamate through the tertiary mechanism [98]. Both pathways suggest that the ammonia reaction is not elementary, and that the concentration of ammonia needs to be raised to a power greater than one in the reaction rate law. This can be evaluated by fitting experimental data to the reaction rate [99]. This can be avoided by using the original formulation and accounting for n through ionic activity. The reaction rate constant for the ammonia reaction is listed in Equation 2. Also, the backward reaction was reported later in the literature [100] and mentioned in Equation 3.

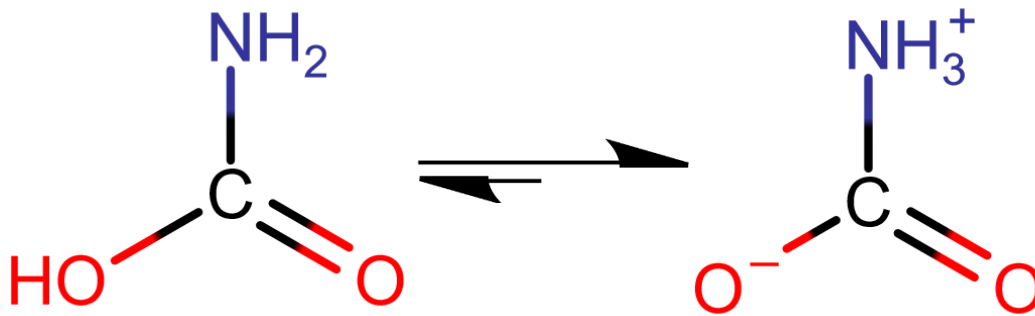


Figure 3: Zwitterion form of carbamic acid.

$$r_{NH_3} = C_{CO_2} \cdot C_{NH_3}^n \cdot k_{NH_3} \rightarrow r_{NH_3} = \gamma_{CO_2} C_{CO_2} \cdot \gamma_{NH_3} C_{NH_3} \cdot k_{NH_3}$$

$$\log k_{NH_3} = 11.13 - \left(\frac{2530}{T} \right) \quad (2)$$

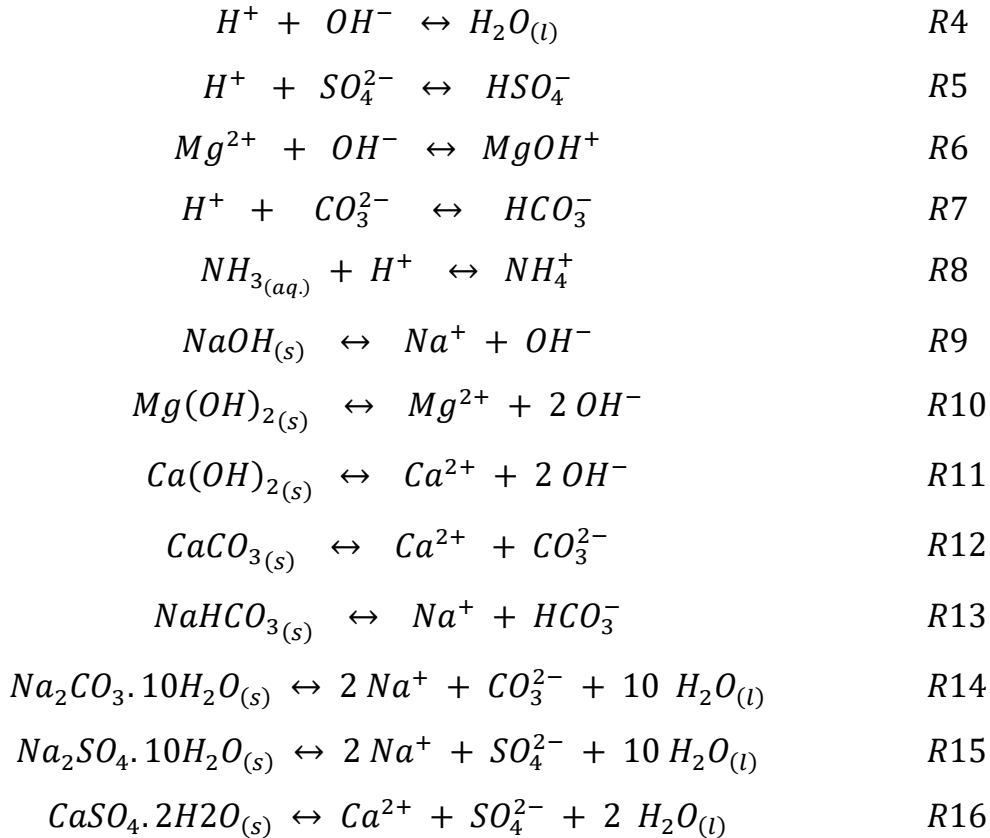
$$\log k_{NH_2COO^-} = 1.03 * 10^{19} * \exp\left(-\frac{16180}{RT}\right) \quad (3)$$

$$\text{where } R = 1.9872 \frac{\text{cal}}{\text{mol K}}$$

3.1.2 Thermodynamic Reactions

Most reactions in the bubble column are considered fast, which are easier to be modeled through equilibrium equations [91]. There are two types of equilibrium reactions: aqueous homogeneous reactions, and precipitation heterogeneous reactions [101]. Homogenous reactions are characterized by having all the species in the equilibrium equation, while heterogenous only consider aqueous species in an inequality. The reaction quotient for heterogenous reactions is an inequality rather than an equation because aqueous species are limited by the corresponding solubility limit [102]. Also,

the inequality is considered only when aqueous species exceed the solubility limit or when the solid species is present. Gibbs free energy (G_f^o) is the thermodynamic property used to calculate equilibrium constant for each reaction. All the thermodynamic reactions considered are listed below (R4 – R16).



3.1.2.1 Water Model: IAPWS

The model used to calculate the G_f^o of water was developed by the International Association for Properties of Water and Steam (IAPWS). There are two relevant models: the IAPWS-95 [103] and IAPWS-IF97 [104] models. The IAPWS-95 model uses temperature and density as its input parameters, while the IAPWS-97 model is non-continuous with temperature and pressure as input parameters. The later model was used as it uses temperature and pressure without sacrificing much accuracy. Also, the IAPWS-97 is used as it has an equation for calculating Gibbs free energy, Equation 4. Another issue to point out is that the G_f^o values obtained by the model are in fact the changes in the value from 25°C (ΔG_f^o), and not the actual value at a particular

temperature (G_f^o). This is remediated by using the standard Gibbs free energy of formation data obtained from JANAF thermochemical tables [105], Equation 5.

$$\frac{G(P, T)}{R T} = \sum_{i=1}^{34} n_i \left(7.1 - \frac{P}{P^*}\right)^{I_i} \cdot \left(\frac{T^*}{T} - 1.222\right)^{J_i} \quad (4)$$

$$\text{where: } P^* = 16.53 \text{ MPa, } T^* = 1386 \text{ K, } R = 0.461526 \frac{\text{kJ}}{\text{kg K}}$$

The list of equations used is tabulated below and the code used can be found in A3.1. All parameters used in Equation 4 can be found in Table 4.

Table 4: Parameters (I_i, J_i, n_i) used in the Gibbs free energy formulation of water, Equation 4.

i	I_i	J_i	n_i	i	I_i	J_i	n_i
1	0	-2	1.4632971213E-01	18	2	3	-4.4141845330846E-06
2	0	-1	-8.4548187169E-01	19	2	17	-7.2694996297594E-16
3	0	0	-3.7563603672E+00	20	3	-4	-3.1679644845054E-05
4	0	1	3.3855169168E+00	21	3	0	-2.8270797985312E-06
5	0	2	-9.5791963388E-01	22	3	6	-8.5205128120103E-10
6	0	3	1.5772038513E-01	23	4	-5	-2.2425281908000E-06
7	0	4	-1.6616417200E-02	24	4	-2	-6.5171222895601E-07
8	0	5	8.1214629984E-04	25	4	10	-1.4341729937924E-13
9	1	-9	2.8319080124E-04	26	5	-8	-4.0516996860117E-07
10	1	-7	-6.0706301566E-04	27	8	-11	-1.2734301741641E-09
11	1	-1	-1.8990068218E-02	28	8	-6	-1.7424871230634E-10
12	1	0	-3.2529748771E-02	29	21	-29	-6.8762131295531E-19
13	1	1	-2.1841717175E-02	30	23	-31	1.4478307828521E-20
14	1	3	-5.2838357970E-05	31	29	-38	2.6335781662795E-23
15	2	-3	-4.7184321073E-04	32	30	-39	-1.1947622640071E-23
16	2	0	-3.0001780793E-04	33	31	-40	1.8228094581404E-24
17	2	1	4.7661393907E-05	34	32	-41	-9.3537087292458E-26

$$G_{f, H_2O(l)}^o(P, T) = -237.141 \text{ kJ} + (G(P, T) - G(P = 1, T = 298.15)) \quad (5)$$

3.1.2.2 Electrolyte model: HKFT

In 1981, Helgeson-Kirkham-Flowers developed an Equation of State for calculating the thermodynamic properties of aqueous ions for Geological applications [106]. In 1988, Helgeson and Tanger expanded the temperature and pressure ranges of the model, and substantially improved accuracy for higher temperatures and pressures [107]. In 1992, the HKFT model was recompiled into the form it is today. This allowed electrolytes and non-electrolytes to be calculated using the same equations [108]. Currently, the model utilizes seven parameters to calculate thermodynamic properties between 0-1000°C and between 1-5000 bar, Equation 6. The only other way to calculate G_f^o is through evaluating equilibrium constants for each aqueous reaction, which can be quite cumbersome and can be inaccurate. The equations used in the HKFT model are listed beneath Equation 6 and the code used can be found in in A3.2.

$$G_f^o(T, P) = G_{ref}^o + G_{non-solvation} + G_{solvation} \quad (6)$$

$$G_{non-solvation} = -S_{ref} * (T - T_{ref}) - c_1 * \left(T * \ln\left(\frac{T}{T_{ref}}\right) + T_{ref} - T \right)$$

$$+ a_1 * (P - P_{ref}) + a_2 * \ln\left(\frac{P + \Psi}{P_{ref} + \Psi}\right) * \frac{P + \Psi}{P_{ref} + \Psi}$$

$$+ a_3 * \frac{P - P_{ref}}{T - \Theta} + \frac{a_4}{T - \Theta} * \ln\left(\frac{P + \Psi}{P_{ref} + \Psi}\right)$$

$$- c_2 \left\{ \frac{\Theta - T}{\Theta} * \left[\frac{1}{T - \Theta} - \frac{1}{T_{ref} - \Theta} \right] - \frac{T}{\Theta^2} * \ln\left(\frac{T_{ref}(T - \Theta)}{T(T_{ref} - \Theta)}\right) \right\}$$

$$G_{solvation} = w_j * \left(\frac{1}{\epsilon} - 1 \right) - w_{ref} * \left(\frac{1}{\epsilon_{ref}} - 1 \right) + w_{ref} * Y_{ref} * (T - T_{ref})$$

where:

$$T_{ref} = 298.15 \text{ K}, \quad P_{ref} = 1 \text{ bar}, \quad \Psi = 2600 \text{ bar}, \quad \Theta = 228 \text{ K}$$

$$w_j = n * \left(\frac{Z_j^2}{r_{e,j}} - \frac{Z_j}{3.082 + g} \right)$$

$$r_{e,j} = r_{e,j,ref} + Z_j * g$$

$$r_{e,j,ref} = Z_j^2 / \left(\frac{w_{ref}}{n} + \frac{Z_j}{3.082} \right)$$

where Z_j is charge of species

$$n = \frac{N_A * e^2}{2} * 2.39 \quad \text{where } N_A = 6.02252 * 10^{23} \quad \& \quad e = 4.80298 * 10^{-10}$$

$$g(\rho_w, T) = a_g * (1.0 - \rho_w)^{b_g} - f$$

$$\text{where: } a_g = \sum_{i=0}^2 a_{g,i} * T^i, \quad b_g = \sum_{i=0}^2 b_{g,i} * T^i$$

$$f = \left[\left(\frac{T - 155}{300} \right)^{4.8} + f_{g,0} \left(\frac{T - 155}{300} \right)^{16} \right] [f_{g,1}(1000 - P)^3 + f_{g,2}(1000 - P)^4]$$

for $g(\rho_w, T)$ function, T is in deg C & ρ_w is in $\frac{gm}{cm^3}$

Table 5 contains the parameters used in the g function, and Table 6 contains all the parameters used to calculate G_f^o for dissolved species in the reaction system.

Table 5: Parameters ($a_{g,i}, b_{g,i}, f_{g,i}$) used in the $g(\rho_w, T)$ function

i	$a_{g,i}$	$b_{g,i}$	$f_{g,i}$
0	-2.037662	6.107361	3.666666e1
1	5.747000E-3	-1.074377E-2	-1.504956E-8
2	-6.557892E-6	1.268348E-5	5.017997E-12

Table 6: List of species and their corresponding parameters for the HKFT model

Species	G_{ref}^o	S_{ref}	a_1	a_2	a_3	a_4	c_1	c_2	w_{ref}
H	0	0	0	0	0	0	0	0	0
OH	-157297	-10.7110	0.5241296	30.87792	7.708183	-116403	17.3636	-432876	721572
Na	-261881	58.40864	0.769438	-956.044	13.6231	-114056	76.06512	-124725	138323
K	-282462	101.0436	1.489086	-616.303	22.74004	-113470	30.9616	-74935.4	80625.68
Mg	-453985	-138.072	-0.3438	-3597.82	35.10376	-99997.6	87.0272	-246521	643164.5
Ca	-552790	-56.484	-0.08146	-3034.24	22.16097	-103730	37.656	-105520	517393.4
HCO ₃	-586940	98.44952	3.163983	481.3692	5.165566	-118265	54.13887	-199071	532748.7
CO ₃	-527983	-49.9988	1.193444	-1667.07	26.83701	-109382	-13.8934	-719301	1418962
SO ₄	-744459	18.828	3.473306	-830.357	-25.9918	-112842	6.86176	-753036	1316412
CO ₂ (aq)	-385974	117.5704	6.400014	-4199.98	-232	310000.9	154.0001	150000.6	-31070
NH ₃ (aq)	-26706.5	107.8217	2.130116	1170.265	36.08616	-121110	84.9352	-48952.8	-20920
NH ₄	-79454.2	111.1689	1.621844	981.0643	35.81713	-120328	73.0108	-878.64	62843.68

The HKFT model [108] uses seven parameters c_1 & c_2 are only temperature dependent, a_1 & a_2 are both pressure dependent, while a_3 & a_4 are both temperature and pressure dependent. The final parameter w_{ref} depends on the Born radius of the ion and its solvation thermodynamics. G_f^o calculation depends on the non-solvation energy needed to dissolve species in solution, and the electrostatic energy required to interact in aqueous media. An important factor to consider is that the HKFT model requires the evaluation dielectric constant of water, which is accounted for in the terms:

$$\frac{1}{\epsilon}, \frac{1}{\epsilon_{ref}}, \text{ and } Y_{ref}.$$

IAPWS recommends using the ‘*formulation for static permittivity*,’ developed by Fernández et al. [109], for calculating the dielectric constant (ϵ) of water, Equation 7. A factor to consider is that the dielectric constant of water is a function of the density of water. The molar density is the inverse of the molar volume. The molar volume of water (V_w) was calculated by taking the derivate of the Gibbs free energy, from the IAPWS-97 model, with respect to pressure at a constant temperature. This dielectric constant can be

plugged in the G_f^o from the HKFT model to evaluate G_f^o for any dissolved species in water. The code used can be found in in A3.3.

$$\epsilon(\rho, T) = \frac{1 + A + 5B + \sqrt{9 + 2A + 18B + A^2 + 10AB + 9B^2}}{4 - 4B} \quad (7)$$

where:

$$A = \frac{N_A * \mu^2}{\epsilon_0 * k} * \frac{\rho * g}{T} \quad \& \quad B = \frac{N_A * \alpha}{3\epsilon_0} * \rho$$

$$\epsilon_0 = (4 * 10^{-7} * \pi * 299792458^2)^{-1} \quad \& \quad k = 1.380658 * 10^{-23}$$

$$N_A = 6.0221367 * 10^{23} \quad \& \quad \alpha = 1.636 * 10^{-40} \quad \& \quad \mu = 6.138 * 10^{-30}$$

$$g(\rho, T) = 1 + \sum_{k=i}^{11} N_k * \left(\frac{\rho}{\rho_c}\right)^i * \left(\frac{T_c}{T} - 1\right)^j + N_{12} * \left(\frac{\rho}{\rho_c}\right) * \left(\frac{T}{228} - 1\right)^{-1.2}$$

$$\text{where: } T_c = 647.096 \text{ K} \quad \& \quad \rho_c = \frac{322 \frac{kg}{m^3}}{MW_w} \quad \& \quad MW_w = 0.018015268$$

$$\epsilon_{ref} = \epsilon(\rho(P_{ref}, T_{ref}), T_{ref}) \quad \& \quad Y_{ref} = \frac{1}{\epsilon_{ref}} * \left. \frac{\partial \epsilon}{\partial T} \right|_{P=P_{ref}}$$

$$\rho_w = \frac{1}{V_w} \quad \& \quad V_w = \left. \frac{\partial g}{\partial P} \right|_T$$

Table 7 contains the parameters used in the g function.

Table 7: Parameters (N_i , i , j) used in the $g(\rho, T)$ function.

k	N_k	i	j	k	N_k	i	j
1	0.978224486826	1	0.25	7	0.949327488264E-1	4	2
2	-0.957771379375	1	1	8	-0.980469816509E-2	5	2
3	0.237511794148	1	2.5	9	0.165167634970E-4	6	5
4	0.714692244396	2	1.5	10	0.937359795772E-4	7	0.5
5	-0.298217036956	3	1.5	11	-0.123179218720E-9	10	10
6	-0.108863472196	3	2.5	12	0.196096504426E-2	1	1.2

3.1.2.3 Gases and Solids: Heat Capacity Method

There is no specific model to obtain thermodynamic properties for solids and gases. A common method to calculate G_f^o at any temperature and pressure is to use heat capacity and molar volume. Heat capacity power functions account for changes in temperature and molar volume accounts for changes in pressure. One common heat capacity function is the Maier-Kelly heat capacity power function [110], Equation 8. There is data to calculate G_f^o for gases, Equation 9, and simple inorganic salts, Equation 10. For complex salts with multiple cations and anions, there is a group contribution method that can estimate G_f^o using heat capacity power functions [111], [112], Equation 11 and 12 respectively. The code used can be found in in A3.3 and A3.4, for gas calculations and solid calculations, respectively.

$$C_{p,maier-kelly}(T) = a + b * T + \frac{c}{T^2} \quad (8)$$

$$G_{f, gas}^o(T, P) = \Delta H + \Delta(T S) \quad (9)$$

$$\Delta H = \int_{T_{ref}}^T Cp dT = a * (T - T_{ref}) + b * \frac{T^2 - T_{ref}^2}{2} - c * \left(\frac{1}{T} - \frac{1}{T_{ref}} \right)$$

$$\Delta(T S) = T * S^o - T_{ref} * S_{ref}^o$$

$$S^o = a * \ln\left(\frac{T}{T_{ref}}\right) + b * (T - T_{ref}) - \frac{c}{2} * \left(\frac{1}{T^2} - \frac{1}{T_{ref}^2} \right) - R * \ln\left(\frac{P}{P_{ref}}\right)$$

$$G_{f, salt}^o = G_{ref}^o - S_{ref}^o (T - T_{ref}) + a * \left(T - T_{ref} - T \ln\left(\frac{T}{T_{ref}}\right) \right) + \quad (10)$$

$$b * \left(T * T_{ref} - \frac{T^2}{2} - \frac{T_{ref}^2}{2} \right) + c * \left(\frac{1}{T_{ref}} - \frac{1}{2T} - \frac{T}{2T_{ref}^2} \right) + V * \Delta P$$

$$C_{p,multi-salt}(T) = a + b * T + \frac{c}{T^2} + d * T^2 \quad (11)$$

The $G_{f, multi-salt}^o$ uses $C_{p, multi-salt}$ to calculate changes in G_f^o in complex salts.

$$\begin{aligned}
G_{f, multi-salt}^o = & G_{ref}^o - S_{ref}^o (T - T_{ref}) + a \\
& * \left(T - T_{ref} - T \ln \left(\frac{T}{T_{ref}} \right) \right) + \\
b * & \left(T * T_{ref} - \frac{T^2}{2} - \frac{T_{ref}^2}{2} \right) + c * \left(\frac{1}{T_{ref}} - \frac{1}{2T} - \frac{T}{2T_{ref}^2} \right) + \\
& d * \left(\frac{T^3}{6} - T * \frac{T_{ref}^2}{2} + \frac{T_{ref}^3}{3} \right)
\end{aligned} \tag{12}$$

where $T_{ref} = 298.15 \text{ K}$, $P_{ref} = 1 \text{ bar}$, $R = 8.314462 \frac{\text{J}}{\text{mol K}}$

Table 8 contains the gases and simple solids and all the parameters used to calculate G_f^o , while Table 9 contains the parameters used to calculate G_f^o of complex solids using specific groups.

Table 8: List of solids and gases, and their respective parameters for Gibbs free energy calculation

Species	G_{ref}^o	S_{ref}^o	V	a	b	c
CO _{2(g)}	-394358.736	213.73964	94	44.22488	8.7864	-8.61904
Mg(OH) ₂	-831992	59.428	24.63	102.20	15.11	-26.17
Ca(OH) ₂	-896887	83.390	33.056	89.26	33.11	-10.36
CaCO ₃	-1129177.92	92.6756	104.51632	21.92416	-25.94080	36.934
NaHCO ₃	-851156	101.936	38.080	87.61	0	0
Na ₂ SO ₄ :10H ₂ O	-3646334	591.900	219.180	574.46	0	0
Na ₂ CO ₃ :10H ₂ O	-3427945	564.710	0	550.32	0	0
CaSO ₄ :2H ₂ O	-1797387	193.930	74.690	186.20	0	0

Table 9: List of specific groups used to calculate Gibbs free energy for complex solids

Species	G_{ref}^o	a	b	c	d
Na	-199.801	14.186	9.665	0.529	4.851
K	-211.713	25.309	-2.284	0.218	5.174
Mg	-372.414	14.639	-0.637	-0.074	-0.609
Ca	-432.414	20.470	-6.225	-0.026	-3.219
NH ₄	-53.199	4.205	116.120	1.206	2.166
OH	-230.428	28.917	30.730	-0.628	3.257
HCO ₃	-643.288	26.758	138.905	-0.373	-5.013
CO ₃	-635.990	47.278	86.757	-0.887	-5.133
SO ₄	-795.046	85.866	52.357	-1.925	-0.047
H ₂ O	-244.317	15.458	66.593	0.470	-40.518

3.1.3 Fugacity and Activity

Fugacity is the measure of how effectively a set of molecules interact in a given system. For gases, fugacity is a measure of the effective partial pressure, as it indicates the real/effective interaction in a mixture of gases. For liquids, the fugacity of a species is a measure of the real concentration and its activity/effectiveness. The fugacity/activity of a solid refers to its interaction in a solid solution to form specific lattice structures. In another way, fugacity is a departure from ideality. As pressure increases, pure gases and gas mixtures become non-ideal, while solutions with high concentrations become non-ideal. For solids, non-ideality appears when solid solutions form new crystalline structure that differs from the respective pure solid structures.

3.1.3.1 Gas Phase: Peng Robinson

Any Equation of State (EOS) for gases will relate the pressure of a gas with its temperature and molar volume. There are many equations that describe real gas behavior, and among the most accurate are Peng-Robinson EOS (PR) [113], and its precedent, Soave-Redlich-Kwong EOS (SRK) [114]. PR-EOS has a slightly more accurate estimation of thermodynamic properties near the critical point, yet there is a use case for each EOS. PR-EOS is better for condensate systems, while SRK is better for

polar solutions [115]. However, this is not a distinctive factor for the system being developed, as the EOS chosen will be used for the gas phase only. PR-EOS was chosen for its more accurate gas-phase behavior prediction near the critical point. The gas calculations were made using Equation 13 and the code used can be found in A3.5.

$$\ln(\phi) = Z - 1 - \ln(Z - B) - \frac{A}{\ln(8) * B} * \ln\left(\frac{Z + (1 + \sqrt{2}) * B}{Z + (1 - \sqrt{2}) * B}\right) \quad (13)$$

$$\text{where: } Z^3 + (B - 1) * Z^2 + (A - 3B^2 - 2B) * Z + (B^2 + B^3 - AB) = 0$$

$$A = \frac{a * \alpha * P}{R^2 T^2} \quad \& \quad B = \frac{b * P}{R T}$$

$$a = \frac{0.457235 * R^2 * T_c^2}{P_c} \quad \& \quad b = \frac{0.0777961 * R * T_c}{P_c}$$

$$\alpha = \left(1 + k \left(1 - \sqrt{\frac{T}{T_c}} \right) \right)^2 \quad \& \quad k = 0.37464 + 1.54226 * w - 0.26992 * w^2$$

$$\text{where: } R = 8.314462 * 10^{-5} \frac{m^3 * bar}{K mol}$$

3.1.3.2 Aqueous Phase: Pitzer Model

At infinite dilution, aqueous solutions behave ideally, but as species concentration starts to increase, solutions need to be approximated using an activity model. Below 0.1M, aqueous solutions can be modeled through the standard Debye-Hückel (D-H) equation [116], or through the extended form of the equation for slightly higher concentrations. Beyond the D-H equation, there are many equations attempting to extend the model to concentrated solutions. Among the few are Bromley's equation [117], Specific-ion Interaction Theory (SIT) [118]–[120], and Pitzer model [121], [122]. Bromley's equation is the easiest to fit, while Pitzer model uses rigorous thermodynamics and can model three-ion interactions. SIT sits in between in terms of modeling complexity.

Another approach to modeling excess Gibbs free energy in liquid mixtures is the Universal Quasi-Chemical equation, or for UNIQUAC short. The model uses statistical

thermodynamics to approximate molecular-level interactions. The model combines a combinatorial term (entropic term) with a residual term (enthalpic term). The model was very successful, as it was expanded to various applications, such as UNIFAC [123], LIQUAC [124], and extended UNIQUAC [125]. Mixed Solvent Electrolyte (MSE) developed by OLI Systems [126] is the most accurate with no limitation on ionic strength. MSE is extremely accurate with seldom any public data, while the Pitzer model provides relatively high accuracy with huge amounts of public data. In fact, it is characterized by having the most amount of research/data, yet the model works within certain temperatures and chemical concentration ranges, depending on each interaction coefficient. The Pitzer model, specified in Equation 14-17, has different equations for water, cations, anions, and neutral species. The parameters $B^0, B^1, B^2, C^0, \theta$, and λ used in two-ion interactions are listed in appendix A1.1 along with the references used. In addition, parameters ψ, ξ , and μ used in three-ion interactions are listed in appendix A1.2. The code used for the Pitzer equations can be found in in A3.6.

$$\begin{aligned}
 \ln(\gamma_w) &= \frac{\sum m_i * MW_w}{1000} * \phi & (14) \\
 (\phi - 1) &= \frac{2}{\sum m_i} * \left\{ -\frac{A^\phi \sqrt{I}^3}{1 + b\sqrt{I}} + \sum_c \sum_a m_c m_a (B_{ca}^\phi + ZC_{ca}) + \right. \\
 &\sum_c \sum_{c'} m_c m_{c'} \left(\Phi_{cc'}^\phi + \sum_a m_a \psi_{cc'a} \right) + \sum_a \sum_{a'} m_a m_{a'} \left(\Phi_{aa'}^\phi + \sum_c m_c \psi_{caa'} \right) + \\
 &\sum_n \sum_c m_n m_c \lambda_{nc} + \sum_n \sum_a m_n m_a \lambda_{na} + \sum_n \sum_c \sum_a m_n m_c m_a \xi_{nca} + \\
 &\left. 3 \sum_n \sum_{n'} \sum_c m_n m_{n'} m_c \mu_{nn'c} + 3 \sum_n \sum_{n'} \sum_a m_n m_{n'} m_a \mu_{nn'a} \right\} \\
 \ln(\gamma_M) &= z_M^2 F + \sum_a m_a (2B_{Ma} + ZC_{Ma}) \\
 &+ \sum_c m_c \left(2\Phi_{Mc} + \sum_a m_a \psi_{Mca} \right) + & (15)
 \end{aligned}$$

$$\begin{aligned}
& \sum_a \sum_{a'} m_a m_{a'} \psi_{Maa'} + |z_M| \sum_c \sum_a m_c m_a C_{ca} + \\
& 2 \sum_n m_n \lambda_{nM} + \sum_n \sum_a m_n m_a \xi_{nMa} + 6 \sum_n \sum_{n'} m_n m_{n'} \mu_{nn'M} \\
\ln(\gamma_X) = & z_X^2 F + \sum_c m_c (2B_{cX} + ZC_{cX}) + \sum_a m_a \left(2\Phi_{aX} + \sum_c m_c \psi_{caX} \right) + \\
& \sum_c \sum_{c'} m_c m_{c'} \psi_{cc'X} + |z_M| \sum_c \sum_a m_c m_a C_{ca} + \\
& 2 \sum_n m_n \lambda_{nX} + \sum_n \sum_c m_n m_c \xi_{ncX} + 6 \sum_n \sum_{n'} m_n m_{n'} \mu_{nn'X}
\end{aligned} \tag{16}$$

$$\begin{aligned}
\ln(\gamma_N) = & 2 \sum_c m_c \lambda_{Nc} + 2 \sum_a m_a \lambda_{Na} + 2 \sum_n m_n \lambda_{Nn} + \sum_c \sum_a m_c m_a \xi_{Nca} \\
& + 6 \sum_n \sum_c m_n m_c \mu_{Nnc} + 6 \sum_n \sum_a m_n m_a \mu_{Nna}
\end{aligned} \tag{17}$$

In which Equations 15 and 16 use F to be:

$$\begin{aligned}
F = & -A^\phi \left(\frac{\sqrt{I}}{1 + b\sqrt{I}} + \frac{2}{b} \ln(1 + b\sqrt{I}) \right) + \sum_c \sum_a m_c m_a B'_{ca} + \\
& \sum_c \sum_{c'} m_c m_{c'} \Phi'_{cc'} + \sum_a \sum_{a'} m_a m_{a'} \Phi'_{aa'}
\end{aligned}$$

where:

$$\mathbf{b} = 1.2 \quad \& \quad A^\phi(\rho, \epsilon, T) = \frac{1}{3} \left(\frac{2 * \pi * \rho * N_A}{1000} \right)^{1/2} * \left(\frac{100 * e^2}{(4 * \pi * \epsilon_0) * \epsilon * k * T} \right)^{3/2}$$

$$N_A = 6.0221367 * 10^{23} \quad \& \quad e = 1.6021773 * 10^{-19}$$

$$\epsilon_0 = 8.8541878 * 10^{-12} \quad \& \quad k = 1.380658 * 10^{-23}$$

$$I = \frac{1}{2} \sum m_i z_i^2 \quad \& \quad Z = \sum m_i |z_i|$$

$$B_{mx}^\phi = B^0 + B^1 * \exp(-\alpha_1 \sqrt{I}) + B^2 * \exp(-\alpha_2 \sqrt{I})$$

$$B_{mx} = B^0 + B^1 * g(\alpha_1 \sqrt{I}) + B^2 * g(\alpha_2 \sqrt{I})$$

$$B'_{mx} = [B^1 * g'(\alpha_1 \sqrt{I}) + B^2 * g'(\alpha_2 \sqrt{I})]/I$$

$$g(x) = \frac{2}{x^2} [1 - (1 + x) \exp(-x)]$$

$$g'(x) = \frac{-2}{x^2} \left[1 - \left(1 + x + \frac{x^2}{2} \right) \exp(-x) \right]$$

$$\begin{cases} \text{if } B^2 = 0, & \alpha_1 = 2, \alpha_2 = 0 \\ \text{otherwise} & \alpha_1 = 1.4, \alpha_2 = 12 \end{cases}$$

$$C_{mx} = \frac{C^0}{2\sqrt{|z_m z_x|}}$$

$$\Phi_{ij}^\phi = \theta_{ij} + {}^E\theta_{ij} + I * {}^E\theta'_{ij}$$

$$\Phi_{ij} = \theta_{ij} + {}^E\theta_{ij}$$

$$\Phi_{ij}' = {}^E\theta'_{ij}$$

$${}^E\theta_{ij} = \frac{z_i z_j}{4I} \left(J_0(x_{ij}) - \frac{J_0(x_{ii})}{2} - \frac{J_0(x_{jj})}{2} \right)$$

$${}^E\theta'_{ij} = \frac{z_i z_j}{8I^2} \left(x_{ij} * J_1(x_{ij}) - \frac{x_{ii} * J_1(x_{ii})}{2} - \frac{x_{jj} * J_1(x_{jj})}{2} \right) - \frac{{}^E\theta_{ij}}{I}$$

$$x_{ij} = 6 z_i z_j A^\phi \sqrt{I}$$

$$J_0 = x * (4 + 4.581 * x^{-0.7237} * \exp(-0.0120 * x^{0.528}))^{-1}$$

$$J_1 = \frac{d(J_0)}{dx}$$

3.1.3.3 Solids

The fugacity of solids, or the activity coefficient, is assumed to be ideal and valued at 1. This assumption is based on the fact that solids are formed instantly and do take geological time scales to change structure [127], [128].

3.2 Absorption System Model

CO₂ reactive absorption in aqueous brines is modeled in a Bubble Column Reactor. Bubble columns are not the most efficient at absorbing CO₂. However, bubble columns are the simplest form of column reactors that provide enough information on how CO₂ absorption affects sodium removal. The absorption process is temporal and spatial in nature. Absorption dynamics should be considered as the process is slower than the slowest reaction [99], the hydroxide reaction with aqueous CO₂. In addition, the absorption process depends on the column dimensions and constituents. For example, the mass transfer coefficient will be different, for different bubble sizes, column types, and column dimensions [68]. This will require solving bubble hydrodynamics and bubble mass transfer dynamics simultaneously [129]. If the spatial dynamics are not included, the absorption model is considered a lumped process [130]. This approach assumes the whole column is lumped into one volume that exchanges content with the gas phase. On the other hand, the distributed method differentiates the spatial coordinates into smaller domains each with its own lumped process.

3.2.1 Lumped Method

The lumped method uses one mass transfer coefficient for the whole process [130]. There are three resistances to the mass transfer from the gas phase to the liquid phase: gas-side resistance, liquid-side resistance, and bulk-phase where the reaction happens. For CO₂ absorption, the liquid-side coefficient has most of the resistance, and it can be used to estimate the overall mass transfer [68]. The process involves CO₂ bubbling in a nonflowing liquid brine inside a cylindrical contactor. The contactor has an inner diameter of 54 mm and a height of 600 mm [18]. Many research papers are reporting various mass transfer coefficients [131]. The accuracy of the lumped method depends on the range of each input variable.

Hikita et al. [132] published work that was found to have a similar process to the system being developed. The paper reports O₂ absorption in 100-190 mm columns at low pressures [132]. The authors argued that systems with low gas velocities will have low gas holdup, making column diameter irrelevant to mass transfer. In addition, Oak Ridge National Laboratory was able to extend Hikita's work to CO₂ absorption in basic media

[133]. Their experimental results matched with Hikita's model. Furthermore, Schumpe et al. found that mass transfer at high pressures is a function of gas density, in coalescing and non-coalescing liquids [134]. The process modeled assumes bubbly flow with no coalescing [18]. Equation 18 requires obtaining physical properties for saline water solution and gaseous CO₂. These properties were obtained from [99], [135] and the equations used are listed beneath Equation 18.

$$k_l a = 14.9 * G^{0.752} * U_g^{0.76} * \rho_{sol}^{0.852} * \mu_{CO_2}^{0.243} * \mu_{sol}^{-0.079} * \sigma_{sol}^{-1.016} * D_{CO_2}^{0.604} * \left(\frac{\rho_{CO_2}}{\rho_{ref}} \right)^{0.46} \quad (18)$$

Equation 18 defines $k_l a$ as a function of multiple coefficients and all these are defined below.

$$\mu_{sol} = \mu_{water} (1 + A * S + B * S^2)$$

where:

$$\mu_{water} = 4.2844 * 10^{-5} + (0.157 * (T + 64.993)^2 - 91.296)^{-1}$$

$$A = 1.541 + 1.998 * 10^{-2} * T - 9.520 * 10^{-5} * T^2$$

$$B = 7.974 - 7.561 * 10^{-2} * T + 4.724 * 10^{-4} * T^2$$

$$\sigma_{sol} = \sigma_{water} * (1 + (2.26 * 10^{-4} * T + 9.46 * 10^{-3} * \ln(1 + 0.0331 * S)))$$

where:

$$\sigma_{water} = 0.2358 * \left(1 - \frac{T}{647.096} \right)^{1.256} * \left(1 - 0.625 * \left(1 - \frac{T}{647.096} \right) \right)$$

$$\rho_{sol} = \rho_{water} + \rho_{salt}$$

where:

$$\rho_{water} = a_0 + a_1 * T + a_2 * T^2 + a_3 * T^3 + a_4 * T^4$$

$$\rho_{salt} = b_0 * S + b_1 * S * T + b_2 * S * T^2 + b_3 * S * T^3 + b_4 * S^2 * T^4$$

$$\mu_{CO_2} = \frac{0.0148}{1000} * \frac{0.555 * 527.67 + 240}{0.555 * \left(\frac{9}{5} * T\right) + 240} * \left(\frac{9/5 * T}{527.67}\right)^{1.5}$$

$$\rho_{CO_2} = \frac{P * MW_{CO_2}}{Z * R * T}$$

$$D_{CO_2} = 2.35 * 10^{-6} * \exp\left(\frac{-2119}{T}\right) * \left(\frac{\mu_{sol}}{\mu_{water}}\right)^{0.8}$$

Table 10 includes the parameters used in calculating the density of saline water (ρ_{sol}), defined above.

Table 10: Parameters used in calculating the density of saline water.

	0	1	2	3	4
a	9.999E2	2.034E-2	-6.162E-3	2.261E-5	-4.567E-8
b	8.020E2	-2.001	1.677E-2	-3.060E-5	-1.613E-5

3.2.2 Distributed Method

CO₂-absorption modeling using a distributed method will require spatial dynamics along with temporal dynamics. This will result in a Partial Differential Equation system that is usually solved using the Finite Volume Method [136]. There are three phases to be considered: brine liquid, gas bubbles, and suspended solids. Fine solid particles can be included as part of the liquid phase to reduce the total number of phases to two. Each phase can be modeled either using the Euler continuous method or Lagrange discrete method [137]. The Eulerian approach treats a phase as a continuum that occupies a volume fraction of a defined control volume. The Lagrangian approach tracks a particle in space, not being confined by any control volume. At one time step, a particle can be in one control volume, while in another time step, the particle can be in another control volume.

For Bubble Column Reactors, there are three CFD models: Reynolds Averaged Navier Stokes (RANS) [138], Large Eddy Simulations (LES) [139], and Direct Numerical Simulation (DNS) or Discrete Bubble Model (DBM) [140]. RANS model uses an Euler-Euler approach; the Euler method for the liquid phase and the Euler method for the gas phase. DBM model uses an Euler-Lagrange approach; the Euler method for the liquid phase and the Lagrange method for each bubble in the gas phase. The LES model uses an Euler-Euler approach in which small eddies are modeled, while large eddies are resolved at simulation time. In terms of computation complexity, the RANS model is the fastest with the lowest resolution, the LES model is slower with more eddy details, and the DNS model is the slowest with the most simulation details [141].

Another approach to modeling spatial dynamics in a bubble column is the axial dispersion model. The equation assumes that changes in concentration are due to dispersion, advection, or reaction [142]. Effective dispersion is a combination of axial dispersion and cross-phase diffusion. Dispersion coefficients are empirical and need to be evaluated for each column configuration. Diffusion coefficients are easily obtainable, depending on mass transfer assumptions [143]. Numerical solutions could be as simple as using finite difference method to solve PDEs [144].

Gas absorption in Bubble Column Reactors was modeled using the RANS model [129]. The RANS model was combined with species and energy transport equations. Species balance includes diffusion, advection, and reaction in each phase [145]. Energy balance contains advection, thermal diffusion, and heat produced-from/consumed-by reaction and cross-phase absorption [145]. Momentum transfer is represented in the Navier-Stokes equation for incompressible fluids. The equation includes extra momentum terms for bubble drag and momentum from cross-phase diffusion [146]. Also, the system is considered a bubbly flow in which the k - ϵ turbulence model is not included. The turbulence term is removed from the effective viscosity [147]. The equations used in the RANS model are included in Equation 19-22.

Species transport:

$$\frac{\partial \epsilon_j c_i}{\partial t} + \nabla (u \cdot \epsilon_j c_i) = \sum_k D_{ik} \nabla^2 \epsilon_j c_i + r_i \quad (19)$$

Energy transport:

$$\begin{aligned} \frac{\partial (\epsilon_l \rho_l C_p T)}{\partial t} + \nabla (u \cdot \epsilon_l \rho_l C_p T) \\ = -\nabla (k_l \cdot \nabla T) + \sum_i r_i \Delta H + \sum_j r_{abs,j} \Delta H \end{aligned} \quad (20)$$

$$\frac{\partial (\epsilon_g \rho_g C_p T)}{\partial t} + \nabla (u \cdot \epsilon_g \rho_g C_p T) = -\nabla (k_g \cdot \nabla T) + \sum_j r_{abs,j} \Delta H \quad (21)$$

Momentum transport:

$$\frac{\partial (\epsilon_j \rho_j u_j)}{\partial t} + \nabla (\epsilon_j \rho_j u_j u_j) \quad (22)$$

$$= \epsilon_j \nabla \tau_j - \epsilon_j \nabla P + (M_D + \dot{m}_j u_j' - \dot{m}_j u_j) + \epsilon_j \rho_j g$$

$$\tau_j = -\mu_{eff} [\nabla u_j + \nabla u_j^T - \frac{2}{3} I(\nabla \cdot u_j)]$$

$$\mu_{eff} = \mu_l + \mu_T$$

$$\mu_T = 0$$

$$\mu_g = \frac{\rho_g}{\rho_l} \mu_{eff}$$

$$M_D = \frac{1}{2} \rho C_D \frac{A_p}{V} (u_g - u_l) |u_g - u_l|$$

$$C_D = \begin{cases} \frac{24}{Re} (1 + 0.15 Re^{0.687}) & Re < 1000 \\ 0.44 & Re > 1000 \end{cases}$$

$$Re = \frac{\rho_l |u_l - u_g| d_p}{\mu_g}$$

$$\frac{A_p}{V} = \frac{6 \epsilon_g}{d_p}$$

$$\frac{d(\epsilon_j \rho_j)}{dt} + \nabla (\epsilon_j \rho_j u_j) = 0$$

3.3 Numerical Solutions

The capability to simulate and optimize mathematical models depends on the numerical solvers and hardware computation used. All equations developed were solved on an intel core i7-6700HQ processor with 16 GB RAM. The mathematical solver used was the IPOpt [148], a Non-Linear Programming solver, under the GEKKO library [149] in python version 3.8.5 [150]. The challenge in the reaction system was with solving both kinetic and equilibrium reactions simultaneously. There was a challenge with solving spatial dynamics of absorption and temporal dynamics of reactions. This led to some simplifications in the current model proposed, as outlined below.

3.3.1 Reaction System

The challenge with solving reaction systems is that some species are involved in both fast and slow reactions. This makes the system impossible to solve with traditional Differential-Algebraic-Equation solvers. These solvers require the number of equations to be equal to the number of species. A proper way to handle this issue is to combine reactions in a way that a species only shows up in either a kinetic or equilibrium reaction. The systematic way of achieving so is through Gauss-Jordan elimination, as described in the RAFT simulator [151]. The reaction-species matrix is built by having the coefficient of each species in a separate column, in which each row represents a reaction. This coefficient matrix can be reduced to row echelon form in order to separate kinetic species from equilibrium species. All row operations performed on the coefficient matrix are applied to a species identity matrix. The resultant species matrix produces the new reaction system. This final system will include kinetic reactions, equilibrium reactions, and mass balance equations.

initial system: (with 10 equations & 9 variables)

$$K_{H_2O(l)} = \frac{\gamma_{H^+}H^+ * \gamma_{OH^-}OH^-}{\gamma_{H_2O(l)}} \quad (23)$$

$$\begin{aligned} \frac{d CO_{2(aq.)}}{dt} = & k_1 a \left(\phi_{CO_{2(g)}} CO_{2(g)} - \gamma_{CO_{2(aq.)}} CO_{2(aq.)} \right) + k_{HCO_3^-} \\ & * \gamma_{HCO_3^-} HCO_3^- - k_{OH^-} * \gamma_{CO_{2(aq.)}} CO_{2(aq.)} * \gamma_{OH^-} OH^- \end{aligned} \quad (24)$$

$$\begin{aligned} & + k_{NH_2COO^-} * \gamma_{NH_2COO^-} NH_2COO^- * \gamma_{H^+} H^+ - k_{NH_3(aq.)} \\ & * \gamma_{CO_{2(aq.)}} CO_{2(aq.)} * \gamma_{NH_3(aq.)} NH_{3(aq.)} \\ & CO_{2(g)} = K_{CO_{2(g)}} * P_{CO_{2(g)}} \end{aligned} \quad (25)$$

$$\begin{aligned} \frac{d HCO_3^-}{dt} = & k_{OH^-} * \gamma_{CO_{2(aq.)}} CO_{2(aq.)} * \gamma_{OH^-} OH^- - k_{HCO_3^-} \\ & * \gamma_{HCO_3^-} HCO_3^- \end{aligned} \quad (26)$$

$$K_{CO_3^{2-}} = \frac{\gamma_{HCO_3^-} HCO_3^-}{\gamma_{H^+} H^+ * \gamma_{CO_3^{2-}} CO_3^{2-}} \quad (27)$$

$$\begin{aligned} \frac{d NH_2COO^-}{dt} = & k_{NH_3(aq.)} * \gamma_{CO_{2(aq.)}} CO_{2(aq.)} * \gamma_{NH_3(aq.)} NH_{3(aq.)} \\ & - k_{NH_2COO^-} * \gamma_{NH_2COO^-} NH_2COO^- * \gamma_{H^+} H^+ \end{aligned} \quad (28)$$

$$K_{NH_4^+} = \frac{\gamma_{NH_4^+} NH_4^+}{\gamma_{H^+} H^+ * \gamma_{NH_3(aq.)} NH_{3(aq.)}} \quad (29)$$

$$NH_{3(aq.)} + NH_2COO^- + NH_4^+ = NH_{3t=0} \quad (30)$$

$$CO_{2(aq.)} + HCO_3^- + NH_2COO^- + CO_3^{2-} = CO_{2t=0} + \int CO_{2(aq.)} dt \quad (31)$$

$$H^+ + NH_4^+ = OH^- + HCO_3^- + 2 * CO_3^{2-} + NH_2COO^- \quad (32)$$

final system: (with 9 equations & 9 variables)

$$K_{H_2O(l)} = \frac{\gamma_{H^+} H^+ * \gamma_{OH^-} OH^-}{\gamma_{H_2O(l)}} \quad (23)$$

$$\begin{aligned} \frac{d CO_{2(aq.)}}{dt} + \frac{d HCO_3^-}{dt} + \frac{d CO_3^{2-}}{dt} + \frac{d NH_2COO^-}{dt} \\ = k_1 a \left(\phi_{CO_{2(g)}} CO_{2(g)} - \gamma_{CO_{2(aq.)}} CO_{2(aq.)} \right) \end{aligned} \quad (33)$$

$$CO_{2(g)} = K_{CO_{2(g)}} * P_{CO_{2(g)}} \quad (25)$$

$$\begin{aligned} \frac{d HCO_3^-}{dt} + \frac{d CO_3^{2-}}{dt} &= k_{OH^-} * \gamma_{CO_2(aq)} CO_{2(aq)} * \gamma_{OH^-} OH^- - k_{HCO_3^-} \\ &* \gamma_{HCO_3^-} HCO_3^- \end{aligned} \quad (34)$$

$$K_{CO_3^{2-}} = \frac{\gamma_{HCO_3^-} HCO_3^-}{\gamma_{H^+} H^+ * \gamma_{CO_3^{2-}} CO_3^{2-}} \quad (27)$$

$$\begin{aligned} \frac{d NH_2COO^-}{dt} &= k_{NH_3(aq)} * \gamma_{CO_2(aq)} CO_{2(aq)} * \gamma_{NH_3(aq)} NH_{3(aq)} \\ &- k_{NH_2COO^-} * \gamma_{NH_2COO^-} NH_2COO^- * \gamma_{H^+} H^+ \end{aligned} \quad (28)$$

$$K_{NH_4^+} = \frac{\gamma_{NH_4^+} NH_4^+}{\gamma_{H^+} H^+ * \gamma_{NH_3(aq)} NH_{3(aq)}} \quad (29)$$

$$\frac{d NH_{3(aq)}}{dt} + \frac{d NH_2COO^-}{dt} + \frac{d NH_4^+}{dt} = 0 \quad (35)$$

$$\begin{aligned} \frac{d H^+}{dt} &= \frac{d HCO_3^-}{dt} + 2 \frac{d CO_3^{2-}}{dt} + \frac{d NH_{3(aq)}}{dt} + 2 \frac{d NH_2COO^-}{dt} \\ &+ \frac{d OH^-}{dt} \end{aligned} \quad (36)$$

3.3.2 Reaction and Absorption System

Solving spatial dynamics with temporal dynamics can become challenging with highly non-linear reaction equations. Operator splitting decouples reaction systems from absorption dynamics to reduce computation complexity but with error introduction [152]. This can be alleviated using higher orders of operator splitting. The most common operator splitting is Strang splitting which is a second-order splitting scheme [153]. Operator splitting applied to the reaction-absorption system will split the species transport equations into Partial Differential Equations and Differential Algebraic Equations, instead of the complex Partial Differential Algebraic Equations. This will reduce complexity and computation time.

3.3.3 Model Simplification

The current model was complex and needed to be simplified so that it could be simulated within a reasonable time frame. The spatial dynamics were removed, as hydrodynamics had a negligible effect on Na^+ removal. So, the distributed method was

replaced with the lumped method, using a mass transfer coefficient. There were hundreds of possible complex solids that could precipitate, but since the concentration of dissolved ions is low, only six salts were considered (NaHCO_3 , $\text{Na}_2\text{CO}_3 \cdot 10\text{H}_2\text{O}$, CaCO_3 , $\text{CaSO}_4 \cdot 2\text{H}_2\text{O}$, $\text{Mg}(\text{OH})_2$, $\text{Ca}(\text{OH})_2$). Aqueous complexation reactions (such as CaHCO_3^+ and MgOH^+) were removed since the brine solution is treated in basic-to-neutral media [154], [155]. Gas mixtures were not considered because this research aims to study CO_2 effects on salt precipitation. Gas mixtures will impact results, yet with or without their consideration, results should show similar patterns. Equilibrium constants were poly-fitted to remove complexity on the solver. The final code used model used in simulation and optimization can be found in A3.7 and A3.8, respectively. Figure 4 shows a flowchart overview of the final model being used and how it is built.

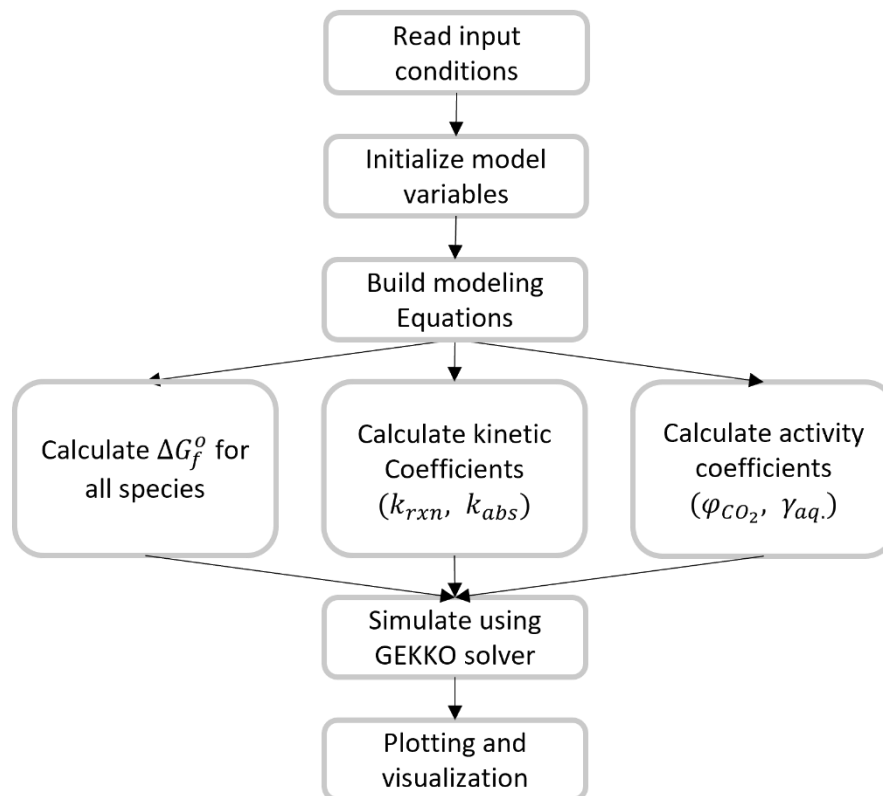


Figure 4: Simulation flowchart of modeling reactions.

3.4 Gas Hydrate Model

Gas-Based hydrate is a thermal desalination process that removes salts by freezing brine solutions [57]. It is more attractive than freezing distillation because it can be operated at higher temperatures, above 0°C [57]. In addition, CO_2 absorbed in the sodium removal process can be utilized to remove chloride ions by lowering the

operating column temperature by 7-10°C. The hydration process is quite complex to model using traditional methods, so black-box modeling was utilized to study the system. The process starts with CO₂-H₂O clathrate forming in which a snow-like structure traps water molecules [58]. Some ions in the brine solution are trapped inside the clathrate and do not allow the process to achieve 100% ion removal [58]. Machine Learning will be used to study the entrapment process for simulation and optimization.

3.4.1 Data Collection

Experimental data was collected from CO₂-based hydrate systems that had initial concentration, final concentration, and operating condition data points. The ions included were Na⁺, K⁺, Mg²⁺, Ca²⁺, Cl⁻, and SO₄²⁻. The initial concentration of the samples was collected from seawater, synthetic brines, salt solutions, or a combination of seawater solutions. The final concentration of the samples was either reported as removal efficiency, ion's removed concentration, or ion's remaining concentration. The final concentration of all the samples was changed to the ion's removed concentration. For some samples, the anions' composition was not available, and it had to be estimated. The initial concentration was estimated using Na⁺/Cl⁻ ratio for Cl⁻ and Cl⁻/SO₄²⁻ for SO₄²⁻. The ratio used was obtained from the given salt composition. The final composition was estimated using the average removal efficiency of all the other ions. The list of the data used, and all the references can be found in Appendix A2.

3.4.2 Machine Learning Algorithms

Machine Learning (ML) is a subset of Artificial Intelligence in which the system learns from the given dataset and can keep improving with newly added data [156]. There are three important components to a machine learning system: the learning algorithm, the loss function, and optimization [157]. The learning algorithm is the representation function that allows the system to process data and extract patterns. The loss function is the cost associated with deviating from the correct output. Optimization is the solver used to search in the space; grid-space. In addition, regularization can be added to the loss function to prevent overfitting to the data, allowing it to generalize to new input data better [158]. Also, hyperparameters are parameters used to tweak the ML process [159].

The error from variance and bias is another important factor to consider in a Machine Learning system [160]. Model bias culminates in how well it can represent the data. Input data that have a quadratic relationship with the output variable will have a high bias, if the given model was linear. This is because the model cannot capture the essential information. Model variance is the change in model structure when new data is represented. A model with high variance will keep changing when new data is added, indicating its inability to learn. An ideal machine learning system should have low bias and low variance; however, this is impossible, as they are inversely proportional [160].

3.4.2.1 Weighted Regression

Regression using weights is one approach to Machine Learning in which the system estimates the importance of each feature through a coefficient. Linear proportionality is the simplest weighting technique. Non-linear weightings apply a nonlinear transformation to the input feature. Multi-layer Perceptron is the ultimate form of a non-linear weighting technique. Below is a summary of the key factors to assess when considering each.

3.4.2.1.1 Linear Regression

Linear regression is the oldest and most studied modeling technique in statistical analysis [161]. The regression maps the relationship between the dependent variable and independent variables using linear proportionality. This is considered the learning function, and the loss function is the mean squared error. Regularization can be applied in the context of linear regression, in cases, such as, lasso regression and ridge regression [161]. There are no hyperparameters used in linear regression other than the constants in regularization.

3.4.2.1.2 Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) is a form of Artificial Neural Network that is composed of layers [162], as seen in Figure 5. The first layer is the input layer, and it consists of all the dataset features. Each feature is a neuron. Data is passed to subsequent middle layers, called hidden layers. Each neuron in a hidden layer takes a linear weighted sum of all the neurons in the previous layer and passes it through an activation function

[163]. The activation/non-linear function allows certain data to pass to the next layer. The final layer is the output layer which consists of all the output-responses/neurons [162]. An MLP with one hidden layer is sufficient to solve any problem [164]. There are many optimization functions for MLP, but the most common one is gradient descent. The network updates its weights by differentiating the error in the output layer, and its rippling effect all the way to the input layer; known as backpropagation [163]. This requires the activation to be differentiable in addition to being non-linear. Regularization comes in the form of dropout, such as, neuron dropout, weight dropout, or layer dropout [165]. Common hyperparameters are the learning rate for gradient descent, the number of hidden layers, the number of neurons in each layer, and the dropout rate for regularization.

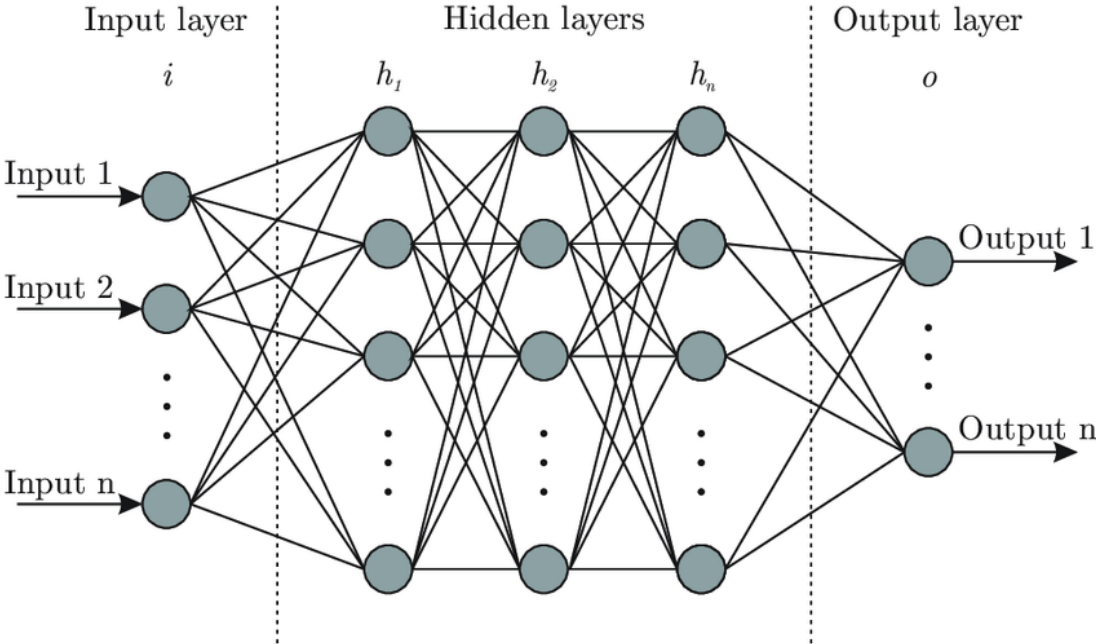


Figure 5: Multi-Layer Perceptron architecture with an input layer composed of n inputs, n hidden layers, and an output layer with n responses [166].

3.4.2.2 Regression Trees

3.4.2.2.1 Decision Tree

A decision tree is a form of a binary tree in which each node is either a decision node or a leaf node [167]. A decision node uses a condition to evaluate whether to choose the left child node or the right child node. Each decision node uses one feature in its condition to evaluate how to split the data. If the condition is evaluated to be true, the

decision is passed to the left node, and vice versa for the right node [168]. Finally, the decision ends in a leaf node in which its value is equal to the average of all the training data points in that node, as seen in Figure 6. The loss function for a decision tree is to minimize the entropy of the data [169]. The data is split recursively to maximize the amount of information gained. A hyperparameter for decision trees is the number of features to consider for decision nodes.

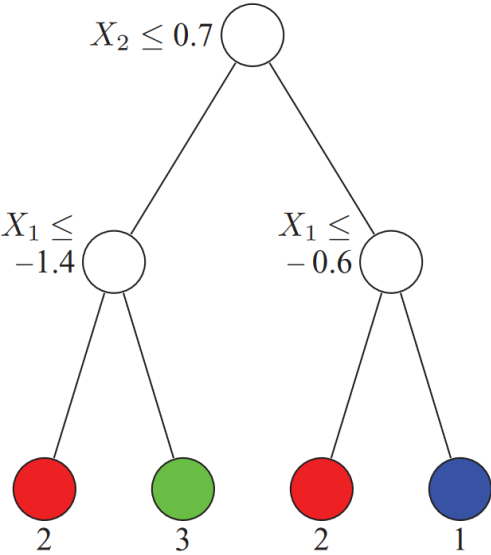


Figure 6: A typical structure of a decision tree [168].

3.4.2.2.2 *Random Forest*

Random Forest is a combination of decision trees that use random sampling to achieve lower variance; a problem with decision trees [170]. The Random Forest algorithm creates new datasets from training data using random sampling with replacement [171]. In addition, the algorithm randomly samples a subset of features for training each decision tree. The final decision, or the regression value, is equal to aggregating the decisions from all the decision tree instances and taking the average for the final value. This method applies two random processes, making the learning algorithm less sensitive to the original training data. The number of feature subsets is usually equal to the logarithm or square root of the feature-set size. Common hyperparameters include the number of decision trees and the number of features for each tree. One problem with Random Forest is that the learning system sacrifices interpretability for lower variance [171].

3.4.2.2.3 XGBoost

eXtreme Gradient Boost (XGBoost) is the ultimate form of a decision tree that is used commonly in Machine Learning tasks [172]. This system combines an ensemble of methods that give it the ‘*extreme*’ title in its name. The system uses gradient boosting that gradually adds more trees to improve on older mis-modeled data. In contrast, Random Forest adds decision trees independent of the tree’s performance [170]. In addition, the system can handle missing values, as it treats them as zeros [172]. The learning system applies cross-validation at each step to lower model bias. The system runs in parallel to use computing resources efficiently. The system applies pruning, allowing for deeper yet optimized trees [172]. The system uses regularization in the form of lasso or ridge regularization. Common hyperparameters are learning rate, max depth of the tree, and minimum child node weight.

3.4.2.3 Grouping Algorithms

The main concept behind grouping algorithms is the kernel function. A kernel is a linear/non-linear transformation that aims to separate data into higher-dimensional space [173]. The kernel is the eigenvector solution in Principal Component Analysis. The kernel is used in Support Vector Machines to separate data by a hyperplane. The kernel is the distance functionality in the k-Nearest Neighbors algorithm.

3.4.2.3.1 Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction algorithm that uses Singular Value Decomposition [174]. The learning system, first, calculates the covariance matrix of all the feature-set. Then, the algorithm calculates the eigenvectors applied to the covariance matrix [175]. The eigenvalues establish a hierarchy of the eigenvectors based on their corresponding values. The top N eigenvectors are the principal components, and the dimension of the feature set is reduced to N [174]. There is no loss function or regularization needed since SVD does calculations analytically. As the feature set grows and the amount of training data, analytical methods become slow, and other optimization algorithms can be used [176]. The value of N is the only hyperparameter to consider.

3.4.2.3.2 Support Vector Machine

Support Vector Machine, or Network, (SVM) is a Machine Learning algorithm that uses a hyperplane to separate data into groups [173]. The distance between the nearest data points and the hyperplane is the margin. The loss function maximizes the marginal distance and penalizes data points that are on the wrong side of the hyperplane. For regression, the learning machine tries to find the best fit hyperplane to the training data [177]. Figure 7 shows a typical SVM regressor with the maximum margin as a dotted line on either side of the plane [178]. The system can model non-linear behavior through a non-linear transformation of the data. Regularization influences the number of data points considered when calculating the margin [179]. Common hyperparameters are the kernel transformation function, the penalty constant for the loss function, and the penalty tolerance level.

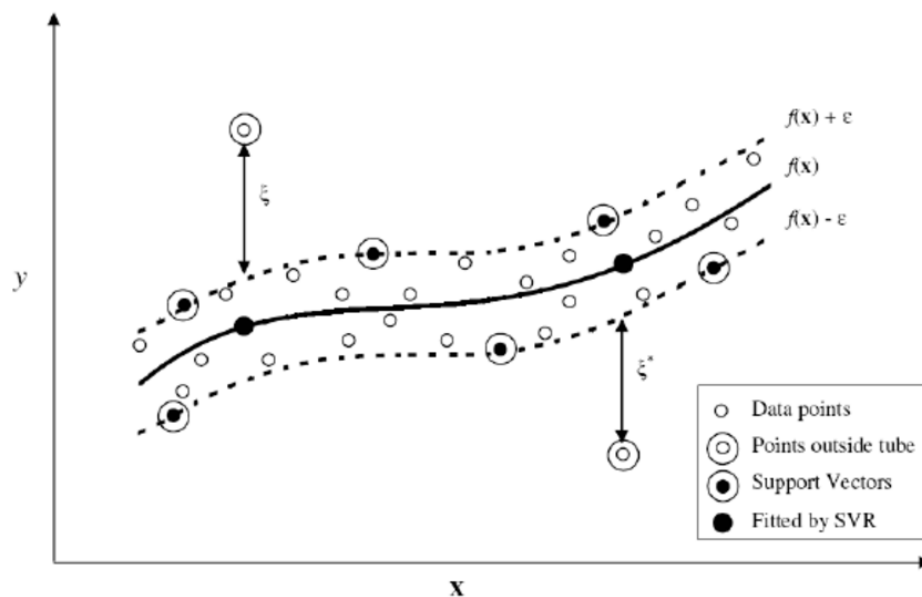


Figure 7: A typical Support Vector Machine used for Regression with the hyperplane being the bold, solid line in the middle [178].

3.4.2.3.3 k-Nearest Neighbors

The k-Nearest Neighbors is a simple yet powerful Machine Learning method [180]. The system functions under the assumption that objects in proximity behave similarly [181]. Training data are clustered into groups based on a previously chosen number. For regression, the clustering algorithm separates the data into N clusters using a distribution function. In prediction, the k nearest data points are used to calculate the

average value of the new data point [182]. The distance is calculated using vector-norm distance or the simple difference between the two points. This becomes a disadvantage when the number of features increases because evaluation becomes slower [183]. The system does not use a loss function or a regularization parameter, but the clustering algorithm might use mean squared error. The hyperparameters to consider are the k number of neighbors and the N number of clusters.

Chapter 4: Results and Discussion

This chapter presents an answer to the research objectives by utilizing the theoretical framework established. The reaction system built gives insight into how CO₂ absorption can achieve the objective of maximizing salt removal. Firstly, model validation verifies that the mathematical equations can be examined within the validated region. Three components are validated: CO₂ absorption, CO₂ reaction in basic solutions, and salt solubility diagrams. The CO₂-hydrate formation is modeled, and the Cl⁻ removal data is validated using Machine Learning. Secondly, simulations provide an understanding of the underlying factors affecting modeling capabilities, and a reference to compare the performance of the studied absorbents. Also, simulations provide insight into reaction pathways to achieve the desired product. Finally, optimization presents the final solution at which the Bubble Column Reactor should be operated. This sets the stage for optimizing operating conditions to reach research objectives of maximizing salt removal (Na⁺ and Cl⁻) and maximizing CO₂ absorption. These results can be taken into the lab and used as an initial point for optimization using statistical methods.

4.1 Model Validation

Model validation aims to make sure that the equations being developed in the theoretical framework agree with the experimental results published. For CO₂ absorption, the developed model is compared to CO₂ solubility data that is a function of pressure (1 – 90 bar) and temperature (25 – 100°C). In addition, the CO₂ reaction with hydroxide and ammonia is modeled and results are compared with simulated kinetic data. Finally, the thermodynamic and activity models are validated over a temperature range of 10 – 50°C.

4.1.1 CO₂ Absorption

CO₂ absorption thermodynamics is important to be validated because it is one of the main components of modeling Bubble Column Reactors. The CO₂ absorption phenomenon being modeled considers phase transition, aqueous CO₂ dissociation reactions, and water dissociation for pH balance. In addition, both phases, gaseous and aqueous, included non-ideality that was modeled using fugacity coefficients. Figure 8

shows a good agreement between experimental data and model predictions for aqueous CO₂ solubility (mol/kg) as a function of pressure (1 – 90 bar) and temperature (25 – 100°C). The experimental data used in Figure 8 was obtained from Appelo et al. [184].

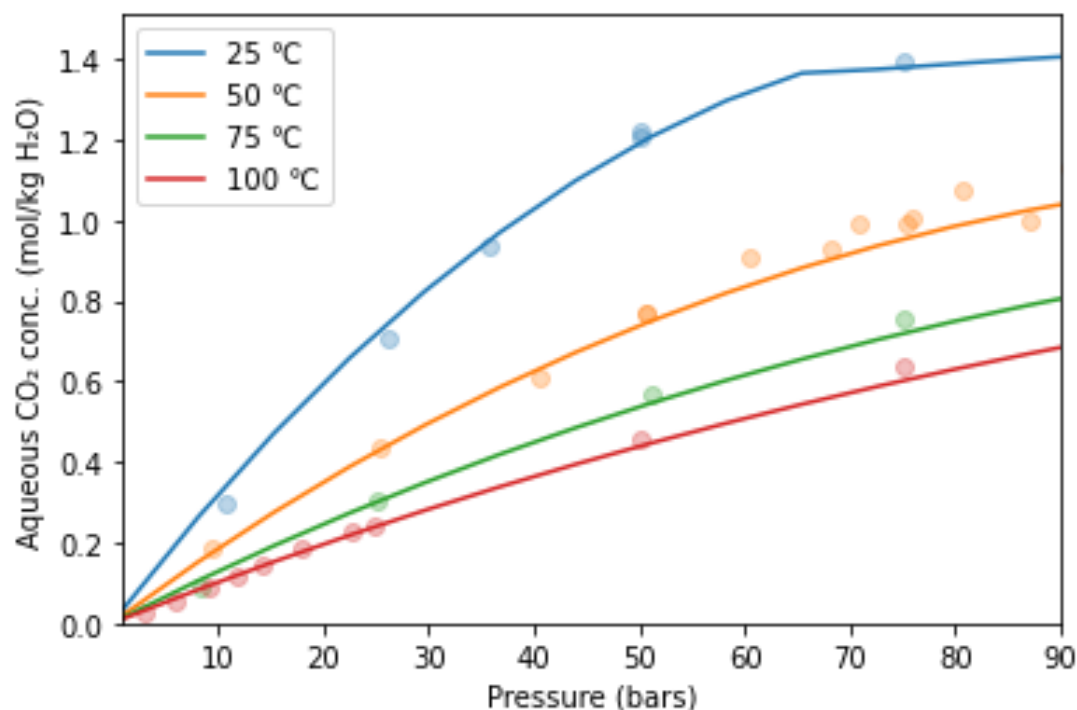


Figure 8: CO₂ solubility in water at various temperatures and pressures, experimental data (dots), and our model (lines).

4.1.2 CO₂ Reaction

CO₂ reaction with absorbents is the second component of modeling Bubble Column Reactors. CO₂ reaction has only two kinetic reactions: CO₂-OH⁻ reaction and CO₂-NH₃ reaction. These bases are the only way aqueous CO₂ interacts in the bulk phase. There are many publications reporting a wide range of CO₂-NH₃ rate constants with order of magnitudes of difference [185]–[189]. The original CO₂-NH₃ reaction rate from Pinsent et. al [94] was chosen because of its wide range of applicability and its reporting in citations. There has been no experimental data reported on CO₂ reaction dynamics in basic media; however, there was one report found with simulation data [190]. The validation was done by reacting 3.8 mM CO_{2(aq.)} with 4.0 mM NH_{3(aq.)}. The validation graphs, in figures 9b and 10b, show four to five times slower kinetics when compared to simulated results [190], in figures 9a and 10a. Both figures show similar

behavior of an NH_2COO^- peak (around the one-second mark) and the rise of HCO_3^- and NH_4^+ .

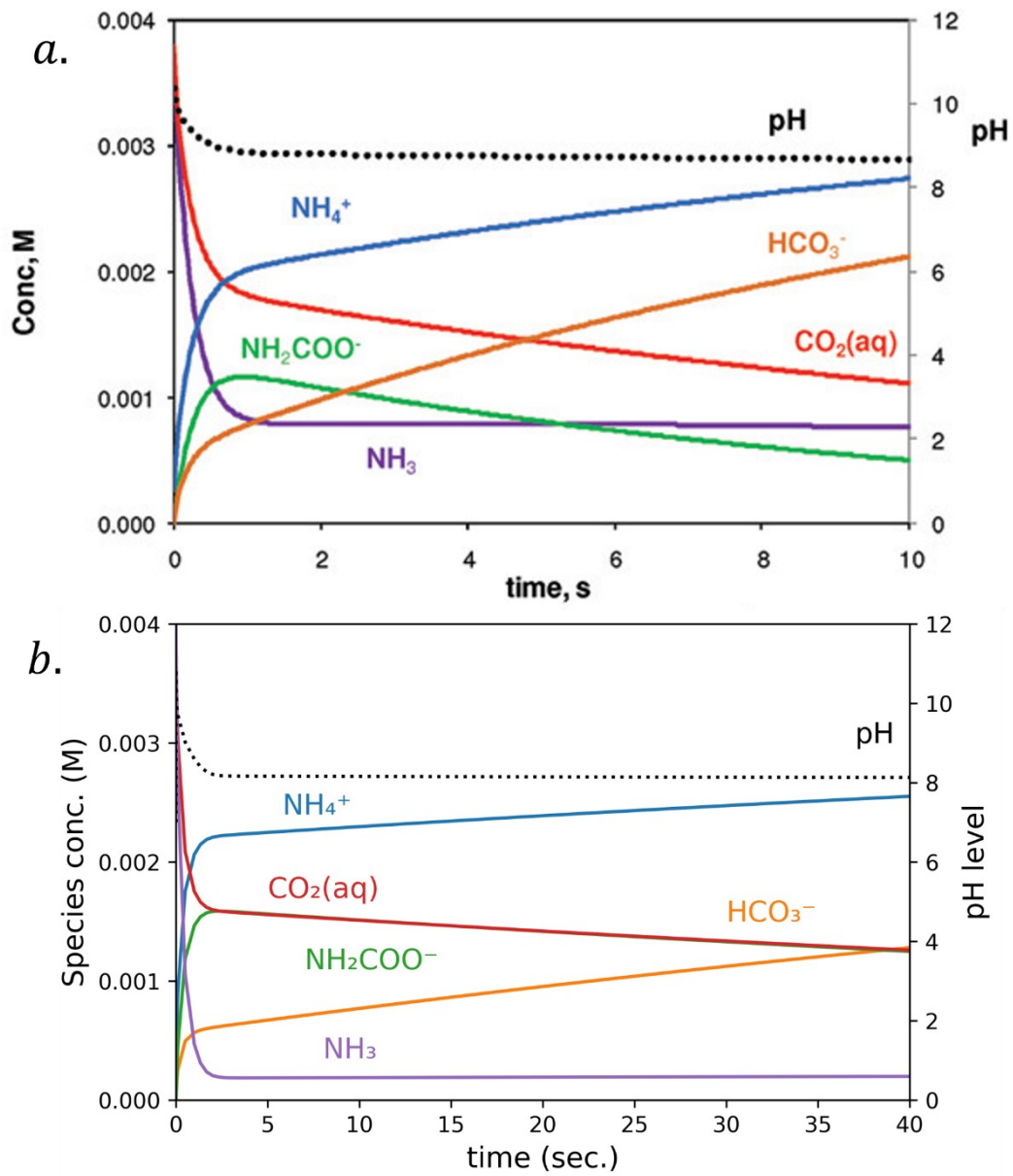


Figure 9: Simulation of 3.8 mM $\text{CO}_2(\text{aq.})$ and 4.0 mM $\text{NH}_3(\text{aq.})$ reaction a) for 10 seconds [190], and b) for 40 seconds (this study).

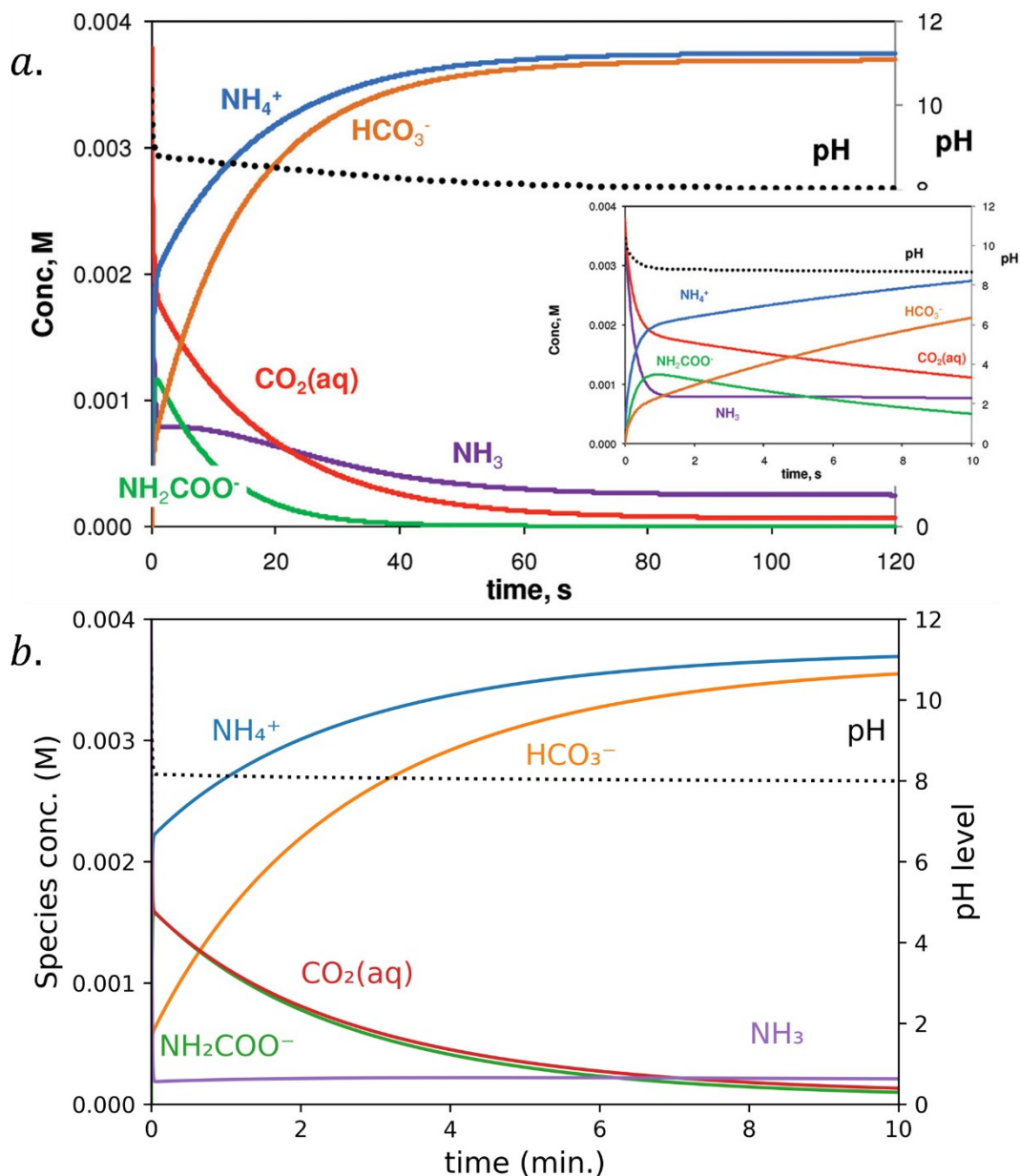


Figure 10: Simulation of 3.8 mM CO₂(aq.) and 4.0 mM NH₃(aq.) reaction a) for 120 seconds [190], and b) for 10 minutes (this study).

4.1.3 Solubility Diagrams

Solubility diagrams are good at showing the effects of concentration and non-ideality of aqueous species. The diagrams plot the solubility of inorganic salts over a certain temperature range. Solubilities are affected by temperature, and its study can validate solubility product and ionic activity of species in water. The six solids used in the model were evaluated: NaHCO₃, Na₂CO₃·10H₂O, CaCO₃, CaSO₄·2H₂O, Mg(OH)₂, Ca(OH)₂. Figure 11 shows the solubilities of these solids.

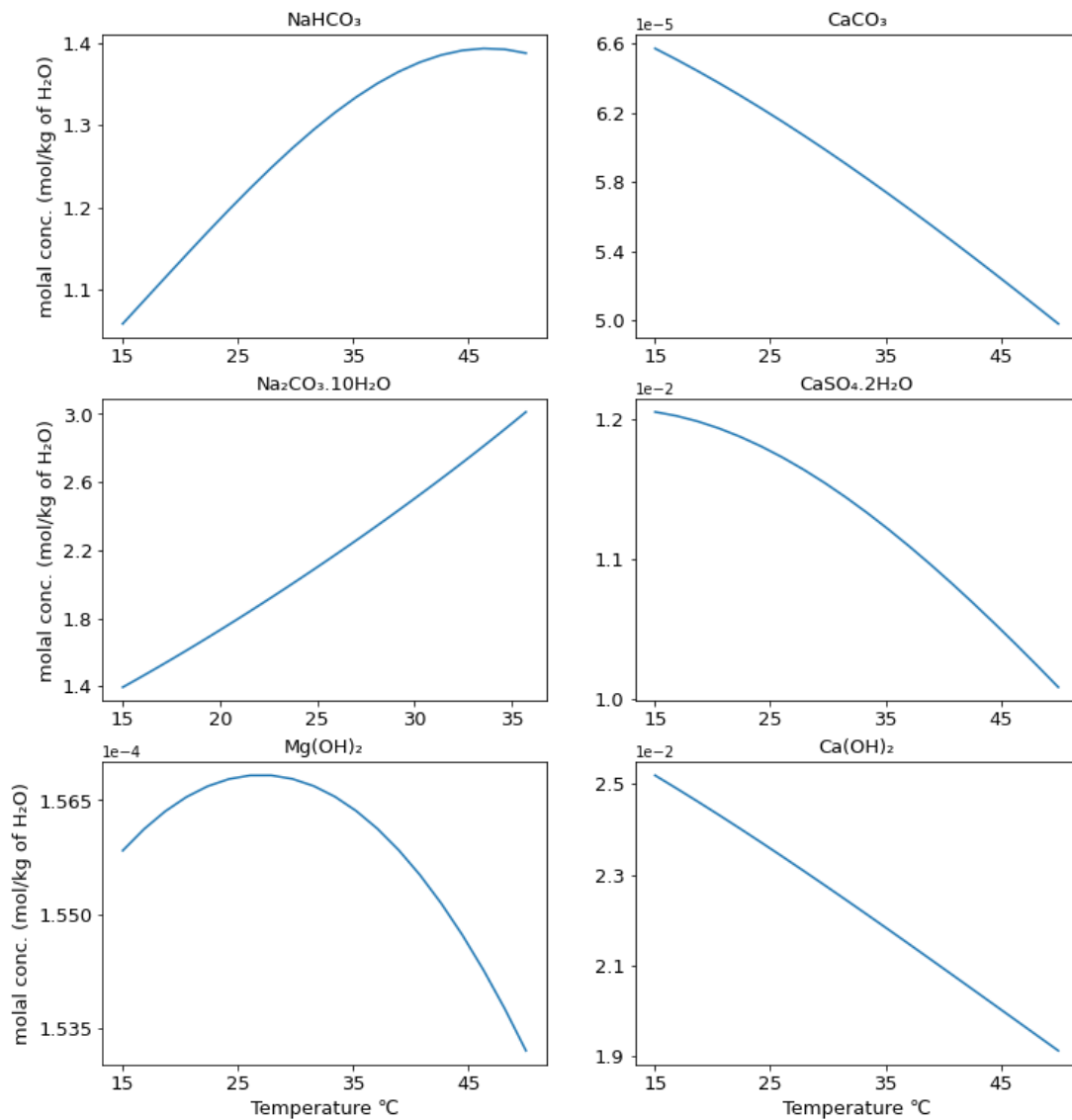


Figure 11: Solubility plots of the six salts used in the reaction model.

4.1.4 Validation of CO₂-hydrate data

Chloride removal using CO₂-hydrate was modeled and optimized using Machine Learning. The amount of data collected pushed us towards the machine learning approach. Data validation is achieved using a three-step process, preprocessing, data mining, and post-processing. The input data includes the initial concentration of four cations (Na⁺, K⁺, Ca²⁺, Mg²⁺), two anions (Cl⁻ and SO₄²⁻), and the operating conditions (temperature and pressure). The output data includes the final concentration of all ions present. This data was collected from multiple sources that comes often with noise. Input features were expanded and reduced using Principal Component Analysis, to the most significant ones. This was followed by a data mining step, in which three families of

Machine Learning algorithms were assessed using two scoring functions. Postprocessing analysis looks at the effect of preprocessing on the model evaluation. The final model will be used in the optimization section of this thesis.

4.1.4.1 Preprocessing

The preprocessing step involves cleaning the raw data collected, attached in Appendix A2, and preparing it for data mining. The input data includes an initial concentration of cations, the initial concentration of anions, and the operating temperature and pressure. The output data includes the final concentration of all ions present. The final concentration of chloride was the only output response considered. The final concentrations of the other ions were not considered to ensure no data leakage. Data points with no chloride ion present were treated as missing values and removed. Also, data points with zero chloride removal were not considered because their count was small, less than five out of 207 data points, to study CO₂ hydrate formation. The formation behavior can be modeled analytically/numerically using the Celsius-Clapeyron equation [191]. Finally, the input data was separated into three groupings (cation data, anion data, and operating conditions data) for Principal Component Analysis (PCA).

PCA is a dimensionality reduction technique that was used to extract important features from the input data. The initial concentration of cations was expanded by evaluating the square, cube, logarithm, square root, and inverse of each cation's initial concentration. This resulted in a six-times expansion, from four input features (initial concentration of Na⁺, K⁺, Ca²⁺, Mg²⁺) to 24 features. The new feature set was studied using PCA and the top five principal components were considered. These principal components are a linear combination of all the cation-expanded feature-set. Figure 12 plots the three different output variables against the five principal components, PC 1 through PC 5. The output variables considered were the amount of chloride removed, the amount of chloride remaining, and the percentage of chloride removal, or process efficiency. Looking at the plots of the three output variables, the amount of chloride removed is the best output response, since it shows some variation with the input principal components. Another conclusion made is that the first principal component, PC

1, is the only feature showing any variation with the output response, while the other features scatter with the output responses. So, the first principal component (PC 1) was only chosen for the final regression analysis.

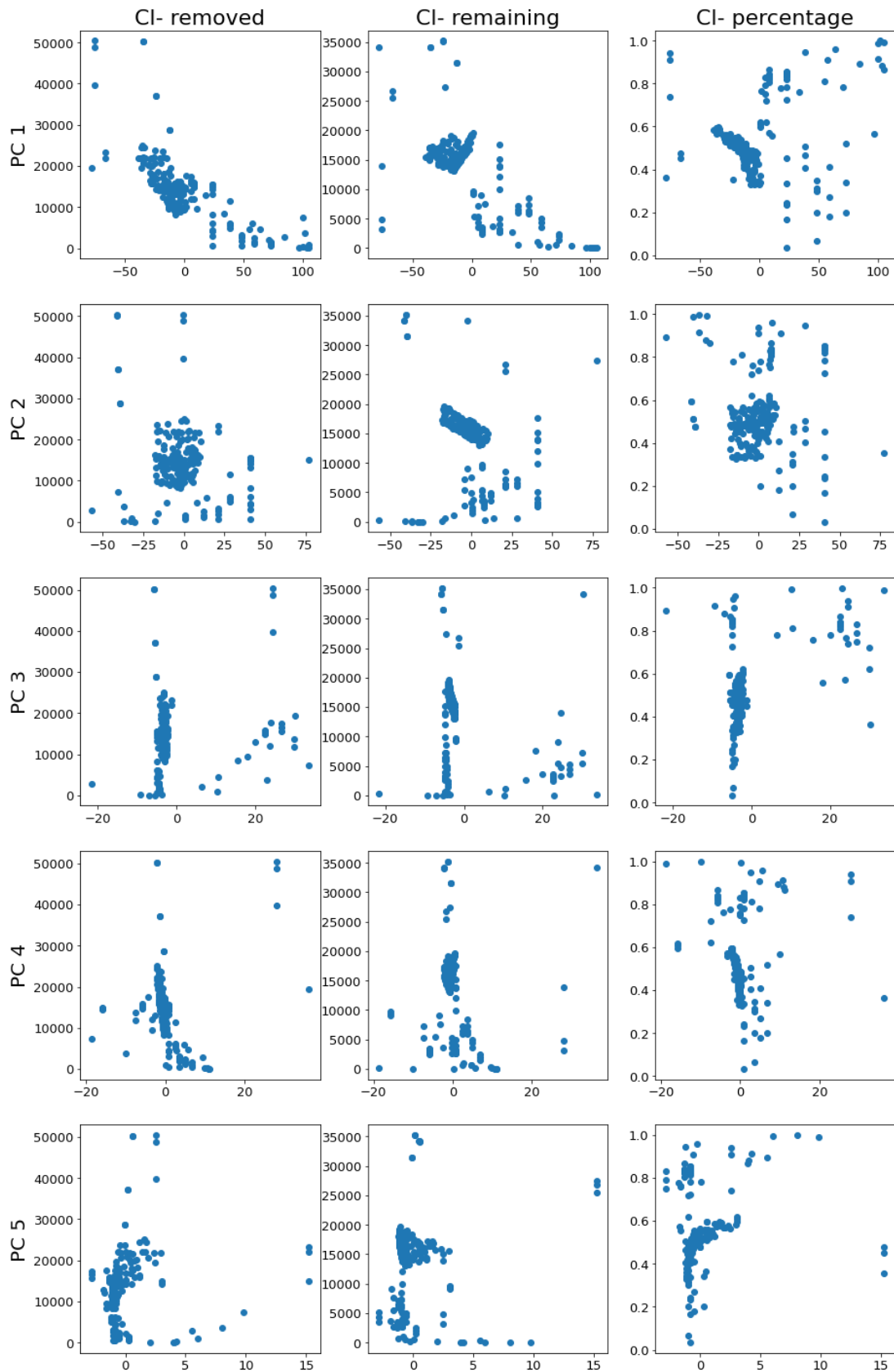


Figure 12: Scatter plots of the three output responses vs the five principal components used for cation data.

A similar procedure was applied to the anion input data in which the concentration of two anions (Cl^- , SO_4^{2-}) was expanded to 12 features and the top five features were selected. These principal components are a linear combination of all the anion-expanded feature-set. Figure 13 shows the three output responses against the five principal components, PC 6 through PC 10. A similar conclusion can be made about the best output response. The amount of chloride removed is the best output response, since it shows some variation with the input principal components. In addition, the first two principal components (PC 6 and PC 7) show variation in the output response and were chosen for the final regression analysis.

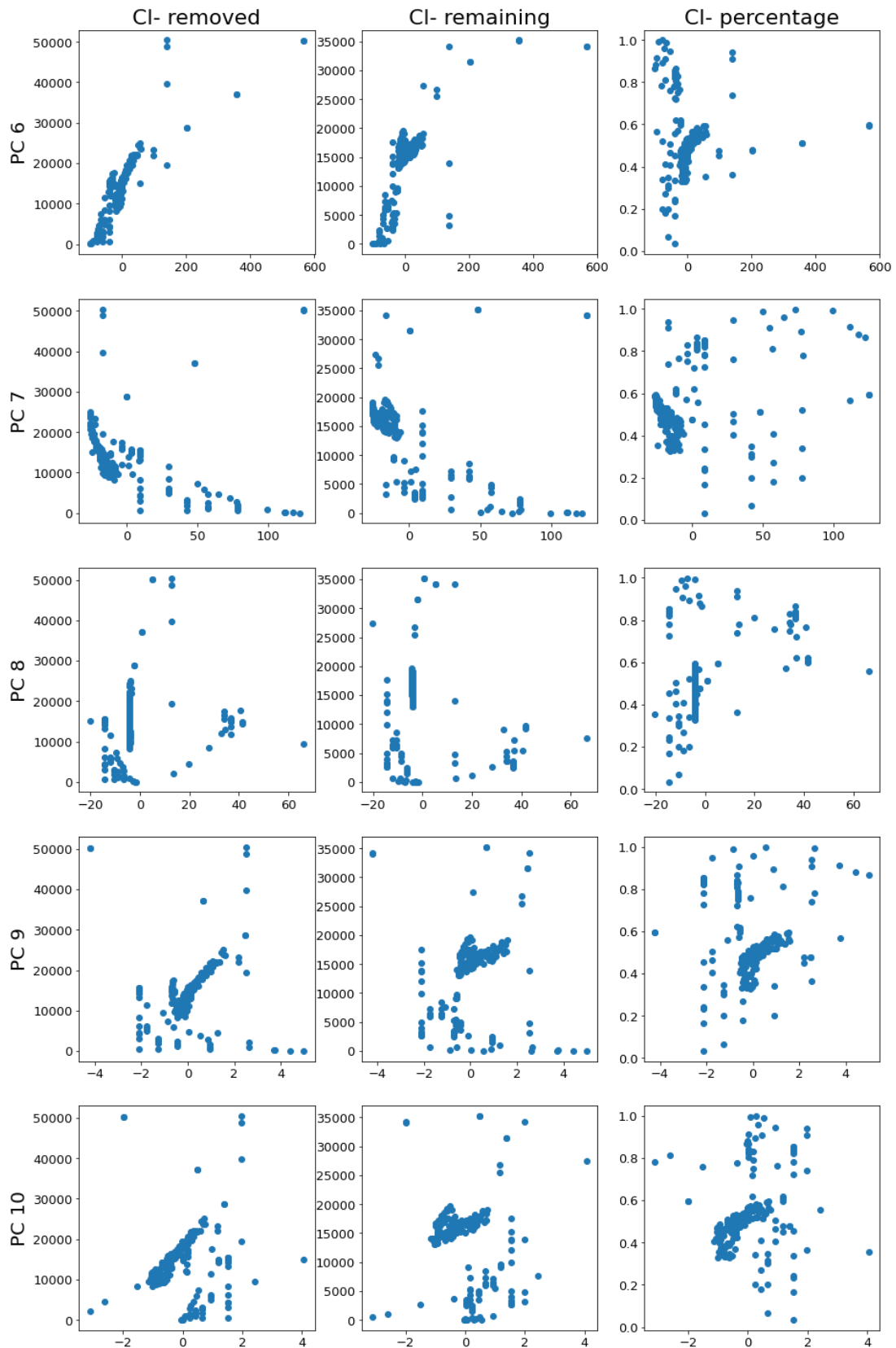


Figure 13: Scatter plots of the three output responses vs the five principal components used for anion data.

This procedure failed to work when applying it to the operating temperature and pressure. The operating conditions were expanded from two to 12 features and reduced to the top five principal components, PC 11 through PC 15. These principal components are a linear combination of all the operating-conditions expanded feature-set. It was found that the operating conditions did not produce any statistically significant response, as seen in Figure 14. So, none of the principal components were considered, while only one feature from the cation principal components (PC 1) and two features from the anion principal components (PC 6 and PC 7) were considered.

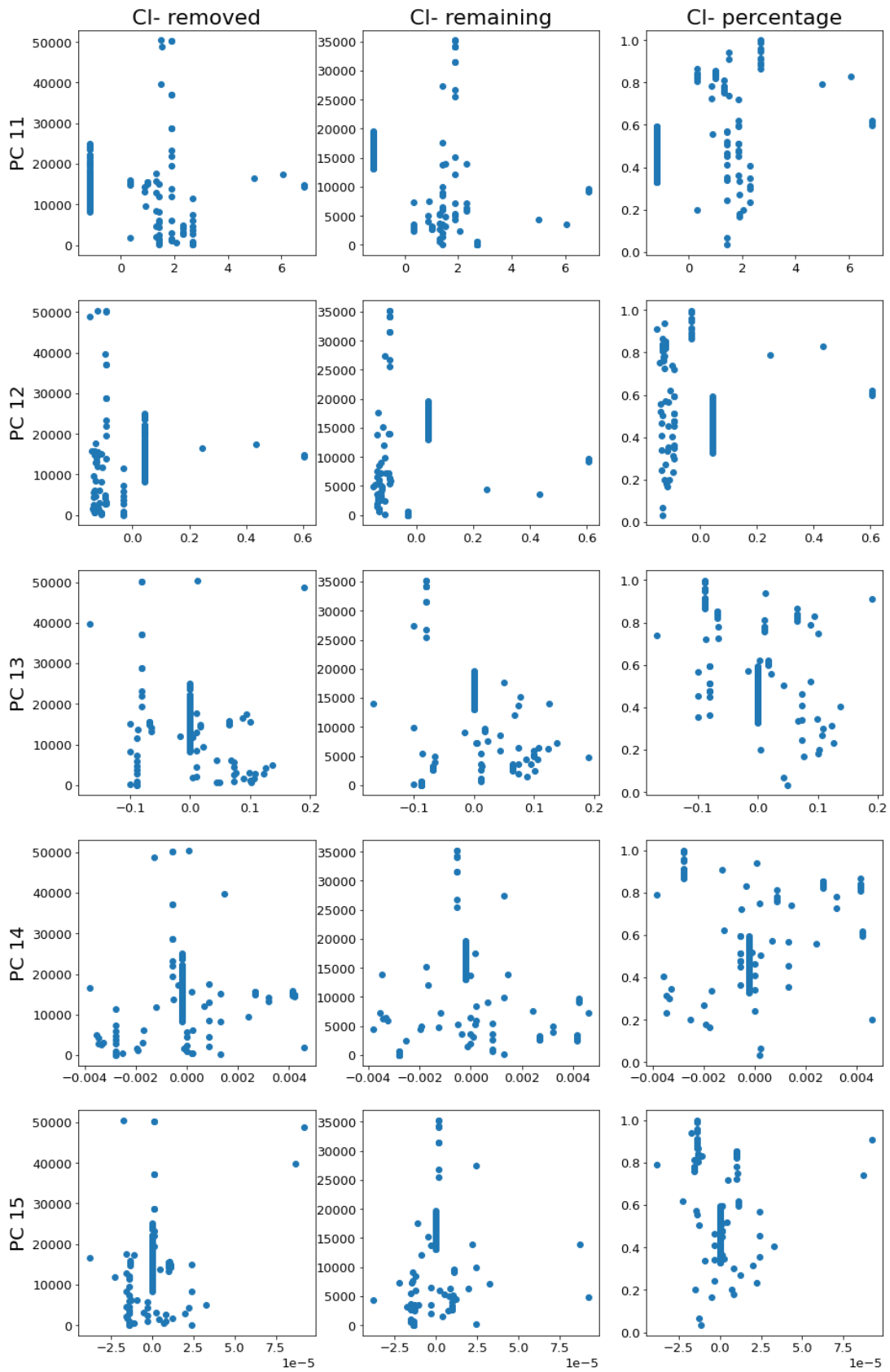


Figure 14: Scatter plots of the three output responses vs the five principal components used for the operating temperature and pressure data.

4.1.4.2 Data Mining

The results obtained from the PCA were used in the model selection phase of the Machine Learning optimization. There were seven algorithms tested: two weighting systems, three tree techniques, and two grouping methods, as seen in Table 11. All algorithms were implemented in python software version 3.8.5 [150] using scikit-learn library version 1.0.2 [192] and XGBoost library version 1.5.0 [172]. The Input data were randomly split into training and testing data with a 70/30 split, respectively. The training dataset was passed through the PCA preprocessing step and only three features were chosen, PC 1, PC 6, and PC 7. All learning systems were trained using standard procedure except for the Support Vector Machine (SVM).

SVM training was studied using a hyperparameter grid search to obtain the best possible result from the model. The hyperparameters tested were the kernel type, the constant C, and the error ϵ . The kernel types used were radial kernels, linear kernels, and polynomial kernels. The constant C encompassed orders of magnitude, from 10^{-2} to 10^1 . A similar order was followed for ϵ error, from 10^{-2} to 10^0 . The eight models were tested, and the results are reported in Table 11. The results show that all three regression tree algorithms reported the highest R^2 score, compared to the other learning systems. The model coefficients of each principal component are plotted in Figure 15, showing PC 6 to be the most prominent feature.

Table 11: List of the eight ML algorithms along with their respective results.

Weight Algorithms			
Model	Linear model		Multi-Layer Perceptron
R^2	0.7014		0.6977
Regression Trees			
Model	Decision Tree	Random Forest	XGBoost
R^2	0.9039	0.9064	0.9053
Grouping Algorithms			
Model	Support Vector Machine		K-Nearest Neighbors
R^2	0.5355		0.7176

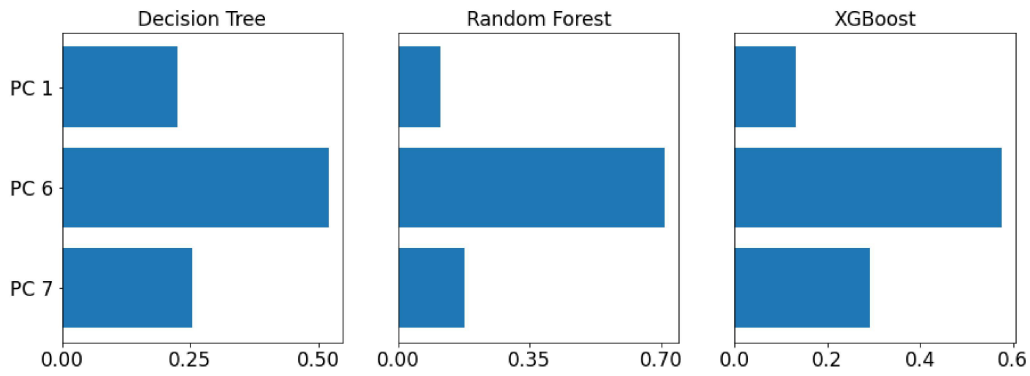


Figure 15: Feature importance for the top three models selected (decision tree, Random Forest, and XGBoost).

A second scoring metric was used to determine if there is a difference among the tree models. Mean Absolute Error (MAE) was chosen because of its robustness to outliers [193]. The MAE results in Table 12 show that the Random Forest model is two times worse than the other decision tree models, so the Random Forest model was eliminated. Similarly, Figures 16-18 display the predicted vs actual scatter plots with the Random Forest model showing different scattering than the other two regression tree models. In addition, the decision tree model was chosen for its interpretability since both models (decision tree and XGBoost) had similar behavior. The coefficients of both models are nearly identical, and the predicted vs actual plots show similar scattering.

Table 12: Comparison of all three regression trees using MAE score.

Model	Decision Tree	Random Forest	XGBoost
R^2	0.9039	0.9064	0.9053
MAE	1284	1366	1284

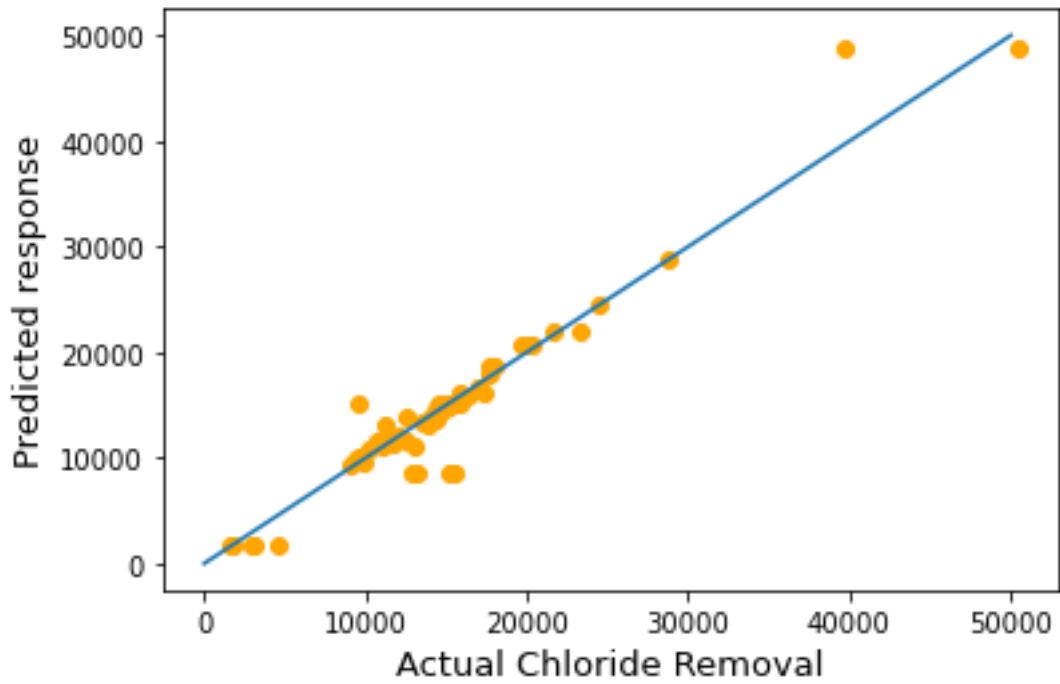


Figure 16: Predicted vs actual scatter plot for decision tree regression model.

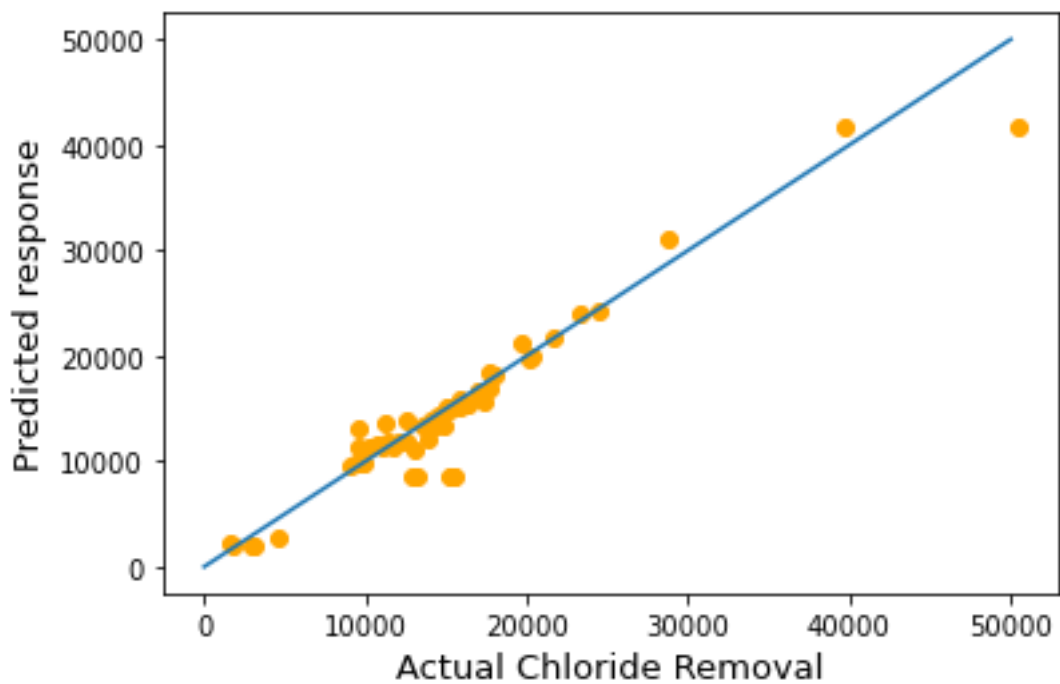


Figure 17: Predicted vs actual scatter plot for Random Forest regression model.

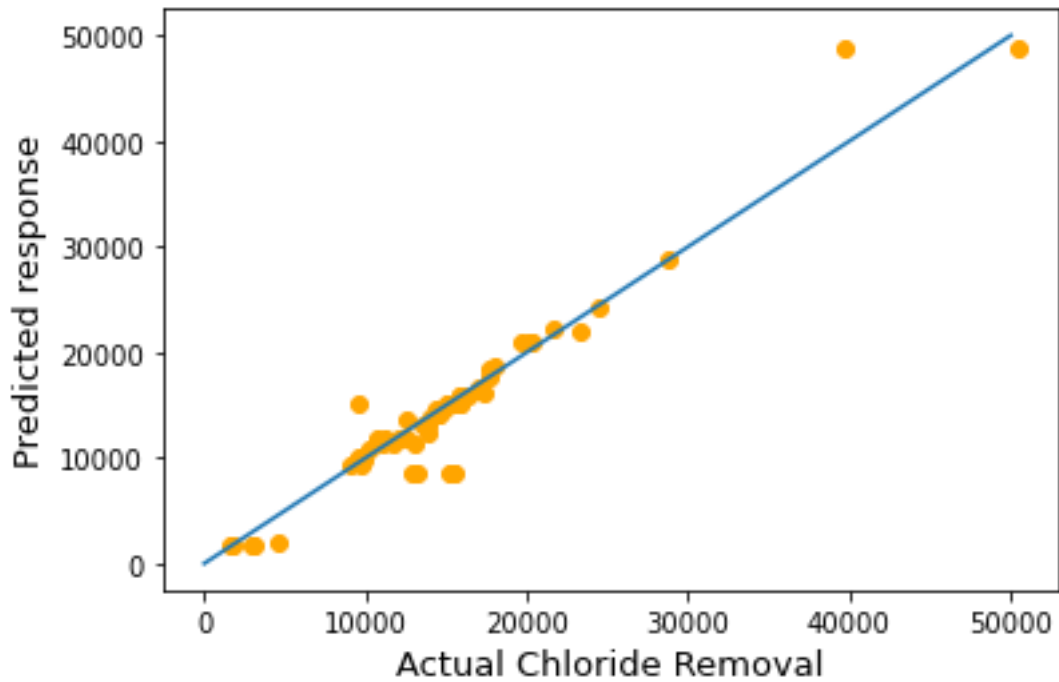


Figure 18: Predicted vs Actual scatter plot for XGBoost regression model.

4.1.4.3 Postprocessing

The decision tree model was shown to be the best Machine Learning algorithm to model chloride-removal using CO₂-Hydrate. However, one factor worth considering is the effectiveness of PCA in the preprocessing phase. So, the input features, initial concentration and operating conditions, were trained/tested on a decision tree regression model without PCA. Table 13 highlights the results of using decision trees along with and without PCA as a preprocessing step. R² and MAE results show the importance of using the PCA preprocessing step. In addition, similar results are shown in the predicted vs actual plots in Figure 19 vs Figure 20. This is expected due to the nonlinear transformation applied when expanding the feature set. Another factor to consider is the anomaly data point found that was incorrectly predicted by the model in Figure 20. The outlier was found to be from a synthetic brine-water sample that had a very low chloride removal percentage (36.3%) [194]. This synthetic solution corresponds to a sample with low electric conductivity. The authors argue that brines with low electric conductivity will generate more clathrates that are smaller in size. This abundance of smaller clathrates will entrap more ions, allowing for less ion removal. Finally, after analyzing

the results, the PCA + decision tree model is ready for salt-removal optimization, as highlighted in Figure 21.

Table 13: Scoring metric comparison for decision tree regression model, with and without PCA.

Model Type	Decision Tree without PCA	Decision Tree with PCA
R^2	0.588	0.904
MAE	3915	1284

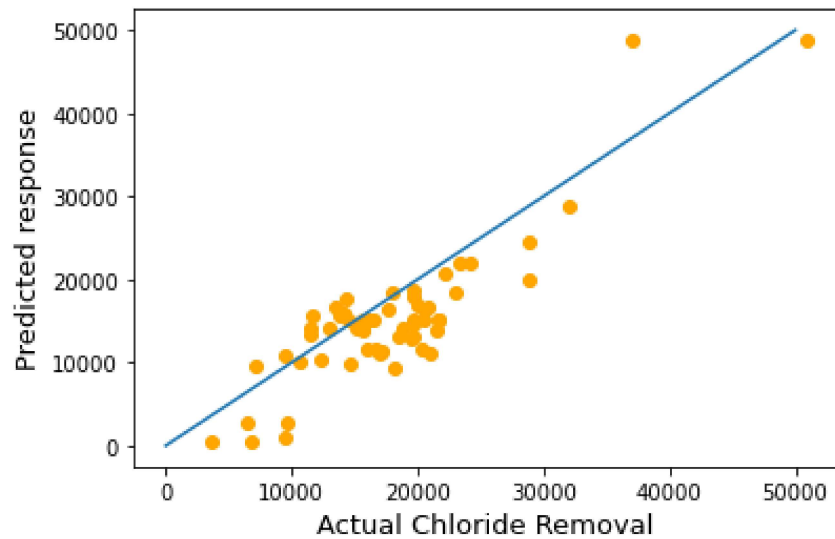


Figure 19: Predicted vs actual scatter plot for decision tree regression model without PCA as a preprocessing step.

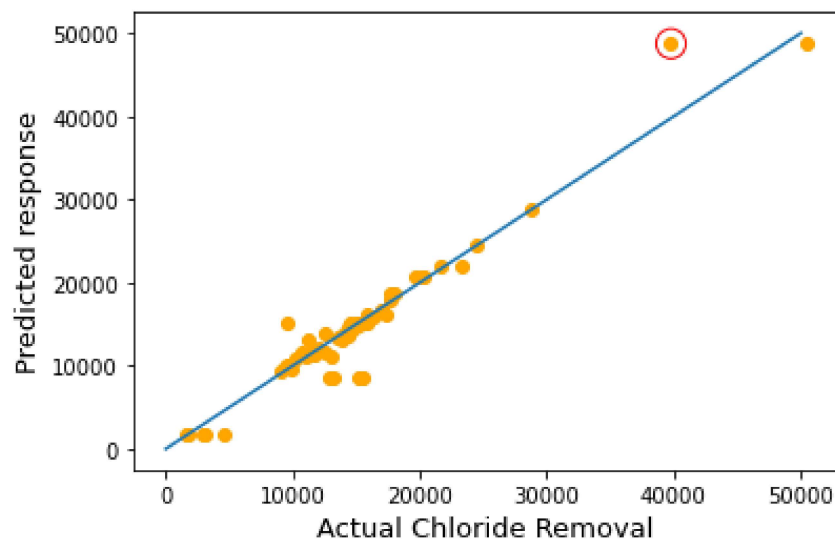


Figure 20: Predicted vs actual scatter plot for decision tree regression model with PCA as a preprocessing step.

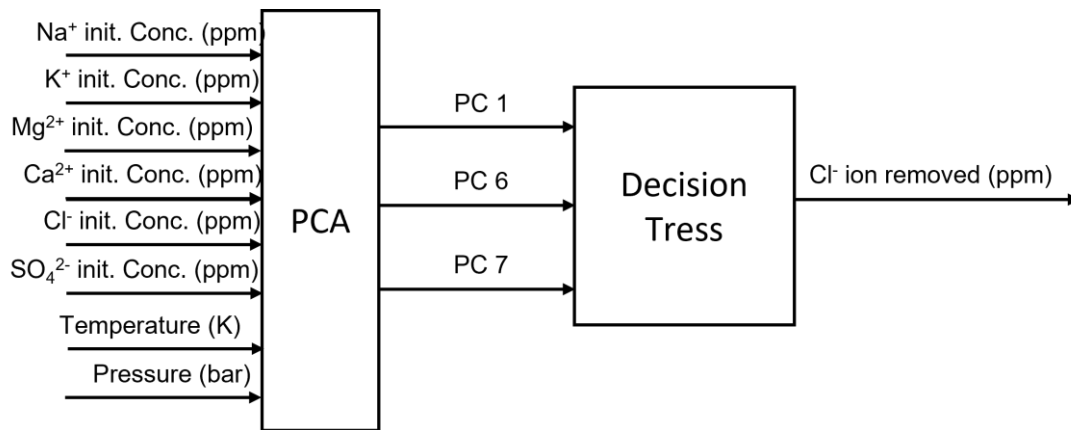


Figure 21: Final Machine Learning system used for Chloride removal modeling.

4.2 Simulation Results

Simulation of Bubble Column Reactors provides the possibility of testing how effective the CO₂ bubbling dynamics are in precipitating dissolved sodium ions. It is important to study the relationship between CO₂ and the reacting absorbent. In addition, it is important to study the effect of operating conditions (pressure, temperature, and pH level) on achieving research goals; minimizing Na⁺ dissolved and maximizing the amount of CO₂ absorbed. The simulation results will emphasize the important factors to consider when bubbling CO₂ in brine solutions.

4.2.1 CO₂ Bubbling in Basic Solution

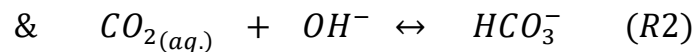
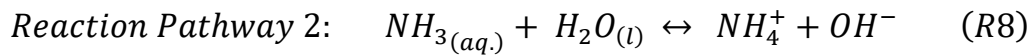
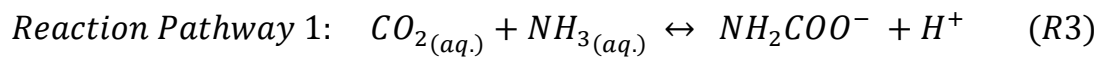
The relationship between CO₂ and the base used requires analyzing the reaction mechanism, CO₂ absorption rate, CO₂ absorption capacity, and possible precipitating salts. Studying the absorption mechanism facilitates determining the best absorbent for CO₂ bubbling. In addition, CO₂ absorption rate and absorption capacity can be optimized for maximum Na⁺ removal. Finally, the type of base used determines the types of salts that can precipitate, which will be important for regeneration and fractional crystallization costs.

4.2.1.1 Reaction Mechanism

The pH level, or the hydrogen concentration, is a key factor for understanding the reaction mechanism. It determines which reaction pathways are taken to produce certain

species. pH dynamics in hydroxide-base solutions behave like a titration process, since NaOH and Ca(OH)₂ are relatively stronger bases. In Figure 22a, the pH level starts at a constant level in a buffer zone, and then, the pH level drops sharply around the equivalence point. Finally, the pH level flattens, as the solution saturates with CO₂ in the form of bicarbonate, mainly. In Figure 22, the reaction time on the x-axis is scaled by dividing over the molar concentration of the base used.

For ammonia, the pH-dynamics graph has a different S-shaped curve due to the affinity of ammonia to dissolved CO₂ in the solution. In the beginning, ammonia reacts with CO₂ to form carbamate and produce H⁺ (R3), which lowers the pH level sharply, as seen in Figure 22b. Then, around the pH level of 9, ammonia dissolves and produces hydroxide ions (R8), which start reacting with CO₂ and producing HCO₃⁻ (R2). This means that higher pH levels will require more time to dissolve ammonia and produce bicarbonate ions, as seen in Figure 23. In the figure, the left end of the pH curve depends on carbamate production, while the right hand depends on ammonia dissolution.



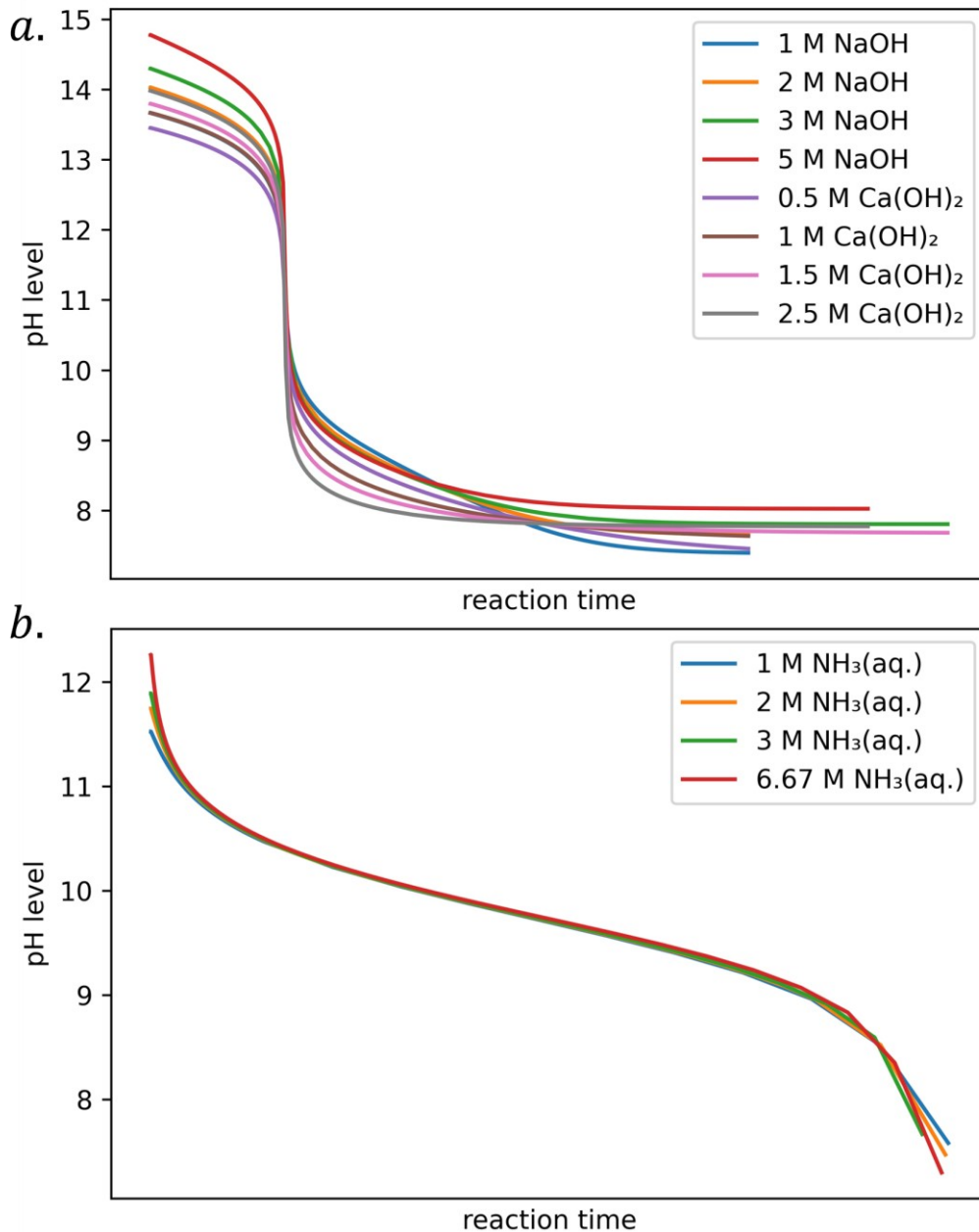


Figure 22: The measure of pH level in CO₂ bubbling in a) hydroxide brine solution and b) ammonia brine solution, operating at temperature of 25°C and pressure of 1 atm.

Ammonia as a base provides two reaction pathways, but the first reaction pathway will produce carbamate ion, an undesired product. Carbamate is produced at the beginning of the reaction and reverts to HCO₃⁻ as the pH drops, shifting from reaction pathway 1 to reaction pathway 2. This indicates that carbamate concentration is a function of pH and its production can be reduced by operating at lower pH levels. Caplow [97] shows that carbamate production and breakdown depends on the pK_b of

ammonia ($pK_b = 4.75$). This suggests that pH levels below 9.25 should be good to operate at.

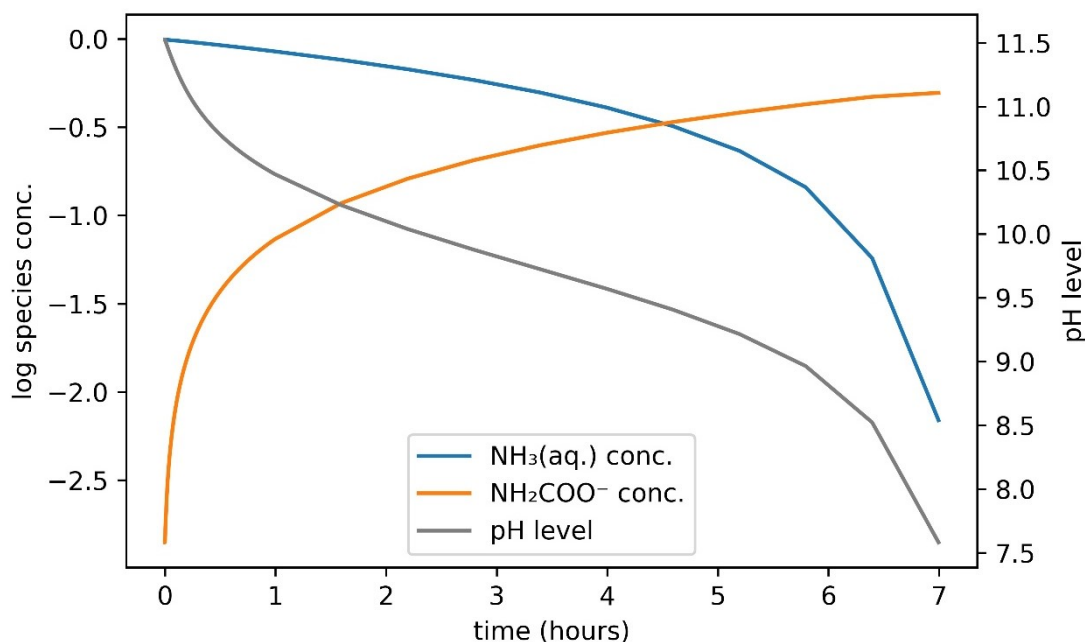


Figure 23: Reaction mechanism of bubbling CO_2 in 1 M ammonia solution at a temperature of 25°C and pressure of 1 atm.

4.2.1.2 CO_2 Absorption Rate

The rate of CO_2 absorption is important for delivering a viable, feasible process. Residence time captures the essence of feasibility in dynamic systems. Higher absorption rates will lower residence time to achieve a feasible solution. Pressure is mainly considered to affect the absorption rate. The amount of CO_2 was differentiated numerically to calculate the absorption rate. As can be seen in Figure 24, the CO_2 initial absorption rate increases rapidly with pressure. Also, the figure shows that the CO_2 absorption rate decreases abruptly over time. The absorption rate in the semi-batch setup can be translated to the absorption efficiency measured in continuous processes. Research shows that high pressure is essential for CO_2 absorption [87], [195].

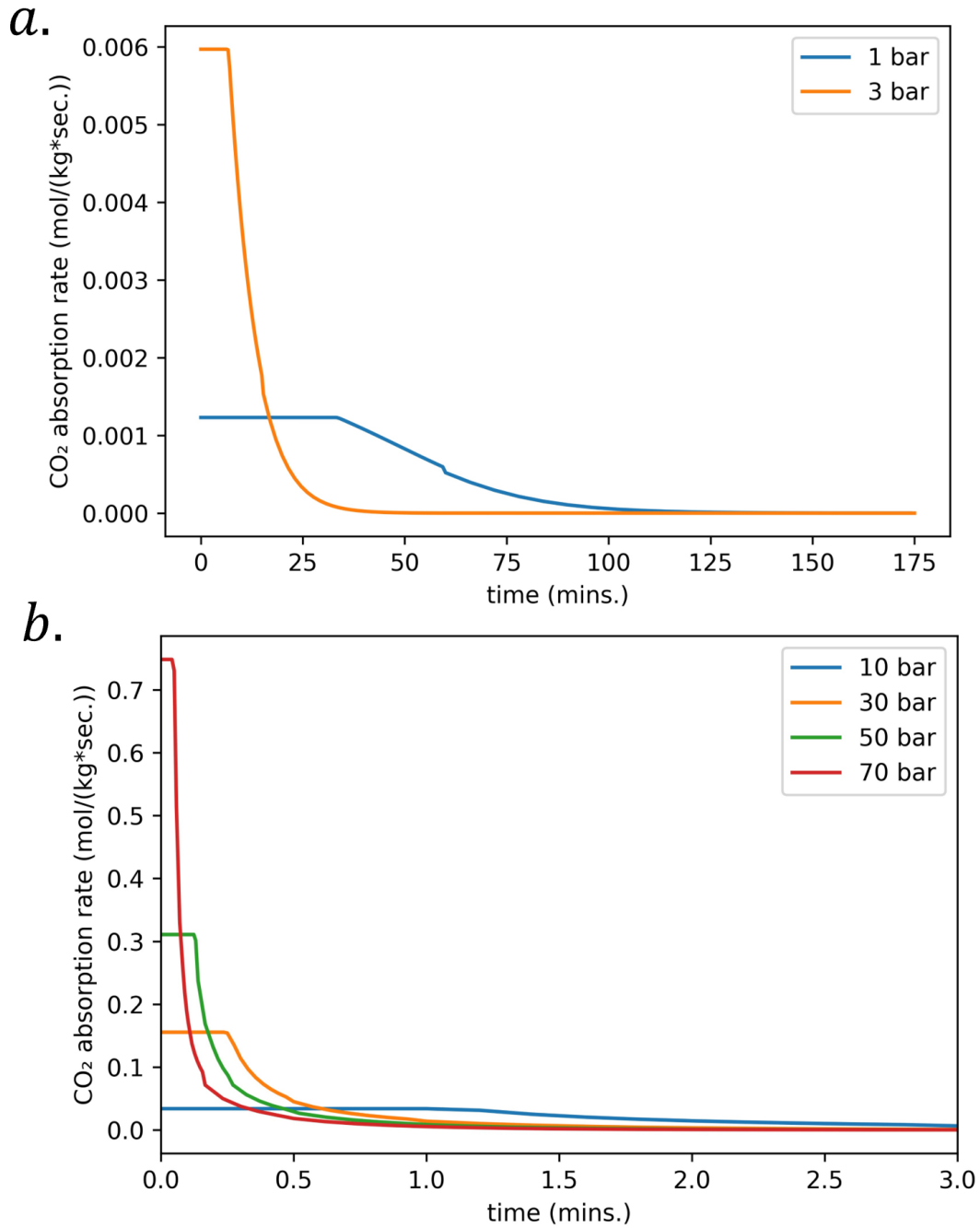


Figure 24: CO₂ absorption rate in 5M NaOH, at a temperature of 25°C, and different pressures.

4.2.1.3 CO₂ Absorption Capacity

CO₂ absorption capacity is important to ensure maximum Na⁺ removal, which is the first objective of this research. The absorption capacity is also important because the second objective of this research, maximize CO₂ capture. The absorption capacity is determined by two factors: temperature and type of base used. In the simulations, it was

found that temperature does not have a huge effect on absorption capacity. From Figure 25, there is a difference of about 0.2 mol of CO₂ absorbed over a temperature range of 15 – 40°C [84], [196]. In addition, the curvature of the absorption capacity curve (T= 15°C), compared to the other temperatures, shows that lower temperatures will require more time to saturate. Experimental work on primary/tertiary amines shows similar results [197]. Figure 26 highlights the effects of the type of base used on the absorption capacity. Sodium hydroxide provides the maximum absorption capacity among the bases used, NaOH, Ca(OH)₂, and NH₃. This is due to the strength of the base and its affinity towards CO₂ [198]. Also, Figure 26 uses different molarities for each absorbent to ensure similar OH⁻ concentration is maintained.

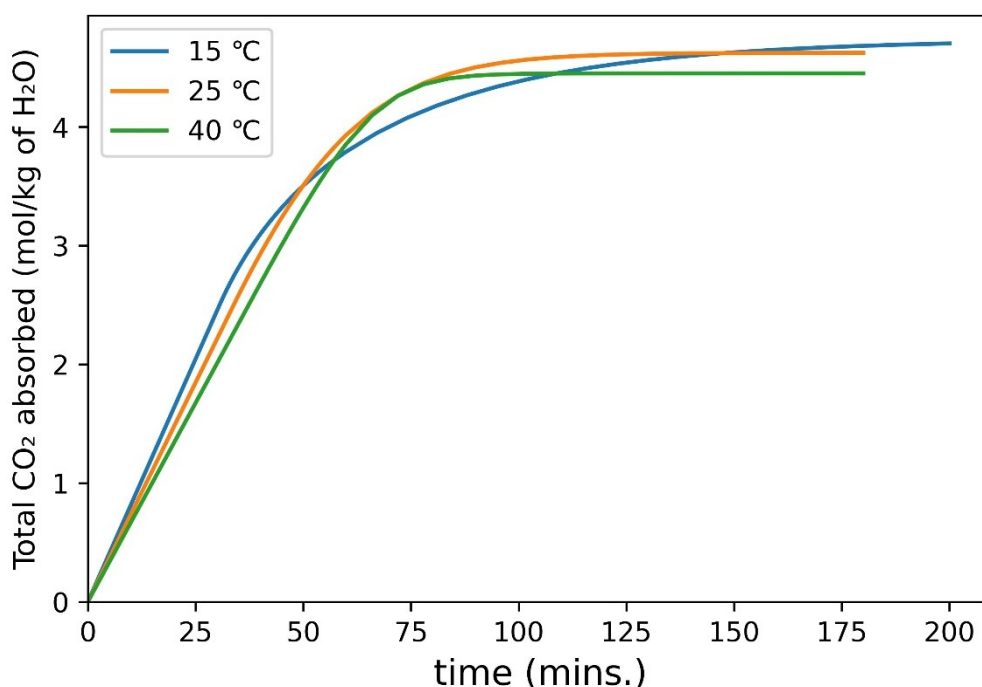


Figure 25: CO₂ absorption capacity in 5M NaOH, at a pressure of 1 atm, and different temperatures.

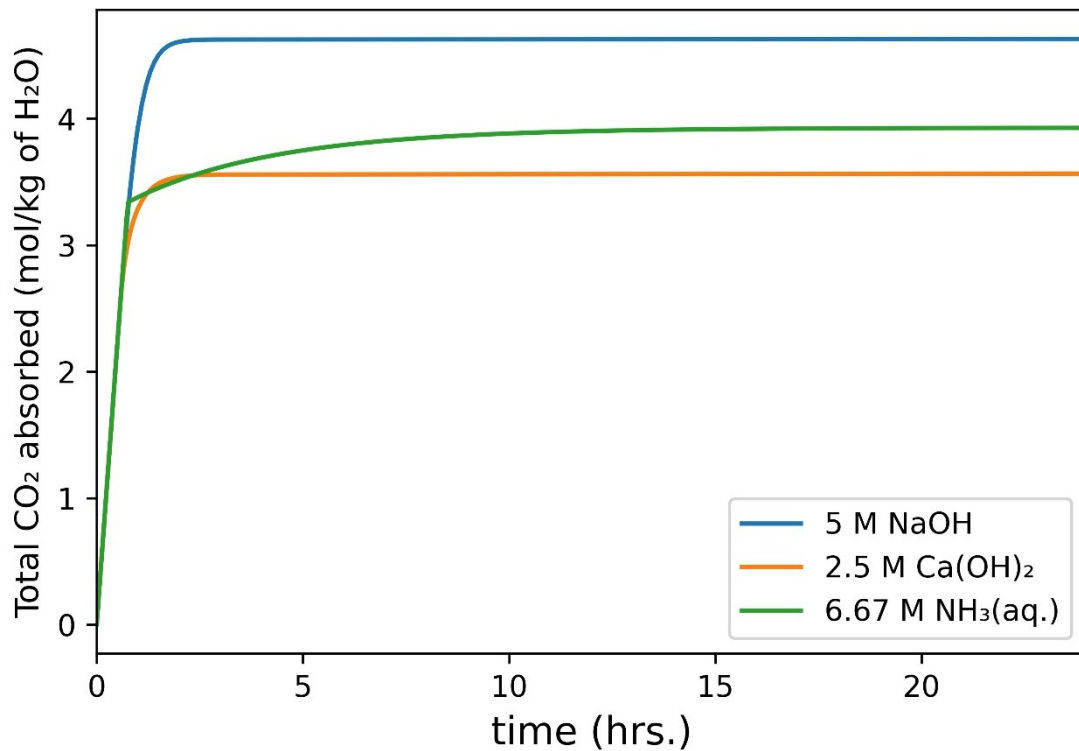
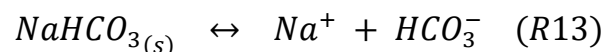


Figure 26: CO₂ absorption capacity with the absorbent as a parameter, at a temperature of 25°C and pressure of 1 atm.

For CO₂ absorption capacity, a more important factor is the aqueous CO₂ to HCO₃⁻ ratio. It is important to dissolve more CO₂ and maximize absorption capacity, but what must not be ignored, is the amount of Na⁺ ions removed. NaHCO₃ precipitation depends on the HCO₃⁻ concentration. The higher the concentration of HCO₃⁻ is, the higher the amount of Na⁺ ions removed, according to le Chatelier's principle (R13). Maximum HCO₃⁻ concentration is achieved by maximizing CO₂ absorption capacity and operating at a certain pH level. Simulations with ionic activity in water show that a pH level around 7.3 is the best pH level to maximize NaHCO₃ precipitation, as seen in Figure 27. Carbonic speciation as a function of pH depends on the amount of CO₂ dissolved. HCO₃ maximum relative concentration is around pH level of 8 in low amounts of CO₂ dissolved (CO_{2,tot} = 1 mmol) [199], and the maximum relative concentration lowers pH level to around 7.5 at high concentration (CO_{2,tot} = 1 molal) [200].



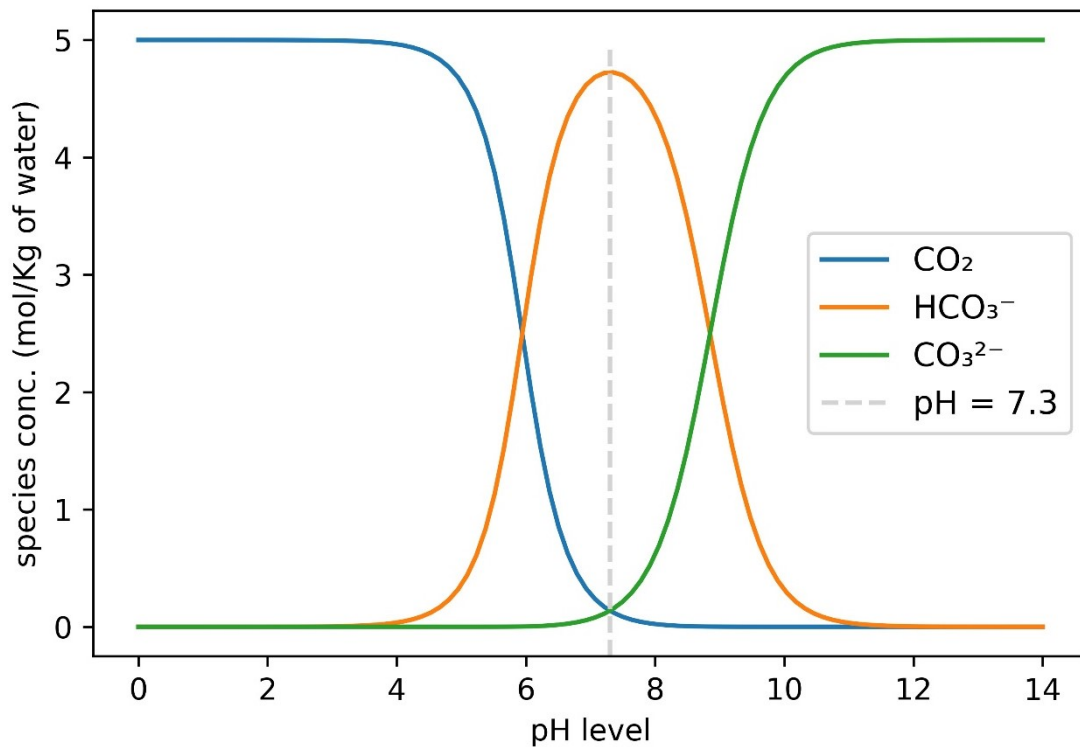


Figure 27: 5 molal of dissolved CO₂ and its speciation in water as a function of pH level, at a temperature of 25°C and pressure of 1 atm.

4.2.1.4 Salt Precipitation

CO₂ bubbling in hydroxide-based brine solutions can coprecipitate undesired salts, while CO₂ bubbling in ammonia will produce undesired species, carbamate ions. In NaOH brine solutions, CO₂ bubbling will precipitate NaHCO₃ and coprecipitate CaCO₃. In Ca(OH)₂ brine solutions, CO₂ bubbling will precipitate CaCO₃ along with CaSO₄·2H₂O. This can be seen in Figure 28a and has been reported in the literature [201], [202]. Fractional crystallization will be an important factor to determine the best absorbent for a sustainable ZLD process. The working principle behind fractional crystallization is exploiting a difference in solubility to precipitate the least soluble salt first. In the case of NaOH, the solubility product of NaHCO₃ ($K_{sp}= 0.4$) is orders of magnitude higher than that of CaCO₃ ($K_{sp}= 3.3e-9$), at 25°C. This makes the crystallization process feasible to accomplish. For the case of Ca(OH)₂, the solubilities of the co-precipitated salts (CaCO₃ and CaSO₄·2H₂O) are within the same order of magnitude ($K_{sp}= 3.3e-9$ and $2.5e-5$, respectively), making it infeasible to separate the

salts [203]. Figure 28 highlights the different salts that can precipitate from using hydroxides.

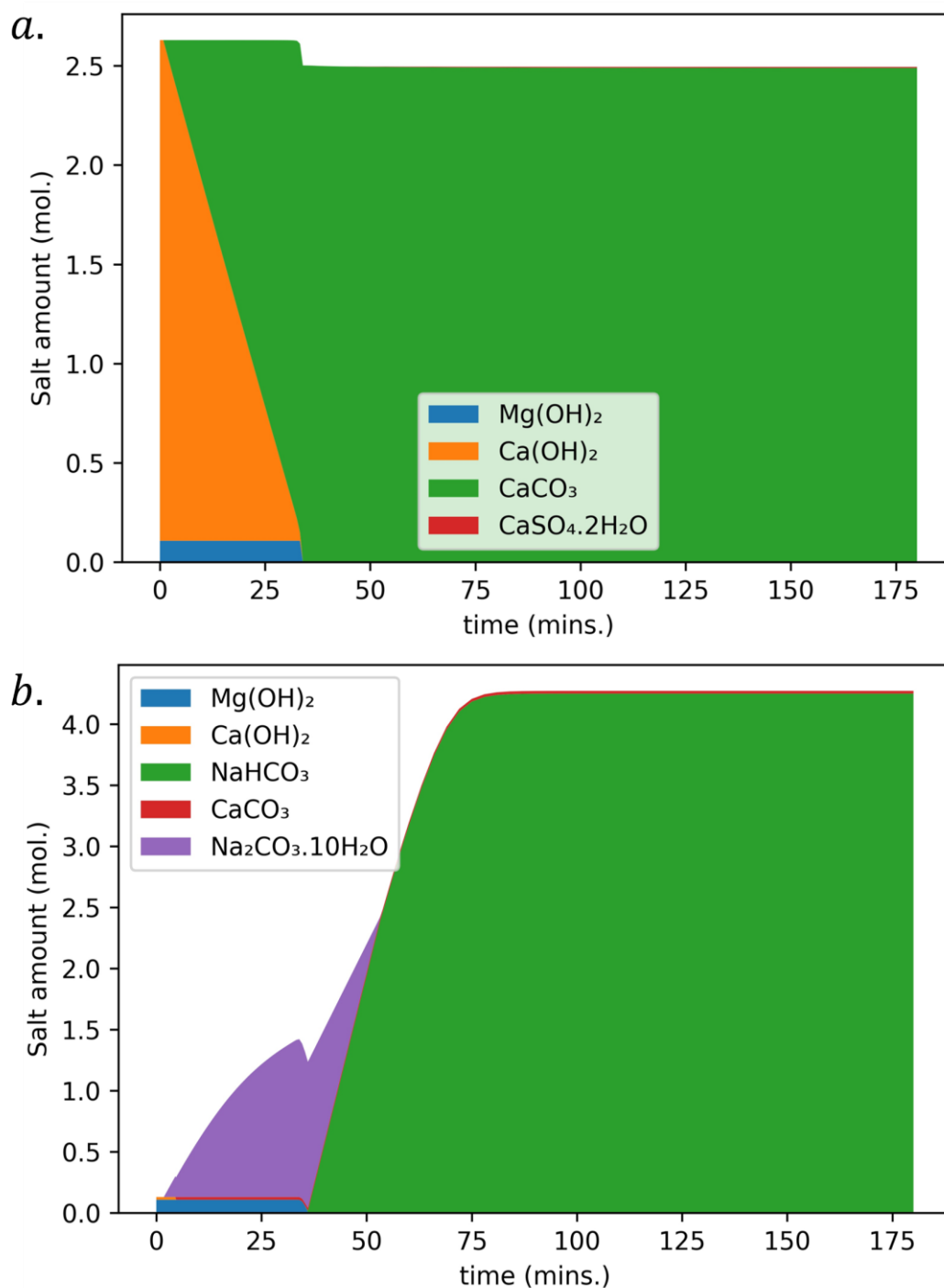


Figure 28: Salt precipitation in hydroxides, at a temperature of 25°C and pressure of 1 atm, in a) 2.5M of Ca(OH)_2 and b) 5M of NaOH .

4.2.2 Operating Condition Effects

This section summarizes the results drawn from the previous section, as it provides context into the appropriate operating conditions. Simulation results were studied to understand the underlying operating conditions for maximum Na^+ removal and CO_2 capture, the first two objectives of this study.

4.2.2.1 Temperature

The operating temperature of the Bubble Column Reactor can and will affect CO_2 absorption, aqueous chemical reactions, and salt precipitation reactions. Figure 29 shows key factors and the effect of temperature on them. The plots show that lower temperatures are important for sodium removal and CO_2 absorption capacity (CO_2 solubility), while higher temperatures are better for the absorption rate (CO_2 mass transfer coefficient). In Figure 30, the CO_2 absorption rate is initially higher at lower temperature ($T=15^\circ\text{C}$), but takes longer to reach full capacity, at zero. Higher temperature ($T=40^\circ\text{C}$) has a smaller initial value, yet a steeper curve that reaches zero faster. So, temperature (15 – 40°C) does not play an important role in maximizing CO_2 absorption [84].

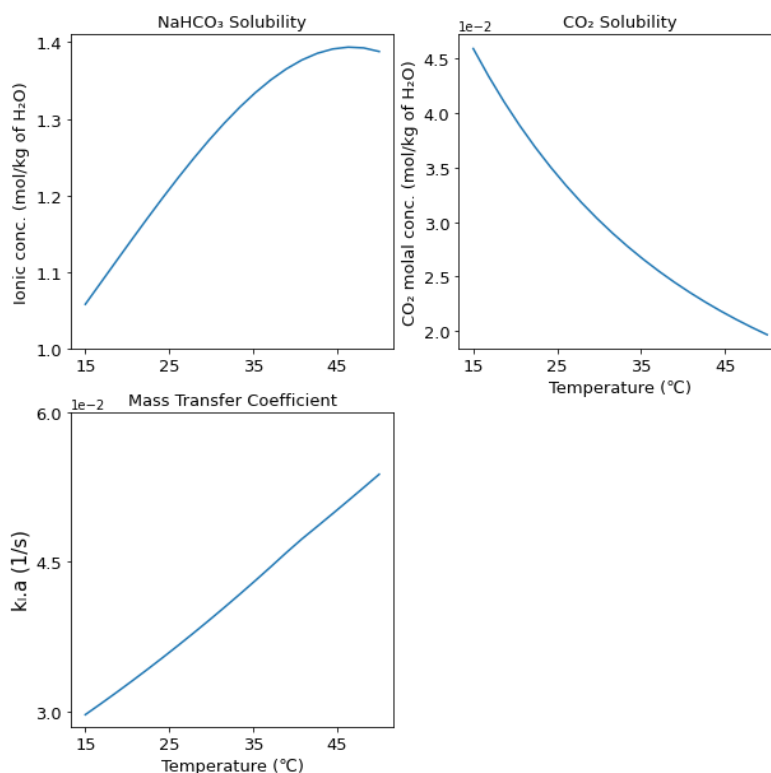


Figure 29: Key factors and their effects on temperature.

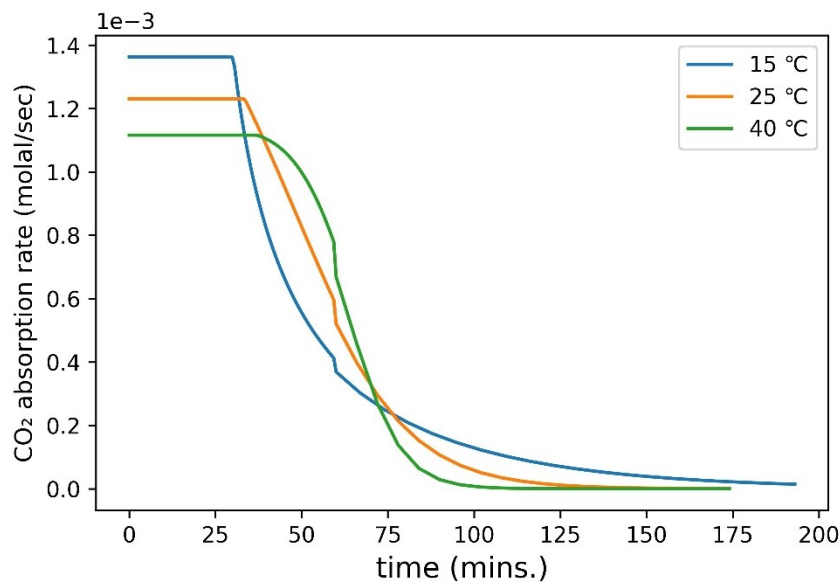


Figure 30: CO₂ absorption rate in 5M NaOH at 1 bar and different temperatures.

4.2.2.2 Pressure

The operating pressure mainly affects the CO₂ absorption rate and CO₂ absorption capacity. High pressures will increase the absorption rate rapidly [195], going from three hours (1 bar) down to around 1 minute (70 bar). In addition, CO₂ absorption capacity increases with pressure, but starts to have diminishing returns, in the range of 30 bar to 70 bar. Figure 24b highlights the effect of high pressure on CO₂ solubility.

4.2.2.3 pH Level

The pH level was found to be important for the HCO₃⁻ to CO_{2(aq)} ratio, and when using ammonia as a base. In Figure 27, HCO₃⁻ relative concentration is maximum around a pH level of 7.3 for 5 M aqueous CO₂ dissolved at a temperature of 25°C and a pressure of 1 bar. Also, the pH level is important when using NH₃ as an absorbent. At high pH levels, NH₃ reacts with CO₂ through the first reaction pathway to produce NH₂COO⁻ as a by-product. At lower pH levels, NH₃ dissolves to produce OH⁻, which then reacts with CO₂ to produce HCO₃⁻, reaction pathway 2. Simulation results obtained by Caplow [97] showed the optimum pH level to operate is at the dissociation constant of NH₃ (pK_b = 4.76), below the pH level of 9.25

4.2.2.4 Strength of Base

There are two parts to consider when choosing the right absorbent: the type of base, and the concentration of the base used. NaOH could be a better absorbent than $\text{Ca}(\text{OH})_2$ based on the types of solids it can precipitate. NaOH precipitates NaHCO_3 and CaCO_3 , while CaCO_3 precipitates CaCO_3 and $\text{CaSO}_4 \cdot 2\text{H}_2\text{O}$. This will be an important factor, if fractional crystallization costs were to be considered. Also, NaOH could be a better absorbent than NH_3 based on CO_2 absorption capacity, as highlighted in Figure 26. Finally, the concentration of the base will be an important factor, as higher concentrations are needed to absorb huge amounts of CO_2 [84], [195], [196]. Figure 31 shows the effect of concentration on the absorbent (NaOH) at a temperature of 25°C and a pressure of 1 atm. A similar trend can be observed in the other bases ($\text{Ca}(\text{OH})_2$ and NH_3).

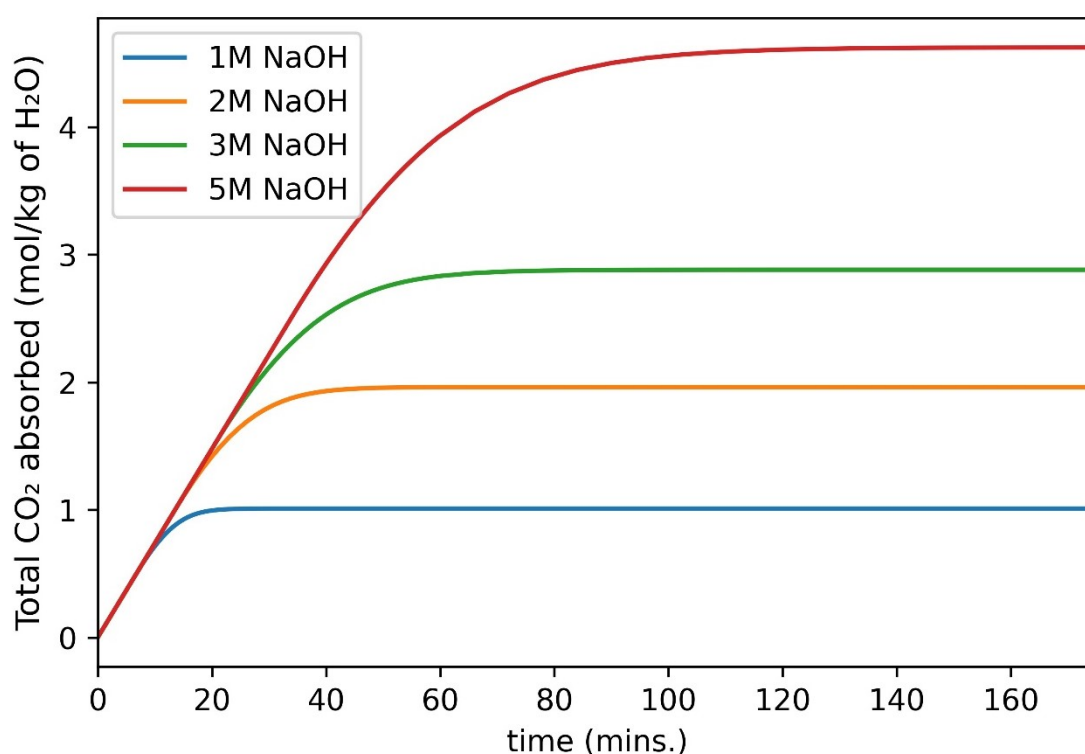


Figure 31: CO_2 absorption capacity with NaOH concentration as a parameter, operating at a temperature of 25°C and pressure of 1 atm.

4.3 Optimization of Solid Precipitation

CO_2 absorption in Bubble Column Reactor was modeled and optimized using non-linear programming, while the CO_2 -hydrate formation was modeled and optimized

using Machine Learning. The CO₂ bubbling process is optimized first, followed by the CO₂-hydrate process optimization. A final simulation of CO₂ bubbling at the optimum conditions is presented.

4.3.1 Base Comparison

There are three criteria chosen to compare the absorbents considered: percentage of Na⁺ removal, the amount of CO₂ absorbed, and the ability to separate by-products. The optimization work includes all the mathematical equations used in building the simulations. These equations include CO₂ absorption using mass transfer coefficient, CO₂ reaction with the absorbent used, aqueous thermodynamic reactions, and solid precipitation reactions. In addition, both phases, gaseous and aqueous, account for non-ideality using fugacity coefficients. The results from the optimization study reveal that only NH₃ was able to reduce the sodium content (72.5%). Also, it was found that ammonia precipitates NaHCO₃ in slightly acidic media (pH = 6.25), when operating at a high pressure of 70 bar. On the other hand, NaOH was able to precipitate Na⁺ ions, but the base leaves more Na⁺ content than what was originally present in brine solutions. The results showed 1.545 moles left vs 1.009 moles initially present (-53.1%), when operating at 15°C and 70 bar. Within the optimization boundaries, Ca(OH)₂ was not able to precipitate any Na⁺ ions because of replacement by chemical affinity [201], [204]. The initial concentrations of all the ions used in the optimization work are listed in Table 14. The main observations found are highlighted in Table 15.

Table 14: Initial concentration of the ions considered in the CO₂ bubbling process.

Ion	Na ⁺	K ⁺	Mg ²⁺	Ca ²⁺	Cl ⁻	SO ₄ ²⁻
Initial conc. (ppm)	23200	808	2610	890	44000	6090

Table 15: Conclusions made from the optimization work.

NaOH	Ca(OH) ₂	NH ₃ (aq.)
<ul style="list-style-type: none"> • NaHCO₃ ppt. (-53.1%) • Defeats the purpose 	<ul style="list-style-type: none"> • No ppt of NaHCO₃ • Only CaCO₃ 	<ul style="list-style-type: none"> • NaHCO₃ ppt. (72.5%) • ppt in slightly acidic media

4.3.2 Simulation of Optimum Conditions

The results from the optimization work were used to simulate the best absorbent with the optimum conditions. CO₂ bubbling using ammonia dissolves a total of 8.702 moles of CO₂ in which 1.15 moles of NH₃ dissolved were used per mole of CO₂ absorbed. The process was able to precipitate 0.732 moles of Na⁺ ions (72.5%), operating at 22.2°C and 68.7 bars. In addition, the amount of ammonia used was dissolved over 10 seconds to ensure the pH level does not increase rapidly. The plot of the final simulation with the optimum conditions is shown in Figure 32.

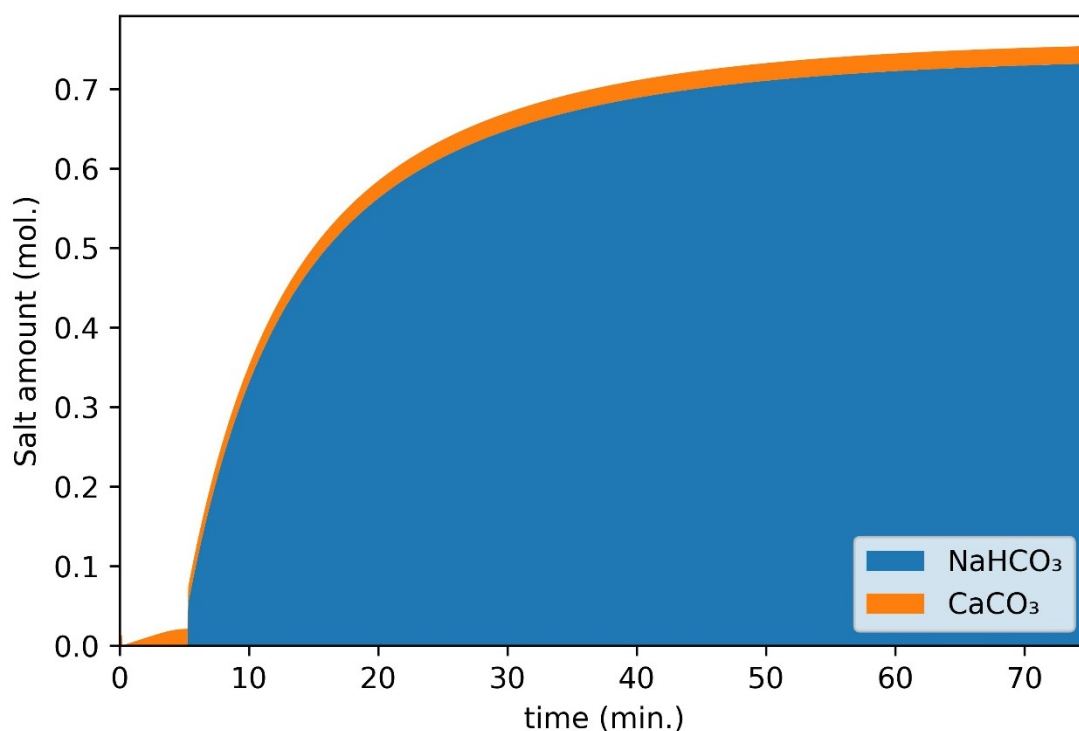


Figure 32: Simulation of salt precipitation in bubble column reactor, operating at 22.2°C and 68.7 bars.

4.3.3 CO₂-Hydrate Optimization

The PCA + decision tree model was applied to predict Cl⁻ removal for the CO₂-hydrate formation process. The CO₂-hydrate formation depends on CO₂ bubbling, and the bubbling will precipitate the dissolved Na⁺ ions. A hypothetical situation to consider is the CO₂-hydrate formation before the Na⁺ ions precipitation. If the results from the hypothetical situation were drastically better, then the operating conditions would have to be optimized for that hypothetical situation. So, there are two situations considered,

before NaHCO_3 precipitation and after the precipitation in ammonia solution. Using the initial conditions in Table 14, chloride removal amounted to 25083 ppm. Using the final conditions of 72.5% Na^+ removal, chloride removal was equal to 23828 ppm. Since the results show little to no difference, the optimum process sequence would be the CO_2 bubbling process followed by the CO_2 -hydrate formation process. The overall process removes 72.5% of Na^+ ions, and 54.2% of Cl^- ions dissolved.

Chapter 5: Conclusion

Water Scarcity will become a bigger concern, as nations develop, urbanize, and the population grows. Desalination is our only hope for surviving and prospering as nations. Desalination is very costly, and as capacities grow exponentially, revitalizing the brine waste produced will be essential. Zero-Liquid Discharge (ZLD) technology will mitigate environmental problems while driving the cost of desalination down. While many research institutions and governments have been focusing on improving membrane efficiency, desalination costs can go down by revitalizing the waste produced. The current treatment of reject brine utilizes evaporators and crystallizers to achieve Zero Liquid discharge. This research provides a path to minimize desalination costs by chemical precipitation and hydrate formation, since the dissolved-salt concentrations are high enough.

The literature surveyed common brine treatment solutions, along with conventional CO₂ absorption techniques. For brine treatment solutions, emerging technologies are the only pathway toward ZLD. The best emerging technologies are Forward Osmosis and Membrane Distillation. FO relies on the draw solution and the type of membrane used, and MD combines a thermal-based method with a membrane-based technology. There has been some commercial success, yet nothing can replace the conventional method for ZLD. For carbon capture techniques, three absorbents were considered (NH₃, NaOH, and Ca(OH)₂). Hydroxide absorbents have low absorption costs, while amines have lower regeneration costs. The dual-alkali process could be cheaper than the thermal regeneration of the absorbent. In addition, absorption columns should be operated at high pressures and low temperatures, but temperatures should be higher than and pressures should be lower than the condensation point.

The theoretical framework introduced the essential equations for modeling multiphase phenomena in a Bubble Column Reactor. Absorption was modeled using a mass transfer coefficient, it was combined with the kinetic reactions. The reaction system combined the kinetic and thermodynamic reactions into a system of Differential Algebraic Equations. These equations were simulated and optimized using non-linear programming. Hydrate formation for chloride removal was modeled using Machine

Learning. Experimental data was collected from CO₂-based hydrate systems that reported the initial concentration, final concentration, and operating condition. Eight algorithms were considered: two weighting systems (linear regression and Multi-Layer Perceptron), three regression trees (decision tree, Random Forest, and XGBoost), and three grouping methods (Principal Component Analysis, Support Vector Machine, and k-Nearest Neighbors).

The simulation and optimization results provided the best operating conditions for the Bubble Column Reactor. The dynamic simulations showed the effect of pH level on bicarbonate production. Also, the simulations showed that high pressures were essential, while the temperature did not have a significant effect (15 – 50°C). The machine learning optimization showed that the PCA preprocessing step was essential, and the decision tree was the best regression model. The optimization results indicated that ammonia was the best absorbent, being able to remove 72.5% of Na⁺ ions. The process will absorb 0.87 mol of CO₂ per mole of NH₃ dissolved with a total of 8.702 mol of CO₂ absorbed. This will allow the hydrate formation process to remove 54.2% of Cl⁻ ions.

This body of research provided a pathway for using chemical precipitation to remove Na⁺ and Cl⁻ ions in reject-brine solutions. One future research direction can focus on ammonia escape in Bubble Column Reactor. Research shows that CO₂ bubbling can act as a driving force, diffusing NH₃ into the bubbles and out of the solution [205]. In addition, another important research direction is to study CO₂ interaction with ammonia in the hydrate formation process. Ammonia can penetrate CO₂-water clathrate formed, affecting ionic interactions and water recovery [206]. Finally, there can be some focus on optimizing/maximizing the salt recovery process for the partial crystallization process. The intensification of the system will allow for better salt recovery and a more sustainable solution [207].

References

- [1] “Desalination has increasingly become a viable option to close a water supply/demand gap,” in *The role of desalination in an increasingly water-scarce world*, World Bank, pp. 5–12. 2019, doi: 10.1596/31416.
- [2] E. Jones, M. Qadir, M. T. H. van Vliet, V. Smakhtin, and S. mu Kang, “The state of desalination and brine production: A global outlook,” *Sci. Total Environ.*, vol. 657, pp. 1343–1356, Mar. 2019, doi: 10.1016/J.SCITOTENV.2018.12.076.
- [3] M. A. Maupin, “Total water use,” in *Summary of estimated water use in the United States in 2015*, Reston, VA, pp. 7–17, 2018. doi: 10.3133/fs20183035.
- [4] S. Igor, “World fresh water resources,” in *Water in crisis: A guide to the world’s fresh water resources*, Oxford, UK: Oxford University Press, Inc, pp. 13–24, 1993.
- [5] A. Boretti and L. Rosa, “Reassessing the projections of the world water development report,” *npj Clean Water*, vol. 2, no. 15, pp. 1–6, Jul. 2019, doi: 10.1038/s41545-019-0039-9.
- [6] M. E. Mann and P. H. Gleick, “Climate change and California drought in the 21st century,” *Proc. Natl. Acad. Sci.*, vol. 112, no. 13, pp. 3858–3859, Mar. 2015, doi: 10.1073/pnas.1503667112.
- [7] S. Pascale, S. B. Kapnick, T. L. Delworth, and W. F. Cooke, “Increasing risk of another Cape Town ‘Day Zero’ drought in the 21st century,” *Proc. Natl. Acad. Sci.*, vol. 117, no. 47, pp. 29495–29503, Nov. 2020, doi: 10.1073/pnas.2009144117.
- [8] R. Bedawy, “Water resources management: Alarming crisis for Egypt,” *J. Manag. Sustain.*, vol. 4, no. 3, pp. 108–124, Aug. 2014, doi: 10.5539/jms.v4n3p108.
- [9] G. Nilanjan, “Water-scarce economies and scarcity values: Can water futures trading combat water scarcity?,” pp. 1–37, 2022.
- [10] W. Barnaby, “Do nations go to war over water?,” *Nature*, vol. 458, no. 7236, pp. 282–283, Mar. 2009, doi: 10.1038/458282a.
- [11] J. Eliasson, “The rising pressure of global water shortages,” *Nature*, vol. 517, no. 7532, pp. 6–6, Dec. 2014, doi: 10.1038/517006a.
- [12] A. Siddiqi and L. D. Anadon, “The water–energy nexus in Middle East and North Africa,” *Energy Policy*, vol. 39, no. 8, pp. 4529–4540, Aug. 2011, doi: 10.1016/J.ENPOL.2011.04.023.
- [13] A. M. Hamiche, A. B. Stambouli, and S. Flazi, “A review of the water-energy nexus,” *Renew. Sustain. Energy Rev.*, vol. 65, pp. 319–331, Nov. 2016, doi: 10.1016/J.RSER.2016.07.020.
- [14] M. S. M. Alasam Alzaabi and T. Mezher, “Analyzing existing UAE national water, energy and food nexus related strategies,” *Renew. Sustain. Energy Rev.*, vol. 144, no. C, Art. no. 111031, 2021, doi: 10.1016/j.rser.2021.111031.

- [15] H. C. Moog, F. Bok, C. M. Marquardt, and V. Brendler, “Disposal of nuclear waste in host rock formations featuring high-saline solutions – Implementation of a thermodynamic reference database (THEREDA),” *Appl. Geochemistry*, vol. 55, pp. 72–84, Apr. 2015, doi: 10.1016/J.APGEOCHEM.2014.12.016.
- [16] O. A. Hamed, “Overview of hybrid desalination systems — current status and future prospects,” *Desalination*, vol. 186, no. 1–3, pp. 207–214, Dec. 2005, doi: 10.1016/j.desal.2005.03.095.
- [17] N. T. Dung, R. Hay, A. Lesimple, K. Celik, and C. Unluer, “Influence of CO₂ concentration on the performance of MgO cement mixes,” *Cem. Concr. Compos.*, vol. 115, no. 0, Art. no.103826, 2021, doi: 10.1016/j.cemconcomp.2020.103826.
- [18] M. H. El-Naas, A. H. Al-Marzouqi, and O. Chaalal, “A combined approach for the management of desalination reject brine and capture of CO₂,” *Desalination*, vol. 251, no. 1–3, pp. 70–74, Feb. 2010, doi: 10.1016/J.DESAL.2009.09.141.
- [19] IWA, “The reuse opportunity,” pp. 1–24, 2018.
- [20] I. C. Karagiannis and P. G. Soldatos, “Water desalination cost literature: Review and assessment,” *Desalination*, vol. 223, no. 1–3, pp. 448–456, Mar. 2008, doi: 10.1016/J.DESAL.2007.02.071.
- [21] Center for Sustainable Systems - University of Michigan, “U.S. wastewater treatment factsheet,” pp. 1–2, 2021.
- [22] J. Wang and S. Wang, “Removal of pharmaceuticals and personal care products (PPCPs) from wastewater: A review,” *J. Environ. Manage.*, vol. 182, pp. 620–640, Nov. 2016, doi: 10.1016/J.JENVMAN.2016.07.049.
- [23] Y. Hu, M. Gong, J. Wang, and A. Bassi, “Current research trends on microplastic pollution from wastewater systems: A critical review,” *Rev. Environ. Sci. Bio/Technology 2019 182*, vol. 18, no. 2, pp. 207–230, Apr. 2019, doi: 10.1007/S11157-019-09498-W.
- [24] J. Drewnowski, A. Remiszewska-Skwarek, S. Duda, and G. Łagód, “Aeration process in bioreactors as the main energy consumer in a wastewater treatment plant. Review of solutions and methods of process optimization,” *Processes*, vol. 7, no. 5, pp. 311–332, May 2019, doi: 10.3390/PR7050311.
- [25] A. Zapata-Sierra, M. Cascajares, A. Alcayde, and F. Manzano-Agugliaro, “Worldwide research trends on desalination,” *Desalination*, vol. 519, no. 0, Art. no. 115305, 2021, doi: 10.1016/j.desal.2021.115305.
- [26] H. Cooley, R. Phurisamban, and P. Gleick, “The cost of alternative urban water supply and efficiency options in California,” *Environ. Res. Commun.*, vol. 1, no. 4, Art. no. 42001, 2019, doi: 10.1088/2515-7620/ab22ca.
- [27] Y. Zhou and R. S. J. Tol, “Evaluating the costs of desalination and water transport,” *Water Resour. Res.*, vol. 41, no. 3, pp. 1–10, Mar. 2005, doi: 10.1029/2004WR003749.

- [28] P. S. Goh, T. Matsuura, A. F. Ismail, and N. Hilal, "Recent trends in membranes and membrane processes for desalination," *Desalination*, vol. 391, pp. 43–60, Aug. 2016, doi: 10.1016/J.DESAL.2015.12.016.
- [29] D. Davenport, M. A. Deshmukh, J. R. Werber, and M. Elimelech, "High-pressure reverse osmosis for energy-efficient hypersaline brine desalination: Current status, design considerations, and research needs," *Environ. Sci. Technol. Lett.*, vol. 5, no. 8, pp. 467–475, Aug. 2018, doi: 10.1021/ACS.ESTLETT.8B00274.
- [30] W. Yu, D. Song, W. Chen, and H. Yang, "Antiscalants in RO membrane scaling control," *Water Res.*, vol. 183, no. 0, Art. no. 115985, Sep. 2020, doi: 10.1016/J.WATRES.2020.115985.
- [31] S. K. Patel, P. M. Biesheuvel, and M. Elimelech, "Energy consumption of brackish water desalination: Identifying the sweet spots for electrodialysis and reverse osmosis," *ACS ES&T Eng.*, vol. 1, no. 5, pp. 851–864, May 2021, doi: 10.1021/ACSESTENGG.0C00192.
- [32] T. Tong and M. Elimelech, "The global rise of Zero Liquid Discharge for wastewater management: Drivers, technologies, and future directions," *Environ. Sci. Technol.*, vol. 50, no. 13, pp. 6846–6855, Jul. 2016, doi: 10.1021/ACS.EST.6B01000.
- [33] V. V. Ranade and V. M. Bhandari, "Industrial wastewater treatment, recycling, and reuse: An overview," *Ind. Wastewater Treat. Recycl. Reuse*, pp. 1–80, Jan. 2014, doi: 10.1016/B978-0-08-099968-5.00001-5.
- [34] Muhammad Yaqub and W. Lee, "Zero-liquid discharge (ZLD) technology for resource recovery from wastewater: A review," *Sci. Total Environ.*, vol. 681, pp. 551–563, Sep. 2019, doi: 10.1016/J.SCITOTENV.2019.05.062.
- [35] A. A.-H. I. Mourad, A. F. Mohammad, M. Altarawneh, A. H. Al-Marzouqi, M. H. El-Naas, and M. H. Al-Marzouqi, "Effects of potassium hydroxide and aluminum oxide on the performance of a modified solvay process for CO₂ capture: A comparative study," *Int. J. Energy Res.*, vol. 45, no. 9, pp. 13952–13964, Jul. 2021, doi: <https://doi.org/10.1002/er.6737>.
- [36] G. U. Semblante, J. Z. Lee, L. Y. Lee, S. L. Ong, and H. Y. Ng, "Brine pre-treatment technologies for zero liquid discharge systems," *Desalination*, vol. 441, pp. 96–111, Sep. 2018, doi: 10.1016/J.DESAL.2018.04.006.
- [37] X. Li, D. Hasson, R. Semiat, and H. Shemer, "Intermediate concentrate demineralization techniques for enhanced brackish water reverse osmosis water recovery – A review," *Desalination*, vol. 466, pp. 24–35, Sep. 2019, doi: 10.1016/J.DESAL.2019.05.004.
- [38] T. van den Brand, L. Snip, L. Palmen, P. Weij, J. Sipma, and M. van Loosdrecht, "Sulfate reducing bacteria applied to domestic wastewater," *Water Pract. Technol.*, vol. 13, no. 3, pp. 542–554, Sep. 2018, doi: 10.2166/WPT.2018.068.

- [39] M. Ahmed *et al.*, “Feasibility of salt production from inland RO desalination plant reject brine: A case study,” *Desalination*, vol. 158, no. 1–3, pp. 109–117, Aug. 2003, doi: 10.1016/S0011-9164(03)00441-7.
- [40] G. J. Millar, J. Lin, A. Arshad, and S. J. Couperthwaite, “Evaluation of electrocoagulation for the pre-treatment of coal seam water,” *J. Water Process Eng.*, vol. 4, no. C, pp. 166–178, Dec. 2014, doi: 10.1016/J.JWPE.2014.10.002.
- [41] A. Venkatesan and P. C. Wankat, “Simulation of ion exchange water softening pretreatment for reverse osmosis desalination of brackish water,” *Desalination*, vol. 271, no. 1–3, pp. 122–131, Apr. 2011, doi: 10.1016/J.DESAL.2010.12.022.
- [42] S. Azimibavil and A. Jafarian, “Heat transfer evaluation and economic characteristics of falling film brine concentrator in zero liquid discharge processes,” *J. Clean. Prod.*, vol. 285, no. 0, Art. no. 124892, Feb. 2021, doi: 10.1016/J.JCLEPRO.2020.124892.
- [43] J. Weinberg and Y. Rotstein, “Operation of a horizontal tube brine concentrator,” *Desalination*, vol. 40, no. 1–2, pp. 159–168, Jan. 1982, doi: 10.1016/S0011-9164(00)88679-8.
- [44] D. von Eiff, P. W. Wong, Y. Gao, S. Jeong, and A. K. An, “Technical and economic analysis of an advanced multi-stage flash crystallizer for the treatment of concentrated brine,” *Desalination*, vol. 503, no. 0, Art. no. 114925, May 2021, doi: 10.1016/J.DESAL.2020.114925.
- [45] W. L. Ang, A. Wahab Mohammad, D. Johnson, and N. Hilal, “Forward osmosis research trends in desalination and wastewater treatment: A review of research trends over the past decade,” *J. Water Process Eng.*, vol. 31, no. 0, Art. no. 100886, Oct. 2019, doi: 10.1016/J.JWPE.2019.100886.
- [46] E. Drioli, A. Ali, and F. Macedonio, “Membrane distillation: Recent developments and perspectives,” *Desalination*, vol. 356, pp. 56–84, Jan. 2015, doi: 10.1016/J.DESAL.2014.10.028.
- [47] R. K. McGovern, A. M. Weiner, L. Sun, C. G. Chambers, S. M. Zubair, and J. H. Lienhard V, “On the cost of electrodialysis for the desalination of high salinity feeds,” *Appl. Energy*, vol. 136, pp. 649–661, Dec. 2014, doi: 10.1016/J.APENERGY.2014.09.050.
- [48] M. J. Fernández-Torres, D. G. Randall, R. Melamu, and H. von Blottnitz, “A comparative life cycle assessment of eutectic freeze crystallisation and evaporative crystallisation for the treatment of saline wastewater,” *Desalination*, vol. 306, pp. 17–23, Nov. 2012, doi: 10.1016/J.DESAL.2012.08.022.
- [49] Y. N. Wang, K. Goh, X. Li, L. Setiawan, and R. Wang, “Membranes and processes for forward osmosis-based desalination: Recent advances and future prospects,” *Desalination*, vol. 434, pp. 81–99, May 2018, doi: 10.1016/J.DESAL.2017.10.028.

- [50] H. Luo *et al.*, “A review on the recovery methods of draw solutes in forward osmosis,” *J. Water Process Eng.*, vol. 4, no. C, pp. 212–223, Dec. 2014, doi: 10.1016/J.JWPE.2014.10.006.
- [51] A. Panagopoulos, K. J. Haralambous, and M. Loizidou, “Desalination brine disposal methods and treatment technologies - A review,” *Sci. Total Environ.*, vol. 693, no. 0, Art. no. 133545, Nov. 2019, doi: 10.1016/J.SCITOTENV.2019.07.351.
- [52] Z. Wang, A. Deshmukh, Y. Du, and M. Elimelech, “Minimal and zero liquid discharge with reverse osmosis using low-salt-rejection membranes,” *Water Res.*, vol. 170, no. 0, Art. no. 115317, Mar. 2020, doi: 10.1016/J.WATRES.2019.115317.
- [53] A. Panagopoulos, “Energetic, economic and environmental assessment of zero liquid discharge (ZLD) brackish water and seawater desalination systems,” *Energy Convers. Manag.*, vol. 235, no. 0, Art. no. 113957, May 2021, doi: 10.1016/J.ENCONMAN.2021.113957.
- [54] Z. Rahimi-Ahar, M. S. Hatamipour, and L. R. Ahar, “Air humidification-dehumidification process for desalination: A review,” *Prog. Energy Combust. Sci.*, vol. 80, no. 0, Art. no. 100850, Sep. 2020, doi: 10.1016/J.PECS.2020.100850.
- [55] M. Faegh, P. Behnam, and M. B. Shafii, “A review on recent advances in humidification-dehumidification (HDH) desalination systems integrated with refrigeration, power and desalination technologies,” *Energy Convers. Manag.*, vol. 196, pp. 1002–1036, Sep. 2019, doi: 10.1016/J.ENCONMAN.2019.06.063.
- [56] A. Subramani and J. G. Jacangelo, “Treatment technologies for reverse osmosis concentrate volume minimization: A review,” *Sep. Purif. Technol.*, vol. 122, pp. 472–489, Feb. 2014, doi: 10.1016/J.SEPPUR.2013.12.004.
- [57] J. Lee and J. W. K. III, “Clathrate Hydrates,” in *Solidification*, IntechOpen, 2018, pp. 129–152. doi: 10.5772/INTECHOPEN.72956.
- [58] J. Zheng, F. Cheng, Y. Li, X. Lü, and M. Yang, “Progress and trends in hydrate based desalination (HBD) technology: A review,” *Chinese J. Chem. Eng.*, vol. 27, no. 9, pp. 2037–2043, Sep. 2019, doi: 10.1016/J.CJCHE.2019.02.017.
- [59] S. Ho-Van, B. Bouillot, J. Douzet, S. M. Babakhani, and J. M. Herri, “Cyclopentane hydrates – A candidate for desalination?,” *J. Environ. Chem. Eng.*, vol. 7, no. 5, Art. no. 103359, Oct. 2019, doi: 10.1016/J.JECE.2019.103359.
- [60] W. Pulles, G. J. G. Juby, and R. W. Busby, “Development of the Slurry Precipitation and Recycle Reverse Osmosis (SPARRO) Technology for Desalinating Scaling Mine Waters,” *Water Sci. Technol.*, vol. 25, no. 10, pp. 177–192, May 1992, doi: 10.2166/WST.1992.0246.
- [61] D. A. Glasscock, J. E. Critchfield, and G. T. Rochelle, “CO₂ absorption/desorption in mixtures of methyldiethanolamine with monoethanolamine or diethanolamine,” *Chem. Eng. Sci.*, vol. 46, no. 11, pp. 2829–2845, 1991, doi: 10.1016/0009-2509(91)85152-N.

- [62] Z. (Henry) Liang *et al.*, “Recent progress and new developments in post-combustion carbon-capture technology with amine based solvents,” *Int. J. Greenh. Gas Control*, vol. 40, pp. 26–54, 2015, doi: 10.1016/j.ijggc.2015.06.017.
- [63] F. Chu, C. Jon, L. Yang, X. Du, and Y. Yang, “CO₂ absorption characteristics in ammonia solution inside the structured packed column,” *Ind. Eng. Chem. Res.*, vol. 55, no. 12, pp. 3696–3709, Mar. 2016, doi: 10.1021/acs.iecr.5b03614.
- [64] L. E. Øi, “Aspen HYSYS simulation of CO₂ removal by amine absorption from a gas based power plant,” in *The 48th Scandinavian Conference on Simulation and Modeling (SIMS 2007); 30-31 October; 2007; Göteborg (Särö)*, 2007, no. 027, pp. 73–81.
- [65] I. P. Koronaki, L. Prentza, and V. Papaefthimiou, “Modeling of CO₂ capture via chemical absorption processes – An extensive literature review,” *Renew. Sustain. Energy Rev.*, vol. 50, pp. 547–566, 2015, doi: 10.1016/j.rser.2015.04.124.
- [66] J. Mustafa, A. A.-H. I. Mourad, A. H. Al-Marzouqi, and M. H. El-Naas, “Simultaneous treatment of reject brine and capture of carbon dioxide: A comprehensive review,” *Desalination*, vol. 483, no. 0, Art. no. 114386, 2020, doi: 10.1016/j.desal.2020.114386.
- [67] V. Romanov, Y. Soong, C. Carney, G. E. Rush, B. Nielsen, and W. O’Connor, “Mineralization of carbon dioxide: A literature review,” *ChemBioEng Rev.*, vol. 2, no. 4, pp. 231–256, Aug. 2015, doi: 10.1002/cben.201500002.
- [68] C. Wang, M. Perry, G. T. Rochelle, and A. F. Seibert, “Packing characterization: Mass transfer properties,” *Energy Procedia*, vol. 23, pp. 23–32, 2012, doi: 10.1016/j.egypro.2012.06.037.
- [69] F. Qin, S. Wang, A. Hartono, H. F. Svendsen, and C. Chen, “Kinetics of CO₂ absorption in aqueous ammonia solution,” *Int. J. Greenh. Gas Control*, vol. 4, no. 5, pp. 729–738, 2010, doi: 10.1016/j.ijggc.2010.04.010.
- [70] R. B. Bird, W. E. Stewart, and E. N. Lightfoot, *Transport Phenomena, Revised 2nd Edition*. Wiley & Sons, 2007.
- [71] K. Hornbostel *et al.*, “Packed and fluidized bed absorber modeling for carbon capture with micro-encapsulated sodium carbonate solution,” *Appl. Energy*, vol. 235, pp. 1192–1204, 2019, doi: 10.1016/j.apenergy.2018.11.027.
- [72] R. Chang, S. Kim, S. Lee, S. Choi, M. Kim, and Y. Park, “Calcium Carbonate Precipitation for CO₂ Storage and Utilization: A Review of the Carbonate Crystallization and Polymorphism,” *Front. Energy Res.*, vol. 5, p. 17, 2017, doi: 10.3389/fenrg.2017.00017.
- [73] Y. E. Kim, J. A. Lim, Soon Kwan Jeong, Y. Il Yoon, Shin Tae Bae, and S. C. Nam, “Comparison of carbon dioxide absorption in aqueous MEA, DEA, TEA, and AMP solutions,” *Bull. Korean Chem. Soc.*, vol. 34, no. 3, pp. 783–787, Mar. 2013, doi: 10.5012/BKCS.2013.34.3.783.

- [74] E. S. Sanz-Pérez, C. R. Murdock, S. A. Didas, and C. W. Jones, “Direct Capture of CO₂ from Ambient Air,” *Chem. Rev.*, vol. 116, no. 19, pp. 11840–11876, Oct. 2016, doi: 10.1021/acs.chemrev.6b00173.
- [75] F. Qin, S. Wang, I. Kim, H. F. Svendsen, and C. Chen, “Heat of absorption of CO₂ in aqueous ammonia and ammonium carbonate/carbamate solutions,” *Int. J. Greenh. Gas Control*, vol. 5, no. 3, pp. 405–412, 2011, doi: 10.1016/j.ijggc.2010.04.005.
- [76] J. L. Oscarson, R. H. Van Dam, J. J. Christensen, and R. M. Izatt, “Enthalpies of absorption of carbon dioxide in aqueous diethanolamine solutions,” *Thermochim. Acta*, vol. 146, pp. 107–114, 1989, doi: 10.1016/0040-6031(89)87081-9.
- [77] A. B. Rao, E. S. Rubin, D. W. Keith, and M. Granger Morgan, “Evaluation of potential cost reductions from improved amine-based CO₂ capture systems,” *Energy Policy*, vol. 34, no. 18, pp. 3765–3772, 2006, doi: 10.1016/j.enpol.2005.08.004.
- [78] H. P. Huang, Y. Shi, W. Li, and S. G. Chang, “Dual Alkali Approaches for the Capture and Separation of CO₂,” *Energy & Fuels*, vol. 15, no. 2, pp. 263–268, Mar. 2001, doi: 10.1021/ef0002400.
- [79] H. E. Benson and R. W. Parrish, “HiPure process removes CO₂/H₂S,” *Hydrocarbon Processing*, vol. 53, no. 4, pp. 81–82, 1974.
- [80] T. N. G. Borhani, A. Azarpour, V. Akbari, S. R. Wan Alwi, and Z. A. Manan, “CO₂ capture with potassium carbonate solutions: A state-of-the-art review,” *Int. J. Greenh. Gas Control*, vol. 41, pp. 142–162, 2015, doi: <https://doi.org/10.1016/j.ijggc.2015.06.026>.
- [81] M. H. El-Naas, A. F. Mohammad, M. I. Suleiman, M. Al Musharfy, and A. H. Al-Marzouqi, “A new process for the capture of CO₂ and reduction of water salinity,” *Desalination*, vol. 411, pp. 69–75, Jun. 2017, doi: 10.1016/J.DESAL.2017.02.005.
- [82] A. A.-H. I. Mourad, A. F. Mohammad, A. H. Al-Marzouqi, M. H. El-Naas, M. H. Al-Marzouqi, and M. Altarawneh, “KOH-based modified solvay process for removing Na ions from high salinity reject brine at high temperatures,” *Sustainability*, vol. 13, no. 8, Art. no. 10200. 2021. doi: 10.3390/su131810200.
- [83] Q. Shu, L. Legrand, P. Kuntke, M. Tedesco, and H. V. M. Hamelers, “Electrochemical regeneration of spent alkaline absorbent from direct air capture,” *Environ. Sci. Technol.*, vol. 54, no. 14, pp. 8990–8998, Jul. 2020, doi: 10.1021/acs.est.0c01977.
- [84] J. T. Yeh, H. W. Pennline, and K. P. Resnik, “Study of CO₂ absorption and desorption in a packed column,” *Energy & Fuels*, vol. 15, no. 2, pp. 274–278, Mar. 2001, doi: 10.1021/ef0002389.
- [85] N. Yang, H. Yu, L. Li, D. Xu, W. Han, and P. Feron, “Aqueous ammonia (NH₃) based post combustion CO₂ capture: A review,” *Oil Gas Sci. Technol. d’IFP Energies Nouv.*, vol. 69, no. 5, pp. 931–945, 2014, doi: 10.2516/ogst/2013160.

- [86] B. Aghel, S. Sahraie, and E. Heidaryan, "Carbon dioxide desorption from aqueous solutions of monoethanolamine and diethanolamine in a microchannel reactor," *Sep. Purif. Technol.*, vol. 237, no. 0, Art. no. 116390, 2020, doi: 10.1016/j.seppur.2019.116390.
- [87] M. Stewart and K. Arnold, "Acid gas considerations," in *Gas sweetening and processing field manual*, Boston: Gulf Professional Publishing, 2011, pp. 14–17. doi: 10.1016/B978-1-85617-982-9.00002-8.
- [88] A. Kazemi, M. Malayeri, A. Gharibi kharaji, and A. Shariati, "Feasibility study, simulation and economical evaluation of natural gas sweetening processes – Part 1: A case study on a low capacity plant in Iran," *J. Nat. Gas Sci. Eng.*, vol. 20, pp. 16–22, 2014, doi: 10.1016/j.jngse.2014.06.001.
- [89] N. K. Sarker, "Theoretical effect of concentration, circulation rate, stages, pressure and temperature of single amine and amine mixture solvents on gas sweetening performance," *Egypt. J. Pet.*, vol. 25, no. 3, pp. 343–354, 2016, doi: 10.1016/j.ejpe.2015.08.004.
- [90] D. Aaron and C. Tsouris, "Separation of CO₂ from Flue Gas: A Review," *Sep. Sci. Technol.*, vol. 40, no. 1–3, pp. 321–348, Feb. 2005, doi: 10.1081/SS-200042244.
- [91] E. L. Haseltine and J. B. Rawlings, "Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics," *J. Chem. Phys.*, vol. 117, no. 15, pp. 6959–6969, Sep. 2002, doi: 10.1063/1.1505860.
- [92] E. Hairer and G. Wanner, "Examples of stiff equations," in *Solving ordinary differential equations II: Stiff and differential-algebraic problems*, Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 2–14. doi: 10.1007/978-3-642-05221-7_1.
- [93] B. R. W. Pinsent, L. Pearson, and F. J. W. Roughton, "The kinetics of combination of carbon dioxide with hydroxide ions," *Trans. Faraday Soc.*, vol. 52, no. 0, pp. 1512–1520, 1956, doi: 10.1039/TF9565201512.
- [94] B. R. W. Pinsent, L. Pearson, and F. J. W. Roughton, "The kinetics of combination of carbon dioxide with ammonia," *Trans. Faraday Soc.*, vol. 52, no. 0, pp. 1594–1598, 1956, doi: 10.1039/TF9565201594.
- [95] R. Pohorecki and W. Moniuk, "Kinetics of reaction between carbon dioxide and hydroxyl ions in aqueous electrolyte solutions," *Chem. Eng. Sci.*, vol. 43, no. 7, pp. 1677–1684, Jan. 1988, doi: 10.1016/0009-2509(88)85159-5.
- [96] X. Wang, W. Conway, R. Burns, N. McCann, and M. Maeder, "Comprehensive study of the hydration and dehydration reactions of carbon dioxide in aqueous solution," *J. Phys. Chem. A*, vol. 114, no. 4, pp. 1734–1740, Feb. 2010, doi: 10.1021/jp909019u.
- [97] M. Caplow, "Kinetics of carbamate formation and breakdown," *J. Am. Chem. Soc.*, vol. 90, no. 24, pp. 6795–6803, Nov. 1968, doi: 10.1021/ja01026a041.

- [98] P. W. J. Derks and G. F. Versteeg, “Kinetics of absorption of carbon dioxide in aqueous ammonia solutions,” *Energy Procedia*, vol. 1, no. 1, pp. 1139–1146, 2009, doi: 10.1016/j.egypro.2009.01.150.
- [99] S. Lillia, D. Bonalumi, P. L. Fosbøl, K. Thomsen, and G. Valenti, “Experimental study of the aqueous CO₂-NH₃ rate of reaction for temperatures from 15 °C to 35 °C, NH₃ concentrations from 5% to 15% and CO₂ loadings from 0.2 to 0.6,” *Int. J. Greenh. Gas Control*, vol. 70, pp. 117–127, 2018, doi: 10.1016/j.ijggc.2018.01.009.
- [100] D. Bonalumi, S. Lillia, G. Valenti, P. L. Fosbøl, and K. Thomsen, “Kinetic study of a layout for the carbon capture with aqueous ammonia without salt precipitation,” *Energy Procedia*, vol. 114, pp. 1352–1359, Jul. 2017, doi: 10.1016/J.EGYPRO.2017.03.1256.
- [101] A. M. M. Leal, D. A. Kulik, W. R. Smith, and M. O. Saar, “An overview of computational methods for chemical equilibrium and kinetic calculations for geochemical and reactive transport modeling,” *Pure Appl. Chem.*, vol. 89, no. 5, pp. 597–643, 2017, doi: doi:10.1515/pac-2016-1107.
- [102] G. T. Yeh, G. A. Iskra, J. E. Szecsody, J. M. Zachara, and G. P. Streile, “KEMOD: A mixed chemical kinetic and equilibrium model of aqueous and solid phase geochemical reactions,” Pacific Northwest Lab., 1995.
- [103] W. Wagner and A. Pruß, “The IAPWS formulation 1995 for the thermodynamic properties of ordinary water substance for general and scientific use,” *J. Phys. Chem. Ref. Data*, vol. 31, no. 2, p. 387, Jun. 2002, doi: 10.1063/1.1461829.
- [104] W. Wagner *et al.*, “The IAPWS industrial formulation 1997 for the thermodynamic properties of water and steam,” *J. Eng. Gas Turbines Power*, vol. 122, no. 1, pp. 150–184, Jan. 2000, doi: 10.1115/1.483186.
- [105] M. W. Chase, “NIST-JANAF thermochemical tables,” *J. Phys. Chem. Ref. Data Monogr.*, vol. 9, pp. 1–1, 1998.
- [106] H. C. Helgeson, D. H. Kirkham, and G. C. Flowers, “Theoretical prediction of the thermodynamic behavior of aqueous electrolytes by high pressures and temperatures; IV, Calculation of activity coefficients, osmotic coefficients, and apparent molal and standard and relative partial molal properties to 600 d,” *Am. J. Sci.*, vol. 281, no. 10, pp. 1249–1516, Dec. 1981, doi: 10.2475/ajs.281.10.1249.
- [107] J. C. Tanger and H. C. Helgeson, “Calculation of the thermodynamic and transport properties of aqueous species at high pressures and temperatures; revised equations of state for the standard partial molal properties of ions and electrolytes,” *Am. J. Sci.*, vol. 288, no. 1, pp. 19 LP – 98, Jan. 1988, doi: 10.2475/ajs.288.1.19.

- [108] J. W. Johnson, E. H. Oelkers, and H. C. Helgeson, "SUPCRT92: A software package for calculating the standard molal thermodynamic properties of minerals, gases, aqueous species, and reactions from 1 to 5000 bar and 0 to 1000°C," *Comput. Geosci.*, vol. 18, no. 7, pp. 899–947, 1992, doi: 10.1016/0098-3004(92)90029-Q.
- [109] D. P. Fernández, A. R. H. Goodwin, E. W. Lemmon, J. M. H. Levelt Sengers, and R. C. Williams, "A formulation for the static permittivity of water and steam at temperatures from 238 K to 873 K at pressures up to 1200 MPa, including derivatives and Debye–Hückel coefficients," *J. Phys. Chem. Ref. Data*, vol. 26, no. 4, pp. 1125–1166, Jul. 1997, doi: 10.1063/1.555997.
- [110] H. C. Helgeson, C. E. Owens, A. M. Knox, and L. Richard, "Calculation of the standard molal thermodynamic properties of crystalline, liquid, and gas organic molecules at high temperatures and pressures," *Geochim. Cosmochim. Acta*, vol. 62, no. 6, pp. 985–1081, Mar. 1998, doi: 10.1016/S0016-7037(97)00219-6.
- [111] A. T. M. G. Mostafa, J. M. Eakman, and S. L. Yarbrow, "Prediction of standard heats and Gibbs free energies of formation of solid inorganic salts from group contributions," *Ind. Eng. Chem. Res.*, vol. 34, no. 12, pp. 4577–4582, Dec. 1995, doi: 10.1021/ie00039a053.
- [112] A. T. M. G. Mostafa, J. M. Eakman, M. M. Montoya, and S. L. Yarbrow, "Prediction of Heat Capacities of Solid Inorganic Salts from Group Contributions," *Ind. Eng. Chem. Res.*, vol. 35, no. 1, pp. 343–348, Jan. 1996, doi: 10.1021/ie9501485.
- [113] D.-Y. Peng and D. B. Robinson, "A new two-constant equation of state," *Ind. Eng. Chem. Fundam.*, vol. 15, no. 1, pp. 59–64, Feb. 1976, doi: 10.1021/i160057a011.
- [114] G. Soave, "Equilibrium constants from a modified Redlich-Kwong equation of state," *Chem. Eng. Sci.*, vol. 27, no. 6, pp. 1197–1203, 1972, doi: 10.1016/0009-2509(72)80096-4.
- [115] P. Ghosh, "Prediction of vapor-liquid equilibria using Peng-Robinson and Soave-Redlich-Kwong equations of state," *Chem. Eng. Technol.*, vol. 22, no. 5, pp. 379–399, May 1999, doi: 10.1002/(SICI)1521-4125(199905)22:5<379::AID-CEAT379>3.0.CO;2-Q.
- [116] P. Debye and E. Hückel, "The theory of electrolytes I. The lowering of the freezing point and related occurrences," *Phys. Zeitschrift*, vol. 24, pp. 185–206, 1923.
- [117] L. A. Bromley, "Thermodynamic properties of strong electrolytes in aqueous solutions," *AIChE J.*, vol. 19, no. 2, pp. 313–320, Mar. 1973, doi: 10.1002/aic.690190216.
- [118] J. N. Brønsted, "Studies on solubility. IV. The principle of the specific interaction of ions," *J. Am. Chem. Soc.*, vol. 44, no. 5, pp. 877–898, May 1922, doi: 10.1021/ja01426a001.

- [119] E. A. Guggenheim and J. C. Turgeon, "Specific interaction of ions," *Trans. Faraday Soc.*, vol. 51, no. 0, pp. 747–761, 1955, doi: 10.1039/TF9555100747.
- [120] G. Scatchard, "Concentrated solutions of strong electrolytes," *Chem. Rev.*, vol. 19, no. 3, pp. 309–327, Dec. 1936, doi: 10.1021/cr60064a008.
- [121] K. S. Pitzer, "Thermodynamics of electrolytes. I. Theoretical basis and general equations," *J. Phys. Chem.*, vol. 77, no. 2, pp. 268–277, Jan. 1973, doi: 10.1021/j100621a026.
- [122] K. S. Pitzer, *Activity Coefficients in Electrolyte Solutions*. CRC Press, 2018. doi: 10.1201/9781351069472.
- [123] A. Fredenslund, R. L. Jones, and J. M. Prausnitz, "Group-contribution estimation of activity coefficients in nonideal liquid mixtures," *AIChE J.*, vol. 21, no. 6, pp. 1086–1099, Nov. 1975, doi: 10.1002/aic.690210607.
- [124] J. Li, H.-M. Polka, and J. Gmehling, "A gE model for single and mixed solvent electrolyte systems: 1. Model and results for strong electrolytes," *Fluid Phase Equilib.*, vol. 94, pp. 89–114, 1994, doi: 10.1016/0378-3812(94)87052-7.
- [125] K. Thomsen, "Aqueous electrolytes model parameters and process simulation." Technical University of Denmark, Kgs. Lyngby, 1997. doi: 10.11581/dtu:00000074.
- [126] P. Wang, A. Anderko, and R. D. Young, "A speciation-based model for mixed-solvent electrolyte systems," *Fluid Phase Equilib.*, vol. 203, no. 1, pp. 141–176, 2002, doi: 10.1016/S0378-3812(02)00178-4.
- [127] J. W. Bullard and G. W. Scherer, "An ideal solid solution model for C-S-H," *J. Am. Ceram. Soc.*, vol. 99, no. 12, pp. 4137–4145, Dec. 2016, doi: 10.1111/jace.14493.
- [128] B. Lothenbach, "Thermodynamic equilibrium calculations in cementitious systems," *Mater. Struct.*, vol. 43, no. 10, pp. 1413–1433, 2010, doi: 10.1617/s11527-010-9592-x.
- [129] A. F. Mohammad *et al.*, "Computational fluid dynamics simulation of an Inert Particles Spouted Bed Reactor (IPSBR) system," *Int. J. Chem. React. Eng.*, vol. 18, pp. 5–6, 2020, doi: doi:10.1515/ijcre-2020-0025.
- [130] A. Sharifi Haddad, H. Hassanzadeh, J. Abedi, and Z. Chen, "Lumped mass transfer coefficient for divergent radial solute transport in fractured aquifers," *J. Hydrol.*, vol. 495, pp. 113–120, 2013, doi: 10.1016/j.jhydrol.2013.05.007.
- [131] Y. T. Shah, B. G. Kelkar, S. P. Godbole, and W.-D. Deckwer, "Design parameters estimations for bubble column reactors," *AIChE J.*, vol. 28, no. 3, pp. 353–379, May 1982, doi: <https://doi.org/10.1002/aic.690280302>.
- [132] H. Hikita, S. Asai, K. Tanigawa, K. Segawa, and M. Kitao, "The volumetric liquid-phase mass transfer coefficient in bubble columns," *Chem. Eng. J.*, vol. 22, no. 1, pp. 61–69, 1981, doi: 10.1016/0300-9467(81)85006-X.

- [133] A. Kasturi *et al.*, “CO₂ absorption from simulated flue gas in a bubble column,” vol. 54, pp. 2034–2046, 2019, doi: 10.1080/01496395.2019.1617745.
- [134] I. Dewes and A. Schumpe, “Gas density effect on mass transfer in the slurry bubble column,” *Chem. Eng. Sci.*, vol. 52, no. 21, pp. 4105–4109, 1997, doi: 10.1016/S0009-2509(97)00252-2.
- [135] M. H. Sharqawy, J. H. Lienhard, and S. M. Zubair, “Thermophysical properties of seawater: A review of existing correlations and data,” *Desalin. Water Treat.*, vol. 16, no. 1–3, pp. 354–380, Apr. 2010, doi: 10.5004/dwt.2010.1079.
- [136] W. Jeong and J. Seong, “Comparison of effects on technical variances of computational fluid dynamics (CFD) software based on finite element and finite volume methods,” *Int. J. Mech. Sci.*, vol. 78, pp. 19–26, 2014, doi: 10.1016/j.ijmecsci.2013.10.017.
- [137] J. Ma, G. L. Chahine, and C.-T. Hsiao, “Spherical bubble dynamics in a bubbly medium using an Euler–Lagrange model,” *Chem. Eng. Sci.*, vol. 128, pp. 64–81, 2015, doi: 10.1016/j.ces.2015.01.056.
- [138] P. Shi and R. Rzehak, “Solid-liquid flow in stirred tanks: Euler-Euler/RANS modeling,” *Chem. Eng. Sci.*, vol. 227, p. 115875, 2020, doi: 10.1016/j.ces.2020.115875.
- [139] B. Ničeno, M. Boucker, and B. L. Smith, “Euler-Euler large eddy simulation of a square cross-sectional bubble column using the Neptune_CFD code,” *Sci. Technol. Nucl. Install.*, vol. 2009, no. 0, Art. no. 410272, 2009, doi: 10.1155/2009/410272.
- [140] D. Darmana, R. L. B. Henket, N. G. Deen, and J. A. M. Kuipers, “Detailed modelling of hydrodynamics, mass transfer and chemical reactions in a bubble column using a discrete bubble model: Chemisorption of CO₂ into NaOH solution, numerical and experimental study,” *Chem. Eng. Sci.*, vol. 62, no. 9, pp. 2556–2575, 2007, doi: 10.1016/j.ces.2007.01.065.
- [141] H. Xiao and P. Cinnella, “Quantification of model uncertainty in RANS simulations: A review,” *Prog. Aerosp. Sci.*, vol. 108, pp. 1–31, 2019, doi: 10.1016/j.paerosci.2018.10.001.
- [142] V. Bedekar, E. D. Morway, C. D. Langevin, and M. J. Tonkin, “MT3D-USGS version 1: A U.S. Geological Survey release of MT3DMS updated with new and expanded transport capabilities for use with MODFLOW,” Reston, VA, 2016, pp. 1–84. doi: 10.3133/tm6A53.
- [143] D. M. Himmelblau, “Diffusion of dissolved gases in liquids,” *Chem. Rev.*, vol. 64, no. 5, pp. 527–550, Oct. 1964, doi: 10.1021/cr60231a002.
- [144] S. C. Cardona, F. López, A. Abad, and J. Navarro-Laboulais, “On bubble column reactor design for the determination of kinetic rate constants in gas–liquid systems,” *Can. J. Chem. Eng.*, vol. 88, no. 4, pp. 491–502, Aug. 2010, doi: 10.1002/cjce.20327.

- [145] F. Chu, L. Yang, X. Du, and Y. Yang, “Mass transfer and energy consumption for CO₂ absorption by ammonia solution in bubble column,” *Appl. Energy*, vol. 190, pp. 1068–1080, 2017, doi: 10.1016/j.apenergy.2017.01.027.
- [146] M. Babanezhad, A. Marjani, and S. Shirazian, “Multidimensional machine learning algorithms to learn liquid velocity inside a cylindrical bubble column reactor,” *Sci. Rep.*, vol. 10, no. 1, Art. no. 21502, 2020, doi: 10.1038/s41598-020-78388-x.
- [147] A. Sokolichin and G. Eigenberger, “Applicability of the standard k– ϵ turbulence model to the dynamic simulation of bubble columns: Part I. Detailed numerical simulations,” *Chem. Eng. Sci.*, vol. 54, no. 13, pp. 2273–2284, 1999, doi: 10.1016/S0009-2509(98)00420-5.
- [148] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006, doi: 10.1007/s10107-004-0559-y.
- [149] L. Beal, D. Hill, R. Martin, and J. Hedengren, “GEKKO optimization suite,” *Processes*, vol. 6, no. 8, Art. no. 106, Jul. 2018, doi: 10.3390/pr6080106.
- [150] “Python Software Foundation,” 2020. <https://www.python.org/>
- [151] A. Chilakapati, “RAFT: A simulator for ReActive Flow and Transport of groundwater contaminants,” United States, 1995. doi: 10.2172/527466.
- [152] A. J. Valocchi and M. Malmstead, “Accuracy of operator splitting for advection-dispersion-reaction problems,” *Water Resour. Res.*, vol. 28, no. 5, pp. 1471–1476, May 1992, doi: 10.1029/92WR00423.
- [153] G. Strang, “On the construction and comparison of difference schemes,” *SIAM J. Numer. Anal.*, vol. 5, no. 3, pp. 506–517, Sep. 1968, doi: 10.1137/0705041.
- [154] I. V. Dolgaleva, I. G. Gorichev, A. D. Izotov, and V. M. Stepanov, “Modeling of the effect of pH on the calcite dissolution kinetics,” *Theor. Found. Chem. Eng.*, vol. 39, no. 6, pp. 614–621, 2005, doi: 10.1007/s11236-005-0125-1.
- [155] S. L. Clegg, J. A. Rard, and K. S. Pitzer, “Thermodynamic properties of 0–6 mol kg⁻¹ aqueous sulfuric acid from 273.15 to 328.15 K,” *J. Chem. Soc. Faraday Trans.*, vol. 90, no. 13, pp. 1875–1894, Jan. 1994, doi: 10.1039/FT9949001875.
- [156] J. M. I. and M. T. M., “Machine learning: Trends, perspectives, and prospects,” *Science (80-.)*, vol. 349, no. 6245, pp. 255–260, Jul. 2015, doi: 10.1126/science.aaa8415.
- [157] P. Domingos, *The master algorithm: How the quest for the ultimate learning machine will remake our world*. Basic Books, 2015.
- [158] F. Girosi, M. Jones, and T. Poggio, “Regularization Theory and Neural Networks Architectures,” *Neural Comput.*, vol. 7, no. 2, pp. 219–269, 1995, doi: 10.1162/neco.1995.7.2.219.

- [159] M. Claesen and B. De Moor, “Hyperparameter search in machine learning,” *arXiv Prepr. arXiv1502.02127*, 2015.
- [160] M. Belkin, D. Hsu, S. Ma, and S. Mandal, “Reconciling modern machine-learning practice and the classical bias–variance trade-off,” *Proc. Natl. Acad. Sci.*, vol. 116, no. 32, pp. 15849 LP – 15854, Aug. 2019, doi: 10.1073/pnas.1903070116.
- [161] D. Maulud and A. M. Abdulazeez, “A review on linear regression comprehensive in machine learning,” *J. Appl. Sci. Technol. Trends*, vol. 1, no. 4, pp. 140–147, 2020.
- [162] J. Zou, Y. Han, and S.-S. So, “Overview of Artificial Neural Networks BT - Artificial Neural Networks: Methods and Applications,” D. J. Livingstone, Ed. Totowa, NJ: Humana Press, 2009, pp. 14–22. doi: 10.1007/978-1-60327-101-1_2.
- [163] F. Murtagh, “Multilayer perceptrons for classification and regression,” *Neurocomputing*, vol. 2, no. 5, pp. 183–197, 1991, doi: [https://doi.org/10.1016/0925-2312\(91\)90023-5](https://doi.org/10.1016/0925-2312(91)90023-5).
- [164] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989, doi: [https://doi.org/10.1016/0893-6080\(89\)90020-8](https://doi.org/10.1016/0893-6080(89)90020-8).
- [165] E. Phaisangittisagul, “An analysis of the regularization between L2 and dropout in single hidden layer neural network,” in *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, 2016, pp. 174–179. doi: 10.1109/ISMS.2016.14.
- [166] F. Bre, J. Gimenez, and V. Fachinotti, “Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks,” *Energy Build.*, vol. 158, Nov. 2017, doi: 10.1016/j.enbuild.2017.11.045.
- [167] L. Yang, S. Liu, S. Tsoka, and L. G. Papageorgiou, “A regression tree approach using mathematical programming,” *Expert Syst. Appl.*, vol. 78, pp. 347–357, 2017, doi: <https://doi.org/10.1016/j.eswa.2017.02.013>.
- [168] W.-Y. Loh, “Classification and regression trees,” *WIREs Data Min. Knowl. Discov.*, vol. 1, no. 1, pp. 14–23, Jan. 2011, doi: <https://doi.org/10.1002/widm.8>.
- [169] A. Navada, A. N. Ansari, S. Patil, and B. A. Sonkamble, “Overview of use of decision tree algorithms in machine learning,” in *2011 IEEE Control and System Graduate Research Colloquium*, 2011, pp. 37–42. doi: 10.1109/ICSGRC.2011.5991826.
- [170] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
- [171] A. Cutler, D. R. Cutler, and J. R. Stevens, “Random Forests,” in *Ensemble Machine Learning: Methods and applications*, C. Zhang and Y. Ma, Eds. Boston, MA: Springer US, 2012, pp. 157–175. doi: 10.1007/978-1-4419-9326-7_5.
- [172] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

- [173] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995, doi: 10.1007/BF00994018.
- [174] M. E. Wall, A. Rechtsteiner, and L. M. Rocha, "Singular Value Decomposition and Principal Component Analysis," in *A practical approach to microarray data analysis*, D. P. Berrar, W. Dubitzky, and M. Granzow, Eds. Boston, MA: Springer US, 2003, pp. 91–109. doi: 10.1007/0-306-47815-3_5.
- [175] F. Song, Z. Guo, and D. Mei, "Feature selection using Principal Component Analysis," in *2010 International Conference on System Science, Engineering Design and Manufacturing Informatization*, 2010, vol. 1, pp. 27–30. doi: 10.1109/ICSEM.2010.14.
- [176] T. Elgamal, M. Yabandeh, A. Abounaga, and M. Hefeeda, "sPCA: Scalable Principal Component Analysis for big data on distributed platforms," Mar. 2015.
- [177] M. Awad and R. Khanna, "Support Vector Regression," in *Efficient learning machines: Theories, concepts, and applications for engineers and system designers*, M. Awad and R. Khanna, Eds. Berkeley, CA: Apress, 2015, pp. 67–80. doi: 10.1007/978-1-4302-5990-9_4.
- [178] S. Lahiri and K. Ghanta, "The Support Vector Regression with the parameter tuning assisted by a differential evolution technique: Study of the critical velocity of a slurry flow in a pipeline," *Chem. Ind. Chem. Eng. Q.*, vol. 14, pp. 191–203, Jul. 2008, doi: 10.2298/CICEQ0803191L.
- [179] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu, "The entire regularization path for the Support Vector Machine," *J. Mach. Learn. Res.*, vol. 5, pp. 1391–1415, Oct. 2004.
- [180] O. Kramer, "K-Nearest Neighbors," in *Dimensionality reduction with unsupervised nearest neighbors*, O. Kramer, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 13–23. doi: 10.1007/978-3-642-38652-7_2.
- [181] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, 1967, doi: 10.1109/TIT.1967.1053964.
- [182] O. Anava and K. Levy, "k*-nearest neighbors: From global to local," *Adv. Neural Inf. Process. Syst.*, vol. 29, 2016.
- [183] S. Har-Peled, P. Indyk, and R. Motwani, "Approximate nearest neighbor: Towards removing the curse of dimensionality," 2012.
- [184] C. A. J. Appelo, D. L. Parkhurst, and V. E. A. Post, "Equations for calculating hydrogeochemical reactions of minerals and gases such as CO₂ at high pressures and temperatures," *Geochim. Cosmochim. Acta*, vol. 125, pp. 49–67, 2014, doi: 10.1016/j.gca.2013.10.003.
- [185] G. Puxty, R. Rowland, and M. Attalla, "Comparison of the rate of CO₂ absorption into aqueous ammonia and monoethanolamine," *Chem. Eng. Sci.*, vol. 65, no. 2, pp. 915–922, 2010, doi: 10.1016/j.ces.2009.09.042.

- [186] J. Liu, S. Wang, G. Qi, B. Zhao, and C. Chen, “Kinetics and mass transfer of carbon dioxide absorption into aqueous ammonia,” *Energy Procedia*, vol. 4, pp. 525–532, 2011, doi: 10.1016/j.egypro.2011.01.084.
- [187] V. Darde, W. J. M. van Well, P. L. Fosboel, E. H. Stenby, and K. Thomsen, “Experimental measurement and modeling of the rate of absorption of carbon dioxide by aqueous ammonia,” *Int. J. Greenh. Gas Control*, vol. 5, no. 5, pp. 1149–1162, 2011, doi: 10.1016/j.ijggc.2011.07.008.
- [188] H. Jilvero, F. Normann, K. Andersson, and F. Johnsson, “The rate of CO₂ absorption in ammonia —Implications on absorber design,” *Ind. Eng. Chem. Res.*, vol. 53, no. 16, pp. 6750–6758, Apr. 2014, doi: 10.1021/ie403346a.
- [189] D. Bonalumi, S. Lillia, G. Valenti, P. L. Fosbøl, and K. Thomsen, “Kinetic study of a Layout for the Carbon Capture with Aqueous Ammonia without Salt Precipitation,” *Energy Procedia*, vol. 114, pp. 1352–1359, 2017, doi: 10.1016/j.egypro.2017.03.1256.
- [190] X. Wang *et al.*, “Kinetics of the reversible reaction of CO₂(aq) with ammonia in aqueous solution,” *J. Phys. Chem. A*, vol. 115, no. 24, pp. 6405–6412, Jun. 2011, doi: 10.1021/jp108491a.
- [191] G. K. Anderson, “Enthalpy of dissociation and hydration number of carbon dioxide hydrate from the Clapeyron equation,” *J. Chem. Thermodyn.*, vol. 35, no. 7, pp. 1171–1183, 2003, doi: 10.1016/S0021-9614(03)00093-4.
- [192] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
- [193] T. Chai and R. R. Draxler, “Root mean square error (RMSE) or mean absolute error (MAE)?—Arguments against avoiding RMSE in the literature,” *Geosci. Model Dev.*, vol. 7, no. 3, pp. 1247–1250, 2014, doi: 10.5194/gmd-7-1247-2014.
- [194] H. Fakharian, H. Ganji, A. Naderifar, H. R. Mofrad, and M. Kakavand, “Effect of gas type and salinity on performance of produced water desalination using gas hydrates,” *J. Water Reuse Desalin.*, vol. 9, no. 4, pp. 396–404, Nov. 2019, doi: 10.2166/wrd.2019.013.
- [195] L. S. Tan, A. M. Shariff, K. K. Lau, and M. A. Bustam, “Factors affecting CO₂ absorption efficiency in packed column: A review,” *J. Ind. Eng. Chem.*, vol. 18, no. 6, pp. 1874–1883, 2012, doi: 10.1016/j.jiec.2012.05.013.
- [196] Y. Peng, B. Zhao, and L. Li, “Advance in post-combustion CO₂ capture with alkaline solution: A brief review,” *Energy Procedia*, vol. 14, pp. 1515–1522, 2012, doi: <https://doi.org/10.1016/j.egypro.2011.12.1126>.
- [197] G. Liu, L. Kou, and C. Li, “Absorption performance for CO₂ capture process using MDEA-AMP aqueous solution,” *IOP Conf. Ser. Earth Environ. Sci.*, vol. 59, p. 12011, 2017, doi: 10.1088/1755-1315/59/1/012011.

- [198] M. Yoo, S.-J. Han, and J.-H. Wee, “Carbon dioxide capture capacity of sodium hydroxide aqueous solution,” *J. Environ. Manage.*, vol. 114, pp. 512–519, 2013, doi: 10.1016/j.jenvman.2012.10.061.
- [199] A. Stefánsson, P. Bénézech, and J. Schott, “Carbonic acid ionization and the stability of sodium bicarbonate and carbonate ion pairs to 200°C – A potentiometric and spectrophotometric study,” *Geochim. Cosmochim. Acta*, vol. 120, pp. 600–611, 2013, doi: 10.1016/j.gca.2013.04.023.
- [200] V. Lagneau, A. Pipart, and H. Catalette, “Reactive transport modelling of CO₂ sequestration in deep saline aquifers,” *Oil Gas Sci. Technol. – Rev. IFP*, vol. 60, pp. 231–247, Mar. 2005.
- [201] Z. Duan and D. Li, “Coupled phase and aqueous species equilibrium of the H₂O–CO₂–NaCl–CaCO₃ system from 0 to 250°C, 1 to 1000bar with NaCl concentrations up to saturation of halite,” *Geochim. Cosmochim. Acta*, vol. 72, no. 20, pp. 5128–5145, 2008, doi: 10.1016/j.gca.2008.07.025.
- [202] J. W. Morse and F. T. B. T.-D. in S. Mackenzie, Eds., “Interactions between carbonate minerals and solutions,” in *Geochemistry of sedimentary carbonates*, vol. 48, Elsevier, pp. 39–86, 1990, doi: 10.1016/S0070-4571(08)70331-5.
- [203] M. Sudmalis and R. Sheikholeslami, “Precipitation and co-precipitation of CaCO₃ and CaSO₄,” *Can. J. Chem. Eng.*, vol. 78, pp. 21–31, Feb. 2000, doi: 10.1002/cjce.5450780106.
- [204] H. E. Doner and P. F. Pratt, “Solubility of calcium carbonate precipitated in montmorillonite suspensions,” *Soil Sci. Soc. Am. J.*, vol. 32, no. 5, pp. 743–744, Sep. 1968, doi: 10.2136/sssaj1968.03615995003200050046x.
- [205] R. Zevenhoven, D. Legendre, A. Said, and M. Järvinen, “Carbon dioxide dissolution and ammonia losses in bubble columns for precipitated calcium carbonate (PCC) production,” *Energy*, vol. 175, pp. 1121–1129, 2019, doi: 10.1016/j.energy.2019.03.112.
- [206] D. Li, L. Chen, G. Liu, J. Xiao, B. Li, and W. Yu, “Comparison between the replacements of CH₄ in natural gas hydrate with CO₂ and NH₃,” *Energy Reports*, vol. 7, pp. 3639–3646, 2021, doi: 10.1016/j.egy.2021.04.024.
- [207] L. Cisternas, C. Vásquez, and R. Swaney, “On the design of crystallization-based separation processes: Review and extension,” *AIChE J.*, vol. 52, pp. 1754–1769, May 2006, doi: 10.1002/aic.10768.
- [208] C. Christov and N. Møller, “Chemical equilibrium model of solution behavior and solubility in the H–Na–K–OH–Cl–HSO₄–SO₄–H₂O system to high concentration and temperature,” *Geochim. Cosmochim. Acta*, vol. 68, no. 6, pp. 1309–1331, Mar. 2004, doi: 10.1016/J.GCA.2003.08.017.
- [209] B. S. Krumgalz, R. Pogorelsky, and K. S. Pitzer, “Volumetric properties of single aqueous electrolytes from zero to saturation concentration at 298.15 K represented by pitzer’s ion-interaction equations,” *J. Phys. Chem. Ref. Data*, vol. 25, no. 2, pp. 663–689, Oct. 1996, doi: 10.1063/1.555981.

- [210] S. L. Clegg, P. Brimblecombe, and A. S. Wexler, "Thermodynamic model of the system H^+ - NH_4^+ - SO_4^{2-} - NO_3^- - H_2O at tropospheric temperatures," *J. Phys. Chem. A*, vol. 102, no. 12, pp. 2137–2154, Mar. 1998, doi: 10.1021/jp973042r.
- [211] J. P. Greenberg and N. Møller, "The prediction of mineral solubilities in natural waters: A chemical equilibrium model for the Na-K-Ca-Cl- SO_4 - H_2O system to high concentration from 0 to 250°C," *Geochim. Cosmochim. Acta*, vol. 53, no. 10, pp. 2503–2518, Oct. 1989, doi: 10.1016/0016-7037(89)90124-5.
- [212] S. He and J. W. Morse, "The carbonic acid system and calcite solubility in aqueous Na-K-Ca-Mg-Cl- SO_4 solutions from 0 to 90°C," *Geochim. Cosmochim. Acta*, vol. 57, no. 15, pp. 3533–3554, Aug. 1993, doi: 10.1016/0016-7037(93)90137-L.
- [213] R. C. Phutela, K. S. Pitzer, and P. P. S. Saluja, "Thermodynamics of aqueous magnesium chloride, calcium chloride, and strontium chloride at elevated temperatures," *J. Chem. Eng. Data*, vol. 32, no. 1, pp. 76–80, Jan. 2002, doi: 10.1021/JE00047A022.
- [214] Z. Dai, A. Kan, F. Zhang, and M. Tomson, "A thermodynamic model for the solubility prediction of barite, calcite, gypsum, and anhydrite, and the association constant estimation of $\text{CaSO}_4^{(0)}$ ion pair up to 250°C and 22000 psi," *J. Chem. Eng. Data*, vol. 60, no. 3, pp. 766–774, Mar. 2014, doi: 10.1021/JE5008873.
- [215] K. S. Pitzer and G. Mayorga, "Thermodynamics of electrolytes. II. Activity and osmotic coefficients for strong electrolytes with one or both ions univalent," *J. Phys. Chem.*, vol. 77, no. 19, pp. 2300–2308, 2002, doi: 10.1021/J100638A009.
- [216] L. F. Silvester and K. S. Pitzer, "Thermodynamics of electrolytes. X. Enthalpy and the effect of temperature on the activity coefficients," *J. Solut. Chem. 1978* 75, vol. 7, no. 5, pp. 327–337, May 1978, doi: 10.1007/BF00662893.
- [217] C. E. Harvie, N. Møller, and J. H. Weare, "The prediction of mineral solubilities in natural waters: The Na-K-Mg-Ca-H-Cl- SO_4 -OH- HCO_3^- - CO_3^- - CO_2 - H_2O system to high ionic strengths at 25°C," *Geochim. Cosmochim. Acta*, vol. 48, no. 4, pp. 723–751, Apr. 1984, doi: 10.1016/0016-7037(84)90098-X.
- [218] Á. Pérez-Salado Kamps, R. Sing, B. Rumpf, and G. Maurer, "Influence of NH_4Cl , NH_4NO_3 , and NaNO_3 on the simultaneous solubility of ammonia and carbon dioxide in water," *J. Chem. Eng. Data*, vol. 45, no. 5, pp. 796–809, Sep. 2000, doi: 10.1021/je000106+.
- [219] W. E. Thiessen and J. M. Simonson, "Enthalpy of dilution and the thermodynamics of ammonium chloride (aq) to 523 K and 35 MPa," *J. Phys. Chem.*, vol. 94, no. 20, pp. 7794–7800, 1990, doi: 10.1021/J100383A011.
- [220] L. Plummer, D. Parkhurst, G. Fleming, and S. Dunkle, "PHRQPITZ - A computer program incorporating pitzer's equations for calculation of geochemical reactions in brines," *US Geol. Surv. Water-Resour. Invest. Rep.*, vol. 884153, Jan. 1988.

- [221] K. C. Kang, P. Linga, K. Park, S.-J. Choi, and J. D. Lee, “Seawater desalination by gas hydrate process and removal characteristics of dissolved ions (Na^+ , K^+ , Mg^{2+} , Ca^{2+} , B^{3+} , Cl^- , SO_4^{2-}),” *Desalination*, vol. 353, pp. 84–90, 2014, doi: 10.1016/j.desal.2014.09.007.
- [222] Q. Lv, X. Li, and G. Li, “Seawater desalination by hydrate formation and pellet production process,” *Energy Procedia*, vol. 158, pp. 5144–5148, 2019, doi: 10.1016/j.egypro.2019.01.684.
- [223] J.-H. Cha and Y. Seol, “Increasing gas hydrate formation temperature for desalination of high salinity produced water with secondary guests,” *ACS Sustain. Chem. Eng.*, vol. 1, no. 10, pp. 1218–1224, Oct. 2013, doi: 10.1021/sc400160u.
- [224] K. Kang, S. Hong, S. Cho, D. H. Kim, and J. Lee, “Evaluation of desalination by nanostructured hydrate formation and pellet production process,” *J. Nanosci. Nanotechnol.*, vol. 17, pp. 4059–4062, Jun. 2017, doi: 10.1166/jnn.2017.13383.
- [225] M. Maniavi Falahieh, M. Bonyadi, and A. Lashanizadegan, “A new hybrid desalination method based on the CO_2 gas hydrate and capacitive deionization processes,” *Desalination*, vol. 502, p. 114932, 2021, doi: 10.1016/j.desal.2021.114932.
- [226] M. Sarshar and A. H. Sharafi, “Simultaneous water desalination and CO_2 capturing by hydrate formation,” *Desalin. Water Treat.*, vol. 28, no. 1–3, pp. 59–64, Apr. 2011, doi: 10.5004/dwt.2011.2201.
- [227] P. Babu *et al.*, “Hydrate-based desalination (HyDesal) process employing a novel prototype design,” *Chem. Eng. Sci.*, vol. 218, pp. 115563, 2020, doi: 10.1016/j.ces.2020.115563.
- [228] M. Sadi, H. Fakharian, H. Ganji, and M. Kakavand, “Evolving artificial intelligence techniques to model the hydrate-based desalination process of produced water,” *J. Water Reuse Desalin.*, vol. 9, no. 4, pp. 372–384, Oct. 2019, doi: 10.2166/wrd.2019.024.
- [229] S.-W. Han, “RO pretreatment using hydrate based ice desalination process,” Graduate School of UNIST, 2016. [Online]. Available: [https://scholarworks.unist.ac.kr/bitstream/201301/18325/1/RO PRETREATMENT USING HYDRATE BASED ICE DESALINATION PROCESS.pdf](https://scholarworks.unist.ac.kr/bitstream/201301/18325/1/RO%20PRETREATMENT%20USING%20HYDRATE%20BASED%20ICE%20DESALINATION%20PROCESS.pdf)
- [230] J. Niu, “Desalination of produced water via gas hydrate formation and post treatment,” Virginia Tech, 2012. [Online]. Available: https://vttechworks.lib.vt.edu/bitstream/handle/10919/76822/etd-07122012-172310_Niu_J_T_2012.pdf?sequence=1
- [231] H. Fakharian, H. Ganji, and A. Naderifar, “Saline produced water treatment using gas hydrates,” *J. Environ. Chem. Eng.*, vol. 5, no. 5, pp. 4269–4273, 2017, doi: 10.1016/j.jece.2017.08.008.
- [232] I. Kim *et al.*, “Highly porous CO_2 hydrate generation aided by silica nanoparticles for potential secure storage of CO_2 and desalination,” *RSC Adv.*, vol. 7, no. 16, pp. 9545–9550, 2017, doi: 10.1039/C6RA26366F.

- [233] J. Zheng and M. Yang, “Experimental investigation on novel desalination system via gas hydrate,” *Desalination*, vol. 478, no. 0, Art. no. 114284, 2020, doi: 10.1016/j.desal.2019.114284.

Appendix

A.1 Pitzer Interactions

This section includes a list of all the interactions used in the Pitzer Model. The data is separated into two tables: Table A1.1 has two-ion interactions, while Table A1.2 involves three-ion interactions. Below is a list of equations that were used by each reference to calculate an interaction (P_{ion}) at some temperature (T) and pressure (P). Each equation will require some parameters (a_1, a_2, \dots, a_{11}) which could be found the tables below.

$$P_{ion} = a_1 \quad (1)$$

$$P_{ion} = a_1 + a_2 * (T - 298.15) + a_3 * (P - 1) \quad (2)$$

$$P_{ion} = a_1 + a_2 * T + a_3 * T^2 + \frac{a_4}{T} + a_5 * \ln(T) \quad (3)$$

$$P_{ion} = a_1 + a_2 * T + a_3 * T^2 + a_4 * T^3 + \frac{a_5}{T} \quad (4)$$

$$P_{ion} = \frac{a_1}{T} + a_2 + a_3 * \ln(T) + a_4 * T + a_5 * T^2 + \frac{a_6}{T^2} \quad (5)$$

$$P_{ion} = a_1 + \frac{a_2}{T} + a_3 * T + a_4 * T^2 + \frac{a_5}{T^2} + \frac{a_6}{T^3} \quad (6)$$

$$P_{ion} = a_1 + a_2 * T + a_3 * T^2 + \frac{a_4}{T} + a_5 * \ln(T) + a_6 * (P - 1) \quad (7)$$

$$P_{ion} = a_1 + a_2 * T + a_3 * T^2 + a_4 * T^3 + \frac{a_5}{T} + a_6 * \ln(T) + \frac{a_7}{T - 263.} \quad (8)$$

$$+ \frac{a_8}{680. - T} + a_9 * (P - 1)$$

$$P_{ion} = a_1 + a_2 * T + \frac{a_3}{T} + a_4 * \ln(T) + \frac{a_5}{T - 263.} + a_6 * T^2 + \frac{a_7}{680. - T} \quad (9)$$

$$+ \frac{a_8}{T - 227.} + a_9 * (P - 1)$$

$$P_{ion} = a_1 + a_2 \left(\frac{T}{2} + \frac{298^2}{2T} - 298 \right) + a_3 \left(\frac{T^2}{6} + \frac{298^3}{3T} - \frac{298^2}{2} \right) \quad (10)$$

$$+ a_4 \left(\frac{T^3}{12} + \frac{298^4}{4T} - \frac{298^3}{3} \right) + a_5 \left(298 - \frac{298^2}{T} \right) + (P - 1)$$

$$* \left[\frac{a_6}{T} + a_7 + a_8 * T + a_9 * T^2 \right]$$

$$P_{ion} = a_1 + a_2 * \ln(T) + a_3 * T + a_4 * T^2 + \frac{a_5}{T - 227} + \frac{a_6}{648 - T} + (P - 70) * \left[a_7 + a_8 * \ln(T) + a_9 * T + a_{10} * T^2 + \frac{a_{11}}{648 - T} \right] \quad (11)$$

A.1.1 Two-Ion Interaction

Table 16: List of two-ion interactions used in the Pitzer Model

Ion 1	Ion 2	P_{ion}	a1	a2	a3	a4	a5	a6	a7	a8	a9	T	P	Eqn.	Ref.
H	Cl	B0	0.052098	0.000627	-2.18E-06	0	0	0	0.100115	48.79791	5.50E-07	0.0 - 250.0	1 - 500	(8)	[208], [209]
H	Cl	B1	2.195496	-0.00778	1.85E-05	0	0	0	-0.4065	-461.358	-7.40E-06	0.0 - 250.0	1 - 500	(8)	[208], [209]
H	SO4	B0	-1.48304	0.017793	-6.30E-05	7.04E-08	0	0	0	0	0	0.0 - 225.0	1	(8)	[208]
H	SO4	C0	-2.5412	0.021434	-5.70E-05	4.81E-08	0	0	0	0	0	0.0 - 225.0	1	(8)	[208]
H	Na	θ	-4.05426	0.048136	0	0	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
H	K	θ	-55.8754	0.202784	0	0	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
H	Mg	θ	0	0.516539	0	-0.00313	5.83E-06	0	0	0	0	0.0 - 120.0	1	(5)	[15]
H	Ca	θ	0	0.096862	0	0	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
H	NH4	θ	-0.019	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]
Na	Cl	B0	14.37832	0.005608	-422.185	-2.51227	0	-2.62E-06	4.438545	-1.70502	1.23E-05	0.0 - 250.0	1 - 500	(9)	[209], [211]
Na	Cl	B1	-0.48306	0.001407	119.312	0	0	0	0	-4.23433	4.35E-06	0.0 - 250.0	1 - 500	(9)	[209], [211]
Na	Cl	C0	-0.10059	-1.81E-05	8.611855	0.012488	0	3.41E-08	0.068304	0.293923	-6.58E-07	0.0 - 250.0	1 - 500	(9)	[209], [211]
Na	OH	B0	0.748451	-0.00105	0	0	-98.8884	0	0	0	0	0.0 - 250.0	1	(4)	[208]
Na	OH	B1	1.20223	-0.0013	0	0	-206.112	0	0	0	0	0.0 - 250.0	1	(4)	[208]
Na	OH	C0	-0.09113	0.000118	0	0	17.30006	0	0	0	0	0.0 - 250.0	1	(4)	[208]

Na	HCO3	B0	-37.2624	-0.01446	0	682.886	6.899586	-1.16E-05	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
Na	HCO3	B1	-61.4635	-0.02447	0	1129.389	11.41086	0.000178	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
Na	CO3	B0	-60.5388	-0.0233	0	1108.376	11.19856	5.98E-05	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
Na	CO3	B1	-237.516	-0.09989	0	4412.512	44.58207	8.16E-05	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
Na	CO3	C0	0.0052	0	0	0	0	-3.25E-06	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
Na	SO4	B0	81.692	0.03011	-2321.94	-14.378	-0.6665	-1.04E-05	0	0	5.33E-05	0.0 - 250.0	1 - 500	(9)	[209], [212]
Na	SO4	B1	1004.63	0.577454	-21843.4	-189.111	-0.20355	-0.00032	1467.722	0	0.000129	0.0 - 250.0	1 - 500	(9)	[209], [212]
Na	SO4	C0	-80.7817	-0.03545	2024.388	14.61977	-0.0917	1.44E-05	-2.42272	0	-2.91E-06	0.0 - 250.0	1 - 500	(9)	[209], [212]
Na	K	θ	-0.05023	0	14.02131	0	0	0	0	0	0	0.0 - 250.0	1	(9)	[212]
Na	Mg	θ	0	-0.06334	0	0.000447	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
Na	Ca	θ	0.05	0	0	0	0	0	0	0	0	0.0 - 250.0	1	(1)	[212]
Na	NH4	θ	0.0044	0	0	0	0	0	0	0	0	25	1	(1)	[210]
K	Cl	B0	26.73756	0.010072	-758.485	-4.70624	0	-3.76E-06	0	0	1.28E-05	0.0 - 250.0	1 - 500	(9)	[209], [212]
K	Cl	B1	-7.4156	0	322.893	1.164386	0	0	0	-5.94578	8.95E-06	0.0 - 250.0	1 - 500	(9)	[209], [212]
K	Cl	C0	-3.30531	-0.0013	91.27121	0.58645	0	4.96E-07	0	0	-7.13E-07	0.0 - 250.0	1 - 500	(9)	[209], [212]
K	OH	B0	-0.59064	0.000788	0	0	147.0094	0	0	0	0	0.0 - 170.0	1	(4)	[208]
K	OH	B1	12.65747	-0.01713	0	0	-2151.13	0	0	0	0	0.0 - 170.0	1	(4)	[208]
K	OH	C0	0.136927	-0.0002	0	0	-22.316	0	0	0	0	0.0 - 170.0	1	(4)	[208]

K	HCO3	B0	-0.30882	0.001	0	0.000699	-4.70E-06	-2.71E-06	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
K	HCO3	B1	-0.2802	0.0011	0	0.000937	6.16E-06	0.00017	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
K	CO3	B0	-0.19916	0.0011	0	1.81E-05	0	3.50E-05	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
K	CO3	B1	0.133058	0.00436	0	0.00119	0	0.000165	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
K	CO3	C0	0.0005	0	0	0	0	-8.48E-07	0	0	0	0.0 - 90.0	1 - 500	(7)	[209], [212]
K	SO4	B0	40.79088	0.008269	-1418.43	-6.74729	0	0	0	0	-2.32E-05	0.0 - 250.0	1 - 500	(9)	[209], [212]
K	SO4	B1	-13.167	0.023579	2067.126	0	0	0	0	0	0.000364	0.0 - 250.0	1 - 500	(9)	[209], [212]
K	SO4	C0	-0.0188	0	0	0	0	0	0	0	2.91E-05	0.0 - 250.0	1 - 500	(9)	[209], [212]
K	Mg	θ	-1048.6	5.878668	0	-0.00792	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
K	Ca	θ	0.1156	0	0	0	0	0	0	0	0	0.0 - 250.0	1	(1)	[211]
Mg	Cl	B0	0.35093	0.031018	-0.00023	3.84E-07	-9.88E-05	6.31185	-0.05596	0.000164	-1.58E-07	0.0 - 200.0	1 - 175	(10)	[213]
Mg	Cl	B1	1.6508	0.031018	-0.00023	3.84E-07	0.002666	6.31185	-0.05596	0.000164	-1.58E-07	0.0 - 200.0	1 - 175	(10)	[213]
Mg	Cl	C0	0.002301	0.031018	-0.00023	3.84E-07	-7.62E-05	6.31185	-0.05596	0.000164	-1.58E-07	0.0 - 200.0	1 - 175	(10)	[213]
Mg	HCO3	B0	13697.1	8.25084	-0.00434	-273406	-2607.12	0	0	0	0	0.0 - 90.0	1	(3)	[212]
Mg	HCO3	B1	-157840	-92.7779	0.047764	3203210	29927.15	0	0	0	0	0.0 - 90.0	1	(3)	[212]
Mg	SO4	B0	165424	-3907.31	658.4623	-1.15985	0.000383	-7748191	0	0	0	0.0 - 120.0	1	(5)	[15]
Mg	SO4	B1	55777.55	-1872.46	340.9427	-0.92295	0.00047	-1819568	0	0	0	0.0 - 120.0	1	(5)	[15]
Mg	SO4	B2	0	13318.13	-3193.09	21.9253	-0.01912	0	0	0	0	0.0 - 120.0	1	(5)	[15]

Mg	SO4	C0	2488.001	-73.3291	12.70537	-0.02426	6.94E-06	-67851.9	0	0	0	0.0 - 120.0	1	(5)	[15]
Mg	Ca	θ	-4785.63	225.3627	-42.2631	0.123292	-5.95E-05	0	0	0	0	0.0 - 120.0	1	(5)	[15]
Mg	NH4	θ	0.0124	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]
Ca	Cl	B0	-94.1896	-0.04048	2345.504	17.09123	-0.92289	1.51E-05	-1.39082	0	1.31E-05	0.0 - 250.0	1 - 500	(9)	[209], [211]
Ca	Cl	B1	3.4787	-0.01542	0	0	0	3.18E-05	0	0	-2.46E-05	0.0 - 250.0	1 - 500	(9)	[209], [211]
Ca	Cl	C0	19.3056	0.009771	-428.384	-3.57996	0.088207	-4.62E-06	9.911135	0	-1.27E-07	0.0 - 250.0	1 - 500	(9)	[209], [211]
Ca	OH	B0	415.173	-1.58106	0	0.000264	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
Ca	OH	B1	0	-0.2303	0	0	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
Ca	OH	B2	0	-5.72	0	0	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
Ca	HCO3	B0	29576.53	18.44731	-0.00999	-576521	-5661.12	0	0	0	0	0.0 - 90.0	1	(3)	[212]
Ca	HCO3	B1	-1028.85	-0.37259	8.97E-05	26492.24	183.1316	0	0	0	0	0.0 - 90.0	1	(3)	[212]
Ca	SO4	B0	-0.015	0	0	0	0	0	0	0	0	0.0 - 250.0	1 - 1516	(1)	[214]
Ca	SO4	B1	3	0	0	0	0	0	0	0	0	0.0 - 250.0	1 - 1516	(1)	[214]
Ca	SO4	B2	-129.399	0	0.400431	0	0	0	0	0	0	0.0 - 250.0	1 - 1516	(6)	[214]
NH4	SO4	B0	0.0545	0	3.15E-05	0	0	0	0	0	0	0.0 - 50.0	1 - 500	(2)	[209], [215], [216]
NH4	SO4	B1	0.878	0	5.85E-05	0	0	0	0	0	0	0.0 - 50.0	1 - 500	(2)	[209], [215], [216]
NH4	SO4	C0	-0.00219	0	-9.92E-07	0	0	0	0	0	0	0.0 - 50.0	1 - 500	(2)	[209], [215], [216]

Cl	OH	θ	0.110486	0	0	0	-49.3613	0	0	0	0	0.0 - 250.0	1	(8)	[208]
Cl	HCO ₃	θ	0.03	0	0	0	0	0	0	0	0	25.00	1	(1)	[217]
Cl	CO ₃	θ	-0.02	0	0	0	0	0	0	0	0	25.00	1	(1)	[217]
Cl	SO ₄	θ	0.07	0	0	0	0	0	0	0	0	0.0 - 250.0	1	(1)	[208]
OH	SO ₄	θ	0.230122	-0.00123	7.78E-07	0	0	0	0	-21.0214	0	0.0 - 250.0	1	(8)	[208]
HCO ₃	CO ₃	θ	-0.04	0	0	0	0	0	0	0	0	25.00	1	(1)	[217]
HCO ₃	SO ₄	θ	0.01	0	0	0	0	0	0	0	0	25.00	1	(1)	[217]
CO ₃	OH	θ	0.1	0	0	0	0	0	0	0	0	25.00	1	(1)	[217]
CO ₃	SO ₄	θ	0.01	0	0	0	0	0	0	0	0	25.00	1	(1)	[217]
NH ₃	NH ₃	λ	0.01478	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]
NH ₃	Na	λ	0.14716	0	-0.00054	5.33E-07	0	0	0	0	0	40.0 - 160.0	1	(6)	[218]
NH ₃	K	λ	0.0454	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]
NH ₃	Mg	λ	-0.21	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]
NH ₃	Ca	λ	-0.081	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]
NH ₃	Cl	λ	-0.12282	-0.17183	0.000541	-5.33E-07	0	0	0	0	0	40.0 - 120.0	1	(6)	[218]
NH ₃	OH	λ	0.103	0	0	0	0	0	0	0	0	25.00	1	(1)	[218]
NH ₃	HCO ₃	λ	0.2857	-99.466	0	0	0	0	0	0	0	37.0 - 197.0	1	(6)	[210]
NH ₃	CO ₃	λ	-0.3391	151.28	0	0	0	0	0	0	0	37.0 - 197.0	1	(6)	[218]
NH ₃	SO ₄	λ	0.14	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]
NH ₃	NH ₂ COO	λ	-0.03933	25.263	0	0	0	0	0	0	0	37.0 - 197.0	1	(6)	[218]
CO ₂	H	λ	-0.005	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]
CO ₂	Na	λ	-5496.38	-3.32657	0.001753	109399.3	1047.022	0	0	0	0	0.0 - 160.0	1	(3)	[212], [218]
CO ₂	K	λ	2856.528	1.767008	-0.00095	-55954.2	-546.074	0	0	0	0	0.0 - 90.0	1	(3)	[212]

CO2	Mg	λ	-479.363	-0.54184	0.000388	3589.474	104.3453	0	0	0	0	0.0 - 90.0	1	(3)	[212]		
CO2	Ca	λ	-12774.6	-8.10156	0.004425	245541.5	2452.51	0	0	0	0	0.0 - 90.0	1	(3)	[212]		
CO2	Cl	λ	1659.945	0.996433	-0.00052	-33159.6	-315.828	0	0	0	0	0.0 - 160.0	1	(3)	[212], [218]		
CO2	OH	λ	0.005	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]		
CO2	HCO ₃	λ	0.084301	-16.148	0	0	0	0	0	0	0	37.0 - 197.0	1	(6)	[218]		
CO2	CO3	λ	0.01	0	0	0	0	0	0	0	0	25.00	1	(1)	[210]		
CO2	SO4	λ	2274.657	1.827095	-0.00114	-33927.8	-457.016	0	0	0	0	0.0 - 90.0	1	(4)	[212]		
Ion 1	Ion 2	P	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	T	P	Eqn.	Ref.
NH4	Cl	B0	-0.92969	0.220237	-0.00103	5.49E-07	-0.91506	0	0.048877	-0.01096	5.62E-05	-3.63E-08	0.019357	25-250	1 - 350	(11)	[219]
NH4	Cl	B1	0.613399	-0.14256	0.001094	0	0	0	0	0	0	0	0	25-251	1 - 350	(11)	[219]
NH4	Cl	C0	-0.0022	0	-7.41E-06	0	0.212221	0	0	0	0	0	0	25-252	1 - 350	(11)	[219]

A.1.2 Three-Ion Interaction

Table 17: List of three-ion interactions used in Pitzer Model

Ion 1	Ion 2	Ion 3	P	a1	a2	a3	a4	a5	a6	a7	a8	T	P	Eqn.	Ref.
H	Na	Cl	ψ	3.593046	-0.01456	0	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
H	Na	SO4	ψ	2.478988	0.004762	0	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
H	K	Cl	ψ	43.64163	-0.80328	0.111868	2.12E-05	0	0	0	0	0.0 - 120.0	1	(5)	[15]
H	K	SO4	ψ	-40.4662	0.129875	0	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
H	Mg	Cl	ψ	-1357.29	61.24457	-11.3092	0.029271	-1.12E-05	-33.6837	0	0	0.0 - 120.0	1	(5)	[15]
H	Ca	Cl	ψ	19.8165	-0.20841	0.017166	0.000108	0	0	0	0	0.0 - 120.0	1	(5)	[15]
H	NH4	Cl	ψ	-0.0091	0	0	0	0	0	0	0	25.00	1	(1)	[220]
H	NH4	SO4	ψ	-0.02245	0	0	0	0	0	0	0	25.00	1	(1)	[210]
Na	Cl	OH	ψ	12.7602	0.003665	0	0	-355.227	-2.21051	0.003231	-27.1989	0.0 - 250.0	1	(8)	[208]
Na	Cl	HCO3	ψ	0.015	0	0	0	0	0	0	0	25.00	1	(1)	[217]
Na	Cl	CO3	ψ	0.0085	0	0	0	0	0	0	0	25.00	1	(1)	[15]
Na	Cl	SO4	ψ	-0.009	0	0	0	0	0	0	0	0.0 - 250.0	1	(1)	[211]
Na	OH	CO3	ψ	-0.017	0	0	0	0	0	0	0	25.00	1	(1)	[217]
Na	OH	SO4	ψ	0.101804	-7.30E-05	0	0	-25.3106	0	0	0	0.0 - 250.0	1	(8)	[208]
Na	HCO3	CO3	ψ	0.002	0	0	0	0	0	0	0	25.00	1	(1)	[217]
Na	HCO3	SO4	ψ	-0.005	0	0	0	0	0	0	0	25.00	1	(1)	[217]

Na	CO3	SO4	ψ	-0.005	0	0	0	0	0	0	0	0.0 - 120.0	1	(1)	[217]
Na	K	Cl	ψ	0.013421	0	-5.10213	0	0	0	0	0	0.0 - 250.0	1	(9)	[211]
Na	K	OH	ψ	-184.025	1.091896	0	-0.0016	0	0	0	0	0.0 - 120.0	1	(5)	[15]
Na	K	HCO3	ψ	-0.003	0	0	0	0	0	0	0	25.00	1	(1)	[217]
Na	K	CO3	ψ	0.003	0	0	0	0	0	0	0	25.00	1	(1)	[217]
Na	K	SO4	ψ	0.034812	0	-8.21657	0	0	0	0	0	0.0 - 250.0	1	(9)	[211]
Na	Mg	Cl	ψ	0	-2.67634	0.618873	-0.00368	2.64E-06	0	0	0	0.0 - 120.0	1	(5)	[15]
Na	Mg	SO4	ψ	-73.6843	0.464812	0	-0.00078	0	0	0	0	0.0 - 120.0	1	(5)	[15]
Na	Ca	Cl	ψ	-0.003	0	0	0	0	0	0	0	0.0 - 250.0	1	(1)	[211]
Na	Ca	OH	ψ	0	5086.727	-1196.01	7.53769	-0.00584	0	0	0	0.0 - 120.0	1	(5)	[15]
Na	Ca	SO4	ψ	-0.012	0	0	0	0	0	0	0	0.0 - 250.0	1	(1)	[211]
Na	NH4	Cl	ψ	-0.0031	0	0	0	0	0	0	0	25.00	1	(1)	[210]
Na	NH4	SO4	ψ	-0.00326	0	0	0	0	0	0	0	25.00	1	(1)	[210]
K	Cl	OH	ψ	-0.00354	2.02E-05	0	0	-1.7041	0	0	0	0.0 - 180.0	1	(8)	[208]
K	Cl	HCO3	ψ	-0.0037	0	0	0	0	0	0	0	25.00	1	(1)	[217]
K	Cl	CO3	ψ	0.004	0	0	0	0	0	0	0	25.00	1	(1)	[217]
K	Cl	SO4	ψ	-0.21248	0.000285	37.56196	0	0	0	0	0	0.0 - 250.0	1	(9)	[211]
K	OH	CO3	ψ	-0.01	0	0	0	0	0	0	0	25.00	1	(1)	[217]

K	OH	SO4	ψ	-42.0903	0.131441	0	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
K	HCO3	CO3	ψ	0.012	0	0	0	0	0	0	0	25.00	1	(1)	[217]
K	CO3	SO4	ψ	-0.009	0	0	0	0	0	0	0	25.00	1	(1)	[217]
K	Mg	Cl	ψ	332.1117	-2.40744	0	0.005377	-3.73E-06	0	0	0	0.0 - 120.0	1	(5)	[15]
K	Ca	Cl	ψ	0.047628	0	-27.0771	0	0	0	0	0	0.0 - 250.0	1	(9)	[211]
Mg	Cl	HCO3	ψ	-0.096	0	0	0	0	0	0	0	25.00	1	(1)	[217]
Mg	Cl	SO4	ψ	-669.972	5.835016	0	-0.01647	1.48E-05	0	0	0	0.0 - 120.0	1	(5)	[15]
Mg	HCO3	SO4	ψ	-0.161	0	0	0	0	0	0	0	25.00	1	(1)	[217]
Mg	Ca	Cl	ψ	-1.24709	-16.4105	3.898083	-0.02566	2.08E-05	0	0	0	0.0 - 120.0	1	(5)	[15]
Mg	Ca	SO4	ψ	-2574.07	54.25584	-8.00941	0	0	0	0	0	0.0 - 120.0	1	(5)	[15]
Mg	NH4	Cl	ψ	-0.0249	0	0	0	0	0	0	0	25.00	1	(1)	[210]
Mg	NH4	SO4	ψ	-0.0439	0	0	0	0	0	0	0	25.00	1	(1)	[210]
Ca	Cl	OH	ψ	98.19793	-0.82029	0	0.00152	0	0	0	0	0.0 - 120.0	1	(5)	[15]
Ca	Cl	SO4	ψ	-0.018	0	0	0	0	0	0	0	0.0 - 250.0	1	(1)	[211]
NH3	NH3	Na	μ	0.000416	-0.18982	0	0	0	0	0	0	40.0 - 160.0	1	(6)	[218]
NH3	NH3	K	μ	-0.00032	0	0	0	0	0	0	0	25.00	1	(1)	[210]
NH3	NH3	NH4	μ	-0.00075	0	0	0	0	0	0	0	25.00	1	(1)	[210]
NH3	NH3	Cl	μ	-0.00073	-0.18982	0	0	0	0	0	0	40.0 - 160.0	1	(6)	[218]
NH3	NH3	CO3	μ	0.000625	0	0	0	0	0	0	0	25.00	1	(1)	[210]
NH3	K	OH	ζ	0.0231	0	0	0	0	0	0	0	25.00	1	(1)	[210]

NH3	Ca	Cl	ζ	-0.00804	0	0	0	0	0	0	0	25.00	1	(1)	[210]
NH3	NH4	SO4	ζ	-0.00919	0	0	0	0	0	0	0	25.00	1	(1)	[210]
CO2	H	Cl	ζ	-804.122	-0.47047	0.000241	16334.39	152.3839	0	0	0	0.0 - 90.0	1	(3)	[212]
CO2	Na	Cl	ζ	-379.459	-0.25801	0.000148	6879.031	73.74512	0	0	0	0.0 - 90.0	1	(3)	[212]
CO2	Na	SO4	ζ	67030.02	37.93052	-0.01895	-1399082	-12630.3	0	0	0	0.0 - 90.0	1	(3)	[212]
CO2	K	Cl	ζ	-379.686	-0.25789	0.000147	6853.264	73.79977	0	0	0	0.0 - 90.0	1	(3)	[212]
CO2	K	SO4	ζ	-2907.03	-2.86076	0.001951	30756.87	611.3756	0	0	0	0.0 - 90.0	1	(3)	[212]
CO2	Mg	Cl	ζ	-1342.6	-0.77229	0.000392	27726.81	253.6232	0	0	0	0.0 - 90.0	1	(3)	[212]
CO2	Mg	SO4	ζ	-7374.24	-4.60833	0.002489	143162.6	1412.303	0	0	0	0.0 - 90.0	1	(3)	[212]
CO2	Ca	Cl	ζ	-166.065	-0.018	-2.47E-05	5256.844	27.37745	0	0	0	0.0 - 90.0	1	(3)	[212]
CO2	NH4	Cl	ζ	-0.00175	0	0	0	0	0	0	0	40.0 - 160.0	1	(1)	[218]

A.2 CO₂-Hydrate Experimental Data

This section includes the experimental data used in the modeling of chloride removal using CO₂-Hydrate. The data collected includes initial concentration of cations and anions, as well as the operating conditions of the process. The output responses reported are the final concentration of the ions present after solid separation and CO₂ degasification.

Table 18: CO₂-hydrate data used in the machine learning process.

Initial Concentration						Operating Conditions		Final Concentration						Ref.
<i>Na</i> ⁺	<i>K</i> ⁺	<i>Mg</i> ²⁺	<i>Ca</i> ²⁺	<i>Cl</i> ⁻	<i>SO</i> ₄ ²⁻	<i>T</i>	<i>P</i>	<i>Na</i> ⁺	<i>K</i> ⁺	<i>Mg</i> ²⁺	<i>Ca</i> ²⁺	<i>Cl</i> ⁻	<i>SO</i> ₄ ²⁻	
10247.78	716.32	1199.69	390.89	20934.98	2419.57	280	29	2561.945	121.7744	299.9225	93.8136	5233.745	653.2839	[221]
10247.78	716.32	1199.69	390.89	20934.98	2419.57	280	80	2254.512	107.448	263.9318	85.9958	4396.346	556.5011	
10247.78	716.32	1199.69	390.89	20934.98	2419.57	280	100	1844.6	42.9792	215.9442	70.3602	3558.947	411.3269	
8689	382.7	981	647	17133	6465	277.15	25	1258	61.5	189	141	7587	2451	[222]
24609.11	4532.951	1028.05	4874.56	53620.85	1131.56	271.15	31	6644.46	1142.304	259.0686	1277.135	13941.42	294.2056	[223]
24609.11	4532.951	1028.05	4874.56	53620.85	1131.56	283.15	31	2300.952	362.6361	85.84218	431.3986	4825.877	101.8404	
24609.11	4532.951	1028.05	4874.56	53620.85	1131.56	277.15	31	1439.633	203.9828	49.75762	252.5022	3217.251	67.8936	
1276	56	165	68	2712.894	375.4891	277	29	275.39	13.01	34.29	14.93	593.799	82.18718	[224]
2657	125	324	133	5649.027	781.8767	277	29	495.07	23.6	60.11	25.59	1063.51	147.1995	
5231	248	625	268	11121.59	1539.329	277	29	1264.97	65.34	141.58	62.1	2679.007	370.7989	
7808	369	948	399	16600.53	2297.664	277	29	1785.54	78.7	212.59	87.9	3679.141	509.2265	
10849	516	1288	547	23065.97	3192.54	277	29	2652.22	111.84	309.56	131.81	5435.044	752.2595	
9000	600	2000	700	19134.83	2648.434	274	35	2610	78	780	217	5357.752	741.5616	[225]
9000	600	0	0	0	0	274	35	2570	82.19	0	0	0	0	
9000	600	0	700	0	0	274	35	2563.3	100	0	223	0	0	
9000	600	2000	700	0	0	274	35	2776.8	108.08	750.21	239.26	0	0	
9000	600	2000	700	19134.83	2648.434	277	35	3381.3	124.2	1048	287.875	7261.428	1005.048	

11804.96	0	0	0	18195.04	0	279.45	35	9837.468	0	0	0	15162.53	0	[226]
11804.96	0	0	0	18195.04	0	278.35	30	11411.46	0	0	0	17588.54	0	
11804.96	0	0	0	18195.04	0	279.15	30	8932.421	0	0	0	13767.58	0	
11804.96	0	0	0	18195.04	0	279.15	35	7830.624	0	0	0	12069.38	0	
11804.96	0	0	0	18195.04	0	281.25	40	9050.47	0	0	0	13949.53	0	
7869.974	0	0	0	12130.03	0	281.65	40	4682.635	0	0	0	7217.365	0	
7869.974	0	0	0	12130.03	0	279.15	30	4210.436	0	0	0	6489.564	0	
7869.974	0	0	0	12130.03	0	278.15	30	3895.637	0	0	0	6004.363	0	
5902.481	0	0	0	9097.519	0	276.15	20	4721.985	0	0	0	7278.015	0	
5902.481	0	0	0	9097.519	0	278.15	30	5508.982	0	0	0	8491.018	0	
5902.481	0	0	0	9097.519	0	280.65	40	4131.737	0	0	0	6368.263	0	
5902.481	0	0	0	9097.519	0	281.15	40	4053.037	0	0	0	6246.963	0	
5902.481	0	0	0	9097.519	0	280.35	40	3856.287	0	0	0	5943.713	0	
3934.987	0	0	0	6065.013	0	279.15	30	2321.642	0	0	0	3578.358	0	
3934.987	0	0	0	6065.013	0	280.25	35	3226.689	0	0	0	4973.311	0	
3934.987	0	0	0	6065.013	0	280.45	35	2872.541	0	0	0	4427.459	0	
1967.494	0	0	0	3032.506	0	279.15	30	1298.546	0	0	0	2001.454	0	
1967.494	0	0	0	3032.506	0	279.65	30	944.3969	0	0	0	1455.603	0	
1967.494	0	0	0	3032.506	0	280.35	37	1573.995	0	0	0	2426.005	0	
8819	555	1340	678	21100	2295	276.15	30	3783.351	238.095	574.86	290.862	9051.9	984.555	
11804.96	0	0	0	18195.04	0	274.2	26	1404.79	0	0	0	3038.571	0	[227]
11804.96	0	0	0	18195.04	0	274.2	26	1570.06	0	0	0	3256.912	0	
11804.96	0	0	0	18195.04	0	274.2	26	1593.67	0	0	0	2674.671	0	
11804.96	0	0	0	18195.04	0	274.2	26	1605.475	0	0	0	2656.476	0	
11804.96	0	0	0	18195.04	0	274.2	26	1629.085	0	0	0	2929.401	0	
11804.96	0	0	0	18195.04	0	274.2	26	1322.156	0	0	0	2838.426	0	
11804.96	0	0	0	18195.04	0	274.2	24.7	2915.825	0	0	0	5003.636	0	
11804.96	0	0	0	18195.04	0	274.2	24.7	2266.553	0	0	0	3966.518	0	
15200	985	1512	11720	120500	694	274.2	35	15200	985	1512	11720	120500	694	[194]

12160	788	1209.6	9376	96400	555.2	274.2	35	12160	788	1209.6	9376	96400	555.2	
10640	689.5	1058.4	8204	84350	485.8	274.2	35	4309.2	279.2475	428.652	3322.62	34161.75	196.749	
9120	591	907.2	7032	72300	416.4	274.2	35	4441.44	287.817	441.8064	3424.584	35210.1	202.7868	
7600	492.5	756	5860	60250	347	274.2	35	3974.8	257.5775	395.388	3064.78	31510.75	181.481	
27500	490	607	3900	48744	291	274.2	35	15100.88	269.0702	333.3175	2141.579	26766.44	159.7947	
24609	6114	1028	4875	53621	1132	274.2	35	15678.32	3895.21	654.9355	3105.847	34161.77	721.1936	
27500	490	607	3900	48744	291	274.2	35	14384.61	256.3077	317.5077	2040	25496.86	152.2154	
7600	492.5	756	5860	60250	347	274.2	35	3972.727	257.4432	395.1818	3063.182	31494.32	181.3864	
9120	591	907.2	7032	72300	416.4	274.2	35	4439.206	287.6722	441.5841	3422.861	35192.39	202.6848	
10640	689.5	1058.4	8204	84350	485.8	274.2	35	4305.203	278.9885	428.2544	3319.538	34130.06	196.5665	
13594.16	246.6934	307.4243	1987.766	24681.39	147.1882	274.2	9.8	7219.547	131.0132	163.266	1055.657	13107.71	78.16823	
8665.389	255.8139	358.333	2587.066	28644.86	167.4047	274.2	9.8	5825.808	171.9857	240.91	1739.304	19258.16	112.5475	
12268.24	248.2496	319.6412	2136.969	25630.06	151.9568	274.2	9.8	7047.632	142.6099	183.6216	1227.607	14723.49	87.29333	
9660.286	272.9811	379.3704	2720.659	30334.95	177.5144	274.2	9.8	5116.152	144.5726	200.9171	1440.879	16065.59	94.0128	
16990.33	388.4539	516.2311	3557.495	41344.75	243.7804	274.2	9.8	6949.785	158.8945	211.1609	1455.17	16911.8	99.71677	
13349.82	237.8731	294.6729	1893.296	23663.15	141.2681	274.2	9.8	7926.9	141.2451	174.9719	1124.208	14050.79	83.88265	
11665.37	297.7425	405.5637	2858.161	32452.45	190.5455	274.2	9.8	5814.554	148.4085	202.1515	1424.638	16175.79	94.9766	
16725.66	306.5789	383.2792	2486.615	30767.52	183.3769	274.2	9.8	7834.259	143.6008	179.5271	1164.725	14411.43	85.8933	
11199.7	246.4263	324.417	2215.996	25991.43	153.5023	274.2	9.8	7032.43	154.7341	203.7054	1391.451	16320.34	96.38597	
18094.17	364.9435	469.4642	3135.772	37644.75	223.2261	274.2	9.8	7609.384	153.4746	197.43	1318.728	15831.25	93.87623	[228]
9509.741	270.8903	377.0221	2707.232	30145.61	176.363	274.2	9.8	5212.811	148.4898	206.6665	1483.983	16524.46	96.67424	
13385.67	245.0457	306.2278	1985.883	24582.64	146.5252	274.2	9.8	7352.319	134.5957	168.2011	1090.781	13502.45	80.48157	
9384.237	252.27	347.2613	2470.013	27776.79	162.8035	274.2	9.8	6240.598	167.7617	230.9317	1642.579	18471.8	108.2657	
11940.9	288.0784	387.6374	2702.063	31031.65	182.5813	274.2	9.8	6018.334	145.1945	195.3732	1361.867	15640.27	92.02285	
10601.77	245.0742	326.5429	2255.774	26150.19	154.1195	274.2	9.8	6995.208	161.7037	215.4579	1488.394	17254.29	101.6904	
15646.75	318.4254	410.6499	2749.718	32925.51	195.1558	274.2	9.8	7276.663	148.0866	190.9764	1278.781	15312.31	90.75896	
15163.41	358.7349	480.5736	3336.351	38477.7	226.5635	274.2	9.8	6652.498	157.3844	210.8375	1463.726	16880.96	99.39806	
9676.019	270.8392	375.7274	2690.451	30045.51	175.8725	274.2	9.8	5242.937	146.7538	203.5873	1457.817	16280.12	95.29628	
9433.814	262.748	364.1614	2605.538	29121.59	170.4909	274.2	9.8	5661.136	157.6724	218.5296	1563.557	17475.57	102.3099	

13122.65	377.9944	527.1589	3791.848	42147.12	246.4931	274.2	9.8	5703.627	164.2915	229.1243	1648.088	18318.82	107.1357
18427.92	342.3499	429.8142	2800.879	34497.43	205.451	274.2	9.8	8016.347	148.926	186.9739	1218.413	15006.76	89.37342
12364.84	303.0427	409.2026	2861.428	32753.88	192.5996	274.2	9.8	6076.049	148.9144	201.0811	1406.098	16095.17	94.64294
10237.03	256.9983	348.8419	2450.778	27917.18	164.0136	274.2	9.8	6169.718	154.8895	210.2423	1477.051	16825.31	98.8488
14047.92	276.2203	352.7625	2339.273	28294.62	167.997	274.2	9.8	7059.448	138.808	177.2725	1175.547	14218.79	84.42291
14903.25	343.209	456.8907	3153.608	36589.89	215.6804	274.2	9.8	6720.805	154.7743	206.0405	1422.158	16500.67	97.26375
8626.552	256.9773	360.5322	2606.405	28819.07	168.3788	274.2	9.8	5752.731	171.3687	240.4257	1738.116	19218.38	112.2857
14770.65	289.2064	368.8937	2443.221	29589.86	175.7257	274.2	9.8	7236.343	141.6862	180.726	1196.967	14496.47	86.09041
17535.51	352.4392	452.9322	3022.393	36320.45	215.4107	274.2	9.8	7576.098	152.269	195.6863	1305.805	15692.01	93.06673
18892.21	365.9325	465.2832	3071.758	37326.01	221.7934	274.2	9.8	7860.892	152.2615	193.6005	1278.133	15531.05	92.28642
11396	269.2109	360.5226	2502.132	28866.02	169.9781	274.2	9.8	5925.486	139.9793	187.4579	1301.013	15009.22	88.38208
15819.87	282.2605	349.8123	2248.623	28090.53	167.686	274.2	9.8	7750.368	138.2832	171.3778	1101.631	13761.93	82.15167
11388.15	260.6779	346.5227	2388.615	27752.58	163.6291	274.2	9.8	6090.385	139.4106	185.3204	1277.432	14842.09	87.5089
11546.48	241.7841	314.245	2120.255	25188.5	149.0934	274.2	9.8	7412.939	155.2275	201.748	1361.222	16171.24	95.71928
11013.39	249.8209	331.3657	2279.495	26540.8	156.5434	274.2	9.8	6772.446	153.622	203.7662	1401.726	16320.7	96.26299
13781.63	361.0877	494.5097	3501.632	39562.12	232.0789	274.2	9.8	6112.745	160.158	219.3363	1553.124	17547.51	102.937
8139.656	260.8181	370.4043	2704.98	29595.74	172.5707	274.2	9.8	5397.843	172.9625	245.635	1793.817	19626.53	114.4409
10352.95	253.3904	342.0544	2391.237	27379.41	161.0048	274.2	9.8	6416.249	157.039	211.9885	1481.971	16968.42	99.78287
8860.284	257.4677	359.6384	2590.345	28752.02	168.1091	274.2	9.8	5800.133	168.5439	235.427	1695.696	18821.69	110.0478
12874.48	294.442	391.3237	2696.906	31340.88	184.7924	274.2	9.8	6374.247	145.7803	193.7471	1335.257	15517.09	91.49204
16277.72	326.2697	418.9786	2793.688	33598.71	199.2955	274.2	9.8	7431.216	148.9509	191.275	1275.393	15338.71	90.98375
11414.31	260.0724	345.3349	2377.974	27658.57	163.106	274.2	9.8	6134.016	139.7621	185.5819	1277.916	14863.63	87.65267
8433.742	273.2588	388.7578	2843.12	31060.32	181.0583	274.2	9.8	4863.901	157.5936	224.204	1639.682	17913.08	104.4198
12753.99	286.3543	378.8774	2600.257	30349.05	179.0823	274.2	9.8	6395.022	143.5819	189.9743	1303.804	15217.42	89.7943
14483.61	309.4002	404.2501	2741.452	32396.55	191.5821	274.2	9.8	6907.666	147.5622	192.799	1307.48	15450.88	91.37122
14343.98	258.0575	320.6861	2067.368	25748.92	153.6322	274.2	9.8	7373.183	132.6483	164.8411	1062.681	13235.62	78.971
16752.58	394.2092	527.4432	3657.59	42232.27	248.7238	274.2	9.8	6802.817	160.0788	214.1819	1485.258	17149.5	101.0007
11969.37	248.4322	322.1179	2168.348	25821.86	152.9061	274.2	9.8	7005.089	145.3952	188.5199	1269.028	15112.28	89.48849
8859.425	256.3001	357.7212	2574.792	28599.55	167.2398	274.2	9.8	5862.015	169.5861	236.6934	1703.662	18923.46	110.6575

9395.775	292.9061	414.1201	3013.11	33093.78	193.1089	274.2	9.8	4815.733	150.1268	212.2541	1544.347	16961.96	98.97648
16106.04	299.7048	376.4661	2454.54	30215.05	179.9306	274.2	9.8	7669.501	142.7158	179.2686	1168.822	14388.04	85.68076
8933.009	273.2263	385.0693	2794.351	30775.61	179.6758	274.2	9.8	4976.205	152.2029	214.506	1556.616	17143.8	100.0899
14772.76	278.2164	350.7737	2295.834	28148.95	167.5154	274.2	9.8	7352.459	138.4694	174.5814	1142.645	14009.84	83.37306
12517.19	268.0605	350.4646	2378.183	28085.51	166.0693	274.2	9.8	6452.739	138.1879	180.6681	1225.977	14478.36	85.6104
9115.553	274.3026	385.5133	2791.093	30814.06	179.9827	274.2	9.8	4952.775	149.0375	209.4619	1516.491	16742.28	97.79045
9296.099	273.1691	382.3316	2758.434	30564.15	178.6454	274.2	9.8	5047.423	148.3203	207.5913	1497.723	16595.15	96.99756
12319.82	317.9755	434.1303	3065.773	34735.42	203.8701	274.2	9.8	5925.96	152.9495	208.8212	1474.669	16708.1	98.06364
11868.28	255.0293	333.7218	2266.486	26742.9	158.1062	274.2	9.8	6523.268	140.174	183.4265	1245.748	14698.93	86.90127
12260.68	357.0446	498.9223	3594.727	39886.82	233.1979	274.2	9.8	5560.942	161.9408	226.2908	1630.421	18091.03	105.769
11128.03	373.9509	535.0196	3930.795	42737.85	248.9004	274.2	9.8	4972.916	167.1119	239.0905	1756.6	19098.77	111.229
10543.83	275.8978	377.7427	2674.185	30220.73	177.2883	274.2	9.8	5474.563	143.2516	196.1314	1388.49	15691.2	92.05159
14016.37	349.0803	473.0199	3318.098	37857.25	222.4762	274.2	9.8	6320.854	157.4221	213.3142	1496.337	17072.19	100.3284
13088.43	246.2108	310.3112	2030.262	24902.24	148.2036	274.2	9.8	7252.171	136.423	171.9404	1124.949	13798.09	82.11816
9583.626	259.6567	357.9783	2549.635	28632.47	167.7756	274.2	9.8	5871.913	159.0923	219.3342	1562.168	17543.19	102.7966
9361.679	275.3284	385.4113	2781.004	30810.19	180.0791	274.2	9.8	4937.171	145.203	203.2586	1466.648	16248.71	94.97026
10184.72	332.1102	472.9599	3461.769	37786.46	220.2303	274.2	9.8	4898.957	159.7485	227.4987	1665.147	18175.68	105.9331
12892.46	308.4351	414.2447	2882.567	33163.93	195.19	274.2	9.8	6254.016	149.6191	200.9464	1398.307	16087.52	94.68493
9833.21	272.639	377.548	2699.338	30193	176.7886	274.2	9.8	5160.313	143.0767	198.1312	1416.57	15844.81	92.77587
13190.79	282.9399	370.0725	2512.243	29656.38	175.3451	274.2	9.8	6618.935	141.9748	185.6966	1260.604	14881.11	87.98544
13702.24	272.2366	348.7166	2319.387	27966.92	165.9633	274.2	9.8	6946.735	138.018	176.7916	1175.878	14178.61	84.13974
11195.3	358.8709	509.6873	3722.324	40724.53	237.4594	274.2	9.8	5119.26	164.1004	233.0641	1702.103	18622.06	108.5828
15718.26	343.0402	450.679	3072.447	36109.97	213.3374	274.2	9.8	7030.034	153.4257	201.5674	1374.16	16150.28	95.41572
10637.77	356.8124	510.3554	3748.732	40768.05	237.4393	274.2	9.8	4907.728	164.6152	235.4521	1729.475	18808.31	109.5425
11972.95	256.9916	336.1925	2282.633	26941.17	159.2864	274.2	9.8	6389.811	137.1532	179.4217	1218.212	14378.16	85.00913
9996.356	266.5573	366.3412	2602.094	29304.62	171.8045	274.2	9.8	5546.301	147.8946	203.2579	1443.726	16259.15	95.32267
10097.39	290.352	404.8045	2910.982	32365.06	189.2935	274.2	9.8	5160.356	148.3869	206.8788	1487.682	16540.44	96.74007
12530.77	251.0758	322.3857	2149.403	25852.83	153.3525	274.2	9.8	6854.857	137.349	176.3585	1175.814	14142.59	83.89027
15371.82	321.2909	417.372	2814.71	33455.34	198.0429	274.2	9.8	7137.397	149.1808	193.793	1306.918	15533.88	91.9547

13496.11	286.1454	373.1302	2525.614	29904.8	176.9076	274.2	9.8	6722.073	142.5218	185.8468	1257.945	14894.83	88.1132
11789.5	284.3766	382.6414	2667.144	30631.75	180.2296	274.2	9.8	5981.39	144.2782	194.1327	1353.172	15540.99	91.43929
9742.932	270.6432	374.9174	2681.349	29982.25	175.5442	274.2	9.8	5264.736	146.246	202.5921	1448.906	16201.35	94.8579
12099.7	251.8918	326.8684	2202.063	26201.88	155.1343	274.2	9.8	6767.167	140.879	182.8122	1231.578	14654.29	86.76413
17004.98	366.9275	480.6623	3267.786	38516.48	227.67	274.2	9.8	7214.414	155.6701	203.9224	1386.368	16340.73	96.58969
11227.66	239.3253	312.5154	2118.189	25045.47	148.125	274.2	9.8	7516.471	160.2187	209.2166	1418.043	16766.94	99.16379
11181.97	248.0257	327.1809	2239.132	26210.92	154.7446	274.2	9.8	6921.74	153.5301	202.5279	1386.043	16224.8	95.78829
12077.37	254.2626	330.9366	2235.976	26525.01	156.965	274.2	9.8	6593.388	138.8094	180.6679	1220.684	14480.77	85.69177
14716.18	290.6124	371.6065	2467.324	29804.65	176.9235	274.2	9.8	7198.736	142.1594	181.7794	1206.945	14579.59	86.54597
13656.71	325.3278	436.509	3034.812	34947.61	205.7222	274.2	9.8	6411.972	152.7448	204.9456	1424.876	16408.27	96.58873
9241.309	267.1099	372.7486	2682.59	29801.15	174.271	274.2	9.8	5374.206	155.3355	216.7689	1560.038	17330.61	101.3458
9890.057	249.9711	339.7916	2390.257	27191.5	159.7113	274.2	9.8	6514.602	164.6565	223.8214	1574.467	17911.1	105.2022
11676.75	264.7389	351.1117	2415.063	28122.48	165.8759	274.2	9.8	6093.12	138.1451	183.2159	1260.22	14674.78	86.55676
13521.99	248.2805	310.5642	2016.007	24929.83	148.5693	274.2	9.8	7091.111	130.2016	162.8639	1057.221	13073.53	77.91169
9261.927	285.9522	403.6388	2932.938	32257.97	188.2815	274.2	9.8	4798.668	148.1538	209.128	1519.575	16713.08	97.54994
10812.14	286.1429	392.6668	2785.417	31412.16	184.2069	274.2	9.8	5521.624	146.1296	200.5301	1422.478	16041.8	94.07218
8696.379	292.8322	419.0903	3079.829	33476.97	194.9562	274.2	9.8	4486.297	151.0666	216.2008	1588.825	17270.13	100.5742
16624.77	364.2178	478.9667	3268.306	38375.09	226.6815	274.2	9.8	7114.779	155.8716	204.9798	1398.713	16423.1	97.01119
16079.4	387.6897	521.6042	3635.448	41756.34	245.6879	274.2	9.8	6654.717	160.4516	215.8742	1504.588	17281.53	101.6819
16812.99	300.774	373.0807	2400.42	29958.01	178.8057	274.2	9.8	7937.534	141.9976	176.1341	1133.256	14143.39	84.41549
12735.54	249.9672	319.0686	2114.733	25592.58	151.9679	274.2	9.8	6947.976	136.3716	174.0705	1153.71	13962.24	82.90731
8795.314	257.6508	360.4131	2599.087	28812.51	168.4225	274.2	9.8	5770.653	169.0461	236.469	1705.275	18904.04	110.5029
10602.33	311.5747	436.09	3146.321	34861.67	203.7637	274.2	9.8	5233.556	153.8005	215.2642	1553.097	17208.53	100.5825
15407.27	309.5575	397.7846	2654.139	31898.31	189.1869	274.2	9.8	7279.598	146.2592	187.9446	1254.023	15071.26	89.38671
8129.883	266.502	379.839	2782.047	30345.84	176.8405	274.2	9.8	5116.916	167.7353	239.0691	1751.009	19099.55	111.3027
11830.68	252.3143	329.523	2233.766	26408.34	156.1816	274.2	9.8	6713.26	143.1745	186.9862	1267.539	14985.28	88.62445
11594.53	254.0011	334.0211	2279.218	26761.98	158.0832	274.2	9.8	6570.657	143.9432	189.2908	1291.64	15166.1	89.58627
16002.58	346.3847	454.1185	3089.711	36388.38	215.0607	274.2	9.8	7097.835	153.6365	201.4211	1370.42	16139.82	95.38869
9826.458	248.8484	338.4051	2381.378	27080.14	159.0462	274.2	9.8	6563.84	166.2249	226.0466	1590.704	18088.89	106.2391

11847.35	246.5229	319.8611	2154.591	25640.29	151.8127	274.2	9.8	7127.056	148.3017	192.42	1296.145	15424.53	91.32652	
10887.17	323.2784	453.2966	3275.487	36234.89	211.7262	274.2	9.8	5269.153	156.4597	219.3857	1585.264	17536.89	102.4709	
12381.22	346.3159	480.3705	3439.374	38413.61	224.8605	274.2	9.8	5702.88	159.5157	221.2622	1584.201	17693.6	103.5724	
14672.95	361.4591	488.6321	3420.315	39110.08	229.9314	274.2	9.8	6442.769	158.7136	214.5542	1501.831	17172.9	100.9609	
8525.541	281.3578	401.4314	2942.698	32069.75	186.8548	274.2	9.8	4470.906	147.5477	210.516	1543.19	16817.8	97.98912	
10869.82	263.4301	354.8304	2475.654	28404.3	167.0938	274.2	9.8	5857.535	141.9574	191.2113	1334.082	15306.53	90.04366	
12559.18	249.1587	319.0207	2120.978	25585.73	151.844	274.2	9.8	7000.852	138.8883	177.8314	1182.295	14262.23	84.64227	
13159.63	241.2935	301.6921	1957.514	24218.06	144.3389	274.2	9.8	7643.108	140.1432	175.2227	1136.924	14065.84	83.83198	
13324.38	273.1702	353.0056	2368.471	28301.48	167.6882	274.2	9.8	6769.994	138.7953	179.3589	1203.398	14379.72	85.20081	
12771.88	247.0714	314.0335	2072.432	25192.8	149.7071	274.2	9.8	7171.535	138.7329	176.3328	1163.691	14146	84.06199	
10793.78	269.0163	364.5865	2557.828	29178.83	171.471	274.2	9.8	5644.385	140.6765	190.653	1337.563	15258.46	89.66725	
10407.49	250.0654	336.1797	2341.427	26913.19	158.3741	274.2	9.8	6635.367	159.431	214.3338	1492.794	17158.7	100.9726	
12419.29	299.5462	403.0462	2809.33	32265.24	189.8412	274.2	9.8	6130.448	147.8629	198.9529	1386.751	15926.87	93.71001	
9503.214	273.5887	381.5149	2744.002	30502.78	178.3952	274.2	9.8	5057.643	145.6048	203.0436	1460.368	16233.68	94.94257	
196.7494	0	0	0	303.2506	0	273.35	30	85.09521	0	0	0	131.1576	0	
11804.96	0	0	0	18195.04	0	273.35	30	6448.76	0	0	0	9939.502	0	[229]
27544.91	0	0	0	42455.09	0	273.35	30	17770.51	0	0	0	27389.76	0	
3147.99	0	0	0	4852.01	0	274.15	44.82	127.7445	0	0	0	196.8932	0	
4249.786	0	0	0	6550.214	0	274.15	44.82	388.024	0	0	0	598.063	0	
7869.974	0	0	0	12130.03	0	274.15	44.82	422.0131	0	0	0	650.4507	0	
0	0	331.8277	0	968.1723	0	274.15	44.82	0	0	2.404549	0	7.015741	0	
0	0	1276.261	0	3723.739	0	274.15	44.82	0	0	1.849653	0	5.396724	0	
0	0	2552.521	0	7447.479	0	274.15	44.82	0	0	29.59445	0	86.34758	0	[230]
0	0	0	18.65825	33.07175	0	274.15	44.82	0	0	0	2.487767	4.409567	0	
0	0	0	64.48693	114.3031	0	274.15	44.82	0	0	0	7.663664	13.58384	0	
0	0	0	180.3427	319.6573	0	274.15	44.82	0	0	0	15.4206	27.33302	0	
0	0	0	1760.144	3119.856	0	274.15	44.82	0	0	0	188.7691	334.5932	0	
15200	985	1512	11720	120500	694	275.2	35	15200	985	1512	11720	120500	694	[231]
15200	985	1512	11720	120500	694	276.2	35	15200	985	1512	11720	120500	694	

15200	985	1512	11720	120500	694	277.2	35	15200	985	1512	11720	120500	694	
15200	985	1512	11720	120500	694	274.2	30	15200	985	1512	11720	120500	694	
15200	985	1512	11720	120500	694	274.2	20	15200	985	1512	11720	120500	694	
15200	985	1512	11720	120500	694	274.2	10	15200	985	1512	11720	120500	694	
11300	0	600	1300	24024.84	3325.256	277.15	117	3726.74	0	240.12	535.6	9145.456	1265.814	
11300	0	600	1300	24024.84	3325.256	277.15	117	4115.46	0	247.2	516.75	9399.319	1300.951	[232]
11300	0	600	1300	24024.84	3325.256	277.15	117	4075.554	0	260.8383	537.8025	9682.757	1340.182	
10000	360	1180	380	18330	2590	278.15	20	1528.814	55.58644	235.6	69.88136	3184.449	361.722	
10000	360	1180	380	18330	2590	278.15	20	1384.711	50.40808	219.5349	64.60595	2926.809	323.8169	
10000	360	1180	380	18330	2590	278.15	20	1456.762	52.99726	227.5674	67.24365	3055.629	342.7695	
10000	360	1180	380	18330	2590	278.15	20	1734.675	62.9841	258.5502	77.41765	3552.506	415.8722	
10000	360	1180	380	18330	2590	278.15	20	1487.791	54.11228	231.0266	68.37957	3111.105	350.9313	[233]
10000	360	1180	380	18330	2590	278.15	20	1383.475	50.36366	219.3971	64.56069	2924.599	323.4918	
10000	360	1180	380	18330	2590	278.15	20	1579.539	57.40928	241.2551	71.73835	3275.141	375.065	
10000	360	1180	380	18330	2590	278.15	20	1122.368	40.98074	190.2879	55.00193	2457.769	254.8097	
10000	360	1180	380	18330	2590	278.15	20	1487.791	54.11228	231.0266	68.37957	3111.105	350.9313	

A.3 Modeling Codebase

This section includes all the coding files used in the modeling of the process, for the simulation and optimization. There are four files for calculating Gibbs free energy of species in bubble column reactor. The file, *IAPWS_model.py*, calculates Gibbs free energy of water using IAPWS-IF97 model. The file, *HKFT_mode.py*, calculates Gibbs free energy of any dissolved species in water using HKFT model. The file, *gases_mode.py*, calculates Gibbs free energy of CO₂ using the Maier-Kelley heat capacity method. The file, *solids_model.py*, calculates Gibbs free energy of some inorganic salts using the same heat capacity method.

Ideality is captured in the standard Gibbs free energy calculated, and non-ideality is captured using fugacity coefficients. There are two files for calculating fugacity coefficients of species in bubble column reactor. The file, *peng_robinson.py*, calculates the fugacity coefficient for pure CO₂ gas using Peng Robinson Equation of State. The file, *pitzer_model.py*, calculates activity coefficient for water and any species dissolved using Pitzer model.

Finally, the file, *simulation.py*, combines all the models to simulate CO₂ absorption dynamics in bubble column reactor. The file, *optimization.py*, combines all the models to optimize sodium removal in bubble column reactor. The file, *CO2_hydrate.py*, contains the Machine Learning process used for model and optimizing chloride removal using Hydrate-Based Formation.

A.3.1 IAPWS_model.py

```
1. from autograd import grad
2. import autograd.numpy as np
3. from scipy.optimize import fsolve
4.
5. def Gibbs_water(P, T):
6.     Pc = 16.53 ## MPa
7.     Tc = 1386 ## K
8.
9.     Tr = Tc/T; Pr = P/Pc;
10.
11.     I = np.array([ 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 2, 2, 2,
12.                  2, 2, 3, 3, 3, 4, 4, 4, 5, 8, 8, 21, 23, 29, 30, 31, 32])
13.
14.     J = np.array([-2, -1, 0, 1, 2, 3, 4, 5, -9, -7, -1, 0, 1, 3, -3, 0, 1,
15.                  3, 17, -4, 0, 6, -5, -2, 10, -8, -11, -6, -29, -31, -38, -39, -40, -41])
16.
17.     n = np.array([ 0.14632971213167, -0.84548187169114, -0.37563603672040e1,
```

```

18.          0.33855169168385e1, 0.95791963387872, 0.15772038513228,
19.          -0.16616417199501e-1, 0.81214629983568e-3, 0.28319080123804e-3,
20.          -0.60706301565874e-3, -0.18990068218419e-1, 0.32529748770505e-1,
21.          -0.21841717175414e-1, -0.52838357969930e-4, -0.47184321073267e-3,
22.          -0.30001780793026e-3, 0.47661393906987e-4, -0.44141845330846e-5,
23.          -0.72694996297594e-15, -0.31679644845054e-4, -0.28270797985312e-5,
24.          -0.85205128120103e-9, -0.22425281908000e-5, -0.65171222895601e-6,
25.          -0.14341729937924e-12, -0.40516996860117e-6, -0.12734301741641e-8,
26.          17424871230634e-9, -0.68762131295531e-18, 0.14478307828521e-19,
27.          0.26335781662795e-22, -0.11947622640071e-22, 0.18228094581404e-23,
28.          -0.93537087292458e-25])
29.
30.     return np.sum( n[i] * (7.1 - P/Pc)**I[i] * (Tc/T - 1.222)**J[i]
31.                   for i in range(34) )
32.
33. def molar_volume(P, T):
34.     R = 0.000461526 ## MJ / kg / K
35.     dGdP= grad(Gibbs_water, 0)(P, T)
36.     return dGdP * R * T
37.
38. def rho_water(P, T):
39.     return 1 / molar_volume(P, T)

```

A.3.2 HKFT_model.py

```

1. import json
2. from autograd import grad
3. import autograd.numpy as np
4. from os.path import abspath
5. from scipy.optimize import fsolve
6. from .IAPWS_model import rho_water
7.
8.
9. def dielectric(rho, T):
10. rhoc = 322/18.015268*1000 ## mol / m3
11. Tc = 647.096 ## K
12.
13. e0 = 1 / 4e-7 / np.pi / 299792458**2
14. ec = 1.60217733e-19
15. k = 1.380658e-23
16. Na = 6.0221367e23
17. Mw = 0.018015268
18. alpha = 1.636e-40
19. mu = 6.138e-30
20.
21. N = np.array([ 0.978224486826, 0.957771379375, 0.237511794148,
22. 0.714692244396, -0.298217036956, -0.108863472196, 0.949327488264e-1,
23. -0.980469816509e-2, 0.165167634970e-4, 0.937359795772e-4,
24. -0.123179218720e-9, 0.196096504426e-2])
25.
26. i = np.array([ 1, 1, 1, 2, 3, 3, 4, 5, 6, 7, 10])
27. j = np.array([ 0.25, 1, 2.5, 1.5, 1.5, 2.5, 2, 2, 5, 0.5, 10])
28.
29. g = 1 + sum( N[k] * (rho/rhoc)**i[k] * (Tc/T)**j[k] for k in range(11)) +
30.     N[11] * (rho/rhoc) * (T/228 - 1)**-1.2
31. A = Na * mu**2 * rho * g / e0 / k / T
32. B = Na * alpha * rho / 3 / e0
33.
34. return (1 + A + 5*B + np.sqrt(9 + 2*A + 18*B + A**2 + 10*A*B + 9*B**2))/
35.     (4 - 4*B)
36.
37. def wj(P, T, wr, Zj):
38.     Na = 6.02252e23; e = 4.80298e-10;
39.     n = Na * e**2 / 2 * 2.39;

```



```

40.
41.     rho = rho_water(P/10, T+273.15)/1000
42.
43.     ag = np.array([ -0.2037662e1,  0.5747000e-2, -0.6557892e-5 ])
44.     bg = np.array([  0.6107361e1, -0.1074377e-1,  0.1268348e-4 ])
45.     f  = np.array([  0.3666666e2, -0.1504956e-9,  0.5017997e-13])
46.
47.     rej_r = Zj**2 / (wr/n + Zj/3.082)
48.
49.     g = ( ag[0] + ag[1]*T + ag[2]*T**2 ) * \
50.         ( 1 - rho/1000 )***(bg[0] + bg[1]*T + bg[2]*T**2) - \
51.         ( ((T - 155)/300)**4.8 + f[0]*((T - 155)/300)**16 ) * \
52.         ( f[1]*(1000 - P)**3 + f[2]*(1000 - P)**4 )
53.
54.     rej = rej_r + Zj * g
55.
56.     return n * (Zj**2/rej - Zj/(3.082 + g))
57.
58. def Z(P, T):
59.     return -1/dielectric(P/10, T)
60.
61.
62. def HKFT(T, P, species):
63.
64.     with open(abspath('./reaction_data/thermo_data.json')) as json_file:
65.         thermo_data = json.load(json_file)
66.
67.         data = thermo_data[species]['Data']
68.
69.         Gf = data['Gf']; Hf = data['Hf']; Sr = data['Sr']
70.         a1 = data['a1']; a2 = data['a2']; a3 = data['a3']; a4 = data['a4']
71.         c1 = data['c1']; c2 = data['c2']
72.         wr = data['wr']
73.         Zj = data['Zj']
74.
75.         Tc = 228. ## K
76.         Psi = 2600. ## bar
77.         Pr = 1. ## bar
78.         Tr = 298.15 ## K
79.
80.         rho = rho_water(P/10, T)/1000
81.         rho_ref = rho_water(Pr/10, Tr)/1000
82.
83.         e = dielectric(rho*1e3, T)
84.         eref = dielectric(rho_ref*1e3, T)
85.
86.         if Zj == 0: wj = wr
87.         else: wj = _wj(P, T-273.15, wr, Zj)
88.
89.         Yr = grad(_Z, 1)(Pr, Tr)
90.
91.
92.         Gns = Gf - Sr*(T - Tr) - c1*(T*np.log(T/Tr) + Tr - T) + \
93.             a1*(P - Pr) + a2*np.log((P + Psi)/(Pr + Psi))*(P + Psi)/(Pr + Psi) + \
94.             a3*(P - Pr)/(T - Tc) + a4*np.log((P + Psi)/(Pr + Psi))/(T - Tc) + \
95.             c2/Tc - c2/Tc*(T - Tc)/(Tr - Tc) - c2*T*np.log(Tr*(T - Tc)/T/
96.             (Tr - Tc))/Tc**2
97.         Gs = wj*(1/e - 1) - wr*(1/eref - 1) + wr*Yr*(T - Tr)
98.         return Gns + Gs

```

A.3.3 gases_model.py

```
1. import json
2. from numpy import log
3. from os.path import abspath
4.
5. def Gas_Calculations(T, P, gas):
6.
7.     with open(abspath('./reaction_data/thermo_data.json')) as json_file:
8.         thermo_data = json.load(json_file)
9.
10.    data = thermo_data[gas]['Data']
11.
12.    Gf = data['Gf']; Hf = data['Hf']; Sr = data['Sr']
13.
14.    Cpa = data['a']
15.    Cpb = data['b'] / 1e3
16.    Cpc = data['c'] * 1e5
17.
18.    Tref = 298.15 # K
19.    R = 8.31446261815324 # J / K / mol
20.
21.    def enthalpy():
22.        dH = Cpa * (T - Tref) + Cpb * (T**2 - Tref**2)/2 - Cpc * (1/T - 1/Tref)
23.        return Hf + dH
24.
25.    def entropy():
26.        dS = Cpa * log(T/Tref) + Cpb * (T - Tref) - Cpc / 2 * (1/T**2 -
27.            1/Tref**2) - R * log(P)
28.        return Sr + dS
29.
30.    dH = enthalpy() - Hf
31.    dG = dH - (T * entropy() - Tref * Sr)
32.    return Gf + dG
```

A.3.4 solids_model.py

```
1. import json
2. import numpy as np
3. from os.path import abspath
4.
5. def Solid_Calculations(T, P, salt):
6.
7.     with open(abspath('./reaction_data/thermo_data.json')) as json_file:
8.         thermo_data = json.load(json_file)
9.
10.    data = thermo_data[salt]['Data']
11.
12.    Gf = data['Gf']; Hf = data['Hf']; Sr = data['Sr']; Vr = data['Vr']
13.    a = data['a']; b = data['b']; c = data['c']; d = data['d']
14.
15.    if d == 0:
16.        b = b / 1000
17.        c = c * 1e5
18.
19.    Tref = 298.15
20.
21.    G = Gf - Sr * (T - Tref)
22.    G = G + a * (T - Tref - T * np.log(T/Tref))
23.    G = G + b * (T*Tref - T**2/2 - Tref**2/2)
24.    G = G + c * (1/Tref - 1/2/T - T/2/Tref**2)
25.    G = G + d * (T**3/6 - T*Tref**2/2 + Tref**3/3)
26.    return G + Vr * (P - 1)
```

A.3.5 peng_robinson.py

```
1. import json
2. import numpy as np
3. from os.path import abspath
4. from scipy.optimize import newton
5. import matplotlib.pyplot as plt
6.
7. def peng_robinson(T, P, gas='CO2(g)':
8.     with open(abspath('./reaction_data/thermo_data.json')) as json_file:
9.         thermo_data = json.load(json_file)
10.
11.     w = thermo_data[gas]['Data']['w']
12.     Tc = thermo_data[gas]['Data']['Tc']
13.     Pc = thermo_data[gas]['Data']['Pc']
14.     Vc = thermo_data[gas]['Data']['Vc']
15.
16.     Tref = 298.15 # K
17.     R = 8.31446261815324e-5 # m3 * bar / K / mol
18.
19.     a = 0.457235 * R**2 * Tc**2 / Pc
20.     b = 0.0777961 * R * Tc / Pc
21.
22.     k = 0.37464 + 1.54226 * w - 0.26992 * w**2
23.     alpha = ( 1 + k * ( 1 - np.sqrt(T/Tc) ) )**2
24.
25.     a = 0.457235 * R**2 * Tc**2 / Pc
26.     b = 0.0777961 * R * Tc / Pc
27.
28.     A = a * alpha * P / R**2 / T**2
29.     B = b * P / R / T
30.
31.     a0 = B**2 + B**3 - A*B
32.     a1 = A - 3*B**2 - 2*B
33.     a2 = B - 1
34.
35.     func = lambda Z: Z**3 + a2*Z**2 + a1*Z + a0
36.     Z = newton(func, 0.99)
37.
38.     numerator = Z + (1 + np.sqrt(2)) * B
39.     denominator = Z + (1 - np.sqrt(2)) * B
40.
41.     ln_phi = (Z - 1) - np.log(Z - B) - A / np.sqrt(8) / B *
42.         np.log(numerator / denominator)
43.     return ln_phi
```

A.3.6 pitzer_model.py

```
1. import json
2. import numpy as np
3. from os.path import abspath
4. from scipy.optimize import fsolve
5. from IAPWS_model import rho_water
6. from HKFT_model import dielectric
7. from autograd import numpy
8.
9. def ln_activity(T, P, solution, species):
10.
11.     cations = np.array([ "H+", "Na+", "K+", "Mg++", "Ca++"])
```

```

12.
13.     anions = np.array(["Cl-", "OH-", "HCO3-", "CO3--", "SO4--", "NH2COO-"])
14.
15.     neutrals = np.array([ "CO2(aq)", "NH3(aq)"])
16.
17.     with open(abspath('./reaction_data/pitzer_data.json')) as json_file:
18.         pitzer_data = json.load(json_file)
19.
20.     substances = np.array(list(solution.keys()))
21.     molarities = np.array(list(solution.values()))
22.     mask = [True if '(s)' not in substance else False for substance in substances]
23.     substances = substances[mask]
24.     molarities = molarities[mask]
25.     z = np.array([ substance.count('+') - substance.count('-') for substance in substances
])
26.     m = molarities
27.     I = 1/2 * np.sum(m * z**2)
28.     Z = np.sum(m * abs(z))
29.     rho = rho_water(P/10, T))/1000
30.
31.     def A():
32.         Na = 6.0221367e23 # Avogadro's constant (1/mol)
33.         e = 1.6021773e-19 # charge on an electron (C)
34.         D = _dielectric(1e3*rho, T)
35.         k = 1.380658e-23 # Boltzmann constant in J/K
36.         E0 = 8.8541878e-12 # permittivity of vacuum in C**2/(J*m)
37.         return 1 / 3 * (2e-3 * np.pi * rho * Na)**(1/2) * (100 * e**2 / (4*np.pi*E0) / D /
k / T)**(3/2)
38.
39.     def Bca(calculation_type):
40.         Bca = 0
41.
42.         for cation in cations:
43.             for anion in anions:
44.                 B0 = 0; B1 = 0; B2 = 0; a = np.zeros(15)
45.
46.                 if cation in substances and anion in substances and \
47.                    anion in pitzer_data[cation]:
48.                     mc = m[substances == cation][0]
49.                     ma = m[substances == anion][0]
50.                 else:
51.                     continue
52.
53.                 if 'B0' in pitzer_data[cation][anion]:
54.                     parameters = pitzer_data[cation][anion]['B0']['parameters']
55.                     equation = pitzer_data[cation][anion]['B0']['Equation']
56.                     a = np.zeros(16)
57.                     a[:len(parameters)] = parameters
58.                     B0 = eval(equation)
59.
60.                 if 'B1' in pitzer_data[cation][anion]:
61.                     parameters = pitzer_data[cation][anion]['B1']['parameters']
62.                     equation = pitzer_data[cation][anion]['B1']['Equation']
63.                     a = np.zeros(16)
64.                     a[:len(parameters)] = parameters
65.                     B1 = eval(equation)
66.
67.                 if 'B2' in pitzer_data[cation][anion]:
68.                     parameters = pitzer_data[cation][anion]['B2']['parameters']
69.                     equation = pitzer_data[cation][anion]['B2']['Equation']
70.                     a = np.zeros(16)
71.                     a[:len(parameters)] = parameters
72.                     B2 = eval(equation)
73.
74.                 a1 = 1.4; a2 = 12
75.                 if B2 == 0: a1 = 2

```

```

76.
77.         x1 = a1 * np.sqrt(I)
78.         x2 = a2 * np.sqrt(I)
79.
80.         if calculation_type == 'water':
81.             e = lambda x: np.exp(-x)
82.             Bca = Bca + mc * ma * (B0 + B1 * e(x1) + B2 * e(x2))
83.
84.         elif calculation_type == 'derivative':
85.             dg = lambda x: -2 * ( 1 - (1 + x + x**2/2) * np.exp(-x) ) / x**2
86.             Bca = Bca + mc * ma * (B1 * dg(x1) + B2 * dg(x2)) / I
87.
88.         elif calculation_type == 'cation':
89.             g = lambda x: 2 * ( 1 - (1 + x) * np.exp(-x) ) / x**2
90.             Bca = Bca + 2 * ma * (B0 + B1 * g(x1) + B2 * g(x2))
91.
92.         elif calculation_type == 'anion':
93.             g = lambda x: 2 * ( 1 - (1 + x) * np.exp(-x) ) / x**2
94.             Bca = Bca + 2 * mc * (B0 + B1 * g(x1) + B2 * g(x2))
95.
96.     return Bca
97.
98. def Cmx(calculation_type):
99.     Cmx = 0
100.
101.     for cation in cations:
102.         for anion in anions:
103.             C0 = 0; C1 = 0; C2 = 0; a = np.zeros(15)
104.
105.             if cation in substances and anion in substances and \
106.                anion in pitzer_data[cation]:
107.                 mc = m[substances == cation][0]
108.                 ma = m[substances == anion][0]
109.             else:
110.                 continue
111.
112.             if 'C0' in pitzer_data[cation][anion]:
113.                 parameters = pitzer_data[cation][anion]['C0']['parameters']
114.                 equation    = pitzer_data[cation][anion]['C0']['Equation']
115.                 a = np.zeros(16)
116.                 a[:len(parameters)] = parameters
117.                 C0 = eval(equation)
118.
119.             if 'C1' in pitzer_data[cation][anion]:
120.                 parameters = pitzer_data[cation][anion]['C1']['parameters']
121.                 equation    = pitzer_data[cation][anion]['C1']['Equation']
122.                 a = np.zeros(16)
123.                 a[:len(parameters)] = parameters
124.                 C1 = eval(equation)
125.
126.             if 'C2' in pitzer_data[cation][anion]:
127.                 parameters = pitzer_data[cation][anion]['C2']['parameters']
128.                 equation    = pitzer_data[cation][anion]['C2']['Equation']
129.                 a = np.zeros(16)
130.                 a[:len(parameters)] = parameters
131.                 C2 = eval(equation)
132.
133.         a3 = 2.5; a4 = 1.34
134.         x1 = a3 * np.sqrt(I)
135.         x2 = a4 * np.sqrt(I)
136.
137.         if C1 == 0 and C2 == 0:
138.             zc = z[substances == cation][0]
139.             za = z[substances == anion][0]
140.             C0 = C0 / 2 / np.sqrt(zc * -za)
141.

```

```

142.             h = lambda x: 4 * ( 6 - (6 + 6*x + 3*x**2 + x**3) * np.exp(-x) ) /
x**4
143.
144.             if calculation_type == 'water':
145.                 e = lambda x: np.exp(-x)
146.                 Cmx = Cmx + mc * ma * (C0 + C1 * e(x1) + C2 * e(x2))
147.
148.             elif calculation_type == 'derivative':
149.                 dh = lambda x: -2 * (h(x1) - np.exp(-x)) / I
150.                 Cmx = Cmx + mc * ma * (C1 * dh(x1) + C2 * dh(x2)) / 2
151.
152.             elif calculation_type == 'all':
153.                 Cmx = Cmx + mc * ma * (C0 + C1 * h(x1) + C2 * h(x2))
154.
155.             elif calculation_type == 'cation':
156.                 Cmx = Cmx + ma * (C0 + C1 * h(x1) + C2 * h(x2))
157.
158.             elif calculation_type == 'anion':
159.                 Cmx = Cmx + mc * (C0 + C1 * h(x1) + C2 * h(x2))
160.
161.             return Cmx
162.
163.     def Theta(calculation_type):
164.         Theta = 0
165.
166.         if isinstance(calculation_type, str):
167.             if calculation_type.count('+') > 0:
168.                 calculation_type = [calculation_type, 'cations']
169.             else:
170.                 calculation_type = [calculation_type, 'anions']
171.
172.         if calculation_type[1] == 'cations': total_ions = cations
173.         elif calculation_type[1] == 'anions': total_ions = anions
174.
175.         for ion_1 in total_ions:
176.             for ion_2 in total_ions:
177.                 theta = 0; E_theta = 0; dE_theta = 0; a = np.zeros(15)
178.
179.                 if calculation_type[0] in ['water', 'derivative'] \
180.                 and np.where(total_ions == ion_2) <= np.where(total_ions == ion_1):
181.                     continue
182.
183.                 if ion_1 in substances and ion_2 in substances and \
184.                 ion_1 in pitzer_data and ion_2 in pitzer_data[ion_1]:
185.                     m1 = m[substances == ion_1][0]
186.                     m2 = m[substances == ion_2][0]
187.                 else:
188.                     continue
189.
190.                 if 'Theta' in pitzer_data[ion_1][ion_2]:
191.                     parameters = pitzer_data[ion_1][ion_2]['Theta']['parameters']
192.                     equation = pitzer_data[ion_1][ion_2]['Theta']['Equation']
193.                     a = np.zeros(16)
194.                     a[:len(parameters)] = parameters
195.                     theta = eval(equation)
196.
197.                 z1 = z[substances == ion_1][0]
198.                 z2 = z[substances == ion_2][0]
199.
200.                 if z1 != z2:
201.
202.                     J0 = lambda x: x / (4 + 4.581 * x**-0.7237 * np.exp(-0.0120 *
x**0.528))
203.                     J1 = lambda x: 1 / (4 + 4.581 / x**0.7237 / np.exp(0.0120 *
x**0.528)) \

```

```

204.         + x * ( 4.581 * 0.7237 / x**1.7237 / np.exp(0.0120 * x**0.528)
205.         \
206.         + 4.581 * 0.0120 * 0.528 / x**(1.7237-0.528) / np.exp(0.0120 *
207.         x**0.528) ) \
208.         / ( 4 + 4.581 / x**0.7237 / np.exp(0.0120 * x**0.528) )**2
209.
210.         x11 = 6 * z1 * z1 * A() * np.sqrt(I)
211.         x22 = 6 * z2 * z2 * A() * np.sqrt(I)
212.         x12 = 6 * z1 * z2 * A() * np.sqrt(I)
213.
214.         E_theta = z1 * z2 / 4 / I * (J0(x12) - J0(x11)/2 - J0(x22)/2)
215.         dE_theta = z1 * z2 / 8 / I**2 * (J1(x12) - J1(x11)/2 - J1(x22)/2)
216.
217.     - E_theta / I
218.
219.     if calculation_type[0] == 'water':
220.         Theta = Theta + m1 * m2 * (theta + E_theta + I * dE_theta)
221.
222.     elif calculation_type[0] == 'derivative':
223.         Theta = Theta + m1 * m2 * (dE_theta)
224.
225.     elif calculation_type[0] == ion_1:
226.         Theta = Theta + 2 * m2 * (theta + E_theta)
227.
228.     return Theta
229.
230. def Lambda(calculation_type):
231.     Lambda = 0
232.
233.     if isinstance(calculation_type, str):
234.         if calculation_type.count('+') > 0:
235.             calculation_type = [calculation_type, 'cations']
236.         else:
237.             calculation_type = [calculation_type, 'anions']
238.
239.     if calculation_type[1] == 'cations': total_ions = cations
240.     elif calculation_type[1] == 'anions': total_ions = anions
241.
242.     for neutral in neutrals:
243.         for ion in total_ions:
244.             lamda = 0; a = np.zeros(15)
245.
246.             if neutral in substances and ion in substances and \
247.             ion in pitzer_data[neutral]:
248.                 mn = m[substances == neutral][0]
249.                 mi = m[substances == ion][0]
250.             else:
251.                 continue
252.
253.             if 'Lambda' in pitzer_data[neutral][ion]:
254.                 parameters = pitzer_data[neutral][ion]['Lambda']['parameters']
255.                 equation = pitzer_data[neutral][ion]['Lambda']['Equation']
256.                 a = np.zeros(16)
257.                 a[:len(parameters)] = parameters
258.                 lamda = eval(equation)
259.
260.             if calculation_type[0] == 'all':
261.                 Lambda = Lambda + mn * mi * lamda
262.
263.             elif calculation_type[0] in neutrals:
264.                 Lambda = Lambda + 2 * mi * lamda
265.
266.             elif calculation_type[0] in total_ions:
267.                 Lambda = Lambda + 2 * mn * lamda
268.
269.     return Lambda

```

```

267.     def PSI(calculation_type):
268.         PSI = 0
269.
270.         if isinstance(calculation_type, str):
271.             if calculation_type.count('+') > 0:
272.                 calculation_type = [calculation_type, 'cations']
273.             else:
274.                 calculation_type = [calculation_type, 'anions']
275.
276.         if calculation_type[1] == 'cations': total_ions = cations
277.         elif calculation_type[1] == 'anions': total_ions = anions
278.
279.         for cation in cations:
280.             for ion in total_ions:
281.                 for anion in anions:
282.                     psi = 0; a = np.zeros(15)
283.
284.                     if calculation_type[0] == 'all':
285.                         if calculation_type[1] == 'cations':
286.                             if np.where(total_ions == ion) <= np.where(total_ions ==
cation):
287.                                 continue
288.                             else:
289.                                 if np.where(total_ions == anion) <= np.where(total_ions ==
ion):
290.                                     continue
291.
292.                             if cation in substances and ion in substances and anion in
substances and \
293.                                 ion in pitzer_data[cation] and anion in pitzer_data[cation][ion]:
294.                                     mc = m[substances == cation][0]
295.                                     mi = m[substances == ion][0]
296.                                     ma = m[substances == anion][0]
297.                                 else:
298.                                     continue
299.
300.                                 if 'PSI' in pitzer_data[cation][ion][anion]:
301.                                     parameters =
pitzer_data[cation][ion][anion]['PSI']['parameters']
302.                                     equation =
pitzer_data[cation][ion][anion]['PSI']['Equation']
303.                                     a = np.zeros(16)
304.                                     a[:len(parameters)] = parameters
305.                                     psi = eval(equation)
306.
307.                                 if calculation_type[0] == 'all':
308.                                     PSI = PSI + mc * mi * ma * psi
309.
310.                                 elif calculation_type[1] == 'cations':
311.                                     if calculation_type[0] == cation:
312.                                         PSI = PSI + mi * ma * psi
313.                                 else:
314.                                     if calculation_type[0] == anion:
315.                                         PSI = PSI + mc * mi * psi
316.
317.         return PSI
318.
319.     def Zeta(calculation_type):
320.         Zeta = 0
321.
322.         for neutral in neutrals:
323.             for cation in cations:
324.                 for anion in anions:
325.                     zeta = 0; a = np.zeros(15)
326.

```



```

327.             if neutral in substances and cation in substances and anion in
substances and \
328.                 cation in pitzer_data[neutral] and anion in
pitzer_data[neutral][cation]:
329.                     mn = m[substances == neutral][0]
330.                     mc = m[substances == cation][0]
331.                     ma = m[substances == anion][0]
332.                 else:
333.                     continue
334.
335.                 if 'Zeta' in pitzer_data[neutral][cation][anion]:
336.                     parameters =
pitzer_data[neutral][cation][anion]['Zeta']['parameters']
337.                     equation =
pitzer_data[neutral][cation][anion]['Zeta']['Equation']
338.                     a = np.zeros(16)
339.                     a[:len(parameters)] = parameters
340.                     zeta = eval(equation)
341.
342.                 if calculation_type == 'all':
343.                     Zeta = Zeta + mn * mc * ma * zeta
344.
345.                 elif calculation_type in neutrals:
346.                     Zeta = Zeta + mc * ma * zeta
347.
348.                 elif calculation_type in cations:
349.                     Zeta = Zeta + mn * ma * zeta
350.
351.                 elif calculation_type in anions:
352.                     Zeta = Zeta + mn * mc * zeta
353.
354.
355.             return Zeta
356.
357.     def Eta(calculation_type):
358.         Eta = 0
359.
360.         if isinstance(calculation_type, str):
361.             if calculation_type.count('+') > 0:
362.                 calculation_type = [calculation_type, 'cations']
363.             else:
364.                 calculation_type = [calculation_type, 'anions']
365.
366.         if calculation_type[1] == 'cations': total_ions = cations
367.         elif calculation_type[1] == 'anions': total_ions = anions
368.
369.         for neutral in neutrals:
370.             for ion_1 in total_ions:
371.                 for ion_2 in total_ions:
372.                     eta = 0; a = np.zeros(15)
373.
374.                     if calculation_type[0] == 'all' \
375.                       and np.where(total_ions == ion_2) <= np.where(total_ions ==
ion_1):
376.                         continue
377.
378.                     if neutral in substances and ion_1 in substances and ion_2 in
substances and \
379.                       ion_1 in pitzer_data[neutral] and ion_2 in
pitzer_data[neutral][ion_1]:
380.                         mn = m[substances == neutral][0]
381.                         m1 = m[substances == ion_1][0]
382.                         m2 = m[substances == ion_2][0]
383.                     else:
384.                         continue
385.

```

```

386.         if 'Eta' in pitzer_data[neutral][ion_1][ion_2]:
387.             parameters =
    pitzer_data[neutral][ion_1][ion_2]['Eta']['parameters']
388.             equation =
    pitzer_data[neutral][ion_1][ion_2]['Eta']['Equation']
389.             a = np.zeros(16)
390.             a[:len(parameters)] = parameters
391.             eta = eval(equation)
392.
393.             if calculation_type == 'all':
394.                 Eta = Eta + mn * m1 * m2 * eta
395.
396.             elif calculation_type in neutrals:
397.                 Eta = Eta + mn * m1 * m2 * eta
398.
399.             elif calculation_type == ion_1:
400.                 Eta = Eta + mn * m2 * eta
401.
402.         return Eta
403.
404.     def MU(calculation_type):
405.         MU = 0
406.
407.         if isinstance(calculation_type, str):
408.             if calculation_type.count('+') > 0:
409.                 calculation_type = [calculation_type, 'cations']
410.             else:
411.                 calculation_type = [calculation_type, 'anions']
412.
413.         if calculation_type[1] == 'cations': total_ions = cations
414.         elif calculation_type[1] == 'anions': total_ions = anions
415.
416.         for neutral in neutrals:
417.             for ion in total_ions:
418.                 mu = 0; a = np.zeros(15)
419.
420.                 if neutral in substances and ion in substances and \
421.                    neutral in pitzer_data[neutral] and ion in
    pitzer_data[neutral][neutral]:
422.                     mn = m[substances == neutral][0]
423.                     mi = m[substances == ion][0]
424.                 else:
425.                     continue
426.
427.                 if 'MU' in pitzer_data[neutral][neutral][ion]:
428.                     parameters =
    pitzer_data[neutral][neutral][ion]['MU']['parameters']
429.                     equation = pitzer_data[neutral][neutral][ion]['MU']['Equation']
430.                     a = np.zeros(16)
431.                     a[:len(parameters)] = parameters
432.                     mu = eval(equation)
433.
434.                 if calculation_type[0] == 'all':
435.                     MU = MU + 3 * mn * mn * mi * mu
436.
437.                 elif calculation_type[0] in neutrals:
438.                     MU = MU + 6 * mn * mi * mu
439.
440.                 elif calculation_type[0] in total_ions:
441.                     MU = MU + 3 * mn * mn * mu
442.
443.         return MU
444.
445.     def F():
446.         term = 1 + 1.2 * np.sqrt(I)
447.         F = - A() * (np.sqrt(I) / term + 2 / 1.2 * np.log(term))

```

```

448.         F = F + Theta(['derivative', 'cations']) + Theta(['derivative', 'anions']) + \
449.             Bca('derivative') + Cmx('derivative')
450.         return F
451.
452.     def water_activity():
453.         term = 1 + 1.2 * np.sqrt(I)
454.         term = - A() * I**(3/2) / term + Bca('water') + Z * Cmx('water') \
455.             + Theta(['water', 'cations']) + Theta(['water', 'anions']) \
456.             + PSI(['all', 'cations']) + PSI(['all', 'anions']) \
457.             + Lambda(['all', 'cations']) + Lambda(['all', 'anions']) \
458.             + Zeta('all') + Eta(['all', 'cations']) \
459.             + MU(['all', 'cations']) + MU(['all', 'anions'])
460.
461.         osm = 2 * term / np.sum(m) + 1
462.         return - np.sum(m) * osm * 18.015268 / 1000
463.
464.     def ion_activity():
465.         zi = z[substances == species][0]
466.         return zi**2 * F() + Bca(species) + Z * Cmx(species) \
467.             + Theta(species) + np.abs(zi) * Cmx('all') \
468.             + PSI([species, 'cations']) + PSI([species, 'anions']) \
469.             + Lambda(species) + Zeta(species) + Eta(species) + MU(species)
470.
471.     def neutral_activity():
472.         return Lambda([species, 'cations']) + Lambda([species, 'anions']) + \
473.             Zeta(species) + Eta([species, 'cations']) + \
474.             MU([species, 'cations']) + MU([species, 'anions'])
475.
476.     if species == 'H2O(1)': return water_activity()
477.     elif '+' in species or '-' in species: return ion_activity()
478.     else: return neutral_activity()

```

A.3.7 simulation.py

```

1. import json
2. import numpy as np
3. from gekko import GEKKO
4. import matplotlib.pyplot as plt
5. from aqueous_activity import ln_activity
6. from gas_fugacity import peng_robinson
7.
8. ## Molecular Weights
9. MW_WATER = 18.01528      # g/mol
10. MW_Na    = 22.989769    # g/mol
11. MW_K     = 39.0983     # g/mol
12. MW_Mg    = 24.305      # g/mol
13. MW_Ca    = 40.078     # g/mol
14. MW_Cl    = 35.453     # g/mol
15. MW_HCO3  = 61.0168    # g/mol
16. MW_SO4   = 96.06     # g/mol
17.
18. ## kinetic & thermo Constants
19. from rxn_data import K_H2O
20. from abs_data import K1a
21. from rxn_data import K_CO2g, k_CO2aq, K_CO2aq, K_CO3
22. from rxn_data import K_NH3, k_CO2_NH3, k_NH2COO
23. from rxn_data import K_MgOH, K_HSO4
24. from rxn_data import K_NaOH, K_NaHCO3
25. from rxn_data import K_Na2CO3_10H2O, K_Na2SO4_10H2O
26. from rxn_data import K_MgOH2
27. from rxn_data import K_CaOH2, K_CaSO4_2H2O, K_CaCO3
28.
29. ## Operating Conditions
30. T      = 273.15 + 25    # K
31. P      = 1.01325 * 1   # bar

```

```

32.
33. ## Initial Condition
34. NH30 = 1                # variable
35. Na0  = 23200/1000/MW_Na
36. K0   = 808/1000/MW_K
37. Mg0  = 2610/1000/MW_Mg
38. Ca0  = 890/1000/MW_Ca
39. Cl0  = 44000/1000/MW_Cl
40. SO40 = 6090/1000/MW_SO4
41. C0   = 200/1000/MW_HCO3
42.
43.
44. model = GEKKO(remote=True)
45. model.options.MAX_ITER = 5000
46. model.options.IMODE = 4    # Dynamic Simulation
47. model.options.NODES = 5    # collocation nodes
48.
49. ## Model variables & Initial Values
50. Na      = model.Param(value= Na0)
51. K       = model.Param(value= K0)
52. Mg     = model.Param(value= Mg0)
53. Ca     = model.Param(value= Ca0)
54. Cl     = model.Param(value= Cl0)
55. SO4    = model.Param(value= SO40)
56. CO2g   = model.Param(value= P)
57.
58. H       = model.Var(value= 1e-7, lb= 0)
59. CO2aq  = model.Var(value= 0, lb= 0)
60. HCO3   = model.Var(value= C0, lb= 0)
61. CO3    = model.Var(value= 0, lb= 0)
62. NH3aq  = model.Var(value= NH30, lb= 0)
63. NH2COO = model.Var(value= 0, lb= 0)
64. NH4    = model.Var(value= 0, lb= 0)
65. OH     = model.Var(value= 1e-7, lb= 0)
66.
67. solution = {
68.   'H+'      : H,
69.   'Na+'    : Na,
70.   'K+'     : K,
71.   'Mg++'   : Mg,
72.   'Ca++'   : Ca,
73.   'NH4+'   : NH4,
74.   'OH-'    : OH,
75.   'Cl-'    : Cl,
76.   'HCO3-'  : HCO3,
77.   'CO3--'  : CO3,
78.   'SO4--'  : SO4,
79.   'NH2COO-' : NH2COO,
80.   'CO2(aq)': CO2aq,
81.   'NH3(aq)': NH3aq
82. }
83.
84. ## Model Intermediates
85. activities = ln_activity(model, T, P, solution)
86. water_activity = model.Intermediate(model.exp(activities[ 'H2O(l)']))
87. gamma_H        = model.Intermediate(model.exp(activities[ 'H+' ]))
88. gamma_Na       = model.Intermediate(model.exp(activities[ 'Na+' ]))
89. gamma_K        = model.Intermediate(model.exp(activities[ 'K+' ]))
90. gamma_Mg       = model.Intermediate(model.exp(activities[ 'Mg++' ]))
91. gamma_Ca       = model.Intermediate(model.exp(activities[ 'Ca++' ]))
92. gamma_NH4      = model.Intermediate(model.exp(activities[ 'NH4+' ]))
93. gamma_OH       = model.Intermediate(model.exp(activities[ 'OH-' ]))
94. gamma_Cl       = model.Intermediate(model.exp(activities[ 'Cl-' ]))
95. gamma_HCO3     = model.Intermediate(model.exp(activities[ 'HCO3-' ]))
96. gamma_CO3     = model.Intermediate(model.exp(activities[ 'CO3--' ]))
97. gamma_SO4     = model.Intermediate(model.exp(activities[ 'SO4--' ]))

```

```

98. gamma_NH2COO = model.Intermediate(model.exp(activities['NH2COO-']))
99. gamma_CO2aq  = model.Intermediate(model.exp(activities['CO2(aq)']))
100. gamma_NH3aq  = model.Intermediate(model.exp(activities['NH3(aq)']))
101.
102. gamma_CO2g = model.Intermediate(model.exp(peng_robinson(model, T, P)))
103.
104. ## Model equations
105. model.Equation(gamma_H*H * gamma_OH*OH / K_H2O(T) == water_activity)
106.
107. model.Equation(CO2aq.dt() == Kla(T, P)*(gamma_CO2g*K_CO2g(T)*CO2g - gamma_CO2aq*CO2aq)
+ \
108.     -k_CO2aq(T)*(gamma_CO2aq*CO2aq * gamma_OH*OH - gamma_HCO3*HCO3/K_CO2aq(T)) \
109.     -(k_CO2_NH3(T)*gamma_CO2aq*CO2aq*gamma_NH3aq*NH3aq -
k_NH2COO(T)*gamma_H*H*gamma_NH2COO*NH2COO))
110.
111. model.Equation(HCO3.dt() + CO3.dt() == \
112.     k_CO2aq(T)*(gamma_CO2aq*CO2aq * gamma_OH*OH - gamma_HCO3*HCO3/K_CO2aq(T)))
113.
114. model.Equation(gamma_CO3*CO3 * gamma_H*H / K_CO3(T) == gamma_HCO3*HCO3)
115.
116. model.Equation(gamma_NH3aq*NH3aq * gamma_H*H == gamma_NH4*NH4 / K_NH3(T))
117.
118. model.Equation(NH2COO.dt() == k_CO2_NH3(T) * gamma_CO2aq*CO2aq * gamma_NH3aq*NH3aq - \
119.     k_NH2COO(T) * gamma_H*H * gamma_NH2COO*NH2COO )
120.
121. model.Equation(NH3aq + NH2COO + NH4 == NH30)
122. model.Equation(H +C0 +NH30 == HCO3 +2*CO3 +NH3aq + 2*NH2COO +OH)
123.
124. ## Model Solver
125. time = np.logspace(np.log10(1e-6), np.log10(1), 6)
126. time[0] = 0; time[-1] = 0.3
127.
128. time = np.concatenate((time, np.linspace(1, 10, 45+1)[: -1]))
129.
130. time = np.concatenate((time, np.linspace(10, 60, 40+1)[: -1]))
131.
132. time = np.concatenate((time, np.linspace(60, 10*60, 15+1)[: -1]))
133.
134. model.solve(disps=True)

```

A.3.8 optimization.py

```

1. import json
2. import numpy as np
3. from gekko import GEKKO
4. import matplotlib.pyplot as plt
5. from aqueous_activity import ln_activity
6. from gas_fugacity import peng_robinson
7.
8. ## Molecular Weights
9. MW_WATER = 18.01528      # g/mol
10. MW_Na    = 22.989769     # g/mol
11. MW_K     = 39.0983       # g/mol
12. MW_Mg    = 24.305        # g/mol
13. MW_Ca    = 40.078        # g/mol
14. MW_Cl    = 35.453       # g/mol
15. MW_HCO3  = 61.0168      # g/mol
16. MW_SO4   = 96.06        # g/mol
17.
18. ## kinetic & thermo Constants
19. from rxn_data import K_H2O
20. from abs_data import Kla
21. from rxn_data import K_CO2g, k_CO2aq, K_CO2aq, K_CO3
22. from rxn_data import K_NH3, k_CO2_NH3, k_NH2COO
23. from rxn_data import K_MgOH, K_HSO4

```

```

24. from rxn_data import K_NaOH, K_NaHCO3
25. from rxn_data import K_Na2CO3_10H2O, K_Na2SO4_10H2O
26. from rxn_data import K_MgOH2
27. from rxn_data import K_CaOH2, K_CaSO4_2H2O, K_CaCO3
28.
29. ## Initial Condition
30. Na0 = 23200/1000/MW_Na
31. K0 = 808/1000/MW_K
32. Mg0 = 2610/1000/MW_Mg
33. Ca0 = 890/1000/MW_Ca
34. Cl0 = 44000/1000/MW_Cl
35. SO40 = 6090/1000/MW_SO4
36. C0 = 200/1000/MW_HCO3
37.
38. ## GEKKO Model instance
39. model = GEKKO(remote=True)
40.
41. ## Model Variables
42. K = model.Param(value= K0)
43. Mg = model.Param(value= Mg0)
44. Cl = model.Param(value= Cl0)
45. SO4 = model.Param(value= SO40)
46. CO2g = model.Param(value= P)
47.
48. T = model.Var(value= 298.15, lb= 283.15, ub=273.15+40)
49. P = model.Var(value= 1, lb= 1, ub=70)
50.
51. H = model.Var(value= 1e-7, lb= 0)
52. CO2aq = model.Var(value= 0, lb= 0)
53. HCO3 = model.Var(value= C0, lb= 0)
54. CO3 = model.Var(value= 0, lb= 0)
55. OH = model.Var(value= 1e-7, lb= 0)
56. Na = model.Var(value= Na0, lb= 0)
57. Ca = model.Var(value= Ca0, lb= 0)
58. NH3aq = model.Var(value= 0, lb= 0)
59. NH4 = model.Var(value= 0, lb= 0)
60. NH2COO = model.Var(value= 0, lb= 0)
61.
62. NH3_base = model.Var(value= 1, lb= 0, ub=10)
63.
64. NaHCO3 = model.Var(value= 0, lb= 0)
65. CaCO3 = model.Var(value= 0, lb= 0)
66.
67. solution = {
68.     'H+' : H,
69.     'Na+' : Na,
70.     'K+' : K,
71.     'Mg++' : Mg,
72.     'Ca++' : Ca,
73.     'OH-' : OH,
74.     'Cl-' : Cl,
75.     'HCO3-' : HCO3,
76.     'CO3--' : CO3,
77.     'SO4--' : SO4,
78.     'CO2(aq)' : CO2aq,
79.     'NH3(aq)' : NH3aq,
80.     'NH4+' : NH4,
81.     'NH2COO-' : NH2COO
82. }
83.
84. ## Model Intermediates
85. activities = ln_activity(model, T, P, solution)
86. water_activity = model.Intermediate(model.exp(activities[ 'H2O(l)']))
87. gamma_H = model.Intermediate(model.exp(activities[ 'H+' ]))
88. gamma_Na = model.Intermediate(model.exp(activities[ 'Na+' ]))
89. gamma_K = model.Intermediate(model.exp(activities[ 'K+' ]))

```

```

90. gamma_Mg      = model.Intermediate(model.exp(activities[ 'Mg++']))
91. gamma_Ca      = model.Intermediate(model.exp(activities[ 'Ca++']))
92. gamma_OH      = model.Intermediate(model.exp(activities[ 'OH-']))
93. gamma_Cl      = model.Intermediate(model.exp(activities[ 'Cl-']))
94. gamma_HCO3    = model.Intermediate(model.exp(activities[ 'HCO3-']))
95. gamma_CO3     = model.Intermediate(model.exp(activities[ 'CO3--']))
96. gamma_SO4     = model.Intermediate(model.exp(activities[ 'SO4--']))
97. gamma_CO2aq   = model.Intermediate(model.exp(activities[ 'CO2(aq)']))
98. gamma_NH3aq   = model.Intermediate(model.exp(activities[ 'NH3(aq)']))
99. gamma_NH4     = model.Intermediate(model.exp(activities[ 'NH4+']))
100. gamma_NH2COO = model.Intermediate(model.exp(activities[ 'NH2COO-']))
101.
102. gamma_CO2g = model.Intermediate(model.exp(peng_robinson(model, T, P)))
103.
104. ## Model equations
105. model.Equation(gamma_H*H * gamma_OH*OH / K_H2O(T) == water_activity)
106.
107. model.Equation(K1a(T, P)*(gamma_CO2g*K_CO2g(T)*CO2g - gamma_CO2aq*CO2aq) == \
108.     k_CO2aq(T)*(gamma_CO2aq*CO2aq * gamma_OH*OH - gamma_HCO3*HCO3/K_CO2aq(T)) + \
109.     (k_CO2_NH3(T)*gamma_CO2aq*CO2aq*gamma_NH3aq*NH3aq -
110.     k_NH2COO(T)*gamma_H*H*gamma_NH2COO*NH2COO))
111. model.Equation(gamma_CO2aq*CO2aq * gamma_OH*OH * K_CO2aq(T) == gamma_HCO3*HCO3)
112. model.Equation(gamma_CO3*CO3 * gamma_H*H / K_CO3(T) == gamma_HCO3*HCO3)
113.
114. model.Equation(gamma_NH3aq*NH3aq * gamma_H*H / (gamma_NH4*NH4) == 1/K_NH3(T))
115. model.Equation(k_CO2_NH3(T) * gamma_CO2aq*CO2aq * gamma_NH3aq*NH3aq == \
116.     k_NH2COO(T) * gamma_H*H * gamma_NH2COO*NH2COO )
117.
118. model.Equation(gamma_Na*Na * gamma_HCO3*HCO3 / K_NaHCO3(T) == 1)
119. model.Equation(gamma_Ca*Ca * gamma_CO3*CO3 / K_CaCO3(T) == 1)
120.
121. model.Equation(NH3aq +NH4 +NH2COO == NH3_base)
122. model.Equation(H +C0 +NH3_base == HCO3 +2*CO3 +OH +NaHCO3 +2*CaCO3 +NH3aq +2*NH2COO)
123. model.Equation(Na + NaHCO3 == Na0)
124. model.Equation(Ca + CaCO3 == Ca0)
125.
126.
127. ## Research Objective
128. model.Minimize(Na)
129.
130.
131.
132. ## Model Solver
133. model.options.SOLVER = 3 #IPOpt
134. model.options.IMODE = 3 #Optimization
135. model.options.MAX_ITER = 1000
136. model.solve(disp=True)

```

A.3.9 CO₂_hydrate.py

```

1. ## data-holders Libraries
2. import numpy as np
3. import pandas as pd
4.
5. ## Data preparation libraries
6. from sklearn.decomposition import PCA
7. from sklearn.model_selection import train_test_split
8.
9. ## Data Analysis libraries
10. from sklearn.model_selection import GridSearchCV
11. from sklearn.inspection import permutation_importance
12. from sklearn.metrics import r2_score
13. from sklearn.metrics import mean_absolute_error
14.

```

```

15. ## Plotting Libraries
16. from matplotlib import pyplot as plt
17.
18. ## ML Regression Libraries
19. from sklearn import linear_model
20. from sklearn.neural_network import MLPRegressor
21. from sklearn.tree import DecisionTreeRegressor
22. from sklearn.ensemble import RandomForestRegressor
23. from sklearn.neighbors import KNeighborsRegressor
24. from sklearn.svm import SVR
25.
26.
27.
28. def expand_data(input_data):
29.     feature_names = input_data.columns
30.
31.     for feature in feature_names:
32.         input_data[feature+'^2'] = (input_data[feature]/1e4)**2
33.         input_data[feature+'^3'] = (input_data[feature]/1e4)**3
34.         input_data['ln('+feature+')'] = np.log(input_data[feature]).replace(-np.inf, 0)
35.         input_data['sqrt('+feature+')'] = np.sqrt(input_data[feature])
36.         input_data['1/'+feature] = (1/input_data[feature]).replace(np.inf, 0)
37.         input_data[feature] = input_data[feature]/1e4
38.
39.     return input_data
40.
41.
42. ## Data Import & Cleaning
43. data = pd.read_csv("raw_data.csv")
44.
45. data = data[data['Cl-'] != 0]
46. data = data[data['Cl-_rem'] != 0]
47.
48. data['Cl-_rmv'] = data['Cl-_rem']
49. data['Cl-_rmn'] = data['Cl-'] - data['Cl-_rmv']
50. data['Cl-_pct'] = data['Cl-_rmv'] / data['Cl-']
51. del data['Cl-_rem']
52.
53.
54. ## Data Analysis on Cation data
55. cation_data = data[['Na+', 'K+', 'Mg++', 'Ca++']].copy()
56. cation_data = expand_data(cation_data)
57.
58. pos_pca = PCA(n_components=5)
59. cation_pca = pos_pca.fit_transform(cation_data)
60.
61. plt.figure(figsize=(5*3,5*5))
62. for i in range(5):
63.     plt.subplot(5,3,i*3+1)
64.     if i==0: plt.title('Cl- removed')
65.     plt.ylabel('PC '+str(i+1))
66.     plt.scatter(cation_pca[:,i], data['Cl-_rmv'])
67.
68.     plt.subplot(5,3,i*3+2)
69.     if i==0: plt.title('Cl- remaining')
70.     plt.scatter(cation_pca[:,i], data['Cl-_rmn'])
71.
72.     plt.subplot(5,3,i*3+3)
73.     if i==0: plt.title('Cl- percentage')
74.     plt.scatter(cation_pca[:,i], data['Cl-_pct'])
75.
76.
77. ## Data Analysis on Anion data
78. anion_data = data[['Cl-', 'SO4--']].copy()
79. anion_data = expand_data(anion_data)
80.

```



```

81. neg_pca = PCA(n_components=5, svd_solver='full')
82. anion_pca = neg_pca.fit_transform(anion_data)
83.
84. plt.figure(figsize=(5*3,5*5))
85. for i in range(5):
86.     plt.subplot(5,3,i*3+1)
87.     if i==0: plt.title('Cl- removed')
88.     plt.ylabel('PC '+str(i+1+5))
89.     plt.scatter(anion_pca[:,i], data['Cl-_rmv'])
90.
91.     plt.subplot(5,3,i*3+2)
92.     if i==0: plt.title('Cl- remaining')
93.     plt.scatter(anion_pca[:,i], data['Cl-_rmn'])
94.
95.     plt.subplot(5,3,i*3+3)
96.     if i==0: plt.title('Cl- percentage')
97.     plt.scatter(anion_pca[:,i], data['Cl-_pct'])
98.
99.
100. ## Data Analysis on Operating Conditions
101. op_cond_data = data[['T','P']].copy()
102. op_cond_data = expand_data(op_cond_data)
103.
104. pca = PCA(n_components=5, svd_solver='full')
105. op_cond_pca = pca.fit_transform(op_cond_data)
106.
107. plt.figure(figsize=(5*3,5*5))
108. for i in range(5):
109.     plt.subplot(5,3,i*3+1)
110.     if i==0: plt.title('Cl- removed')
111.     plt.ylabel('PC '+str(i+1+10))
112.     plt.scatter(op_cond_pca[:,i], data['Cl-_rmv'])
113.
114.     plt.subplot(5,3,i*3+2)
115.     if i==0: plt.title('Cl- remaining')
116.     plt.scatter(op_cond_pca[:,i], data['Cl-_rmn'])
117.
118.     plt.subplot(5,3,i*3+3)
119.     if i==0: plt.title('Cl- percentage')
120.     plt.scatter(op_cond_pca[:,i], data['Cl-_pct'])
121.
122.
123. ## Prepare the data
124. modified_data = {'feature_1': np.log(cation_pca[:,0]+80),
125.                 'feature_2': anion_pca[:,0],
126.                 'feature_3': np.log(anion_pca[:,1]+35),
127.                 'Cl-_rmv' : data['Cl-_rmv']}
128.
129. modified_data = pd.DataFrame(modified_data)
130.
131. training_set, testing_set = \
132.     train_test_split(modified_data, test_size=0.3, shuffle=True, random_state=42)
133.
134. feature_list = ['feature_1','feature_2','feature_3']
135. x_data = modified_data[feature_list]
136. y_data = modified_data['Cl-_rmv']
137.
138. training_input_data = training_set[feature_list]
139. training_output_data = training_set['Cl-_rmv']
140.
141. testing_input_data = testing_set[feature_list]
142. testing_output_data = testing_set['Cl-_rmv']
143.
144.
145. ## Machine Learning Regression
146. lin_model = linear_model.LinearRegression()

```

```

147. lin_model.fit(training_input_data, training_output_data)
148. print(r2_score(testing_output_data, lin_model.predict(testing_input_data)))
149. imp = lin_model.coef_
150. plt.bar(x=x_data.columns,height=imp)
151.
152.
153. MLP_model = MLPRegressor(hidden_layer_sizes=(5,5), max_iter=int(1e5), random_state=42)
154. MLP_model.fit(training_input_data, training_output_data)
155. print(r2_score(testing_output_data, MLP_model.predict(testing_input_data)))
156. imp = permutation_importance(MLP_model, training_input_data,
157.     training_output_data, scoring='r2').importances_mean
158. plt.bar(x=x_data.columns,height=imp)
159.
160.
161. tree_model = DecisionTreeRegressor(random_state=42)
162. tree_model.fit(training_input_data, training_output_data)
163. print(r2_score(testing_output_data, tree_model.predict(testing_input_data)))
164. dt_imp = tree_model.feature_importances_
165. plt.barh(y=x_data.columns,width=dt_imp)
166.
167.
168. forest_model = RandomForestRegressor(random_state=0)
169. forest_model.fit(training_input_data, training_output_data)
170. print(r2_score(testing_output_data, forest_model.predict(testing_input_data)))
171. rf_imp = forest_model.feature_importances_
172. plt.barh(y=x_data.columns,width=rf_imp)
173.
174.
175. from xgboost import XGBRegressor
176. xgboost_model = XGBRegressor()
177. xgboost_model.fit(training_input_data, training_output_data)
178. print(r2_score(testing_output_data, xgboost_model.predict(testing_input_data)))
179. xgb_imp = xgboost_model.feature_importances_
180. plt.barh(y=x_data.columns,width=xgb_imp)
181.
182.
183. parameters = {
184.     'kernel':['linear', 'rbf','poly'],
185.     'C':[1e-2, 1e-1, 1e0, 1e1, 15],
186.     'epsilon': [1e-2, 0.05, 1e-1, 0.2, 1]
187. }
188.
189. SVR_opt = GridSearchCV(SVR(), parameters)
190. SVR_opt.fit(training_input_data, training_output_data)
191. print(SVR_opt.best_params_)
192. print(SVR_opt.score(testing_input_data, testing_output_data))
193. perm_importance = permutation_importance(SVR_opt, testing_input_data,
194.     testing_output_data)
195. imp=perm_importance.importances_mean
196. plt.bar(x=x_data.columns,height=imp)
197. plt.show()
198.
199. KNeighbors_model = KNeighborsRegressor()
200. KNeighbors_model.fit(training_input_data, training_output_data)
201. print(r2_score(testing_output_data, KNeighbors_model.predict(testing_input_data)))
202. imp = permutation_importance(KNeighbors_model, training_input_data,
203.     training_output_data,
204.     scoring='r2').importances_mean
205. plt.bar(x=x_data.columns,height=imp)
206.
207.
208. ## ML model Comparison using Second scoring metric
209. print(mean_absolute_error(training_output_data,
210.     tree_model.predict(training_input_data)))
210. plt.plot(np.linspace(0,5e4),np.linspace(0,5e4))

```

```

211. plt.xlabel('Actual Chloride Removal', fontsize=13)
212. plt.ylabel('Predicted response', fontsize=13)
213. plt.scatter(39679.429+100, 48794.9735-60, s=200, facecolors='none', edgecolors='r')
214. plt.scatter(testing_output_data, tree_model.predict(testing_input_data), c='orange')
215. plt.show()
216.
217. print(mean_absolute_error(training_output_data,
    forest_model.predict(training_input_data)))
218. plt.plot(np.linspace(0,5e4),np.linspace(0,5e4))
219. plt.xlabel('Actual Chloride Removal', fontsize=13)
220. plt.ylabel('Predicted response', fontsize=13)
221. plt.scatter(testing_output_data, forest_model.predict(testing_input_data), c='orange')
222. plt.show()
223.
224. print(mean_absolute_error(training_output_data,
    xgboost_model.predict(training_input_data)))
225. plt.plot(np.linspace(0,5e4),np.linspace(0,5e4))
226. plt.xlabel('Actual Chloride Removal', fontsize=13)
227. plt.ylabel('Predicted response', fontsize=13)
228. plt.scatter(testing_output_data, xgboost_model.predict(testing_input_data),
    c='orange')
229. plt.show()
230.
231.
232. ## Model Inference
233. initial_conc = pd.DataFrame({'Na+':[23200*(1-
    0.725)], 'K+':[808], 'Mg++':[2610], 'Ca++':[890], 'Cl-':[44000], 'SO4--':[6090]})
234.
235. pos_initial_conc = expand_data(initial_conc[['Na+', 'K+', 'Mg++', 'Ca++']])
236. feature_1 = np.log(pos_pca.transform(pos_initial_conc)[: ,0]+80)[0]
237.
238. neg_initial_conc = expand_data(initial_conc[['Cl-', 'SO4--']])
239. feature_6 = neg_pca.transform(neg_initial_conc)[: ,0][0]
240. feature_7 = np.log(neg_pca.transform(neg_initial_conc)[: ,1]+35)[0]
241.
242.
243. print(tree_model.predict([[feature_1, feature_2, feature_3]]))
244. print(xgboost_model.predict(pd.DataFrame({'feature_1':[feature_1],
    'feature_2':[feature_2], 'feature_3':[feature_3]})))

```

UAEU

جامعة الإمارات العربية المتحدة
United Arab Emirates University



UAE UNIVERSITY MASTER THESIS NO. 2022: 86

This work explores how desalination waste (brine) can be revitalized for salt removal and water recoverability. Sodium ions were removed by bubbling CO₂ in basic brine media. Chloride ions were removed by CO₂ hydrate formation. The final brackish water can be recycled within the system to achieve Zero-Liquid Discharge.

Ahmed Elsayed received his Master of Science in Chemical Engineering from the Department of Chemical and Petroleum Engineering, College of Engineering at UAE University, UAE. He received his B.Sc. in Chemical Engineering from the College of Engineering, Texas A&M University, USA.

www.uaeu.ac.ae

Online publication of thesis:
<https://scholarworks.uaeu.ac.ae/etds/>

UAEU عمادة المكتبات
Libraries Deanship

جامعة الإمارات العربية المتحدة
United Arab Emirates University



قسم الخدمات المكتبية الرقمية - Digital Library Services Section