Characterizing Biosorption of Lanthanides to *Cupriavidus necator*

Presented in Partial Fulfillment of the Requirements for the

Degree of Master of Science

with a

Major in Chemical Engineering

in the

College of Graduate Studies

University of Idaho

by

Bennett Ashton Charles Carv

Major Professor: James G. Moberly, Ph.D.

Committee Members: D. Eric Aston, Ph.D.; Soumya K. Srivastava, Ph.D.

Department Administrator: D. Eric Aston, Ph.D.

August 2017

Authorization to Submit Thesis

This thesis of Bennett Ashton Charles Carv, submitted for the degree of Master of Science with a Major in Chemical Engineering and titled "Characterizing Biosorption of Lanthanides to *Cupriavidus necator*," has been reviewed in final form. Permission, as indicated by the signatures and dates below, is now granted to submit final copies to the College of Graduate Studies for approval.

Major Professor: _____ Date: _____

James G. Moberly, Ph.D.

Committee Members: _____ Date: _____

D. Eric Aston, Ph.D.

_____ Date: _____

Soumya K. Srivastava, Ph.D.

Department

Administrator: _____ Date: _____

D. Eric Aston, Ph.D.

Abstract

Biosorption has the potential to become an inexpensive separation process for rare earth elements (REEs) at trace concentrations. Organisms which possess high specificity for REEs need identification but screening for these organisms is a slow process. This study focused on characterizing sorption of REEs to *Cupriavidus necator* as a model organism toward evaluating DEP as a platform for rapidly screening organisms that hyperaccumulate metals (OHMs). *C. necator* accumulated europium, neodymium, and samarium at 11.34, 15.98, and 57.82 billion atoms per cell, respectively. Increased biosorption for REEs was observed at reduced pH and increased temperatures. No selectivity differences were observed between REEs, though increased uptake was observed by samarium in presence of europium and neodymium. Optimum operating conditions for a DEP device would occur at lower pH values and higher temperatures for REE retention in biosorbent. Overall, *C.* necator is suitable as a model organism due to its history of study and its ability to achieve accumulation of REEs as determined within this thesis within the tested pH range of 4.5 to 6.5, as well as, the tested temperature range of 15°C to 75°C.

Acknowledgements

I would like to give special thanks to my major professor, Dr. James G. Moberly, for being the greatest and most patient advisor. Without his kindness and support, this work would not have been possible. His dedication and work ethic were great inspirations to me throughout my time working with him.

I would like to thank my committee members, Dr. D. Eric Aston and Dr. Soumya K. Srivastava, thanks for being on my committee. In addition, I would like to thank Dr. Jeff Langman, Dr. Erik Coats and Cynthia Brinkman for use of their research equipment critical to this project. Also, I would like to thank Dr. Dan Strawn for access to his MP-AES.

I would like to thank the rest of the faculty and staff in the Department of Chemical and Materials Engineering for the excellent education and supportive environment they provided. In addition, I would like to thank the two undergraduates who helped in the data collection for this research: Jonathan Counts and Amanda Vu. Finally, I would like to thank those who I may have forgotten to specify.

Dedication

To my family.

Table of Contents

Nomenclature

A_a	Arrhenius constant for association	(ml^2/(µmols·second))
A_d	Arrhenius constant for dissociation	(ml/second)
b	Langmuir affinity constant	(unitless)
C_f	Final Concentration	(µmols/ml)
C_i	Initial Concentration	(µmols/ml)
C_{MS}	Bound site concentration	(µmols/billion cells)
C_s	Unbound site concentration	(µmols/billion cells)
C_{Smax}	Maximum binding site concentration	(µmols/billion cells)
E_a	Activation energy of association	(J/mol)
E_d	Activation energy of dissociation	(J/mol)
H	Isosteric Heat	(J)
i	Metal ion index	(unitless)
k_a	adsorption rate constant	(ml^2/(µmols·second))
k_d	desorption rate constant	(ml/second)
K_f	Freundlich equilibrium constant	(unitless)
K_s	Sips equilibrium constant	(unitless)
n_f	Freundlich heterogeneity constant	(unitless)
n_s	Sips heterogeneity constant	(unitless)
Q	Heat of adsorption	(J)
q	Sorbate capacity	(µmols/billion cells)
R	Universal gas constant	(J/(mols·K))

r_a	adsorption reaction rate	(µmols/second)
r_d	desorption reaction rate	(µmols/second)
T	Temperature	(K)
T_0	Reference temperature	(K)
θ	Fractional binding site coverage	(unitless)
V	Aqueous solution volume	(ml)
α	Sips temperature dependence fitted parameter	(unitless)
χ	Sips temperature dependence fitted parameter	(unitless)

List of Figures

List of Tables

Introduction

Rare earth elements (REEs) are in high demand in modern society. These elements have many applications to which they are desirable for the unique properties they possess. Figure 1 describes the various usages within the United Sates and percentage of REE market each industry uses[2]. Within metallurgy, these elements are mixed with other metals and non-metals to increase mechanical properties. The magnetocaloric effects these elements have are used in applications such as extreme temperature storage tanks and vehicle O_2 sensors. Many of the hard disk drives, electric motors and even headphones take advantage of rare earth elements magnetic properties. Their luminescent properties are currently being utilized in modern medicinal imaging and consumer electronics.

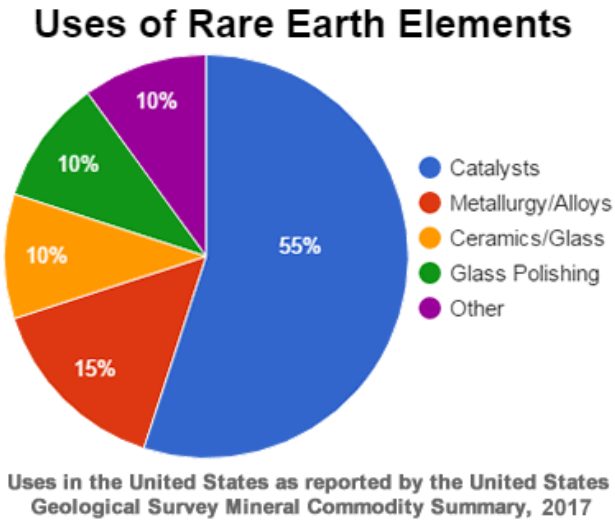


Figure 1: Division of how rare earth elements are used in U.S [2]

Ironically, while rare earth elements are desirable for "green" technologies, mining for these rare earth elements is one of the most destructive mining processes to the environment that is still being used. For every ton of REE mined, 2,000 tons of mine tailings are produced with some deposits containing radioactive thorium. In addition, current recycling rate for these elements is less than 10% with a supply risk index rating of 9.5 out of 10 [6-8]. China, Russia, and Malaysia are the top 3 producing countries today for REEs. The top 3 reserve holders are China, the Commonwealth Independent States (CIS) and USA [9]. The mineral ores bastnaesite, monazite contain 95% of all of Earth's REE supply [2].

Even though the lanthanides are referred to as “rare”, abundance of even the least abundant rare earth element is similar to the abundance of other industrial metals. Figure 2 displays the rare earth elements in abundance to other elements [4]. From this figure, it is seen that REEs are still more abundant than many elements that are utilized more frequently. An example of this is cadmium, which is 100 to 1000 times less abundant than a majority of the rare earth elements, yet it is a major component in NiCad batteries. Tin, another common alloying metal, is approximately as abundant as many REEs and has many industrial and consumer uses.

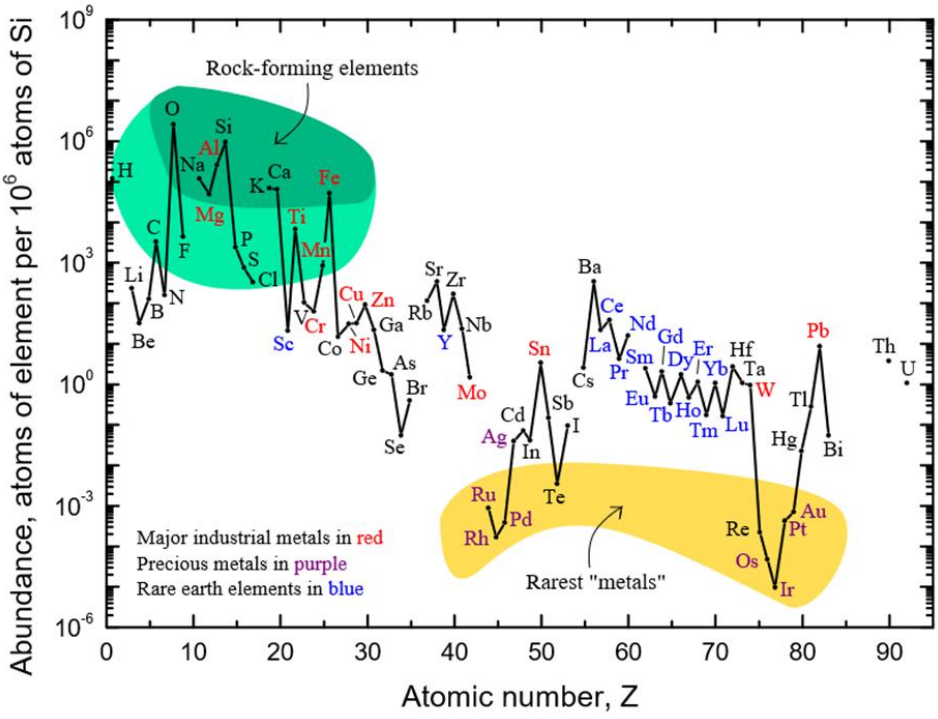


Figure 2: Relative abundance of various elements on earth [4].

As such, it is not the abundance of REEs that is the concern, but rather the concentrations in which they occur. The concentration rates that REEs are typically obtained in is only 0.3 ppm. In comparison, each Prius battery requires up to 15 kg of the rare earth element lanthanum[4]. For a full hybrid to be built requires up to 30 kg of REEs. The extent of REE usage in hybrids is depicted in Figure 3[5]. As a result, a “green” and more efficient mining and separation method is needed to achieve the quantities of REEs to meet the current increasing demand.

Figure 3: Extent of REE usage in hybrid vehicles [5].

The challenge in mining for REEs is the dilute concentrations in which they naturally occur. Pre-concentration of the REEs is needed to achieve the quantities required for separation processes. Biosorption is a means by which pre-concentration may be achieved without as significant an environmental impact. Effectiveness of biosorption, however, depends on proper biosorbent selection due to selectivity differences within these materials. The current screening process for biosorbents is slow and therefore needs innovations to decrease costs and increase throughput. While the overall goal is to improve the current screening process for biosorbents, first characterizing a model system or microorganism which sorbs REEs is needed to validate future high throughput methods. High throughput screening for rapid separation and later identification of organisms which accumulate metals will to help increase the utility of biosorption as a separation process. Rapid screening can potentially reduce the costs and time associated with research to analyze biosorption capabilities of individual biomass types and reserve slower, more expensive testing for identified high performing biomass. Additionally, high throughput techniques could be used as a separation technique of biomass exposed to metals to increase separation efficiency by recycling biomass with low accumulation back into the processes and retaining high metals content biomass for further processing.

The bulk of this thesis is focused on the development of a model organism for which high throughput screening technologies can use to compare biosorption of REEs. Considerations throughout this work are based on the premise that the primary high throughput screening technology to be utilized would be dielectrophoresis, however many of the same considerations apply to other high throughput screening or separation devices. The model organism investigated in this work is the bacterium *Cupriavidus necator*. Biosorption characterization, including characterization under differing temperature, pH, and metal and multi-metal concentrations, were performed to develop this organism as a model bacterium for use in evaluating future high throughput screening technologies. The following chapters are organized to develop *C. necator* as a model organism for REEs in mind. Chapter 1 provides background on chemical behavior, coordination, and current (and potential) separation techniques used for REE separation. Chapter 2 outlines biosorption as a separation process for metals with development of the fundamental processes and considerations which impact biosorption. Chapter 2 also included mathematical models commonly used to characterize the biosorption processes for both engineering design and comparison between biosorbents. As biosorption evaluation is currently a slow process, Chapter 3 discusses potential high throughput processes with focus on dielectrophoresis (currently in development stages at UI). Chapter 4 details efforts performed in this thesis to quantify REE concentrations and discriminate between mixtures of REEs through colorimetric and fluorescence methods. Chapter 5 presents the results of biosorption testing of REEs, specifically Eu, Nd, and Sm, on *C. necator* biomass and its development as a model organism with Chapter 6 outlining future work to be done in this area.

1. Chemical Behavior and Separation of Rare Earth Elements

Increases in the utilization of rare earth elements in current technology has led to a growing demand for these elements. The challenge in meeting the demand is not a matter of abundance, but rather a matter of cost and environmental impact to obtain quantity of rare earths needed. As a result, understanding rare earths and their chemical behaviors will allow for reduction in costs and lower damages to environment caused by their procurement and improve recycling efforts.

1.1 Background of REEs

Initial discovery of the rare earth elements occurred in 1788 by a miner in Ytterby, Sweden when he discovered a strange black rock, later found to be a mineral consisting of cerium, lanthanum, and yttrium in iron ore [10]. These elements were labeled "rare earths" because they were initially believed to be scarce since they had never been found elsewhere. Following the invention of the spectroscope in 1859, as well as the development of the periodic table in 1869, 15 of the rare earth elements were identified [10]. The first application of rare earth elements was in 1880s with the invention of gas mantles and later the flint stone (misch metal) [10]. For a period, during the nuclear weapons race, uranium and thorium were considered rare earth elements due to close affiliation and geological occurrences with the lanthanide group that currently make up the rare earth elements. German scientists in 1939 discovered neutron-induced nuclear fission of uranium and identified rare earth elements existing in fission products. After a nuclear weapons test in 1964, the Cascade Theory of Countercurrent Extraction was discovered [10]. This method of separation allowed for reduced processing costs allowing for mass production of consumer electronics, taking advantage of rare earth element properties. Discovery of the ability to use rare earths in petroleum cracking as catalysts increased US consumption from 2000 to 10,000 tons annually [10].

1.2 General Chemistry

The group of elements termed "rare earth elements" is a group of seventeen elements which includes the fifteen lanthanide group elements plus scandium and yttrium. In nature, these elements usually occur together and are very difficult to separate. REEs are typically

divided into two main groups: light REEs and heavy REEs. Light REEs include the REEs with atomic numbers of 57 to 62 and have no paired electrons in outer shell. Heavy REEs include the REEs with atomic numbers 63 to 71 and Yttrium. There are two types of electronic configurations, as seen in Table 2, that REEs usually adapt: $[Xe]4f^{n}6s^{2}$ and $[Xe]4f^{n-1}5d^{1}6s^{2}$ [11]. These configurations follow the lowest energy principle with "n" representing the number of electrons in the specified orbital ranging from 1 to 14. While scandium and yttrium do not contain the 4f electron configuration but instead have the outermost electron configuration of $(n-1)d^{1}ns^{2}$, as seen in Table 1. This configuration provides scandium and yttrium with similar chemical properties of the lanthanides allowing them to be considered with the lanthanides as rare earth elements. The rare earths all typically exhibit a 3+ valence state with some exceptions. Lanthanum is the most reactive of the group with gradual decreases as move along periodic table toward lutetium and scandium. This group reacts with acids and releases hydrogen when reacting with water. No reaction occurs with bases.

Table 1: Properties of Rare Earth Elements

Element	*Symbol*	*Oxidation states*	*Electron shell configuration*	*Pauling electro-negativity‡*
Lanthanum	La	+3	[Xe]6s^2**5d^1**	1.1
Cerium	Ce	+4, +3	[Xe]6s^2**5d^{1}4f^1**	1.12
Praseodymium	Pr	+3	[Xe]6s^{2}4f^3	1.13
Neodymium	Nd	+3	[Xe]6s^{2}4f^4	1.14
Promethium	Pm	+3	[Xe]6s^{2}4f^5	-
Samarium	Sm	+3	[Xe]6s^{2}4f^6	1.17
Europium	Eu	+3, +2	[Xe]6s^{2}4f^7	-
Gadolinium	Gd	+3	[Xe]6s^{2}4f^7**5d^1**	1.2
Terbium	Tb	+3	[Xe]6s^{2}4f^9	-
Dysprosium	Dy	+3	[Xe]6s^{2}4f^{10}	1.22
Holmium	Ho	+3	[Xe]6s^{2}4f^{11}	1.23
Erbium	Er	+3	[Xe]6s^{2}4f^{12}	1.24
Thulium	Tm	+3	[Xe]6s^{2}4f^{13}	1.25
Ytterbium	Yb	+3, +2	[Xe]6s^{2}4f^{14}	-
Lutetium	Lu	+3	[Xe]6s^{2}4f^{14}5d^1	1.27
Scandium	Sc	+3	[Ar]4s^{2}3d^1	1.36
Yttrium	Y	+3	[Kr]5s^{2}4d^1	1.22

bold-electron shells filled in violation of Madelung's rule of electron configuration

‡ from Shannon [12]

1.2.1 Lanthanide Contraction

Most of the lanthanides exhibit a phenomenon called “lanthanide contraction” with cerium, europium, and ytterbium being the exceptions. This contraction occurs because electrons are added to the inner 4f electron shell rather than to the outermost shell. Since the shielding effect of 4f electron shell is less than the other electron shells with only a partial shielding effect, the effective attraction between the nucleus and the outer electrons increases. This contraction gives the lanthanides similar radii as the third row of the d-block elements (zirconium, molybdenum, tungsten, etc.) resulting in difficult separation from natural minerals. Lanthanide contraction is at the root of the increasing stability constant of lanthanide complexes, as well as, the decrease in alkalinity of lanthanide ions and decrease in ionic radius as atomic number increases.

1.2.2 Coordination and Complexation

Rare earth elements complexes range in coordination numbers between 3 and 12 with 8 being the most common [11]. An example of the coordination chemistry of lanthanides is displayed in Figure 4. The most common coordination number is related to the sum the outer shell orbitals (6s, 6p, and 5d). Ionic radius size is also predicted to be a contributing factor in the high coordination numbers rare earth elements can achieve. REEs possess hard Lewis acid behavior and prefer to bond with hard Lewis base donors such as fluorine, oxygen, and nitrogen. Since rare earths prefer higher coordinating numbers, ligands consisting of oxygen and nitrogen atoms are the best suited for coordination rather than sulfur and phosphorus[11]. In biological systems, interactions with amino acids and carboxylic acids have been observed with a variety of complexes potentially forming [13]. Complexes that are capable of forming are pH dependent with the REEs hydrolyzing and forming precipitates around the pH of 6-7.

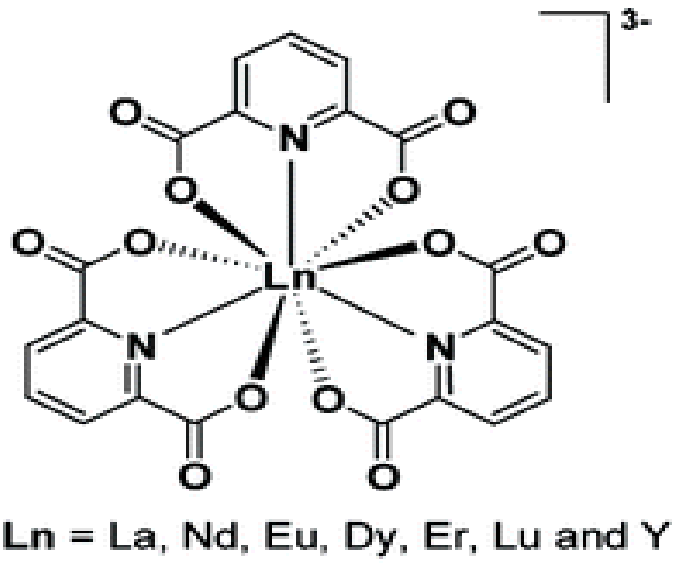


Figure 4: Example coordination chemistry of lanthanides [1]

1.3 Individual REEs Studied

1.3.1 Neodymium

Discovered in 1885, neodymium is part of the lanthanide group in the periodic table with atomic number 60. The element is moderately toxic and can cause irritation to the eyes. Neodymium does not have any known biological roles [7]. It's most important use is as an alloy in the making of strong permanent magnets used in the miniaturization of electronics such as cellular phones, microphones, car windshield wipers, and wind turbines. It is a major component in didymium glass, as well as, a glass coloring agent. Laser technology also uses neodymium as Nd: YAG lasers with applications in construction, military rangefinders, eye surgery, and cutting/welding steel [14]. Many polymerization reactions use neodymium as a catalyst [7].

1.3.2 Samarium

Discovered in 1879, samarium is part of the lanthanide group in the periodic table with atomic number 62. This element has low toxicity and no known biological roles. Samarium shares many applications with europium and neodymium. Its magnets, like neodymium magnets, are used in the miniaturization of electronics, however, not to the same extent. Samarium magnets are more frequently used for microwave applications because of its ability to remain magnetic at high temperatures [8]. Like europium, samarium also is used in control rods in nuclear reactors due to its ability to absorb neutrons [8]. Due to samarium's ability to absorb infrared, it is a component in various glass applications [8].

1.3.3 Europium

Discovered in 1901, europium is part of the lanthanide group in the periodic table with atomic number 63. This element has low toxicity and has no known biological roles. In a reducing environment, europium assumes a divalent state [6]. Currently, this element is employed in the printing of Euro bills as an anti-forgery technique due to its red glow under UV light [6]. In addition, it is used in control rods in nuclear reactors due to its ability to absorb neutrons. Some metal alloys are made with Europium to create superconducting alloys [6].

1.4 Separation of REEs (Current State of Art)

Two main challenges exist in procurement of rare earth elements for industrial and commercial use: 1) the relatively low concentrations of natural occurrence and 2) the low separation factor between the adjacent rare earths. The low concentrations from mines are enriched in what is termed in industry as the primary separation. This process results in REE content high enough to proceed to the secondary separation process where the REEs are separated from each other. Both separation processes results in large amount of waste which is of high concern. The following sections describe in more detail the primary and secondary separation processes.

1.4.1 Primary Separation (Preconcentration)

To obtain rare earth elements, four main ores are typically mined for: bastnaesite, monazite, laterite clays, and loparite. Within these unprocessed ores retrieved from mines, the REE content is typically less than 10%. A separation process, referred to as milling or beneficiation is required to achieve concentrations of above 60% to effectively separate out rare earths. Initially, in the beneficiation processing of bastnaesite, mined ore is ground to fine particles and then a series of conditioning steps using steam, soda ash fluorosilicate, and Tail Oil C-30 is used in tandem with flotation processes to remove gangue (commercially worthless material that surrounds, or is closely mixed with, a wanted mineral in an ore deposit) from REEs. Excess water is then removed by heating and the pH is lowered to under 5 to dissolve the REEs with 10% HCl or H_2SO_4. A roasting stage is used to further concentrate the REE solution and remove excess water. The resulting stream is sent through a thickener to concentrate solids followed by filters to remove smaller undissolved particles remaining in the liquid phase. After this process, REE concentration is high enough to proceed with extraction and separation techniques.

After REEs are obtained in a concentrated supply, the extraction and separation method of rare earth elements revolves around two main strategies: ion exchange and solvent extraction. Ion exchange was used up until the 1960s when demand significantly went up for REEs. This method only produces tiny amounts of high purity REEs at a time. Solvent

extraction method is the more commonly selected separation process due to higher output capability. However, to achieve the same purity as ion exchange method, many stages are required. Generally, copious amounts of poisonous waste are produced. In the following sections both extraction techniques used in industry are discussed in more detail as well as an introduction to the proposed method of biosorption. Biosorption is intended to either replace or potentially be applied in combination with the other two techniques in industrial applications.

1.4.2 Ion Exchange Separation

The driving principle for the ion exchange method is the metal ion (M^{3+}) will exchange with protons or other cations (Ca^{2+}, Na^{+}) on a chemical ligand such as a natural zeolite or a synthetic resin. Introduction of a complexing agent, typically ethylene diamine tetraacetate (EDTA) and hydroxyethylene diamine triacetate (HEDTA), allows for separation based upon REE ion-complexation strength differences. The solution of REEs with complexing agent and retaining ions are flushed through ion exchange columns using an eluent of ammonium. The more stable complexes emerge from the column first followed by decreasing stable complexes. The REEs are then precipitated out of solution with oxalic acid and converted to oxides by heating. The products from this separation process is typically only used for limited quantity applications as in some electronic or analytical applications where high purity is needed. This is due to the limited amount that can be produced at a time. Harsh regeneration conditions also reduce ion exchange resin life, increasing process costs.

1.4.3 Solvent Extraction

For larger quantities of rare earth elements, the solvent extraction method for separation is a more practical approach. In principle, this process uses solvents that are immiscible to each other, generally an organic and aqueous phase. Chelating ligands with limited solubility in aqueous phase are added to allow dissolved (and charged) REEs to partition and concentrated within the organic phase. Rare earths are dissolved in the aqueous solvents and then the solvents are subsequently mixed and allowed to come to equilibrium [15]. Affinities for these ligands determines the concentration and partitioning in each solvent phase. The phase that contains the rare earth complexes is termed the “extract”, while the other phase containing the residuals is termed the “raffinate”. Because this process often

separates by differences in complex formation ability (in which rare earths are very similar, especially adjacent ones), separation by redox reactions are employed rather than conventional methods. This requires several separation stages with refluxes to ensure dilution of rare earths does not occur [15]. The reflux stages also help reduce the number of stages that are required to provide good separation. Separation plants typically employ a scheme consisting of multistage counter-current extractors with mixer-settler tanks all operating on a continuous flow basis [15, 16]. Some common extractants used are tributylphosphate (TBP), di-(2-ethylhexyl) phosphoric acid (D2EHPA) and various long chain amines [15, 17].

1.4.4 Biosorption as REE Extraction Technique

Biosorption is currently regarded as a cost-effective technology for treatment of wastewater. When employed, biosorption processes are generally performed in either a fixed-bed or in a fluidized-bed setup. In a fixed-bed column, the pregnant REE solution is fed through the bottom of the column. Within this setup there exists a 'dead time' for the emptying and reloading of the column once the biomass has become saturated. With a fluidized-bed column, the biomass feed and the stream containing the REEs would be fed counter-currently or co-currently in a continuous fashion. Contact time is sufficient so that the biomass is near saturation at the exit of the process and can be recycled to achieve sufficient contact time. The current technique in industry for extracting REEs, using a variety of solvents, is neither cheap nor is the produced waste beneficial for the environment [18]. Use of bacteria or another type of biosorbent could be relatively inexpensive to cultivate and would not provide toxic waste products [19]. Since a variety of biomasses have shown promise as biosorbents for accumulating various other metals in solution at low concentrations, this technique would be a "green" way to help reduce the environmental impact that results from current rare earth element extraction techniques [18, 20]. A more detailed discussion of biosorption as a competing separation technology for concentration and separation of REEs, as well as its pros and cons, will be addressed in the next section.

2. Biosorption as a Separation Process

2.1 Biosorption

Biosorption is the ability of biological material to uptake a set of chemical species by physio-chemical and metabolic pathways. It is advantageous due to the potential to be low cost and have high efficiency and affinity for the chemical of interest. In most scenarios, biosorption is similar to ion exchange in that both consist of a solid phase where chemical sorption occurs. Unlike ion exchange, the mechanisms for accumulation can be different in biosorption processes. Biosorption is being investigated as a technique for application in rare earth element accumulation [21-24]. There are a few known types of mechanisms through which biosorption occurs: intracellular accumulation, extracellular accumulation, and cell surface sorption. Factors affecting biosorption include ionic strength, pH, competing compound concentration, and temperature [25]. Mechanisms and factors that influence biosorption are explained in more detail in the following sections.

2.2 Biosorption Mechanisms

Due to the complex nature of the structure of cells, there are a variety of methods in which the uptake of rare earth elements and other metals may occur. In most cases, more than one method of biosorption is being utilized at any given point in time. Biosorption methods can be classified in a couple of ways. One classification method is based on whether biosorption is determined to be metabolically dependent or not. The other classification method depends on the location that biosorption may occur. The three locations possible are extracellular accumulation, intracellular accumulation, and cell surface sorption [18]. Extracellular accumulation results as either precipitates occurring in solution or binding of byproducts to cellular waste components. Intracellular accumulation is the precipitation occurring inside the cells, binding to cellular components internally, or remaining ions inside the cells. Cell surface sorption occurs with binding to the cell surface by either ions or

precipitates [9]. The different mechanisms through which biosorption may occur are depicted in Figure 5 and are addressed in further detail in the following subsections.

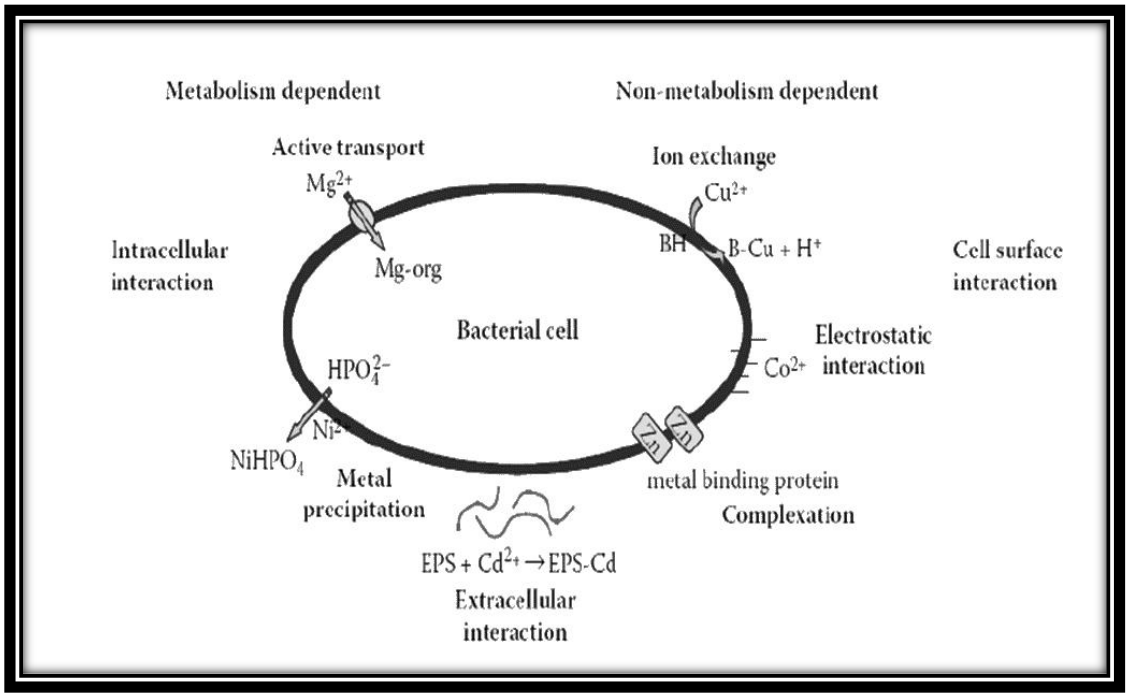


Figure 5: Mechanisms through which biosorption occurs on a bacterial cell

2.2.1 Intracellular Accumulation

Very little is understood about how this mechanism of biosorption occurs. However, it is postulated that intracellular accumulation occurs through the transport of chemical species across the membrane through channels and pumps in the same manner as other metabolically important ions are transported [18, 26, 27]. Inside many eukaryote biomass cells, metal ions are transported into various organelles where they are stored in an inert form [28]. In addition, there is potential for the binding and complexation with various free proteins and other molecules floating in the cell's cytosol. Metal localization techniques such as energy dispersive x-ray spectroscopy may be employed to provide some insight into extent of accumulation within cells, as well as, the general location of occurrence. Recovery of metals from this process requires either a "milking" process or destruction of cells to release metals.

2.2.2 Physical Adsorption

The process of physical adsorption is generally attributed to Van der Waal interactions that occur between ions and cell membranes of bacteria [18, 26, 29]. In addition, electrostatic interactions are known to occur between various metal ions in solution onto biomass [27, 29].A preference exists for surface binding for the metals with smaller ionic radii because of less steric hindrances [30, 31]. Ionic and covalent bonding between metals and the cell wall functional groups are considered to have a significant role in cell surface absorption with ionic having the greater importance. This is because larger polarizability and ionic radius is known to contribute to covalent and ionic bonding much like which occurs between Sr^{2+} and Mn^{2+} in *Saccharomyces cerevisiae* in which it is predicted to have allowed for additional covalent bonding [31]. Relative hardness and softness of metals also affects the mechanism of adsorption that occurs in biomass. Covalent bonding interactions were found to be low at low concentrations for hard metals and covalent bonding increases with metal softness and concentration. It is inferred that hard metals bind to hard ligands on the biomass surface and soft metals bind to the soft ligands[9]

2.2.3 Ion Exchange

Ion exchange method occurs when counter ions on polysaccharides within cell walls interact with and are exchanged for metal ions [18, 26, 27, 32]. An example of this is the release of Ca^{2+} and Mg^{2+} which has been observed to be linked with the uptake of heavy metal ions of living cells of *S. cerevisiae* [28]. In addition, ion exchange occurs between functional groups with protons and heavy metals as well. However, it was also observed that this mechanism only results in a small fraction of biosorbed heavy metal ions. In Gram positive bacteria a correlation is known to exist between the concentration of carboxyl groups in the peptidoglycan layer and the uptake of metal ions. Higher carboxyl group concentration yielded higher metal ion uptake concentrations [33]. Gram positive bacteria generally have enhanced capacity for metal binding due to the higher negative charge density that exists in their cell walls from the teichoic and teichuronic acids attached to their peptidoglycan networks [34, 35].

2.2.4 Complexation/Chelation

Complexation and chelation usually results from bonds forming between ions and the amino and carboxyl groups of proteins in the membranes of the bacterial cells [26]. In algae, it has been shown that there is a correlation where more carboxylic groups allow for more biosorption on the surface. Sulfonic groups have also been shown to have a similar correlation but to a lesser extent than carboxylic groups except under highly acidic conditions [29]. Cadmium, calcium, copper, magnesium, mercury, and zinc have been all found to be biosorbed by *Pseudomonas syringae* through complexation [18].

2.2.5 Precipitation

Precipitation occurs both as inorganic insoluble metal precipitates as well as organic metal precipitates. Organic metal precipitates occur when metal ions bind to Extracellular Polymeric Substances (EPS) produced by some microorganisms such as bacteria and algae [17, 20, 35]. Precipitates can occur both inside and on the exterior of the cells. Commonly with rare earth metal ions, precipitation occurs with the hydrolysis resulting in hydroxides that typically result in precipitates. Other forms of precipitation may also exist through reactions with carbonates, sulfides, and phosphates.

2.3 Factors Influencing Biosorption

Several factors influence biosorption by cells. Factors of interest to biosorption are pH, temperature, concentration of solute, ionic strength, and the construct of biomass being used for biosorption. These factors dictate the chemical speciation of the solute of interest and are discussed in further detail in the following sections. Since mechanisms are dependent on the speciation of the solute that is desired to be biosorbed, it is important to understand how these factors affect speciation to allow for predictions and engineering of biosorption systems.

2.3.1 Influence of pH on Biosorption

The pH of the solution containing metals affects the biosorption ability of the bacteria. Potential effects include a change in metal solution chemistry and speciation, bacterial cellular membrane functional groups activity, and competition by additional ions [18]. Deviations to solution chemistry typically result from one or more of the following: hydrolysis,

complexation by organic and/or inorganic ligands, redox reactions, precipitation, speciation, and the biosorption availability of the heavy metals [28]. Increases in pH have a direct relationship with increased heavy metal biosorption in part with ion-exchange and increased uptake of protons with release of light metals [29]. In a study measuring lanthanum adsorption versus pH, it was determined that there was lower adsorption at lower pH. This is predicted to be attributed to potential of cell walls being protonated in solutions at lower pH resulting in weak complexion affinity between cell walls and lanthanum ions [30].

2.3.2 Influence of Temperature on Biosorption

Metal solution temperature may also affect bacterial biosorption ability. Increasing temperature may increase the diffusion rate of ions to the cell surface, increase metabolic activity of living biomass (impacting metal uptake, efflux, and protein expression), denature proteins, DNA, and other biomolecules, and increasing energy of the system to overcome activation barriers either enhancing reactions or providing energy to break chemical bonds. Overall, temperature effects have generally been deemed to be almost negligible [26, 27]. Conversely, there are competing reports in the case of *S. cerevisiae* with one indicating an increase in biosorption with an increase in temperature and another with results to the contrary [28]. It has been shown, however, that drying temperature has a noticeable effect on biomass absorption ability. In study of lanthanide ions it was shown that, for *Mycobacterium smegmatis*, higher drying temperatures trends toward lower biosorption of lanthanide ions, potentially due to destruction of interaction sites [31].

2.3.3 Influences of REE and other Cation Concentration on Biosorption

Levels of competing metal compounds can affect the amount of desired metal ions that is able to be biosorbed. Competitive inhibition may occur between ions with equivalent size and/or charge at binding sites resulting in less desired metal ions to be biosorbed on the surface [33]. Some metal ions have been determined to influence the uptake of other metal ions by certain bacteria [18, 36]. This may occur when there are two binding sites on a transporter in the cell membrane capable of holding two distinct types of ions (via size or charge). When both binding sites are filled, the transporter may see increased activity. Ion competition can also occur for ions of the same class with preferential to the ions with smaller ionic radius [31]. Al^{3+} ions were found to decrease uptake of La^{3+} ions by 90% and Yb^{3+} ions by 94% [31].

It has been determined that light metal ions such as Na^+ and K^+ have negligible effect on the biosorption of heavy metals [28]. The presence of anions has a dramatic effect on the biosorption of metal ions due to the complexion of the metal cations with the anions in solution. The anion $S_2O_3^{2-}$ was found to increase the biosorption of metals. In contrast, the anion SO_4^{2-} was found to increase inhibition with hard metals [27]. This occurrence may be attributed to complexation resulting in precipitates incapable of being biosorbed. Biomass concentration has been also found to interfere with biomass active binding sites at high enough concentrations effectively reducing the amount of metal ion biosorbed [18]. It has been determined that in general, higher metal ion to biomass concentration ratio yields an increased level of biosorption compared to a lower ratio [28], likely due to increased driving force of higher concentration gradients. Water content also influences the biosorption abilities of biomass. It was found in an experiment using *Mycobacterium smegmatis* that wet biomass had a higher affinity for lanthanide ions compared with dried biomass [31].

2.3.4 Impact of Ionic Strength on Biosorption

While not a particular focus in this body of work, ionic strength also contributes to biosorption capabilities. The number of ions available in solution defines the characteristics of the solution and has the potential to affect biosorption. Higher metal concentrations increase the electrical potential gradient in the solution and between the metal solution and the biosorbent. The greater the gradient between the metal solution and the biosorbent, the more the biosorbent ideally would uptake metal ions. Increased ionic strength also impacts the thermodynamic activity of water and other ions in solution. Increased ions in solution effectively decrease solvent availability and shield charges for ionic species. This is also manifest on charged surfaces where potential metal binding sites may be masked from solution by counter ions in solution, increasing equilibration times and decreasing metal sorption onto surfaces.

2.3.5 Influence of Biomass State on Biosorption

Existence in a biofilm versus “planktonic cells” has a potential effect on the biosorption capacity of bacterial cells. “Planktonic cells” are cells that are not attached to each other and are essentially considered to be non-immobilized [18]. The cell surface area available to adsorption is changed depending on density and geometrical constructs of

biomass, while in planktonic cells, the full cell surface is available for adsorption. Whereas "planktonic cells" uptake is influenced by equilibrium and are governed by diffusion and their own cellular transport capabilities, immobilized biomass cells also have a contributing factor of diffusion and/or transport between cells due to mass transfer resistances, differences in metabolic activity, and extracellular polymeric materials present in biofilms which may affect the kinetics of uptake [18]. In addition, the variation in cellular makeup because of environmental differences between the cells interior to the biomass and those to the exterior can affect the adsorption capacity of the cells. Generally, protein expression due to the differences in exposure environment within each cell will exist [17]. This is of importance to note since these protein products provide binding sites for metal ions thereby effectively changing individual capacities [37].

Since protein expression effects biosorption, it is important to understand the growth phases of the biosorbent of interest. Within this project, bacteria are being studied and follow the growth phase pattern described in Figure 2.3. Expression levels of proteins vary at the different growth phases, so only one growth phase was examined for all experiments. In general, dead biosorbent cells have been found to have a higher biosorption capacity then living biosorbent cells for certain solutes and this may be due to active efflux from living cells and other protection strategies cells utilize.

2.4 Modeling Adsorption

As discussed in previous sections, there are a variety of mechanisms through which cells can accumulate a solute of interest. In addition, the mechanism by which the solute is captured is partially determined by the environmental factors of the biosorption system (e.g. pH, temperature). Isotherms adsorption models can be utilized to predict maximum uptake of solute, to better understand mechanisms of sorption, and compare between studies.

2.4.1 Comparison of Adsorption Capacity

Fundamentally, the purpose of biosorption experiments is to determine the adsorption capacity and affinity a biological sorbent has for a sorbate. Adsorption capacity is the approximate amount of sorbate per unit quantity of sorbent that a sorbent can uptake. The maximum adsorption is determined at the time when the sorbent and the sorbate come to

equilibrium with each other. At equilibrium, the quantity of sorbate bound to sites on sorbent is not changing with time or the adsorption rate of sorbate is equal to the desorption rate of sorbate. To determine adsorption capacity a material balance is generally applied comparing the quantity of sorbate initially in solution and the quantity of sorbate in the solution after exposure to sorbent. This is typically represented as

$$q = V \cdot \frac{(C_i - C_f)}{M} \qquad \text{Equation 2.1}$$

with "V" representing the volume of aqueous solution, "C_i" and "C_f" representing the initial and final concentrations of sorbate respectively, and M represent the unit basis on which the calculations are performed. This basis is typically done on a mass basis with M representing the sorbate mass utilized. In this body of work, however, M is representative of a per billion cell basis due to the adsorption process being performed with planktonic cells rather than with a weighted biomass. Determination of the cell quantity employs a UV-Vis spectroscopy with cell counting to correlate absorbance with cell concentration. The correlation is obtainable due to turbidity of solution caused by the cells. The quantity "q" represents the sorbate capacity with units of quantity adsorbed per unit basis.

A basic concept for adsorption is that on the surface of any sorbent there is a fixed number of binding sites that are available for sorption. From this concept, it follows that for any sorbent, there is a maximum possible quantity of sorbate that can adsorb to a sorbent's surface at any point in time at a set of standard environmental conditions (i.e. pH and temperature). Determination of this quantity for each sorbent provides useful information into if it is a "good" sorbent for the sorption process for a given sorbate. It also allows for comparison between experiments.

Contributing to the adsorption capacity is the "affinity" of a sorbent for sorbate. Depending on the sorbate, the sorbent that has a higher "affinity" for the sorbate will result in a greater number of sites being filled than the sorbate that the sorbent has less "affinity" for. The "affinity" of a sorbent for a sorbate is largely affected by the environmental conditions that the exposure occurs. In general, a sorbent for a higher affinity for a sorbate is more

desirable than that with a lower affinity. However, there are cases where a lesser affinity is slightly more desirable, i.e., easier to promote desorption for desorption processes.

2.4.2 Adsorption Isotherms

In addition to adsorption capacity, it is possible to describe adsorption through the usage of adsorption isotherms. There are a variety of adsorption isotherms that are available and the choice for usage is primarily dependent on the ability to fit the experimental data and the assumptions required for each. Each isotherm has its own set of assumptions that must be acknowledged when analyzing the validity of the parameters to the experimental system. Keeping this in mind, within this body of work three adsorption isotherms are applied to determine the best fit for the data. The information yielded by each is tested for validity based on the required assumptions and potential pitfalls. From these isotherms, it is possible to make estimations on the absorption behavior and certain adsorption parameters to yield optimum conditions for desired sorption quantities.

2.4.2a Langmuir Model

The Langmuir Isotherm provides a model through which the maximum absorption quantity and the ratio of adsorption to desorption rates may be determined from equilibrium concentrations in both aqueous phase and on the sorbent surface[38]. There are three main hypotheses on which the Langmuir model is based:

1. Adsorption sites are uniformly energetic.
2. There is monolayer coverage.
3. Interactions between adsorbed molecules does not exist.

There are two approaches to deriving the Langmuir model: kinetic approach and thermodynamic approach[38]. From the kinetics approach, annotating "S" as sorption sites and "M" as metal ions for sorption, the reaction is

$$M + S \leftrightarrow MS \qquad \text{Equation 2.1}$$

with "MS" representing the binding of metal ions to a sorbent binding site. In addition, the rates of this reaction are expressed as

$$r_a = k_a \cdot C_M \cdot C_S \qquad \text{Equation 2.2}$$

$$r_d = k_d \cdot C_{MS} \qquad \text{Equation 2.3}$$

where "C_M" is the aqueous metal ion concentration. "C_{MS}" and "C_S" are bound and unbound site concentrations, respectively. The rate constants "k_a" and "k_d" are described by the Arrhenius equation as

$$k_a = A_a \cdot e^{\frac{-E_a}{R \cdot T}} \qquad \text{Equation 2.4}$$

$$k_d = A_d \cdot e^{\frac{-E_d}{R \cdot T}} \qquad \text{Equation 2.5}$$

with "A_a" and "A_d" representing the Arrhenius constant for association and dissociation respectively. The "E_a" and "E_d" terms represent the activation energy required for the reaction to proceed for association and dissociation respectively. The presence of the "T" term indicates that temperature has an important role in the biosorption process by effecting the rate of which biosorption occurs.

Since there is a finite total number of sites available, the balance on sites is expressed as

$$C_{Smax} = C_S + C_{MS} \qquad \text{Equation 2.6}$$

where "C_{Smax}" represents the maximum potential binding site concentration. The reaction rates of adsorption and desorption are equal at equilibrium; therefore, the rate equations may be set equal as

$$k_a \cdot C_M \cdot C_s = k_d \cdot C_{MS} = k_a \cdot C_M \cdot \left(C_{Smax} - C_{MS}\right) \qquad \text{Equation 2.7}$$

Solving for the bound concentration, C_{MS}

$$C_{MS} = \frac{C_{Smax} \cdot b \cdot C_M}{1 + b \cdot C_M} \qquad \text{Equation 2.8}$$

which is the form commonly associated with the Langmuir Isotherm. Here "b" is the constant describing the ratio of rate constants of adsorption to desorption defined by

$$b = \frac{k_a}{k_d} \qquad \text{Equation 2.9}$$

with higher values being desirable as indicator of higher adsorption affinity. When derived from thermodynamic equilibrium principles, b is discovered to also represent the equilibrium constant and often annotated with "K_{eq}" or simply "k"[38]. Often, the "C_{Smax}" and "b" terms are also combined as a new representative term "K_m"[38]. Fractional coverage is then determined by relating the concentration of bound binding sites to the total available binding sites as

$$\theta = \frac{C_{MS}}{C_{Smax}} = \frac{b \cdot C_M}{1 + b \cdot C_M} \qquad \text{Equation 2.10}$$

where θ provides insight into the amount of actual binding sites utilized versus potential at any given equilibrium concentration of metal ions with the sorbent.

The Langmuir model also lends itself to extension to provide an estimate for estimation of sorption potential with competition between two metal ions for binding sites. Instead of a single reaction used to derive the Langmuir Isotherm for single metal ions, two reactions are applied for the competing metal ion situations[38]. These reactions are

$$A + S \leftrightarrow AS \qquad \text{Equation 2.11}$$

$$B + S \leftrightarrow BS$$ Equation 2.12

where “A” and “B” are metal ions. “AS” and “BS” are bound sites with respect to metal ions “A” and “B”. For these reactions, the rate equations are

$$\frac{d}{dt}C_{AS} = k_{aA} \cdot C_A \cdot C_s - k_{dA} \cdot C_{AS}$$ Equation 2.13

$$\frac{d}{dt}C_{BS} = k_{aB} \cdot C_B \cdot C_s - k_{dB} \cdot C_{BS}$$ Equation 2.14

and the available site balance is

$$C_{Smax} = C_S + C_{AS} + C_{BS}$$ Equation 2.15

When the system is at equilibrium

$$C_{AS} = b_A \cdot C_A \cdot C_s$$ Equation 2.16

$$C_{BS} = b_B \cdot C_B \cdot C_s$$ Equation 2.17

and

$$C_{Smax} = C_s + b_A \cdot C_A \cdot C_s + b_B \cdot C_B \cdot C_s$$ Equation 2.18

Consequently, each adsorbed metal ion is described by the following Langmuir Isotherm

$$C_{iS} = \frac{C_{Smax} \cdot b_i \cdot C_i}{1 + b_A \cdot C_A + b_B \cdot C_{B.}} \qquad \text{Equation 2.19}$$

where "I" represents the metal ion of interest, i.e. "A" or "B". It follows that fractional coverage if given by

$$\theta_i = \frac{C_{iS}}{C_{Smax}} = \frac{b_i \cdot C_i}{1 + b_A \cdot C_A + b_B \cdot C_{B.}} \qquad \text{Equation 2.20}$$

2.4.2b Freundlich Model

There are some adsorption instances in which the Langmuir model does not provide a good fit for experimental data. Another proposed model that often can describe adsorption data is the Freundlich Isotherm often depicted as

$$C_{MS} = K_f \cdot C_M^{\frac{1}{n}} \qquad \text{Equation 2.21}$$

where K_f and n are empirical constants determined from fitted data[38]. Historically it has been difficult to definitively assign significance to the Freundlich constants; however, derivations of this equation provide some insight into significance. The informational value, though, depends on the derivation from which it is applied. The derivation that best fits this body of work comes from application of fractal kinetics[39]. From fractal kinetics, the rate law is generally expressed as

$$\frac{d}{dt}C = k_{fr} \cdot C^{n} \qquad \text{Equation 2.22}$$

where "k_{fr}" and "n" are the fractal rate coefficient and order of reaction, respectively.

If it is considered that the sorbent is a porous media with heterogeneous surfaces, then all sites are not in equal contact with the bulk concentration and diffusion must occur to unexposed binding sites[39]. Fractal kinetics applies in this case because fractal rate law assumes diffusion-controlled reactions with geometrical constraints can be described by reactions on fractal domains. The effective molecular diffusion coefficient and the rate constant for fractal reactions is predicted to be proportional in the same way as molecular diffusion and rate constant in classical first order kinetic reactions[39].

By applying the fractal kinetic rate equation for both adsorption and desorption

$$\frac{d}{dt}C_M = -k_1 \cdot C_M^{\,n_1} + k_2 \cdot C_{MS}^{\,n_2} \qquad \text{Equation 2.23}$$

where "k_1" and "k_2" are fractal rate constants of association and dissociation, respectively. Also, "n_1" and "n_2" are the fractal dimensions with of association and dissociation reactions, respectively[40]. Since at equilibrium the derivative is zero, after rearrangement, the Freundlich equation is produced as

$$C_{MS} = \left(\frac{k_1}{k_2}\right)^{\frac{1}{n_2}} \cdot C_M^{\frac{n_1}{n_2}} \qquad \text{Equation 2.24}$$

where the ratio of fractal rate constants of association to dissociation is defined for the Freundlich parameter "K_f" and the ratio of fractal dimensions of dissociation to association is defined as the parameter "n"[39]. Studies into the meaning of the fractal dimensions provides a better understanding of the accessibility of adsorption sites and the probability of access to the adsorption sites that these constants signify. In simplest terms, the 1/n term indicates deviation away from a homogenous surface contacting a well-mixed solution.

2.4.2c Sips Model

The Sips Isotherm is a combined form of the Langmuir and Freundlich Isotherms. As such, it is often referred to as the Langmuir-Freundlich Isotherm [38]. This model accounts for the problem of continuing increase in adsorbed amount with increase in concentration that occurs with the Freundlich equation. Often the Sips Isotherm is expressed as

$$C_{MS} = \frac{C_{Smax} \cdot K_S \cdot C_M^{\frac{1}{n_S}}}{1 + K_S \cdot C_M^{\frac{1}{n_S}}} \qquad \text{Equation 2.25}$$

where "K_s" is the Sips equilibrium or "affinity" constant and "n_s" is the Sips model exponent. It is worth noting that when "n_s" is at unity, the Sips model produces the Langmuir isotherm. As a result, n_s is considered the parameter describing system heterogeneity. In addition, at low adsorbate concentrations, the Sips model reduces to the Freundlich isotherm. Unlike the Langmuir and Freundlich isotherms, Sips is derived using an energy distribution approach. It results from an integral equation describing the average of the local Langmuir isotherm over an energy distribution function with the integral being

$$C_{\mu} = C_{\mu s} \cdot \int_{-\infty}^{\infty} \frac{b_{inf} \cdot e^{\frac{E}{R \cdot T}} \cdot P}{1 + b_{inf} \cdot e^{\frac{E}{R \cdot T}} \cdot P} \cdot F(E)\, dE \qquad \text{Equation 2.26}$$

and the energy distribution function having the form of

$$F(E) = \frac{1}{\pi \cdot R \cdot T} \cdot \frac{e^{\frac{1}{n} \cdot \frac{E_m - E}{R \cdot T}} \cdot \sin\left(\frac{\pi}{n}\right)}{1 + 2 \cdot \cos\left(\frac{\pi}{n}\right) \cdot e^{\left(\frac{1}{n} \cdot \frac{E_m - E}{R \cdot T}\right)} + e^{\left(\frac{1}{n} \cdot \frac{E_m - E}{R \cdot T}\right)}} \qquad \text{Equation 2.27}$$

where E_m is the energy of the distribution is the maximum. The energy distribution function is an exponential decay function regarding the energy of adsorption and exhibits the shape of a Gaussian distribution.

2.4.3 Heats of Sorption

Isosteric heat is the ratio of the miniscule change in the enthalpy of an adsorbate to the miniscule change in the amount absorbed. This information is valuable to kinetic studies of adsorption because the kinetics of mass uptake is determined by the cooling rate of the sorbent. As the temperature of the sorbent rises the rate of heating or cooling of the sorbate decreases[38]. The temperature increase is a result from the absorption of a portion of the heat being released by the sorbate during the adsorption process from the sorbate. Isosteric

heat may be affected with loading of the sorbent due to lateral particle-particle interactions[38].

Using the Sips isotherm equation, it is possible to obtain this information through the development of the model into a temperature dependent model. The two parameters with temperature dependencies that must be accounted for is the Sips equilibrium constant and the Sips model exponent[38]. In the temperature dependent form, these constants can be substituted with the temperature dependent functions. The Sips equilibrium constant can be represented as

$$K_S = K_{s0} \cdot e^{\frac{Q}{R \cdot T_0} \cdot \left(\frac{T_0}{T} - 1 \right)} \qquad \text{Equation 2.28}$$

with "K_{s0}" representing the Sips equilibrium constant at a reference temperature "T_0" and Q referring to the heat of adsorption. In addition, the Sips model exponent can be represented as

$$\frac{1}{n} = \frac{1}{n_0} + \alpha \cdot \left(1 - \frac{T_0}{T} \right) \qquad \text{Equation 2.29}$$

where "n_0" is the Sips model exponent at reference temperature and "α" is a fitted parameter.

It is also possible to consider the dependence that temperature may have on the maximum adsorption capacity, it may be determined by a function of temperature in comparison to a reference temperature as

$$C_{Smax} = C_{Smax0} \cdot e^{\left[\chi \cdot \left(1 - \frac{T}{T_0} \right) \right]} \qquad \text{Equation 2.30}$$

with "C_{Smax0}" referring to the maximum adsorption capacity at reference temperature and "χ" being a fitted parameter.

Through nonlinear fitting of the quantity of absorption with temperature at a fixed exposure concentration, with the Sips Isotherm modified for temperature dependence, the constants of "α" and "χ" may be obtained in addition to the heat of adsorption "Q". The isosteric heat, which varies with temperature, can be determined using the van't Hoff equation

$$\frac{d}{dT}\ln\left(K_s\right) = \frac{\Delta H}{R \cdot T^2} \qquad \text{Equation 2.31}$$

and applying the temperature dependent form of the Sips equation for "K_s" resulting in

$$\Delta H = -\left(Q - \alpha \cdot R \cdot T_0 \cdot n^2 \cdot \ln\left(\frac{C_{MS}}{C_{Smax} - C_{MS}}\right)\right) \qquad \text{Equation 2.32}$$

with the substitution of the functions for "C_{Smax}" and "n" may occur if parameters are deemed to be nonnegligible with temperature variation. Through the isosteric heat equation, it is also capable of being shown that the heat of sorption parameter "Q" is equal to the isosteric heat at fractional loading of 0.5 [38].

After a model organism is defined for its biosorption capabilities, many other organisms and/or other types of biomass are of interest to be studied as they may provide better biosorption capabilities (i.e. uptake capacity under different conditions, selectivity, etc.). Due to the expansive field to choose from for biosorbents, it is necessary to have a technique that will allow for rapid screening to quantify sorption. The current technique, which is the one used in this thesis, is biosorption in mesocosms. The development of a device that reflects the results in mesocosms studies for the model organism will solve this need for rapid screening.

3. Potential High Throughput Screening Technologies of Biomass

Knowledge about the adsorption capacity for a particular biomass and the parameters obtained from its adsorption isotherms allows for comparison between biomass types or between metals on the same biomass. However, determining adsorption isotherms is a time and resource intensive process, especially for biomass of unknown ability. The current method being used by most researchers are mesocosms which are both time and resource intensive. Screening technologies for biomasses help characterize biosorbents in a quick and cheap manner before more extensive testing, such as mesocosm adsorption isotherm evaluation, are performed. Potential methods of screening include field flow fractionation, dielectrophoresis, and flow cytometry. Dielectrophoresis is the technique which we considered as it displays characteristics of both of the other separation techniques including small channel sizes (similar to flow cytometry) and imposed electric field (similar to field flow fractionation). However, due to the likely increased separation times unique to dielectrophoresis, increased thermal and pH gradients may form and were a subject of study for this work. Each biomass evaluation technique will be addressed in more detail below.

3.1 Biosorption in mesocosms

The most popular method for biomass screening is through mesocosms. This method is the most accurate method known for quantification. Within this method conditions are controlled with the main goal being to understand the underlying mechanisms and effects of different environmental conditions (e.g., pH, temperature, metal concentration, competing ions or ligands). Some disadvantageous of this technique include relatively large volumes of sample needed for analysis, slow sample processing, and expensive analysis. The preparation of samples can also be costly due to time and equipment requirements.

3.2 Field Flow Fractionation (FFF)

Field flow fractionation is a separation technique that works based on size of molecules. This separation method is accomplished by creating equilibrium layers for each

component in a sample system at unique positions in a parabolic flow profile as seen in Figure 6 [41]. For separations of biomass, biomass particles are separated by density due to induced gradients by the FFF apparatus. The density different is desired to be due to loading variances, but may also be due natural density variations between biomass particles thereby reducing accuracy of this method.

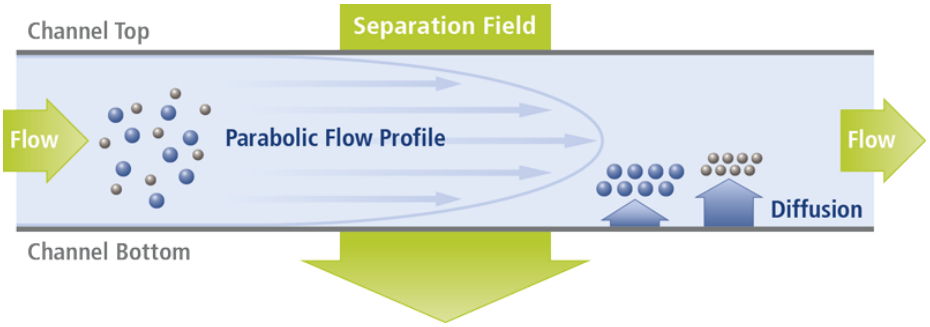


Figure 6: Field Flow Fractionation Principle [42]

3.3 Flow Cytometry

Flow cytometry allows for the ability to measure individual particles. Sample particles entering the system diverted into a stream of single particles that are individually scanned by the instruments detection system utilizing hydrodynamic focusing [3]. This system distinguishes particle types by focusing lasers and measures light scattering or fluorescence emission of each particle. Utilization of an electrostatic cell sorting system separates the cells by charges[3].

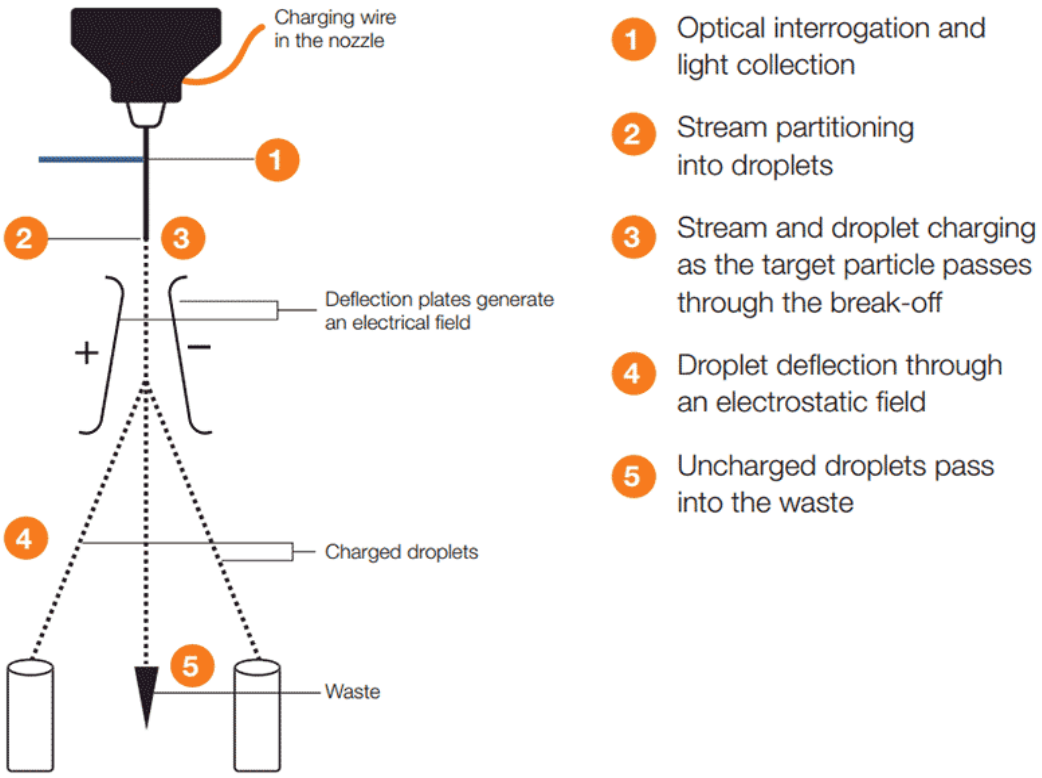


Figure 7: Cell sorting and flow cytometry. [3]

3.4 Electrophoresis

Separation by electrophoresis is performed by separating particles of different charges as they move through an electric field. The variation in charges on particles result in different separation rates as a particle moves along an electric field toward an anode or a cathode.

3.5 Dielectrophoresis (DEP) as a separation technique for OHMs

Dielectrophoresis is a promising process which lends itself for exploitation in separating cells by REE content. The high sensitivity of this process along with its nondestructive behavior makes it an ideal process for separation of cells. It is for this process that the work in this thesis is based.

3.5.1 DEP Theory

The process in which a particle migrates toward a location of maximum field strength in a non-uniform electric field is dielectrophoresis (DEP)[43]. This interaction is a result of an induced dipole in dielectric material. Magnitude and direction of induced DEP force depends on characteristics of applied electric field as wells as the dielectric properties of the particle and its media being studied[43]. The DEP force only is experienced when particles are present in a non-uniform electric field and is not polarity dependent. If the permittivity of a particle exceeds the permittivity of the media that the particle is suspended in, the particles are attracted to regions of stronger electric field[43]. Conversely, if the media permittivity is greater than the particle permittivity, particles are repelled from regions where there are strong electric fields[43]. Most experimental applications use frequencies greater than 100 kHz and magnitudes of less than 20V peak to peak[44]. Applied signal is typically sinusoidal in shape, however some applications have used pulse signals. DEP effect is mostly observed with particles that have diameters between 1 and 1000 micrometers[44].

Dielectrophoresis has various applications through manipulation, transportation, and separation of various sorts of particles[43]. For example, at a given frequency of an applied AC electric field, particles with different polarizabilities move in different directions allowing for their separation[44]. The electrophysical properties that many biological cells possess have made it possible to apply DEP to both characterize and separate them[44]. DEP has shown the ability to separate living and dead cells based from electrophysical differences. It also has shown utility in separating cancer cells from blood as well as the detection of apoptosis[44].

3.5.2 REE-Bacterial Cell Separation

Most cells possess a net charge, either positive or negative, due to the ionizable functional groups that exist on the surfaces of cells. After uptake of rare earths by the cells, the overall charge will change and become more neutral to positive depending on level of uptake and level of binding onto the cell surfaces, as well as, inside the cells. Polarizability of cells may vary depending on morphology, composition and phenotype. These features of

cells remain constant within a subset of cells at a specified growth phase and constant environmental conditions of media allowing for characterization of both loaded and unloaded bacterial cells. The differences between loaded and unloaded cells in charge, mass, and density all allow DEP to separate by varying magnitude and frequency of applied electric field. Design of a system will allow maximum possible applied flow rate while being selective to sets of cells with similar dielectric properties. By applying remote electrodes at the entrance and exit of a microchannel, a non-uniform electric field is generated. Variation in channel geometry or by insulating structures within the channel deform the electric field in the channel while dynamically changing the flow field in which the cells will travel.

The design of the device for separation by dielectrophoresis is beyond the scope of this thesis but the proposed design for which the information developed here is critical is shown in Figure 8. At location "A" in the figure, a sample of cells that were exposed to metals are introduced into a channel on the microdevice. As the cells move through the channel they are subjected to a non-uniform electric field created by imbedded insulating obstacles. Cells that hyperaccumulate metals stratify from cells that possess lesser content of metals as indicated by panel C in the figure. Panel D in the figure demonstrates DC-biased AC DEP or fluid flow tangential to bulk motion, which effects microfluidic separation.

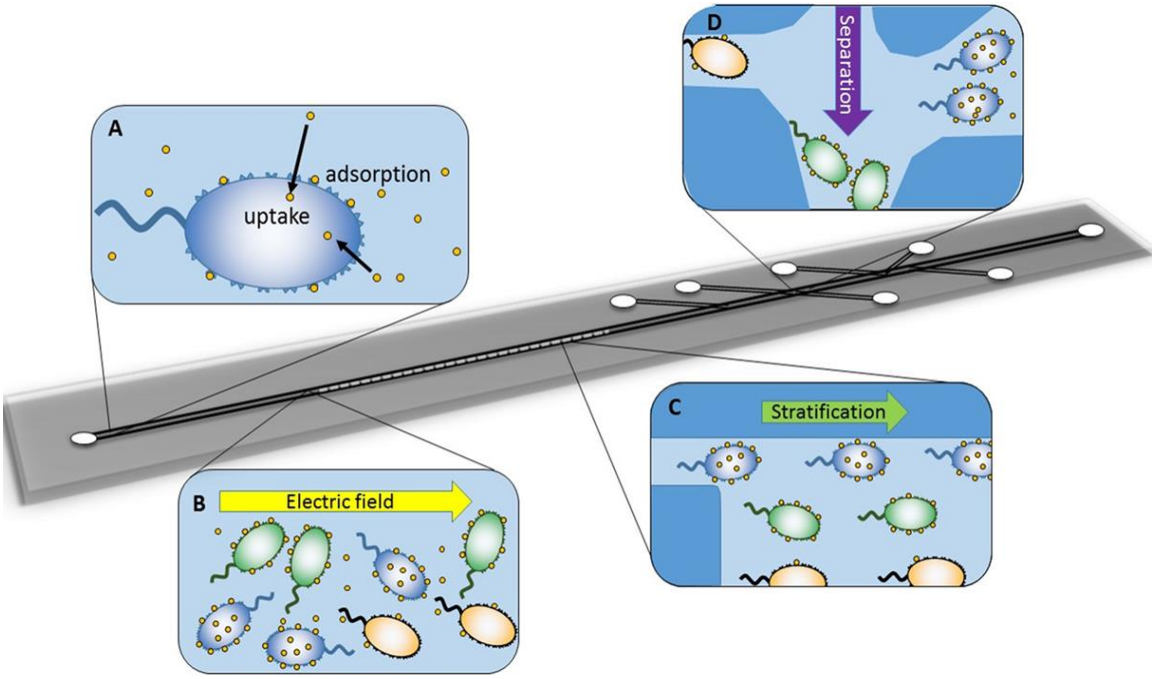


Figure 8: Microfluidic separation of rare earth element (REE) hyper-accumulating bacterial strains. A) Metals (shown as yellow circles) accumulate onto and within microorganisms through two main mechanisms of surface adsorption and cellular uptake through passive diffusion of neutral chemical species and specific and non-specific transport mechanisms, which effects the cell's dielectric properties. B) A heterogeneous mixture of bacteria and dissolved metals showing differential metal loading capacities. OHMs (depicted in blue) both uptake and surface adsorb metals. Cells whose accumulation is strictly governed by surface adsorption of metals are shown in green. Metal deficient cells (in yellow) accumulate few metals. C) When subjected to a non-uniform electric field (under low frequency AC and DC-bias voltages) in a microchannel having embedded insulating obstacles to create non-uniformity, OHMs with higher metal loading stratify from cells with lesser metal content. D) Demonstration DC-biased AC DEP or fluid flow tangential to bulk motion, which effects microfluidic separation. Complex and varied geometries of embedded insulating obstacles are possible to effect OHM separation.

4. Summary of Theory and Applicability

Overall, the chemical behavior of REEs play an important role in the separation of these elements. Due to the similarities in their properties, as outlined in sections 1.2 and 1.3, REEs possess low separation factors between each other, especially so between neighboring REEs. As a result, it is very difficult to separate these elements, as is pointed out in section 1.4. In addition, there is the additional problem of REEs occurring in very dilute concentrations compared to other elements in nature which is the main issue in current mining methods that is pointed out in section 1.4. To resolve the issue in mining REEs in an environmentally friendly manner, biosorption process are seen as an ideal solution. To effectively select a biosorbent to perform the extraction of REEs and concentrate them for further downstream process, a thorough knowledge on the possible mechanisms that a biosorbent may use is needed. For comparison purposes, isotherms are used to give a general basis on the performance of biosorbents versus other types of sorbents. However, through the understanding of biosorbent mechanisms that are involved, it is clear that isotherms are not fully suited to provide the whole picture on how the biosorbent process works. This is mainly due to the assumptions in which isotherms were developed, as seen in section 2.4, which is mainly based on physical adsorption to a surface. Through the knowledge of biosorption mechanisms that were developed through section 2.2, it is very apparent that these isotherms do not describe the biosorption process fully, and results can be expected to possibly reflect this. The isotherms do still have a place within biosorption analysis, though, since they still give some preliminary data that some general assumptions can be made with isotherm limitations and deviations kept in mind. This preliminary data set provides sufficient data for the purposes of comparison between biosorbents, as well as provide key estimates for parameters that influence potential separations processes as those outlined in Chapter 3. It is with all these considerations that the following chapters are developed for provide context for the results as a whole.

5. Toward Inexpensive REE Determination

5.1 Background

To characterize biosorption, a technique to quantify the metals should be employed. Since the concentrations that rare earth elements usually occur in low concentration, a high sensitivity methods is required. The most common method to measure REEs is through inductively coupled plasma mass spectrometry (ICP-MS). However, other techniques are available including inductively coupled plasma optical emission spectroscopy (ICP-OES), microwave plasma atomic emission spectroscopy (MP-AES), and spectrophotometry. For this thesis, the main method used was an MP-AES system. Due to the costs associated with performing analysis on this equipment, alternative methods were desirable to perform quick evaluations at relatively low cost. Additionally, a spectrophotometric method is desirable as it might be able to be used concurrently to measure metal concentrations on the separation device. Two methods investigated for this purpose were through UV-Vis spectroscopy and fluorescence spectroscopy.

5.1.1 Inductively coupled plasma mass spectrometry (ICP-MS)

ICP-MS couples and high-temperature ICP source with a mass spectrometer to convert and quantify elements by ions. Analysis through this equipment is expensive ranging from $1500-2500 [45]. This method is desirable due to its low detection limits in ppb range [45]. ICP-MS is the most common technique in the analysis of rare earth elements. Calibration standards are required to determine the concentration of an element within each sample. Disadvantages for this technique are high sample cost and potential interferences from similar size/charge atoms or molecules also known as isobaric interference. The isobaric interferences can be corrected for by using a collision cell to selectively remove ions with larger cross-sections from the detection path, or selecting non-dominant isotope lines.

5.1.2 Inductively coupled plasma optical emission spectroscopy (ICP-OES)

ICP-OES is similar to the ICP-MS but its detection is based on optical emission rather than mass spectroscopy. To determine quantities of elements, the optical emission spectrometer measures the intensity of radiation at element-specific wavelengths [46-48]. Calibration standards are required to determine the concentration of an element within each

sample. This method can detect in ppb concentrations [46, 48]. The disadvantage of this method, however, is that emission lines may overlap preventing quantification [46].

5.1.3 Microwave plasma atomic emission spectroscopy (MP-AES)

MP-AES determines an analyte element by its electromagnetic spectrum. When an elemental atom is excited, an emission spectrum is created as it returns to its ground state [49]. The source for excitation results from a microwave plasma fueled by nitrogen at temperature nearing 5,000K [49]. An optimized microwave guide is used in the MP-AES to concentrate electromagnetic fields at the torch. An axial magnetic field and a radial electrical field focus the microwave energy to create a plasma. Sample is introduced to the center of the plasma after being converted by a nebulizer from a liquid into an aerosol. Emission is detected by a fast scanning monochromator onto a high efficiency CCD detector that scans both the background and sample spectra simultaneously [49]. Quantification is determined based on a calibration curve created from a set of sample standards provided by the user. Intensities are then converted to concentrations by the software for the user.

5.1.4 UV-Vis spectroscopy

Since costs of equipment consumables and time are expensive, an alternative technique was desired to determine rare earth quantification that was less expensive and required less time for sample preparation while dealing with small volumes. UV-Vis spectroscopy is a method which seems to meet these requirements. Quantification of rare earth in aqueous solutions by UV-Vis is possible through the application of Beer's Law. Beer's Law is described by

$$A = \varepsilon \cdot b \cdot c$$

with "A" represents unitless quantity absorbance, "ε" is the molar absorptivity, "b" is the path length of the cuvette and "c" is the concentration of the solution. According to Beer's Law, the fraction of the light absorbed by each layer of solution is the same. Since molar absorptivity is constant and the path length of cuvette is generally constant, concentration is directly correlated to absorbance. Where there is a linear correlation between concentration and absorbance, Beer's Law is applicable.

5.1.4a Arsenazo III

Since rare earth elements have low molar absorptivity in the UV-Vis range by themselves, an indicator, arsenazo III, is used that has complexes follow Beer's Law. arsenazo III (1,8-dihydroxynaphtalene-3,6-disulphonic acid-2,7-bis[(azo-2)-phenylarsonic acid]) is an azo-dye that gives color reactions with a variety of different elements allowing for quantification by photometry[50]. The structure of arsenazo III is shown in Figure 9. It is most commonly known for its usage to detect calcium levels in blood [51, 52]. Extensive studies with this reagent have also been previous performed on thorium, zirconium, uranium, lead, zinc, and some lanthanides[53-56]. Reaction color is dependent on pH with complexes at pH greater than 5 exhibiting blue to violet colors while those with pH values below 4 present an emerald green color[57]. Elements with cation charges of 2^+ and 3^+ form 1:1 complexes with arsenazo III, such as is the case with many of the rare earth elements [58]. Smaller cations, such as Ni^{2+} and Mg^{2+} form 2:1 complexes [58-60]. The conformation of arsenazo III has two main conformations: an extended and a compacted. The compacted occurs with the binding of the smaller cations while the extended conformation facilitates only large cations like Ca^{2+} and La^{3+} [58, 61]. Arsenazo III is sensitive to both oxidizing agents and strong reducing agents so it is imperative to either remove these elements from the system or arsenazo III must be in large enough excess to overcome these conditions.

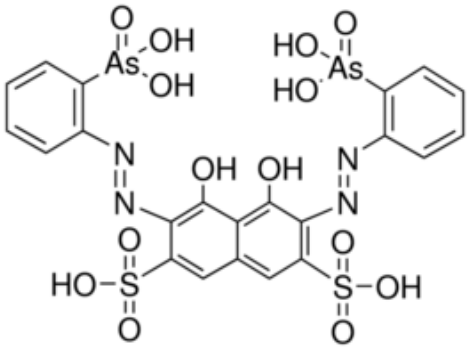


Figure 9: Structure of the colorimetric dye arsenazo III.

5.1.5 Fluorescence Spectroscopy

The method of fluorescence has many current usages including flow cytometry, DNA sequencing, and in medical diagnostics. Detection by fluorescence is a highly sensitive method for detection. Fluorescence is a subset of luminescence defined by excited state of a

molecule. When an electron returns to the ground state from an excited singlet state, fluorescence occurs. This usually happens at an emission rate of approximately 10^8 s^{-1} giving it a lifetime of only 10 ns [62]. Phosphorescence, another subset of luminescence, is the emission of light from triplet excited states. Since transition to ground states are forbidden, phosphorescence lifetimes are longer and range from milliseconds to seconds [62]. Fluorescence usually occurs from aromatic compounds [62]. Data from fluorescence is usually demonstrated as an emission spectrum plotting fluorescence intensity versus wavelength (nm) or wavenumber (cm^{-1}) [62]. A characteristic of fluorophores is the energy of emission is normally less than the energy of absorbance, therefore fluorescence occurs at either lower energies or at longer wavelengths. Emission spectra is also independent of excitation wavelength. photon emission (fluorescence or phosphorescence) occurs in appreciable yield only from the lowest excited state of a given multiplicity (Kasha's rule) [62]. With fluorescent indicators, which in this case is arsenazo III, emission intensity increases as concentration of complex between indicator and metal ion increases. This property possessed by fluorescent indicators is referred to as fluorescence sensing [62].

5.2 UV-Vis Methods

5.2.1 Preparation of Arsenazo III stock solution

To make the stock solution, 0.0506 grams of Arsenazo III was added to 500 mL of 18.2 MΩ H_2O to make a solution of $1.303x10^{-4}$ M. The container was wrapped in aluminum foil to protect from light and refrigerated.

5.2.2 Preparation of Arsenazo III in acetic buffer Solution

To prepare acetic buffer for pH 5.0 at 18°C, 70 mL of sodium acetate of 0.2 M was used in 30 mL of acetic acid. Then 60 mL of Arsenazo III stock solution was combined with 40 mL of acetic buffer solution. The container was wrapped in aluminum foil to protect from light and refrigerated to remain stable.

5.2.3 Preparation of Calibration Curve for REEs

Samples of 10 µM, 15 µM, 20 µM, 25 µM, 30 µM, 40 µM, 50 µM, 60 µM, 80 µM, and 100 µM for each REE of interest were prepared by diluting the stock solution of the REE into 18.2 MΩ H_2O to a volume of 5 mL in 15 mL VMR tubes. Each sample was vortexed for five seconds to ensure homogenous mixture of REE solution. An addition of one milliliter of Arsenazo III in Acetic Buffer solution was added to 5 mL sample of REE solution. Each solution was vortexed for five seconds to mix thoroughly. Process was repeated for each matrix evaluated.

5.3 UV-VIS Analysis and Results

Two mL of Arsenazo III-REE solution were added into a disposable cuvette of 1 cm path length and inserted into UV-Vis spectrometer and UV-Vis spectrum taken from 190 nm to 1100 nm with a 1 nm step size. Five replicates for each concentration were taken. The mean of absorbance for each wavelength was determined at each concentration. The CORREL function in Excel 2016 was used across all concentrations at each wavelength to determine if a linear correlation existed. The wavelength with the highest absolute value of the result of the CORREL function was plotted absorbance versus concentration to obtain a linear correlation that was used to determine concentration of REE in solution. From this correlation, an absorbance reading was only required at the specified wavelength from which the correlation as made to determine concentration.

First spectrum for the absorbance variation due to Arsenazo III concentration alone was obtained to determine Arsenazo concentration effect of UV-Vis spectrum. From there, REE was added to verify the concentration appearance in a variation in absorbance at a wavelength with complex concentration. Figure 10 demonstrates the effect of variation of Arsenazo III on spectrum and complex formation. Optimum molarity of Arsenazo III solution was predicted to be about a 0.6M solution to provide a distinguishable enough absorbance while not oversaturating the system with Arsenazo III.

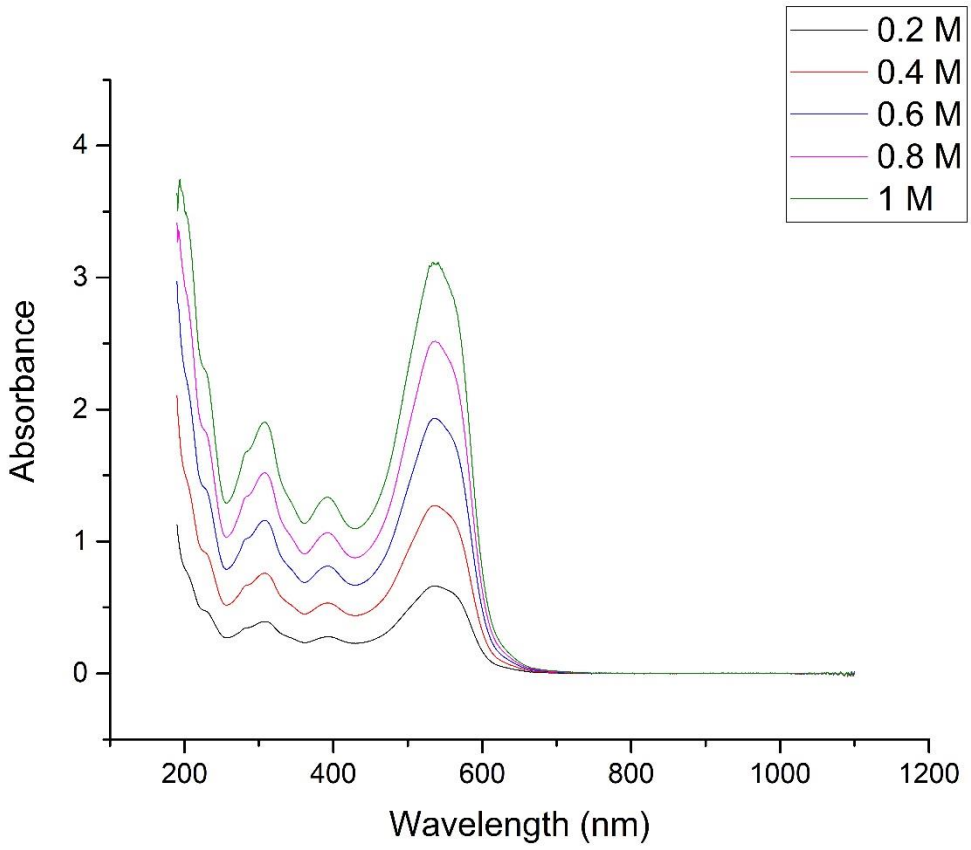


Figure 10: Comparison of Arsenazo Concentration on UV-Vis Absorbance

Since it is desirable to measure at various pH values, an absorbance spectrum was taken over various pH ranges to see effect on absorbance with pH. From Figure 11 it is determined that there is an effect on the spectrum in both peak intensity and position. As a result, it was found to be desirable to add a buffer to the Arsenazo III solution in order to control the effect of pH on Arsenazo III absorbance.

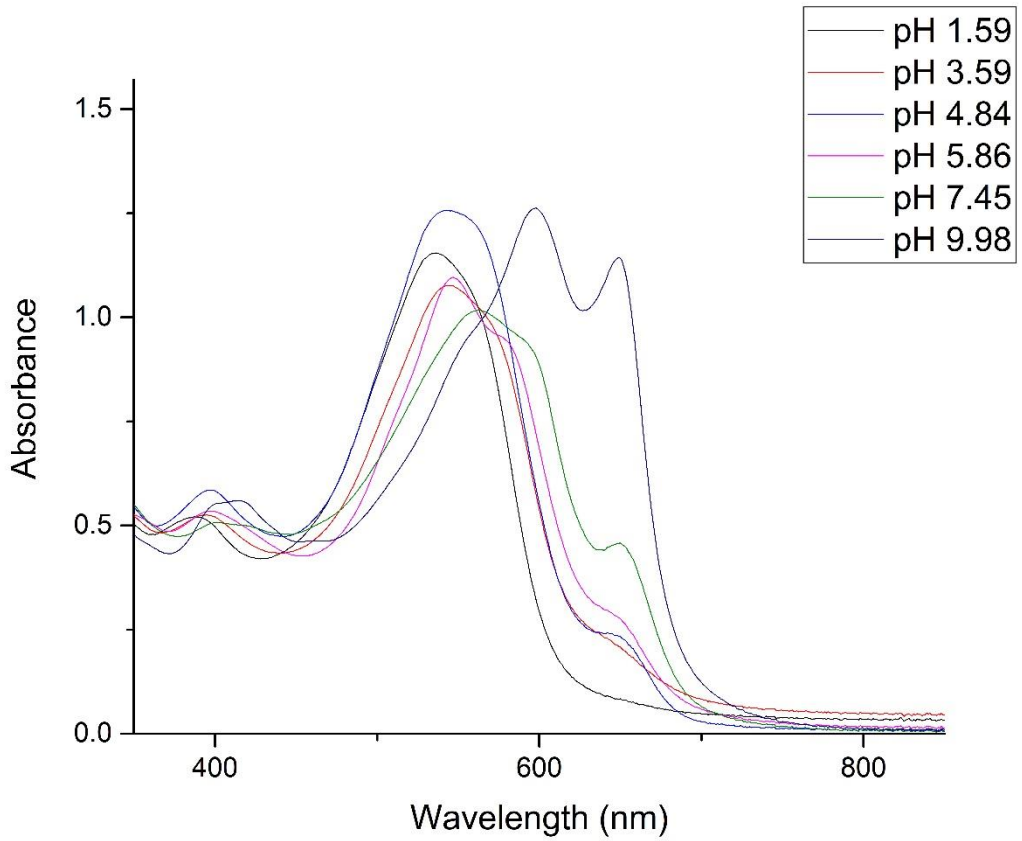


Figure 11: Effect of pH on Arsenazo III only Absorbance

Following the optimization of the Arsenazo III dye solution, REEs were evaluated at various concentrations in various matrixes to determine if correlation could be found between concentration and absorbance. As can be seen in Figure 12 it is noticed that there is a trend between Arsenazo III-REE complex absorbance and REE concentration in both matrices investigated. With increasing concentration of each REE, there is seen an increase in absorbance at most wavelengths throughout the spectra. This trend of increasing concentration resulting in increased absorbance readings is noticed with all REEs tested. This indicates the potential for a possible correlation from which concentration may be determined through knowledge of UV-Vis absorbance.

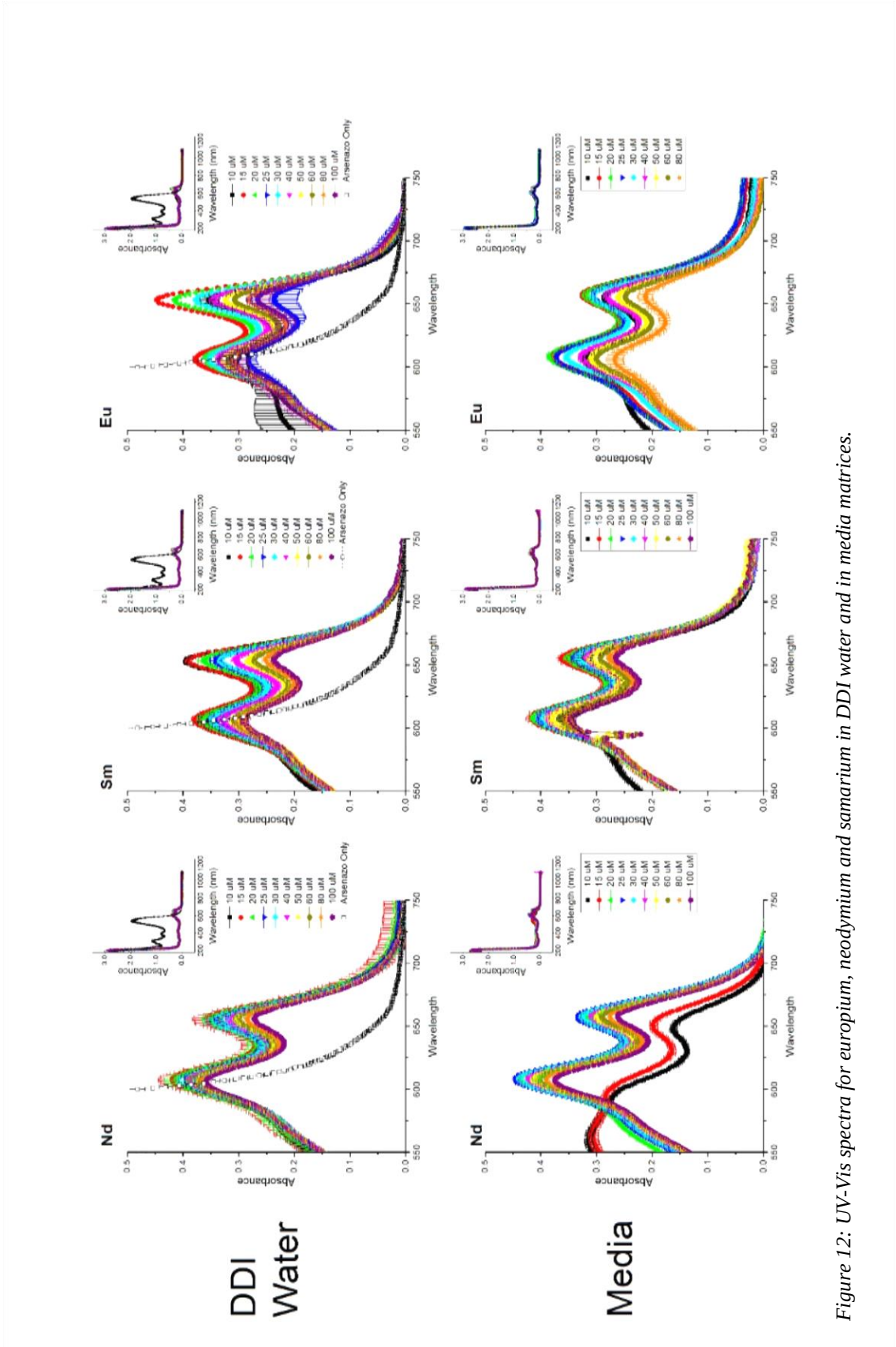


Figure 12: UV-Vis spectra for europium, neodymium and samarium in DDI water and in media matrices.

Resulting from linear correlation search using CORREL function in Microsoft Excel linear correlations for europium, neodymium, and samarium were determined to exist. From these correlations, it is possible to determine the quantity of REE in solution. For each REE, there are two correlations applicable: one for low concentrations up to approximately 15uM, and the other for higher concentrations up to 100 µM. The higher correlation may extend beyond 100 µM but was not studied in this work.

By comparing the REE UV-Vis absorbance spectra, it is apparent that this method would not be useful for analysis to distinguish between REEs. Because the peaks of absorbance overlap with each other. In addition, due to the different intensities of absorbance at the same concentration, it is not possible to use to determine total REE contents. The issue of peak overlap and intensity variation at a fixed concentration is shown in Figure 13. By visual inspection, it can be seen in Figure 12 that the same overlap occurs with media matrix as well. This would indicate that this is not an anomaly, but may occur in many other matrices. Therefore, this method of quantification is only applicable to be used in single, well defined REE systems with detection limits determined by sensitivity of UV-Vis spectrometer.

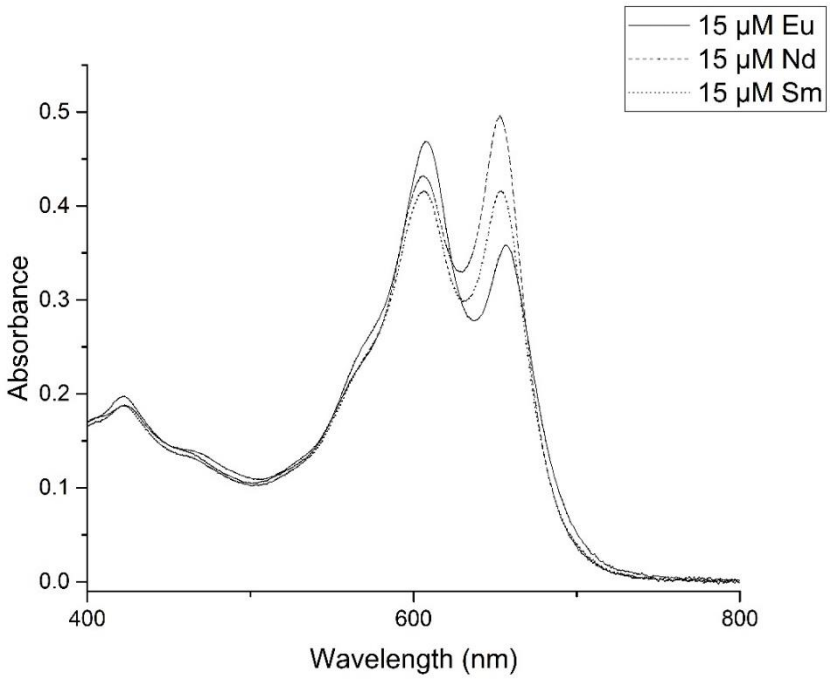


Figure 13: UV-Vis spectra comparison for REEs at fixed concentration of 5uM.

5.4 Fluorescence Methods and Results

In a quartz cuvette, 1 mL of Arsenazo III-REE solution was added. Using a Shimadzu RF-5301PC spectrofluorophotometer with a xenon lamp, a 3d increment emission wavelength scan was run. Excitation and emission wavelengths were set to start at 200 nm. Emission was set to stop at 900 nm. Emission increments of 15 nm was used. The "Excitation Slit Width" and "Emission Slit Width" were set to "Nm_10". Sensitivity was set to "High". Scan speed was set to very fast and the sampling interval to "Nm1_0" with no sipper. All files were saved as *.txt. Files were imported into MATLAB program to determine location of linear correlation between intensity and concentration added. For all samples to be analyzed using curve, the excitation and emission start and stop were set to 100 nm before and after location of peak.

The fluorescence method for rare earth quantification is possible through comparison of concentration versus intensity, much like UV-VIS method. However, determination of a correlation for quantification required a different method of discovery. When a sample for fluorescence was run, 3D data describing the intensity at an excitation and emission wavelength was output by the spectrofluorophotometer. A MATLAB program was developed to overlay and plot the various concentrations over each other in a waterfall plot. An example of the 3D fluorescence intensity at ranging emission and excitation is shown in Figure 14.

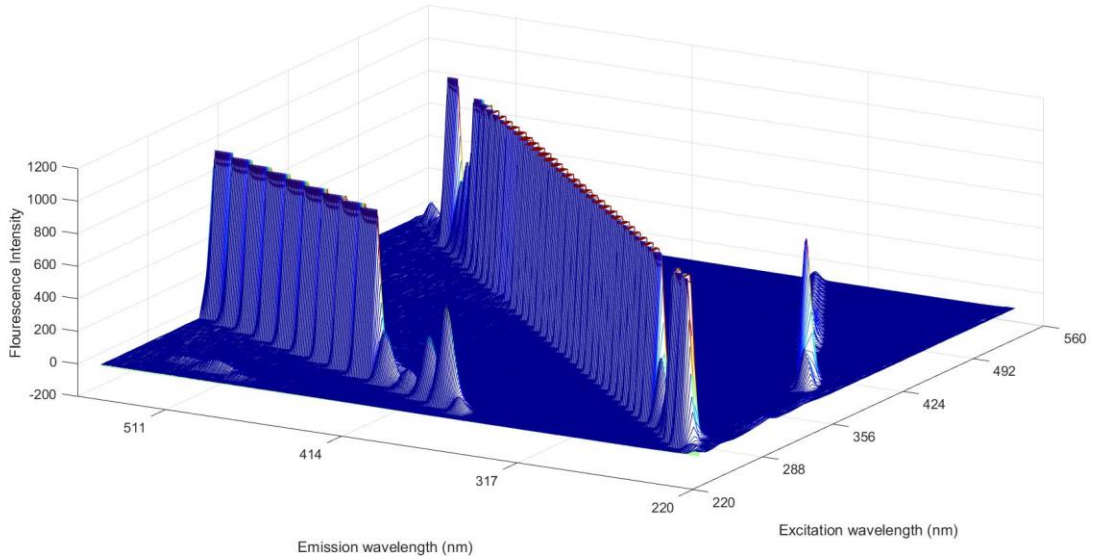


Figure 14: 3D plot of fluorescence intensity of 10 ppm of samarium-arsenazo III complex in acetic acetate buffered saline.

To find a suitable linear correlation, each excitation and emission wavelength was treated like a coordinate location. The MATLAB program developed correlates by plotting the known concentration of the sample against the intensity. The program outputs are recorded into a matrix including the r-squared correlation for each location and identified all locations where the r-squared correlation met a specific criterion. Since a high linear correlation was desired, the criteria to determine the correlation was set to be greater than 0.999. When MATLAB found a correlation meeting the criteria, it was instructed to record the excitation and emission location into a new matrix. Fluorescence emission spectra was analyzed for this process with triplicates for europium, neodymium, and samarium in both a DDH_2O matrix and an acetic acetate buffered saline (AABS) matrix. Also, samples in AABS with 2% nitric acid were analyzed. Initial outputs by MATLAB for Arsenazo III-REE complexes showed multiple locations where the criterion was met. From there, the location with the maximum r-squared value was chosen. After locations where linear correlations were determined, the data points were extracted to Excel to verify goodness of fit using the Excel LINEST function. In the same manner as occurs with UV-Vis spectroscopy, two linear correlations with differing slopes exist, one for concentrations below 2.5 ppm and one for higher concentrations. The slopes, intercepts and goodness of fit parameters for some preliminary data example excitation-emission wavelength location sets where linearity exists are displayed in Table 2 and Table 3. The locations provided in the tables represent the excitation and emission wavelength combination that had the best linear correlation between concentration and fluorescence emission intensity. The slope and intercept in the tables describe the correlation line for the data that exists while the R-squared value describes how well the line fit the data. In the cases displayed in the tables for the samples analyzed, the correlation lines fit the data well with greater than 0.99 R-squared values and low error associated with both the slope and the intercepts predicted for each linear correlation. From these results, it is predicted that fluorescence would be an effective method for REE quantification for analysis of single, well defined REE systems, much like the UV-Vis method.

Table 2: Fluorescence Calibration Correlation for Concentration 0ppm-1.25ppm

Sample	Location (nm) (Excit,Emit)	Slope	Intercept	R^2	±Slope	±intercept
Sm in AABS	(336, 371)	1.2768	0.0095	1	0.0008	0.0095
Nd in AABS	(854,859)	0.2648	-1.5821	0.9994	0.0031	0.0309

Table 3: Fluorescence Calibration Correlation for Concentration 2ppm-10ppm

Sample	Location (nm) (Excit,Emit)	Slope	Intercept	R^2	±Slope	±intercept
Sm in AABS	(386, 689)	-3.3894	117.8009	0.9997	0.0546	1.8458
Nd in AABS	(597, 658)	0.1407	0.3904	0.9966	0.0058	0.2273

Since linear correlations were capable of being determined, it was desired to attempt to also be able to differentiate between REEs in solution. In order to differentiate between REEs, a method to evaluate regions of fluorescence was needed. To determine excitation-emission regions of fluorescence, waterfall plots were created from all the fluorescent datasets using MATLAB. The following figures display top down views of fluorescent maps for various different samples tested. Figure 15 displays the excitation-emission map for the fluorescence of DDH_2O. This is the sample that is used as blank since there is no fluorescence in DDH_2O. Within this figure, three characteristic diagonal patterns are seen. The most prominent is seen where excitation wavelength is equal to the emission wavelength. The fluorescence seen at these locations is a result of Raleigh scattering resulting from either dust or molecules in solution. Second order light of scattered light is seen at the emission wavelength of 440 nm and excitation of 220 nm and proceeds in a diagonal of increasing emission and excitation wavelengths. The Raleigh scattering and second order scattered light are pointed out in Figure 15.

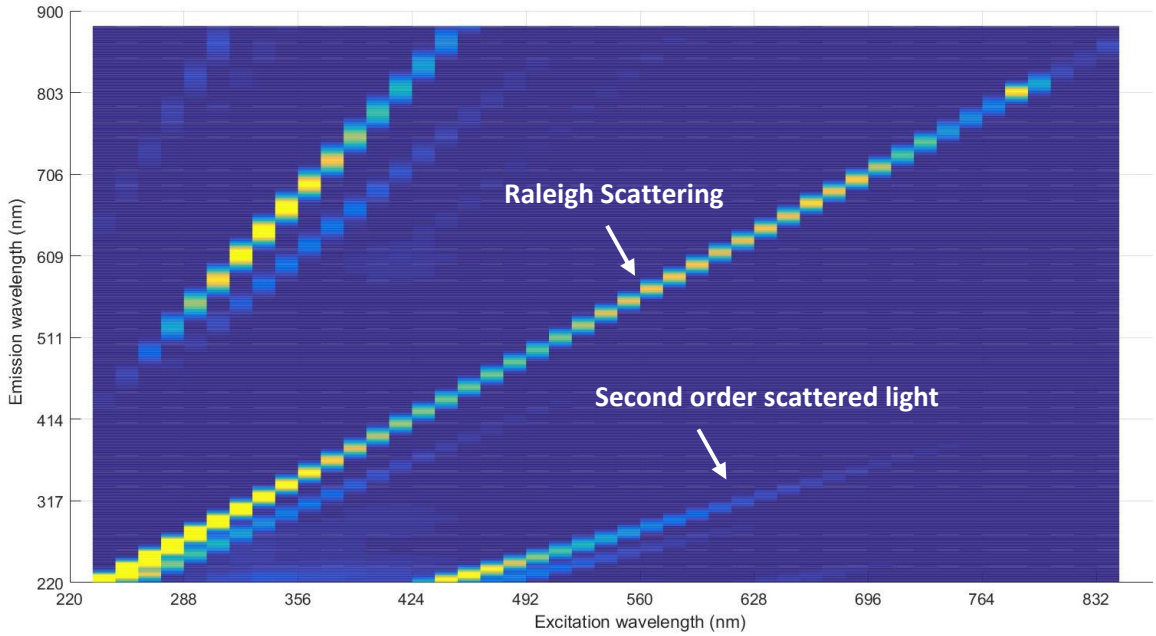


Figure 15: Excitation-emission map of fluorescence for DDH_2O.

The next excitation-emission spectra examined is that of arsenazo III on its own as seen in Figure 16. In comparison to the excitation-emission map of fluorescence for DDH_2O, a new patch of fluorescence appears at excitation wavelength location of 640 nm and emission wavelength at about 550 nm. It is predicted that this fluorescence is a result of arsenazo III molecules due to its aromatic groups.

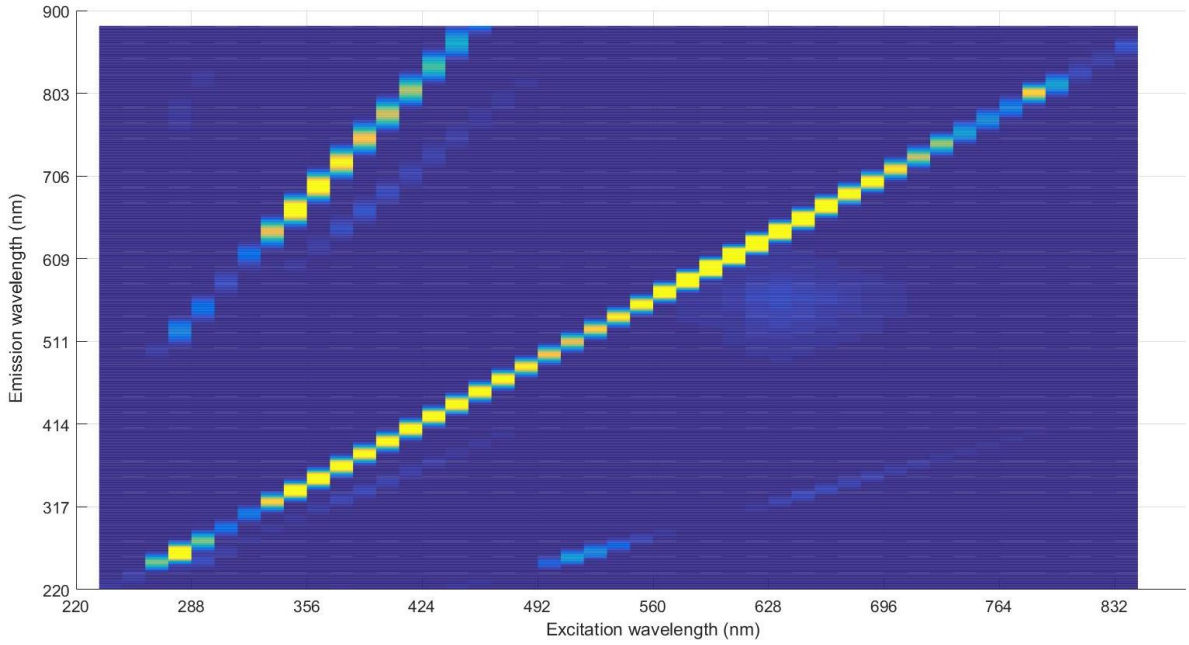


Figure 16: Excitation-emission map of fluorescence for arsenazo III.

To see the excitation-emission map of fluorescence would look like for neodymium-arsenazo III complex, Figure 17 was produced using 5 ppm of neodymium. Within this figure it is seen that the fluorescence patch that was associated with arsenazo III moves closer to the line of fluorescence associated with Raleigh scattering. This shift is predicted to be the result of neodymium-arsenazo III complex.

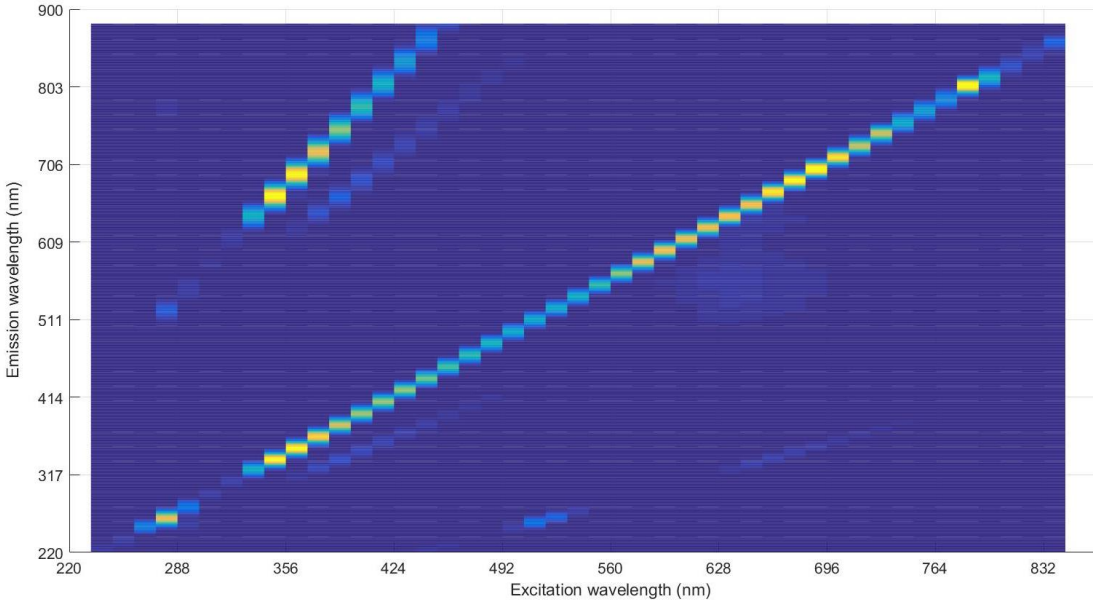


Figure 17: Excitation-emission map of fluorescence for 5 ppm neodymium-arsenazo III complex.

In comparison to Figure 17, Figure 18 shows the complex of 10 ppm of neodymium with arsenazo III. In Figure 18, it is seen that the light patch predicted to be associated with neodymium-arsenazo III complex increases in intensity The increasing intensity that is seen as concentration of neodymium increased from 0 ppm to 5 ppm to 10 ppm across Figures 16, 17, and 18, respectively, indicates that there is a correlation that occurs in this region related to the concentration of neodymium in the solution.

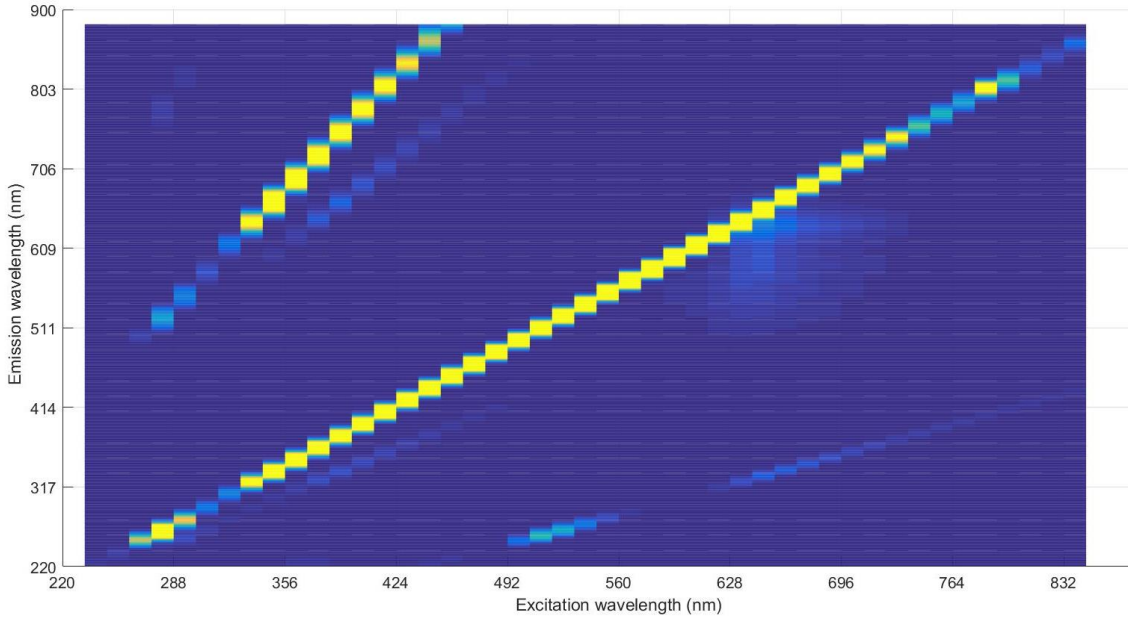


Figure 18: Excitation-emission map of fluorescence for 10 ppm neodymium-arsenazo III complex.

To examine if a noticeable distinction would be shown in the excitation-emission map of fluorescence, a solution of 5ppm neodymium-arsenazo III complex and 5ppm samarium-arsenazo III complex was analyzed. Figure 19 shows the results from the fluorescence scan. In this figure, there is an apparent increase in fluorescence with covering a larger area than the neodymium-arsenazo III complex on its own. This indicates that there is possibly another wavelength set for samarium complex than for the neodymium complex. It is noticed that the patch in Figure 19 shows a much more intense patch at the location of REE-arsenazo III complexes then at the neodymium-arsenazo III complex location. This may be the result of the samarium-arsenazo III complex having a higher intensity than the neodymium-arsenazo III complex. From this it is postulated that differentiation is possible since a nonadditive effect is being displayed between the two complexes. However, it is yet to be determined if they occur at the same wavelength location or at different wavelength locations in a close proximity with signal bleeding occurring between the wavelength locations. In order to determine if the complexes are at different wavelengths with potential signal bleeding occurring, though, maximizing the sensitivity of the spectrofluorophotometer will be needed. Adjusting the resolution to a higher amount by decreasing the monochromator slit size may result in

distinction of the area into multiple isolated patches associated with respective complexes due to subtle structural differences..

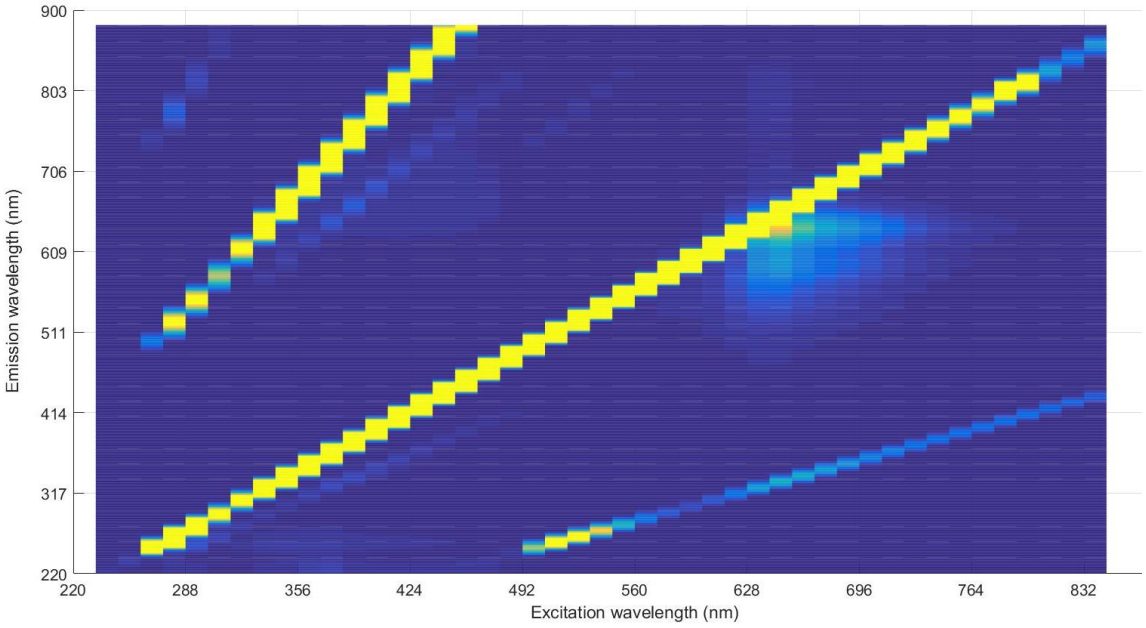


Figure 19: Excitation-emission map of fluorescence for 5 ppm neodymium-arsenazo III complex combined with 5 ppm samarium-arsenazo III complex.

From all these preliminary results obtained, it evident that fluorescence may be a viable technique for REE quantification and differentiation. Within the experiments that were performed, it is evident that linear correlations exist for each REE in various matrices from MATLAB searches through the fluorescent data and is verified through the excitation-emission maps presented. The excitation-emission coordinate changes for which fluorescent emission is present when REE is present versus not present indicates the sensitivity to arsenazo III complex structure changes due to binding with REEs. Since fluorescence is sensitive to vibrational structure at higher resolutions, it is possible that the fluorescence patches seen in Figures 16-19 may have some bleeding of signal occurring between wavelengths in both excitation and emission domains [62]. With the knowledge that the measurements taken in these experiments were taken at slit width of 10 nm, whereas the spectrofluorophotometer is capable of increasing resolution with decreasing slit widths down to 1.5 nm, it is evident that better resolution may allow for the distinction between REEs. This increase in resolution is expected to separate the single patches of fluorescence seen in the

Figures 16-19 into multiple patches of fluorescence with each patch associated with each. Most significantly, it may potentially allow for the distinction between neodymium and samarium since it is apparent that the patch changed in size and intensity as seen in comparison of Figure 18 and Figure 19.

6. Development of C. Necator as a Model Organism for Biosorption of REEs

6.1 Cupriavidus necator as a model biosorbent organism

The organism used in this project is the bacterium *Cupriavidus necator*. Some other names *C. necator* has been referred to are *Alcaligenes eutrophus, Hydrogenomonas eutrophus, Wautersia eutrophus,* and *Ralstonia eutropha*. It is a gram-negative, neutrophilic bacterium. These bacteria have short rod shapes with length of approximately 0.8 by 1.9µm. Optimal growth temperature has been reported to occur at 35°C and a pH of 7.2. *C. necator* are nonobligate predators capable of causing lysis in soil to a variety of gram-positive and gram-negative bacteria, as well as, other nonobligate predators. They are capable of reducing nitrate and do not require organic nitrogen sources besides amino acids. This bacterium has previously been extensively studied for its ability to produce polyhydroxyalkanoates (PHA), a precursor to bioplastics. It has also been shown to be capable of bioprecipitating other metals including palladium, silver, nickel, mercury, copper, lead and zinc. Due to the extensive research this bacterium has undergone in the past, as well as its previous usage in metal accumulation, this organism was an ideal bacterium to work with.

Due to the gram-negative nature, *C. necator* possess an outer membrane containing lipopolysaccharides. They also contain a peptidoglycan layer between the bacterial inner and outer membranes. The extra oxygen functional groups attached to the outer membrane layer provide additional binding sites to the exterior of the bacterium in addition to the nitrogen and oxygen groups present in the peptidoglycan layer that exists in both gram-positive and gram-negative bacteria. In addition, REE binding can occur to proteins inside the cells. Expression of proteins by bacteria change depending on environmental conditions, therefore, environmental conditions are a critical factor to monitor during biosorption processes to obtain replicable results. This is due to the affinity of rare earths to specific functional groups on proteins[63]. In addition, since *C. necator* stores PHA, the various ester groups along this linear polyester are also potential targets for REE binding. As such, since PHA is usually caused by certain deficiency conditions with an excess supply of carbon sources, it is ideal to obtain bacteria at well the same growth phase and at the same environmental conditions to try

and keep PHA products effect constant while also worthwhile investigation on contribution to biosorption abilities.

6.2 Methods

6.2.1 Chemical Speciation Prediction

For effective separation of cells in a DEP device, it is important to ensure that precipitates are limited as much as possible from entering the device, as well as preventing the occurrence of precipitates forming within the device. Formation and accumulation of precipitates may impede flow through the channels on the device rendering the device unusable. Precipitates would also render the methods used to measure REEs invalid as this relies on centrifugation to separate cellular biomass from aqueous media. A prediction of the potential chemical species was performed with the Visual MINTEQ 3.1 software package that is freely available. Concentrations of media components and metal saline solutions were inputted into the system. Thermodynamic sweeps were performed over temperatures from 15 degrees Celsius to 75 degrees Celsius with 10-degree increments at constant pH of 5 were run and over pH from 2 to 10 with 0.5 pH increments at constant temperature of 35 degrees Celsius. The software output includes potential speciation of ions and complexes that can occur based on thermodynamic databases in combination with mass action and mass balance equations. Results for these predictions were used for experimental design and result analysis.

6.2.2 Culture Media Preparation and Growth

C. necator was grown in a media containing 1.2 g/L ammonium sulfate, 0.83 g/L magnesium chloride hexahydrate, 0.097 g/L calcium chloride dihydrate, 10 g/L citric acid anhydrous, 1 g/L yeast extract, 11 g/L PIPES and 1 mL of a trace element solution in 18ΩM H_2O. The trace element solution contains 68 ppm zinc chloride, 67 ppm cupric chloride dihydrate, 64 ppm cobalt chloride, 75.8 ppm manganese chloride, 11.7 ppm sodium molybdate dihydrate, and 31 ppm boric acid. The media was autoclaved for approximately 30 minutes and 125°C to ensure sterile media prior to inoculation. Inoculation was performed under flame from a previous grown and established culture of *C. necator* with 2 mL of inoculate for every 100 mL of media. Incubations were performed at 35ºC and at 130 rpm.

6.2.3 REE Stock Solutions

A europium stock solution was needed to reduce error and provide consistency in all experiments. For all experiments 0.0422 grams of $Eu(NO_3)\cdot 6H_2O$ was added into 250 mL of 18ΩM H_2O. The resulting solution was 378.416 µM and about pH 5.3.

A neodymium stock solution was needed to reduce error and provide consistency in all experiments. For all experiments 0.0548 grams of $Nd(NO_3)\cdot 6H_2O$ was added into 250 mL of 18ΩM H_2O. The resulting solution was 500.057 µM and about pH 5.3.

A samarium stock solution was needed to reduce error and provide consistency in all experiments. For all experiments 0.0556 grams of $Sm(NO_3)\cdot 6H_2O$ was added into 250 mL of 18ΩM H_2O. The resulting solution was 500.371 µM and about pH 5.3.

A saline solution is used for experiments to ensure equilibrium exists inside and outside the cells both sodium and chloride ions. This ensures the hyponatremia and hypernatremia does not occur thereby not killing the cells. The PIPES buffer in the system in the saline solution was used to stabilize pH. A saline solution of 750mM was prepared. To this saline solution, 41.922 mg/L of PIPES was added to a final concentration of 125mM.

Acetic acetate buffered saline has the same function as the PIPES buffered saline except is more advantageous for REE biosorption experiments because it does not produce precipitates in the pH range needed to keep REE in solution. The disadvantage to this saline solution is the complexation that occurs with REEs and the acetate molecules. To prepare this solution, a 0.2M sodium acetate solution is mixed with 0.2M acetic acid. Solution is subsequently diluted with 18ΩM H_2O to 0.1M. A 750mM solution of sodium chloride is made up and adjusted to the desired pH with addition of prepared 0.1M acetic acetate buffer.

6.2.4 Freezer Stock Culture

To ensure that all experiments are performed from the same generation of bacteria and to preventing evolutionary mutations that might occur from many successive transfers, a freezer stock culture was made. This was an added control so future experiments may also be made from the same lineage as the bacteria used in the experiments performed. To do this, a 2 mL cryovial of *C. necator* stock was removed from -80°C freezer and poured into 100mL

sterilized *C. necator* media in a 250mL serum bottle under flame. This serum bottle was incubated at 35º C and at 130 rpm for 12 hours to make an overnight liquid culture. A 50% glycerol solution was made by diluting 100% glycerol in 18ΩM H_2O and autoclaved for sterility. In biosafety cabinet that has been sterilized, 500 µL of 50% glycerol solution is pipetted into each 2mL sterile cryovial. Subsequently, to each cryovial, 500 µL of overnight culture was added. All cryovials were then vortexed for 7 to 10 seconds at 3000 rpm to ensure mixing of culture with glycerol. Freezer boxes with cryovials were stored in a -80°C freezer until use.

6.2.5 Growth Curve Determination

To characterize the growth pattern of *C. necator*, a growth curve was established. From this curve, a specific growth phase was targeted to ensure that all experiments were consistent. UV-Vis spectrometer was first blanked at 590nm with 18ΩM H_2O. All readings were performed with disposable polystyrene cuvettes with 340-800 nm range. Initial absorbance reading were taken for each bottle and indicated as time equals -30 min. From same overnight culture used to make freezer stock, 1 mL of *C. necator* was inoculated into each serum bottle. Time 0 min was the absorbance reading immediately after initial inoculation. One milliliter samples were taken from each serum bottle and tested for absorbance at every hour. Cell counts were also taken every 4 hours from the same one milliliter sample analyzed by spectrometry.

6.2.6 Cell Count Calibration

For determination of the quantity of cells involved in experiments on a per cell basis analysis of experimental samples, a cell calibration curve was developed for optical quantification. Approximately 12 hours prior to experiment, 4 serum bottles filled with 100mL of media were inoculated and incubated under normal growth conditions. After growth occured, two sterile centrifuge bottles were each filled with 2 serum bottles of media plus culture and centrifuged for 10 minutes at 10k RCF at 27 degrees Celsius to remove cells from media. The supernatant was then poured off and cells were suspended in 15mL of PIPES buffered saline (PBS). Both centrifuge bottles were then combined into a 50mL centrifuge tube. The UV-Vis spectrometer was set at 590nm and blanked with PBS. All readings are performed with disposable cuvettes. Five cuvettes were filled with 2 mL aliquots of sample

and readings taken for each cuvette. Four serial dilutions were created with 1mL of sample from each cuvette and 1 mL of PBS and absorbance recorded. Cell counts of initial set of cuvettes were taken and a correlation was developed between concentration and absorbance.

6.2.7 Biosorption Procedure

For all biosorption experiments, the following general procedure followed for all experiments is displayed in Appendix B.1.

6.2.8 Biomass Concentration Panel

This subset of experiments varied the amount of concentration to ensure enough bacterial mass required for experiments was used for biosorption to occur without oversaturating the system with bacteria. For this process, concentration of approximately 80 micromolar of Europium was selected. The biosorption process was held at pH 5 and temperature of 35ºC. Concentrations were manipulated by modifying the amount of serum bottles of media with bacteria to .5, 1 and 2 serum bottles per 50mL centrifuge tube to compare with the 1.5 serum bottles per 50 mL centrifuge tubes used in rest of experiments.

6.2.9 Concentration Effect Panel

The concentration effect panel was used to determine the average absorption capacity per billion cells for each REE being investigated. Prior to each experiment, REE solutions were prepared for biosorption. The concentration panel consisted of approximately 8%, 12%, 16%, 20%, 40%, and 60% of stock solution concentrations. The REE solutions were made by adding a set volume in milliliters of REE stock solution to a 50 mL Falcon tube. The Falcon tube was then filled to 50mL with AABS solution of pH 5. Of the 50 mL of solution, 15 mL was saved for analysis while 35 mL was used to suspend bacteria pellet for biosorption to occur. These experiments were performed at 35ºC.

6.2.10 Temperature Effect Panel

These experiments were used to determine the effect on biosorption by temperature. REE solutions were made up the same way as for the concentration pellet with the concentration being held at 16% stock concentration. The biosorption process was performed at 15ºC, 35ºC, 55ºC, and 75ºC.

6.2.11 pH Effect Panel

These experiments were used to determine the effect on biosorption by pH. REE solutions were made up the same way as for the concentration pellet with the concentration being held at 16% stock concentration. The biosorption process was performed at pH 4.5, 5, 5.5, and 6. Temperature was held constant at 35ºC.

6.2.12 REE Selectivity Panel

These experiments were used to evaluate if there is a preference of the bacteria for one REE over another. For these experiments, two REEs are tested against each other. The concentrations were mixed such that one REE was one third of the solution and the other was two thirds concentration and vice versa, at full concentration of each individually, and an equal concentration of each REE. The pH of the REEs solutions were at pH of 5 and temperature was set at 35ºC during the biosorption process.

6.2.13 Nitric Acid Digestion:

The digestion step in the process was required to free all the REE from the biomass and release them into solution for analysis. After biosorption and washing stages, *C. necator* pellet was transferred to 15 mL balsh tubes filled with 5 mL of Trace Element Grade Nitric Acid and placed in a heating block. The heating block was brought to 30ºC and held for 1 hour. Temperature was then ramped over approximately 60 minutes and held at 70ºC for 60 minutes. Ramping of temperature was again performed for 30 minutes to 175ºC and held for 90 minutes which results in no solution remaining. Samples are then saved for analysis.

6.2.14 Sample Analysis Prep

Prior to analysis, all cells and precipitates are required to be absent from samples to prevent build up in equipment. All samples were vacuum filtered over a 0.2µm pore size membrane filter. The filtrate was spiked with nitric acid to create a 2% nitric acid matrix. The membrane filters were baked in a furnace at 300°C for 30 minutes and then 500°C for 1 hour [64, 65]. After baking, the residual was sonicated in 5 mL of 2% trace metal nitric acid. Samples was sonicated for 8 hours at 80°C in a sonicating bath with frequency of 42kHz, ultrasonic power of 480 W and heating power of 300W. The residual was crushed in the balsh

tube in the nitric acid then sonicated using a sonicating probe at 65% power for 5 minutes. Samples were then filtered to remove ash and aqueous filtrate saved for analysis.

6.2.15 MP-AES Analysis

MP-AES system was warmed up for about 30 minutes. Samples were arranged in auto sampler in increasing order of predicted concentration. Calibration standards of REE standard solutions were prepared to 0.625 ppm, 1.25 ppm, 2.5 ppm, 5 ppm, and 10 ppm to calibrate the equipment. A QC check solution was prepared from REE standard solution to 2 ppm. Within setup of procedure, system was set to run QC check after every 12 samples for drift and to recalibrate if not within 10% error of QC concentration. Sample uptake time is 70 seconds. All samples were run in a 2% trace metal nitric acid matrix. Some duplicate samples and spiked samples were included for verification of MP-AES reproducibility of readings.

6.2.16 Metal Localization:

Metal localization allows for the determination of metal location within the cells that accumulation occurs during biosorption. After the washing stages in the biosorption process, instead of performing nitric acid digestion, cells were pelleted and subsequently fixed in fixing agent composed of 2% Paraformaldehyde, 2% Glutaraldehyde, 0.1 M cacodylate buffer at pH of 7.2 for approximately 2 hours. Microwave fixation was then employed for 2 minutes and 30 seconds at 750 W. Samples were then rinsed and set for 10 minutes in phosphate buffer 3 times. After rinsing and setting in DI water for 10 minutes 3 times, the samples were freeze dried overnight. Cell samples were rinsed with 0.1 M 3 times at 10 min each. Dehydration of samples was done with 30%, 50%, 70%, 95% and 100% ethanol at 10-minute intervals. Dehydration was then performed with 100% ethanol 3 times. A 1:1 propylene oxide (100%)/ethanol solution was introduced and let set for 10 minutes. Removal of propylene oxide/ethanol solution and infiltration was done using a 1:1 propylene oxide (100%)/ SPURRS solution overnight. Replace the propylene oxide/ SPURRS solution with 100% SPURRS and again let set overnight. Bake in oven overnight. Thick sections of samples were taken at 1000nm to verify presence of specimen. Samples were then sectioned using microtome with sections of 70 nm. Sections were visualized using Phillips TEM.

6.2.17 Whole Mounting:

Whole mounting allows the ability to see the morphological effects biosorption and environmental factors have on bacterial cells. On a TEM grid, 5 µL of sample in saline was added. It was let to sit for about 5 minutes then with filter paper the liquid phase was drawn off. Three times a drop of DI water was added onto grid followed by waiting for approximately 30 seconds then DI water was drawn off. Then a drop of uranyl acetate was added, let sit for 5 minutes then drawn off. The grid was then placed under a heat lamp to dry for approximately 10-15 minutes.

6.3 Results and Discussion

6.3.1 Speciation of REEs vs. pH

Rare earth elements are known to have a variation in speciation as pH is varied. Visual MINTEQ ver. 3.1 was used to predict the speciation of europium, neodymium and samarium in the acetic acetate buffered saline that the biosorption experiments were performed. This provides knowledge about speciation of REEs during exposure to *C. necator*. For all REEs, speciation was predicted from pH 2 to 8 with a step size of pH 0.5. The pH of interest in these experiments is the value where precipitates start to form. Precipitation formation is expected to result in decrease biosorption performance. In addition, precipitates present may clog the microdevice, therefore should be minimized.

Within the pH spectrum studied, europium occurs in 4 distinct species at measurable concentrations: Eu^{3+}, $EuCl^{2+}$, Eu-$Acetate^{2+}$, and $EuOH^{2+}$. At pH 2, as shown in Figure 20, the europium species Eu^{3+} and $EuCl^{2+}$ exist. With increasing pH, Eu^{3+} slowly decreases in concentration until pH 5 where it starts to decrease in concentration at a much higher level. The species $EuCl^{2+}$ slowly increases with increasing pH to a maximum at approximately pH 4 where it slowly decreases to pH 6.5. At about pH 6.5, the species concentration decreases at a higher level. Complexation between europium and acetate starts to form Eu-$Acetate^{2+}$ begins to form and increases as pH increases until around pH 6.5 where it reaches a maximum. From pH 6.5 the complex slowly decreases in concentration percent. The europium hydroxide precipitate, $EuOH^{2+}$, begins to form at around pH 4.5. Europium hydroxide exponentially

increases in concentration percent as pH increases with significant amounts occurring at approximately pH 6. At this pH, a significant amount of europium begins to be found as a precipitate which is much less likely to be biosorbed by any organism through passive means. This results in a majority of europium remaining in aqueous phase after the biosorption process instead of being bound to the solid biomass. From these results, the biosorption process should ideally remain below pH 6 to ensure maximum biosorption of europium and minimization of precipitate formation.

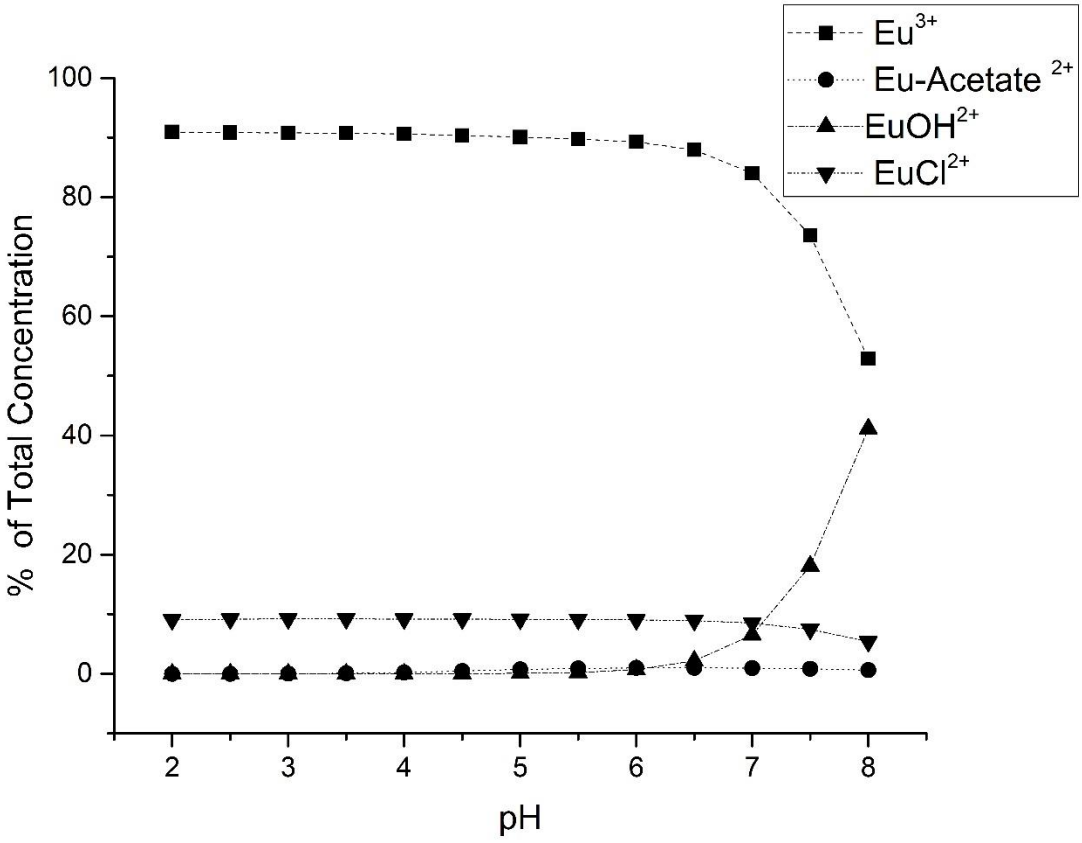


Figure 20: Speciation of europium as pH increases in acetic acetate buffered saline at 35°C.

Neodymium exists as four distinct species in the pH range from 2 to 8. Only the Nd^{3+} ion form is present from pH 2 up to pH 3. From Figure 21, it is shown that at pH 3 formation of a complex with the acetate molecules begins to occur resulting in the species Nd-$Acetate^{2+}$. Nd^{3+} slowly decreases in concentration percent until approximate pH 6.5 where it decreases in concentration percent at a much higher level. The Nd-$Acetate^{2+}$ complex slowly increases

as pH increases until it reaches a maximum at approximate pH 6.5. At pH 4.5, the hydroxide precipitate, $NdOH^{2+}$, begins formation at trace concentrations and slowly increases in concentration as concentration increases. Biosorption of neodymium is not expected to be affected until around pH 6.5 where the hydroxide hydrolysis product begins to increase in concentration at a higher level. When pH is at around pH 7.5, a second precipitate formation begins to occur in the form of species $Nd_2(OH)_2^{4+}$. From these results, it is inferred that biosorption processes must remain below pH 6.5 for neodymium to minimize precipitate formations for maximum biosorption potential and prevent blockage in microdevice.

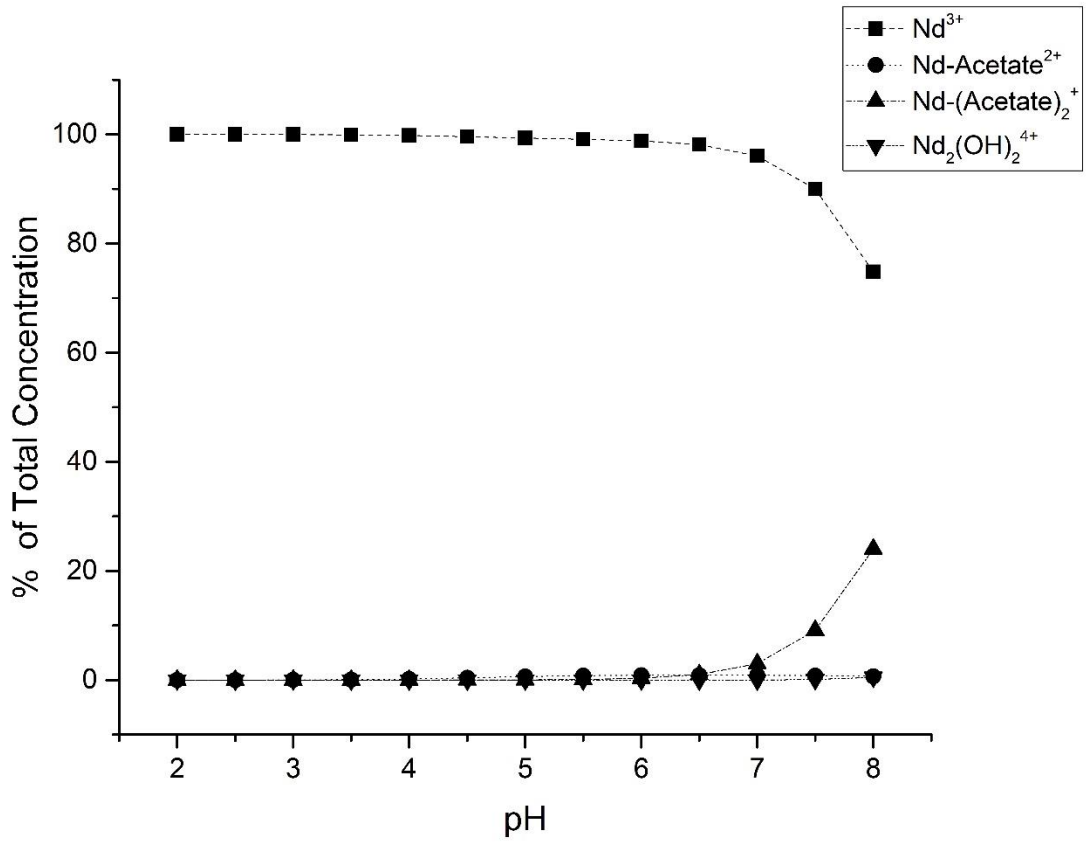


Figure 21: Speciation of neodymium as pH increases in acetic acetate buffered saline at 35°C.

Samarium exists in three species in the pH range of 2 to 8: Sm^{3+}, Sm-$Acetate^{2+}$ and $SmOH^{2+}$. At pH 2, only Sm^{3+} is present. As pH is increased, the Sm-$Acetate^{2+}$ forms and increases in concentration as Sm^{3+} percent concentration decreases. Sm-$Acetate^{2+}$ reaches a

maximum at pH 6.5. Approximately at pH 4.5, the precipitate $SmOH^{2+}$ begins to form. At pH 6, the precipitate formation appears to become more favorable and increases in concentration at higher level. At this same pH, Sm^{3+} species becomes much less favorable and lowers in concentration at about the same level as precipitate increases. From these results, it is apparent that pH must be maintained below pH 6 to reduce precipitate formation to avoid clogging device and maximize biosorption potential. The effect of pH on speciation of samarium is shown in Figure 22.

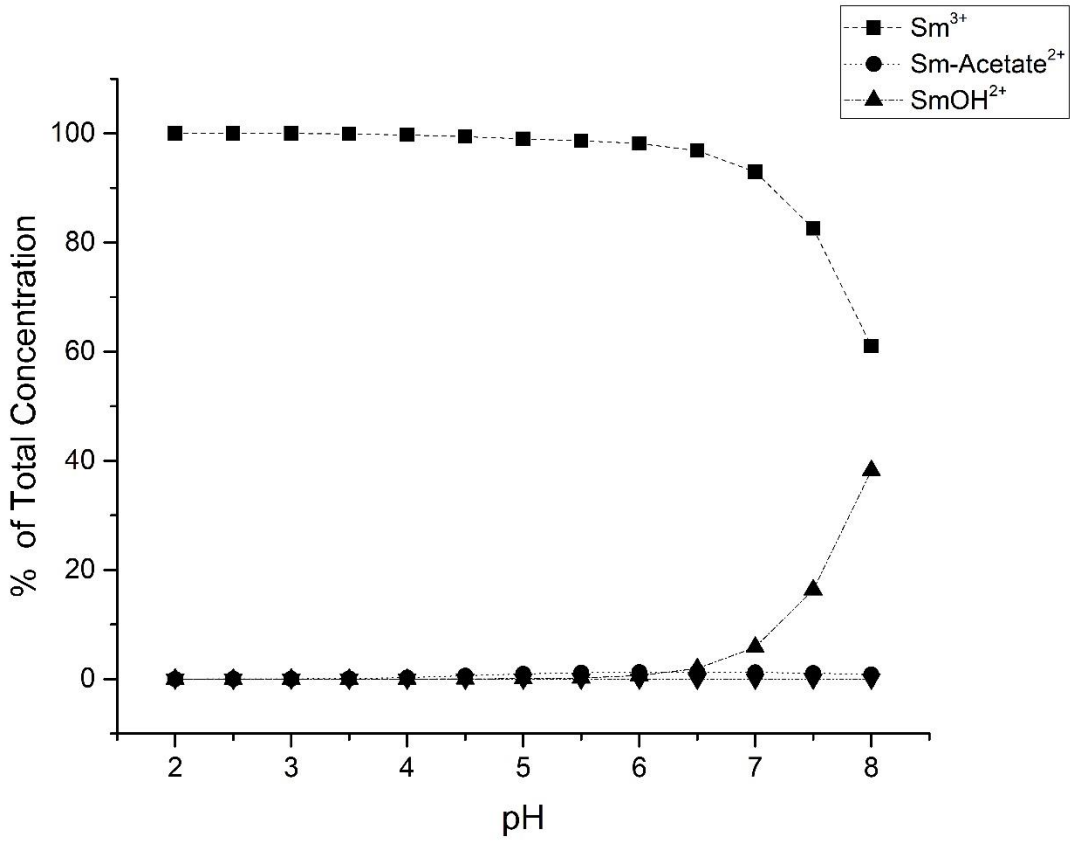


Figure 22: Speciation of samarium as pH increases in acetic acetate buffered saline at 35°C.

6.3.2 Determination of C. necator Growth Phase

Under its optimum growth conditions and in this medium, *C. necator* reaches stationary phase at approximately 9 to 10 hours (see Figure 23). This plot is used to assist in determination of the time required to reach a required growth phase. Figure 23 shows that this bacterium experiences lag phase for approximately the first 2 hours after inoculation. Between the times of 2 hours to about 9 hours the bacteria are in their exponential growth phase. From 9 to about 15 hours the bacteria appear to be transitioning into the stationary phase. Beyond 15 hours the bacteria may be either still be in stationary phase or transitioning into death phase. The location of death phase is not determinable using optical method due to influence of necromass (deceased biomass) contributing to absorbance readings. All the following experiments were performed at 12 hours post inoculation during the early stationary phase period to keep a consistent growth period throughout.

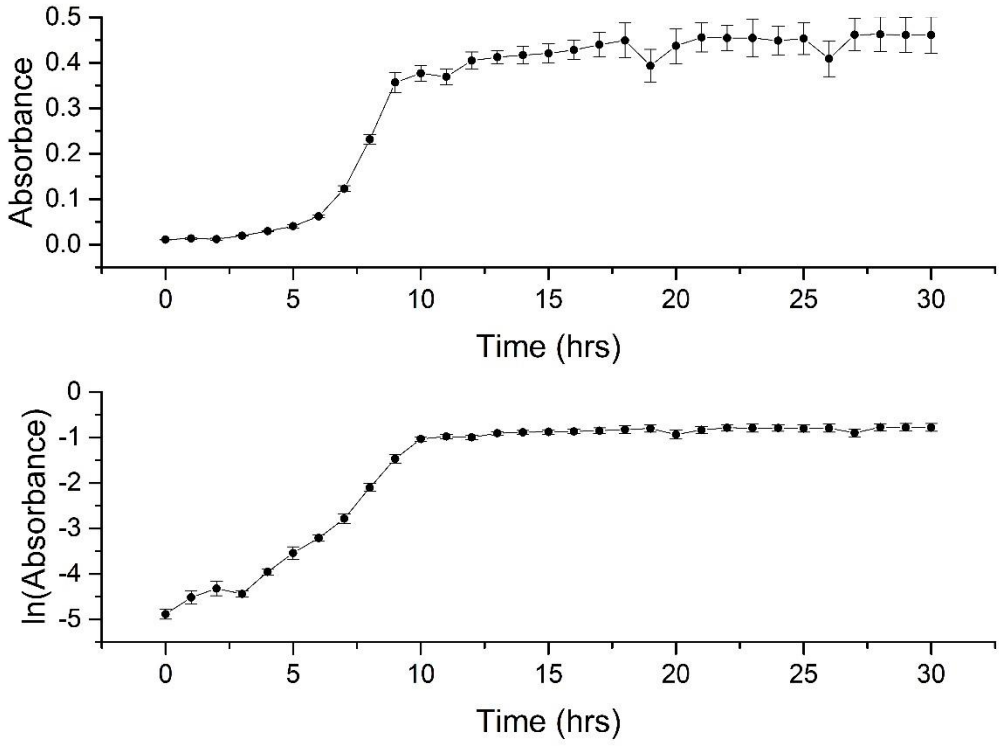


Figure 23: Plot of growth of *C. necator* described in absorbance (top) and the natural log of absorbance (bottom) against time for growth phase determination.

6.3.3 Cell Calibration

A calibration curve was developed to correlate the concentration of cells for each sample to the turbidity measured from spectrophotometry. A linear relationship is found between the natural log of cell concentration and the absorbance. In conjunction with measurements from a UV-Vis spectrometer, this correlation provides a rapid method for quantification of cells per volume of solution from absorbance readings at 590 nm.

6.3.4 Concentration Dependence

Experiments were performed to determine the loading capacity of REEs on *C. necator* biomass. Growth temperature for these experiments were chosen to be at 35°C because it is the bacteria's optimum growth temperature. A pH of 5 was used to prevent hydrolysis reactions with rare earth elements from occurring while still maintaining within viable range for the bacteria. The bacteria were harvested during early stationary phase. All quantities of REEs determined in each aqueous solution resulted from MP-AES analysis. Three main groups of samples resulted from these experiments: initial aqueous, aqueous equilibrium, solid phase biomass samples. Initial aqueous samples quantify the amount of rare earth elements a quantity of biomass was exposed to. Aqueous equilibrium sample set consists of the quantity of rare earths left in aqueous solution after 1-hour exposure to bacteria plus the quantity of rare earths removed from the cells during the washing steps after biosorption was performed. The biosorbed samples consists of the rare earths that were biosorbed by the bacterial cells that remained attached to the cells after washing occurred. It was noticed when performing the washes of the pellets after biosorption, that the pellets were much more difficult to create as the concentration of rare earth elements the bacteria were exposed to increased. It is postulated that this may result from a change in electrostatics from increased adsorption causing the cells to repel each other.

Biosorption is the process of absorbing molecules into the solid biomass phase from an aqueous phase. Figure 24 shows the phase associated with each europium concentration as the initial exposure quantity of europium is increased. From the figure, it is seen that the amount of europium that is in the biomass increases as europium concentration is increased to a limit between 20 and 30 μmols per billion cells. The limit is the sorption capacity of europium by *C. necator*. As the sorption capacity was approached, the quantity of europium

was seen to continuously increase in aqueous phase. This is due to saturation of *C. necator* biomass to europium. Prior to the sorption capacity the biomass phase contained a majority of the europium indicating that there is a desire by the bacteria to accumulate this metal. Figure 24 also displays there is also a portion of europium not absorbed into the solid phase.

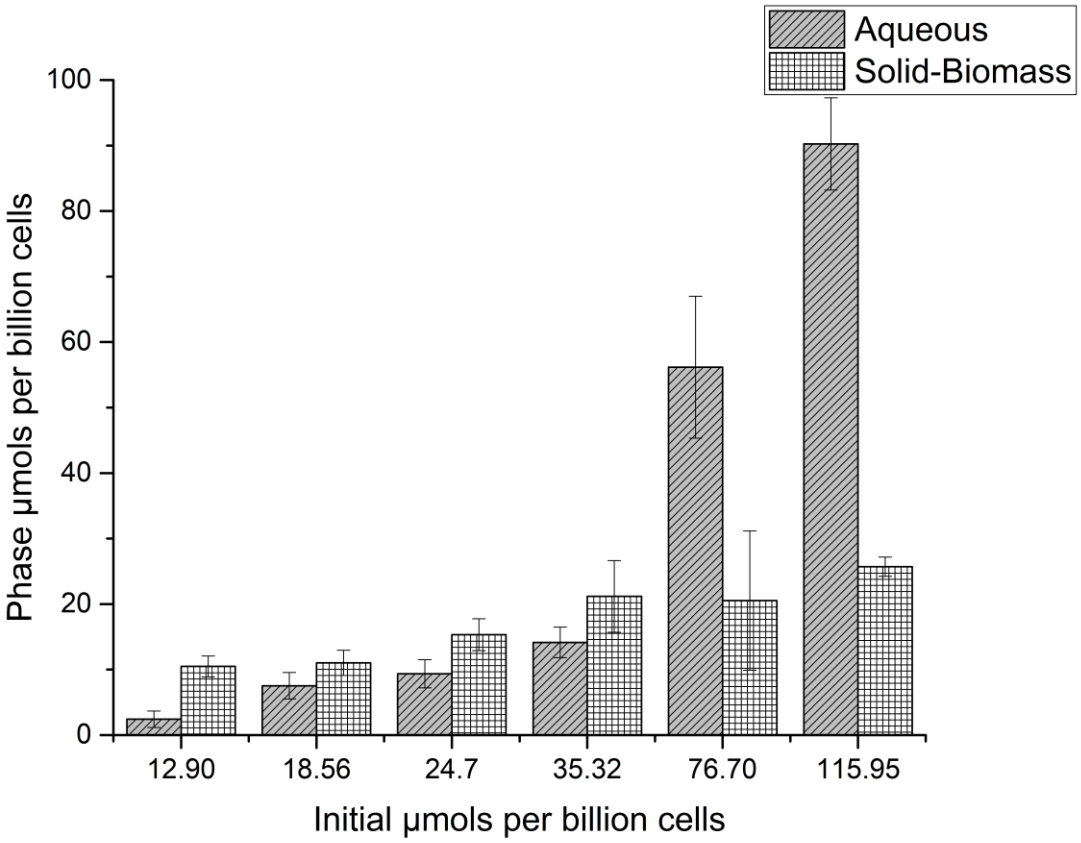


Figure 24: Phase quantification of europium biosorption by C. necator after 1-hour exposure at 35°C and pH 5.

Biosorption of neodymium by *C. necator* is shown in Figure 25 at varying concentrations neodymium. Similar to the europium studies, neodymium concentration in the biomass phase increases as concentration of neodymium increases until it reaches *C. necator*'s sorption capacity for neodymium. From Figure 25, the limit seems to be about 20 µmols per billion cells. As *C. necator* approaches its limit for neodymium sorption, a greater quantity of neodymium remains in the aqueous phase.

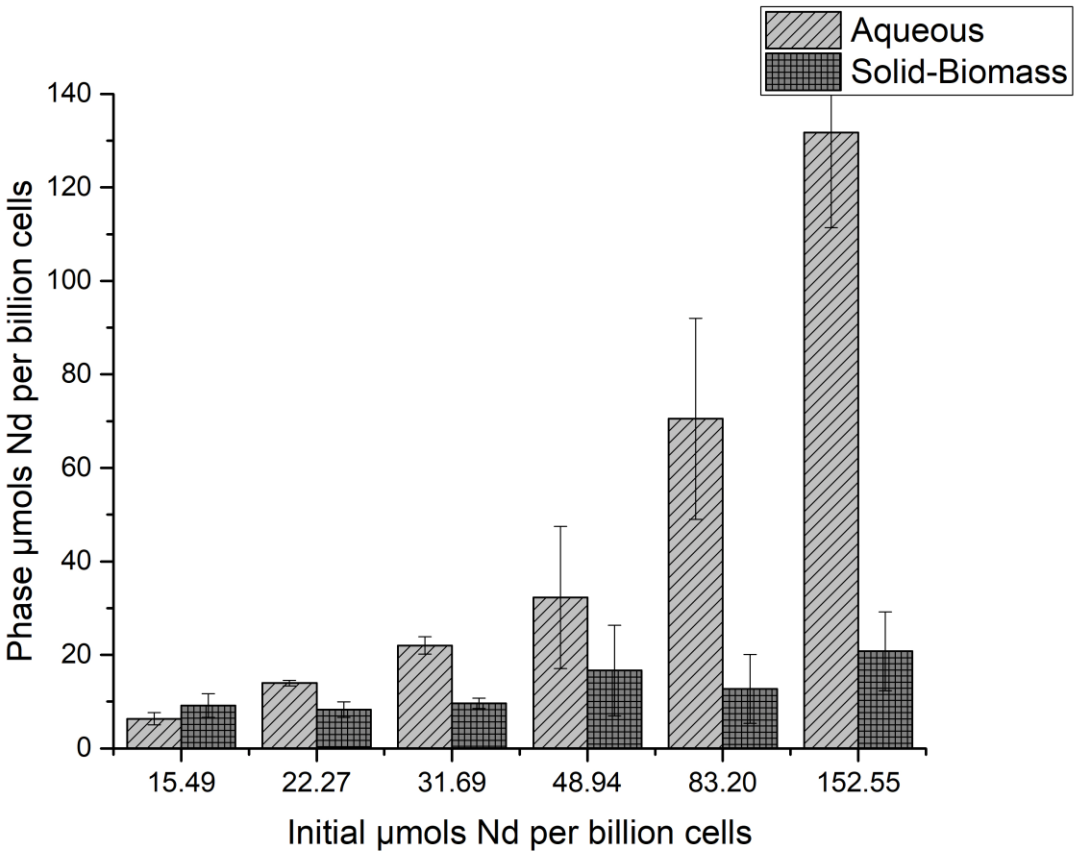


Figure 25: Phase quantification of neodymium biosorption by C. necator after 1-hour exposure at 35°C and pH 5.

The results for biosorption of samarium are similar to both europium and neodymium as seen in Figure 26. From the figure, it appears that *C. necator's* maximum capacity for samarium is approximately 50 µmols per billion cells, which is higher than both europium and neodymium. Noticeably, while the biomass phase contains the greater amount of europium and neodymium until maximum capacity is reached, it is not so with samarium. At comparable concentrations, the aqueous phase contains approximately the same to most of the total exposed samarium, even before maximum capacity is reached.

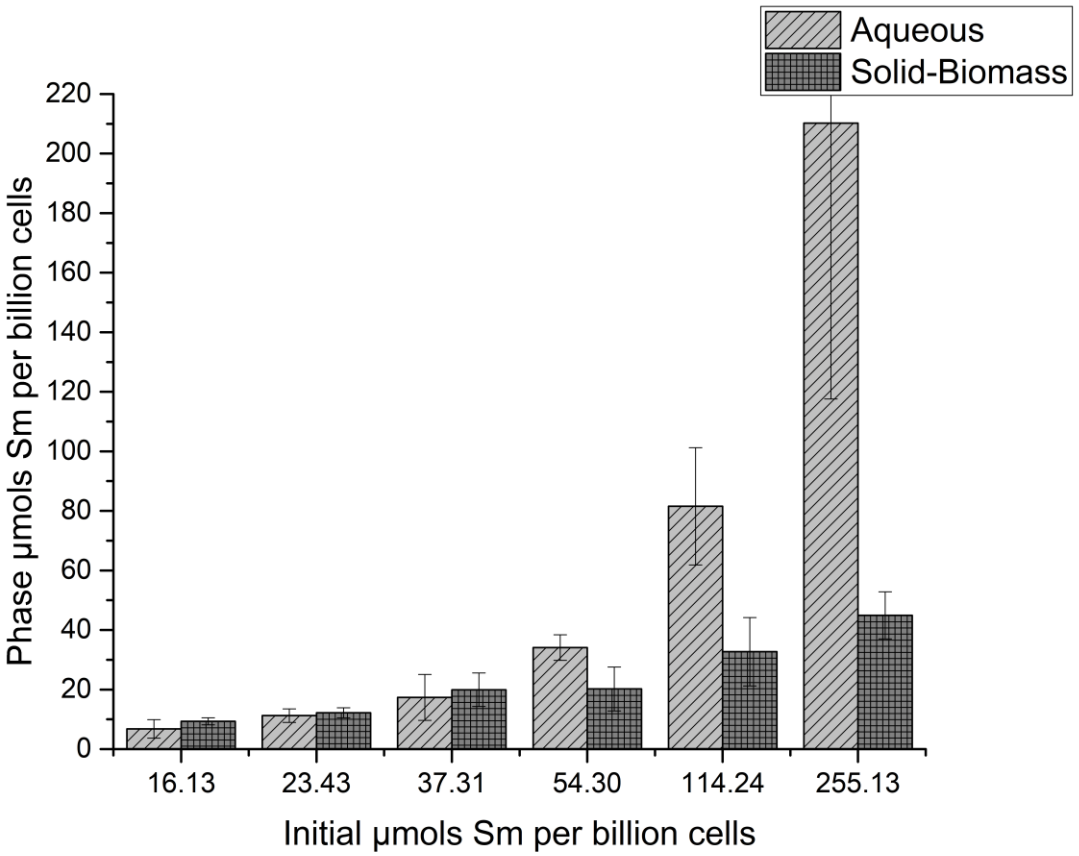


Figure 26: Phase quantification of Samarium biosorption by C. necator after 1-hour exposure at 35°C and pH 5.

From these concentration experiments, the capability of *C. necator* to accumulate rare earth elements is proven. In addition, these experiments provided a general idea about how much of each rare earth element this bacterium is capable of accumulating. With this knowledge assumptions can be made in design of the micro device for which this bacterium is intended to be used to verify capabilities. In addition, these results assisted in selection of the control concentration of 80 µM of each REE that was used in the testing of effects on biosorption by other environmental parameters. This concentration was selected to ensure that the testing systems were never oversaturated with biomass while still having the ability to see an increase or decrease in accumulation.

Since throughout literature isotherms are often used to describe the biosorption capabilities of organisms for various solutes, three isotherms models were compared to determine which isotherm has best fit: Langmuir, Freundlich and Sips. Fitting these isotherms

allows for an understanding of how the biosorption process proceeds between the aqueous and biomass phases with variation of REE concentration. The equations for the isotherms were fit using nonlinear regression. The values for the fitting parameters for the Langmuir, Freundlich, and Sips are displayed in Table 4 for europium, neodymium and samarium. The "C_{smax}" terms for both Langmuir and Sips describe the absorption capacity of *C. necator* for each rare earth element. The "b" and "k" terms in Langmuir and Freundlich provide a measure of the affinity of the biomass for each REE. Additionally, the "n_s" term in Sips describes the heterogeneity level of the system. Since the heterogeneity factor for the three rare earth elements are not the same, the system heterogeneity may stem from either the rare earth elements on their own or from a combination of the rare earth elements with C. necator. It is less likely the heterogeneity would be due solely to *C. necator* itself since the "n_s" values are not similar. The Freundlich constants are empirical constants that provide little information physically about the system.

Table 4 Isotherm Fitting Parameters

Element	Langmuir		Freundlich		Sips		
	C_{smax}	b	K_f	n_f	C_{smax}	K_s	n
Europium	18.80	0.58	11.75	0.11	18.83	0.59	1.01
Neodymium	8.73	0.83	4.83	0.15	26.54	0.21	4.76
Samarium	48.10	0.03	5.27	0.39	96.02	0.04	1.83

A comparison of the Langmuir, Freundlich and Sips isotherms are shown in Figure 27 for europium. From this figure, it appears that all three isotherms fit the data well to some extent. The scatter and deviations from the isotherms can be attributed to variability being caused by living cells. Noticeably, the Langmuir and Sips isotherms overlap each other. This overlap would result from the "n_s" term in the Sips isotherm being close to unity as is the case (see Table 4). As such, it is possible that the system is displaying nearly homogenous surface binding characteristics (uniform energy distribution) with regard to biosorption of europium by C. necator and both isotherms will give comparable results. Both Langmuir and Sips Isotherms approximate the maximum absorption to be approximately 19 µmols per billion cells (see Table 4). The initial steepness of the curve in Figure 27 indicates that there is

relatively high affinity by *C. necator* for europium and is confirmed by the relatively low values for the "b" Langmuir constant and the "K_s: Sips constant shown in Table 4.

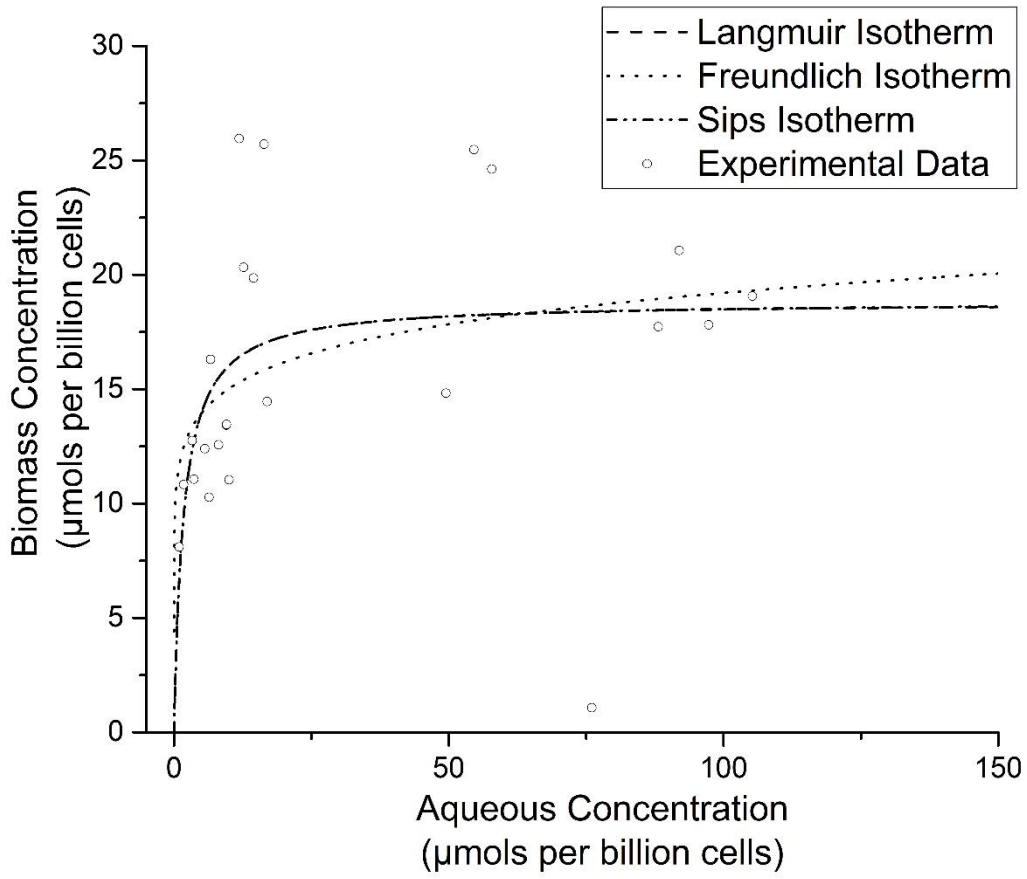


Figure 27: Isotherm comparison for europium biosorption by C. necator at 35°C and pH 5.

Since throughout literature the linearized version of the Langmuir is often utilized, the linearized Langmuir isotherm was also investigated and is shown in Figure 28. However, the parameters obtained from the linearized Langmuir fit is often criticized due to error distribution changes and inconsistency in parameters depending on the way the isotherm is linearized. Keeping in mind these pitfalls, it is noticeable from the R-Square and Pearson's r values that Langmuir fits the experimental data well. Also, there is very little predicted error with the slope and intercept which are theoretically related to maximum adsorption capacity and the affinity of *C. necator* for europium. The agreement between the linearized Langmuir

isotherm and the nonlinear Langmuir fit that was performed appears to indicate that the linearized Langmuir isotherm may be substituted for engineering purposes with little difference in error.

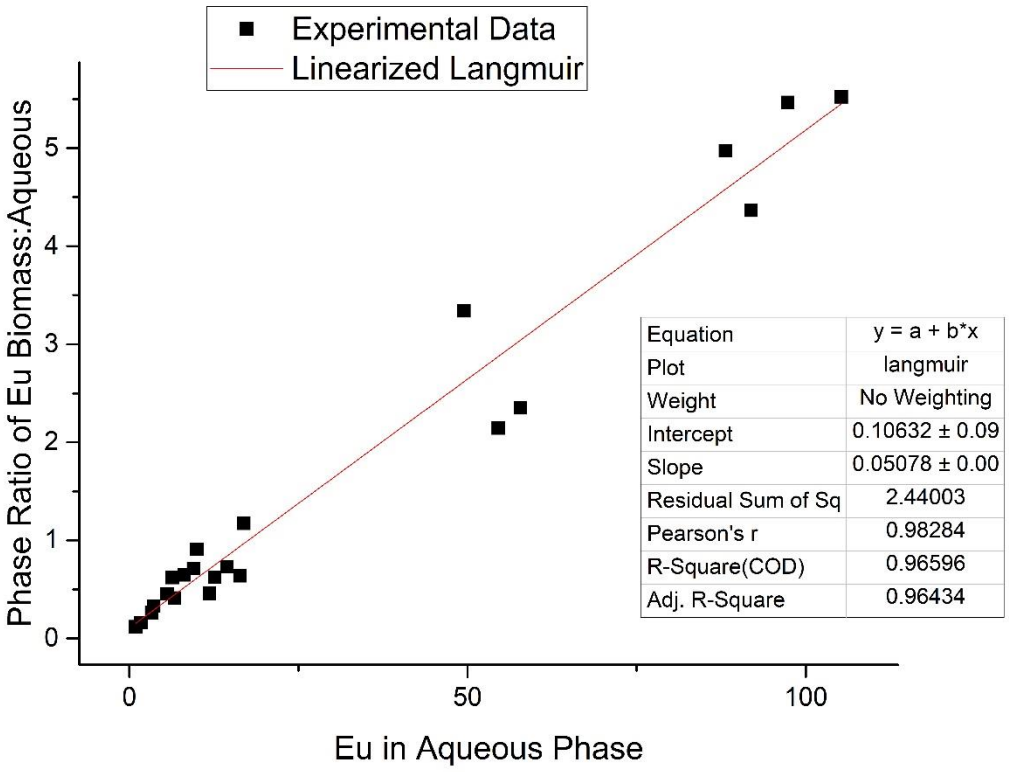


Figure 28: Linearized Langmuir Isotherm for biosorption of europium by C. necator.

Similar to the europium data, there is a lot of variation that appears to be present in the neodymium data set. Like europium, all three of the isotherms still seem to fit generally the experimental data obtained for neodymium (see Figure 29). Unlike europium, however, the Sips isotherm for neodymium overlaps the Freundlich isotherm rather than the Langmuir isotherm. Evaluation of the Sips parameter "n_s" (see Table 4) displays that this system deviates from homogeneity to some extent. Sips and Freundlich begin to deviate at higher concentrations, possibly due to the limitation of continuing increase in adsorbed amount with increasing concentration in Freundlich. There is also a difference between the maximum absorption capacities predicted by Langmuir and by Sips. The Langmuir isotherm predicts a maximum absorption of approximately 9 µmols per billion cells. The reason for the difference may be due to the assumptions from which Langmuir was derived. Langmuir is based on a

homogeneous monolayer coverage in which Sips exponential factor "n_s" predicts is not true. In practice, due to the relatively low affinity that occurs and is seen by the slope at higher concentrations, Langmuir's adsorption capacity would be true for lower concentration samples. However, in samples with relatively higher concentration, the "true" maximum is more likely obtained as the Sips maximum absorbance capacity of approximately 27 µmols per billion cells demonstrated in Figure 25. This may be a result of a significantly decreased affinity as seen by the decreasing slope of the isotherms as concentration increases.

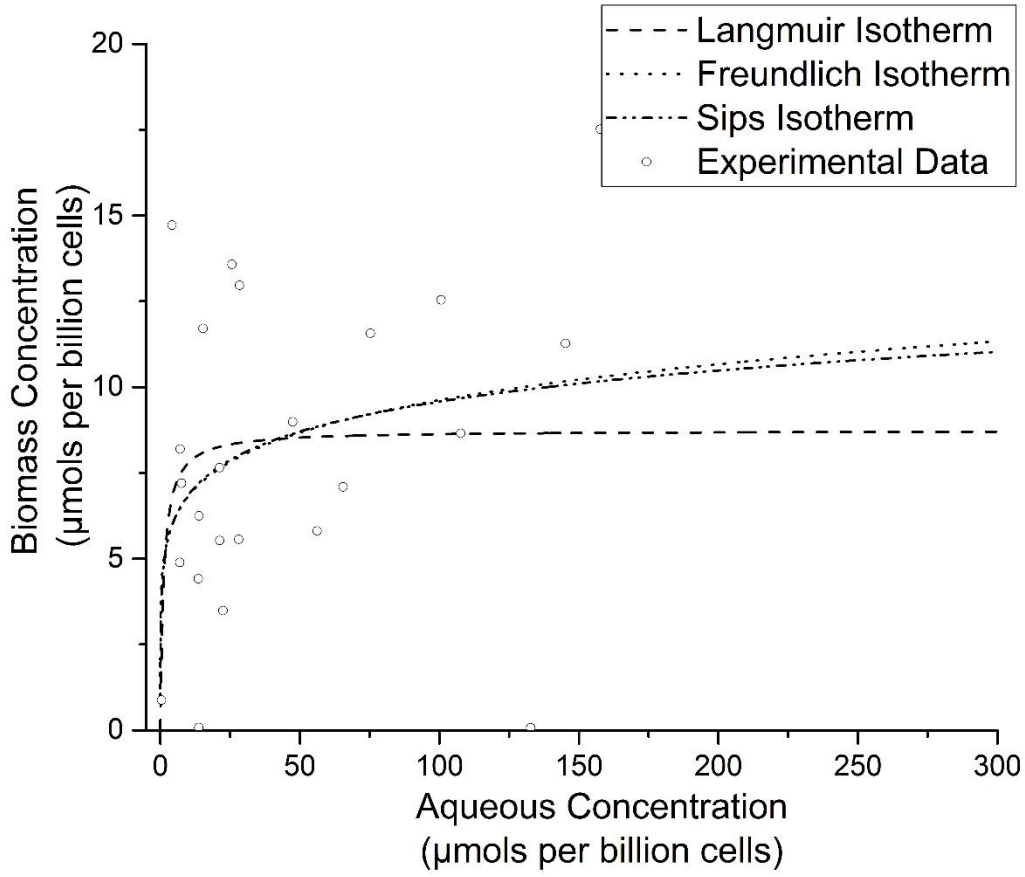


Figure 29: Isotherm comparison for neodymium biosorption by C. necator at 35°C and pH 5.

The linearized Langmuir isotherm was also fitted to neodymium and is shown in Figure 30. Unlike the linearized Langmuir for europium, neodymium is not fitted as well.

This conclusion is the result of the lower R-squared and Pearson's r values describing the linearized Langmuir fit. This follows the findings in Figure 29 in which Langmuir isotherm did not fit the data as well as the Freundlich indicating possible deviations from the Langmuir assumptions.

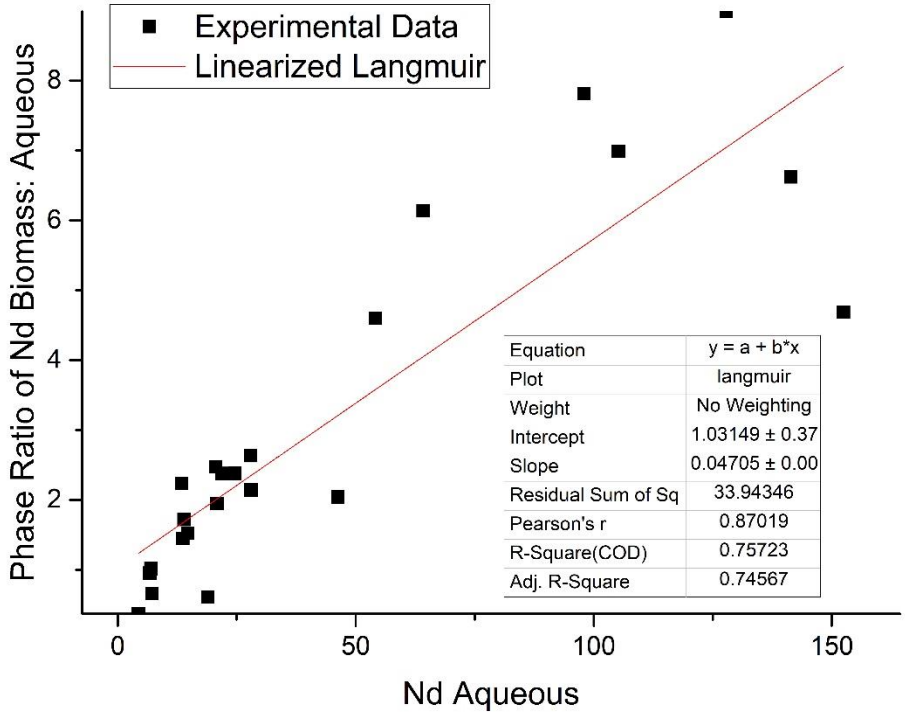


Figure 30: Linearized Langmuir Isotherm for biosorption of neodymium by C. necator.

Like europium and neodymium, all three isotherms for samarium appear to fit the data well. Conversely to europium and neodymium cases, the Sips isotherm does not overlap either isotherm when defining samarium. Instead, as seen in Figure 4.36, Sips isotherm takes on more of an average of Langmuir and Freundlich. As seen by the isotherm plots in Figure 31 as well as demonstrated in Table 4, *C. necator* has a higher affinity for samarium than it does for europium and neodymium. In addition, the maximum absorption is higher for samarium than europium and neodymium. The differences in maximum absorption between Langmuir and Sips may be for the same reasons as seen with neodymium with its roots possibly in Langmuir's assumptions.

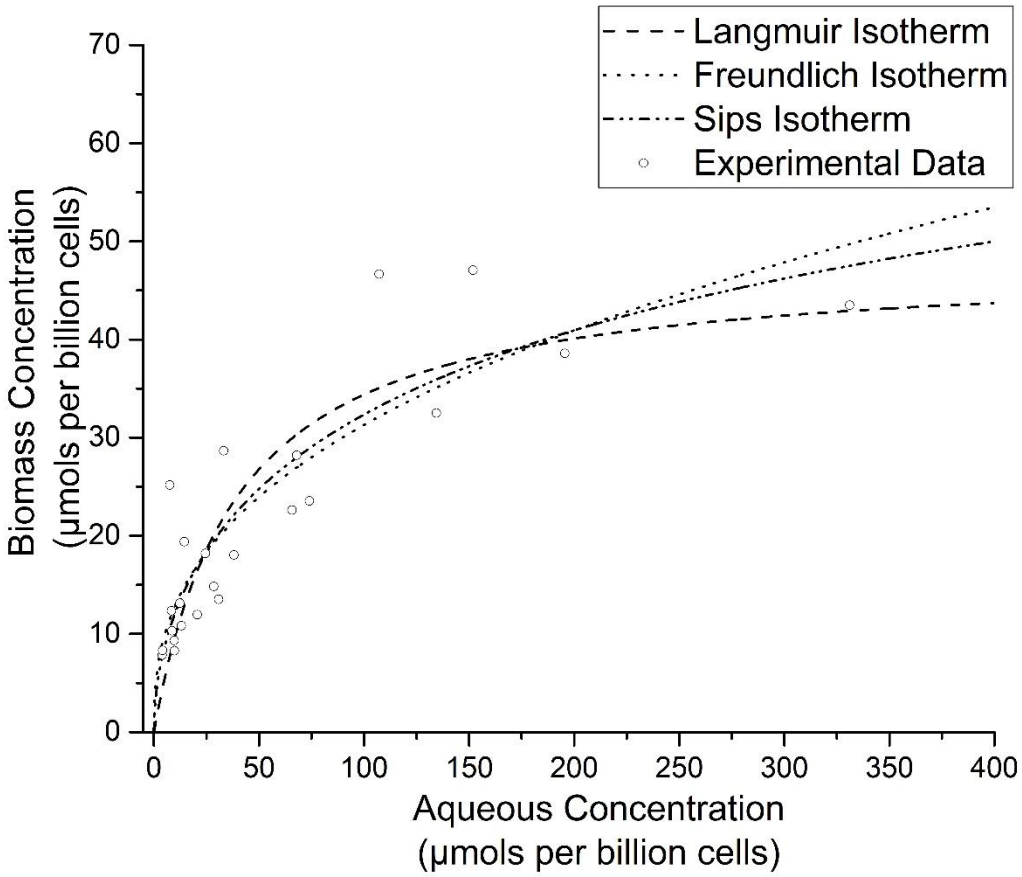


Figure 31: Isotherm comparison for samarium biosorption by C. necator at 35°C and pH 5.

The linearized Langmuir isotherm was also evaluated for samarium as it was for europium and neodymium to follow trends in literature. The fitting of linearized Langmuir to the samarium data set is shown in Figure 32. Based on the R- squared and Pearson's r values, it appears as though for samarium the linearized form of Langmuir isotherm provides a decent fit for modeling the data. The variation within the data may be a contributing factor to a lesser fitting ability resulting in the lower correlation values.

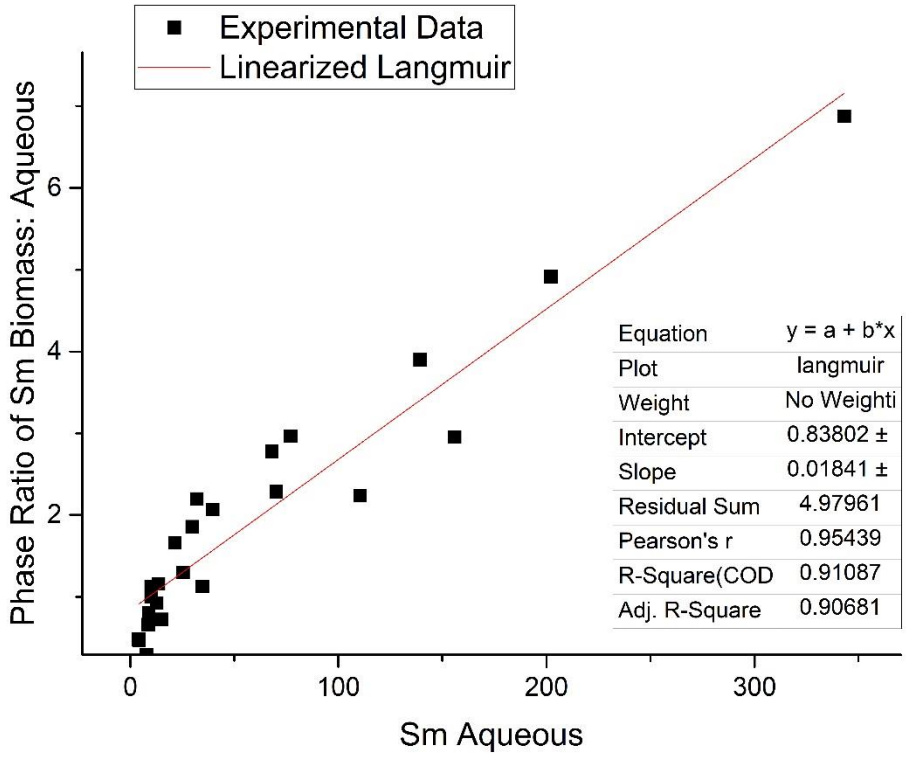


Figure 32: Linearized Langmuir Isotherm for biosorption of neodymium by C. necator.

Based from the assumptions made in its derivation, as well as its ability to fit the data, Sips isotherm appears best suited as a basis for all calculations and derivations of other system properties for comparison purposes. Figure 33 shows a comparison of the Sips isotherms for the europium, neodymium, and samarium. In the figure, it is shown that there is a much higher capacity for samarium than the other to rare earths indicating that more samarium is recoverable by *C. necator*. This figure shows that at very low concentrations, the three REEs have comparable affinity by *C. necator*. However, it is apparent that, as concentration increases, europium initially has a higher affinity until it approaches its maximum absorption. At this point, samarium is still continously biosorbed. This provides predictions for how REE sorption might occur when mixed with initially europium being highly preferred except at higher concentrations, in which, samarium would be prefered.

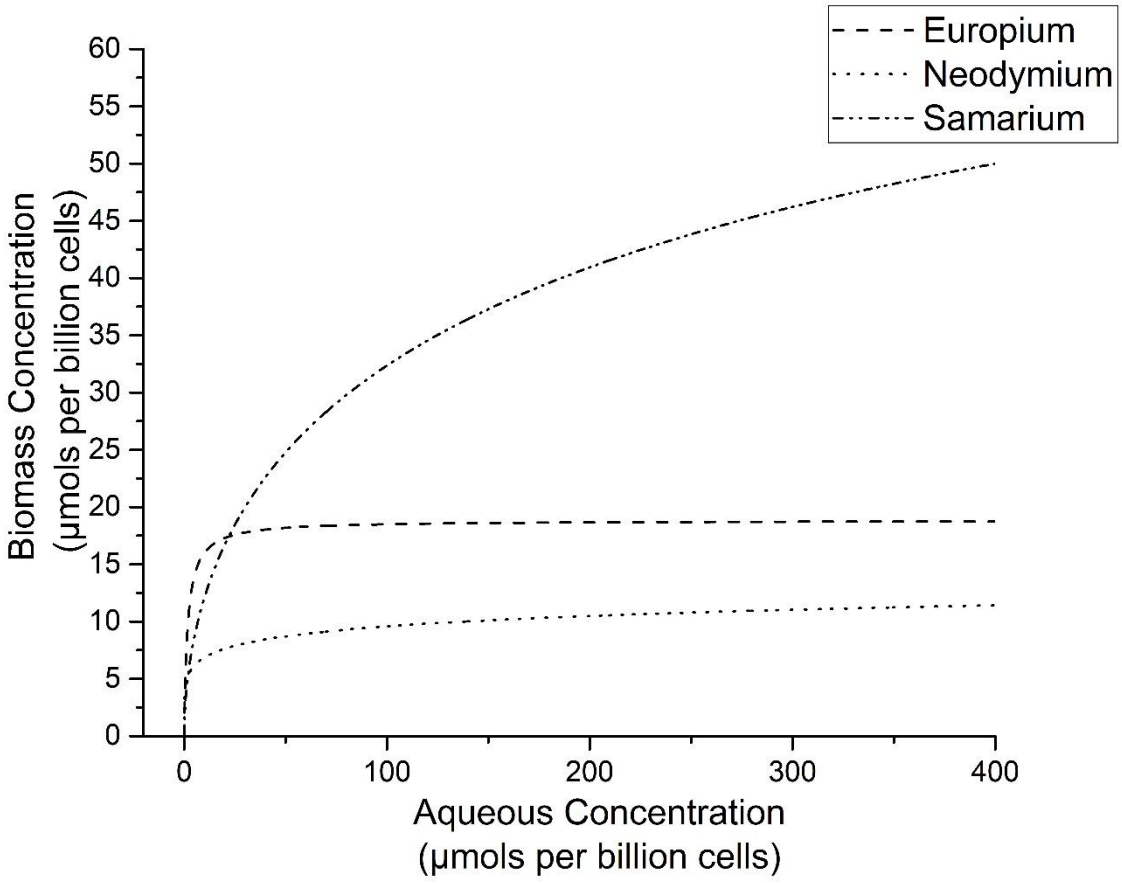


Figure 33: Comparison of Sips Isotherms for biosorption by the bacterium C. necator of europium, neodymium and samarium at 35°C and pH 5.

Quantification of the number of binding sites filled provides an understanding how well *C. necato*r performs in different concentration for different REEs. It also provides a prediction for the level of loading that will be present when subjected to the microdevice. From Figure 34, it is predicted that 60-100 µmols per billion cells will saturate *C. necator* with both europium and neodymium. Saturation of *C. necator* with samarium occurs at greater than 250 µmols per billion cells. It is worth noting, however, the variability that was present in the data obtained which signifies the importance to the degree of error associated with each value. This data variability is to be expected for two main reasons: 1) variability frequently exists in biological systems and 2) isotherms were designed for surface adsorption reactions. In the biosorption processes performed in this thesis other mechanisms of uptake

are expected to be occuring resulting in some deviation from isotherms. With that understanding, general estimates about the performance of *C. necator* can still be compared with acknowledment to the errors.

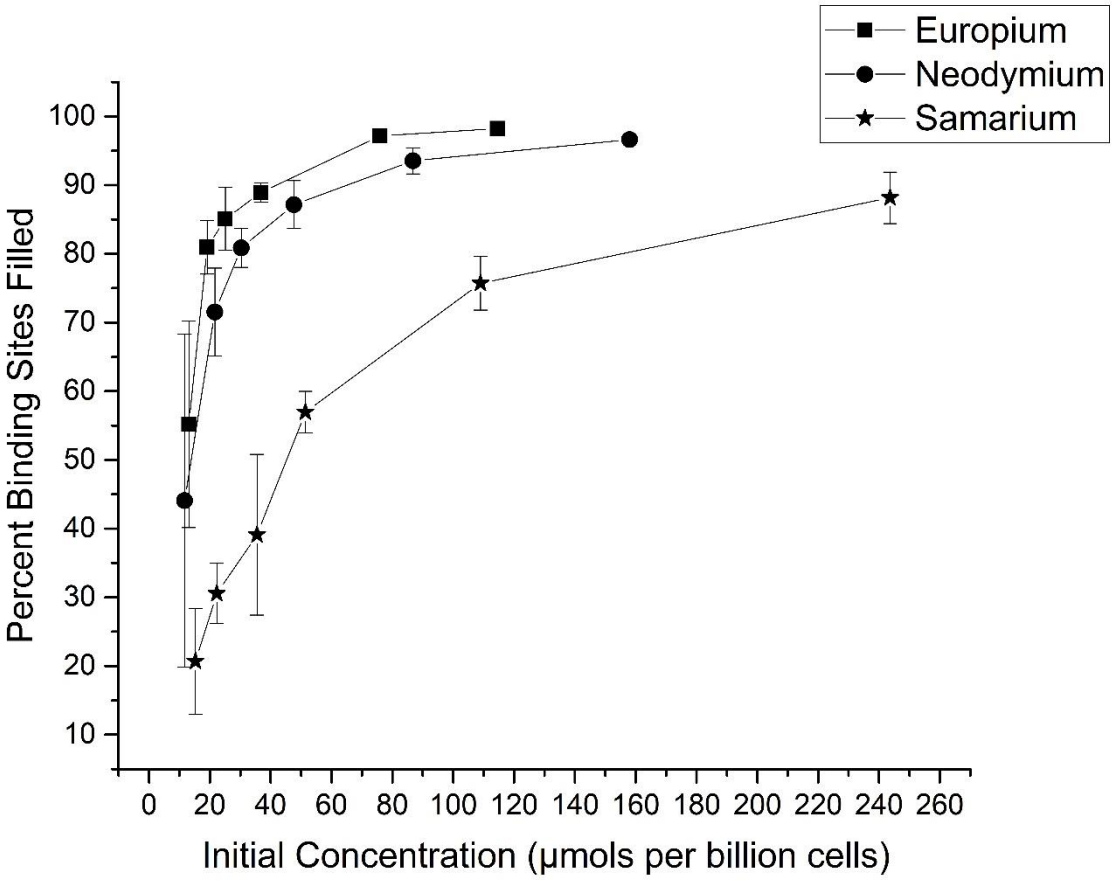


Figure 34: Percent of binding sites filled on C. necator with REE

6.3.5 Temperature Dependence

Temperature experiments were performed to evaluate the effect of temperature on biosorption of rare earth elements by *C. necator*. This was also important because temperatures gradients may occur in the microdevice which can affect retention of the REEs

by *C. necator*. From Figure 35 it is seen that all tested REEs' biosorption by *C. necator* increases at higher temperatures. From a biosorption perspective, biosorption experiments at higher temperatures allows for maximum sorption of REEs. Since *C. necator*'s uptake is greater at higher temperatures, temperatures of up to 75°C are acceptable for this bacterium to experience in microdevice with desorption occurring. The cells during experimentation though had a different phenotype expressed at 75°C compared to the lower temperatures studied. The 75°C exposure pellets that were formed appeared to have a white coloration and where much weaker than the pellets from the lower exposure temperatures. Since cells experienced a noticeable change in phenotype between the 55°C and 75°C, to ensure that the cells are have the same characteristics during separation in the DEP device, it may be advisable to ensure temperatures do not exceed 55°C within the device for *C. necator* cells.

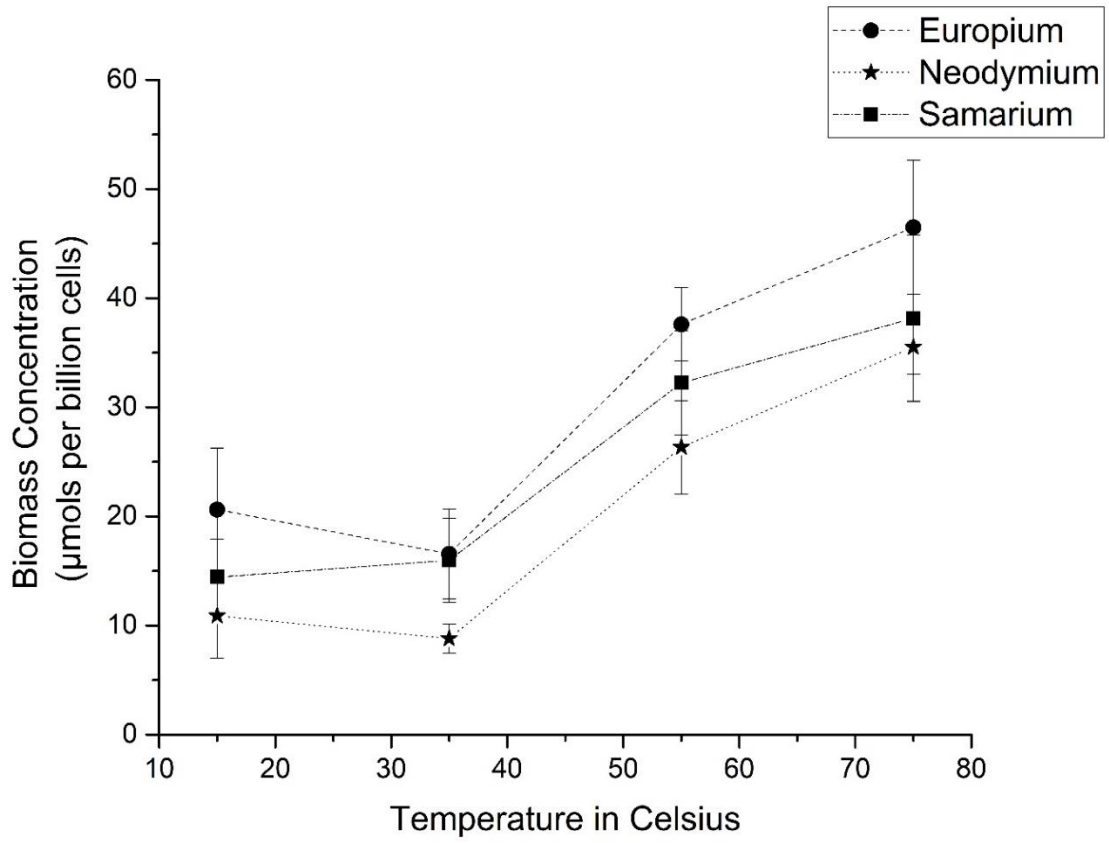


Figure 35: Temperature profile for biosorption by C. necator of europium, neodymium, and samarium at pH 5 and an average exposure concentration of 35 µmols per billion cells for samarium and neodymium and 43 µmols per billion cells for europium.

6.3.6 pH Dependence

Since pH is known to have effects on both the speciation of REEs and functional groups that may be involved with binding with REEs during biosorption, pH variation studies were performed. Microdevices may experience pH differences when running for a long time, as well, exposing the cells to a gradient as it moves along the device. From Figure 36, as pH of exposure is increased, the biosorption capacity of europium, neodymium, and samarium decrease and then level out as preferred pH conditions for *C. necator* are approached. At the pH around 5.5 and up, the rare earth elements start to approach the pH where hydrolysis reactions occur that may form precipitates. From these results, it appears that *C. necator* may be more effective at lower pH. This may be closely related to the speciation of the REEs in solution. Since the REEs are more likely to be in ionic form or in complexes less susceptible to steric hindrances, REEs and their complexes are possibly more likely to be either adsorbed to the surface or transported into the *C. necator* cells then the complexes that are formed at the higher pH values.

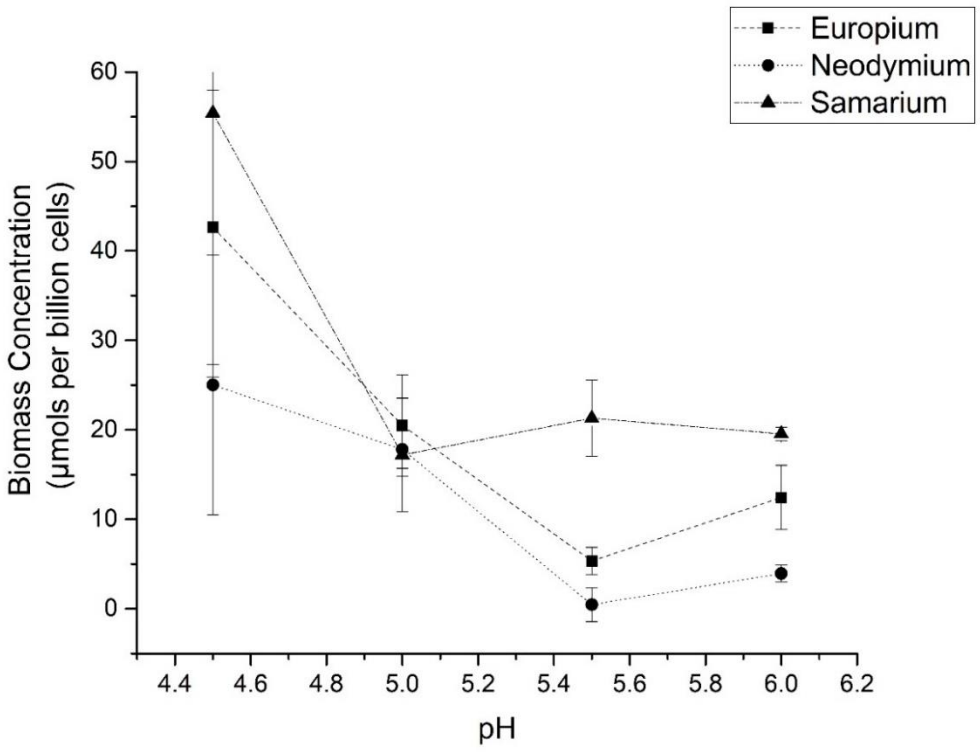


Figure 36: Effect of pH on biosorption by C. necator of europium, neodymium and samarium at 35°C and exposure of approximately 30 µmols per billion cells for europium and neodymium and 40 µmols per billion cells for samarium.

It was also observed that the pH of the equilibrium solutions appears to approach the bacteria's preferred pH value as pH was lowered for the initial solution. The pH adjustment may be attributed to the bacteria adjusting its environmental pH [19, 66]. This phenomenon has been observed in other bacteria to occur and is referred to as acid homeostasis [66]. The adjustment of pH in equilibrium solutions from the initial pH is shown in Figure 37. Values of pH measured after the sorption period are in the range that hydrolysis reactions occur in. As a result, there is the potential that precipitates may have formed before entering the cells potentially preventing some accumulation to occur internally. The adjustments made by this bacterium by overcoming the buffering capacity of the acetic acetate buffer may have partially contributed to some of the variability in sorption seen in the data throughout the experiments performed on *C. necator*.

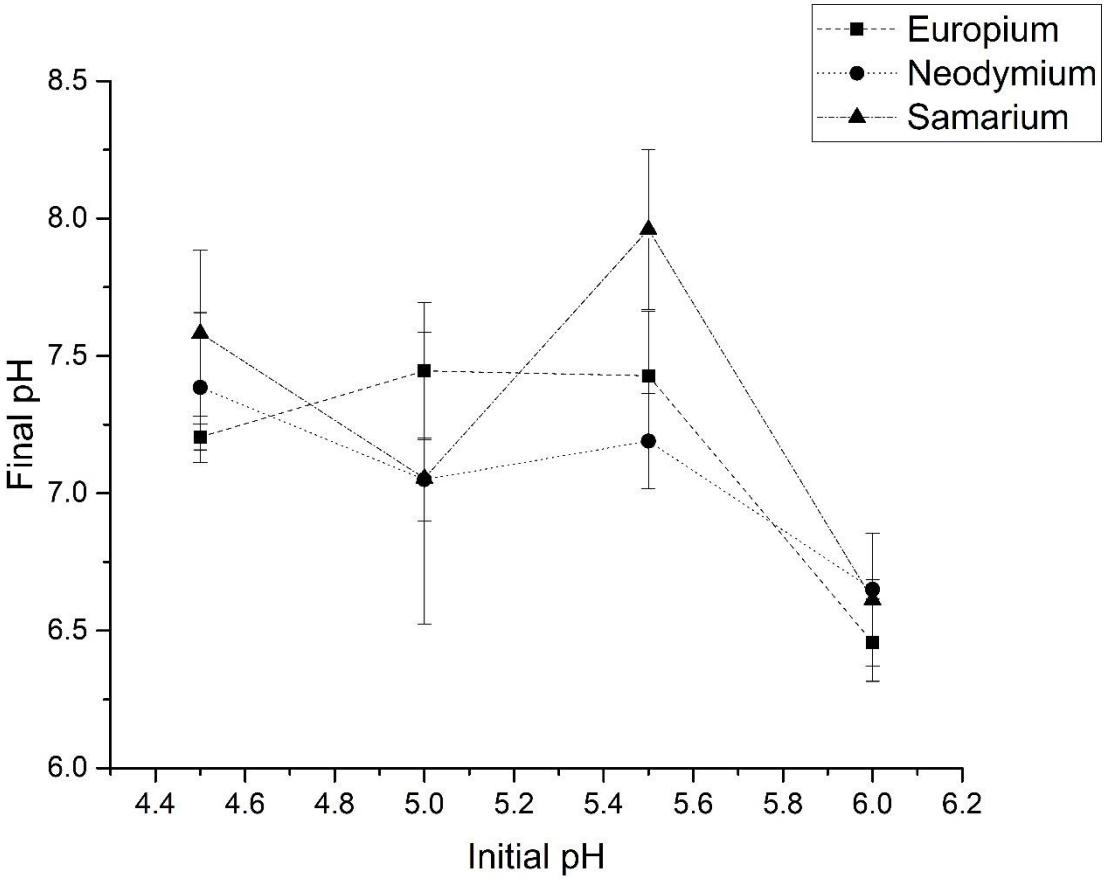


Figure 37: Adjustments in pH from initial pH by C. Necator during biosorption at 35°C.

6.2.7 REE Selectivity

To determine if *C. necator* has a preference of REE biosorption over another REE, two REEs were compared at a range of concentrations. The effect on biosorption of europium with the presence of samarium, and vice versa, is described in Figure 38. In this figure, europium biomass uptake increases as europium content is increased. The same is noted for the samarium content. It appears though that after an initial addition of europium content, samarium occurred at a higher percentage in the biomass phase instead of the aqueous phase. This is found to occur at the 20:80 initial molar ratio per billion cells for europium to samarium. As samarium content decreased, europium occurred more in the biomass phase than in the aqueous phase. However, at the 38:62 and 68:32 ratios, both europium and samarium occurred more in the aqueous phase than in the biomass. From these results, it appears that at low concentrations, europium may have a promoting effect on *C. necator* uptake of samarium into the biomass phase. However, from the results, samarium does not have the same promoting effect on europium biosorption. These results indicates that increased biosorption in the presence of other REEs is potentially a unique characteristic to samarium that europium does not possess.

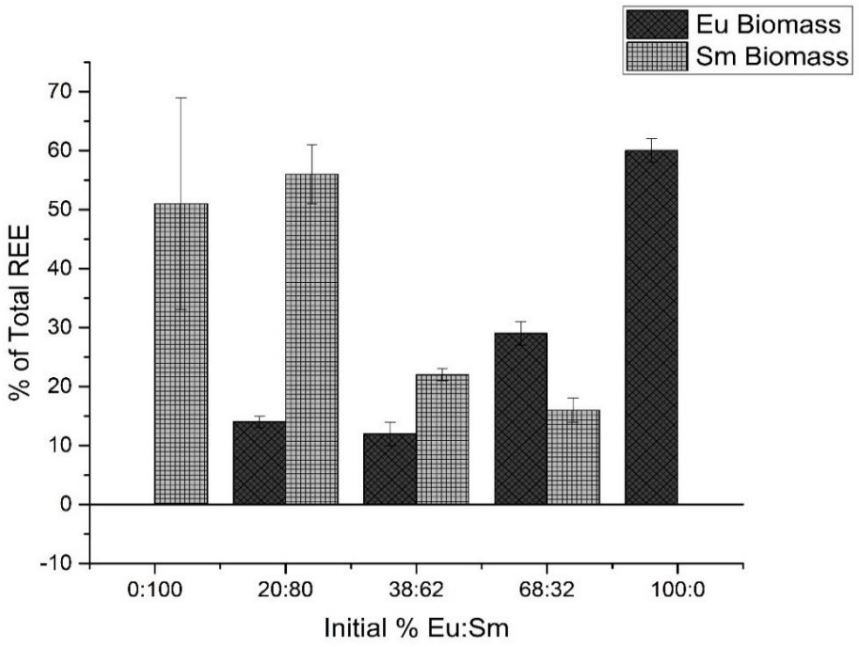


Figure 38: Effect of biosorption of europium and samarium by C. necator with variation of molar ratio per billion cells of europium to samarium in AABS.

The effect on biosorption of neodymium with the presence of samarium, and vice versa, was also studied and is described in Figure 39. The same increase in samarium biomass to aqueous phase ratio occurs with an initial lower percent concentration of neodymium as it did with europium at the same 20:80 percent molar composition per billion cells. This indicates that neodymium, like europium, may act also as a promotor for the uptake of samarium at relatively low neodymium to samarium ratios. From the data obtained, samarium does not appear to have the same effect on neodymium uptake. Similar to the europium mixed with samarium experiments, the neodymium mixed with samarium experiments show at 40:60 molar ratio lower biomass to aqueous ratio of neodymium and samarium. At higher ratios of neodymium to samarium, neodymium uptake appears to increase to *C. necator*'s maximum sorption capacity for neodymium determined by the single concentration biosorption experiments.

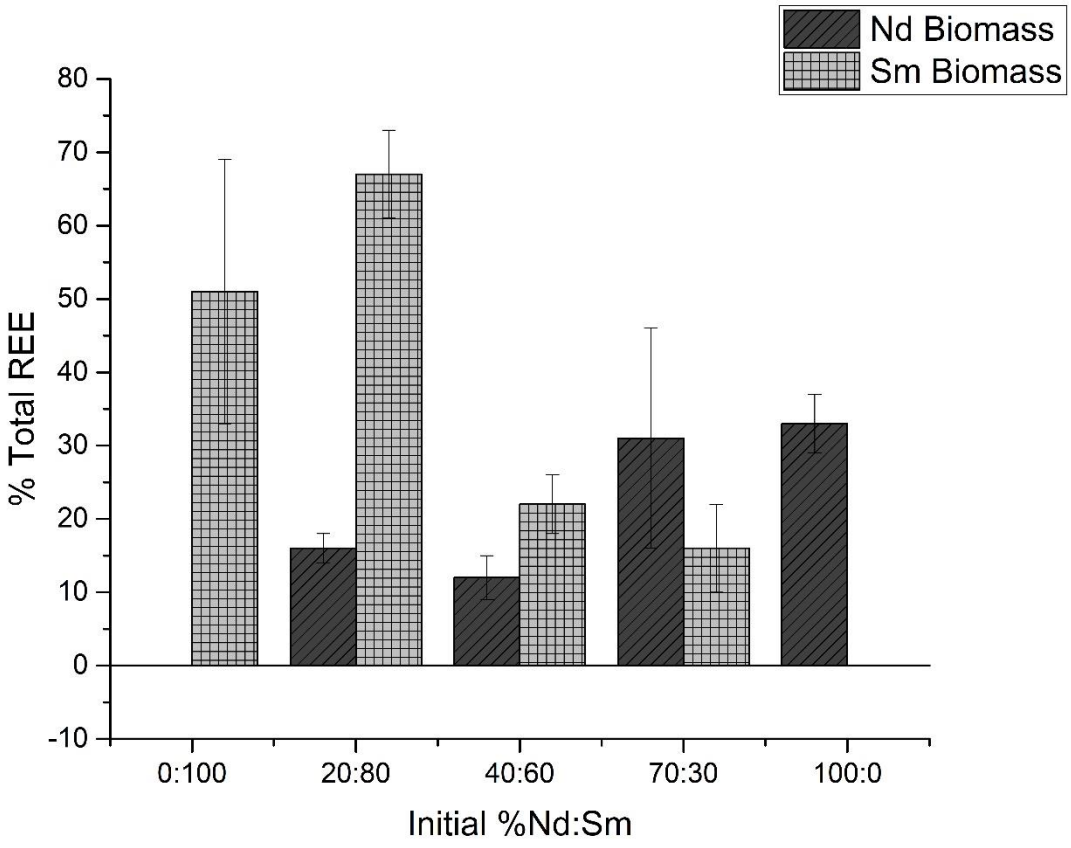


Figure 39: Effect of biosorption of neodymium and samarium by C. necator with variation of molar ratio per billion cells of neodymium to samarium in AABS.

The effect on biosorption of europium with the presence of neodymium, and vice versa, is described in Figure 40. As europium content is increased, neodymium presence in biomass phase is decreased. In addition, europium in the biomass phase increases. From the results it is apparent that europium and neodymium binding occurs independent of one another. Neither europium nor neodymium show an increase in adsorption in the presence of the other REE, also. This indicates that there may be a conformational change by some binding sites on *C. necator* that allows for additional specific binding of samarium resulting from the binding of another REE. There however does seem to be a preference for europium over neodymium binding. This is indicated by the greater amount of europium being bound to neodymium at the concentration level where both REEs were present in approximately the same concentration.

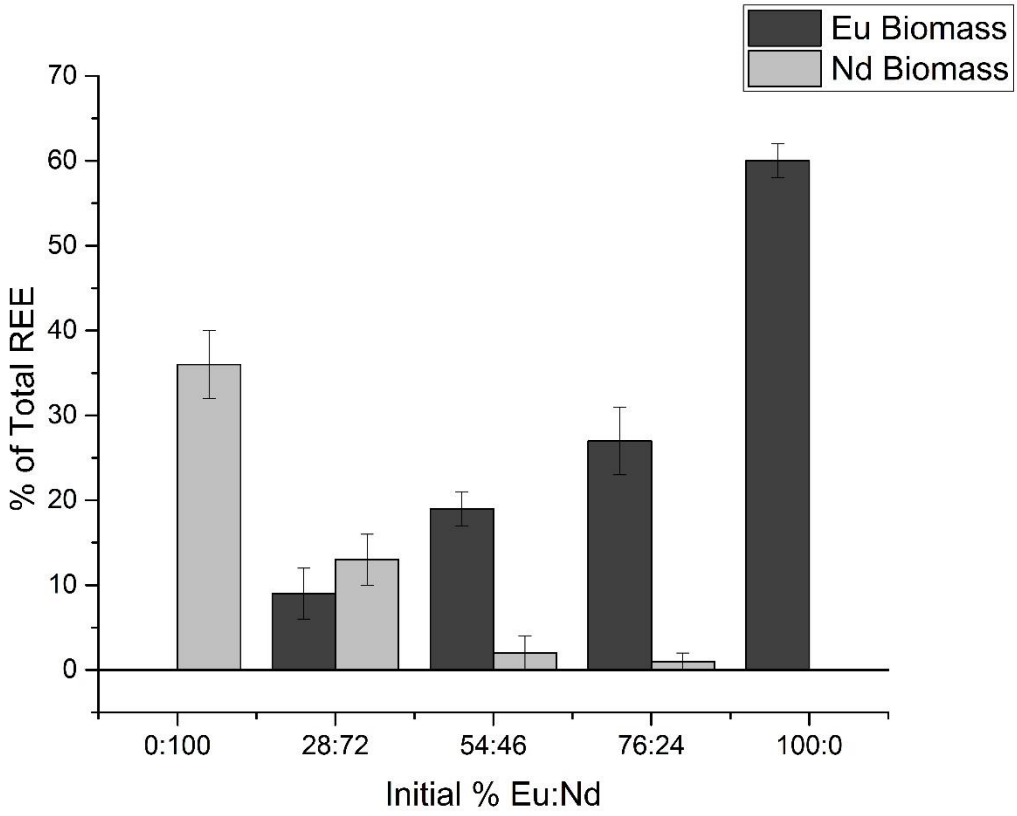


Figure 40: Effect of biosorption of europium and neodymium by C. necator with variation of molar ratio per billion cells of europium to neodymium in AABS.

These three mixed REE experiments overall suggest that samarium is slightly preferred by *C. necator* over europium and neodymium. It also appears that in the presence

of other REEs in low concentrations, *C. necator* has an increased affinity for samarium allowing for greater quantity to be biosorbed. It is then predicted that europium is next preferred. Neodymium seems to either have a low affinity by *C. necator* or it is not capable of being biosorbed in as significant amounts.

6.2.8 TEM Results

To analyze *C. necator* and get some insight into its biosorption ability, *C. necator* samples were viewed using a TEM microscope. The following micrographs show whole mounts of this bacterium as a means of comparison between the normal growth condition of media pH 7.2 compared with the media at pH 6.08. Geometric structure of the bacterial cells appears to be unchanged on average due to the change in growth media pH in the following two micrographs. This indicates that the change in growth pH did not affect structure of the bacterium. Therefore, while during these experiments it was intended for the bacteria to be in a non-growth conditions during exposure, this shows that no apparent physiological changes should occur during exposure at lower pH values.

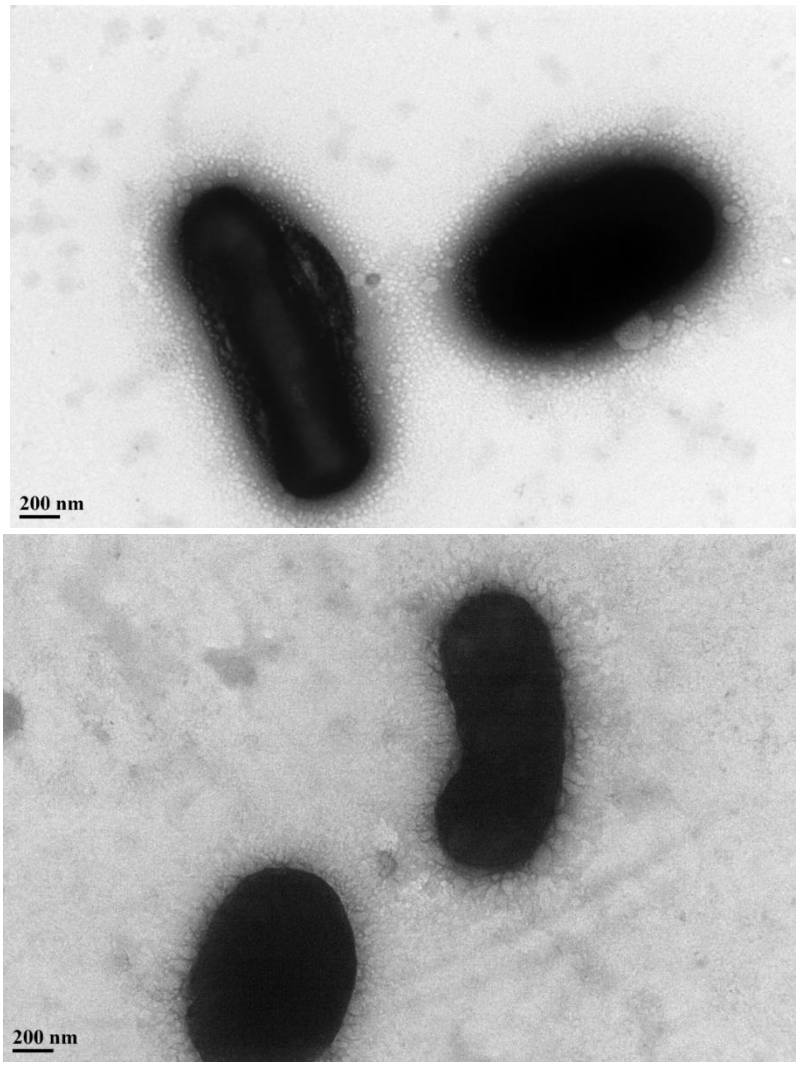


Figure 41: Transmission electron micrograph of C. Necator after growth under media pH of 6.08 (top) and pH of 7.2 (bottom). Whole mount performed and post stained with 2% aq. uranyl acetate. Magnification is SA 8800x. Bennett Carv, Department of Chemical and Materials Engineering, Fall 2016.

The following micrograph displays the general localization of heavy metals. Displayed is a thin section cut at 70nm with sections showing darker contrast. It can be concluded from this micrograph that metals accumulate within the cytosol of the bacterial cell with some that appears to be around the cell membrane. Occlusions are also noticed in the center of these bacterial cells which may be the internal reserves of PHA *C. necator* produce. From this micrograph, it appears that there is a significant amount of neodymium that enters the *C. necator* cells during exposure. It suggests that the distribution of metals is present within the

cytosol. There is a small portion near the occlusion within the cell, however, that appears to have a higher contrast suggesting a higher density of heavy metal.

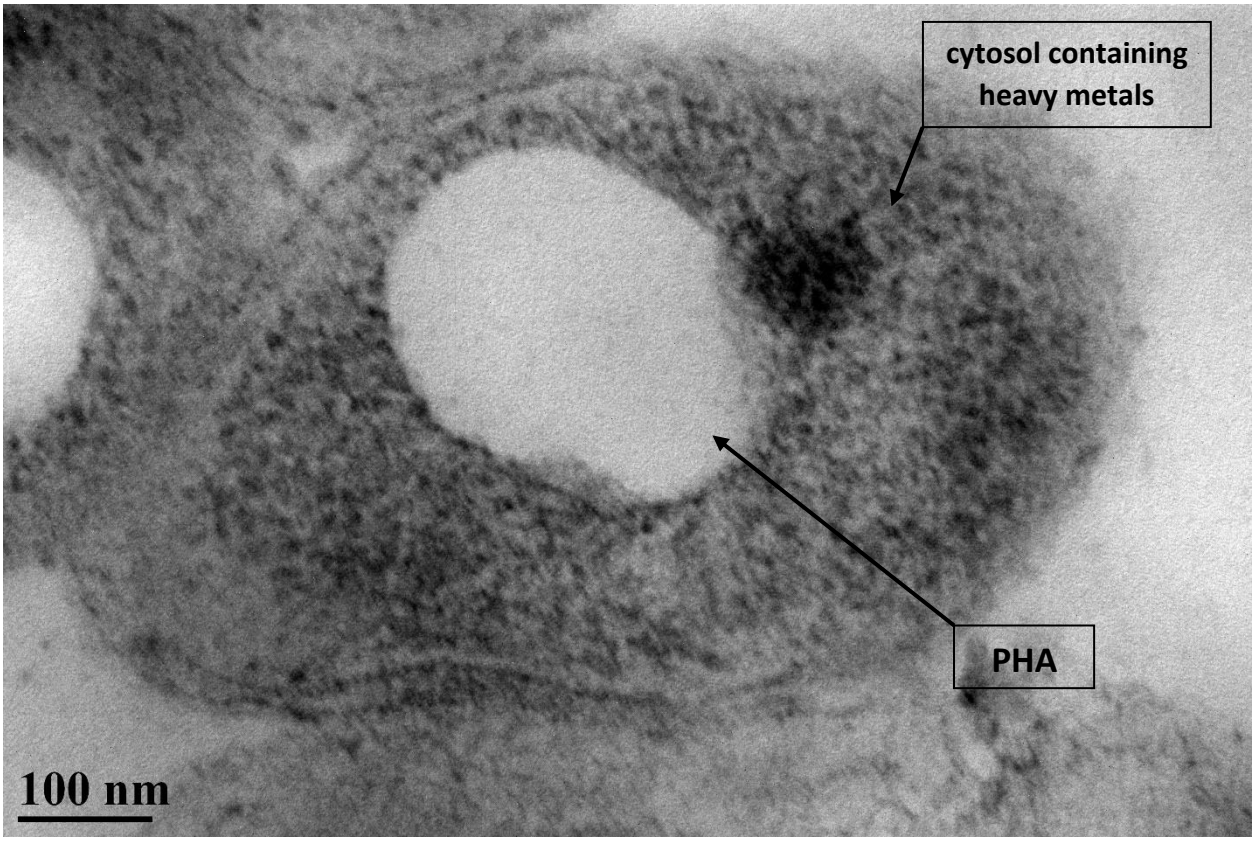


Figure 42: Transmission electron micrograph of C. Necator after sorption of Neodymium under pH of 4.8. 70nm section under SA 20000x magnification through a part of bacterial cell is shown with noticeable concentration of heavy metal within the cytosol of the bacterium. Darker regions indicate higher concentrated regions of Neodymium. Fixation was with 2.5% glutaraldehyde, 2%paraformaldehyde in 0,1M Phosphate buffer (pH 7.2) Spurr's resin, post stained with 2% aq. uranyl acetate and 1% aq. lead citrate. Bennett Carv, Department of Chemical and Materials Engineering, Fall 2016.

7. Concluding Remarks

7.1 Final Remarks for Biosorption of REEs by C. necator

Throughout Chapter 6, *C. necator* is characterized in attempt to develop a model organism to which other organisms that hyperaccumulate metals can be compared. First the ability to accumulate REEs was established with results showing indicating that *C. necator* can achieve sorption of quantities of approximately 30 µmols, 20 µmols, and 50 µmols of europium, neodymium and samarium, respectively per billion cells. These quantities are expected to be the maximum uptake that may occur under idealized conditions as being the only metal ion present. In real systems where other ions are present, these values are expected to fluctuate as indicated by the selectivity experimental results outlined in section 6.2.7. The selectivity results demonstrate the concepts discussed in section 2.3.3 in where an REE may have an effect on the sorption of another REE. The effect was displayed through the increase in percent uptake of samarium into the biomass phase when relatively low initial concentrations of europium or neodymium were present with regard to samarium concentration in the initial aqueous solution prior to exposure. The ability of *C. necator* to uptake europium or neodymium did not show there to be any influence as a result of increased uptake.

The temperature obtained provide some context into the overall optimization of conditions for the biosorption process for which *C. necator* may deployed, as well as, some design factors for which the DEP microdevice this organism is expected to be applied to testing. With regards to temperature, the results in section 6.3.5 establish that the quantity of uptake for all the REEs by *C. necator* is greater at higher temperatures. This follows the discussion outlined in section 2.3.2 on the influence of temperature on biosorption. The exact determination of what mechanism is the factor resulting in higher uptake is not determinable from these experiments. However, this may be worth investigating in other studies to either improve the overall performance of this organisms or harvest the proteins or byproducts of it depending on the cause for the increased sorption. Valuable information was also determined for the separation process from observations made about *C. necator* at the temperature of 75°C in which the physiology of the bacterium appeared to be dead. This

is important to note since the DEP device ability to function is dependent on factors exhibited by the *C. necator* cells, as discussed in section 3.5 in detail.

From the pH results discussed in section 6.3.6, it is understood that the lower pH provided better uptake ability for the REEs than the higher pH. This predicted to be directly related to results obtained about the speciation prediction in section 6.3.1 in which the REEs are predicted to be in predominately the ionic species. Coupled with the results that found in 6.3.6 that *C. necator* was overcoming the buffer and modifying the pH of the environment, it follows that the chemistry of the REEs were altered based on the discussion in section 2.3.1. The lower uptake can be either attributed to increased steric hindrances and change in charge caused by the complexes formed, as well as, precipitates that may have formed.

While the isotherms evaluated for *C. necator* to describe its capacity for adsorption of REEs, it is understandable as discussed in relation to section 2.4 where the isotherms were developed through theory that they did not fit the system well. However, upon comparison to the usage of these isotherm in literature, specifically and most commonly the linearized Langmuir, it match other process in which bacterium were analyzed as biosorbents [67]. As a result, it is understood that the information is for comparative purposes and is greatly limited due in part to the fact as stated in section 2.4 that these isotherms are mainly for surface sorption process whereas other mechanisms persist during biosorption.

In summary, *C. necator* was capable of providing a basis for which comparison of other organisms that hyperaccumulate metals can be compared as a model organism. It accomplishes the task of accumulating the REEs of interest, as well as, being an organism that has been well studied to allow for further modifications to be determined based on further investigations into the biosorption mechanisms that occurred. It also successfully brought to light other considerations that need attention such as the overcoming of the buffer that this bacteria showed potential to do.

7.2 Future Work for Biosorption of REEs

While this work provides a good starting basis by providing performance information of *C. necator* as a basis for other biosorbents may be compared, more work is still needed to improve it as a model organism. Within the results obtained, it was found that the bacterium

was actively adjusting its media. As a result, the pH of the system was adjusted to the range that was known to result in hydrolysis occurring and forming precipitates. This occurrence means that the *C. necator* bacteria may have been feasting on the acetate buffer that was used to maintain the system. Work will need to be done to find another suitable biological buffer that would not provide any future bacteria studies with a carbon source to metabolize. It was noticed that the isotherm models followed trends of the data but did not fit the data well. This may have been due to variation between data points potentially resulting from REEs being actively transported into the biomass at different rates. This may also have occurred if equilibrium time was not reached, however unlikely since previously published studies use similar equilibrium times [63, 68]. For these reasons, a kinetic study of absorption will need to be performed to determine when equilibrium is reached for all biomass samples.

Another area where future research would also benefit if it is expanded to acidophilic bacteria such as *Acidithiobacillus ferrooxidans*. This type of bacteria thrives in the low pH required to ensure the rare earths do not create precipitates. Also, these bacteria thrive in regions where there are sulfuric acids similar to regions where the rare earth elements are mined. The challenge with *A.T. ferrooxidans*, however, is the development of media for growth that does not contain precipitates. This is needed to ensure no precipitates enter the DEP device.

After production of the DEP device, verification will need to be performed through analysis of device outputs. In addition, other biomass types will need to be characterized and outputs analyzed from device to ensure device is universally useful. Data from outputs will also allow for better optimization of input conditions of biomass for the device and provide an understanding of further areas of interest that need to be researched to effectively compare biomass cell types for this separation process.

7.3 Final Remarks for Inexpensive Alternatives to Metal Analysis

With regard to the results of the experiments to determine the inexpensive alternative to current metal analysis techniques, it is exhibited by the results in section 6.3 that UV-Vis analysis is applicable. However, this analysis is limited in applicability due to the potential to interference by competing metal ions. For utilization of this method of analysis, it is

determined that system must already be well characterized, which provides little opportunity for effective utilization.

In contrast, however, fluorescence spectroscopy shows from the results in section 4.4 much promise. In this thesis it is shown that linear correlations exist that may be applied in the determination of concentration when fluorescence intensity is obtained at a specific excitation and emission wavelength set. In addition, for reasons discussed in section 4.4, the results indicate that there is potential to differentiate REEs allowing for a cheaper quantification method without losing much sensitivity.

7.4 Future Work for Inexpensive Alternatives to Metal Analysis

The final area of future research interest is the continued investigation of fluorescence to quantify REEs in aqueous solutions. While the results presented in this thesis show that there is ability to quantify REEs, higher resolution studies are still needed to fully discover the potential this technique may have. One specific area that needs to be researched is the effect of reducing the slit width on the monochromator on the spectrofluorophotometer. By doing this, it is expected that higher resolutions are achievable. This is expected to increase the sensitivity of the spectrofluorophotometer to the structural differences that occur while binding to the different REEs allowing for differential presence determination.

References

1. Maury, D.O. ENS Lyon.
2. *REE - Rare Earth Elements and their Uses*. [Webpage] [cited 2017; Available from: http://geology.com/articles/rare-earth-elements/.
3. Bio-Rad. *Principles of the Flow Cytometer*. 2016 [cited 2017; Available from: https://www.bio-rad-antibodies.com/flow-cytometry-electrostatic-cell-sorting.html.
4. Generalic, E. *Rare Earth Elements (REE)*. [Web Page] 2017 [cited 2017; Available from: http://www.periodni.com/rare_earth_elements.html.
5. Schelmetic, T. *Are Hybrid Vehicle Manufacturers Shifting Gears Away from Rare Earth Elements?* Industry News 2012 11 December 2012; Available from: http://news.thomasnet.com/imt/2012/12/11/are-hybrid-vehicle-manufacturers-shifting-gears-away-from-rare-earth-elements.
6. Chemistry, R.S.o. *Europium- Element information, properties, and uses*. 2017 [cited 2017; Available from: http://www.rsc.org/periodic-table/element/63/europium.
7. Chemistry, R.S.o. *Neodymium- Elementinformation, properties, and uses*. [Web Page] 2017 [cited 2017; Available from: http://www.rsc.org/periodic-table/element/60/neodymium.
8. Chemistry, R.S.o. *Samarium- Element information, properties, and uses*. [Web Page] 2017 [cited 2017; Available from: http://www.rsc.org/periodic-table/element/62/samarium.
9. Ozaki, T., et al., *Sorption behavior of europium(III) and curium(III)on the cell surfaces of microorganisms.* Radiochim. Acta. **92**: p. 741-748.
10. Klinger, J.M., *A historical geography of rare earth elements: From discovery to the atomic age.* The Extractive Industries and Society, 2015. **2**(3): p. 572-580.
11. STURZA1, C.M., R. BOSCENCU2, and V. NACEA2, *THE LANTHANIDES: PHYSICO–CHEMICAL PROPERTIES RELEVANT FOR THEIR BIOMEDICAL APPLICATIONS.* FARMACIA, 2008. **LVI**(3): p. 326-338.
12. Shannon, R., *Revised effective ionic radii and systematic studies of interatomic distances in halides and chalcogenides.* Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography, 1976. **32**(5): p. 751-767.
13. Martinez-Gomez, N.C., H.N. Vu, and E. Skovran, *Lanthanide Chemistry: From Coordination in Chemical Complexes Shaping Our Technology to Coordination in Enzymes Shaping Bacterial Metabolism.* Inorganic Chemistry, 2016. **55**(20): p. 10083-10089.
14. Shaik, A. *Nd:YAG laser*. [Webpage] 2014 [cited 2017; 14 OCtober 2014:[Available from: http://www.physics-and-radio-electronics.com/physics/laser/ndyaglaser.html.
15. Long, K.R., Van Gosen, B.S., Foley, N.K. and Cordier, Daniel, *The principal rare earth elements deposits of the United States—A summary of domestic deposits and a global perspective: U.S. Geological Survey Scientific Investigations Report*. 2010.
16. Michalak, I., K. Chojnacka, and A. Witek-Krowiak, *State of the art for the biosorption process--a review.* Appl Biochem Biotechnol, 2013. **170**(6): p. 1389-416.
17. Dogsa, I., et al., *Structure of Bacterial Extracellular Polymeric Substances at Different pH Values as Determined by SAXS.* Biophysical Journal, 2005. **89**(4): p. 2711-2720.
18. Veglio, F. and F. Beolchini, *Removal of metals by biosorption: a review.* Hydrometallurgy, 1997. **44**(3): p. 301-316.
19. Nagy, B., et al., *Linear and nonlinear regression analysis for heavy metals removal using Agaricus bisporus macrofungus.* Arabian Journal of Chemistry, 2017. **10**: p. S3569-S3579.
20. Gadd, G.M., *Biosorption: critical review of scientific rationale, environmental importance and significance for pollution treatment.* Journal of Chemical Technology & Biotechnology, 2009. **84**(1): p. 13-28.
21. Das, N. and D. Das, *Recovery of rare earth metals through biosorption: An overview*. Journal of Rare Earths, 2013. **31**(10): p. 933-943.

22. Xu, S., et al., *Biosorption of La^{3+} and Ce^{3+} by Agrobacterium sp. HN1*. Journal of Rare Earths, 2011. **29**(3): p. 265-270.
23. Diniz, V. and B. Volesky, *Biosorption of La, Eu and Yb using Sargassum biomass.* Water Research, 2005. **39**(1): p. 239-247.
24. Oliveira, R.C., et al., *Characterization of metal–biomass interactions in the lanthanum(III) biosorption on Sargassum sp. using SEM/EDX, FTIR, and XPS: Preliminary studies.* Chemical Engineering Journal, 2014. **239**(0): p. 381-391.
25. Lo, Y.-C., et al., *Recovery of high-value metals from geothermal sites by biosorption and bioaccumulation.* Bioresource Technology, 2014. **160**: p. 182-190.
26. Ahalya, N., T.V. Ramachandra, and R.D. Kanamadi, *Biosorption of Heavy Metals.* Research Journal Of Chemistry And Environment 2003. **7**(4).
27. Tobin, S.V.A.a.J.M., *Mechanism of Adsorption of Hard and Soft Metal Ions to Saccharomyces Cerevisiae and Influence of Hard and Soft Anions.* Applied and Environmental Microiology. **59**(9): p. 851-856.
28. Wang, J.a.C.C., *Biosorption of Heavy Metals by Saccharomyces Cerevisiae: A Review.* Biotechnology Advances, 2006. **24**(5): p. 301-16.
29. Davis, T.A.a.B.V.a.A.M., *A Review of the Biochemistry of Heavy Metal Biosorption by Brown Algae.* Water Research, 2003. **37**(18): p. 4311.
30. Texier, A.C., Y. Andres, and Pierre Le Cloirec, *Selective Biosorption of Lanthanide (La, Eu) Ions by Mycobacterium Smegmatis.* Environmental Technology, 1997. **18**(8): p. 835-841.
31. Texier, A.-C., Y. Andrès, and P. Le Cloirec, *Selective Biosorption of Lanthanide (La, Eu, Yb) Ions by Pseudomonas aeruginosa.* Environmental Science & Technology, 1999. **33**(3): p. 489-495.
32. Schiewer, S. and B. Volesky, *Modeling of the Proton-Metal Ion Exchange in Biosorption.* Environmental Science & Technology, 1995. **29**(12): p. 3049-3058.
33. Kratochvil, D.a.B.V., *Advanceds in the Biosorption of Heavy Metals.* TIBTECH, 1998. **16**: p. 229-242.
34. Moriwaki, H. and H. Yamamoto, *Interactions of microorganisms with rare earth ions and their utilization for separation and environmental technology.* Applied Microbiology and Biotechnology, 2013. **97**(1): p. 1-8.
35. Tsezos, M., Emmanouela Remoundaki, and Artin Hatzikiosyian, *Biosorptin - Principles And Applications For Metal Immobilization From Waste-Water Streams.* Workshop on Clean Production and Nano Technologies: p. 23-33.
36. Saad, A.M., *Biosorption of soluble and insoluble inorganic compounds by non-trained and cobalt-trained Mucor rouxii NRRL 1894 and Rhizopus sp. biomass.* European Journal of Biotechnology and Bioscience 2014. **2**(5): p. 21-26.
37. Nitz, M., et al., *Structural Origin of the High Affinity of a Chemically Evolved Lanthanide-Binding Peptide.* Angewandte Chemie International Edition, 2004. **43**(28): p. 3682-3685.
38. Dana D. Makrovic, B.M.L., Vladana N. Rajakovic-Ognjanovic, Antonije E. Onjia, and Ljubinka V. Rajakovic, *A New Approach in Regression Analysis for Modeling Adsorption Isotherms.* The Scientific World Journal, 2014. **2014**.
39. Skopp, J., *Derivation of the Freundlich Adsorption Isotherm from Kinetics.* Journal of Chemical Education, 2009. **86**(11): p. 1341-1343.
40. Clement, G.P.J.a.T.P., *A modified Langmuir-Freundlich isotherm model for simulating pH-dependent adsorption effects.* Journal of Contaminant Hydrology, 2012. **129-130**: p. 46-53.
41. Postnova. *General Theory about Field-Flow Fractionation.* [cited 2017; Available from: http://www.postnova.com/general-theory.html.
42. Postnova, *The Field-Flow Fractionation Principle*, F. Principle, Editor., Postnova: postnova.com.
43. Pethig, R., *Review Article—Dielectrophoresis: Status of the theory, technology, and applications.* Biomicrofluidics, 2010. **4**(2): p. 022811.

44. L, D., *Encyclopedia of Microfluidics and Nanofluidics*. Springer Reference.
45. USGS. *What is ICP-MS?* 2017 [cited 2017; Available from: https://crustal.usgs.gov/laboratories/icpms/intro.html.
46. Laboratories, E. *Inductively Coupled Plasma Optical Emission Spectroscopy (ICP-OES)*. 2017 [cited 2017; Available from: http://www.eag.com/inductively-coupled-plasma-icp-oes/.
47. Chemiasoft. *Inductively Coupled Plasma Optical Emission Spectrometry (ICP-OES)*. 2014 [cited 2017 7 March 2017]; Available from: http://www.chemiasoft.com/chemd/node/52.
48. Chausseau, M., et al., *High-Resolution ICP-OES for the Determination of Trace Elements in a Rare Earth Element Matrix and in NdFeB Magnetic Materials.* Spectroscopy, 2014. **29**(11).
49. Technologies, A., *Microwave Plasma Atomic Emission Spectroscopy (MP-AES)*. Agilent Technologies.
50. Lu, Y.W., G. Laurent, and H. Pereira, *A novel methodology for evaluation of formation constants of complexes: example of lanthanide–Arsenazo III complexes.* Talanta, 2004. **62**(5): p. 959-970.
51. Kratochvil, B. and X.-W. He, *A study of the Ca2+–Arsenazo III system and its application to the spectrophotometric determination of free calcium in solution.* Canadian journal of chemistry, 1990. **68**(11): p. 1932-1936.
52. Gratzer, W.B. and G.H. Beaven, *Use of the metal-ion indicator, Arsenazo III, in the measurement of calcium binding.* Analytical Biochemistry, 1977. **81**(1): p. 118-129.
53. Savvin, S.B., *Analytical use of arsenazo III.* Talanta, 1961. **8**(9): p. 673-685.
54. Yong, P., H. Eccles, and L.E. Macaskie, *Determination of uranium, thorium and lanthanum in mixed solutions using simultaneous spectrophotometry.* Analytica Chimica Acta, 1996. **329**(1): p. 173-179.
55. Zheng, Z., et al., *Influence of gamma irradiation on uranium determination by Arsenazo III in the presence of Fe(II)/Fe(III).* Chemosphere, 2014. **107**: p. 373-378.
56. Konstantinou, M. and I. Pashalidis, *Speciation and spectrophotometric determination of uranium in seawater.* Mediterranean Marine Science; Vol 5, No 1 (2004), 2004.
57. Savvin, S.B., *Analytical applications of arsenazo III—III.* Talanta, 1964. **11**(1): p. 7-19.
58. Rowatt, E. and R.J. Williams, *The interaction of cations with the dye arsenazo III.* Biochemical Journal, 1989. **259**(1): p. 295-298.
59. Rohwer, H. and E. Hosten, *pH dependence of the reactions of arsenazo III with the lanthanides.* Analytica Chimica Acta, 1997. **339**(3): p. 271-277.
60. Rohwer, H., N. Collier, and E. Hosten, *Spectrophotometric study of arsenazo III and its interactions with lanthanides.* Analytica Chimica Acta, 1995. **314**(3): p. 219-223.
61. Rowatt, E.a.R.J.P.W., *The interaction of cations with the dye arsenazo III.* Biochemical Journal, 1989. **259**: p. 295-298.
62. Lakowicz, J.R., *Principles of Fluorescence Spectroscopy*. 3rd ed, ed. L. Springer Science+Business Media. 2006: Springer.
63. Oves, M., M.S. Khan, and A. Zaidi, *Biosorption of heavy metals by Bacillus thuringiensis strain OSM29 originating from industrial effluent contaminated north Indian soil.* Saudi Journal of Biological Sciences, 2013. **20**(2): p. 121-129.
64. Kometani, T.Y., et al., *Dry ashing of airborne particulate matter on paper and glass fiber filters for trace metal analysis by atomic absorption spectrometry.* Environmental Science & Technology, 1972. **6**(7): p. 617-620.
65. Vasile, C., et al., *Thermoxidative Decomposition of Some Polysulfones under Dynamic Conditions of Heating.* Journal of Thermal Analysis and Calorimetry, 1998. **52**(2): p. 569-579.
66. Krulwich, T.A., G. Sachs, and E. Padan, *Molecular aspects of bacterial pH sensing and homeostasis.* Nature reviews. Microbiology, 2011. **9**(5): p. 330-343.
67. John E. Aston, W.A.A., Brady D. Lee and Brent M. Peyton, *Effects of cell condition, pH, and temperature on lead, zinc, and copper sorption to Acithiobacillus caldus strain BC13.* Journal of Hazardous Materials, 2010. **184**: p. 34-41.

68. Wierzba, S., *Biosorption of lead(II), zinc(II) and nickel(II) from industrial wastewater by Stenotrophomonas maltophilia and Bacillus subtilis.* Polish Journal of Chemical Technology, 2015. **17**(I): p. 79-87.

Appendix A: Supporting Materials for Metal Quantification

A.1 Example UV-Vis REE Quantification Calibration Curves

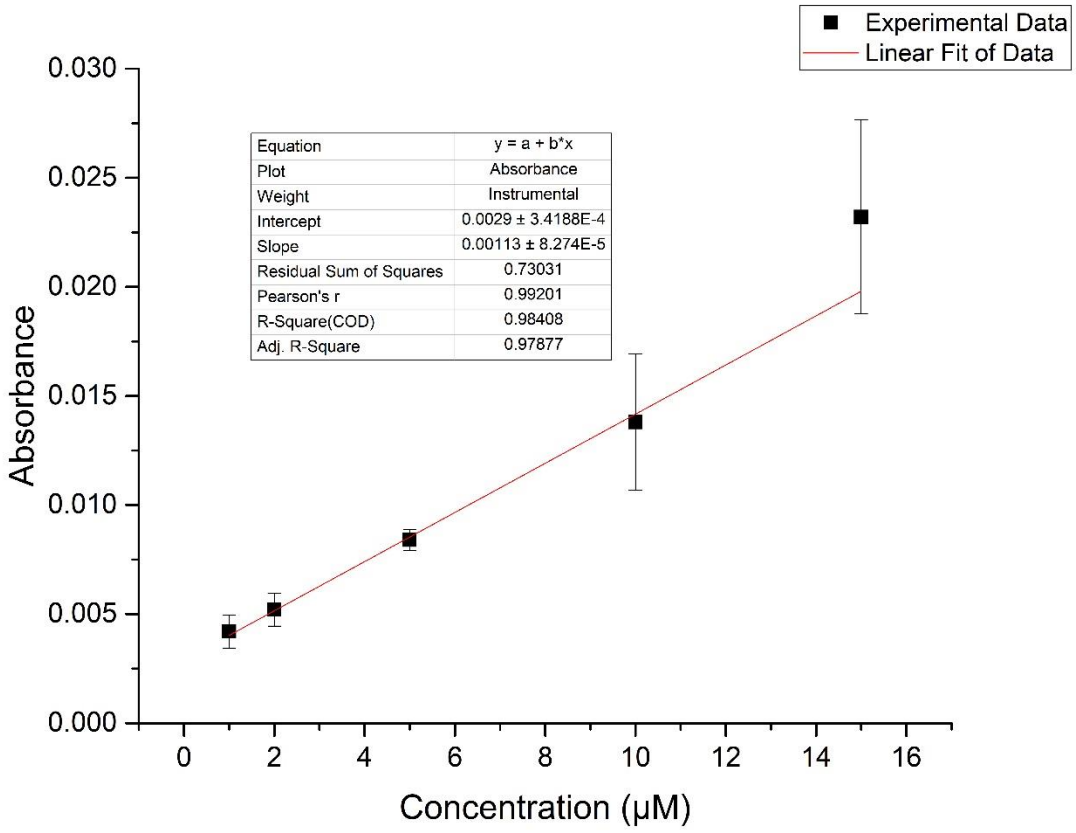


Figure A1 Sample calibration curve for europium determination using UV-Vis at 714 nm with arsenazo III complexing agent

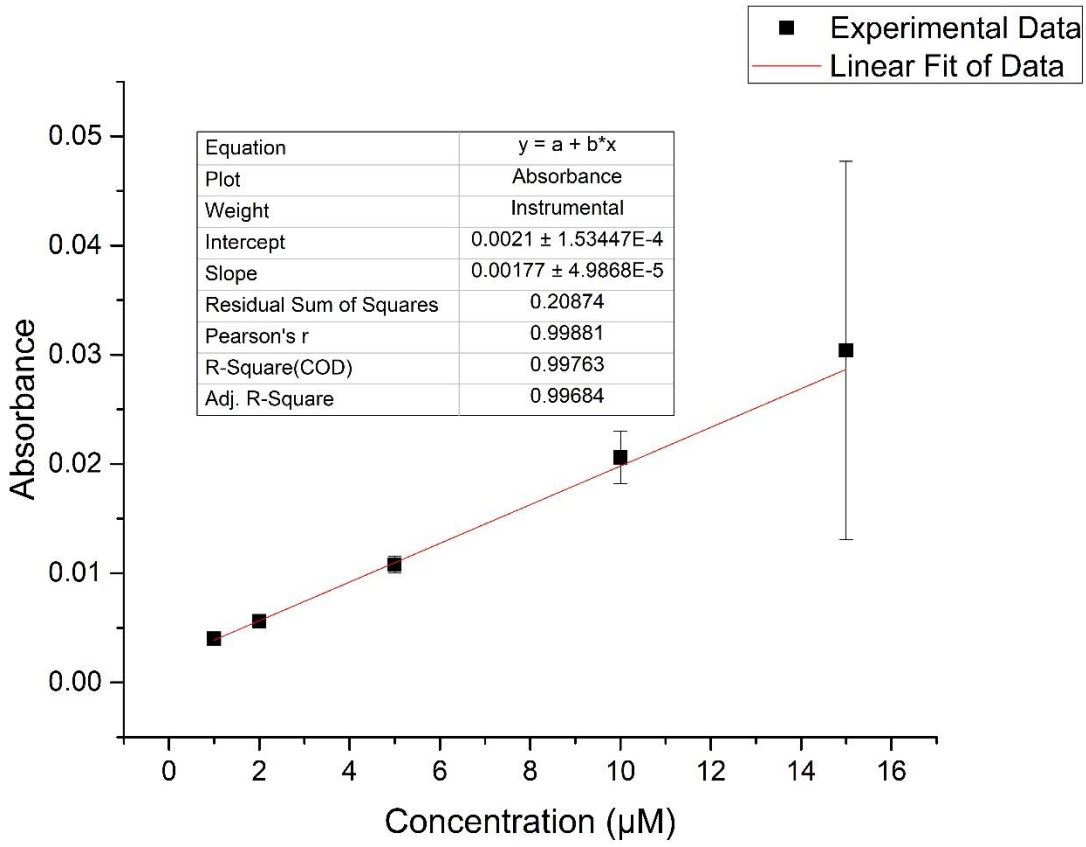


Figure A2 Sample calibration curve for neodymium determination using UV-Vis at 712 nm with arsenazo III complexing agent

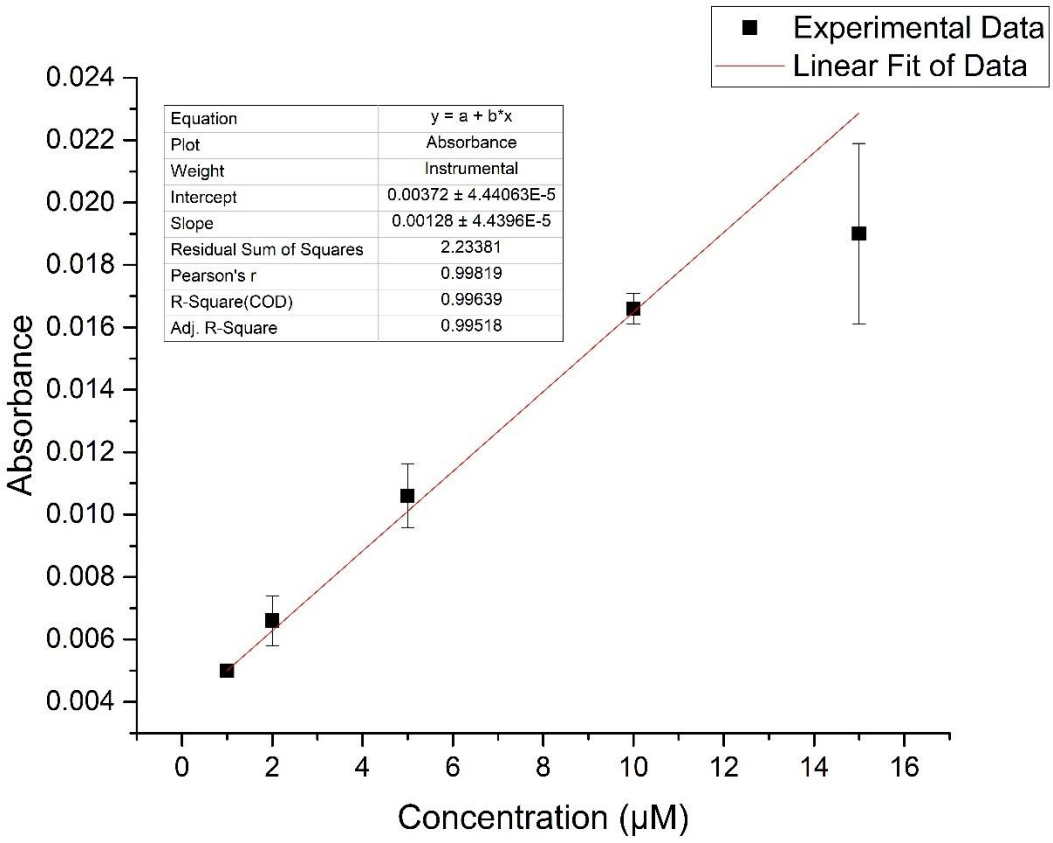


Figure A.3 Sample calibration curve for samarium determination using UV-Vis at 716 nm with arsenazo III complexing agent

A.2 Fluorescence Sample MATLAB Code

```
clear all
clc
% THIS FILE INCLUDES ARSENEZOL BLANKS, WHICH WILL GIVE US A ZERO
% CONCENTRATION VALUE
% 0 to 10uM graphing absorbance vs concentration gives

disp('.625 to 10ppm Eu in DDH2O')
%loading data into Matrices
dA1=csvread('0.625ppm Nd in AABS spike _a_.csv',4,1);
dB1=csvread('1.25ppm Nd in AABS spike _a_.csv',4,1);
dC1=csvread('2.5ppm Nd in AABS spike _a_.csv',4,1);
dD1=csvread('5ppm Nd in AABS spike _a_.csv',4,1);
dE1=csvread('10ppm Nd in AABS spike _a_.csv',4,1);
dA2=csvread('0.625ppm Nd in AABS spike _b_.csv',4,1);
dB2=csvread('1.25ppm Nd in AABS spike _b_.csv',4,1);
dC2=csvread('2.5ppm Nd in AABS spike _b_.csv',4,1);
dD2=csvread('5ppm Nd in AABS spike _b_.csv',4,1);
Nd_E2=csvread('10ppm Nd in AABS spike _b_.csv',4,1);
figure(21)
waterfall(Nd_E2)
xticklabels([220 288 356 424 492 560 628 696 764 832 900])
yticklabels([220 317 414 511 609 706 803 900])
xlabel('Excitation wavelength (nm)')
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')
dA3=csvread('0.625ppm Nd in AABS spike _c_.csv',4,1);
dB3=csvread('1.25ppm Nd in AABS spike _c_.csv',4,1);
dC3=csvread('2.5ppm Nd in AABS spike _c_.csv',4,1);
%dD3=csvread('5ppm Nd in AABS spike _c_.csv',4,1);
dE3=csvread('10ppm Nd in AABS spike _c_.csv',4,1);
%
dF1=csvread('0ppm Nd in AABS spike _a_.csv',4,1);
dF2=csvread('0ppm Nd in AABS spike _b_.csv',4,1);
dF3=csvread('0ppm Nd in AABS spike _c_.csv',4,1);


dN1=csvread('DDH2O + 0.2M Acetic Acid (2mL in 10mL)-1.csv',4,1);
% dG1=csvread('2ppm Nd in AABS spike _a_.csv',4,1);
% dG2=csvread('2ppm Nd in AABS spike _b_.csv',4,1);
% dG3=csvread('2ppm Nd in AABS spike _c_.csv',4,1);
%
% dH1=csvread('Nd PostSorp _a_.csv',4,1);
% dH2=csvread('Nd PostSorp _b_.csv',4,1);
% dH3=csvread('Nd PostSorp _c_.csv',4,1);
%
% dI1=csvread('Nd Wash _a_.csv',4,1);
% dI2=csvread('Nd Wash _b_.csv',4,1);
% dI3=csvread('Nd Wash _c_.csv',4,1);
%
dJ1=csvread('5ppm Nd 5ppm Sm _a_.csv',4,1);
figure(12)
waterfall(dJ1)
xlabel('Excitation wavelength (nm)')
```

```
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')
xticklabels([220 288 356 424 492 560 628 696 764 832 900])
yticklabels([220 317 414 511 609 706 803 900])
% dJ2=csvread('5ppm Nd 5ppm Sm _b_.csv',4,1);

% Averaging Trials of Data at each Concentration
A=(dA1+dA2+dA3)/3;
% B=(dB1+dB2+dB3)/3;
C=(dC1+dC2+dC3)/3;
% D=(dD1+dD2)/2;
E=(dE1+Nd_E2+dE3)/3;
F=(dF1+dF2+dF3)/3;
% G=(dG1+dG2+dG3)/3;
% H=(dH1+dH2+dH3)/3;
% I=(dI1+dI2+dI3)/3;
% J=(dJ1+dJ2)/2;

% Plotting of Data
figure(1)
waterfall(A)
xticklabels([220 288 356 424 492 560 628 696 764 832 900])
yticklabels([220 317 414 511 609 706 803 900])
xlabel('Excitation wavelength (nm)')
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')


figure(2)
waterfall(F)
xticklabels([220 288 356 424 492 560 628 696 764 832 900])
yticklabels([220 317 414 511 609 706 803 900])
xlabel('Excitation wavelength (nm)')
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')


figure(3)
waterfall(C)
xticklabels([220 288 356 424 492 560 628 696 764 832 900])
yticklabels([220 317 414 511 609 706 803 900])
xlabel('Excitation wavelength (nm)')
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')

% hold on
% waterfall(B,'FaceColor','m')
% waterfall(C,'FaceColor','y')
% waterfall(D,'FaceColor','g')

figure(9)
waterfall(E)
xticklabels([220 288 356 424 492 560 628 696 764 832 900])
yticklabels([220 317 414 511 609 706 803 900])
```

```
xlabel('Excitation wavelength (nm)')
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')


% O=csvread('orange fluoro beads 2.csv',4,1);
% figure(10) %orange fluoro
% waterfall(O)
% xticklabels([220 288 356 424 492 560 628 696 764 832 900])
% yticklabels([220 317 414 511 609 706 803 900])
%
% P=csvread('orange fluoro beads dilute.csv',4,1);
% figure(11) %orange fluoro
% waterfall(P)
% %xticklabels([220 288 356 424 492 560 628 696 764 832 900])
% %yticklabels([220 317 414 511 609 706 803 900])

% waterfall(F)
% waterfall(G)
%
% hold off
% legend('A','B','C','D','E','F','G')
%
xlabel('Excitation wavelength (nm)')
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')

xticklabels([220 288 356 424 492 560 628 696 764 832 900])
yticklabels([220 317 414 511 609 706 803 900])

%
 SA=((dA1+dA2+dA3)/3)-F;
% SB=((B1+B2+B3)/3)-F;
% SC=((C1+C2+C3)/3)-F;
% SD=((D1+D2)/2)-F;
% SE=((E1+E2+E3)/3)-F;
% SG=((G1+G2+G3)/3)-F;
% SH=((H1+H2)/2)-F;
% SI=(I1-F);
 SF=(F-F);
% SJ=J-F;
%
% figure(5)
% waterfall(SA)
% xticklabels([220 288 356 424 492 560 628 696 764 832 900])
% yticklabels([220 317 414 511 609 706 803 900])
% xlabel('Excitation wavelength (nm)')
% ylabel('Emission wavelength (nm)')
% zlabel('Flourescence Intensity')

% S=csvread('SFP data.csv',4,1);
% figure(7)
% waterfall(S)
% xticklabels([220 288 356 424 492 560 628 696 764 832 900])
% yticklabels([220 317 414 511 609 706 803 900])
```

```
% xlabel('Excitation wavelength (nm)')
% ylabel('Emission wavelength (nm)')
% zlabel('Flourescence Intensity')

W=csvread('Water Test.csv',4,1);
figure(8)
waterfall(W)
xticklabels([220 288 356 424 492 560 628 696 764 832 900])
yticklabels([220 317 414 511 609 706 803 900])
xlabel('Excitation wavelength (nm)')
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')




figure(6)
waterfall(dN1)
% xticklabels([220 288 356 424 492 560 628 696 764 832 900])
% yticklabels([220 317 414 511 609 706 803 900])
xlabel('Excitation wavelength (nm)')
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')

% hold on
% waterfall(SB,'FaceColor','m')
% waterfall(SC,'FaceColor','y')
% waterfall(SD,'FaceColor','g')
% waterfall(SE,'FaceColor','b')
% waterfall(SF,'FaceColor','c')
% waterfall(SG,'FaceColor','k')
% hold off
% legend('zeroed A','zeroed B','zeroed C','zeroed D','zeroed E','zeroed F','zeroed G')
%
%
% A1=csvread('0.625ppm Sm in AABS spike _a_.csv',4,1);
% B1=csvread('1.25ppm Sm in AABS spike _a_.csv',4,1);
% C1=csvread('2.5ppm Sm in AABS spike _a_.csv',4,1);
% D1=csvread('5ppm Sm in AABS spike _a_.csv',4,1);
% E1=csvread('10ppm Sm in AABS spike _a_.csv',4,1);
% A2=csvread('0.625ppm Sm in AABS spike _b_.csv',4,1);
% B2=csvread('1.25ppm Sm in AABS spike _b_.csv',4,1);
% C2=csvread('2.5ppm Sm in AABS spike _b_.csv',4,1);
% D2=csvread('5ppm Sm in AABS spike _b_.csv',4,1);
Sm_E2=csvread('10ppm Sm in AABS spike _b_.csv',4,1);
figure(20)
waterfall(Sm_E2)
xticklabels([220 288 356 424 492 560 628 696 764 832 900])
yticklabels([220 317 414 511 609 706 803 900])
xlabel('Excitation wavelength (nm)')
ylabel('Emission wavelength (nm)')
zlabel('Flourescence Intensity')
```

```
% A3=csvread('0.625ppm Sm in AABS spike _c_.csv',4,1);
% B3=csvread('1.25ppm Sm in AABS spike _c_.csv',4,1);
% C3=csvread('2.5ppm Sm in AABS spike _c_.csv',4,1);
% D3=csvread('5ppm Sm in AABS spike _c_.csv',4,1);
% E3=csvread('10ppm Sm in AABS spike _c_.csv',4,1);
%
% F1=csvread('0ppm Nd in AABS spike _a_.csv',4,1);
% F2=csvread('0ppm Nd in AABS spike _b_.csv',4,1);
% F3=csvread('0ppm Nd in AABS spike _c_.csv',4,1);
%
% G1=csvread('2ppm Sm in AABS spike _a_.csv',4,1);
% G2=csvread('2ppm Sm in AABS spike _b_.csv',4,1);
% G3=csvread('2ppm Sm in AABS spike _c_.csv',4,1);
%
% H1=csvread('Sm PostSorp _a_.csv',4,1);
% H2=csvread('Sm PostSorp _b_.csv',4,1);
% %H3=csvread('Sm PostSorp _c_.csv',4,1);
%
% I1=csvread('Sm Wash _a_.csv',4,1);
% %I2=csvread('Sm Wash _b_.csv',4,1);
% %I3=csvread('Sm Wash _c_.csvH',4,1);
%
% %{
% H1=csvread('Nd PostSorp _a_.csv',4,1);
% H2=csvread('Nd PostSorp _b_.csv',4,1);
% H3=csvread('Nd PostSorp _c_.csv',4,1);
%
% I1=csvread('Nd Wash _a_.csv',4,1);
% I2=csvread('Nd Wash _b_.csv',4,1);
% I3=csvread('Nd Wash _c_.csv',4,1);
% %}
% % Averaging Trials of Data at each Concentration
% A=(A1+A2+A3)/3;
% B=(B1+B2+B3)/3;
% C=(C1+C2+C3)/3;
% D=(D1+D2+D3)/3;
% E=(E1+E2+E3)/3;
% F=(F1+F2+F3)/3;
% G=(G1+G2+G3)/3;
% H=(H1+H2)/2;
% I=I1;
%
% SAS=((A1+A2+A3)/3)-F;
% SBS=((B1+B2+B3)/3)-F;
% SCS=((C1+C2+C3)/3)-F;
% SDS=((D1+D2+D3)/3)-F;
% SES=((E1+E2+E3)/3)-F;
% SGS=((G1+G2+G3)/3)-F;
% SHS=((H1+H2)/2)-F;
% SIS=(I1-F);
% SFS=(F-F);
%
%
% figure(2)
% waterfall(SAS,'FaceColor','r')
% hold on
```

```
% waterfall(SBS,'FaceColor','m')
% waterfall(SCS,'FaceColor','y')
% waterfall(SDS,'FaceColor','g')
% waterfall(SES,'FaceColor','b')
% waterfall(SFS,'FaceColor','c')
% waterfall(SGS,'FaceColor','c')
% hold off
% legend('zeroed A','zeroed B','zeroed C','zeroed D','zeroed E','zeroed F','zeroed G')
%
%
% % Plotting of Data
% figure(4)
% waterfall(A,'FaceColor','r')
% hold on
% waterfall(B,'FaceColor','m')
% waterfall(C,'FaceColor','y')
% waterfall(D,'FaceColor','g')
% waterfall(E,'FaceColor','b')
% waterfall(F,'FaceColor','c')
% waterfall(G,'FaceColor','c')
% hold off
% legend('A','B','C','D','E','F','G')
%
% xlabel('Excitation wavelength (nm)')
% ylabel('Emission wavelength (nm)')
% zlabel('Flourescence Intensity')
%
%
% tic
%
% % Linear Correlation Determination
% for j=1:46
% if j<47
% for i=1:681
%     if i<681
% x=[0 .625 1.25 2]; % Initial four addresses before arsenazo changes emission
% y=[F(i,j) A(i,j) B(i,j) G(i,j)];
% linear_model=fitlm(x,y);
% R_Squared(i,j)=linear_model.Rsquared.Ordinary;
% i=i+1;
%
%     else
%     end
% end
% end
% j=j+1;
%
% end
%
% for k=1:46
% if k<47
% for l=1:681
%     if l<681
% x2=[2.5 5 10]; % Final four adresses after arsenazo changes emission
% y2=[C(l,k) D(l,k) E(l,k)];
% linear_model=fitlm(x2,y2);
```

```
% R2_Squared(l,k)=linear_model.Rsquared.Ordinary;
% l=l+1;
%
%     else
%     end
% end
% end
% k=k+1;
%
% end
%
% % Display Max R Squared Coefficient of Determination Value and location
%
% Matrix_R_Squared = sort(R_Squared,2,'descend');        % SORTING R^2'S
% Matri_R_Squared = sort(Matrix_R_Squared,1,'descend');
% Matrix_R2_Squared = sort(R2_Squared,2,'descend');        % SORTING R^2'S
% Matri_R2_Squared = sort(Matrix_R2_Squared,1,'descend');
%
% R1 = Matri_R_Squared(1);                    % DEFINES BEST R^2
% R2 = Matri_R_Squared(2);                    % AND 2ND BEST
% R3 = Matri_R_Squared(3);                    % AND 3RD for low concentration
% R5 = Matri_R_Squared(5);
% r1 = Matri_R2_Squared(1);
% r2 = Matri_R2_Squared(2);                    % high concentration (top 2 r^2 values)
%
% Max_R_Squared1=max(R_Squared);                % FINDING BEST R^2 (REDUNDANT)
% Max_R_Squared=max(Max_R_Squared1');
% Max_R2_Squared1=max(R_Squared);
% Max_R2_Squared=max(Max_R2_Squared1');
%
% if Max_R_Squared < 0.99                    % IF YOU GOT R SQUARED LESS THAN 0.99
%     disp('No good fit for R squared > 0.99')    % YOU'LL GET A MESSAGE
% else
% end
%
% if Max_R_Squared ~= R1                    % Check R1 is max value
%     disp('Error. Need to change code for Matri_R_Squared its not sorting')
% else
% end
%
% [i,j]=find(R_Squared==R1);
% Location_1=[i j] ;                    % SHOWS LOCATION (X,Y) OF BEST R^2
% Exitation_Wavelength_1=i+219;
% Emission_Wavelength_1=15.111*j+204.88;
% a=i;
% d=j;
%
% [l,k]=find(R2_Squared==r1);
% Location_a=[l,k];
% Excitation_Wavelength_a=l+219;
% Emission_Wavelength_a=15.111*k+204.88;
% aa=l;
% bb=k;
%
% [l,k]=find(R2_Squared==r2);
% Location_b=[l,k];
```

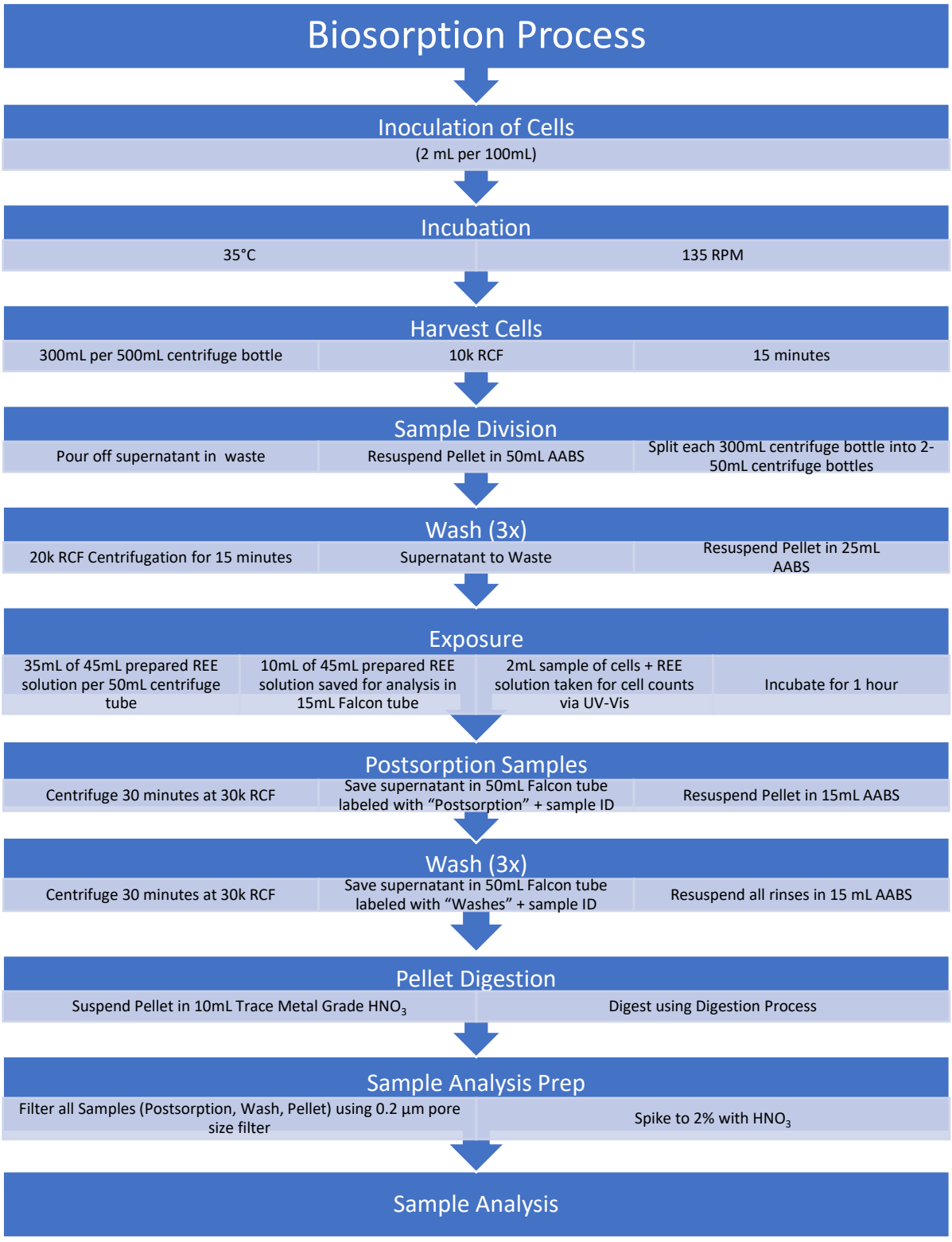
```
% Excitation_Wavelength_b=l+219;
% Emission_Wavelength_b=15.111*k+204.88;
% cc=l;
% dd=k;
%
% [i,j]=find(R_Squared==R2);
% Location_2=[i j];
% Exitation_Wavelength_2=i+219;
% Emission_Wavelength_2=15.111*j+204.88;
% b=i;
% e=j;
%
% [i,j]=find(R_Squared==R3);
% Location_3=[i j];
% Exitation_Wavelength_3=i+219;
% Emission_Wavelength_3=15.111*j+204.88;
% c=i;
% f=j;
%
% [i,j]=find(R_Squared==R5);
% Location_5=[i j];
% Exitation_Wavelength_5=i+219;
% Emission_Wavelength_5=15.111*j+204.88;
% k=i;
% m=j;
%
% Mat_Eu_Ars_DDH2O = [a d R1; b e R2] %c f R3; g h R4; k m R5]
% Matrix_Eu_Ars_DDH2O = [aa bb r1; cc dd r2]
%
% A2=A(:,d);  % .625ppm              % Makes a new matrix holding the excitation constant
% A3=A(a,:);              % Makes a new matrix holding the emission constant
% B2=B(:,d);  % 1.25ppm
% B3=B(a,:);
% C2=C(:,bb);  % 2.5ppm
% C3=C(aa,:);
% D2=D(:,bb);   % 5 ppm
% D3=D(aa,:);
% E2=E(:,bb);   % 10 ppm
% E3=E(aa,:);
% F2=F(:,d);  % 0ppm
% F3=F(a,:);
% G2=G(:,d);  % 2ppm
% G3=G(a,:);
%
% ex=1:46;                % Defines columns of A,B,C etc matrix
% wy=1:681;                % Defines rows of A,B,C etc matrix
% wi=(1./wy).*10000000;
% wa=wi';
% wp=fliplr(wa');
% wt=[wp(:,1) wp(:,100) wp(:,200) wp(:,300) wp(:,400) wp(:,500) wp(:,600)];
% AF=[A2 B2 C2 D2 E2 F2 G2];
% AG=[A3 B3 C3 D3 E3 F3 G3];
%
% figure(2)
% plot(wy,F2,wy,A2,wy,B2,wy,D2,wy,G2,wy,E2,wy,C2)  % Plots absorbance vs exitation
% legend('F','A','B','D','G','E','C')
```

```
% xlabel('Emission')
% ylabel('Absorbance')
%
% %{
% ax1=gca;
% ax1.XColor='r';
% ax1.YColor='r';
% ax1_pos=ax1.Position;
% ax2=axes('Position',ax1_pos,'XAxisLocation','top','YAxisLocation','right','Color','none');
% line(wt,0,'Parent',ax2,'Color','k')
% legend('baseline')
% % This is trying to plot wavenumber and emission vs absorbance
% %}
%
% figure(3)
% plot(ex,F3,ex,A3,ex,B3,ex,G3)  % Plots absorbance vs emission low concentration
% hold on
% legend('F','A','B','G')
% plot(ex,C3,ex,D3,ex,E3)  % high concentrations
% legend('C','D','E')
% hold off
% xlabel('Excitation')
% ylabel('Absorbance')
%
% figure(4)
% A4=A(a,d);  % low concentration points
% B4=B(a,d);
% F4=F(a,d);
% G4=G(a,d);
% H4=H(a,d);
% I4=I(a,d);
% C4=C(aa,bb); % high concentration points
% D4=D(aa,bb);
% E4=E(aa,bb);
% J4=J(a,d);  %Test point low concentration
% J5=J(aa,bb); %Test point high concentration
%
% A5=A(cc,dd);  % The A5-E5 points are not useful points. They are the low concentration values at the
% B5=B(cc,dd);  % high concentration addresses and vice versa. They are included in the plot to show
% F5=F(cc,dd);  % how points deviate from linearity as concetrations change
% G5=G(cc,dd);
% H5=H(cc,dd);
% I5=I(cc,dd);
% C5=C(cc,dd);
% D5=D(cc,dd);
% E5=E(cc,dd);
%
%
% x=[0 .625 1.25 2 2.5 5 7 ];   % These matrices allow a scatterplot of
% y=[F4 A4 B4 G4 C5 D5 H5 ];     % points at 2 different adresses 7 and 8 are post sorp and wash
% xx=[2.5 5 10 .625 1.25 0 2];
% yy=[C4 D4 E4 A5 B5 F5 G5];
%
% xa=[0 .625 1.25 2];    % These values are for the first 4 and 3 points
% ya=[F4 A4 B4 G4];
% xxa=[2.5 5 10];
```

```
% yya=[C4 D4 E4];
%
% [mean, ~, confidence_bounds_95, ~] = normfit(ya);
% low=confidence_bounds_95(1)
% high=confidence_bounds_95(2)
% [mean, ~, confidence_bounds_95, ~] = normfit(yya);
% low2=confidence_bounds_95(1)
% high2=confidence_bounds_95(2)
%
% scatter(x,y,25,'b') % creates scatter plot of concentration vs absorbance
% hold on
% scatter(xx,yy,25,'c')
% Q=polyfit(xxa,yya,1);
% yyfit=Q(1)*xxa+Q(2);
% plot(xxa,yyfit,'r-.');
% P=polyfit(xa,ya,1);
% yfit=P(1)*xa+P(2);
% plot(xa,yfit,'c-.');
% xlabel('Concentration')
% ylabel('Absorbance')
% ylim([-100 600]);
% m=P(1);
% b=P(2);
% mm=Q(1);
% bb=Q(2);
% Y2= ['y2 =', num2str(mm),'*x +', num2str(bb)];
% Y = ['y =', num2str(m), '*x +', num2str(b)]; % display the equation of the line
% disp('low concentration line')
% disp(Y)
% disp('high concentration line')
% disp(Y2)
%
% %{
% concentration=[0; .625; 1.25; 2; 2.5; 5; 10];
% absorbance=[F4; A4; B4; G4; C5; D5; E5];
% Table=table(concentration,absorbance);
% writetable(Table,'Nd_with_Arszo_AABS_spike_ppm.xlsx')
%
%
% figure(5)
% plot(wy,A,wy,confidence_bounds_95)
% legend('A','bounds')
% xlabel('Emission')
% ylabel('Absorbance')
% %}
% toc
% %}
```

Appendix B: Supporting Materials for Biosorption Experiments

B.1: Biosorption Experiment PFD



B.2: Sample Cell Counts Image

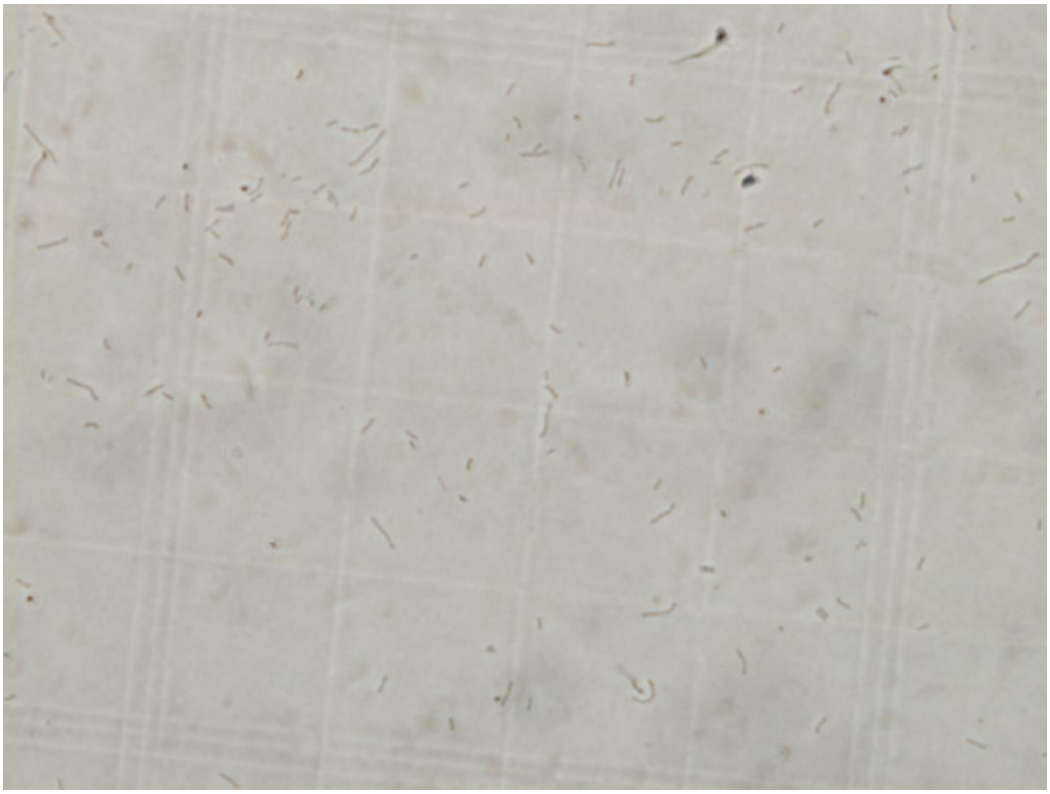


Figure B2: Sample image of C. necator on a cell counting grid used for cell concentration determination

B.3: Cell Calibration Curve

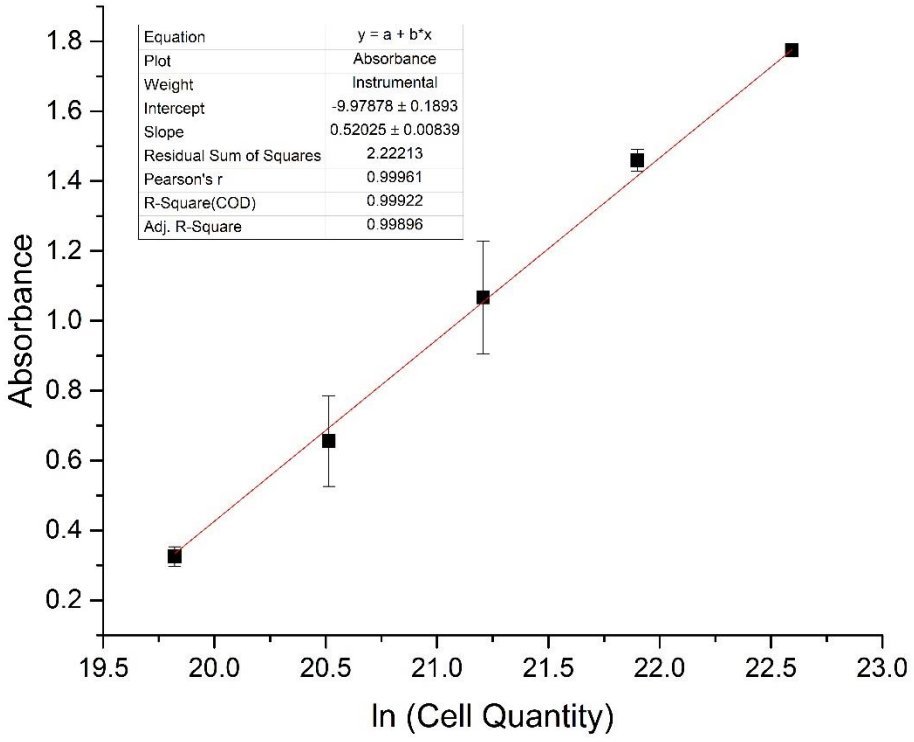


Figure B3: Cell Calibration Curve used to determine cell counts in biosorption experiments.

B.4 VMINTEQ Tables

Table B.1 Saturation Indices for Minerals for Europium

Mineral	log IAP	Sat. Index	Stoichiometry and Mineral Components									
		(=log IAP - log Ks)										
Eu(OH)3(s)	9.868	-5.622	1	Eu+3	-3	H+1	3	H2O	-4	E-1	6	H2O
Halite	-1.497	-3.047	1	Na+1	1	Cl-1	-2	H+1	1	H2O	1	H2O

Table B.2: Concentrations and activities of aqueous inorganic species (mol/L)

	Concentration	Activity	Log activity
Acetate-1	2.62E-01	1.90E-01	-0.72
Cl-1	1.36E-01	9.91E-02	-1.004
Eu-(Acetate)2+	2.01E-02	1.46E-02	-1.835
Eu-(Acetate)3 (aq)	1.63E-02	1.80E-02	-1.745
Eu+3	1.39E-04	7.86E-06	-5.105
Eu-Acetate+2	3.47E-03	9.67E-04	-3.015
EuCl+2	1.04E-05	2.89E-06	-5.539
EuOH+2	8.67E-08	2.42E-08	-7.617
H+1	1.38E-05	1.00E-05	-5
H-Acetate (aq)	9.90E-02	1.09E-01	-0.961
Na+1	4.40E-01	3.20E-01	-0.495
Na-Acetate (aq)	4.64E-02	5.13E-02	-1.29
NaCl (aq)	1.29E-02	1.43E-02	-1.844
NaOH (aq)	7.86E-10	8.69E-10	-9.061
OH-	2.82E-09	2.05E-09	-8.689

B.5 Digestion Block Controller Arduino Code

```
#include "DueTimer.h"

int count=0;
int onoff=0;
int heaterOn=0;

#define PINEN 7 //Mux Enable pin
#define PINA0 4 //Mux Address 0 pin
#define PINA1 5 //Mux Address 1 pin
#define PINA2 6 //Mux Address 2 pin
#define PINSO 12 //TCAmp Slave Out pin (MISO)
#define PINSC 13 //TCAmp Serial Clock (SCK)
#define PINCS 9  //TCAmp Chip Select Change this to match the position of the Chip Select Link

#define PINPWR 8 // Solid State relay On/off
#define PINLED 11 // LED for Solid State relay monitoring
#define PINPWR2 3 // Solid State relay On/off #2
#define PINLED2 2 // LED for Solid State relay #2 monitoring

int SensorFail[8];
float floatTemp, floatInternalTemp;
char failMode[8];
int internalTemp, intTempFrac;
unsigned int Mask;
//char data[16];
unsigned char i, j, k, NumSensors =8, UpdateDelay=2;  //delay in seconds
char Rxchar, Rxenable, Rxptr, Cmdcomplete, R;
char Rxbuf[32];
char adrbuf[3], cmdbuf[3], valbuf[12];
int val = 0, Param;
unsigned long int timems=0;
unsigned short int T16, mask;
```

```
String ts="01234567890123456789012345678901";

int freq = 2;
int oldFreq=2;
int ssrOnOff=1;

int freq2= 2;
int oldFreq2=2;
int ssrOnOff2=1;


char command[10];
char instr[3];
int inbyte;
int chan;
unsigned int value;
unsigned int value2;

unsigned int hex2int(char *a, unsigned int len)
{
  int i;
  unsigned int val = 0;

  for(i=0;i<len;i++)
    if(a[i] <= 57)
     val += (a[i]-48)*(1<<(4*(len-1-i)));
    else
     val += (a[i]-87)*(1<<(4*(len-1-i))); //lower case hex. use -55 instead of -87 for upper case hex.
  return val;
}

void firstHandler(){
 //Serial.println("[-  ] First Handler!");
```

```
}

void secondHandler(){
  //Serial.println("[ - ] Second Handler!");
  digitalWrite(PINPWR, LOW);
  digitalWrite(PINLED, LOW);
  digitalWrite(PINPWR2, LOW);
  digitalWrite(PINLED2, LOW);
  Timer4.stop();
  if(heaterOn==1){
    digitalWrite(PINPWR2, HIGH);
    digitalWrite(PINLED2, HIGH);
    heaterOn=2;
    Timer4.setFrequency(2048/(double)freq2);
    Timer4.start();

  }
}

void thirdHandler(){
  //Serial.println("[  -] Third Handler!");
      digitalWrite(PINPWR, HIGH);
      digitalWrite(PINLED, HIGH);


  ssrOnOff=1-ssrOnOff;
  Timer4.start();



        Timer4.stop();
        Timer4.setFrequency(2048/(double)freq);
        Timer4.start();
```

```
        oldFreq=freq;

        heaterOn=1;

}



void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

  pinMode(PINEN, OUTPUT);
  pinMode(PINA0, OUTPUT);
  pinMode(PINA1, OUTPUT);
  pinMode(PINA2, OUTPUT);
  pinMode(PINSO, INPUT);
  pinMode(PINCS, OUTPUT);
  pinMode(PINSC, OUTPUT);
  pinMode(PINPWR, OUTPUT);
  pinMode(PINLED, OUTPUT);
  pinMode(PINPWR2, OUTPUT);
  pinMode(PINLED2, OUTPUT);

  digitalWrite(PINEN, HIGH);   // enable on
  digitalWrite(PINA0, LOW); // low, low, low = channel 1
  digitalWrite(PINA1, LOW);
  digitalWrite(PINA2, LOW);
  digitalWrite(PINSC, LOW); //put clock in low

 digitalWrite(PINPWR, LOW);
 digitalWrite(PINLED, LOW);
 digitalWrite(PINPWR2, LOW);
```

```
 digitalWrite(PINLED2, LOW);

 Timer5.attachInterrupt(thirdHandler).setFrequency(0.5);
 Timer5.start();
 Timer4.attachInterrupt(secondHandler).setFrequency(1);

}

void loop()
{
//millis() returns the number of ms since starting sketch
  if (millis() > (timems + ((unsigned int)UpdateDelay*1000)))
  {
    timems = millis();
//    Serial.print("N=");
//    Serial.println(count,DEC);
    count++;


if(Serial.available()>6)
  {
  k=0;
  while(Serial.available()>0)
    {
    inbyte=Serial.read();
    if(inbyte != 13 && inbyte != 10)
        command[k++]=(char)inbyte;
    }
   command[k]=0;
   Serial.println(command);
   instr[0]=command[0];
   instr[1]=command[1];
   instr[2]=0;
```

```
if(strcmp(instr,"SF")==0)  //set heater #1
    {
    chan=command[2]-48;
    value=hex2int(command+3,3);  // convert 3 lower case hex digits to integer
    if(chan==1)
      {
       freq=value;

      }
    }
 if(strcmp(instr,"HT")==0)  //set heater #2
    {
    chan=command[2]-48;
    value2=hex2int(command+3,3);  // convert 3 lower case hex digits to integer
    if(chan==1)
      {
       freq2=value2;

      }
    }
  }



  Serial.println("T8");
  for(j=0;j<8;j++)
    {
    switch (j) //select channel
      {
      case 0:
        digitalWrite(PINA0, LOW);
        digitalWrite(PINA1, LOW);
```

```
  digitalWrite(PINA2, LOW);
break;
case 1:
  digitalWrite(PINA0, HIGH);
  digitalWrite(PINA1, LOW);
  digitalWrite(PINA2, LOW);
break;
case 2:
  digitalWrite(PINA0, LOW);
  digitalWrite(PINA1, HIGH);
  digitalWrite(PINA2, LOW);
break;
case 3:
  digitalWrite(PINA0, HIGH);
  digitalWrite(PINA1, HIGH);
  digitalWrite(PINA2, LOW);
break;
case 4:
  digitalWrite(PINA0, LOW);
  digitalWrite(PINA1, LOW);
  digitalWrite(PINA2, HIGH);
break;
case 5:
  digitalWrite(PINA0, HIGH);
  digitalWrite(PINA1, LOW);
  digitalWrite(PINA2, HIGH);
break;
case 6:
  digitalWrite(PINA0, LOW);
  digitalWrite(PINA1, HIGH);
  digitalWrite(PINA2, HIGH);
break;
case 7:
```

```
  digitalWrite(PINA0, HIGH);
  digitalWrite(PINA1, HIGH);
  digitalWrite(PINA2, HIGH);
 break;
 }

delay(5);
digitalWrite(PINCS, LOW); //stop conversion
delay(5);
digitalWrite(PINCS, HIGH); //begin conversion
delay(100);  //wait 100 ms for conversion to complete
digitalWrite(PINCS, LOW); //stop conversion, start serial interface
delay(1);
 /*
 Temp[j] = 0;
 failMode[j] = 0;
 SensorFail[j] = 0;
 internalTemp = 0;
 */

T16=0;
for (k=0;k<32;k++)   //for (i=31;i>=0;i--)
  {
  i=31-k;
  digitalWrite(PINSC, HIGH);
  delay(1);

  if ((i<=31) && (i>=18))
     {
     // these 14 bits are the thermocouple temperature data
     // bit 31 sign
     // bit 30 MSB = 2^10
     // bit 18 LSB = 2^-2 (0.25 degC)
```

```
          Mask = 1<<(i-18);
          if (digitalRead(PINSO)==1)
           {
           if (i == 31)
            {
            T16 += (0b11<<14);//pad the temp with the bit 31 value so we can read negative values correctly
            }

           T16 += Mask;
           }
          }
       digitalWrite(PINSC, LOW);
       delay(1);
       }  //end of loop over bits
       T16= T16 | 0x4000; // set bit 14 to a 1,   Bit 15 will be a 0 for positive 1 for negative
//       Temp[j]=T16;
      if(j==0)
       ts=String(T16,HEX);
      else
       {
       ts=String(ts + String(T16,HEX));
       }
     } //end of j  loop over sensors
 Serial.println(ts);
 }//end of delay
}  //end of program  loop
```

B.6 Biosorption Isotherm MATLAB Sample Code

```
%This program determines the Langmuir and Freundlich Parameters and
%profiles of biosorption of Europium by C. Necator

clear all
clc


options=optimoptions(@lsqcurvefit,'MaxFunctionEvaluations',1000000,'Algorithm','levenberg-marquardt');
lbs=[0,0,0];% [Keq,n,Qmax]
ubs=[inf,inf,inf];
lbl=[0,0];
ubl=[100,500];

tic
MW_Eu=151.9641; %MW of Europium
MW_Nd=144.24; %MW of Neodymium
MW_Sm=150.36; %MW of Samarium

%Data Sorting
[data,txt,raw]=xlsread('MP-AES Eu data compiled.xlsx'); %import data from Excel file
raw(1:2,:)=[]; %Delete 1st two rows
T=cell2table(raw(2:end,:)); %convert to a table excluding 1st row of raw data
T.Properties.VariableNames=raw(1,:); %Assign Labels to Table using 1st row of raw data
T2=T;%Copy Table T for modification
T2(:,[2,4,7,8,11,12,13,14,16,23,24,25])=[];%Remove unneeded columns to analysis
T3=sortrows(T2,'Type'); %Create new table sorted by Type
T3([1:250],:)=[];% Remove rows containing Standards and Blanks
T3=sortrows(T3,'Element');
Eu_381=sortrows(T3,'Element'); %Create new table sorted by Elemental Wavelength
Eu_381([216:end],:)=[]; %Delete all data except those at Wavelength 381.967 nm
Eu_412=sortrows(T3,'Element'); %Create new table sorted by Elemental Wavelength
Eu_412([1:215,430:end],:)=[]; %Delete all data except those at Wavelength 412.973 nm
Eu_420=sortrows(T3,'Element'); %Create new table sorted by Elemental Wavelength
Eu_420([1:431,645:end],:)=[]; %Delete all data except those at Wavelength 412.973 nm

%Initials for Eu @ 381nm
Eu_381=sortrows(Eu_381,'Label'); %Sort by Label
Eu_381Init=Eu_381; %Create Initial Table for Eu ate 381.967nm containing all values from Eu_381 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,~]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,~]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Pellet and Postsorption
samples
pattern='2%'; % Data set Want to remove identified in Label Column
```

```
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Pellet and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
Cell Conc and Stock samples

%Initials for Eu @412nm
Eu_412Init=Eu_412; %Create Initial Table for Eu ate 412.973nm containing all values from Eu_381 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Pellet and Postsorption
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Pellet and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash,Pellet, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 412.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Initials for Eu @420nm
Eu_420Init=Eu_420; %Create Initial Table for Eu ate 420.505nm containing all values from Eu_381 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Pellet and Postsorption samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Pellet and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,~]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Pellet and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash, Pellet, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash,Pellet, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Postsorption for Eu @ 381nm
Eu_381=sortrows(Eu_381,'Label'); %Sort by Label
Eu_381Post=Eu_381; %Create Postial Table for Eu ate 381.967nm containing all values from Eu_381 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash samples
```

```
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Pellet samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Pellet and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
Cell Conc and Stock samples

%Postsorption for Eu @412nm
Eu_412Post=Eu_412; %Create Postial Table for Eu ate 412.973nm containing all values from Eu_381 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Pellet samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Pellet and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Pellet and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash, Pellet, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash,Pellet, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 412.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Postsorption for Eu @420nm
Eu_420Post=Eu_420; %Create Postial Table for Eu ate 420.505nm containing all values from Eu_381 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Pellet samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Pellet and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Pellet and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Pellet and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash, Pellet, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash,Pellet, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Pellet for Eu @ 381nm
Eu_381=sortrows(Eu_381,'Label'); %Sort by Label
Eu_381Pellet=Eu_381; %Create Pelletial Table for Eu ate 381.967nm containing all values from Eu_381 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Postsorption and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash and Postsorption and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash, Postsorption, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Pellet for Eu @412nm
Eu_412Pellet=Eu_412; %Create Pelletial Table for Eu ate 412.973nm containing all values from Eu_381 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Postsorption and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash and Postsorption and HNO3
and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash, Postsorption, 2% HNO3,
Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Wash,Postsorption, 2% HNO3,
Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 412.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples


%Pellet for Eu @420nm
Eu_420Pellet=Eu_420; %Create Pelletial Table for Eu ate 420.505nm containing all values from Eu_381
Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Postsorption and 2%HNO3
samples
```

```
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash and Postsorption and HNO3
and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash, Postsorption, 2% HNO3,
Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Wash,Postsorption, 2% HNO3,
Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Wash for Eu @ 381nm
Eu_381=sortrows(Eu_381,'Label'); %Sort by Label
Eu_381Wash=Eu_381; %Create Washial Table for Eu ate 381.967nm containing all values from Eu_381
Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet and Postsorption and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet and Postsorption and HNO3
and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet, Postsorption, 2% HNO3,
Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet,Postsorption, 2% HNO3,
Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Wash,Wash, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Wash for Eu @412nm
Eu_412Wash=Eu_412; %Create Washial Table for Eu ate 412.973nm containing all values from Eu_381
Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet and Postsorption and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet and Postsorption and HNO3
and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet, Postsorption, 2% HNO3,
Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet,Postsorption, 2% HNO3,
Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 412.967 excluding Wash,Wash, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Wash for Eu @420nm
Eu_420Wash=Eu_420; %Create Washial Table for Eu ate 420.505nm containing all values from Eu_381
Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet and Postsorption and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet and Postsorption and HNO3
and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet, Postsorption, 2% HNO3,
Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet,Postsorption, 2% HNO3,
Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.967 excluding Wash,Wash, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Filter for Eu @ 381nm
Eu_381=sortrows(Eu_381,'Label'); %Sort by Label
Eu_381Filter=Eu_381; %Create Filterial Table for Eu ate 381.967nm containing all values from Eu_381 Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet and Postsorption and 2%HNO3 samples
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet and Postsorption and HNO3 and Wash samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet, Postsorption, 2% HNO3, Wash and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Pellet,Postsorption, 2% HNO3, Wash, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 381.967 excluding Filter,Filter, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Filter for Eu @412nm
Eu_412Filter=Eu_412; %Create Filterial Table for Eu ate 412.973nm containing all values from Eu_381 Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet and Postsorption and 2%HNO3 samples
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet and Postsorption and HNO3 and Wash samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet, Postsorption, 2% HNO3, Wash and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 412.973 excluding Pellet,Postsorption, 2% HNO3,
Wash, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 412.967 excluding Filter,Filter, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Filter for Eu @420nm
Eu_420Filter=Eu_420; %Create Filterial Table for Eu ate 420.505nm containing all values from Eu_381 Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet and Postsorption and 2%HNO3
samples
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet and Postsorption and HNO3
and Wash samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet, Postsorption, 2% HNO3,
Wash and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pellet,Postsorption, 2% HNO3,
Wash, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 412.967 excluding Filter,Filter, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Removal of Spikes and Duplicates where applicable
Eu_381Filter_Spikes_Dups=Eu_381Filter;
Eu_412Filter_Spikes_Dups=Eu_412Filter;
Eu_420Filter_Spikes_Dups=Eu_420Filter;
Eu_381Wash_Spikes_Dups=Eu_381Wash;
Eu_412Wash_Spikes_Dups=Eu_412Wash;
```

```
Eu_420Wash_Spikes_Dups=Eu_420Wash;
Eu_381Post_Spikes_Dups=Eu_381Post;
Eu_412Post_Spikes_Dups=Eu_412Post;
Eu_420Post_Spikes_Dups=Eu_420Post;
Eu_381Pellet_Spikes_Dups=Eu_381Pellet;
Eu_412Pellet_Spikes_Dups=Eu_412Pellet;
Eu_420Pellet_Spikes_Dups=Eu_420Pellet;

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_412Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_420Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Filter(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates.

pattern='Rerun'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,~]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates

pattern='Rerun'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Duplicates

% Remove Samples Redone
pattern='100uM Pellet "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates

pattern='100uM Postsorption "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates

pattern='100uM Wash "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates

pattern='100uM Initial "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='80uM Pellet "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates

pattern='80uM Postsorption "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates

pattern='80uM Wash "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates

pattern='80uM Initial "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='60uM Pellet "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates

pattern='60uM Postsorption "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates

pattern='60uM Wash "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates

pattern='60uM Initial "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='40uM Pellet "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates

pattern='40uM Postsorption "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates

pattern='40um Wash "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates

pattern='40uM Initial "A"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='100uM Pellet "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates

pattern='100uM Postsorption "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates

pattern='100uM Wash "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates

pattern='100uM Initial "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='80uM Pellet "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates

pattern='80uM Postsorption "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates

pattern='80uM Wash "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates

pattern='80uM Initial "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='60uM Pellet "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates

pattern='60uM Postsorption "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates

pattern='60uM Wash "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates

pattern='60uM Initial "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='40uM Pellet "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_412Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates
TF=contains(Eu_420Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Pellet(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Pelletlicates

pattern='40uM Postsorption "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_412Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_412Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates
TF=contains(Eu_420Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Post(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Postsorptionlicates

pattern='40um Wash "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_412Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Eu_412Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates
TF=contains(Eu_420Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Wash(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Washlicates

pattern='40uM Initial "B"'; % Data set Want to remove identified in Label Column
TF=contains(Eu_381Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_381Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_412Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True stringsEu_381In
Eu_412Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(Eu_420Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Eu_420Init(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

%Sort Rows
Eu_381Init=Eu_381Init([15 16 13 14 19 20 17 18 23 24 21 22 3 4 1 2 5:12],:);
Eu_412Init=Eu_412Init([17 18 1 2 19 20 3 4 21 22 5 6 23 24 7 8 9:16],:);
Eu_420Init=Eu_420Init([17 18 1 2 19 20 3 4 21 22 5 6 23 24 7 8 9:16],:);
Eu_381Post=Eu_381Post([15 16 13 14 19 20 17 18 23 24 21 22 3 4 1 2 5:12],:);
Eu_412Post=Eu_412Post([17 18 1 2 19 20 3 4 21 22 5 6 23 24 7 8 9:16],:);
Eu_420Post=Eu_420Post([17 18 1 2 19 20 3 4 21 22 5 6 23 24 7 8 9:16],:);
Eu_381Pellet=Eu_381Pellet([15 16 13 14 19 20 17 18 23 24 21 22 3 4 1 2 5:12],:);
Eu_412Pellet=Eu_412Pellet([17 18 1 2 19 20 3 4 21 22 5 6 23 24 7 8 9:16],:);
Eu_420Pellet=Eu_420Pellet([17 18 1 2 19 20 3 4 21 22 5 6 23 24 7 8 9:16],:);
Eu_381Wash=Eu_381Wash([15 16 13 14 19 20 17 18 23 24 21 22 3 4 1 2 5:12],:);
Eu_412Wash=Eu_412Wash([17 18 1 2 19 20 3 4 21 22 5 6 23 24 7 8 9:16],:);
Eu_420Wash=Eu_420Wash([17 18 1 2 19 20 3 4 21 22 5 6 23 24 7 8 9:16],:);

%Convert Table Columns to Array For Calculations
Eu_381PostConc=table2array(Eu_381Post(:,5));
Eu_381WashConc=table2array(Eu_381Wash(:,5));
Eu_381InitConc=table2array(Eu_381Init(:,5));
Eu_381PelletConc=table2array(Eu_381Pellet(:,5));
Eu_412PostConc=table2array(Eu_412Post(:,5));
Eu_412WashConc=table2array(Eu_412Wash(:,5));
Eu_412InitConc=table2array(Eu_412Init(:,5));
Eu_412PelletConc=table2array(Eu_412Pellet(:,5));
Eu_420PostConc=table2array(Eu_420Post(:,5));
Eu_420WashConc=table2array(Eu_420Wash(:,5));
Eu_420InitConc=table2array(Eu_420Init(:,5));
Eu_420PelletConc=table2array(Eu_420Pellet(:,5));

Eu_381PostConcppm=cell2mat(Eu_381PostConc);
Eu_381WashConcppm=cell2mat(Eu_381WashConc);
Eu_381InitConcppm=cell2mat(Eu_381InitConc);
Eu_381PelletConcppm=cell2mat(Eu_381PelletConc);
Eu_412PostConcppm=cell2mat(Eu_412PostConc);
Eu_412WashConcppm=cell2mat(Eu_412WashConc);
Eu_412InitConcppm=cell2mat(Eu_412InitConc);
Eu_412PelletConcppm=cell2mat(Eu_412PelletConc);
Eu_420PostConcppm=cell2mat(Eu_420PostConc);
Eu_420WashConcppm=cell2mat(Eu_420WashConc);
```

```
Eu_420InitConcppm=cell2mat(Eu_420InitConc);
Eu_420PelletConcppm=cell2mat(Eu_420PelletConc);

%Convert ppm to uM basis
Eu_381PostConc=Eu_381PostConcppm*MW_Eu;
Eu_381WashConc=Eu_381WashConcppm*MW_Eu;
Eu_381InitConc=Eu_381InitConcppm*MW_Eu;
Eu_381PelletConc=Eu_381PelletConcppm*MW_Eu;
Eu_412PostConc=Eu_412PostConcppm*MW_Eu;
Eu_412WashConc=Eu_412WashConcppm*MW_Eu;
Eu_412InitConc=Eu_412InitConcppm*MW_Eu;
Eu_412PelletConc=Eu_412PelletConcppm*MW_Eu;
Eu_420PostConc=Eu_420PostConcppm*MW_Eu;
Eu_420WashConc=Eu_420WashConcppm*MW_Eu;
Eu_420InitConc=Eu_420InitConcppm*MW_Eu;
Eu_420PelletConc=Eu_420PelletConcppm*MW_Eu;

Eu_381Postumol=Eu_381PostConc*35/1000;
Eu_381Washumol=Eu_381WashConc*45/1000;
Eu_381Pelletumol=Eu_381PelletConc*30/1000;
Eu_381Initumol=Eu_381InitConc*35/1000;
Eu_412Postumol=Eu_412PostConc*35/1000;
Eu_412Washumol=Eu_412WashConc*45/1000;
Eu_412Pelletumol=Eu_412PelletConc*30/1000;
Eu_412Initumol=Eu_412InitConc*35/1000;
Eu_420Postumol=Eu_420PostConc*35/1000;
Eu_420Washumol=Eu_420WashConc*45/1000;
Eu_420Pelletumol=Eu_420PelletConc*30/1000;
Eu_420Initumol=Eu_420InitConc*35/1000;

%Material Balances
Eu_381MatB=Eu_381Initumol-Eu_381Postumol-Eu_381Washumol-Eu_381Pelletumol;
Eu_412MatB=Eu_412Initumol-Eu_412Postumol-Eu_412Washumol-Eu_412Pelletumol;
Eu_420MatB=Eu_420Initumol-Eu_420Postumol-Eu_420Washumol-Eu_420Pelletumol;

%Percent Recovery
PercentRecov_Eu381=(Eu_381Postumol+Eu_381Washumol+Eu_381Pelletumol)./Eu_381Initumol;
PercentRecov_Eu412=(Eu_412Postumol+Eu_412Washumol+Eu_412Pelletumol)./Eu_412Initumol;
PercentRecov_Eu420=(Eu_420Postumol+Eu_420Washumol+Eu_420Pelletumol)./Eu_420Initumol;

%Langmuir Isotherm per billion cell basis
Tcells=cell2table(raw([888:end],:)); %convert to a table excluding 1st row of raw data
Tcells.Properties.VariableNames=raw(1,:); %Assign Labels to Table using 1st row of raw data
Tcells(:,[2,4,6:28])=[];%Remove unneeded columns to analysis
CellConc_Eu381=sortrows(Tcells,'Element'); %Create new table sorted by Elemental Wavelength
CellConc_Eu381([33:end],:)=[]; %Delete all data except those at Wavelength 381.967 nm
CellConc_Eu412=sortrows(Tcells,'Element'); %Create new table sorted by Elemental Wavelength
CellConc_Eu412([1:32,65:end],:)=[]; %Delete all data except those at Wavelength 412 nm
CellConc_Eu420=sortrows(Tcells,'Element'); %Create new table sorted by Elemental Wavelength
CellConc_Eu420([1:64,97:end],:)=[]; %Delete all data except those at Wavelength 412 nm

%Removal of Cell Conc for those redone

pattern='40uM "A"'; % Data set Want to remove identified in Label Column
```

```
TF=contains(CellConc_Eu381.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu381(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu412.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True stringsEu_381In
CellConc_Eu412(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu420.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu420(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='60uM "A"'; % Data set Want to remove identified in Label Column
TF=contains(CellConc_Eu381.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu381(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu412.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True stringsEu_381In
CellConc_Eu412(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu420.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu420(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='80uM "A"'; % Data set Want to remove identified in Label Column
TF=contains(CellConc_Eu381.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu381(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu412.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True stringsEu_381In
CellConc_Eu412(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu420.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu420(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='100uM "A"'; % Data set Want to remove identified in Label Column
TF=contains(CellConc_Eu381.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu381(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu412.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True stringsEu_381In
CellConc_Eu412(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu420.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu420(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='40uM "B"'; % Data set Want to remove identified in Label Column
TF=contains(CellConc_Eu381.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu381(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu412.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True stringsEu_381In
CellConc_Eu412(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu420.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu420(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='60uM "B"'; % Data set Want to remove identified in Label Column
```

```
TF=contains(CellConc_Eu381.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu381(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu412.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True stringsEu_381In
CellConc_Eu412(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu420.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu420(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='80uM "B"'; % Data set Want to remove identified in Label Column
TF=contains(CellConc_Eu381.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu381(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu412.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True stringsEu_381In
CellConc_Eu412(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu420.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu420(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

pattern='100uM "B"'; % Data set Want to remove identified in Label Column
TF=contains(CellConc_Eu381.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu381(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu412.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True stringsEu_381In
CellConc_Eu412(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates
TF=contains(CellConc_Eu420.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
CellConc_Eu420(row,:)=[]; % Table of Eu at wavelength 420.505 excluding Initlicates

%Convert table data and put on per billion cell basis
CellConc_Eu381=table2array(CellConc_Eu381(:,5))/1000000000;
Eu381Init_pbcells=Eu_381Initumol./CellConc_Eu381;
Eu381Post_pbcells=Eu_381Postumol./CellConc_Eu381;
Eu381Wash_pbcells=Eu_381Washumol./CellConc_Eu381;
Eu381Pellet_pbcells=Eu_381Pelletumol./CellConc_Eu381;

Ci_381=Eu381Init_pbcells;
Ce_381=Eu381Post_pbcells+Eu381Wash_pbcells;
Qe_381=Eu381Pellet_pbcells;
Ce381_Qe381=Ce_381./Qe_381;

%Average Data of Repeats
Eu381InitAvg_pbcells=mean(reshape(Eu381Init_pbcells,4,[]))';
Eu381PostAvg_pbcells=mean(reshape(Eu381Post_pbcells,4,[]))';
Eu381WashAvg_pbcells=mean(reshape(Eu381Wash_pbcells,4,[]))';
Eu381PelletAvg_pbcells=mean(reshape(Eu381Pellet_pbcells,4,[]))';

CellConc_Eu412=table2array(CellConc_Eu412(:,5))/1000000000;
Eu412Init_pbcells=Eu_412Initumol./CellConc_Eu412;
Eu412Post_pbcells=Eu_412Postumol./CellConc_Eu412;
Eu412Wash_pbcells=Eu_412Washumol./CellConc_Eu412;
Eu412Pellet_pbcells=Eu_412Pelletumol./CellConc_Eu412;
```

```
CellConc_Eu420=table2array(CellConc_Eu420(:,5))/1000000000;
Eu420Init_pbcells=Eu_420Initumol./CellConc_Eu420;
Eu420Post_pbcells=Eu_420Postumol./CellConc_Eu420;
Eu420Wash_pbcells=Eu_420Washumol./CellConc_Eu420;
Eu420Pellet_pbcells=Eu_420Pelletumol./CellConc_Eu420;

%Average Data of Repeats
Eu381InitAvg_pbcells=mean(reshape(Eu381Init_pbcells,4,[]))';
Eu381PostAvg_pbcells=mean(reshape(Eu381Post_pbcells,4,[]))';
Eu381WashAvg_pbcells=mean(reshape(Eu381Wash_pbcells,4,[]))';
Eu381PelletAvg_pbcells=mean(reshape(Eu381Pellet_pbcells,4,[]))';

Eu412InitAvg_pbcells=mean(reshape(Eu412Init_pbcells,4,[]))';
Eu412PostAvg_pbcells=mean(reshape(Eu412Post_pbcells,4,[]))';
Eu412WashAvg_pbcells=mean(reshape(Eu412Wash_pbcells,4,[]))';
Eu412PelletAvg_pbcells=mean(reshape(Eu412Pellet_pbcells,4,[]))';

Eu420InitAvg_pbcells=mean(reshape(Eu420Init_pbcells,4,[]))';
Eu420PostAvg_pbcells=mean(reshape(Eu420Post_pbcells,4,[]))';
Eu420WashAvg_pbcells=mean(reshape(Eu420Wash_pbcells,4,[]))';
Eu420PelletAvg_pbcells=mean(reshape(Eu420Pellet_pbcells,4,[]))';

Ci_381Avg=Eu381InitAvg_pbcells;
Ce_381Avg=Eu381PostAvg_pbcells+Eu381WashAvg_pbcells;
Qe_381Avg=Eu381PelletAvg_pbcells;
Ce381Avg_Qe381Avg=Ce_381Avg./Qe_381Avg;



Ci_381Avg=Eu381InitAvg_pbcells;
Ce_381Avg=Eu381PostAvg_pbcells+Eu381WashAvg_pbcells;
Qe_381Avg=Eu381PelletAvg_pbcells;
Ce381Avg_Qe381Avg=Ce_381Avg./Qe_381Avg;

Ci_412=Eu412Init_pbcells;
Ce_412=Eu412Post_pbcells+Eu412Wash_pbcells;
Qe_412=Eu412Pellet_pbcells;
Ce412_Qe412=Ce_412./Qe_412;



Ci_412Avg=Eu412InitAvg_pbcells;
Ce_412Avg=Eu412PostAvg_pbcells+Eu412WashAvg_pbcells;
Qe_412Avg=Eu412PelletAvg_pbcells;
Ce412Avg_Qe412Avg=Ce_412Avg./Qe_412Avg;

Ci_420=Eu420Init_pbcells;
Ce_420=Eu420Post_pbcells+Eu420Wash_pbcells;
Qe_420=Eu420Pellet_pbcells;
Ce420_Qe420=Ce_420./Qe_420;

Ci_420Avg=Eu420InitAvg_pbcells;
Ce_420Avg=Eu420PostAvg_pbcells+Eu420WashAvg_pbcells;
```

```
Qe_420Avg=Eu420PelletAvg_pbcells;
Ce420Avg_Qe412Avg=Ce_420Avg./Qe_420Avg;



%Standard Deviation of Replicates
Eu381InitStdev_pbcells=std(reshape(Eu381Init_pbcells,4,[]))';
Eu381PostStdev_pbcells=std(reshape(Eu381Post_pbcells,4,[]))';
Eu381WashStdev_pbcells=std(reshape(Eu381Wash_pbcells,4,[]))';
Eu381PelletStdev_pbcells=std(reshape(Eu381Pellet_pbcells,4,[]))';

Eu412InitStdev_pbcells=std(reshape(Eu412Init_pbcells,4,[]))';
Eu412PostStdev_pbcells=std(reshape(Eu412Post_pbcells,4,[]))';
Eu412WashStdev_pbcells=std(reshape(Eu412Wash_pbcells,4,[]))';
Eu412PelletStdev_pbcells=std(reshape(Eu412Pellet_pbcells,4,[]))';

Eu420InitStdev_pbcells=std(reshape(Eu420Init_pbcells,4,[]))';
Eu420PostStdev_pbcells=std(reshape(Eu420Post_pbcells,4,[]))';
Eu420WashStdev_pbcells=std(reshape(Eu420Wash_pbcells,4,[]))';
Eu420PelletStdev_pbcells=std(reshape(Eu420Pellet_pbcells,4,[]))';

Ci_381Stdev=Eu381InitStdev_pbcells;
Ce_381Stdev=Eu381PostStdev_pbcells+Eu381WashStdev_pbcells;
Qe_381Stdev=Eu381PelletStdev_pbcells;
Ce381Stdev_Qe381Stdev=Ce_381Stdev./Qe_381Stdev;

Ci_412Stdev=Eu412InitStdev_pbcells;
Ce_412Stdev=Eu412PostStdev_pbcells+Eu412WashStdev_pbcells;
Qe_412Stdev=Eu412PelletStdev_pbcells;
Ce412Stdev_Qe412Stdev=Ce_412Stdev./Qe_412Stdev;

Ci_420Stdev=Eu420InitStdev_pbcells;
Ce_420Stdev=Eu420PostStdev_pbcells+Eu420WashStdev_pbcells;
Qe_420Stdev=Eu420PelletStdev_pbcells;
Ce420Stdev_Qe412Stdev=Ce_420Stdev./Qe_420Stdev;

Qe_381=Ci_381-Ce_381;
Qe_412=Ci_412-Ce_412;
Qe_420=Ci_420-Ce_420;
%Qe_381Avg=Ci_381Avg-Ce_381Avg;

p_381=polyfit(Ce_381,Ce381_Qe381,1);%Obtain Fit Parameters for Linear Correlation Data
slope_381=p_381(1,1);
intercept_381=p_381(1,2);
x=(0.0001:.1:500)';
y_381=slope_381*x+intercept_381;
Q0_381=1/(slope_381);
kL_381=1/(intercept_381*Q0_381);
Qe_381calc=Q0_381*((kL_381*x)./(1+(kL_381*x)));

beta=[1,30]; %Initial Guesses for parameters
Langmuir=@(a,x)a(2)*((a(1)*(x))./(1+(a(1)*(x)))); %Create Langmuir Function
```

```
Langmuir_381=lsqcurvefit(Langmuir,beta,Ce_381,Qe_381,lbl,ubl) %Fit nonlinear Langmuir Function
%a=Langmuir_381.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Q0381=Langmuir_381(1,2)
kL381=Langmuir_381(1,1)
Langmuir_381nonlin=Q0381*(kL381*(x))./(1+(kL381*(x)));


beta2=[1,1]; %Initial Guesses for parameters
Freundlich=@(b,x)b(1)*(x).^(1/b(2)); %Create Freundlich Function
Freundlich_381=lsqcurvefit(Freundlich,beta2,Ce_381,Qe_381) %Fit nonlinear Freundlich Function
%b=Freundlich_381.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Kf_381=Freundlich_381(1,1);
nf_381=Freundlich_381(1,2);
Freundlich_381nonlin=Kf_381*(x).^(1/nf_381);


beta3=[1,1,1]; %Initial Guesses for parameters
Sips=@(c,x)c(3).*c(1).*((x).^c(2))./(1+c(1).*((x).^c(2))); %Create Langmuir Function assuming concentration
in 35mL (Derived from experiment used volumes)
Sips_381=lsqcurvefit(Sips,beta3,Ce_381,Qe_381,lbs,ubs)
%Sips_381=lsqcurvefit(Sips,beta3,Ce_381,Qe_381) %Fit nonlinear Sips Function
%c=Sips_381.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
n381=Sips_381(1,2)
Keq381=Sips_381(1,1)
Qmax381=Sips_381(1,3)
Sips_381nonlin=Qmax381*Keq381*((x).^n381)./(1+Keq381*((x).^n381));


p_412=polyfit(Ce_412,Ce412_Qe412,1);%Obtain Fit Parameters for Linear Correlation Data
slope_412=p_412(1,1);
intercept_412=p_412(1,2);
y_412=slope_412*x+intercept_412;
Q0_412=1/(slope_412);
kL_412=1/(intercept_412*Q0_412);
Qe_412calc=Q0_412*((kL_412*x)./(1+(kL_412*x)));

beta=[1,20]; %Initial Guesses for parameters
Langmuir=@(a,x)a(2)*((a(1)*(x))./(1+(a(1)*(x)))); %Create Langmuir Function
Langmuir_412=lsqcurvefit(Langmuir,beta,Ce_412,Qe_412,lbl,ubl) %Fit nonlinear Langmuir Function
%a=Langmuir_412.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Q0412=Langmuir_412(1,2)
kL412=Langmuir_412(1,1)
Langmuir_412nonlin=Q0412*(kL412*(x))./(1+(kL412*(x)));


beta2=[1,1]; %Initial Guesses for parameters
Freundlich=@(b,x)b(1)*(x).^(1/b(2)); %Create Freundlich Function
Freundlich_412=lsqcurvefit(Freundlich,beta2,Ce_412,Qe_412) %Fit nonlinear Freundlich Function
%b=Freundlich_412.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Kf_412=Freundlich_412(1,1);
nf_412=Freundlich_412(1,2);
Freundlich_412nonlin=Kf_412*(x).^(1/nf_412);
```

```
beta3=[1,1,30]; %Initial Guesses for parameters
Sips=@(c,x)c(3).*c(1).*((x).^c(2))./(1+c(1).*((x).^c(2))); %Create Langmuir Function assuming concentration in 35mL (Derived from experiment used volumes)
Sips_412=lsqcurvefit(Sips,beta3,Ce_412,Qe_412,lbs,ubs) %Fit nonlinear Sips Function
% c=Sips_412.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
n412=Sips_412(1,2)
Keq412=Sips_412(1,1)
Qmax412=Sips_412(1,3)
Sips_412nonlin=Qmax412*Keq412*((x).^n412)./(1+Keq412*((x).^n412));


% beta3=[1,1,30]; %Initial Guesses for parameters
% Sips=@(c,x)c(3).*c(1).*((x).^c(2))./(1+c(1).*((x).^c(2))); %Create Langmuir Function assuming concentration in 35mL (Derived from experiment used volumes)
% Sips_412=NonLinearModel.fit(Ce_412,Qe_412,Sips,beta3)
% %Sips_381=lsqcurvefit(Sips,beta3,Ce_381,Qe_381) %Fit nonlinear Sips Function
% c=Sips_412.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
% n412=c(2)
% Keq412=c(1)
% Qmax412=c(3)
% Sips_412nonlin=Qmax412*Keq412*((x).^n412)./(1+Keq412*((x).^n412));


p_420=polyfit(Ce_420,Ce420_Qe420,1);%Obtain Fit Parameters for Linear Correlation Data
slope_420=p_420(1,1);
intercept_420=p_420(1,2);
y_420=slope_420*x+intercept_420;
Q0_420=1/(slope_420);
kL_420=1/(intercept_420*Q0_420);
Qe_420calc=Q0_420*((kL_420*x)./(1+(kL_420*x)));

beta=[1,20]; %Initial Guesses for parameters
Langmuir=@(a,x)a(2)*((a(1)*(x))./(1+(a(1)*(x)))); %Create Langmuir Function
Langmuir_420=lsqcurvefit(Langmuir,beta,Ce_420,Qe_420,lbl,ubl) %Fit nonlinear Langmuir Function
%a=Langmuir_420.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Q0420=Langmuir_420(1,2)
kL420=Langmuir_420(1,1)
Langmuir_420nonlin=Q0420*(kL420*(x))./(1+(kL420*(x)));


beta2=[1,1]; %Initial Guesses for parameters
Freundlich=@(b,x)b(1)*(x).^(1/b(2)); %Create Freundlich Function
Freundlich_420=lsqcurvefit(Freundlich,beta2,Ce_420,Qe_420) %Fit nonlinear Freundlich Function
%b=Freundlich_420.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Kf_420=Freundlich_420(1,1);
nf_420=Freundlich_420(1,2);
Freundlich_420nonlin=Kf_420*(x).^(1/nf_420);


beta3=[1,1,30]; %Initial Guesses for parameters
Sips=@(c,x)c(3).*c(1).*((x).^c(2))./(1+c(1).*((x).^c(2))); %Create Langmuir Function assuming concentration in 35mL (Derived from experiment used volumes)
Sips_420=lsqcurvefit(Sips,beta3,Ce_420,Qe_420,lbs,ubs) %Fit nonlinear Sips Function
% c=Sips_420.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
```

```
n420=Sips_420(1,2)
Keq420=Sips_420(1,1)
Qmax420=Sips_420(1,3)
Sips_420nonlin=Qmax420*Keq420*((x).^n420)./(1+Keq420*((x).^n420));


%Binding Site Affinity (Phi)
Phi_381=Qe_381*100/Qmax381;
Phi_412=Qe_412*100/Qmax412;
Phi_420=Qe_420*100/Qmax420;


%Plots
figure(101)
plot(Ce_381,Qe_381,'*',x,Langmuir_381nonlin,x,Freundlich_381nonlin,x,Sips_381nonlin)
title('Sorption Isotherms for Eu @ 381.967nm')
legend('Experimental Data','Nonlinear Langmuir Fit','Nonlinear Freundlich Fit','Nonlinear Sips Fit')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')
xlim([0 300])
figure(102)
plot(Ce_412,Qe_412,'*',x,Langmuir_412nonlin,x,Freundlich_412nonlin,x,Sips_412nonlin)
title('Sorption Isotherms for Eu @ 412.967nm')
legend('Experimental Data','Nonlinear Langmuir Fit','Nonlinear Freundlich Fit','Nonlinear Sips
Fit','Location','northeastoutside')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

figure(103)
plot(Ce_420,Qe_420,'*',x,Langmuir_420nonlin,x,Freundlich_420nonlin,x,Sips_420nonlin)
title('Sorption Isotherms for Eu @ 420.967nm')
legend('Experimental Data','Nonlinear Langmuir Fit','Nonlinear Freundlich Fit','Nonlinear Sips
Fit','Location','northeastoutside')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')
%
% figure(104)
%
plot(Ce_381,Ce381_Qe381,'*',Ce_412,Ce412_Qe412,'x',Ce_412,Ce412_Qe412,'o',x,y_381,x,y_412,x,y_420)
% title('Linearized Langmuir Correlation')
% legend('Experimental Data 381nm','Experimental Data 412nm','Experimental Data 420nm','Eu 381nm','Eu
412nm','Eu 420nm')
% xlabel('Equilibrium Concentration (umols/billion cells)')
% ylabel('Sorbed Concentration (umols/billion cells)')
%
% figure(105)
%
plot(Ce_381,Qe_381,'*',Ce_412,Qe_412,'x',Ce_420,Qe_420,'o',x,Langmuir_381nonlin,x,Langmuir_412nonlin,
x,Langmuir_420nonlin)
% title('Nonlinear Langmuir Fit')
% legend('Experimental Data 381nm','Experimental Data 412nm','Experimental Data 420nm','Eu 381nm','Eu
412nm','Eu 420nm')
% xlabel('Equilibrium Concentration (umols/billion cells)')
% ylabel('Sorbed Concentration (umols/billion cells)')
%
```

```
% figure(106)
%
plot(Ce_381,Qe_381,'*',Ce_412,Qe_412,'x',Ce_420,Qe_420,'o',x,Freundlich_381nonlin,x,Freundlich_412nonli
n,x,Freundlich_420nonlin)
% title('Nonlinear Freundlich Fit')
% legend('Experimental Data 381nm','Experimental Data 412nm','Experimental Data 420nm','Eu 381nm','Eu
412nm','Eu 420nm')
% xlabel('Equilibrium Concentration (umols/billion cells)')
% ylabel('Sorbed Concentration (umols/billion cells)')
%
figure(107)
plot(Ce_381,Qe_381,'*',Ce_412,Qe_412,'x',Ce_420,Qe_420,'x',x,Sips_381nonlin,x,Sips_412nonlin,x,Sips_420
nonlin)
title('Nonlinear Sips Fit')
legend('Experimental Data 381nm','Experimental Data 412nm','Experimental Data 412nm','Eu 381nm','Eu
412nm','Eu 420nm')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

% figure(108)
% plot(Ci_381,Ce_381,'*',Ci_412,Ce_412,'x',Ci_420,Ce_420,'o')
% title('Postsorption Aqueous Concentration vs Initial Aqueous Concentration')
% legend('Eu 381nm','Eu 412nm','Eu 420nm')
% xlabel('Initial Concentration (umols/billion cells)')
% ylabel('Equilibrium Concentration (umols/billion cells)')
%
% figure(109)
% plot(Ci_381,Phi_381,'*',Ci_412,Phi_412,'x',Ci_420,Phi_420,'o')
% title('Percent Filled Binding Sites vs Initial Concentration')
% legend('Eu 381nm','Eu 412nm','Eu 420nm')
% xlabel('Initial Available Concentration (umols/billion cells)')
% ylabel('% Sites Bound')
%
% % figure(110)
% % plot(Ce_412Avg,Qe_412Avg,'*')
% % errorbar(Qe_412Avg,Qe_412Stdev,-Qe_412Stdev,'o')
% % title('Average Equilibrium Concentration vs. Sorbed Concentration')
% % xlabel('Equilibrium Concentration (umols/billion cells)')
% % ylabel('Sorbed Concentration (umols/billion cells)')


toc
%
%This program determines the Langmuir and Freundlich Parameters and
%profiles of biosorption of Neodynium by C. Necator



%Data Sorting
[data,txt,raw]=xlsread('MP-AES Nd data compiled.xlsx'); %import data from Excel file
raw(1:2,:)=[]; %Delete 1st two rows
T=cell2table(raw(2:end,:)); %convert to a table excluding 1st row of raw data
T.Properties.VariableNames=raw(1,:); %Assign Labels to Table using 1st row of raw data
T2=T;%Copy Table T for modification
T2(:,[2,4,7,8,11,12,13,14,16,23,24,25])=[];%Remove unneeded columns to analysis
```

```
T2=sortrows(T2,'Type');%Sort rows by type
T3=sortrows(T2,'Type'); %Create new table sorted by Type
T3([1:182],:)=[];% Remove rows containing Standards and Blanks
T3=sortrows(T3,'Element');
Nd_401=sortrows(T3,'Element'); %Create new table sorted by Elemental Wavelength
Nd_401([1:142,285:end],:)=[]; %Delete all data except those at Wavelength 428.967 nm
Nd_415=sortrows(T3,'Element'); %Create new table sorted by Elemental Wavelength
Nd_415([1:284,427:end],:)=[]; %Delete all data except those at Wavelength 443.973 nm
Nd_430=sortrows(T3,'Element'); %Create new table sorted by Elemental Wavelength
Nd_430(1:426,:)=[]; %Delete all data except those at Wavelength 443.973 nm

%Initials for Nd @ 401nm
Nd_401=sortrows(Nd_401,'Label'); %Sort by Label
Nd_401Init=Nd_401; %Create Initial Table for Nd at 428.967nm containing all values from Nd_401 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,~]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,~]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Pellet and Postsorption
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
Cell Conc and Stock samples

%Initials for Nd @415nm
Nd_415=sortrows(Nd_415,'Label'); %Sort by Label
```

```
Nd_415Init=Nd_415; %Create Initial Table for Nd ate 443.973nm containing all values from Nd_401 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Pellet and Postsorption
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 443.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
Cell Conc and Stock samples

%Initials for Nd @430nm
Nd_430=sortrows(Nd_430,'Label'); %Sort by Label
Nd_430Init=Nd_430; %Create Initial Table for Nd ate 446.505nm containing all values from Nd_401 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Pellet and Postsorption
samples
```

```
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS,
Cell Conc and Stock samples

%Postsorption for Nd @ 401nm
Nd_401=sortrows(Nd_401,'Label'); %Sort by Label
Nd_401Post=Nd_401; %Create Postial Table for Nd ate 428.967nm containing all values from Nd_401 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Pellet samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Pellet and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
```

```
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Postsorption for Nd @415nm
Nd_415Post=Nd_415; %Create Postial Table for Nd ate 443.973nm containing all values from Nd_401 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Pellet samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Pellet and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 443.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Postsorption for Nd @430nm
Nd_430Post=Nd_430; %Create Postial Table for Nd ate 446.505nm containing all values from Nd_401 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Pellet samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Pellet and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Pellet for Nd @ 401nm
Nd_401=sortrows(Nd_401,'Label'); %Sort by Label
Nd_401Pellet=Nd_401; %Create Pelletial Table for Nd ate 428.967nm containing all values from Nd_401
Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Postsorption and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash and Postsorption and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash, Postsorption, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Pellet for Nd @415nm
Nd_415Pellet=Nd_415; %Create Pelletial Table for Nd ate 443.973nm containing all values from Nd_401 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Postsorption and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash and Postsorption and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash, Postsorption, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Wash,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 443.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples


%Pellet for Nd @430nm
Nd_430Pellet=Nd_430; %Create Pelletial Table for Nd ate 446.505nm containing all values from Nd_401 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Postsorption and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash and Postsorption and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash, Postsorption, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Wash,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Wash for Nd @ 401nm
Nd_401=sortrows(Nd_401,'Label'); %Sort by Label
```

```
Nd_401Wash=Nd_401; %Create Washial Table for Nd ate 428.967nm containing all values from Nd_401 Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet and Postsorption and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet and Postsorption and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet, Postsorption, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Wash,Wash, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Wash for Nd @415nm
Nd_415Wash=Nd_415; %Create Washial Table for Nd ate 443.973nm containing all values from Nd_401 Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet and Postsorption and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet and Postsorption and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet, Postsorption, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 443.967 excluding Wash,Wash, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Wash for Nd @430nm
Nd_430Wash=Nd_430; %Create Washial Table for Nd ate 446.505nm containing all values from Nd_401 Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet and Postsorption and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet and Postsorption and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet, Postsorption, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.967 excluding Wash,Wash, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Filter for Nd @ 401nm
Nd_401=sortrows(Nd_401,'Label'); %Sort by Label
Nd_401Filter=Nd_401; %Create Filterial Table for Nd ate 428.967nm containing all values from Nd_401 Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet and Postsorption and 2%HNO3 samples
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet and Postsorption and HNO3 and Wash samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet, Postsorption, 2% HNO3, Wash and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Pellet,Postsorption, 2% HNO3, Wash, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 428.967 excluding Filter,Filter, 2% HNO3, Filter, AABS, Cell Conc and Stock samples
```

```
%Filter for Nd @415nm
Nd_415Filter=Nd_415; %Create Filterial Table for Nd ate 443.973nm containing all values from Nd_401
Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet and Postsorption and
2%HNO3 samples
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet and Postsorption and HNO3
and Wash samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet, Postsorption, 2% HNO3,
Wash and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 443.973 excluding Pellet,Postsorption, 2% HNO3,
Wash, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 443.967 excluding Filter,Filter, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Filter for Nd @430nm
Nd_430Filter=Nd_430; %Create Filterial Table for Nd ate 446.505nm containing all values from Nd_401
Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
```

```
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet and Postsorption and
2%HNO3 samples
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet and Postsorption and HNO3
and Wash samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet, Postsorption, 2% HNO3,
Wash and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Pellet,Postsorption, 2% HNO3,
Wash, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 443.967 excluding Filter,Filter, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Removal of Spikes and Duplicates where applicable
Nd_401Filter_Spikes_Dups=Nd_401Filter;
Nd_415Filter_Spikes_Dups=Nd_415Filter;
Nd_430Filter_Spikes_Dups=Nd_430Filter;
Nd_401Wash_Spikes_Dups=Nd_401Wash;
Nd_415Wash_Spikes_Dups=Nd_415Wash;
Nd_430Wash_Spikes_Dups=Nd_430Wash;
Nd_401Post_Spikes_Dups=Nd_401Post;
Nd_415Post_Spikes_Dups=Nd_415Post;
Nd_430Post_Spikes_Dups=Nd_430Post;
Nd_401Pellet_Spikes_Dups=Nd_401Pellet;
Nd_415Pellet_Spikes_Dups=Nd_415Pellet;
Nd_430Pellet_Spikes_Dups=Nd_430Pellet;

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
pattern='Dup'; % Data set Want to remove identified in Label Column
```

```
TF=contains(Nd_401Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_415Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_430Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Filter(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,~]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_415Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_430Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Init(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates.

pattern='Rerun'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_415Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_430Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Wash(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes

pattern='Rerun'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,~]=find(TF); % Location of True strings
Nd_401Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_415Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_430Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Post(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates

pattern='Rerun'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Nd_401Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_401Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_415Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_415Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicates
TF=contains(Nd_430Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Nd_430Pellet(row,:)=[]; % Table of Nd at wavelength 446.505 excluding Duplicate

Nd_401Init=Nd_401Init([13:24,1:12],:);
Nd_415Init=Nd_415Init([13:24,1:12],:);
Nd_430Init=Nd_430Init([13:24,1:12],:);
Nd_401Post=Nd_401Post([13:24,1:12],:);
Nd_415Post=Nd_415Post([13:24,1:12],:);
Nd_430Post=Nd_430Post([13:24,1:12],:);
Nd_401Pellet=Nd_401Pellet([13:24,1:12],:);
Nd_415Pellet=Nd_415Pellet([13:24,1:12],:);
Nd_430Pellet=Nd_430Pellet([13:24,1:12],:);
Nd_401Wash=Nd_401Wash([13:24,1:12],:);
Nd_415Wash=Nd_415Wash([13:24,1:12],:);
Nd_430Wash=Nd_430Wash([13:24,1:12],:);

%Convert Table Columns to Array For Calculations
```

```
Nd_401PostConc=table2array(Nd_401Post(:,5));
Nd_401WashConc=table2array(Nd_401Wash(:,5));
Nd_401InitConc=table2array(Nd_401Init(:,5));
Nd_401PelletConc=table2array(Nd_401Pellet(:,5));
Nd_415PostConc=table2array(Nd_415Post(:,5));
Nd_415WashConc=table2array(Nd_415Wash(:,5));
Nd_415InitConc=table2array(Nd_415Init(:,5));
Nd_415PelletConc=table2array(Nd_415Pellet(:,5));
Nd_430PostConc=table2array(Nd_430Post(:,5));
Nd_430WashConc=table2array(Nd_430Wash(:,5));
Nd_430InitConc=table2array(Nd_430Init(:,5));
Nd_430PelletConc=table2array(Nd_430Pellet(:,5));

Nd_401PostConcppm=cell2mat(Nd_401PostConc);
Nd_401WashConcppm=cell2mat(Nd_401WashConc);
Nd_401InitConcppm=cell2mat(Nd_401InitConc);
Nd_401PelletConcppm=cell2mat(Nd_401PelletConc);
Nd_415PostConcppm=cell2mat(Nd_415PostConc);
Nd_415WashConcppm=cell2mat(Nd_415WashConc);
Nd_415InitConcppm=cell2mat(Nd_415InitConc);
Nd_415PelletConcppm=cell2mat(Nd_415PelletConc);
Nd_430PostConcppm=cell2mat(Nd_430PostConc);
Nd_430WashConcppm=cell2mat(Nd_430WashConc);
Nd_430InitConcppm=cell2mat(Nd_430InitConc);
Nd_430PelletConcppm=cell2mat(Nd_430PelletConc);

%Convert ppm to uM basis
Nd_401PostConc=Nd_401PostConcppm*MW_Nd;
Nd_401WashConc=Nd_401WashConcppm*MW_Nd;
Nd_401InitConc=Nd_401InitConcppm*MW_Nd;
Nd_401PelletConc=Nd_401PelletConcppm*MW_Nd;
Nd_415PostConc=Nd_415PostConcppm*MW_Nd;
Nd_415WashConc=Nd_415WashConcppm*MW_Nd;
Nd_415InitConc=Nd_415InitConcppm*MW_Nd;
Nd_415PelletConc=Nd_415PelletConcppm*MW_Nd;
Nd_430PostConc=Nd_430PostConcppm*MW_Nd;
Nd_430WashConc=Nd_430WashConcppm*MW_Nd;
Nd_430InitConc=Nd_430InitConcppm*MW_Nd;
Nd_430PelletConc=Nd_430PelletConcppm*MW_Nd;

Nd_401Postumol=Nd_401PostConc*35/1000;
Nd_401Washumol=Nd_401WashConc*45/1000;
Nd_401Pelletumol=Nd_401PelletConc*30/1000;
Nd_401Initumol=Nd_401InitConc*35/1000;
Nd_415Postumol=Nd_415PostConc*35/1000;
Nd_415Washumol=Nd_415WashConc*45/1000;
Nd_415Pelletumol=Nd_415PelletConc*30/1000;
Nd_415Initumol=Nd_415InitConc*35/1000;
Nd_430Postumol=Nd_430PostConc*35/1000;
Nd_430Washumol=Nd_430WashConc*45/1000;
Nd_430Pelletumol=Nd_430PelletConc*30/1000;
Nd_430Initumol=Nd_430InitConc*35/1000;

%Material Balances
Nd_401MatB=Nd_401Initumol-Nd_401Postumol-Nd_401Washumol-Nd_401Pelletumol;
```

```
Nd_415MatB=Nd_415Initumol-Nd_415Postumol-Nd_415Washumol-Nd_415Pelletumol;
Nd_430MatB=Nd_430Initumol-Nd_430Postumol-Nd_430Washumol-Nd_430Pelletumol;

%Percent Recovery
PercentRecov_Nd401=(Nd_401Postumol+Nd_401Washumol+Nd_401Pelletumol)./Nd_401Initumol;
PercentRecov_Nd415=(Nd_415Postumol+Nd_415Washumol+Nd_415Pelletumol)./Nd_415Initumol;
PercentRecov_Nd430=(Nd_430Postumol+Nd_430Washumol+Nd_430Pelletumol)./Nd_430Initumol;

%Langmuir Isotherm per billion cell basis
Tcells=cell2table(raw([656:end],:)); %convert to a table excluding 1st row of raw data
Tcells.Properties.VariableNames=raw(1,:); %Assign Labels to Table using 1st row of raw data
Tcells(:,[2,4,6:28])=[];%Remove unneeded columns to analysis
Tcells=sortrows(Tcells,'Element'); %Sort Table Tcells by Element
CellConc_Nd401=sortrows(Tcells,'Element'); %Create new table sorted by Elemental Wavelength
CellConc_Nd401([1:24,49:end],:)=[];%Delete all data except those at Wavelength 428.967 nm
CellConc_Nd401=sortrows(CellConc_Nd401,'Label');
CellConc_Nd415=sortrows(Tcells,'Element'); %Create new table sorted by Elemental Wavelength
CellConc_Nd415([1:48,73:end],:)=[]; %Delete all data except those at Wavelength 443 nm
CellConc_Nd415=sortrows(CellConc_Nd415,'Label');
CellConc_Nd430=sortrows(Tcells,'Element'); %Create new table sorted by Elemental Wavelength
CellConc_Nd430(1:72,:)=[]; %Delete all data except those at Wavelength 443 nm
CellConc_Nd430=sortrows(CellConc_Nd430,'Label');


%Convert table data and put on per billion cell basis
CellConc_Nd401=table2array(CellConc_Nd401(:,5))/1000000000;
Nd401Init_pbcells=Nd_401Initumol./CellConc_Nd401;
Nd401Post_pbcells=Nd_401Postumol./CellConc_Nd401;
Nd401Wash_pbcells=Nd_401Washumol./CellConc_Nd401;
Nd401Pellet_pbcells=Nd_401Pelletumol./CellConc_Nd401;

Ci_401=Nd401Init_pbcells;
Ce_401=Nd401Post_pbcells+Nd401Wash_pbcells;
Qe_401=Nd401Pellet_pbcells;
Ce401_Qe401=Ce_401./Qe_401;

CellConc_Nd415=table2array(CellConc_Nd415(:,5))/1000000000;
Nd415Init_pbcells=Nd_415Initumol./CellConc_Nd415;
Nd415Post_pbcells=Nd_415Postumol./CellConc_Nd415;
Nd415Wash_pbcells=Nd_415Washumol./CellConc_Nd415;
Nd415Pellet_pbcells=Nd_415Pelletumol./CellConc_Nd415;

Ci_415=Nd415Init_pbcells;
Ce_415=Nd415Post_pbcells+Nd415Wash_pbcells;
Qe_415=Nd415Pellet_pbcells;
Ce415_Qe415=Ce_415./Qe_415;

CellConc_Nd430=table2array(CellConc_Nd430(:,5))/1000000000;
Nd430Init_pbcells=Nd_430Initumol./CellConc_Nd430;
Nd430Post_pbcells=Nd_430Postumol./CellConc_Nd430;
Nd430Wash_pbcells=Nd_430Washumol./CellConc_Nd430;
Nd430Pellet_pbcells=Nd_430Pelletumol./CellConc_Nd430;

Ci_430=Nd430Init_pbcells;
```

```
Ce_430=Nd430Post_pbcells+Nd430Wash_pbcells;
Qe_430=Nd430Pellet_pbcells;
Ce430_Qe430=Ce_430./Qe_430;

%Average Data of Repeats
Nd401InitAvg_pbcells=mean(reshape(Nd401Init_pbcells,4,[]))';
Nd401PostAvg_pbcells=mean(reshape(Nd401Post_pbcells,4,[]))';
Nd401WashAvg_pbcells=mean(reshape(Nd401Wash_pbcells,4,[]))';
Nd401PelletAvg_pbcells=mean(reshape(Nd401Pellet_pbcells,4,[]))';

Nd415InitAvg_pbcells=mean(reshape(Nd415Init_pbcells,4,[]))';
Nd415PostAvg_pbcells=mean(reshape(Nd415Post_pbcells,4,[]))';
Nd415WashAvg_pbcells=mean(reshape(Nd415Wash_pbcells,4,[]))';
Nd415PelletAvg_pbcells=mean(reshape(Nd415Pellet_pbcells,4,[]))';

Nd430InitAvg_pbcells=mean(reshape(Nd430Init_pbcells,4,[]))';
Nd430PostAvg_pbcells=mean(reshape(Nd430Post_pbcells,4,[]))';
Nd430WashAvg_pbcells=mean(reshape(Nd430Wash_pbcells,4,[]))';
Nd430PelletAvg_pbcells=mean(reshape(Nd430Pellet_pbcells,4,[]))';

Ci_401Avg=Nd401InitAvg_pbcells;
Ce_401Avg=Nd401PostAvg_pbcells+Nd401WashAvg_pbcells;
Qe_401Avg=Nd401PelletAvg_pbcells;
Ce401Avg_Qe401Avg=Ce_401Avg./Qe_401Avg;

Ci_415Avg=Nd415InitAvg_pbcells;
Ce_415Avg=Nd415PostAvg_pbcells+Nd415WashAvg_pbcells;
Qe_415Avg=Nd415PelletAvg_pbcells;
Ce415Avg_Qe415Avg=Ce_415Avg./Qe_415Avg;

Ci_430Avg=Nd430InitAvg_pbcells;
Ce_430Avg=Nd430PostAvg_pbcells+Nd430WashAvg_pbcells;
Qe_430Avg=Nd430PelletAvg_pbcells;
Ce430Avg_Qe430Avg=Ce_430Avg./Qe_430Avg;

Nd401InitStdev_pbcells=std(reshape(Nd401Init_pbcells,4,[]))';
Nd401PostStdev_pbcells=std(reshape(Nd401Post_pbcells,4,[]))';
Nd401WashStdev_pbcells=std(reshape(Nd401Wash_pbcells,4,[]))';
Nd401PelletStdev_pbcells=std(reshape(Nd401Pellet_pbcells,4,[]))';

Nd415InitStdev_pbcells=std(reshape(Nd415Init_pbcells,4,[]))';
Nd415PostStdev_pbcells=std(reshape(Nd415Post_pbcells,4,[]))';
Nd415WashStdev_pbcells=std(reshape(Nd415Wash_pbcells,4,[]))';
Nd415PelletStdev_pbcells=std(reshape(Nd415Pellet_pbcells,4,[]))';

Nd430InitStdev_pbcells=std(reshape(Nd430Init_pbcells,4,[]))';
Nd430PostStdev_pbcells=std(reshape(Nd430Post_pbcells,4,[]))';
Nd430WashStdev_pbcells=std(reshape(Nd430Wash_pbcells,4,[]))';
Nd430PelletStdev_pbcells=std(reshape(Nd430Pellet_pbcells,4,[]))';

Ci_401Stdev=Nd401InitStdev_pbcells;
Ce_401Stdev=Nd401PostStdev_pbcells+Nd401WashStdev_pbcells;
Qe_401Stdev=Nd401PelletStdev_pbcells;
Ce401Stdev_Qe401Stdev=Ce_401Stdev./Qe_401Stdev;
```

```
Ci_415Stdev=Nd415InitStdev_pbcells;
Ce_415Stdev=Nd415PostStdev_pbcells+Nd415WashStdev_pbcells;
Qe_415Stdev=Nd415PelletStdev_pbcells;
Ce415Stdev_Qe415Stdev=Ce_415Stdev./Qe_415Stdev;

Ci_430Stdev=Nd430InitStdev_pbcells;
Ce_430Stdev=Nd430PostStdev_pbcells+Nd430WashStdev_pbcells;
Qe_430Stdev=Nd430PelletStdev_pbcells;
Ce430Stdev_Qe430Stdev=Ce_430Stdev./Qe_430Stdev;



p_401=polyfit(Ce_401,Ce401_Qe401,1);%Obtain Fit Parameters for Linear Correlation Data
slope_401=p_401(1,1);
intercept_401=p_401(1,2);
x=(0.0001:.1:500)';
y_401=slope_401*x+intercept_401;
Q0_401=1/(slope_401);
kL_401=1/(intercept_401*Q0_401);
Qe_401calc=Q0_401*((kL_401*x)./(1+(kL_401*x)));

Qe_401=Ci_401-Ce_401;
Qe_415=Ci_415-Ce_415;
Qe_430=Ci_430-Ce_430;

%Qe_401=Ci_401-Ce_401;

options = optimoptions('lsqcurvefit','MaxFunctionEvaluations',9999999)


beta=[1,1]; %Initial Guesses for parameters
Langmuir=@(a,x)a(2)*((a(1)*(x))./(1+(a(1)*(x)))); %Create Langmuir Function
Langmuir_401=lsqcurvefit(Langmuir,beta,Ce_401,Qe_401,lbl,ubl) %Fit nonlinear Langmuir Function
%a=Langmuir_401.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Q0401=Langmuir_401(1,2)
kL401=Langmuir_401(1,1)
Langmuir_401nonlin=Q0401*(kL401*(x))./(1+(kL401*(x)));

beta2=[1,1]; %Initial Guesses for parameters
Freundlich=@(b,x)b(1)*(x).^(1/b(2)); %Create Freundlich Function
Freundlich_401=lsqcurvefit(Freundlich,beta2,Ce_401,Qe_401) %Fit nonlinear Freundlich Function
%b=Freundlich_401.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Kf_401=Freundlich_401(1,1)
n_401=Freundlich_401(1,2)
Freundlich_401nonlin=Kf_401*(x).^(1/n_401);


beta3=[1,1,max(Qe_401)]; %Initial Guesses for parameters
Sips=@(c,x)c(3)*c(1)*((x).^c(2))./(1+c(1)*((x).^c(2))); %Create Langmuir Function assuming concentration
in 35mL (Derived from experiment used volumes)
Sips_401=lsqcurvefit(Sips,beta3,Ce_401,Qe_401,lbs,ubs,options) %Fit nonlinear Sips Function
%c=Sips_401.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
```

```
n401=Sips_401(1,2)
Keq401=Sips_401(1,1)
Qmax401=Sips_401(1,3)
Sips_401nonlin=Qmax401*Keq401*((x).^n401)./(1+Keq401*((x).^n401));


Ci_415=Nd415Init_pbcells;
Ce_415=Nd415Post_pbcells+Nd415Wash_pbcells;
Qe_415=Nd415Pellet_pbcells;
Ce415_Qe415=Ce_415./Qe_415;

p_415=polyfit(Ce_415,Ce415_Qe415,1);%Obtain Fit Parameters for Linear Correlation Data
slope_415=p_415(1,1);
intercept_415=p_415(1,2);
y_415=slope_415*x+intercept_415;
Q0_415=1/(slope_415);
kL_415=1/(intercept_415*Q0_415);
Qe_415calc=Q0_415*((kL_415*x)./(1+(kL_415*x)));


beta=[1,1]; %Initial Guesses for parameters
Langmuir=@(a,x)a(2)*((a(1)*(x))./(1+(a(1)*(x)))); %Create Langmuir Function
Langmuir_415=lsqcurvefit(Langmuir,beta,Ce_415,Qe_415,lbl,ubl) %Fit nonlinear Langmuir Function
%a=Langmuir_415.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Q0415=Langmuir_415(1,2)
kL415=Langmuir_415(1,1)
Langmuir_415nonlin=Q0415*(kL415*(x))./(1+(kL415*(x)));

beta2=[1,1]; %Initial Guesses for parameters
Freundlich=@(b,x)b(1)*(x).^(1/b(2)); %Create Freundlich Function
Freundlich_415=lsqcurvefit(Freundlich,beta2,Ce_415,Qe_415) %Fit nonlinear Freundlich Function
%b=Freundlich_415.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Kf_415=Freundlich_415(1,1)
n_415=Freundlich_415(1,2)
Freundlich_415nonlin=Kf_415*(x).^(1/n_415);


beta3=[1,1,max(Qe_415)]; %Initial Guesses for parameters
Sips=@(c,x)c(3)*c(1)*((x).^c(2))./(1+c(1)*((x).^c(2))); %Create Langmuir Function assuming concentration
in 35mL (Derived from experiment used volumes)
Sips_415=lsqcurvefit(Sips,beta3,Ce_415,Qe_415,lbs,ubs,options) %Fit nonlinear Sips Function
%c=Sips_415.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
n415=Sips_415(1,2)
Keq415=Sips_415(1,1)
Qmax415=Sips_415(1,3)
Sips_415nonlin=Qmax415*Keq415*((x).^n415)./(1+Keq415*((x).^n415));



Ci_430=Nd430Init_pbcells;
Ce_430=Nd430Post_pbcells+Nd430Wash_pbcells;
Qe_430=Nd430Pellet_pbcells;
Ce430_Qe430=Ce_430./Qe_430;
```

```
p_430=polyfit(Ce_430,Ce430_Qe430,1);%Obtain Fit Parameters for Linear Correlation Data
slope_430=p_430(1,1);
intercept_430=p_430(1,2);
y_430=slope_430*x+intercept_430;
Q0_430=1/(slope_430);
kL_430=1/(intercept_430*Q0_430);
Qe_430calc=Q0_430*((kL_430*x)./(1+(kL_430*x)));

beta=[1,1]; %Initial Guesses for parameters
Langmuir=@(a,x)a(2)*((a(1)*(x))./(1+(a(1)*(x)))); %Create Langmuir Function
Langmuir_430=lsqcurvefit(Langmuir,beta,Ce_430,Qe_430,lbl,ubl) %Fit nonlinear Langmuir Function
%a=Langmuir_430.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Q0430=Langmuir_430(1,2)
kL430=Langmuir_430(1,1)
Langmuir_430nonlin=Q0430*(kL430*(x))./(1+(kL430*(x)));

beta2=[1,1]; %Initial Guesses for parameters
Freundlich=@(b,x)b(1)*(x).^(1/b(2)); %Create Freundlich Function
Freundlich_430=lsqcurvefit(Freundlich,beta2,Ce_430,Qe_430) %Fit nonlinear Freundlich Function
%b=Freundlich_430.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Kf_430=Freundlich_430(1,1)
n_430=Freundlich_430(1,2)
Freundlich_430nonlin=Kf_430*(x).^(1/n_430);


beta3=[1,1,max(Qe_430)]; %Initial Guesses for parameters
Sips=@(c,x)c(3)*c(1)*((x).^c(2))./(1+c(1)*((x).^c(2))); %Create Langmuir Function assuming concentration
in 35mL (Derived from experiment used volumes)
Sips_430=lsqcurvefit(Sips,beta3,Ce_430,Qe_430,lbs,ubs,options) %Fit nonlinear Sips Function
%c=Sips_430.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
n430=Sips_430(1,2)
Keq430=Sips_430(1,1)
Qmax430=Sips_430(1,3)
Sips_430nonlin=Qmax430*Keq430*((x).^n430)./(1+Keq430*((x).^n430));

%Binding Site Affinity (Phi)
Phi_401=Qe_401*100/Qmax401;
Phi_415=Qe_415*100/Qmax415;
Phi_430=Qe_430*100/Qmax430;


%Plots
figure(201)
plot(Ce_401,Qe_401,'*',x,Langmuir_401nonlin,x,Freundlich_401nonlin,x,Sips_401nonlin)
title('Sorption Isotherms for Nd @ 401nm')
legend('Experimental Data','Nonlinear Langmuir Fit','Nonlinear Freundlich Fit','Nonlinear Sips Fit')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')
xlim([0 300])
figure(202)
plot(Ce_415,Qe_415,'*',x,Langmuir_415nonlin,x,Freundlich_415nonlin,x,Sips_415nonlin)
title('Sorption Isotherms for Nd @ 415 nm')
```

```
legend('Experimental Data','Nonlinear Langmuir Fit','Nonlinear Freundlich Fit','Nonlinear Sips
Fit','Location','northeastoutside')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

figure(203)
plot(Ce_430,Qe_430,'*',x,Langmuir_430nonlin,x,Freundlich_430nonlin,x,Sips_430nonlin)
title('Sorption Isotherms for Nd @ 430 nm')
legend('Experimental Data','Nonlinear Langmuir Fit','Nonlinear Freundlich Fit','Nonlinear Sips
Fit','Location','northeastoutside')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')
%
% figure(204)
%
plot(Ce_401,Ce401_Qe401,'*',Ce_415,Ce415_Qe415,'x',Ce_430,Ce430_Qe430,'o',x,y_401,x,y_415,x,y_430)
% title('Linearized Langmuir Correlation')
% legend('Experimental Data 401nm','Experimental Data 415nm','Experimental Data 430nm','Nd 401nm','Nd
415nm','Nd 430nm')
% xlabel('Equilibrium Concentration (umols/billion cells)')
% ylabel('Sorbed Concentration (umols/billion cells)')
%
% figure(205)
%
plot(Ce_401,Qe_401,'*',Ce_415,Qe_415,'x',Ce_430,Qe_430,'o',x,Langmuir_401nonlin,x,Langmuir_415nonlin,
x,Langmuir_430nonlin)
% title('Nonlinear Langmuir Fit')
% legend('Experimental Data 401nm','Experimental Data 415nm','Experimental Data 430nm','Nd 401nm','Nd
415nm','Nd 430nm')
% xlabel('Equilibrium Concentration (umols/billion cells)')
% ylabel('Sorbed Concentration (umols/billion cells)')
%
% figure(206)
%
plot(Ce_401,Qe_401,'*',Ce_415,Qe_415,'x',Ce_430,Qe_430,'o',x,Freundlich_401nonlin,x,Freundlich_415nonli
n,x,Freundlich_430nonlin)
% title('Nonlinear Freundlich Fit')
% legend('Experimental Data 401nm','Experimental Data 415nm','Experimental Data 430nm','Nd 401nm','Nd
415nm','Nd 430nm')
% xlabel('Equilibrium Concentration (umols/billion cells)')
% ylabel('Sorbed Concentration (umols/billion cells)')
%
% figure(207)
%
plot(Ce_401,Qe_401,'*',Ce_415,Qe_415,'x',Ce_430,Qe_430,'o',x,Sips_401nonlin,x,Sips_415nonlin,x,Sips_430
nonlin)
% title('Nonlinear Sips Fit')
% legend('Experimental Data 401nm','Experimental Data 415nm','Experimental Data 430nm','Nd 401nm','Nd
415nm','Nd 430nm')
% xlabel('Equilibrium Concentration (umols/billion cells)')
% ylabel('Sorbed Concentration (umols/billion cells)')
%
% figure(208)
% plot(Ci_401,Ce_401,'*',Ci_415,Ce_415,'x',Ci_430,Ce_430,'o')
% title('Postsorption Aqueous Concentration vs Initial Aqueous Concentration')
% legend('Nd 401nm','Nd 415nm','Nd 430nm')
```

```
% xlabel('Equilibrium Concentration (umols/billion cells)')
% ylabel('Initial Concentration (umols/billion cells)')
%
% figure(209)
% plot(Ci_401,Phi_401,'*',Ci_415,Phi_415,'x',Ci_430,Phi_430,'o')
% title('Percent Binding Sites Filled vs Initial Aqueous Concentration')
% legend('Nd 401nm','Nd 415nm','Nd 430nm')
% xlabel('Initial Concentration (umols/billion cells)')
% ylabel('Percent Binding Sites Filled')

toc
%
%This program determines the Langmuir and Freundlich Parameters and
%profiles of biosorption of Smropium by C. Necator




%Data Sorting
[data,txt,raw]=xlsread('MP-AES Sm data compiled.xlsx'); %import data from Excel file
raw(1:2,:)=[]; %Delete 1st two rows
T=cell2table(raw(2:end,:)); %convert to a table excluding 1st row of raw data
T.Properties.VariableNames=raw(1,:); %Assign Labels to Table using 1st row of raw data
T2=T;%Copy Table T for modification
T2(:,[2,4,7,8,11,12,13,14,16,23,24,25])=[];%Remove unneeded columns to analysis
T2=sortrows(T2,'Type');%Sort rows by type
T3=sortrows(T2,'Type'); %Create new table sorted by Type
T3([1:182],:)=[];% Remove rows containing Standards and Blanks
T3=sortrows(T3,'Element');
Sm_428=sortrows(T3,'Element'); %Create new table sorted by Elemental Wavelength
Sm_428([1:145,291:end],:)=[]; %Delete all data except those at Wavelength 428.967 nm
Sm_443=sortrows(T3,'Element'); %Create new table sorted by Elemental Wavelength
Sm_443([1:290,436:end],:)=[]; %Delete all data except those at Wavelength 443.973 nm
Sm_446=sortrows(T3,'Element'); %Create new table sorted by Elemental Wavelength
Sm_446([1:435],:)=[]; %Delete all data except those at Wavelength 443.973 nm

%Initials for Sm @ 428nm
Sm_428=sortrows(Sm_428,'Label'); %Sort by Label
Sm_428Init=Sm_428; %Create Initial Table for Sm at 428.967nm containing all values from Sm_428 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,~]=find(TF); % Location of True strings
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,~]=find(TF); % Location of True strings
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Pellet and Postsorption
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Pellet and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Pellet and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash, Pellet, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Initials for Sm @443nm
Sm_443Init=Sm_443; %Create Initial Table for Sm ate 443.973nm containing all values from Sm_428 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Pellet and Postsorption samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Pellet and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Pellet and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash, Pellet, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 443.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Initials for Sm @446nm
Sm_446Init=Sm_446; %Create Initial Table for Sm ate 446.505nm containing all values from Sm_428 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Pellet and Postsorption
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Postsorption for Sm @ 428nm
Sm_428=sortrows(Sm_428,'Label'); %Sort by Label
Sm_428Post=Sm_428; %Create Postial Table for Sm ate 428.967nm containing all values from Sm_428 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
```

```
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Pellet samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Pellet and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Postsorption for Sm @443nm
Sm_443Post=Sm_443; %Create Postial Table for Sm ate 443.973nm containing all values from Sm_428 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Pellet samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Pellet and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 443.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Postsorption for Sm @446nm
Sm_446Post=Sm_446; %Create Postial Table for Sm ate 446.505nm containing all values from Sm_428 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash samples
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Pellet samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Pellet and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Pellet and 2%HNO3
samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Pellet and HNO3 and Filter
samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash, Pellet, 2% HNO3, Filter and
AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Pellet for Sm @ 428nm
Sm_428=sortrows(Sm_428,'Label'); %Sort by Label
Sm_428Pellet=Sm_428; %Create Pelletial Table for Sm ate 428.967nm containing all values from Sm_428 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Postsorption and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash and Postsorption and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash, Postsorption, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash,Pellet, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Pellet for Sm @443nm
Sm_443Pellet=Sm_443; %Create Pelletial Table for Sm ate 443.973nm containing all values from Sm_428 Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Postsorption and
2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash and Postsorption and HNO3
and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash, Postsorption, 2% HNO3,
Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Wash,Postsorption, 2% HNO3,
Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 443.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples


%Pellet for Sm @446nm
Sm_446Pellet=Sm_446; %Create Pelletial Table for Sm ate 446.505nm containing all values from Sm_428
Table
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Postsorption and
2%HNO3 samples
```

```
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash and Postsorption and HNO3
and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash, Postsorption, 2% HNO3,
Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Wash,Postsorption, 2% HNO3,
Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.967 excluding Wash,Pellet, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Wash for Sm @ 428nm
Sm_428=sortrows(Sm_428,'Label'); %Sort by Label
Sm_428Wash=Sm_428; %Create Washial Table for Sm ate 428.967nm containing all values from Sm_428
Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet and Postsorption and
2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet and Postsorption and HNO3
and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet, Postsorption, 2% HNO3,
Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Wash,Wash, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Wash for Sm @443nm
Sm_443Wash=Sm_443; %Create Washial Table for Sm ate 443.973nm containing all values from Sm_428 Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet and Postsorption and Initial samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet and Postsorption and 2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet and Postsorption and HNO3 and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet, Postsorption, 2% HNO3, Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet,Postsorption, 2% HNO3, Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 443.967 excluding Wash,Wash, 2% HNO3, Filter, AABS, Cell Conc and Stock samples

%Wash for Sm @446nm
Sm_446Wash=Sm_446; %Create Washial Table for Sm ate 446.505nm containing all values from Sm_428 Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet and Postsorption and
2%HNO3 samples
pattern='Filter'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet and Postsorption and HNO3
and Filter samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet, Postsorption, 2% HNO3,
Filter and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet,Postsorption, 2% HNO3,
Filter, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.967 excluding Wash,Wash, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Filter for Sm @ 428nm
Sm_428=sortrows(Sm_428,'Label'); %Sort by Label
Sm_428Filter=Sm_428; %Create Filterial Table for Sm ate 428.967nm containing all values from Sm_428
Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet and Postsorption and
2%HNO3 samples
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet and Postsorption and HNO3
and Wash samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet, Postsorption, 2% HNO3,
Wash and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Pellet,Postsorption, 2% HNO3,
Wash, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 428.967 excluding Filter,Filter, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Filter for Sm @443nm
Sm_443Filter=Sm_443; %Create Filterial Table for Sm ate 443.973nm containing all values from Sm_428
Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet and Postsorption and
2%HNO3 samples
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet and Postsorption and HNO3
and Wash samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet, Postsorption, 2% HNO3,
Wash and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
```

```
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 443.973 excluding Pellet,Postsorption, 2% HNO3,
Wash, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 443.967 excluding Filter,Filter, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Filter for Sm @446nm
Sm_446Filter=Sm_446; %Create Filterial Table for Sm ate 446.505nm containing all values from Sm_428
Table
pattern='Pellet'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet samples
pattern='Postsorption'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet and Postsorption samples
pattern='Initial'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet and Postsorption and Initial
samples
pattern='2%'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet and Postsorption and
2%HNO3 samples
pattern='Wash'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet and Postsorption and HNO3
and Wash samples
pattern='AABS'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet, Postsorption, 2% HNO3,
Wash and AABS samples
pattern='Stock'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Pellet,Postsorption, 2% HNO3,
Wash, AABS, and Stock samples
pattern='Cell'; % Data set Want to remove identified in Label Column
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 443.967 excluding Filter,Filter, 2% HNO3, Filter,
AABS, Cell Conc and Stock samples

%Removal of Spikes and Duplicates where applicable
Sm_428Filter_Spikes_Dups=Sm_428Filter;
Sm_443Filter_Spikes_Dups=Sm_443Filter;
```

```
Sm_446Filter_Spikes_Dups=Sm_446Filter;
Sm_428Wash_Spikes_Dups=Sm_428Wash;
Sm_443Wash_Spikes_Dups=Sm_443Wash;
Sm_446Wash_Spikes_Dups=Sm_446Wash;
Sm_428Post_Spikes_Dups=Sm_428Post;
Sm_443Post_Spikes_Dups=Sm_443Post;
Sm_446Post_Spikes_Dups=Sm_446Post;
Sm_428Pellet_Spikes_Dups=Sm_428Pellet;
Sm_443Pellet_Spikes_Dups=Sm_443Pellet;
Sm_446Pellet_Spikes_Dups=Sm_446Pellet;

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,~,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_443Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_446Filter.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Filter(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
```

```
[row,col,v]=find(TF); % Location of True strings
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_443Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_446Wash.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Wash(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates

pattern='Spike'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
```

```
Sm_428Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_443Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_446Init.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Init(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates.

pattern='Rerun'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,~]=find(TF); % Location of True strings
Sm_428Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_443Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_446Post.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Post(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates

pattern='Rerun'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Spikes

pattern='Dup'; % Data set Want to remove identified in Label Column
TF=contains(Sm_428Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_428Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_443Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_443Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates
TF=contains(Sm_446Pellet.Label,pattern); %Vector containing logical search ("1" if True, "0" if False)
[row,col,v]=find(TF); % Location of True strings
Sm_446Pellet(row,:)=[]; % Table of Sm at wavelength 446.505 excluding Duplicates

Sm_428Init=Sm_428Init([13:24,1:12],:);
Sm_428Post=Sm_428Post([13:24,1:12],:);
Sm_428Wash=Sm_428Wash([13:24,1:12],:);
```

```
Sm_428Pellet=Sm_428Pellet([13:24,1:12],:);

%Convert Table Columns to Array For Calculations
Sm_428PostConc=table2array(Sm_428Post(:,5));
Sm_428WashConc=table2array(Sm_428Wash(:,5));
Sm_428InitConc=table2array(Sm_428Init(:,5));
Sm_428PelletConc=table2array(Sm_428Pellet(:,5));
Sm_443PostConc=table2array(Sm_443Post(:,5));
Sm_443WashConc=table2array(Sm_443Wash(:,5));
Sm_443InitConc=table2array(Sm_443Init(:,5));
Sm_443PelletConc=table2array(Sm_443Pellet(:,5));
Sm_446PostConc=table2array(Sm_446Post(:,5));
Sm_446WashConc=table2array(Sm_446Wash(:,5));
Sm_446InitConc=table2array(Sm_446Init(:,5));
Sm_446PelletConc=table2array(Sm_446Pellet(:,5));

Sm_428PostConcppm=cell2mat(Sm_428PostConc);
Sm_428WashConcppm=cell2mat(Sm_428WashConc);
Sm_428InitConcppm=cell2mat(Sm_428InitConc);
Sm_428PelletConcppm=cell2mat(Sm_428PelletConc);
Sm_443PostConcppm=cell2mat(Sm_443PostConc);
Sm_443WashConcppm=cell2mat(Sm_443WashConc);
Sm_443InitConcppm=cell2mat(Sm_443InitConc);
Sm_443PelletConcppm=cell2mat(Sm_443PelletConc);
Sm_446PostConcppm=cell2mat(Sm_446PostConc);
Sm_446WashConcppm=cell2mat(Sm_446WashConc);
Sm_446InitConcppm=cell2mat(Sm_446InitConc);
Sm_446PelletConcppm=cell2mat(Sm_446PelletConc);

%Convert ppm to uM basis
Sm_428PostConc=Sm_428PostConcppm*MW_Sm;
Sm_428WashConc=Sm_428WashConcppm*MW_Sm;
Sm_428InitConc=Sm_428InitConcppm*MW_Sm;
Sm_428PelletConc=Sm_428PelletConcppm*MW_Sm;
Sm_443PostConc=Sm_443PostConcppm*MW_Sm;
Sm_443WashConc=Sm_443WashConcppm*MW_Sm;
Sm_443InitConc=Sm_443InitConcppm*MW_Sm;
Sm_443PelletConc=Sm_443PelletConcppm*MW_Sm;
Sm_446PostConc=Sm_446PostConcppm*MW_Sm;
Sm_446WashConc=Sm_446WashConcppm*MW_Sm;
Sm_446InitConc=Sm_446InitConcppm*MW_Sm;
Sm_446PelletConc=Sm_446PelletConcppm*MW_Sm;

Sm_428Postumol=Sm_428PostConc*35/1000;
Sm_428Washumol=Sm_428WashConc*45/1000;
Sm_428Pelletumol=Sm_428PelletConc*30/1000;
Sm_428Initumol=Sm_428InitConc*35/1000;
Sm_443Postumol=Sm_443PostConc*35/1000;
Sm_443Washumol=Sm_443WashConc*45/1000;
Sm_443Pelletumol=Sm_443PelletConc*30/1000;
Sm_443Initumol=Sm_443InitConc*35/1000;
Sm_446Postumol=Sm_446PostConc*35/1000;
Sm_446Washumol=Sm_446WashConc*45/1000;
Sm_446Pelletumol=Sm_446PelletConc*30/1000;
Sm_446Initumol=Sm_446InitConc*35/1000;
```

```
%Material Balances
Sm_428MatB=Sm_428Initumol-Sm_428Postumol-Sm_428Washumol-Sm_428Pelletumol;
Sm_443MatB=Sm_443Initumol-Sm_443Postumol-Sm_443Washumol-Sm_443Pelletumol;
Sm_446MatB=Sm_446Initumol-Sm_446Postumol-Sm_446Washumol-Sm_446Pelletumol;

%Percent Recovery
PercentRecov_Sm428=(Sm_428Postumol+Sm_428Washumol+Sm_428Pelletumol)./Sm_428Initumol;
PercentRecov_Sm443=(Sm_443Postumol+Sm_443Washumol+Sm_443Pelletumol)./Sm_443Initumol;
PercentRecov_Sm446=(Sm_446Postumol+Sm_446Washumol+Sm_446Pelletumol)./Sm_446Initumol;

%Langmuir Isotherm per billion cell basis
Tcells=cell2table(raw([668:end],:)); %convert to a table excluding 1st row of raw data
Tcells.Properties.VariableNames=raw(1,:); %Assign Labels to Table using 1st row of raw data
Tcells(:,[2,4,6:28])=[];%Remove unneeded columns to analysis
Tcells=sortrows(Tcells,'Element'); %Sort Table Tcells by Element
CellConc_Sm428=sortrows(Tcells,'Element'); %Create new table sorted by Elemental Wavelength
CellConc_Sm428([1:24,49:end],:)=[]; %Delete all data except those at Wavelength 428.967 nm
CellConc_Sm443=sortrows(Tcells,'Element'); %Create new table sorted by Elemental Wavelength
CellConc_Sm443([1:48,73:end],:)=[]; %Delete all data except those at Wavelength 443 nm
CellConc_Sm446=sortrows(Tcells,'Element'); %Create new table sorted by Elemental Wavelength
CellConc_Sm446(1:72,:)=[]; %Delete all data except those at Wavelength 443 nm


%Convert table data and put on per billion cell basis
CellConc_Sm428=table2array(CellConc_Sm428(:,5))/1000000000;
Sm428Init_pbcells=Sm_428Initumol./CellConc_Sm428;
Sm428Post_pbcells=Sm_428Postumol./CellConc_Sm428;
Sm428Wash_pbcells=Sm_428Washumol./CellConc_Sm428;
Sm428Pellet_pbcells=Sm_428Pelletumol./CellConc_Sm428;

Ci_428=Sm428Init_pbcells;
Ce_428=Sm428Post_pbcells+Sm428Wash_pbcells;
Qe_428=Sm428Pellet_pbcells;
Ce428_Qe428=Ce_428./Qe_428;

CellConc_Sm443=table2array(CellConc_Sm443(:,5))/1000000000;
Sm443Init_pbcells=Sm_443Initumol./CellConc_Sm443;
Sm443Post_pbcells=Sm_443Postumol./CellConc_Sm443;
Sm443Wash_pbcells=Sm_443Washumol./CellConc_Sm443;
Sm443Pellet_pbcells=Sm_443Pelletumol./CellConc_Sm443;

Ci_443=Sm443Init_pbcells;
Ce_443=Sm443Post_pbcells+Sm443Wash_pbcells;
Qe_443=Sm443Pellet_pbcells;
Ce443_Qe443=Ce_443./Qe_443;

CellConc_Sm446=table2array(CellConc_Sm446(:,5))/1000000000;
Sm446Init_pbcells=Sm_446Initumol./CellConc_Sm446;
Sm446Post_pbcells=Sm_446Postumol./CellConc_Sm446;
Sm446Wash_pbcells=Sm_446Washumol./CellConc_Sm446;
Sm446Pellet_pbcells=Sm_446Pelletumol./CellConc_Sm446;

Ci_446=Sm446Init_pbcells;
```

```
Ce_446=Sm446Post_pbcells+Sm446Wash_pbcells;
Qe_446=Sm446Pellet_pbcells;
Ce446_Qe446=Ce_446./Qe_446;

%Average Data of Repeats
Sm428InitAvg_pbcells=mean(reshape(Sm428Init_pbcells,4,[]))';
Sm428PostAvg_pbcells=mean(reshape(Sm428Post_pbcells,4,[]))';
Sm428WashAvg_pbcells=mean(reshape(Sm428Wash_pbcells,4,[]))';
Sm428PelletAvg_pbcells=mean(reshape(Sm428Pellet_pbcells,4,[]))';

Sm443InitAvg_pbcells=mean(reshape(Sm443Init_pbcells,4,[]))';
Sm443PostAvg_pbcells=mean(reshape(Sm443Post_pbcells,4,[]))';
Sm443WashAvg_pbcells=mean(reshape(Sm443Wash_pbcells,4,[]))';
Sm443PelletAvg_pbcells=mean(reshape(Sm443Pellet_pbcells,4,[]))';

Sm446InitAvg_pbcells=mean(reshape(Sm446Init_pbcells,4,[]))';
Sm446PostAvg_pbcells=mean(reshape(Sm446Post_pbcells,4,[]))';
Sm446WashAvg_pbcells=mean(reshape(Sm446Wash_pbcells,4,[]))';
Sm446PelletAvg_pbcells=mean(reshape(Sm446Pellet_pbcells,4,[]))';

Ci_428Avg=Sm428InitAvg_pbcells;
Ce_428Avg=Sm428PostAvg_pbcells+Sm428WashAvg_pbcells;
Qe_428Avg=Sm428PelletAvg_pbcells;
Ce428Avg_Qe428Avg=Ce_428Avg./Qe_428Avg;

Ci_443Avg=Sm443InitAvg_pbcells;
Ce_443Avg=Sm443PostAvg_pbcells+Sm443WashAvg_pbcells;
Qe_443Avg=Sm443PelletAvg_pbcells;
Ce443Avg_Qe443Avg=Ce_443Avg./Qe_443Avg;

Ci_446Avg=Sm446InitAvg_pbcells;
Ce_446Avg=Sm446PostAvg_pbcells+Sm446WashAvg_pbcells;
Qe_446Avg=Sm446PelletAvg_pbcells;
Ce446Avg_Qe446Avg=Ce_446Avg./Qe_446Avg;

%Standard Deviation of Replicates
Sm428InitStdev_pbcells=std(reshape(Sm428Init_pbcells,4,[]))';
Sm428PostStdev_pbcells=std(reshape(Sm428Post_pbcells,4,[]))';
Sm428WashStdev_pbcells=std(reshape(Sm428Wash_pbcells,4,[]))';
Sm428PelletStdev_pbcells=std(reshape(Sm428Pellet_pbcells,4,[]))';

Sm443InitStdev_pbcells=std(reshape(Sm443Init_pbcells,4,[]))';
Sm443PostStdev_pbcells=std(reshape(Sm443Post_pbcells,4,[]))';
Sm443WashStdev_pbcells=std(reshape(Sm443Wash_pbcells,4,[]))';
Sm443PelletStdev_pbcells=std(reshape(Sm443Pellet_pbcells,4,[]))';

Sm446InitStdev_pbcells=std(reshape(Sm446Init_pbcells,4,[]))';
Sm446PostStdev_pbcells=std(reshape(Sm446Post_pbcells,4,[]))';
Sm446WashStdev_pbcells=std(reshape(Sm446Wash_pbcells,4,[]))';
Sm446PelletStdev_pbcells=std(reshape(Sm446Pellet_pbcells,4,[]))';

Ci_428Stdev=Sm428InitStdev_pbcells;
Ce_428Stdev=Sm428PostStdev_pbcells+Sm428WashStdev_pbcells;
Qe_428Stdev=Sm428PelletStdev_pbcells;
```

```
Ce428Stdev_Qe428Stdev=Ce_428Stdev./Qe_428Stdev;

Ci_443Stdev=Sm443InitStdev_pbcells;
Ce_443Stdev=Sm443PostStdev_pbcells+Sm443WashStdev_pbcells;
Qe_443Stdev=Sm443PelletStdev_pbcells;
Ce443Stdev_Qe443Stdev=Ce_443Stdev./Qe_443Stdev;

Ci_446Stdev=Sm446InitStdev_pbcells;
Ce_446Stdev=Sm446PostStdev_pbcells+Sm446WashStdev_pbcells;
Qe_446Stdev=Sm446PelletStdev_pbcells;
Ce446Stdev_Qe446Stdev=Ce_446Stdev./Qe_446Stdev;


p_428=polyfit(Ce_428,Ce428_Qe428,1);%Obtain Fit Parameters for Linear Correlation Data
slope_428=p_428(1,1);
intercept_428=p_428(1,2);
x=(0.0001:.1:500)';
y_428=slope_428*x+intercept_428;
Q0_428=1/(slope_428);
kL_428=1/(intercept_428*Q0_428);
Qe_428calc=Q0_428*((kL_428*x)./(1+(kL_428*x)));

Qe_428=Ci_428-Ce_428;
Qe_443=Ci_443-Ce_443;
Qe_446=Ci_446-Ce_446;

beta=[1,1]; %Initial Guesses for parameters
Langmuir=@(a,x)a(2)*((a(1)*(x))./(1+(a(1)*(x)))); %Create Langmuir Function
Langmuir_428=lsqcurvefit(Langmuir,beta,Ce_428,Qe_428,lbl,ubl) %Fit nonlinear Langmuir Function
%a=Langmuir_428.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Q0428=Langmuir_428(1,2)
kL428=Langmuir_428(1,1)
Langmuir_428nonlin=Q0428*(kL428*(x))./(1+(kL428*(x)));


beta2=[1,1]; %Initial Guesses for parameters
Freundlich=@(b,x)b(1)*(x).^(1/b(2)); %Create Freundlich Function
Freundlich_428=lsqcurvefit(Freundlich,beta2,Ce_428,Qe_428) %Fit nonlinear Freundlich Function
%b=Freundlich_428.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Kf_428=Freundlich_428(1,1);
nf_428=Freundlich_428(1,2);
Freundlich_428nonlin=Kf_428*(x).^(1/nf_428);


beta3=[1,1,1]; %Initial Guesses for parameters
Sips=@(c,x)c(3).*c(1).*((x).^c(2))./(1+c(1).*((x).^c(2))); %Create Langmuir Function assuming concentration
in 35mL (Derived from experiment used volumes)
Sips_428=lsqcurvefit(Sips,beta3,Ce_428,Qe_428,lbs,ubs) %Fit nonlinear Sips Function
% c=Sips_428.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
n428=Sips_428(1,2)
Keq428=Sips_428(1,1)
Qmax428=Sips_428(1,3)
Sips_428nonlin=Qmax428*Keq428*((x).^n428)./(1+Keq428*((x).^n428));
```

```
p_443=polyfit(Ce_443,Ce443_Qe443,1);%Obtain Fit Parameters for Linear Correlation Data
slope_443=p_443(1,1);
intercept_443=p_443(1,2);
x=(0.0001:.1:500)';
y_443=slope_443*x+intercept_443;
Q0_443=1/(slope_443);
kL_443=1/(intercept_443*Q0_443);
Qe_443calc=Q0_443*((kL_443*x)./(1+(kL_443*x)));

beta=[1,1]; %Initial Guesses for parameters
Langmuir=@(a,x)a(2)*((a(1)*(x))./(1+(a(1)*(x)))); %Create Langmuir Function
Langmuir_443=lsqcurvefit(Langmuir,beta,Ce_443,Qe_443,lbl,ubl) %Fit nonlinear Langmuir Function
%a=Langmuir_443.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Q0443=Langmuir_443(1,2)
kL443=Langmuir_443(1,1)
Langmuir_443nonlin=Q0443*(kL443*(x))./(1+(kL443*(x)));


beta2=[1,1]; %Initial Guesses for parameters
Freundlich=@(b,x)b(1)*(x).^(1/b(2)); %Create Freundlich Function
Freundlich_443=lsqcurvefit(Freundlich,beta2,Ce_443,Qe_443) %Fit nonlinear Freundlich Function
%b=Freundlich_443.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Kf_443=Freundlich_443(1,1);
nf_443=Freundlich_443(1,2);
Freundlich_443nonlin=Kf_443*(x).^(1/nf_443);


beta3=[1,1,1]; %Initial Guesses for parameters
Sips=@(c,x)c(3).*c(1).*((x).^c(2))./(1+c(1).*((x).^c(2))); %Create Langmuir Function assuming concentration
in 35mL (Derived from experiment used volumes)
Sips_443=lsqcurvefit(Sips,beta3,Ce_443,Qe_443,lbs,ubs) %Fit nonlinear Sips Function
% c=Sips_443.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
n443=Sips_443(1,2)
Keq443=Sips_443(1,1)
Qmax443=Sips_443(1,3)
Sips_443nonlin=Qmax443*Keq443*((x).^n443)./(1+Keq443*((x).^n443));


p_446=polyfit(Ce_446,Ce446_Qe446,1);%Obtain Fit Parameters for Linear Correlation Data
slope_446=p_446(1,1);
intercept_446=p_446(1,2);
x=(0.0001:.1:500)';
y_446=slope_446*x+intercept_446;
Q0_446=1/(slope_446);
kL_446=1/(intercept_446*Q0_446);
Qe_446calc=Q0_446*((kL_446*x)./(1+(kL_446*x)));

beta=[1,1]; %Initial Guesses for parameters
Langmuir=@(a,x)a(2)*((a(1)*(x))./(1+(a(1)*(x)))); %Create Langmuir Function
Langmuir_446=lsqcurvefit(Langmuir,beta,Ce_446,Qe_446,lbl,ubl) %Fit nonlinear Langmuir Function
%a=Langmuir_446.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Q0446=Langmuir_446(1,2)
```

```
kL446=Langmuir_446(1,1)
Langmuir_446nonlin=Q0446*(kL446*(x))./(1+(kL446*(x)));


beta2=[1,1]; %Initial Guesses for parameters
Freundlich=@(b,x)b(1)*(x).^(1/b(2)); %Create Freundlich Function
Freundlich_446=lsqcurvefit(Freundlich,beta2,Ce_446,Qe_446) %Fit nonlinear Freundlich Function
%b=Freundlich_446.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
Kf_446=Freundlich_446(1,1);
nf_446=Freundlich_446(1,2);
Freundlich_446nonlin=Kf_446*(x).^(1/nf_446);


beta3=[1,1,1]; %Initial Guesses for parameters
Sips=@(c,x)c(3).*c(1).*((x).^c(2))./(1+c(1).*((x).^c(2))); %Create Langmuir Function assuming concentration
in 35mL (Derived from experiment used volumes)
Sips_446=lsqcurvefit(Sips,beta3,Ce_446,Qe_446,lbs,ubs) %Fit nonlinear Sips Function
% c=Sips_446.Coefficients.Estimate; %Pull Estimated coefficients from fitnlm function
n446=Sips_446(1,2)
Keq446=Sips_446(1,1)
Qmax446=Sips_446(1,3)
Sips_446nonlin=Qmax446*Keq446*((x).^n446)./(1+Keq446*((x).^n446));


%Binding Site Affinity (Phi)
Phi_428=Qe_428*100/Qmax428;
Phi_443=Qe_443*100/Qmax443;
Phi_446=Qe_446*100/Qmax446;

%Plots
figure(301)
plot(Ce_428,Qe_428,'*',x,Langmuir_428nonlin,x,Freundlich_428nonlin,x,Sips_428nonlin)
title('Sorption Isotherms for Sm @ 428.967nm')
legend('Experimental Data','Nonlinear Langmuir Fit','Nonlinear Freundlich Fit','Nonlinear Sips
Fit','Location','northeastoutside')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

figure(302)
plot(Ce_443,Qe_443,'*',x,Langmuir_443nonlin,x,Freundlich_443nonlin,x,Sips_443nonlin)
title('Sorption Isotherms for Sm @ 443.967nm')
legend('Experimental Data','Nonlinear Langmuir Fit','Nonlinear Freundlich Fit','Nonlinear Sips
Fit','Location','northeastoutside')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

figure(303)
plot(Ce_446,Qe_446,'*',x,Langmuir_446nonlin,x,Freundlich_446nonlin,x,Sips_446nonlin)
title('Sorption Isotherms for Sm @ 446.967nm')
legend('Experimental Data','Nonlinear Langmuir Fit','Nonlinear Freundlich Fit','Nonlinear Sips Fit')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')
xlim([0 300])
figure(304)
```

```
plot(Ce_428,Ce428_Qe428,'*',Ce_443,Ce443_Qe443,'x',Ce_443,Ce443_Qe443,'o',x,y_428,x,y_443,x,y_446)
title('Linearized Langmuir Correlation')
legend('Experimental Data 428nm','Experimental Data 443nm','Experimental Data 446nm','Sm 428nm','Sm
443nm','Sm 446nm')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

figure(305)
plot(Ce_428,Qe_428,'*',Ce_443,Qe_443,'x',Ce_446,Qe_446,'o',x,Langmuir_428nonlin,x,Langmuir_443nonlin,
x,Langmuir_446nonlin)
title('Nonlinear Langmuir Fit')
legend('Experimental Data 428nm','Experimental Data 443nm','Experimental Data 446nm','Sm 428nm','Sm
443nm','Sm 446nm')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

figure(306)
plot(Ce_428,Qe_428,'*',Ce_443,Qe_443,'x',Ce_446,Qe_446,'o',x,Freundlich_428nonlin,x,Freundlich_443nonli
n,x,Freundlich_446nonlin)
title('Nonlinear Freundlich Fit')
legend('Experimental Data 428nm','Experimental Data 443nm','Experimental Data 446nm','Sm 428nm','Sm
443nm','Sm 446nm')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

figure(307)
plot(Ce_428,Qe_428,'*',Ce_443,Qe_443,'x',Ce_446,Qe_446,'o',x,Sips_428nonlin,x,Sips_443nonlin,x,Sips_446
nonlin)
title('Nonlinear Sips Fit')
legend('Experimental Data 428nm','Experimental Data 443nm','Experimental Data 446nm','Sm 428nm','Sm
443nm','Sm 446nm')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

figure(308)
plot(Ci_428,Ce_428,'*',Ci_443,Ce_443,'x',Ci_446,Ce_446,'o')
title('Postsorption Aqueous Concentration vs Initial Aqueous Concentration')
legend('Sm 428nm','Sm 443nm','Sm 446nm')
xlabel('Initial Concentration (umols/billion cells)')
ylabel('Equilibrium Concentration (umols/billion cells)')

figure(309)
plot(Ci_428,Phi_428,'*',Ci_443,Phi_443,'x',Ci_446,Phi_446,'o')
title('Percent Binding Sites Bound vs Initial Aqueous Concentration')
legend('Sm 428nm','Sm 443nm','Sm 446nm')
xlabel('Initial Concentration (umols/billion cells)')
ylabel('Percent Binding Sites Bound ')

%Phi_381Avg=Qe_381Avg*100/Qmax381;
%Phi_401Avg=Qe_401Avg*100/Qmax401;
%Phi_446Avg=Qe_446Avg*100/Qmax446;

Phi_381Avg=(Keq381.*Ce_381Avg)*100./(1+Keq381.*Ce_381Avg);
Phi_430Avg=(Keq430.*Ce_430Avg)*100./(1+Keq430.*Ce_430Avg);
```

```
Phi_428Avg=(Keq428.*Ce_428Avg)*100./(1+Keq428.*Ce_428Avg);

%Standard Deviation of Replicates
Phi_381Stdev=std(reshape(Phi_381,4,[]))';
Phi_430Stdev=std(reshape(Phi_430,4,[]))';
Phi_428Stdev=std(reshape(Phi_428,4,[]))';


p_381=polyfit(Ci_381,Phi_381,3)
Phifunc_401=polyfit(Ci_401,Phi_401,2)
Phifunc_446=polyfit(Ci_446,Phi_446,2)
y=linspace(0,1,max(Ci_381));
f1=polyval(p_381,y);
figure(10)
plot(y,f1)

% figure(11)
% plot(Phifunc_401,Ci_401,Phi_401)
%
% figure(12)
% plot(Phifunc_446,Ci_446,Phi_446)

%Combined Plots
figure(1)
plot(x,Sips_412nonlin,x,Sips_430nonlin,x,Sips_446nonlin)
title('REE Sips Isotherm Comparison')
legend('Europium','Neodynium','Samarium')
xlabel('Equilibrium Concentration (umols/billion cells)')
ylabel('Sorbed Concentration (umols/billion cells)')

% figure(2)
% plot(Ci_381,Ce_381,'*',Ci_401,Ce_401,'x',Ci_446,Ce_446,'o')
% title('Postsorption Aqueous Concentration vs Initial Aqueous Concentration')
% legend('Europium','Neodynium','Samarium')
% xlabel('Initial Concentration (umols/billion cells)')
% ylabel('Equilibrium Concentration (umols/billion cells)')

figure(3)
plot(Ci_381Avg,Phi_381Avg,'*',Ci_430Avg,Phi_430Avg,'x',Ci_446Avg,Phi_428Avg,'o')
title('Percent Binding Sites Bound vs Initial Aqueous Concentration','FontSize',30)
legend('Europium','Neodymium','Samarium','Location','southeast')
ylim([0 105])
xlabel('Initial Concentration (umols/billion cells)','FontSize',24)
ylabel('Percent Binding Sites Bound','FontSize',24)
h=legend;
set(h,'FontSize',16);


figure(4)
ax=[40 60 80 100 200 300];
y=[Ce_381Avg(1) Qe_381(1);Ce_381Avg(2) Qe_381(2);Ce_381Avg(3) Qe_381(3);Ce_381Avg(4)
Qe_381(4);Ce_381Avg(5) Qe_381(5);Ce_381Avg(6) Qe_381(6)];
bar(ax,y,'stacked')
```

toc