A Three-Dimensional Heat Map Matrix for Showing Co-relationships in Network Analysis

A Thesis

Presented in Partial Fulfillment of the Requirements for the

Degree of Master of Science

with a

Major in Computer Science

in the

College of Graduate Studies

University of Idaho

by

Bhuwan Lal Madhikarmi

Major Professor: Xiaogang Ma, Ph.D.

Committee Members: Clint Jeffery, Ph.D.; Kyle I. S. Harrington, Ph.D.

Department Administrator: Frederick T. Sheldon, Ph.D.

May 2018

Authorization to Submit Thesis

This thesis of Bhuwan L. Madhikarmi, submitted for the degree of Master of Science with a Major in Computer Science and titled **“A Three-Dimensional Heat Map Matrix for Showing Co-relationships in Network Analysis,"** has been reviewed in final form. Permission, as indicated by the signatures and dates given below, is now granted to submit final copies to the College of Graduate Studies for approval.

Major Professor: _____ Date: _____

Xiaogang Ma, Ph.D.

Committee Members: _____ Date: _____

Clint Jeffery, Ph.D.

_____ Date: _____

Kyle I. S. Harrington, Ph.D.

Department Administrator: _____ Date: _____

Frederick T. Sheldon, Ph.D.

Abstract

Many datasets are representations of networks in the real world. Exploratory data analysis, as a proven method in data science, can be used to discover patterns in networks and lead to meaningful questions for detailed data analysis. Showing co-occurrence of two related items is a widely used method in the exploratory data analysis of networks. The discovered co-occurrence patterns not only make the association visible, but also provide clues to predict future co-occurrence if the networks scale up. There are many visual techniques to show the co-occurrence association, such as D3 heat map and nodes-relationship cluster graph. Most of those techniques generate two dimensional diagrams as the visualization output. This thesis focuses on adding one more dimension to existing two dimensional heat map to show the co-occurrence between three items. The output is represented in a user interactive three-dimensional matrix. Several functions are developed to support the interactions between the user and the dataset. Among them a key function is a selection panel so, instead of loading a huge dataset, the user can choose records of interest to analyze. The usefulness of the developed three-dimensional heat map is reflected in two successful case studies. One is the co-occurrence of elements in the formation of minerals species, and the other is the co-occurrence of topics in the research interests of people at the University of Idaho. The exploratory data analysis carried out in these two case studies shows interesting patterns of co-occurrence, and helps generate a few more thoughts and ideas for further data analysis. With small adaptations, the output of this research can also be applied to conduct visual co-occurrence analysis in other disciplines.

Acknowledgement

I would like to express my sincere gratitude to my major professor Dr. Xiaogang (Marshall) Ma for his continuous support on my Master's study and research. Because of his superb guidance and suggestions, I learnt many research techniques and writing skills. I feel grateful for getting him as my advisor for my MS study.

I would also like to thank my committee members, Dr. Clint Jeffery and Dr. Kyle I. S. Harrington for their valuable inputs on my thesis. Special thanks to University of Idaho, Computer Science faculty and staffs for providing me knowledge and an environment which enriched me to grow academically. I am also thankful to my funding agencies for my research namely, the Deep Time Data Infrastructure (DTDI), the National Science Foundation (NSF) and the University of Idaho (U of I).

Table of Contents

List of Figures

List of Tables

List of Acronyms

CAES	Center for Advanced Energy Studies
CAVE	Computer Assisted Virtual Environment
CDC	Centers for Disease Control and Prevention
CSV	Comma Separated Values
DOM	Document Object Model
DTDI	Deep Time Data Infrastructure
EDA	Exploratory Data Analysis
EPSCoR	Established Program to Stimulate Competitive Research
FOV	Field of View
IRI	Internationalized Resource Indicator
JS	JavaScript
JSON	JavaScript Object Notation
NSF	National Science Foundation
RDF	Resource Description Format
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
U of I	University of Idaho
URI	Universal Resource Indicator
URL	Universal Resource Locator
WebGL	Web Graphics Library

Chapter 1: Introduction

This chapter firstly describes the brief introduction about the background of the thesis, the purpose and motivation of the project and the technological approach taken to do the project.

1.1 Introduction and Background

Data Science is an emerging science in this new era of massive data. It is not an independent science. It's an interdisciplinary field that combines Statistics, Data Analysis, Machine Learning and many related areas. It tends to extract meaning from data. The meaning extracted using various data science principles must be presented in some form for the users to understand. This is mostly achieved by Visual Communication, or more specifically, Visualization. Visualization should be effective both in functionality and beauty to present information in a clear and effective manner (Friedman, 2008). There are many types of visualization like tables, charts, graphs, maps, heat maps etc. This thesis is focused on the heat map. A heat map represents the data in the form of colors. A two dimensional heat map can be used to show a co-occurrence or co-relationship between two variables, which is discussed in chapter 2.6. Our research added one dimension to the two dimensional heat map to visualize co-occurrence or co-relationship between three variables using two case studies which are discussed in detail in chapter 4.1 and 4.2. We have provided user interaction with the visualization giving options to select a subset of the dataset and visualize only a small portion of the whole dataset. Other user interactions with the application we developed are discussed in chapter 4. This research project is funded by Deep Time Data Infrastructure (DTDI), NSF - Idaho Established Program to Stimulate Competitive Research (EPSCoR) and University of Idaho (U of I). The output from the first case of study of this research has been published in the International Journal of Geo-Information:

Ma, X., Hummer, D., Golden, J.J., Fox, P.A., Hazen, R.M., Morrison, S.M., Downs, R.T., Madhikarmi, B.L., Wang, C. and Meyer, M.B., 2017. (2017) Using Visual Exploratory Data Analysis to Facilitate Collaboration and Hypothesis Generation in Cross-Disciplinary Research. *ISPRS International Journal of Geo-Information*, *6*(11), p.368.

There are many three dimensional visualization libraries that can be used to plot data in a three dimensional matrix. Some of these works are Mayavi (Ramachandran and Gael, 2011), Matplotlib (Hunter, 2007) and the deck.gl framework (Belmonte, 2016). All of these frameworks and libraries provide the user a way to visualize data in three dimensions. Mayavi and Matplotlib take three variables whereas deck.gl takes only two variables. We have used Three.js, a JavaScript library for Web Graphics Library (WebGL) (Three.js, 2018) and three dimensional visualization for this research. The advantage of using the Three.js library is that the visualization can be interactive to the user through a web browser. We have provided the user with various forms of flexibility and interactivity, which are discussed more in-depth in chapter 4.

1.2 Purpose and Motivation

This work is a part of a Data Science process with participants from both Geoscience and Computer Science. Our purpose is to develop a quick prototype to meet the needs of collaboration between the Geoscientists and Computer Scientists. By using the visualization platform we can facilitate the collaboration and help them to get some insights from the dataset and generate research questions for detailed data analysis in the next stage. Computer scientists are concerned about the visualization of co-relationships of the dataset and the interpretation part is done by Geoscientists. This Computer Science thesis only covers the visualization part, not the interpretation part. Some of the research questions generated by the Geoscientists using this prototype are discussed in Chapter 5.2.

This research was initially carried out to show co-relationships between the elements and mineral species in a three dimensional heat map matrix. After developing a pilot system for this case, we have further extended this project to make it more generalized so that it can be used for different disciplines. For this, we chose a dataset related to different research areas of researchers at the U of I and showed it in a three dimensional matrix using this system.

1.3 Technical Approach

The three dimensional heat map matrix is developed using a JavaScript Library Three.js. The dataset is imported into a JavaScript Object using a JavaScript Library Papaparse.js

(Papaparse, 2018). The item selection from the full dataset and generation of corresponding small dataset is done using a JavaScript library Alasql.js (Alasql, 2018). The dataset about elements and mineral species is extracted from the rruff.info database at the University of Arizona. The U of I dataset for research areas and the number of researchers working on them is extracted using triplestore which can be accessed at http://vivo.nkn.uidaho.edu:3030/control-panel.tpl and the data cleansing and transformation is done using R Studio using a package called "SPARQL" (Hage and Kauppinen, 2011). Other JavaScript libraries used for this research project are Trackballcontrols.js (Cohen, 2014) and jQuery.js (The jQuery Foundation, 2018). The details for these technologies are covered in Chapter 3.

Chapter 2 will cover the basic concepts related to the project and related works. The details of use cases for this thesis work are covered in chapter 4. There are two use cases for which pilot systems are developed. The first use case is about the minerals and the elements forming those minerals, for which a dataset from DTDI is used. The second use case is about the sample 30 research areas of U of I and the number of researchers working on those research areas. Finally, the thesis concludes with chapter 5 describing the achievements and advantages of the project along with results, limitations and future work that can be carried out.

Chapter 2: Basic Concepts and Related Work

This chapter explores the basic concepts needed for this thesis work. The related work to this thesis is also described in this chapter.

2.1 Data Science

Data science is an interdisciplinary field that deals with knowledge discovery from data (Dhar, 2013). Data is useless until it can provide us with useful information. We are producing lots of data every day and we need a scientific method to extract useful information from it. Data science unifies different fields, including Statistics, Data Analysis, Machine learning, Artificial Intelligence and Domain Expertise and provides us with insights to understand the actual phenomena with data (Hayashi et al., 2013). The figure below shows Drew Conway's Venn diagram of data science.

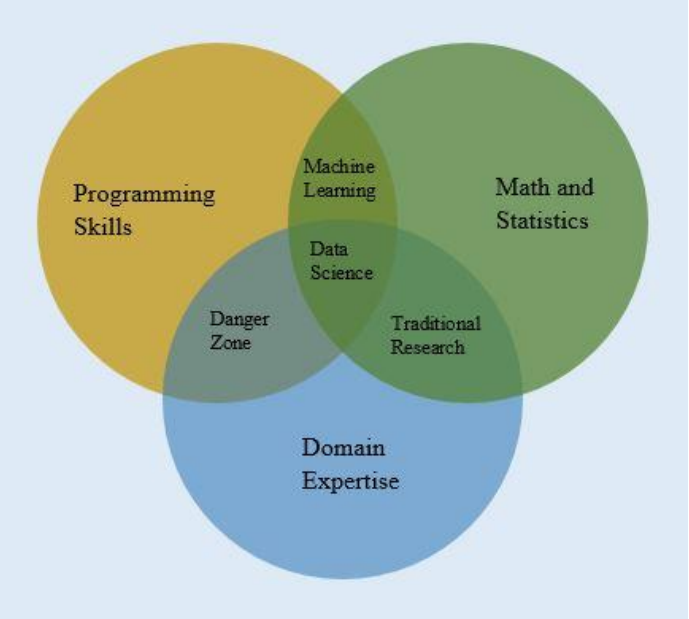

Figure 2.1: Drew Conway's Venn diagram of Data Science (Conway, 2010).

2.2 Exploratory Data Analysis

The data in raw format has no meaning until it is presented in an understandable form. Hence, the researcher must first be familiar with data, which will reduce the amount of work done for analysis (Cox, 2017). By name, Exploratory Data Analysis (EDA) represents the exploration

of data and analyzes different results that can be output from it. It is the preliminary step in Data Science after data collection and cleansing, the result of which might be different from what we expected, so it can be used to explore data properties which are present or absent in data under analysis (Tukey, 1977). With no statistical hypothesis, EDA helps researchers to understand the underlying structures of data and eliminate the interpretive errors during data analysis (Behrens and Yu, 2003). EDA is a crucial step in data science before applying confirmatory data analysis like machine learning algorithms and statistical models (Brillinger, 2011; Schutt and O'Neil, 2014; Ma et al., 2017).

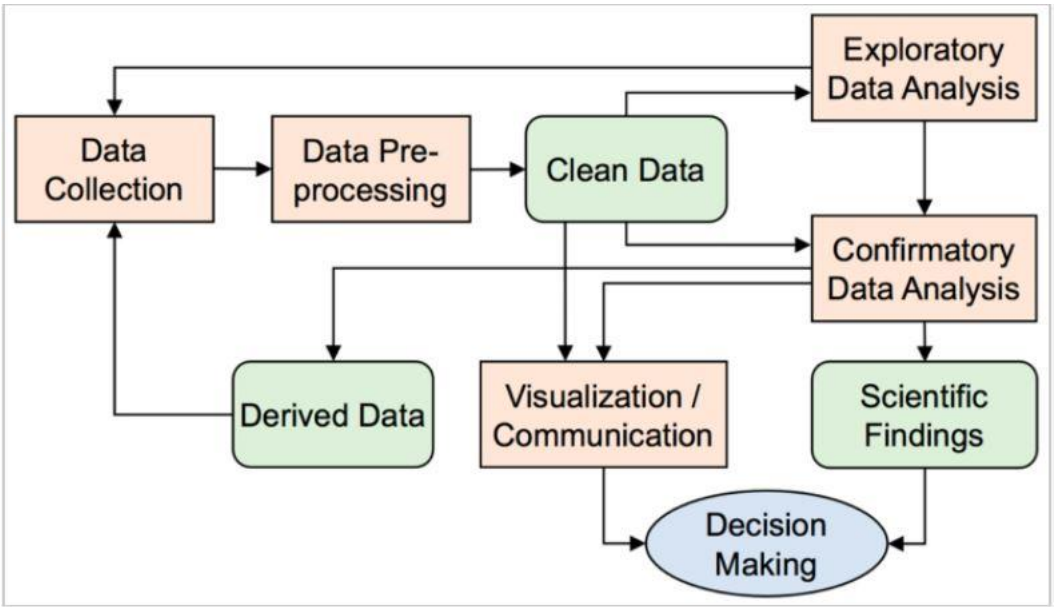

Figure 2.2: Steps in Data Science Process (Schutt and O'Neil, 2013; Ma et al., 2017).

2.3 Visual Data Analysis

Visualization is the method of representing data using pictures. Reading data is useless until it can give us a meaning. The meaning has two levels (Ma et al., 2015): first is to make something visible and second is to make it clear. Graphics are meaningful and easier to understand than reading words and numbers. The difference between reading graphics and texts/numbers has been explained by visual object perceptions studies (Cohen et al., 2002), which show that the human brain deciphers image and language in different ways – images are decoded simultaneously whereas language is decoded in a sequential way, which is slow compared to that for processing images. In contrast to the conventional way of visualizing only the end product, data visualization nowadays is applied in each step of data science (Fox

and Hendler, 2011). Visual Data Analysis is also used in Business Intelligence that incorporates different analytical tools into data to produce useful information for the decision making process (Negash, 2004).

The following chart shows an example of visual analytics, which is created using "highcharter" package in RStudio (Kunst, 2017). The chart shows the smoking trend in the United States, from 1995 through 2010, for four different levels: people those who never smoked, people who smoke some days, people who were former smokers and people who smoke every day. The dataset used for this analysis is downloaded from the Centers for Disease Control and Prevention (CDC) website in Comma Separated Values (CSV) format – "BRFSS_Prevalence_and_Trends_Data__Tobacco_Use_-_Four_Level_Smoking_Data_for_1995-2010.csv" (CDC, 2018).

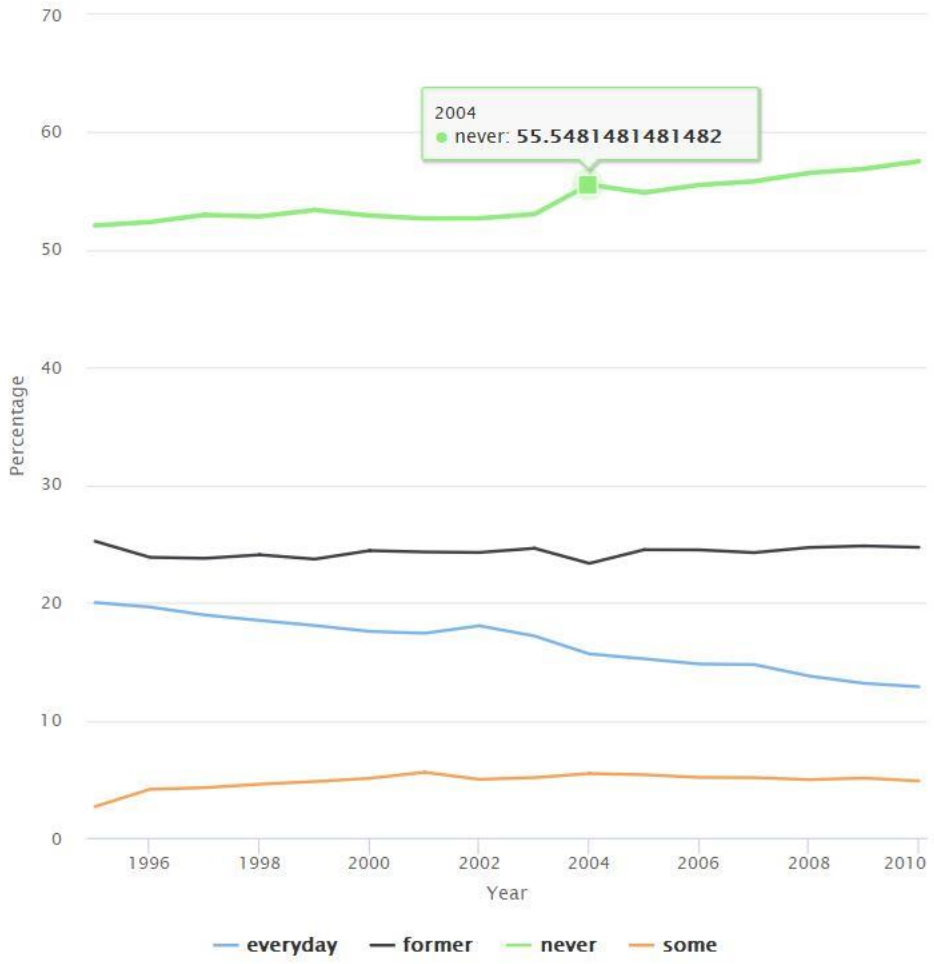

Figure 2.3: Smoking Trend in the United States, 1995–2010, generated using the dataset from CDC.

2.4 Co-occurrence

Co-occurrence shows how two or more keywords are related. In the Case Study I, co-occurrence means the occurrence of the same elements in a mineral formation. The values of co-occurrence denotes the number of minerals which constitutes same elements in its formation. The co-occurrence of keywords in research topics denotes the degree of closeness of two or more research areas and can be a very important measure to see the relation of one topic to another. For example, if the co-occurrence between two research topics "Semiconductor" and "Political-Science" is zero, which is quite obvious, it means that these two research topics are not related. But our dataset of co-occurrence among keywords shows a strong co-occurrence of value 12 for "Plant-Diseases" and "Weeds-Control," showing that these two research areas are closely related to each other. The concept of co-occurrence is useful in many fields like the one research done by Stapley and Benoit in 1999 that shows the occurrence of a pair of genes in similar literature in a specific type of yeast called "Saccharomyces cerevisiae" (Stapley and Benoit, 1999).

2.5 Open Data

In this digital age, the scientific data published by the scientific community should be reusable and there should not be any restrictions on the published data for the progress of scholarship (Murray-Rust, 2008). Open data is the process that defines the way of making data public and reusable without any restrictions. One example of open data is DBpedia which is a community effort to make the Wikipedia data into a structured form that allows both humans and machines to understand the information content in the data (Auer et al., 2007). A user can query this structured data using the endpoint provided and get the results from the linked open data.

2.6 Heat Map

A heat map, also known as a cluster heat map is a representation of two dimensional data with rows and columns that consists of rectangular areas or tiles shaded with different colors (Wilkinson and Friendly, 2009). The color indicates the level of co-relationship of row element with the column element. The darker the color, the stronger is the co-relationship between two elements and vice versa. A heat map compresses a huge amount of information

into a simple and small grid that can shows different coherent patterns in the data (Weinstein, 2008). Figure 2.4 below shows an example heat map generated from the dataset extracted from University of Idaho VIVO. The heat map is generated using d3heatmap package in RStudio (Cheng, 2015).

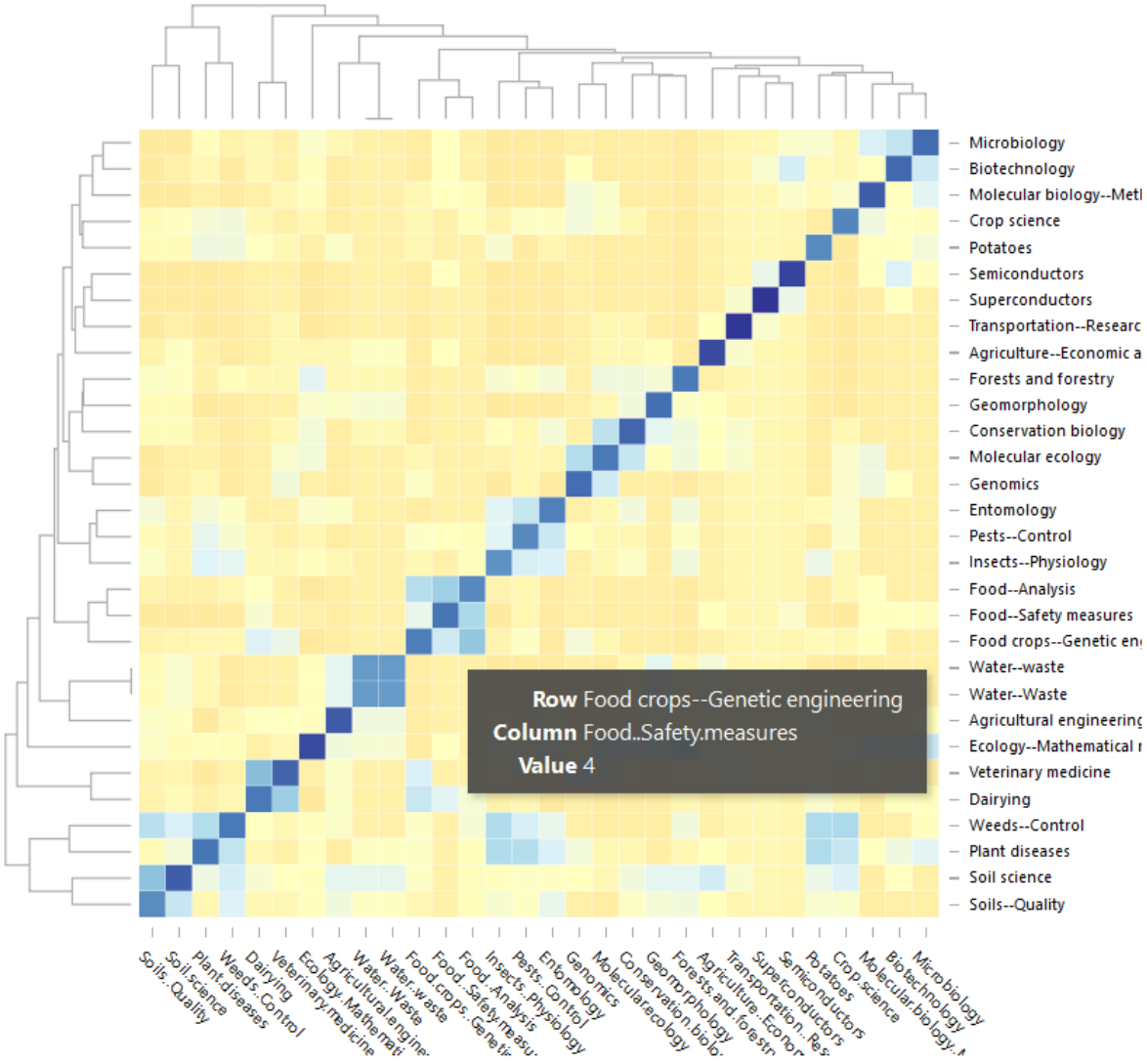

Figure 2.4: d3heatmap for the University of Idaho's 30 sample Research Areas.

2.7 Network Graph

A network graph contains vertices/nodes and edges/relationships. The nodes are connected through edges which could be directional or non-directional in nature. The edges can have weights assigned to them which indicates the level of bonding between the nodes. The higher

the weight, the higher the association or in simple terms, the relationship between the nodes and vice versa. There are many technologies that can build a network graph like Cytoscape (Shannon et al., 2003), Gephi (Bastian et al., 2009), Neo4j Graph database (Neo4j, "Graph NoSQL Database", 2012), RStudio (RStudio, "Integrated development for R", 2015) etc. The figure below shows the network graph for U of I six sample research areas and their co-relationships using "igraph" package in R Studio.

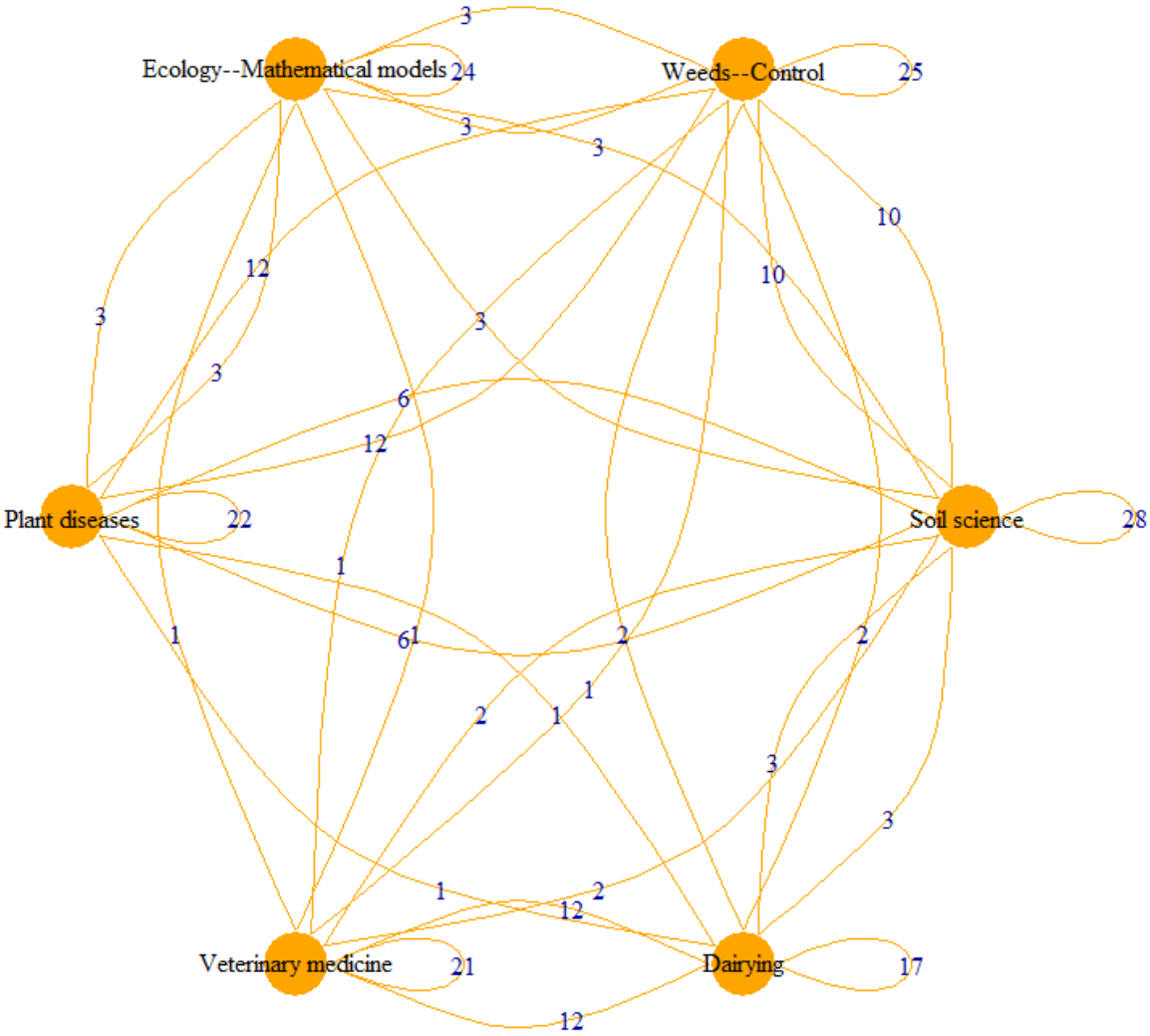

Figure 2.5: Network for the University of Idaho's six sample Research Areas.

In summary, this chapter described different topics that are needed to understand this work. It also tried to discuss some of the related works and the necessity of advancement to

the existing work and hence enlightened the importance of this thesis. Chapter 3 will discuss the technical aspects on which this research is done and also describe the source of dataset.

Chapter 3: Dataset and Technological Development

This chapter explains about the dataset considered for the research, its source and different scientific tools used to develop this three dimensional heat map visualization.

3.1 Dataset

The datasets for this project are taken from two different sources. The first dataset is open data and focuses on the co-relationships between elements and minerals, taken from the rruff.info database in DTDI, as mentioned above. The data collection and preprocessing of the minerals dataset was done in DTDI. There are three different datasets for the first case study. The first one has 30 key elements forming the minerals and the dataset is a 30x30x30 matrix along x, y and z-axes. The values inside the matrix represent the number of minerals in which the three elements along x, y, and z-axes co-existed or co-occurred. The second one is similar to the first use case, but the values inside the 30x30x30 matrix represent the normalized values of the first use case. The third represents 72 key mineral forming elements thus comprising a 72x72x72 matrix. Here, the values represent the chi-square test instead of raw minerals number.

The second dataset is collected from the University of Idaho's triplestore end point VIVO (University of Idaho VIVO, 2018) using SPARQL Protocol and RDF Query Language (SPARQL) (W3Schools, 2018b) and converted to CSV through a series of Extract, Transformation and Load (ETL) operations. Details about VIVO is covered in chapter 3.7. Only the top 30 research areas are considered for this pilot system based on the number of researchers working on those areas. Similar to the Case Study I, the 30 research areas were arranged into three and the values inside this 30x30x30 matrix represent the number of researchers that are involved in all the three research areas.

As this project deals with showing co-occurrence of three items, the dataset logically represents a three dimensional matrix but it is structured in a two dimensional CSV format by replicating the third axis (or z-axis) of the matrix in the x-axis.

Figure 3.1 shows the overall project data flow diagram from the source of data to the destination visualization page.

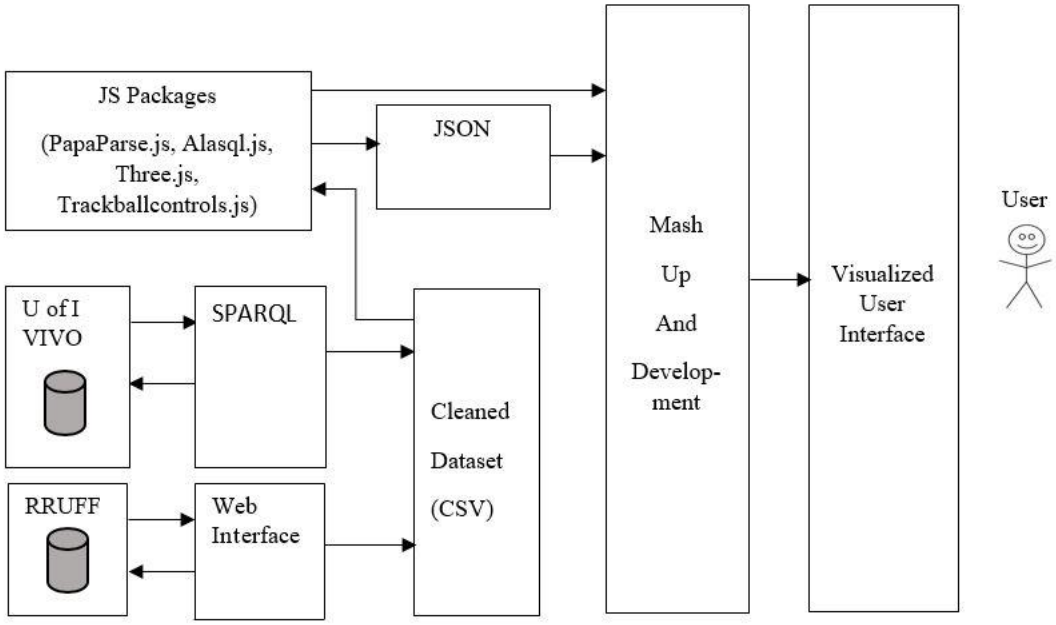

Figure 3.1: Overall Data Flow Diagram from Source to Destination.

After the data is downloaded from the sources for both the cases and converted to CSV format, JavaScript packages take these datasets through a series of transformations until they land on the visualization web page. First of all, the CSV data is imported into a JavaScript object in the browser using Papaparse.js library. Then Alasql.js library allows us to query the subset data of interest, which is converted to JavaScript Object Notation (JSON) and then stored in local storage of the browser. This stored JSON is passed to the visualization page in which it is again converted back to a JavaScript object. Now Three.js comes into action by reading this data row by row and building the three dimensional matrix of cubes, representing different values of the cubes with different colors. Figure 3.2 describes how the CSV data goes through different transformations and the three dimensional matrix of cubes are created.

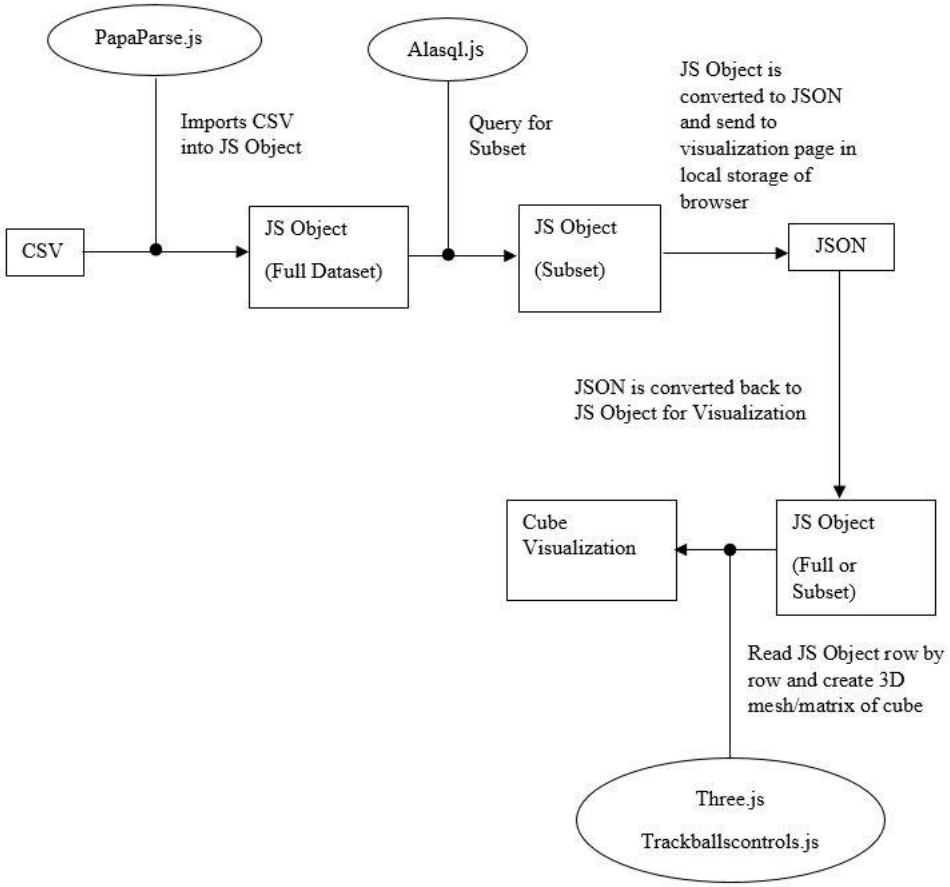

Figure 3.2: Functions of JavaScript Libraries and Data Transformations.

3.2 Papaparse.js

Papaparse.js is an easy and fast parser for CSV files with error handling features. It can stream large files and allows reverse parsing. It supports header row and has pause, resume and abort functions and is also capable of handling line breaks and quotations (Holt, 2018). It can be installed using npm (Nameless Package Manager) or the "papaparse.min.js" file can be directly linked to the project, which can be downloaded from GitHub https://github.com/mholt/PapaParse/blob/master/papaparse.min.js.

Using npm, the parser can be installed using with the command:

```
npm install papaparse
```

The CSV file can be parsed using Papa Parse using the following function:

```
Papa.parse(file,config)
```

Here `file` can be file name which is obtained from Document Object Model (DOM) or CSV string text or URL (Uniform Resource Locator) to the file. `config` is a configuration object (Papaparse, 2018) whose function is to handle settings, behavior and callbacks while parsing CSV files. The various properties that `config` takes are as follows:

`delimiter`: This property defines the delimiter used in the file. If left blank, then it will auto detect from a list of delimiters.

`newline`: The value can be "\n" or "\r"

`quoteChar`: This property defines the quoting field's character. It can be left blank for auto detection.

`header`: The header property defines whether the CSV has a header or not. The value can be either true or false.

`dynamicTyping`: If this is true, then the parser will treat the numbers and booleans into their type.

`preview`: If 0, then all the lines in the CSV will be parsed. If greater than 0, then only those rows will be parsed.

`encoding`: This property defines the encoding technique for opening local files.

`worker`: This property is defined if a worker thread is needed to keep the webpage reactive during parsing.

`comments`: Defines a character for comments' purpose.

`step`: This function activates streaming the output of parsing row by row.

`complete`: This callback function is called after the completion of parsing.

`error`: This callback function is executed if the file reader encounters any error.

`download`: In case of parsing remote file using URL, this value should be given 'true' value, else 'false'.

`skipEmptyLines`: If true, the empty lines are skipped.

`chunk`: This function will activate streaming the output of parsing chunk by chunk.

`fastMode`: For large files, it can be set true but this only works for files with no quoting fields.

`beforeFirstChunk`: This function is executed before parsing the first chunk.

`withCredentials`: It is a boolean value which is passed to "withCredentials" property of XMLHttpRequest.

The following sample code shows the example of use of `config` (Papaparse, 2018)

```
{
	delimiter: "",	// auto-detect
	newline: "",	// auto-detect
	quoteChar: '"', //the fields are quoted by double inverted
	comma
	header: true, // the csv file contains header
	dynamicTyping: false,
	preview: 0,
	encoding: "",
	worker: false, //no streaming activated
	comments: false, //no lines are to be treated as comments
	step: undefined,
	complete: undefined,
	error: undefined,
	download: false,
	skipEmptyLines: false,
	chunk: undefined,
	fastMode: undefined,
	beforeFirstChunk: undefined,
	withCredentials: undefined
}
```

After the successful parsing of the CSV file, the result contains three parts which are of type object: "data," "errors" and "meta." If there is a header in the CSV file, "data" contains objects of data, which are keyed by the field name else it contains an array of parsed CSV file content. "errors" is an array of errors if any. "meta" holds the parsing information like the delimiter, newline sequence etc.

Papaparse has an unparse capability. The following code snippet unparses an array of objects into CSV text strings:

```
Papa.unparse(data, config)
```

Here `data` can be an array of objects, or an array of arrays and `config` is optional, which defines different properties for un-parsing as discussed in the parsing section above (Papaparse, 2018).

3.3 Alasql.js

Alasql.js is a JavaScript Structured Query Language (SQL) Library and is an open source project. It is a client side database which allows fast processing for Business Intelligence (BI) and Enterprise Resource Planning (ERP) applications on thick clients and provides easy ETL functionalities (Gershun, 2018). It is supported in all major browsers, Node.js and mobile applications.

It can be installed using package managers like npm, bower or meteor. It can be used directly in the web application by downloading or linking the JavaScript file from http://alasql.org/.

```
npm install --save alasql        # using node
bower install --save alasql      # using bower
import alasql from 'alasql';     # using meteor
npm install -g alasql            # using command line
```

It can be used directly in the webpage including alasql.min.js.

```
<script src="//unpkg.com/alasql@0.4"></script>
```

After installation, it can be used as traditional relational database and write queries for data selection. The following code shows some uses of alasql as a traditional relational database.

```
alasql("CREATE TABLE table1(field1 string, field2 number)");
//creates table with 2 fields
alasql("INSERT INTO table1 VALUES ('example',100)"); //inserts
1 row into table1
```

```
var result = alasql("SELECT * from table1 WHERE field2=100");
//selects all rows with field2 = 100
```

Alasql can be used to query a JavaScript object. The function `alasql` is used for this purpose that takes 2 parameters: a query string and a JavaScript object and returns the result into a variable. In the code snippet below, Alasql takes "query" as one parameter and "dataObject" as another parameter and returns the result into a variable named "alaData".

```
var query = "SELECT * FROM ? where field1='hydrogen' ";
var alaData = (alasql(query, [dataObject]));
```

3.4 Three.js

Three.js is a easy to use and lightweight 3D JavaScript Library that provides <canvas>, <svg>, CSS3D and WebGL renderers (Mr.doob, GitHub). Using this JavaScript library, the 3D objects can be created and rendered in the client's browser. Three.js has two versions: Three.min.js and Three.js. The first version is a minimized version of Three.js which is normally used while deploying the sites in internet (Dirksen, 2013). The Three.js library can be included in HTML using <script> tag or it can be installed or imported using different packet managers.

```
<script src="js/three.min.js"></script>
```

The working of Three.js can be briefly described in following steps:

3.4.1 Creating the scene

There are three basic parts in Three.js: `scene`, `camera` and `renderer`. The following code snippet shows how the `scene`, `camera` and `renderer` are defined in Three.js library (Three.js, 2018).

```
Var scene = new THREE.Scene();
var camera = new THREE.PerspectiveCamera( 75,
window.innerWidth / window.innerHeight, 0.1, 1000 );
```

```
var renderer = new THREE.WebGLRenderer();
renderer.setSize( window.innerWidth, window.innerHeight );
document.body.appendChild( renderer.domElement );
```

The first attribute of PerspectiveCamera is the field of view (FOV) and the second attribute is aspect ratio. The third and fourth attributes define the near and far clipping plane; objects nearer than the near attribute value and farther than the far attribute value will not be rendered (Three.js, 2018).

The renderer uses the `<canvas>` element to display the scene in a webpage. Creating a cube requires objects from three classes: `BoxGeomtry`, `MeshBasicMaterial` and `Mesh`. `BoxGeometry` contains the vertices of the cube, `MeshBasicMaterial` specifies the color of the cube and the `Mesh` object takes Geometry and applies the material or color in the cube (Three.js, 2018).

```
var geometry = new THREE.BoxGeometry( 1, 1, 1 );
var material = new THREE.MeshBasicMaterial( { color: 0x00ff00
} );
var cube = new THREE.Mesh( geometry, material );
scene.add( cube );
```

3.4.2 Rendering the scene

The renderer will create a loop to draw the scene when the screen is refreshed. The `requestAnimationFrame` function here stops the renderer when the user navigates to another tab of the browser (Three.js, 2018).

```
function animate() {
      requestAnimationFrame( animate );
      renderer.render( scene, camera );
}
animate();
```

3.4.3 Animating the cube

The cube can be rotated using the rotation attribute of the cube Mesh object. For this research work, the 0.1 rotation increment has been used.

```
cube.rotation.x += 0.1;
cube.rotation.y += 0.1;
```

3.5 Trackballcontrols.js

Trackballcontrols.js is a JavaScript library that provides super smooth swipe of the camera around a single point in pairing with Three.js (Cohen, 2014). It is the most used JavaScript library for camera, which allows users to move, zoom or pan around the scene (Dirksen, 2013). The code snippet below shows an example of setting speed, zoom and pan controls values used in this research work:

```
controls = new THREE.TrackballControls( camera );
controls.rotateSpeed = 2.0;
controls.zoomSpeed = 1.2;
controls.panSpeed = 0.9;
```

3.6 JQuery.js

JQuery is free, open-source software which is a cross-platform JavaScript library specially designed for making client-side scripting of HTML easier (The jQuery Foundation, 2018). The jQuery library helps to minimize the JavaScript code into methods and simplifies the complicated things like AJAX and DOM manipulation (W3Schools, 2018a). This library handles HTML/DOM manipulation, HTML event methods handling, AJAX and CSS manipulation. The jQuery library can be downloaded from jQuery.com and can be included in HTML using <script> tag.

```
<script src = "jquery-1.9.1.js"></script>
```

The basic syntax for using jQuery is: $(selector).action() (W3Schools, 2018a). Here a dollar ($) sign defines a jQuery, selector is a query to find the HTML elements and action ()

is the action to be performed. The following code snippet shows the click function of a button from this thesis work:

```
$('#btnVisualizeSel').click(function () {
      ……
      ……
}
```

3.7 SPARQL

SPARQL stands for SPARQL Protocol and RDF Query Language. RDF stands for Resource Description format and is used to represent the World Wide Web (WWW) resources in a flexible and extensible format (Prud and Seaborne, 2006). RDF is in the form of triple components: subject, predicate and object. The three components, subject, predicate and object, form an RDF URI (Universal Resource Indicator) reference which produce a valid URI character sequence and do not contain any of these control characters – #x00 - #x1F, #x7F - #x9F (Klyne and Carroll, 2006). Sets of RDF triples form a RDF graph. XML is used to write the RDF documents in what is called RDF/XML. An example of RDF/XML is shown below (W3Schools, 2018b):

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.example.com/movie#">

<rdf:Description
rdf:about="http://www.example.com/movie/Example1">
  <cd:artist>Artist1</cd:artist>
  <cd:country>USA</cd:country>
  <cd:company>Company1</cd:company>
  <cd:price>20</cd:price>
  <cd:year>2000</cd:year>
</rdf:Description>
```

```
<rdf:Description
rdf:about="http://www.example.com/movie/Example2">
  <cd:artist>Artist2</cd:artist>
  <cd:country>USA</cd:country>
  <cd:company>Company2</cd:company>
  <cd:price>30</cd:price>
  <cd:year>2015</cd:year>
</rdf:Description>
.
.
.
</rdf:RDF>
```

SPARQL provides a standardized query language for RDF graphs (Segaran et al., 2009). The syntax of SPARQL is somehow similar to SQL. The keywords such as SELECT, DISTINCT, WHERE, JOIN, SORT, AGGREGATE, ORDER BY from SQL are also used in SPARQL. The keywords are case-insensitive but with an exception of 'a' which represents rdf:type IRI (Prud and Seaborne, 2006). The following table shows the list of keywords in SPARQL:

Table 3.1: List of Keywords in SPARQL (Prud and Seaborne, 2006)

BASE	SELECT	ORDER BY	FROM	GRAPH	STR	isURI
PREFIX	CONSTRUCT	LIMIT	FROM NAMED	OPTIONAL	LANG	isIRI
	DESCRIBE	OFFSET	WHERE	UNION	LANGMATCHES	isLITERAL
	ASK	DISTINCT		FILTER	DATATYPE	REGEX
		REDUCED		a	BOUND	true
					sameTERM	false

RDF databases are called RDF triplestore or SPARQL endpoint. For the second case study in this research work, the data is collected from the U of I triplestore called VIVO. VIVO was developed in Cornell University in 2003 and is a portal to connect and exchange

information about scholars and students through linked open data (Devare et al., 2007). The official website of VIVO describes VIVO as open-source software and ontology for representing scholarship (VIVO, About VIVO). The U of I VIVO can be accessed through the link http://vivo.nkn.uidaho.edu:3030/control-panel.tpl.

3.8 RStudio

RStudio is a free open source software and is very popular among the fields of science, education and industry for data analysis (RStudio, “Integrated development for R”, 2015). It has numerous packages for data analysis. The “SPARQL” package in R enables us to query any triplestore endpoint using SPARQL queries (Hage and Kauppinen, 2011). The dataset from U of I VIVO is collected via the SPARQL queries below using RStudio and “SPARQL” package in R:

```
library(SPARQL)
endpoint <- "http://vivo.nkn.uidaho.edu:3030/VIVO/query"
sparql_prefix <- "PREFIX rdf:   <http://www.w3.org/1999/02/22-
rdf-syntax-ns#>
PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
PREFIX owl:   <http://www.w3.org/2002/07/owl#>
PREFIX swrl:  <http://www.w3.org/2003/11/swrl#>
PREFIX swrlb: <http://www.w3.org/2003/11/swrlb#>
# PREFIX vitro:
<http://vitro.mannlib.cornell.edu/ns/vitro/0.7#>
PREFIX bibo: <http://purl.org/ontology/bibo/>
PREFIX c4o: <http://purl.org/spar/c4o/>
PREFIX cito: <http://purl.org/spar/cito/>
PREFIX event: <http://purl.org/NET/c4dm/event.owl#>
PREFIX fabio: <http://purl.org/spar/fabio/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX geo: <http://aims.fao.org/aos/geopolitical.owl#>
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX ocrer: <http://purl.org/net/OCRe/research.owl#>
```

```
PREFIX ocresd: <http://purl.org/net/OCRe/study_design.owl#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX uidaho: <http://vivo.nkn.uidaho.edu/ontology/uidaho#>
PREFIX vcard: <http://www.w3.org/2006/vcard/ns#>
PREFIX vitro-public:
<http://vitro.mannlib.cornell.edu/ns/vitro/public#>
PREFIX vivo: <http://vivoweb.org/ontology/core#>
PREFIX vlocal: <http://localhost/vivo/ontology/vivo-local#>
PREFIX scires: <http://vivoweb.org/ontology/scientific-
research#>"

query1 <- paste(sparql_prefix,
'SELECT
     ?label (COUNT(?count) AS ?no_count)
 WHERE
      {
      ?s vivo:hasResearchArea ?areas .
      ?areas rdfs:label ?label .
      ?areas vivo:researchAreaOf ?count .
      }GROUP BY ?label'
      )
options <- NULL
result1 <- SPARQL(endpoint,q,extra=options)$results

query2 <- paste(sparql_prefix,
'SELECT
     ?person ?area
WHERE
      {
      ?person vivo:hasResearchArea ?a .
      ?a rdfs:label ?area .
      }'
      )
options <- NULL
result2 <- SPARQL(endpoint,query2,extra=options)$results
```

In summary, chapter 3 discussed the details of tools used to develop this application. It also described the details of different libraries and their use in this project. Chapter 4 focuses on the two case studies that are considered for this thesis work.

Chapter 4: Demo System and Case Studies

This chapter presents the two case studies in which this research work is based.

4.1 Co-occurrence of Elements in Mineral Species

The web application designed here contains 3 options for choosing the dataset for three different datasets that are discussed above. The user can load a dataset using the 3 buttons provided in the web application. After choosing any one of the datasets, the elements on the periodic table that are present in the dataset are highlighted and other elements not present in the dataset are faded. For example, if a user clicks the 30x30x30 option, then only 30 of the elements in periodic table are highlighted and other remaining elements are faded, as shown in the figure below:

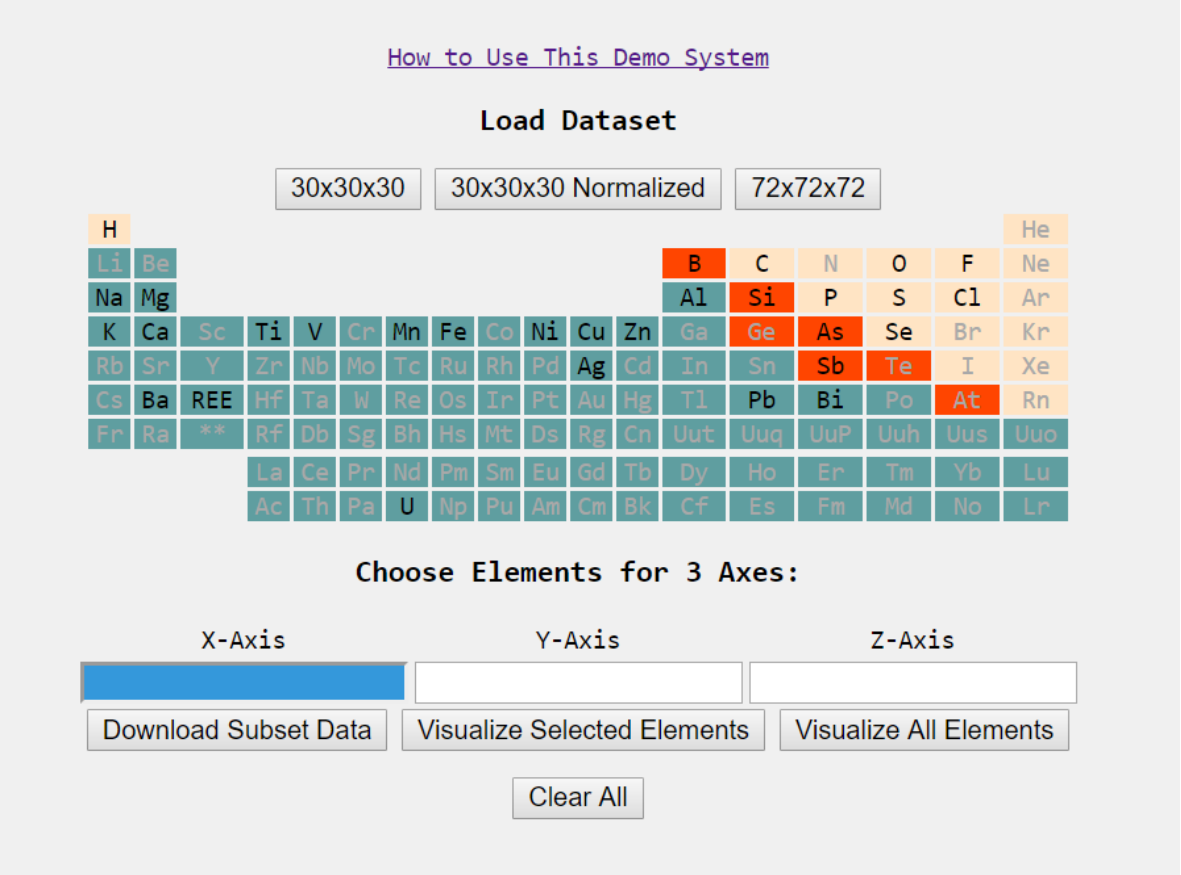

Figure 4.1: 30 Elements highlighted for the 30x30x30 dataset.

After the dataset is loaded into the periodic table, the user can choose elements in three axes for visualization. The element selection is done element by element for each axis. The

default axis for selection is x-axis. Whenever an element is selected, the element is highlighted in green and it will be listed in the text box just below the periodic table. For choosing elements for y-axis and z-axis, the user can click the y-axis and z-axis text box respectively. The chosen textbox axis is highlighted in blue. Switching between different axes for element selection is allowed. Whenever a different textbox axis is selected, the previously listed element for that axis will be shown with green highlight in the periodic table so that the user knows which elements are already selected. There is a feature of element removal from the selected list. If the element that is selected is clicked again, the element is removed from the selection list and un-highlighted from the periodic table.

The whole selection can be cleared by clicking the "Clear" button at the bottom of the web page. This function will take the application to the stage after dataset selection. All the highlighted and listed elements will be cleared from memory.

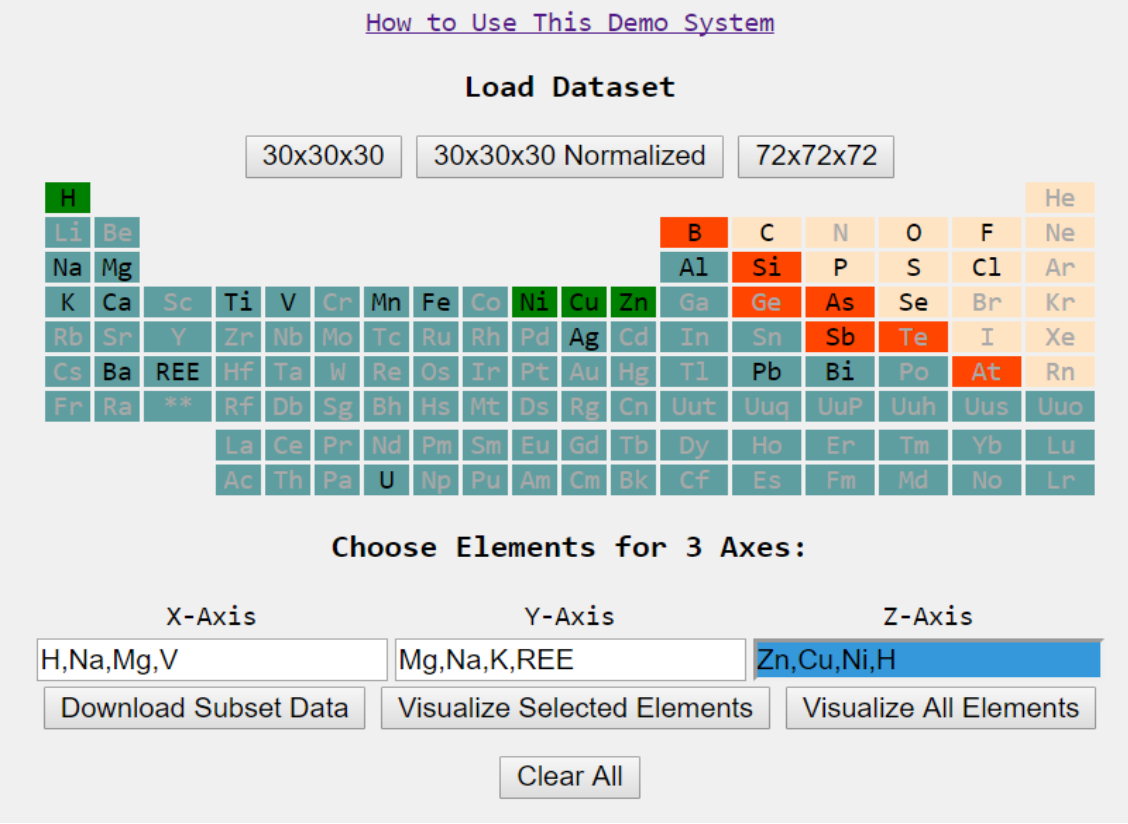

Figure 4.2: Selection of elements for the three different axes.

After the element selection, the user can download the subset data based on the elements selection on three axes. In this case, the downloaded dataset only contains the part of whole dataset with the axes elements specified by the user. The file is saved in CSV format. This download function is not available with the null selection, i.e., the user needs to select at least one element in each of the three axes.

There are two types of visualization options available. One is visualizing only the subset data based on the elements selected by the user, and second is visualizing the whole original dataset.

4.1.1 Visualize only the Selected Elements

This option will visualize the dataset based on the elements selected by the user. In this case, a new dataset is prepared using only the elements highlighted by the user on three axes. The advantage of visualizing only the small portion of the dataset is that the processing time for the cube generation is shorter and the result is shown in a small amount of time. This saves a huge amount of memory and the visualization is very quick.

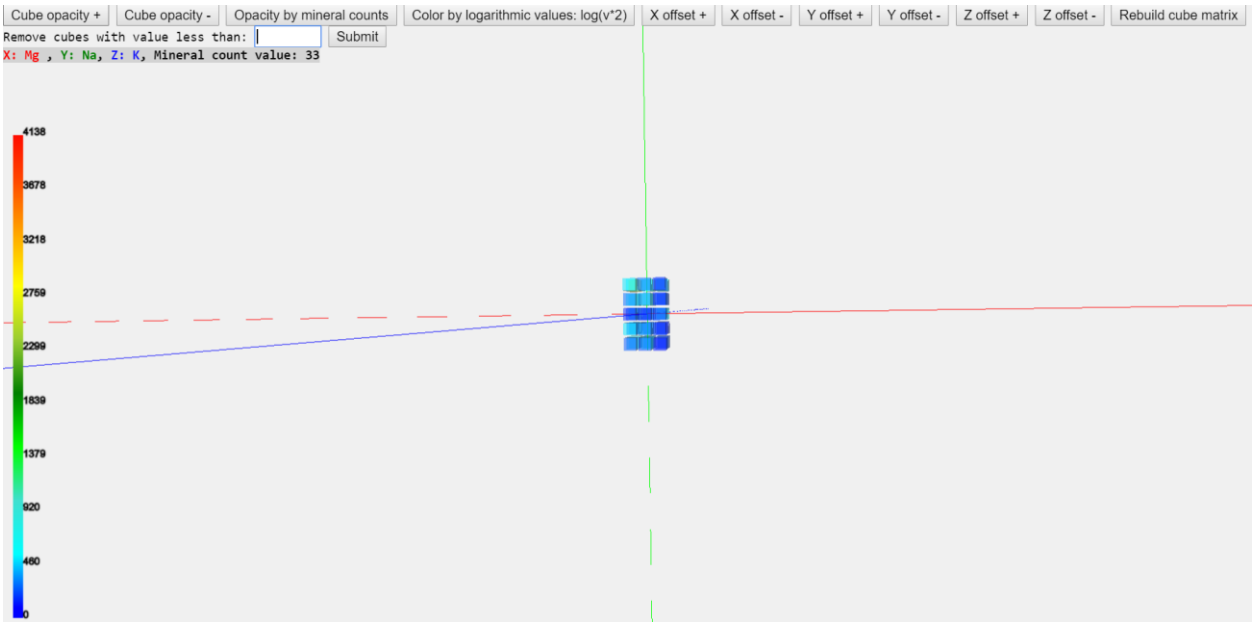

Figure 4.3: Visualizing only the selected elements by the user.

4.1.2 Visualize All the Elements

Based on the dataset that is loaded in the first step of this application, the whole dataset is considered for visualization when this option is selected.

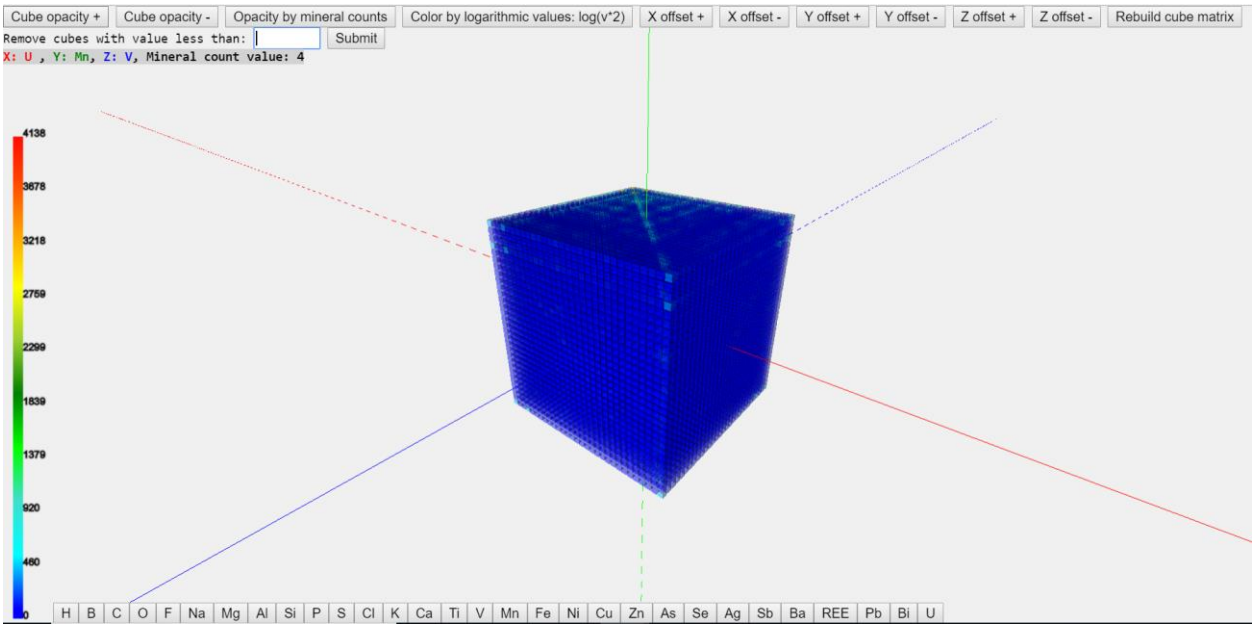

Figure 4.4: Visualize all the elements of the 30x30x30 dataset.

In the visualization page, the cubes are colored based on the values of the mineral counts. The color extends from blue through red. Small values are represented in blue and high values are represented in red. The minimum value in the color legend is zero in all cases of the three datasets while the maximum value is different for the three datasets. The maximum value for 30x30x30 dataset is 4138 and 30x30x30 normalized dataset is 1. For 72x72x72 dataset, it is 315. This color legend helps to visualize the cubes and get the rough estimate of the pattern in the cubes.

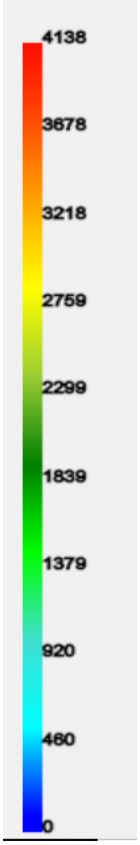

Figure 4.5: Color Legend for the 30x30x30 dataset.

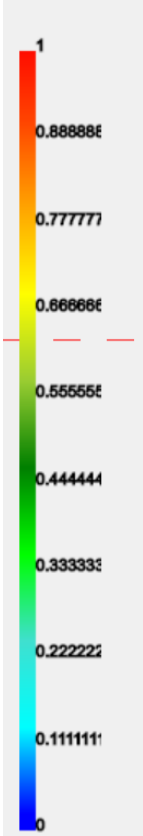

Figure 4.6: Color Legend for the normalized 30x30x30 dataset.

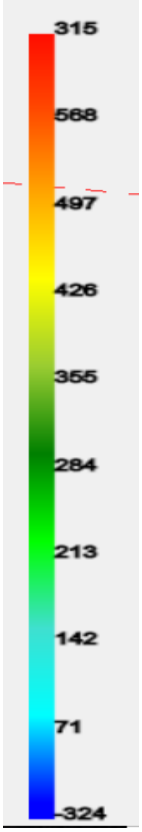

Figure 4.7: Color Legend for the 72x72x72 dataset.

The user can interact with the cube matrix visualization. Subsections 4.1.3 – 4.1.10 describes the functionalities that users can do with the cubes.

4.1.3 View the Values of the Cubes

When the mouse hovers around the cubes in the 3D mesh, the value corresponding to the cube is displayed at the top. It also shows the constituent elements and the count of minerals formed by the combination of these elements. The example in Figure 4.8 shows that there are 235 minerals which constitute H (Hydrogen), O (Oxygen) and U (Uranium).

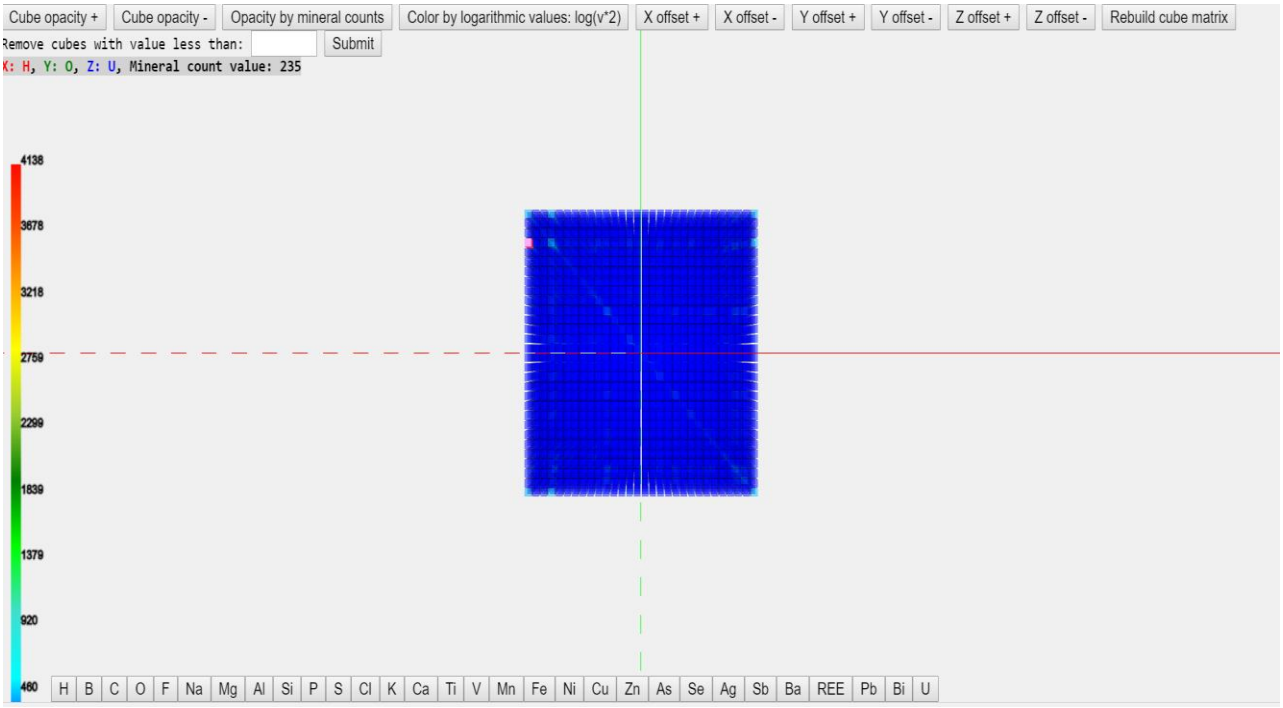

Figure 4.8: Display the value of a Cube.

4.1.4 Change Cubes' Opacity

Opacity means the opaqueness of the cubes. The cubes can be made lighter and darker by the user. When the "Increase Opacity" button is clicked, the color of whole cube mesh is made darker as shown in Figure 4.9.

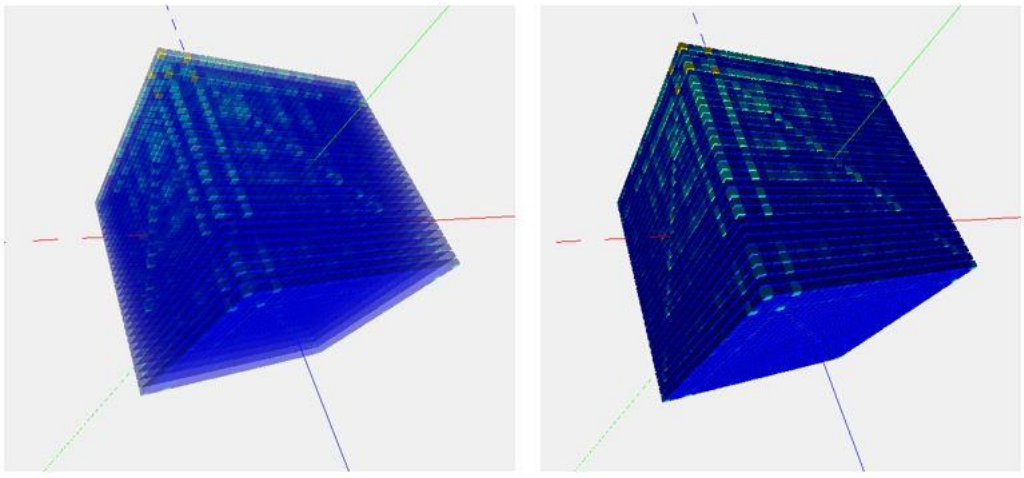

Figure 4.9: Increase in Opacity from left to right of the 30x30x30 dataset.

When the “Decrease Opacity” button is clicked, the whole cube mesh is made more transparent as shown in Figure 4.10.

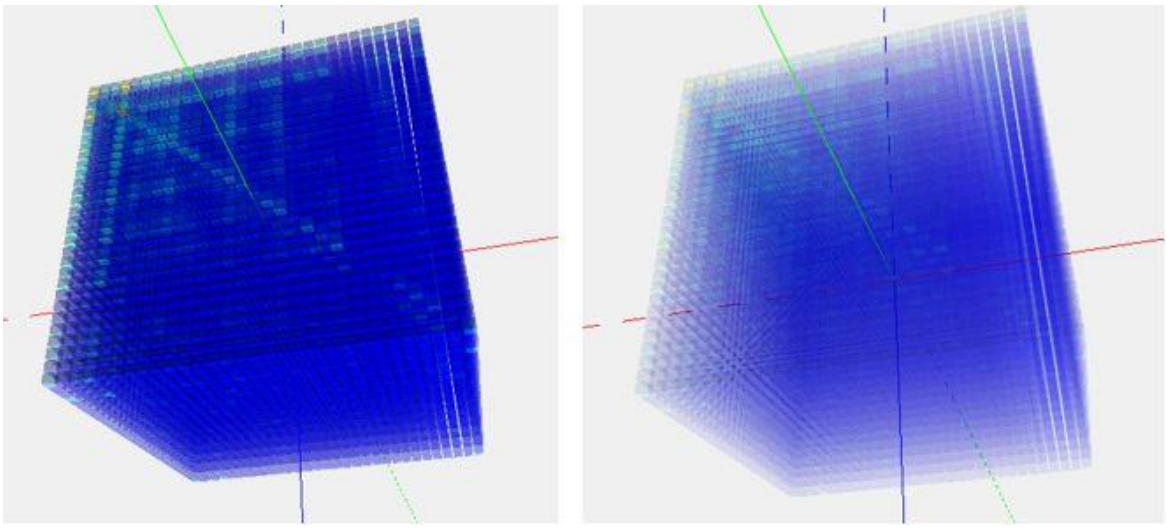

Figure 4.10: Decrease in Opacity from left to right of the 30x30x30 dataset.

4.1.5 Opacity by Minerals Count

This option will fade away the small values and only the cubes with higher values will remain in the mesh.

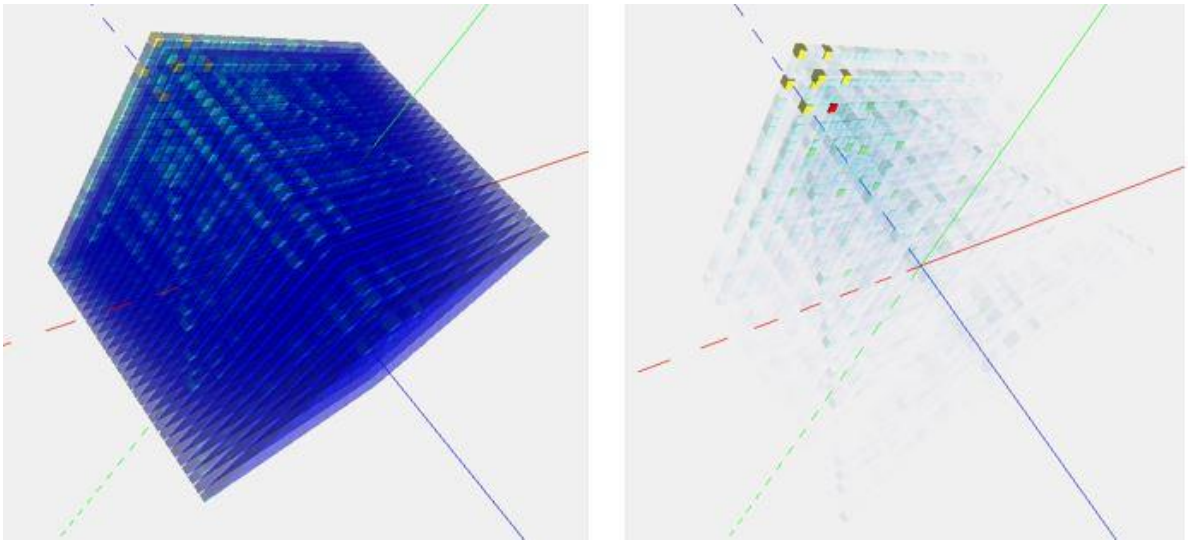

Figure 4.11: Original Opacity (left) versus Opacity by Minerals Count (right) of the 30x30x30 dataset.

4.1.6 Change Color of the Cubes by Logarithmic Values

This function will change the color of the cube mesh based on the logarithmic values of the cubes. It will normalize the values and make the cubes with a lower values more visible.

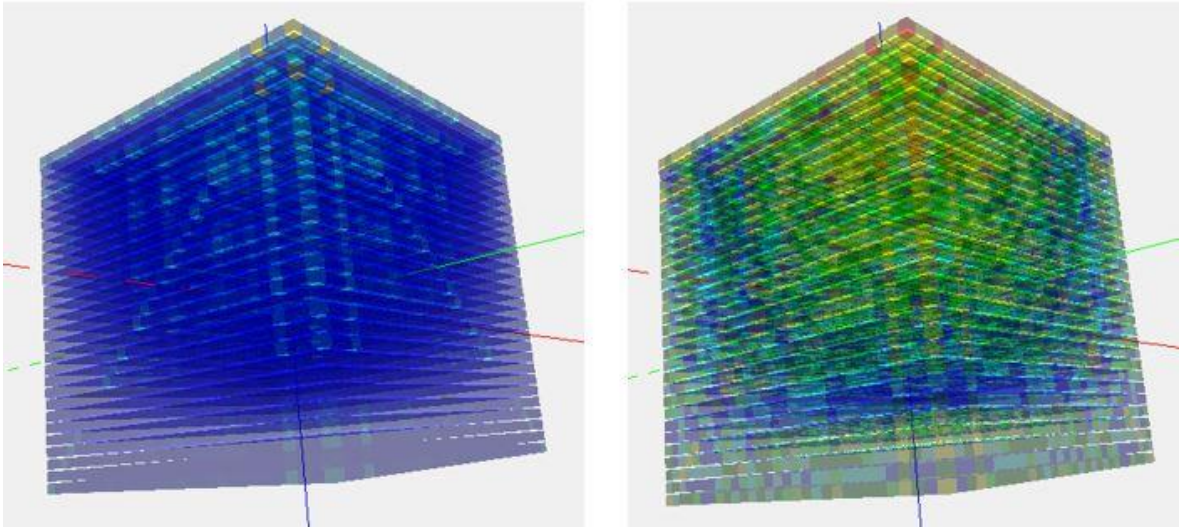

Figure 4.12: Original Cube (left) versus Log Cube (right) of the 30x30x30 dataset.

4.1.7 Change Offset of the Cubes

Here offset means the distance between the cubes in the 3D mesh. The separation between each cube can be increased or decreased in all the three axes for detailed view. There are three functions for increasing or decreasing the offset between the cubes in the 3D mesh in the x, y and z axes.

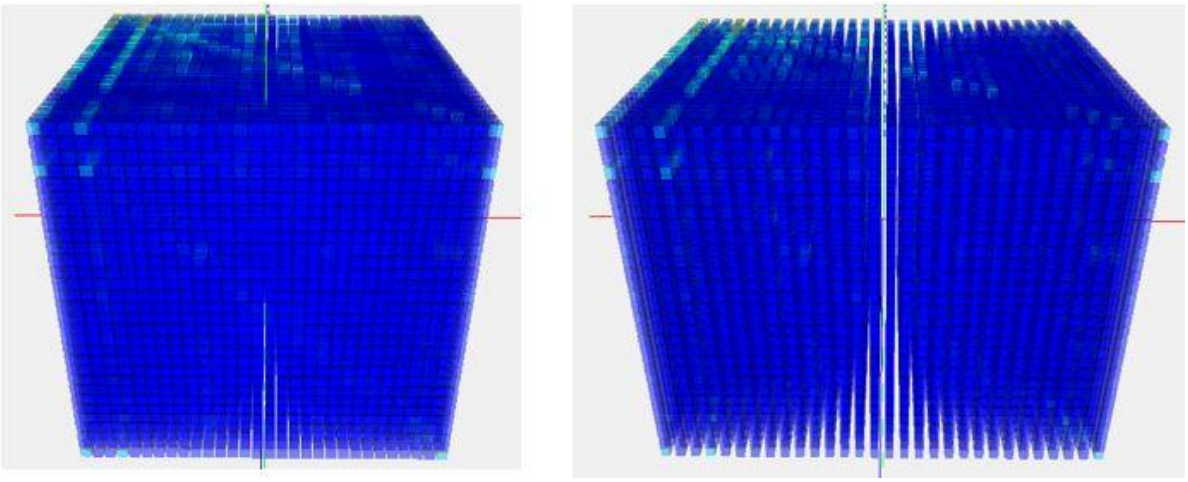

Figure 4.13: Original Offset (left) versus X Offset increased (right).

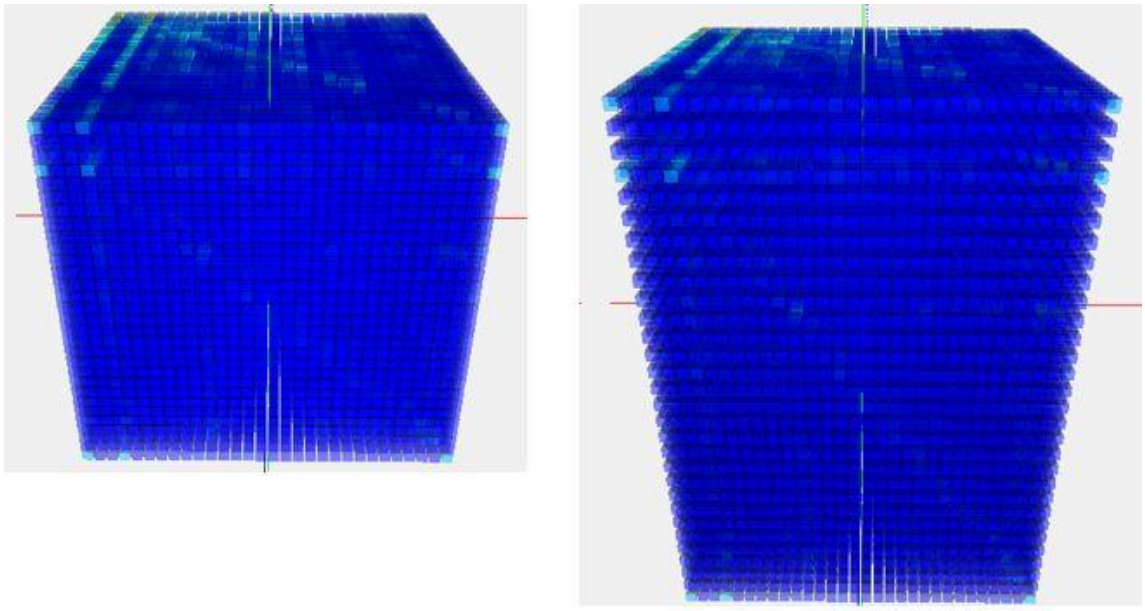

Figure 4.14: Original Offset (left) versus Y Offset increased (right).

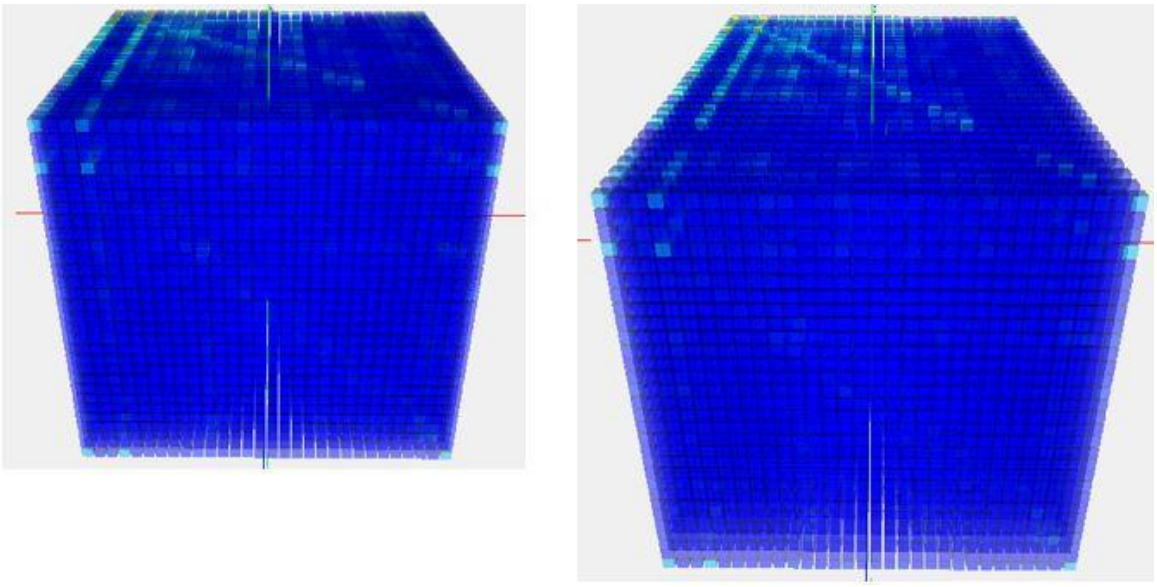

Figure 4.15: Original Offset (left) versus Z Offset increased (right).

4.1.8 Rebuilt the Cube Matrix

The 3D cube mesh after the transformations can be rebuilt into original cube mesh. This will replace the cube mesh with a new one and all the transformations applied by the user will reset so that the user can start the visualization from the initial state.

4.1.9 Remove the Cubes with values less than

If a user wants to omit all the cubes with values less than a certain number, this function will cause the 3D cube mesh to exclude the cubes whose value is smaller than the number given by the user. This is very helpful to visualize the matrix removing the outliers of analysis.

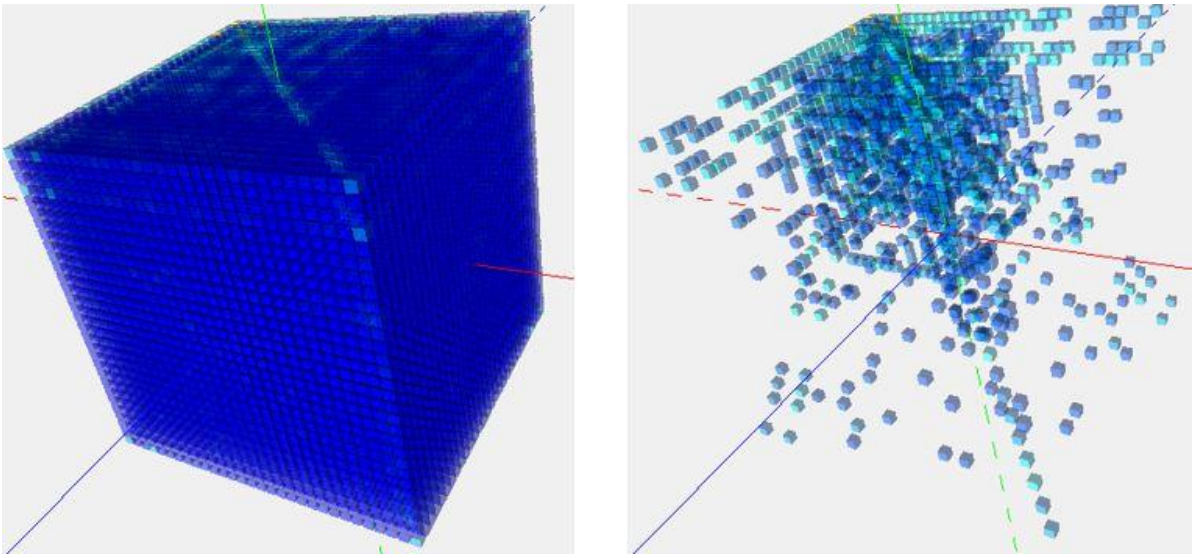

Figure 4.16: Original Cube with all the values (left) versus Cube with values smaller than 100 removed (right).

4.1.10 Extraction of a Cube Plate by Element

The 3D cube mesh, or the matrix, has z-axis overlapped so the values under the first plate covers the plate beneath it. For example, hydrogen (H) plate will cover Boron (B) plate and so on. To solve this overlapping problem, there is a plate extraction technique applied in this project. There is a list of elements at the bottom of the visualization web page. When an element on this list is clicked, the x-y plate related to that z-axis element will be extracted outside the cube. It can be placed back by re-clicking the same element in the list at the bottom. In Figure 4.17, the z-plate of Boron (B) plate is taken out of the mesh, which is beneath the Hydrogen (H) plate.

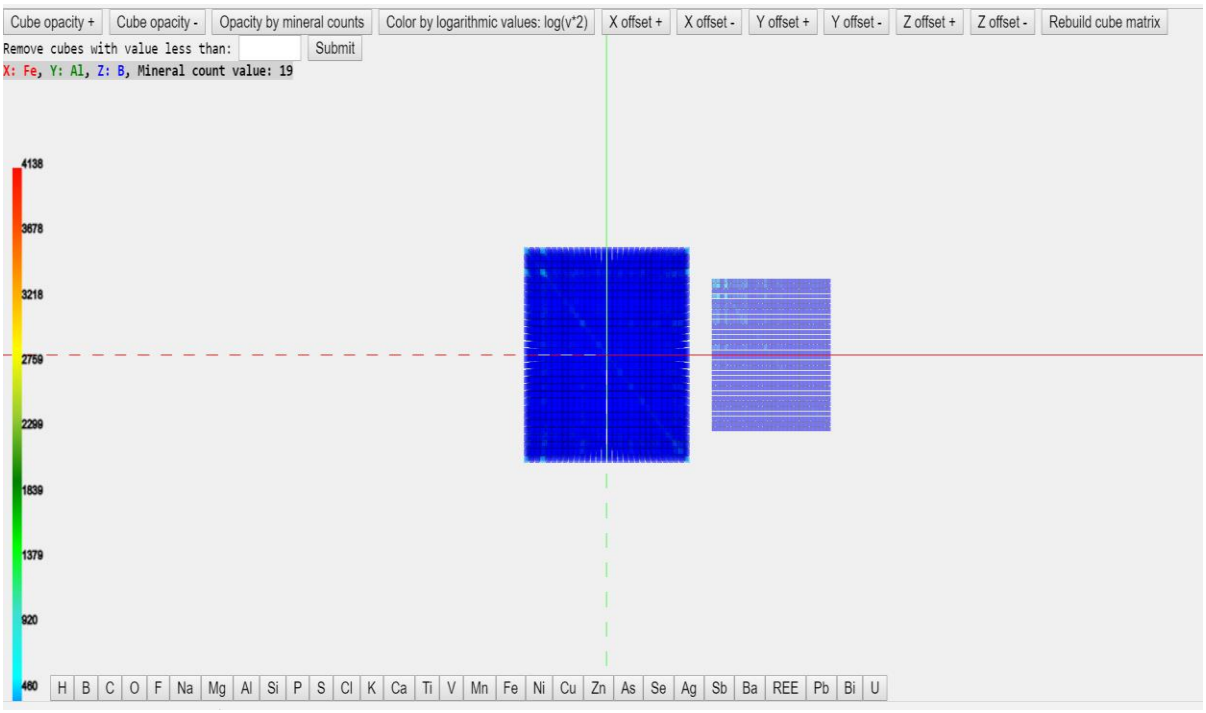

Figure 4.17: Z-plate extraction.

4.2 Co-occurrence of Research Areas in the University of Idaho

This is the second case study, which uses the dataset of researchers from the University of Idaho triplestore. Here, co-occurrence between three research topics are shown in a three dimensional cube using the same framework used in Case Study I. The logic behind calculating co-occurrence between the research topics is based on researchers working on different research areas. If a researcher is involved in two or more research areas, then those areas are said to be co-related, and hence the co-occurrence is counted as 1. The dataset is arranged in a 30x30x30 matrix and the values inside the 3D matrix are the number of researchers that are involved in all three research areas that appear in x, y and z axes.

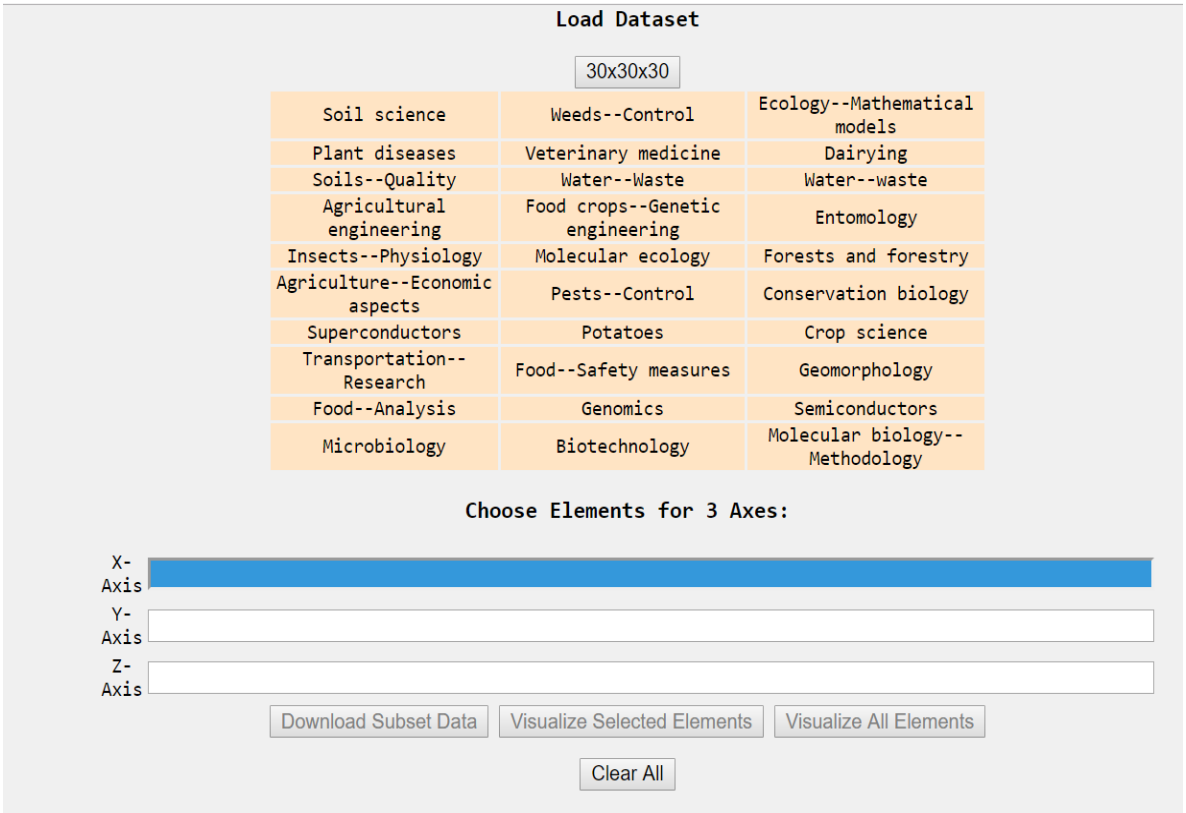

Figure 4.18: Load Dataset in Case Study II - Co-occurrence of the University of Idaho's 30 sample Research Areas.

After loading the dataset, the visualization of three dimensional cubes is similar to Case Study I, as discussed above. The user interaction and other functionalities are also similar. Figure 4.19 shows the research area selection for all three axes. After the selection of desired research areas, the user can visualize the cube of co-occurrence between these research areas. Figure 4.20 shows the visualization of selected research areas by the user and Figure 4.21 shows the visualization of all the research areas. Similar to the Case Study I, there is a feature of z-plate extraction so that the user can visualize the research area which is covered by the top z-plate.

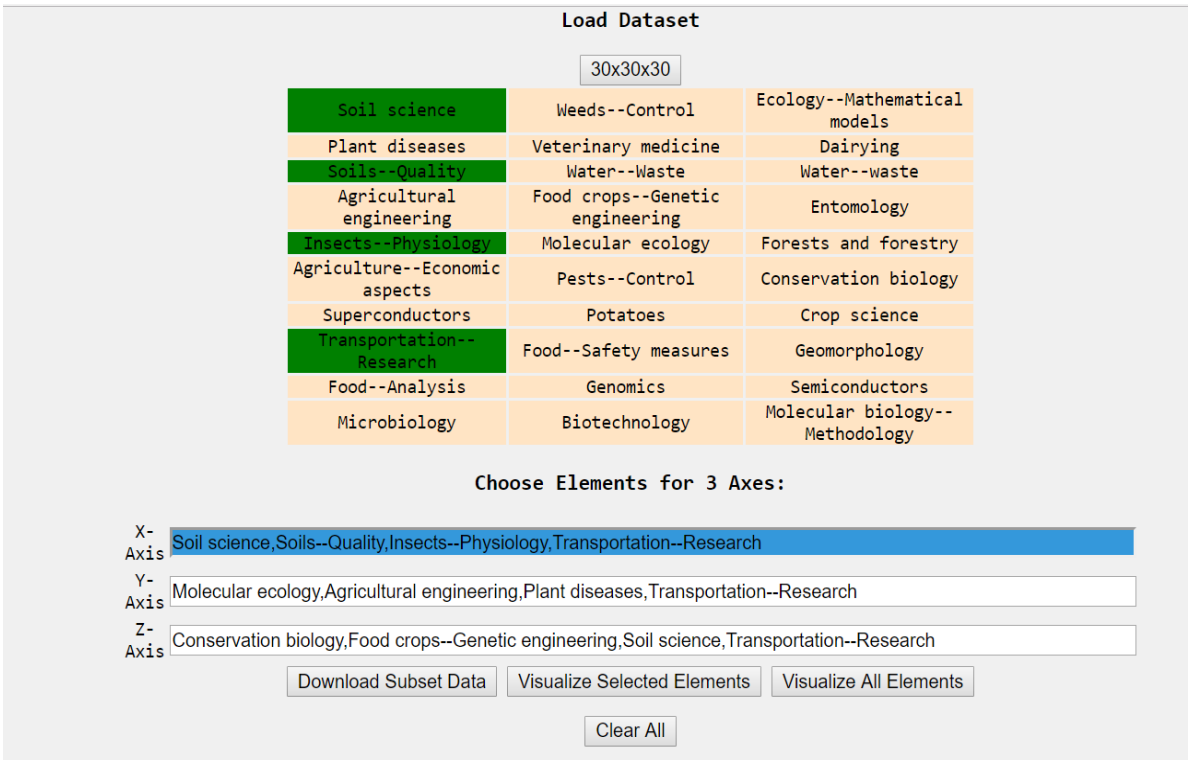

Figure 4.19: Select Research Areas in Case Study II.

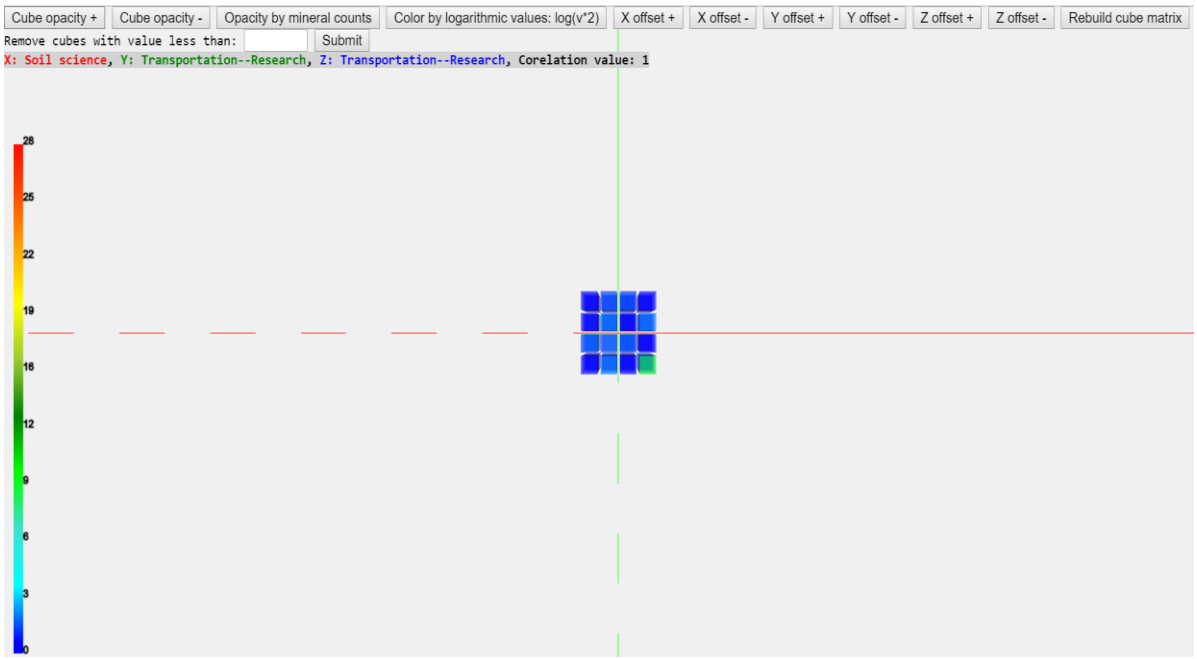

Figure 4.20: Visualize the selected Research Areas in Case Study II

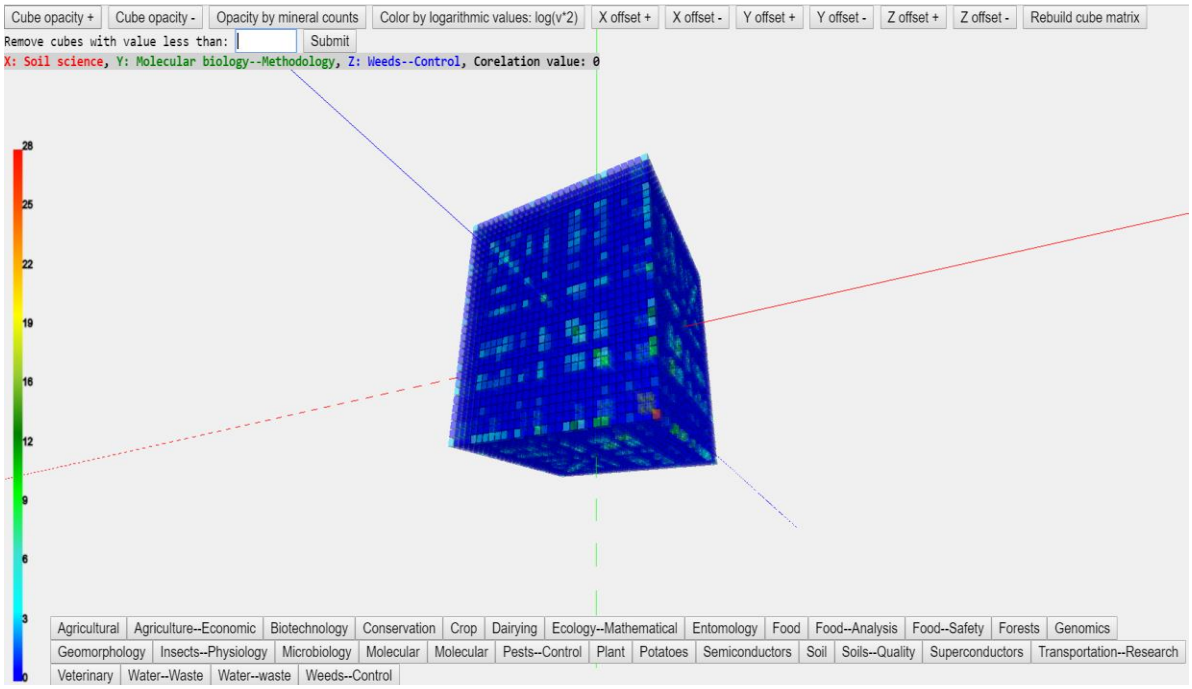

Figure 4.21: Visualize all the Research Areas in Case Study II

This chapter discussed how the user can use the application. It also described all the interactions that a user can do to visualize the data of interest. Chapter 5 summarizes the results and advantages of this thesis work and finally will conclude the thesis by pointing out several possible directions for future work.

Chapter 5: Conclusion

5.1 Achievements and Advantages

This research project aims to visualize the co-relationship between three variables which can be applied in different disciplines. We have applied this project in two areas, one in element and minerals co-relationship from DTDI and the other in U of I research areas. By applying this technique, the domain experts from different backgrounds can generate hypotheses about the dataset under study and gain preliminary understanding of the area under study which will ultimately help to proceed to next step for research (Ma, et al., 2017). In addition to visualizing the whole dataset, we have provided the researcher with the ability to interact with the dataset for example choosing subset data and visualizing the areas of interest. This will be helpful if the hardware the user is using is not powerful enough and has insufficient memory to handle the high graphic rendering. Another advantage of this work is that it is web based and is accessible from browser anywhere. Currently it is hosted in Dr. Ma's U of I web page. The URL of both of the case studies are as follows:

Case Study I: http://www2.cs.uidaho.edu/~max/PeriodicTable/PeriodicTable.html

Case Study II: http://www2.cs.uidaho.edu/~max/Correlations/list.html

5.2 Overview of Results

From a Computer Science point of view, the results of this research can be listed in the following points:

- Three dimensional heat map matrix visualization for showing co-occurrence between three elements forming mineral species.
- Three dimensional heat map matrix visualization for showing co- occurrence between three 30 sample research areas of U of I.
- Selection of data of interest for quick visualization.
- Provide user with the functionalities to interact with three dimensional cube matrix.
- Visualize only the selected subject areas of interest.
- Visualization of whole dataset.

Using the prototype from Case Study I, the Geoscientists have discovered several results and generated some hypotheses based on the results, which are listed below:

- Figure 5.1 shows the Oxygen plate sliced out from the three dimensional cube. The solid red filled cell has Oxygen in all three axes, which has the highest mineral count, 4138, in the whole matrix (Ma et al., 2017).

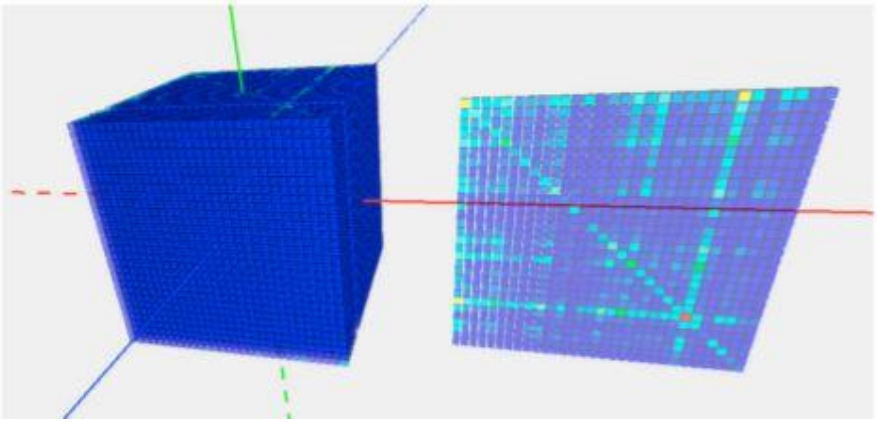

Figure 5.1: Oxygen plate sliced out (Ma et al., 2017).

- Using 30x30x30 normalized dataset visualization, they found that 29.8% of minerals which contains Oxygen (O) also contains Calcium (Ca) (Ma et al., 2017).

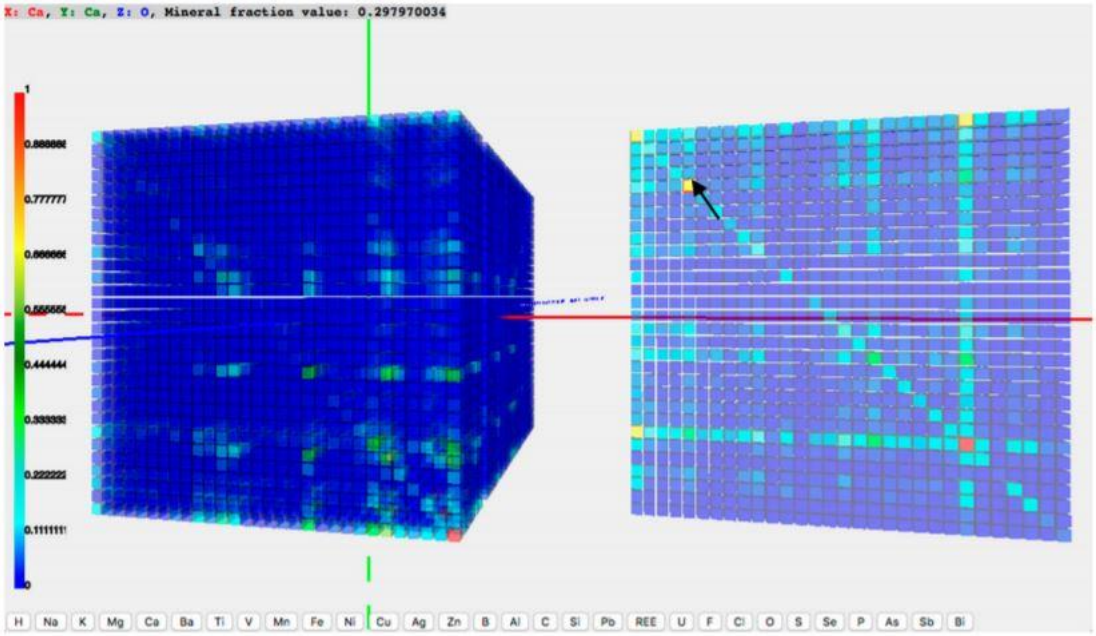

Figure 5.2: Mineral fraction value '0.297970034' means that about 29% of minerals containing Oxygen also contain Calcium (Ma et al., 2017).

- Using 72x72x72 dataset, which is a chi-square test value of the minerals count, they tried to answer the question ‘Does the presence of element Z affect the correlation between elements X and Y in mineral species and up to what extent?’ (Ma et al., 2017). For example, in Figure 5.3, the red and blue colored rows represent the O-H plane. The explanation of an example of this question is copied directly from the original paper.

  > Cells that are colored red represent elements that correlate strongly to O–H bearing minerals, and cells colored blue represent elements that are anti-correlated to O–H bearing minerals. These results indicate that some elements are very common in hydrated mineral species, while others are rarely found in hydrated minerals. This is an entirely new result gained from this use case, and leads geoscientists to new questions regarding what causes an element to associate with hydrated minerals. (Ma et al., 2017)

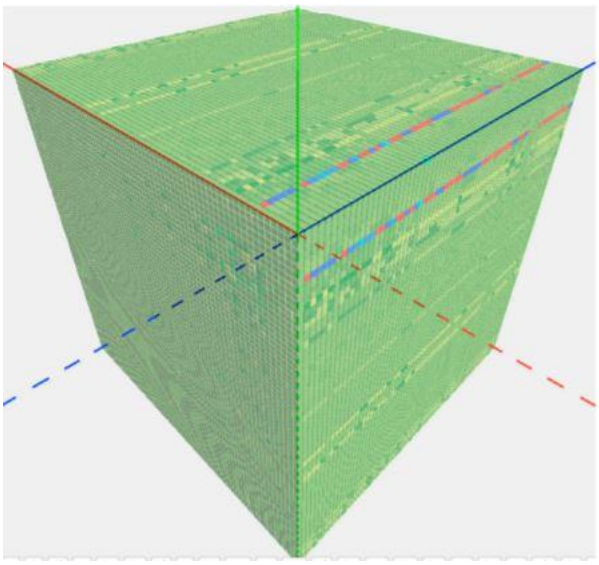

Figure 5.3: 72x72x72 dataset visualization (Ma et al., 2017).

These examples show the usefulness of this application across other disciplines.

5.3 Limitations and Recommendations

As discussed in chapter 1.6, this thesis is focused on visualization purpose only and our collaborators will explain the domain related interpretation. This thesis does not describes the efficiency of any algorithm or tools used, but show the importance of meaningful visualization

to domain scientists and show the usefulness of EDA in a data science process. This project needs HTML5 web browser for three dimensional graphic rendering. This project works in most of the modern browsers like Chrome, Firefox, Safari and Edge. For visualizing the whole dataset, the application response time is high, so users are recommended to visualize only subset data on interest for research purposes. The following table depicts the response time and memory consumption for rendering and loading the full three dimensional cube in a Windows 10 machine with I7 Intel processor and 16 Giga Byte Memory tested in Mozilla Firefox browser:

Table 5.1: Response Time and Memory Consumption for Different Datasets

Dataset	**Loading Time**	**Memory used**
30x30x30	4s	650 MB
30x30x30 Normalized	4s	650 MB
72x72x72	36s	7000 MB

5.4 Future Work

This section discusses the future advancement or improvement that can be done for this project. In addition, out some approaches are mentioned that could improve the performance of the application.

This thesis work is developed using JavaScript libraries. To implement this across different disciplines, we need to change code to some extent for displaying the items and choosing the dataset. For the two case studies implemented in this thesis work, although most of the code for the visualization part is reused, separate code repositories are created to accommodate different types of dataset from two disciplines. To make this application independent of the dataset and to make this thesis work easily implementable for wide variety of disciplines, the code of this project can be developed into a package in R programming similar to other packages that are mentioned previously, like d3heatmap package in R for developing two dimensional heat maps.

The current approach of a loading the dataset imports the full dataset into a JavaScript object. Since JavaScript objects are stored in memory on the client side, the application consumes huge memory and processing time to respond for the whole dataset visualization. Although the feature of partial or subset selection of data is provided in the application, it is

done only after loading the whole dataset into memory. So, another improvement of the project can be finding an approach to load only the dataset of interest from the server side instead of importing the entire dataset into the client side and selecting the data of interest on client side. This will eliminate the high memory requirement in the client machine.

The dataset used in this thesis has many zero values, but they are treated as other non-zero values, and occupies memory. So, the application could be made efficient and faster by using the concept of sparse matrix (Ziganto, 2018).

This project can be extended further by adding different features to three dimensional visualization in a three dimensional environment instead of just being visualized in a flat or two dimensional computer screen. Currently, this research provides the visualization of a three dimensional cube on a two dimensional screen, and users can interact with the cube using their mouse and keyboards. An example of such advanced three dimensional visualization is "Computer Assisted Virtual Environment" (CAVE). CAVE is an ongoing research project in the Applied Visualization Laboratory at the Center of Advanced Energy Studies (CAES) (CAES, 2018). It uses a rear projection technique in three walls and the floor, and researchers can manipulate, control and analyze data using a "wand" and walk through their data (CAES, 2018).

References

Alasql, (2018), Alasql – Javascript SQL Library. Retrieved on Mar 1, 2018 from http://alasql.org/.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. In The semantic web (pp. 722-735). Springer, Berlin, Heidelberg.

Belmonte, N.G. (2016, Nov 10). Visualize Data Sets on the Web with Uber Engineering's deck.gl Framework. Uber Engineering. Retrieved on Mar 2, 2018 from https://eng.uber.com/deck-gl-framework/.

CAES. (2018). Center for Advanced Energy Studies. Retrieved on Mar 3, 2018 from https://caesenergy.org/research/laboratories/avl/.

Centers for Disease Control and Prevention (CDC). Behavioral Risk Factor Surveillance System. Atlanta, Georgia: U.S. Department of Health and Human Services, Centers for Disease Control and Prevention, (2018). Retrieved on Apr 11, 2018 from https://data.cdc.gov/Smoking-Tobacco-Use/BRFSS-Prevalence-and-Trends-Data-Tobacco-Use-Four-/8zak-ewtm/data.

Cheng, J., & Galili, T. (2015). d3heatmap: Interactive Heat Maps Using 'htmlwidgets' and 'D3. js'. R package version 0.6, 1.

Cohen, I. (2014, May 6). TrackballControls. Retrieved on Mar 2, 2018 from https://developer-archive.leapmotion.com/gallery/trackballcontrols.

Cohen, L., Lehéricy, S., Chochon, F., Lemer, C., Rivaud, S., & Dehaene, S. (2002). Language-specific tuning of visual cortex? Functional properties of the Visual Word Form Area. Brain, 125(5), 1054-1069.

Cox, V. (2017). Exploratory data analysis. In Translating Statistics to Make Decisions (pp. 47-74). Apress, Berkeley, CA.

Devare, M., Corson-Rikert, J., Caruso, B., Lowe, B., Chiang, K., & McCue, J. (2007). Connecting people, creating a virtual life sciences community. D-Lib Magazine, 13(7/8), 1082-9873.

Developers, N. (2012). Neo4j. Graph NoSQL Database. Retrieved on March 3 from https://neo4j.com.

Dhar, V. (2013). Data science and prediction. Communications of the ACM, 56(12), 64-73.

Dirksen, J. (2013). Learning Three. js: the JavaScript 3D library for WebGL. Packt Publishing Ltd., Birmingham, UK.

Fox, P., & Hendler, J. (2011). Changing the equation on scientific data visualization. Science, 331(6018), 705-708.

Friedman, V. (2008). Data visualization and infographics. Graphics, Monday Inspiration, 14, 2008. Retrieved on Mar 26, 2018 from https://www.smashingmagazine.com/2008/01/monday-inspiration-data-visualization-and-infographics/.

Friendly, M., & Denis, D. J. (2001). Milestones in the history of thematic cartography, statistical graphics, and data visualization. Retrieved on Mar 2, 2018 from http://www. datavis. ca/milestones, 32.

Gershun, A. (2018). Alasql.js. GitHub. Retrieved on Mar 1, 2018 from https://github.com/agershun/alasql.

Hayashi, C., Yajima, K., Bock, H. H., Ohsumi, N., Tanaka, Y., & Baba, Y. (Eds.). (2013). Data Science, Classification, and Related Methods: Proceedings of the Fifth Conference of the International Federation of Classification Societies (IFCS-96), Kobe, Japan, March 27–30, 1996. Springer Science & Business Media.

Holt, A. (2018). Papaparse.js. GitHub. Retrieved on Mar 1, 2018 from https://github.com/mholt/PapaParse.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95.

Klyne, G., & Carroll, J. J. (2006). Resource description framework (RDF): Concepts and abstract syntax. Retrieved on Mar 26, 2018 from https://www.w3.org/TR/rdf-concepts/.

Kunst, J. et al., (2017). Package 'highcharter'. Retrieved on Mar 3 from https://cran.r-project.org/web/packages/highcharter/highcharter.pdf.

Li, J., Meng, Z. P., Huang, M. L., & Zhang, K. (2017). An interactive visualization approach to the overview of geoscience data. Journal of Visualization, 20(3), 433-451.

Ma, X., Chen, Y., Wang, H., Zheng J.G., Fu, L., West, P., Erickson, J.S., Fox, P., 2015. Data visualization in the Semantic Web. In: Narock, T., Fox, P., (eds.) The Semantic Web in Earth and Space Science: Current Status and Future Directions. IOS Press, Berlin. pp. 149-167. http://dx.doi.org/10.3233/978-1-61499-501-2-149.

Ma, X., Hummer, D., Golden, J. J., Fox, P. A., Hazen, R. M., Morrison, S. M., Downs, R.T., Madhikarmi, B.L., Wang, C. & Meyer, M. B. (2017). Using Visual Exploratory Data Analysis to Facilitate Collaboration and Hypothesis Generation in Cross-Disciplinary Research. ISPRS International Journal of Geo-Information, 6(11), 368.

Mr.doob. (2018). Three.js, GitHub. Retrieved on Mar 1, 2018 from https://github.com/mrdoob/three.js/.

Murray-Rust, P. (2008). Open data in science. Serials Review, 34(1), 52-64.

Negash, S. (2004). Business intelligence. The communications of the Association for Information Systems, 13(1), 54.

PapaParse, (2018), Documentation. Retrieved on Mar 1, 2018 from https://papaparse.com/docs#config.

Prud, E., & Seaborne, A. (2006). SPARQL query language for RDF. Retrieved on Mar 26, 2018 from https://www.w3.org/TR/rdf-sparql-query/.

Ramachandran, P., & Varoquaux, G. (2011). Mayavi: 3D visualization of scientific data. Computing in Science & Engineering, 13(2), 40-51.

The jQuery Foundation (2018). jQuery write less, do more. Retrieved on Mar 2, 2018 from https://jquery.com/.

Schutt, R. & O'Neil, C. (2013) Doing Data Science: Straight Talk from the Frontline; O'Reilly: New York, NY, USA.

Segaran, T., Evans, C., & Taylor, J. (2009). Programming the Semantic Web: Build Flexible Applications with Graph Data. O'Reilly: New York, NY, USA.

Shannon, P., Markiel, A., Ozier, O., Baliga, N. S., Wang, J. T., Ramage, D., Amin, N., Schwikowski, B. & Ideker, T. (2003). Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome research, 13(11), 2498-2504.

Stapley, B. J., & Benoit, G. (1999). Biobibliometrics: information retrieval and visualization from co-occurrences of gene names in Medline abstracts. In Biocomputing 2000 (pp. 529-540).

Team, R. (2015). RStudio: integrated development for R. RStudio, Inc., Boston, MA. Retrieved on Mar 2, 2018 from http://www.rstudio.com.

Three.js. (2010). Manual. Retrieved on Mar 1, 2018 from https://threejs.org/docs/index.html#manual/introduction/Creating-a-scene.

van Hage, W. R., & Kauppinen, (2011). T. SPARQL package for R. Retrieved on Mar 3 from https://cran.r-project.org/web/packages/SPARQL/SPARQL.pdf.

University of Idaho VIVO. (2018). Research and Expertise. Retrieved on Mar 2, 2018 from https://vivo.nkn.uidaho.edu/vivo/.

VIVO. (2018). About VIVO. Retrieved on Mar 2, 2018 from http://vivoweb.org/info/about-vivo.

W3Schools. (2018a). jQuery Tutorial. Retrieved on Mar 2, 2018 from https://www.w3schools.com/jquery/jquery_intro.asp.

W3Schools. (2018b). XML RDF. Retrieved on Mar 2, 2018 from https://www.w3schools.com/xml/xml_rdf.asp.

Wilkinson, L., & Friendly, M. (2009). The history of the cluster heat map. The American Statistician, 63(2), 179-184.

Ziganto, D. (2018). Sparse Matrices for Efficient Machine Learning. Retrieved on Apr 4, 2018 from https://dziganto.github.io/Sparse-Matrices-For-Efficient-Machine-Learning/.

Appendix A. Code Repository

The code for both of the case studies can be found and downloaded from GitHub repository for free. The links to the project are given below:

Case Study I: https://github.com/xgmachina/3dcube

Case Study II: https://github.com/bhumadhi/CoRelations-for-Researchers

Appendix B. Work flow in the code of the project

The different function used in coding for different purposes is summarized below:

1. Load Dataset

   $('#btnLoadThirty').click(function (){…})

   $('#btnLoadThirtyNor').click(function (){…})

   $('#btnLoadSeventyTwoCS').click(function (){…})

   $('#btnLoadSeventyTwo').click(function (){…})

   These functions will load the desired CSV file and save in the variable “results”.

2. Choosing axes (x, y, z)

   $("input[name='axis']").change(function(){….})

   This function is used to change between x, y or z axes, so that the user can choose elements for the respective axes.

   This function will also highlight the periodic table elements with green if the user has already selected the elements before.

3. Clear All

   $('#btnVisualizeAll').click(function (){….})

   This button will clear the selected elements from periodic table. In addition the elements displayed in the text boxes for user selection are also deleted.

4. Download Subset Data

   downloadCSV()

   This function will download the subset data for the elements chosen by the user in csv format.

5. Visualize Selected Elements

   $('#btnVisualizeSel').click(function (){….})

   This button will take the user to next tab for visualization of 3D cube based on the elements based on the user selection.

6. Visualize All Elements

   $('#btnVisualizeAll').click(function (){….})

   This button will take the user to next tab where all the elements present in the data uploaded by the user. This button works even if the user does not select elements in the three axes.

Other main notable functions for the visualization project are:

1. getHeaders(results)

   This function extracts the headers from the dataset uploaded.

2. highlightHeaders()

   This function highlights the elements in the periodic table which are present in the dataset uploaded.

3. selectData()

   This function highlights the elements in the periodic table green which are clicked by the user.

4. createAxes(cell)

   This function catches the selected elements in the periodic table and display in the text boxes below the periodic table for the user.

5. checkAlreadySelected(element,axis)

   This function checks whether the element is already selected. If the user clicks the element for the second time, it will be considered as unselect and the element will be unselected.

6. createSubset()

   This function will create subset of data uploaded by the user. It will perform a SQL like query on the dataset based on the elements selected for different axes.

7. createCSV(str)

   This function will create CSV format data based on subset data created by createSubset() function.

Below are the major functions for actual 3D visualization:

1. init()

   This is initialization function where all the parameters needed for three.js such as the camera, scene, light and renderer.

2. createCubeMatrix()

   This function creates box geometry of size 1 x 1 x 1 for each elements in 3D cube. The small boxes are created based on the number of elements in the dataset and the color are based on the value of each elements combination in dataset for 3 axes in the dataset.

3. createColorLegend()

   This function defines gradient and stroke of the legend at the left side which ranges from minimum to maximum possible values.

4. buildAxes()

   This function calls buildAxis() function which builds axes for x, y and z of 1000 length which range from -1000 to +1000. For positive coordinate the axis line is solid whereas for negative coordinate the axis line is dashed.

5. rebuildAllCubes()

   This function recreates the cube matrix removing the existing cube.

6. removeAllCubes()

   This function removes the existing scene from the canvas.

7. increaseOpacity()

   This function increases opacity of the visualized cube by 0.1.

8. decreaseOpacity()

   This function decreases opacity of the visualized cube by 0.1.

9. opacityByMineralCounts()

This function changes the opacity of the cube by mineral count.

10. colorByLogarithmicValues()

    This function changes the color of the cube taking the log value of the values of the cube.

11. increaseXoffset(), decreaseXoffset(), increaseYoffset(), decreaseYoffset(), increaseZoffset(), decreaseZoffset()

    These functions are used to increase the distances between the boxes in the cube.

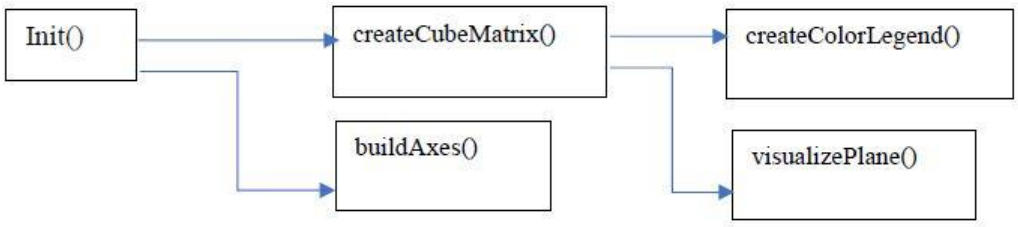

Figure B.1: Primary functions calls for cube visualization part.

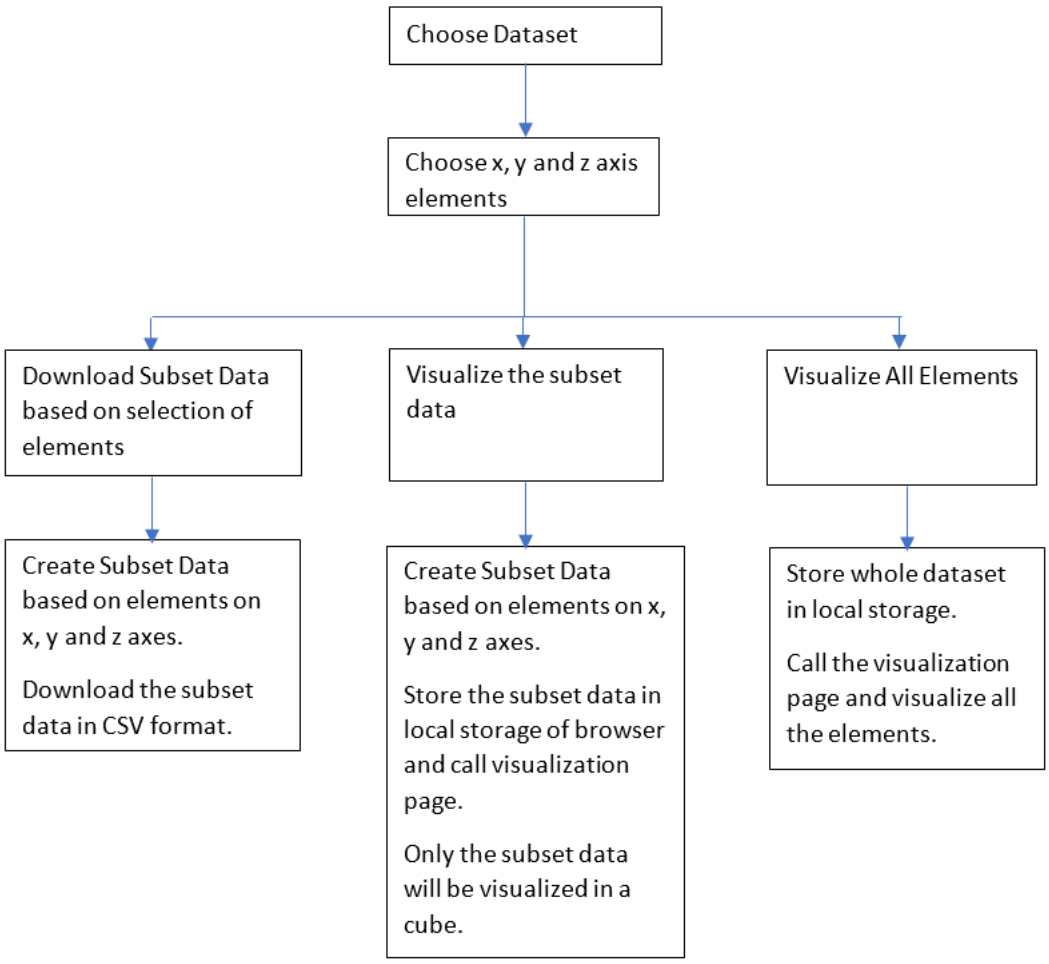

Figure B.2: User Work Flow Process of Dataset Selection and Visualization.

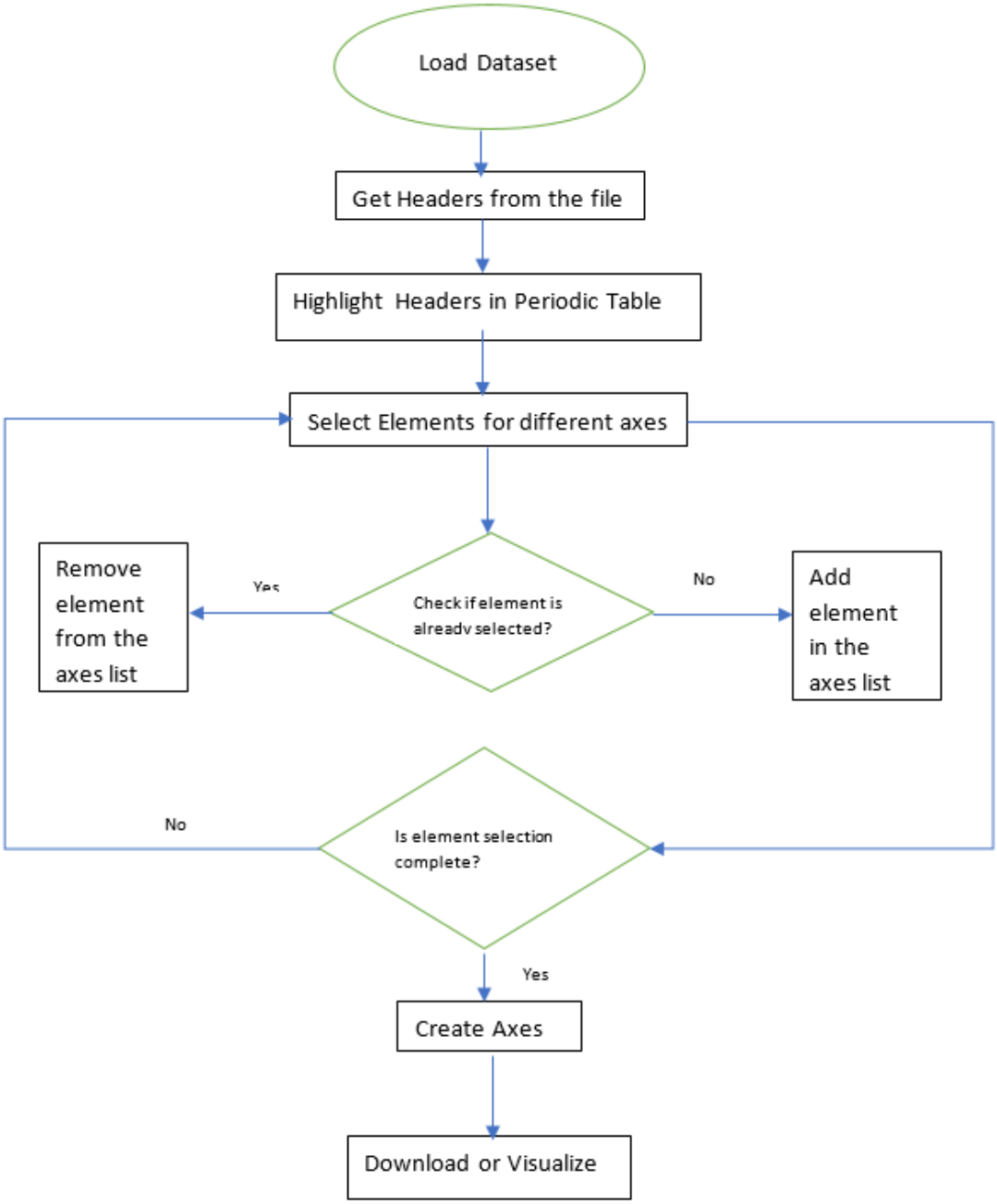

Figure B.3: Flow Chart of Data Selection for Developers.