1-12-2023

Visual Analytics and Modeling of Materials Property Data

Diwas Bhattarai
Louisiana State University and Agricultural and Mechanical College

VISUAL ANALYTICS AND MODELING OF MATERIALS PROPERTY DATA

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

Division of Computer Science and Engineering

by
Diwas Bhattarai
B.S., Computer Science, Southeastern Louisiana University, 2013
May 2023

Acknowledgments

First, and most of all, I would like to thank Dr. Bijaya Karki for his expertise, vision, and motivation while guiding me through this research process. Without his help and suggestions, this dissertation and my academic and intellectual growth would not have been possible. Second, I would also like to thank my committee members Dr. Jian Zhang, Dr. Gerald Baumgartner, and Dr. Hongchao Zhang, for their time and guidance. Special thanks to Dr. Bijaya Karki, Dr. Hui Zhang, Dr. Tatsuya Sakamaki, and many more authors for making the materials data available.

I also extend heartfelt gratitude to my family and friends for their enormous support throughout these years. My parent's guidance, patience, and understanding have been a constant source of motivation. I am deeply thankful to my wife Ruzova, for being there through thick and thin and providing emotional and practical support. Her constant belief in me helped me overcome obstacles I faced over time. I am grateful for my loving family and my brother. Thank you from the bottom of my heart.

Table of Contents

List of Tables

List of Figures

Abstract

Due to significant advancements in experimental and computational techniques, materials data are abundant. To facilitate data-driven research, it calls for a system for managing and sharing data and supporting a set of tools for effective data analysis and modeling. Generally, a given material property M can be considered as a multivariate data problem. The dimensions of M are the values of the property itself, the conditions (pressure P, temperature T, and multi-component composition X) that control the concerned property, and relevant metadata I (source, date). Here we present a comprehensive database considering both experimental and computational sources and an innovative visual analytics system for melt viscosity (η), which can be represented by $M(\eta, P, T, X1, X2, \dots, I1, I2, \dots)$. We implemented the parallel coordinates plot (PCP) method by introducing new non-standard features, such as derived axes/sub-axes, dimension merging, binary scaling, and nested plots. Thus enhanced PCP offers many insights of relevance to underlying physics, data modeling, and guiding future experiments/computations. The construction of viscosity models is a non-trivial process, and extant models are often limited to a sub-parameter space, such as the ambient pressure conditions. To develop a generalized model which applies to wider parameter space, we trained various machine learning models, including neural network, Decision Tree, Random Forest, and XGBoost. We evaluated model performance based on loss function, error distribution, and model continuity. Our results show that neural network models outperformed the physics-based models as well as all tree-based models. A small neural network with two hidden layers, each containing 64 nodes, was found to be sufficient to model both the ambient pressure and complete dataset. Despite a marginal decrease in RMSE, a larger

neural network consisting of four hidden layers with 128 nodes in each layer could provide an even better fit for the complete dataset in terms of model continuity and error distribution. Tree-based models could follow the training data, but the model results show high variations with small changes in parameter space, making them less applicable for continuous numerical data. Our data visualization and modeling approach is expected to be useful to researchers who explore and model material data, for instance, the density property can be incorporated as a new attribute in our system.

Chapter 1. Introduction

Numerous experiments and simulations are producing substantial data on materials properties of interest to many fields of science and engineering. Advanced techniques to deal with massive amounts of material data have been gaining interest in recent years and are employed for tasks ranging from general data exploration to trend analysis. Uncertainty assessment and change detection also happen to be some areas of interest. Physical properties data such as viscosity, density, elasticity, and conductivity are highly sought after for trend analysis as they played a crucial role in the chemical and thermal evolution of the Earth [47, 11]. In this chapter[1], we briefly introduce materials property data and our workflow for visualization and modeling.

Researchers usually have to go through data collection, pre-processing, exploration, and finally modeling phases to better understand a given physical property. Data collection in itself is non-trivial since they are usually spread out across numerous publications each containing only a handful of data points. Dealing with a collection of datasets from various sources also requires careful data pre-processing. Further, the multivariate nature of these data introduces more challenges. They involve different variable types which require their own respective representations. For instance, the value of a material property under consideration itself may be a scalar, vector, or tensor quantity. The parameter space in which the property is defined includes variables such as pressure, temperature, and composition. Other information such as methodology (experiment or computation), the publication (year, authors, source), model-predicted values, uncertainties, errors, etc.

[1]Some parts of this chapter can be found in D. Bhattarai., J. Zhang., and B. B. Karki. Parallel coordinates-based visual analytics for materials property. In Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - IVAPP, pages 83–95. INSTICC, SciTePress, 2019. Copyright permission included in Appendix A.

can be useful in the analysis. The actual data values along with metadata can be examined for completeness, trends, correlations, and modeling. An important class of materials belongs to magma-forming silicate melts [36], and here we take the melt viscosity as a use case for materials property.

Silicate melts are the most common components of Earth's igneous processes. They are usually found in the mantle but sometimes surface through volcanic eruptions. It is likely that the Earth contained a global magma ocean in the past and the reason for such an extreme situation may be related to the moon forming giant impacts. The Earth has since gone through many geological changes but the information regarding this early period can still be studied by investigating silicate melt properties such as viscosity, density, conductivity, and others. In particular, viscosity is perhaps the most important property governing all magmatic processes including melt transport, magma mixing, and volcanic eruptions [1, 59, 73]. Therefore, viscosity data are highly sought after in a wide array of fields. The abundant nature of silicate melts provides some ease in experimental data collection; however, they are confined to a narrow temperature and pressure range. For more extreme temperatures and pressures, the latest advancement in simulations has been successfully producing more and more data points. These computational data cover ranges that are not feasible for experimental measurement. Both experimental and calculated data have been used to model viscosity, but relatively few attempts have been made which cover large temperature and pressure ranges. Trend analysis has been performed on data published over many years. But before data mining for trends, it is often beneficial to explore and understand the data.

1.1. Database

Performing analytics on a dataset starts with data collection and data pre-processing. Materials data have been collected from several published sources. The pre-processing step is then needed to clean and transform all data into a common format. This step is crucial as different authors may publish data in different ways. For example, authors may choose direct values (η) or different logarithmic bases (ln or $\log_{10}$) for viscosity (Pa s) measurements. The temperature may be represented in Kelvin (K) or Celsius (°C) and pressure in Pascal (Pa) or giga-Pascal (GPa). Composition components such as SiO_2 may be given in molar fraction or weight percentage. This collection of cleaned and transformed datasets can then be saved in a database for easy data accessibility, scalability, and maintainability. Relational database systems (RDBMS) are often the most suitable choice for data storage. Data are stored in tabular form with certain entities representing relationships to other tabular forms. A complete query of a data selection may require joining multiple tables. Non-relational databases are chosen for large data systems where data redundancy is used in favor of scalability and schema-less flexibility.

1.2. Data Visualization

Visualization is a way of representing data graphically or pictorially in order to reveal hidden patterns and aid in decision-making. For example, Table 1.1 shows 50 viscosity data points consisting of pure MgO-SiO_2 binary at various temperatures and zero pressure. Scrutinizing data tables for patterns can be quite difficult. With proper visualization, a better representation of the data may emerge. In Figure 1.1, columns of the different composition ratios of MgO-SiO_2 have been given different colors. The x-axis represents

Table 1.1. Pure MgO-SiO_2 binary at various temperatures and zero pressure. Blank spaces denote the absence of data at specific conditions.

		X_MgO = 1 X_SiO_2 = 0 MgO pe	5/6 1/6 Mg_5SiO_7 pe1	2/3 1/3 Mg_2SiO_4 fo	1/2 1/2 $MgSiO_3$ en	1/3 2/3 $MgSi_2O_5$ sil1	1/6 5/6 $MgSi_5O_{11}$ sil2	0 1 SiO_2 sil
	Atoms	64	104	112	80	160	85	72
T (K)	10000/T							
2000	5.000			0.0420	0.0950			
2500	4.000		0.0055	0.0120	0.0240	0.12	0.6	
2750	3.636							21
3000	3.333	0.0035	0.0035	0.0047	0.0065	0.0140	0.0570	5.5
3500	2.857	0.00178	0.0026	0.0032	0.0047	0.0052	0.0099	0.22
4000	2.500	0.00148	0.00140	0.0023	0.00230	0.00190	0.00260	0.01400
5000	2.000	0.00078	0.00081	0.00085	0.00086	0.00054	0.00085	0.00070
6000	1.667	0.00047	0.00038	0.00050	0.00054	0.00026	0.00022	0.00015
8000	1.250	0.00019	0.00020	0.00016	0.00018	0.00012	0.0001	0.00004

the inverse of temperature (K^{-1}) and the y-axis represents viscosity (Pa s) in the logarithmic scale. The inverse relationship between viscosity and temperature in a temperature-flipped y-axis graph shows a positive relationship for each composition. Furthermore, each composition can be seen behaving differently. Similar to the insights discussed, a visualization system should be helpful to present patterns not easily visible by merely looking at the numbers. Visualization applications tend to operate with one or more datasets. They may further provide detailed plots with interaction capabilities. There are many ways a visualization application can be used to explore data therefore, significant planning is required ahead of implementation to assess user requirements completely. Therefore, a practical visualization application must fulfill at least the following three criteria [37]:

- Must be based on non-visual data source
- Must produce an image
- The result is readable and recognizable

These criteria were designed with focus on information visualization, but they work equally well for scientific visualization. Information visualization graphically repre-

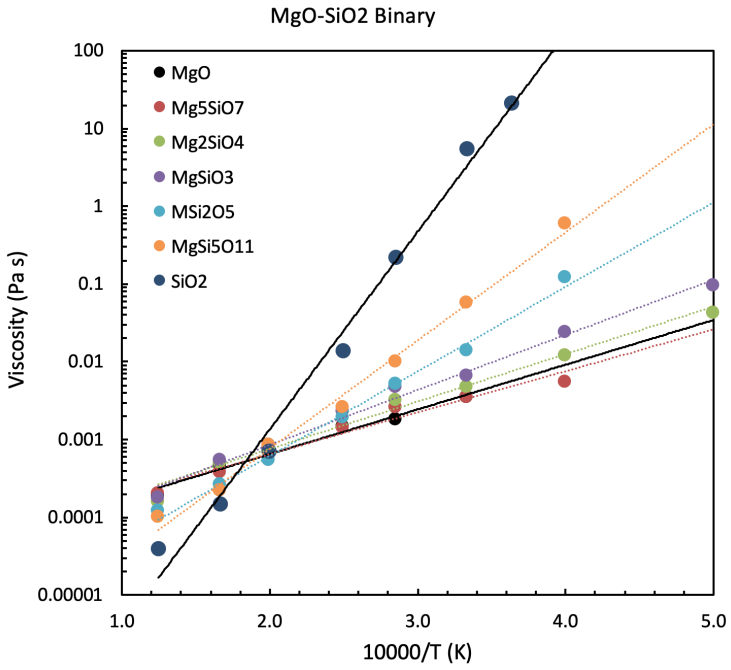


Figure 1.1. Plot for pure MgO-SiO_2 binary at various temperatures and zero pressure.

sents abstract data with no inherent spatial structure, for instance, visualizing viscosity changes with respect to temperature or displaying election results. In Figure 1.2 water surface height is shown for the timeline when tropical storm Barry (2019) hit Lafayette, Louisiana. Barry started in the Gulf of Mexico and, with an average speed of around 5 miles per hour, made landfall on Marsh Island and Intracoastal City, Louisiana. It was a Category 1 hurricane that weakened to a tropical storm after landfall. Hurricane Barry traveled North passing nearby Lafayette, Louisiana. Flooding is a major concern in Louisiana and hence several pump stations have been set up to control water flow around the state. Some of these pumps are set up in Lafayette, Louisiana with IoT technologies that continuously monitor data such as temperature, pressure, pump station run time status, and water surface level of the reservoir. The reservoir at this particular station fills up gradually until a certain water level threshold is met, after which, a collection of

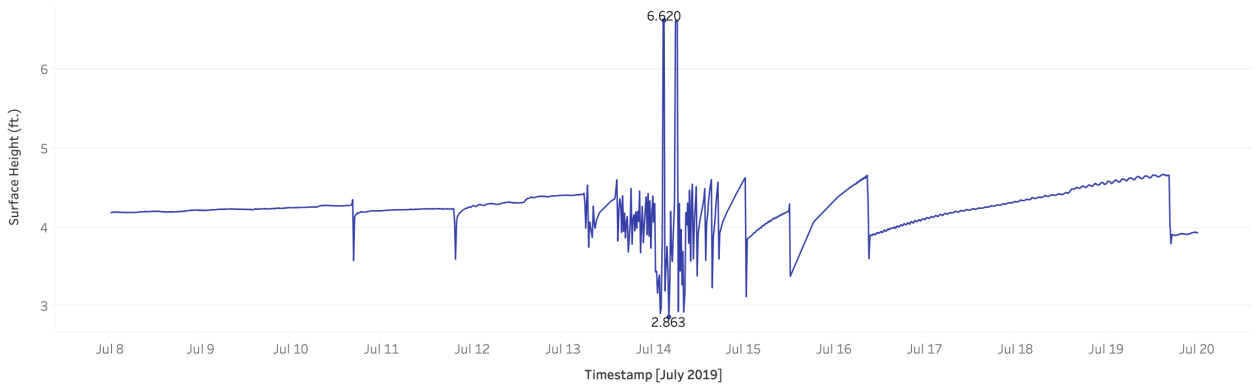


Figure 1.2. Water surface height recorded at a pump station in Lafayette, LA, during hurricane Barry.

pumps is turned on to remove water from the reservoir. The pump-off process takes mere minutes, while it generally takes a long time to fill up the reservoir, depending on rainfall. Between July 8 and early July 13th, the station behaved normally. However, the reservoir filled up very quickly between July 13th and July 15th (when tropical storm Barry was close to Lafayette). Therefore, the water pumps were turned on more frequently during this period to prevent flooding. We can see the pump-off activities as dips in the water surface level in Figure 1.2. The dips before the storm are spaced apart by days, while it is more concentrated during the storm. Pump stations equipped with IoT sensors provide information and notification in near real-time, which helps to make better decisions to prevent any damage by water.

Scientific visualization is concerned with graphically representing the true physical nature of the data. Visualization is often used to gain insights into material data with direct 3D space and time-relevant microscopic properties such as atomic configurations (crystal structures), charge distributions and bonding, and many microscopic phenomena such as molecular diffusion, crack propagation, fluid dynamics, etc. Data generated using

first principle molecular dynamics for microscopic properties can be visualized as a time-series trajectory [5]. This technique, paired with animation, can be used to understand the global and local spatial-temporal details, which can then be used to study bonding, pair correlation, coordination, structural units, clustering, structural stabilities, defects, diffusion, and other dynamical processes [5]. We can perform aggregation of these microscopic properties to understand higher-level phenomena containing macroscopic properties.

On the other hand, macroscopic properties such as density, elasticity, viscosity, etc., tend to generate only a few data samples at a time of a single complete study. Therefore, sophisticated visualization is not needed in all instances. However, since key macroscopic properties data have been piling up from years of work, we are interested in collecting, visualizing, and modeling them. To explore the macroscopic materials data, we can take the information visualization approach where the data's inherent physical structure is irrelevant. For instance, data on a scatterplot is shown in Figure 1.1. Parallel Coordinates Plot (PCP) is one of the widely used multivariate data visualization techniques [26] to get an overall view of data and to reveal the relationships, clusters, etc. Many recent works have been focused on either expanding the feature set of plot components, reducing visual clutter, or novel approaches to make correlations more visible [22, 27]. However, user-centric analyses based on PCP are still rare. Our study presents a user-specific enhancement of PCP for visual analytics on viscosity data. It is implemented as part of a web-based framework for managing, exploring, and analyzing data. The user interface for visualization is implemented in the client-side web browser. As part of an analytics framework, our PCP visual system has several unique features compared to other stand-alone tools:

- Our system connects and loads data directly from the database.

- Existing as well as model result data can be visualized using several tools.

- Finally, this web-based application enables users to conduct analysis from anywhere without the need for installation.

We have used enhanced PCP [26] (Figure 3.3) for data exploration. PCP has helped reveal the fundamental nature of the physical properties of silicate melts. Further, we have also integrated model result comparison and the multi-dimensional representation of the materials data in the plot. The model results can be visualized along with the actual data points in the same plot.

1.3. Data Modeling

Our focus on the viscosity data has led us to some of the studies of viscosity models we plan to include in our platform. Many standard models exist for representing the $\eta - T$ relationships for specific as well as multi-component silicate melts (e.g., [34, 24, 18]). These models are often fitted with the data for interpolation and extrapolation. The complex behavior of materials property requires a large number of data spanning a wide range of temperature, pressure, and composition ranges to get the most accurate generalized model. Due to limited data availability, most models span a narrow temperature, pressure, and composition regime. Evaluating existing models with a wider range of data provides a better understanding of their predictability. Currently, models exist in some variation of standard forms (or their combinations) by fine-tuning parameters such as composition components, temperature, and pressure. Another approach to data modeling is using ma-

chine learning. In this approach, models are constructed by iteratively evaluating models outputs against different parameter values by going through each data while minimizing errors with respect to the actual value. Models trained using this approach often overfit and hence require techniques such as cross-validation to confirm model generalization. Cross-validation splits data into many training and hold-out sets. A training set is used during model construction, while a test set is used to assess the model's validity on unseen data. Hence, an optimal model has high accuracy on both the train and test sets. We have experimented with many such models, some of which have produced close results relative to state-of-the-art physical models.

1.4. Web-platform

We aim to provide a database and web visualization platform to facilitate data analysis. Previous works on viscosity data collections were confined to either published materials or simple databases with some model calculations [24, 18]. Data are mostly in Excel or CSV format and are not centrally located. Experiments have been helpful in generating viscosity data; however, they are usually confined in low $P - T$ regime [58, 68, 71]. Computational techniques have started to generate data over experimentally inaccessible conditions (e.g., [2, 34, 16]). A combination of experimental and computational data appears to be a promising avenue to fully understand the viscous behavior of magmatic melts. Exploring these data due to the multivariate nature can be realized with visualization.

1.5. Requirements

The primary goal of this study is to facilitate a cloud-based software framework for geoscience research which involves data sharing, modeling, and visual analytics over the entire geologically relevant parameter range. Data will be available via a web application so researchers can access it anywhere. Further, we aim to provide a workflow for analyzing silicate melt viscosity model results. In more detail, the following are the requirements:

- **Public availability:** All materials data and software tools will be publicly available for anyone to explore through a web interface. Users can access data and tools using any modern web browser and the Internet. Since the application is on the web, no download, installation, pre-processing, or configuration steps will be necessary.

- **Visualization tools for data exploration and model analysis:** A large number of dimensions provide challenges to exploring data. Further, model analysis becomes non-trivial to perform along with the existing data. Therefore, visualization tools for large multivariate data exploration will be provided to facilitate visual analytics. These tools can be used for trend as well as relationship analysis between different data dimensions. The same tools can be used to evaluate different model results side-by-side.

- **Train and compare various machine learning models:** Some machine learning models that work well on regression tasks are trained and evaluated on various data subsets. These models are trained to lower the overall root mean

square (RMSE). Further, derivative properties such as glass transition temperature are also calculated and compared against published results. These models are also tested for other properties such as parameter space continuity, including the anomalous behavior of certain silicate melts viscosity with P.

Users can access https://lsuviz.github.io/pages/viscosity to explore and download the collected data using tools such as tables, PCP, scatter plots, and histograms. Users can also download a subset of the data using axis brushes on different dimensions and data table filters. Data display customization by adjusting dimension visibility and selecting different colormaps is also available.

1.6. Dissertation Layout

The next chapter takes a closer look at the materials data. After that, we will give more details regarding the enhanced PCP. Then we will explain various physics-based models and an approach to using machine learning models. We also include specific implementation and architectural details of the software framework. Finally, the visual analytics, training details, and model results will be discussed in the analysis section which will be followed by conclusions and future work.

Chapter 2. Database

In this chapter[1], we discuss materials property data and how they can be stored using approaches of various databases. Thereafter we discuss silicate melt viscosity data and their sources.

2.1. Materials Property Data

A given material property, such as melt viscosity considered in this study, can be viewed as a multivariate entity M such that:

$$M \rightarrow M(\eta, P, T, X1, X2, \cdots, I1, I2, \cdots)$$

In this multidimensional representation, the value of the property itself is considered one of the variables (attributes). It is a scalar quantity for the melt viscosity (η). But the property can be a multi-valued quantity, for instance, diffusion coefficients (defined on a per-atom basis) or elastic stiffness tensor. The parameter space in which a property is defined or determined involves factors such as pressure P, temperature T, and composition X. The compositional factor is multi-component in nature ($X1, X2, \cdots$), representing molar fractions of over ten oxides in the case of molten silicates. Additional information on data such as methodology, publication details, research group, and comments may also provide valuable insight during analysis. The metadata information can be included as additional attributes ($I1, I2, \cdots$). Researchers are also interested in building models using property values with respect to the parameter space and also to assess uncertainties/errors. We can use one or more derived variables to compare predicted results with actual data. Thus, compiled full information is visually mapped for meaningful analysis

[1]Some parts of this chapter can be found in D. Bhattarai., J. Zhang., and B. B. Karki. Parallel coordinates-based visual analytics for materials property. In Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - IVAPP, pages 83–95. INSTICC, SciTePress, 2019. Copyright permission included in Appendix A.

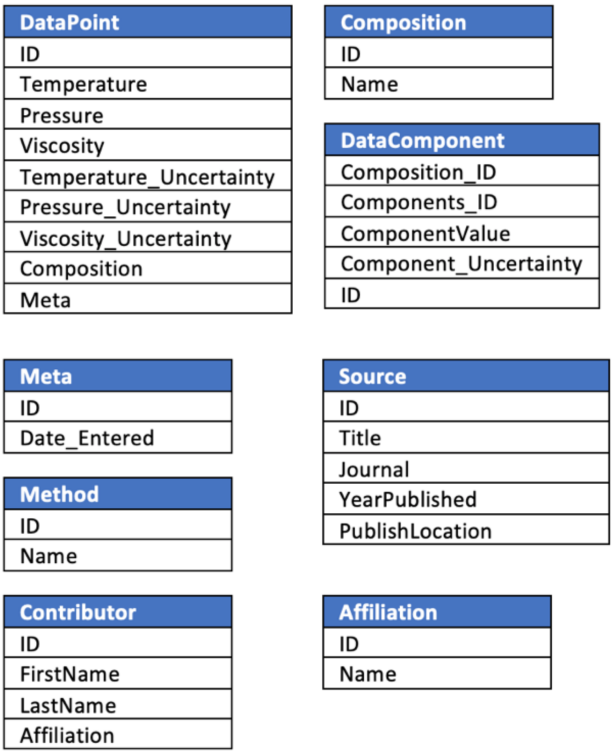


Figure 2.1. Database schema showing central tables used in data storage [6].

and modeling of material property (e.g., melt viscosity) in the question. Any other such properties can be represented in the same format by simply adding their data values as additional attributes to M.

Viscosity data are collected from various published sources. Data formats across datasets are usually different and must be transformed into a common format before storage and analysis. At the heart of it, our dataset is a list of viscosity values collected over different pressure, temperature, and composition ranges. Previous works on the viscosity databases have focused on only essential components for analysis and model building. Here we also include metadata and other relevant information (for example, model values and comments) along with actual viscosity values. Similarly, rather than storing data in a general text file format, we can organize a table structure for efficient data storage and re-

trieval (Figure 2.1). Here, data values are stored as real numbers in respective standard units whereas each composition component mass can be stored as a weight percentage or molar fraction. Therefore, summing up all components for a single data point always yields a 100% (1 for molar fraction). Detailed information for each data point can be extracted by joining appropriate tables.

In another approach, data can be stored in a document structure where each data point has complete information about itself (Figure 2.2). Unlike relational databases, here, data is stored in a schema-less structure, often in JSON (Javascript Object Notation) format. JSON has a flexible structure that allows each data point to store and append rich metadata. Further, JSON allows sparse data structure. For instance, some data points might not be eligible to be in the model and hence will not contain model results, however, other data points can contain fields to store model values. Each document represents a single data point and contains its complete information which enables high-traffic systems to store duplicate data in multiple nodes for better scalability. This is useful in big data and distributed computing scenarios where faster reads and writes are priorities over storage space. This loose data structure can then be queried and transformed into a rigid structure in the application layer. This way we can store all possible metadata with granular control over each data point.

Using any of these approaches, the end goal is to give users the ability to query data not only based on the actual data variables (η, P, T, X), but also by metadata. Further filtering and transformation are also possible on the application side. A public-facing software, such as a web application, will also be able to access and share all collected data. Researchers will be able to download all or a filtered set of data. This provides easy data

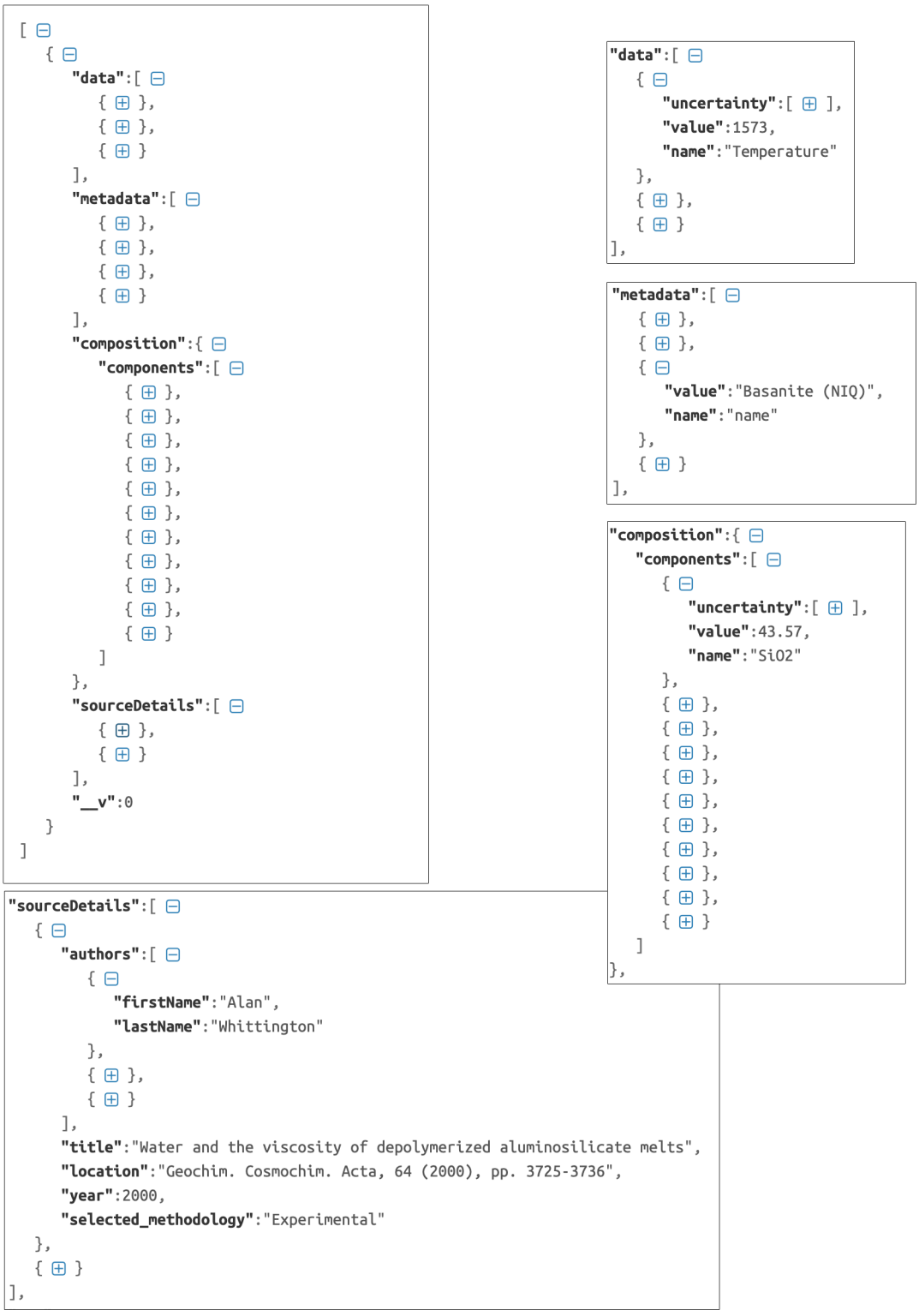


Figure 2.2. Simplified data structure (top-left) showing document structure in a non-relational database. Expanding each object reveals data further structured either inside other raw objects or within arrays (right and bottom).

accessibility for all interested material scientists.

2.2. Data Compilation

The proposed viscosity web application aims to collect all new and old viscosity data from various published sources. These viscosity data fall under three categories:

1. **Experimental $\eta - T$ data at zero-pressure:** This category includes viscosity-temperature experimental data for various compositions ([24, 18] and several others therein). These data were used to develop predictive models at a low-temperature range (< 1800 K). The composition range spans all terrestrial volcanic rocks.

2. **Data at elevated pressure:** The second category includes experimental data at a moderate pressure [54, 43]. Recent development has yielded 60 data points at up to 7 GPa for as many as 12 melt compositions [71, 13]. More measured data can be found in the literature for many silicate liquids including diopside [57, 54], albite [38, 48, 25, 62], Jadeite [39, 62, 64], dacite [65], andesite [42], peridotite [43], diopside-jadeite system [4, 63]. Unlike the zero-pressure data of the first category, these data points are relatively fewer, in the order of hundreds.

3. **Calculated melt viscosity results:** This category of viscosity data has recently shown a lot of progress and is on the rise to generate data over large parameter space. Specifically, first-principles simulations allow us to explore a much wider temperature and pressure range for silicate melts viscosity study relative to experimental data [33, 29, 31]. The MgO-SiO_2 binary melt composition temperature was

explored in high temperature (2000 - 6000 K) and pressure range (0 to 140 GPa). Other melt composition such as MgO, CaO, $MgSiO_3$,Mg_2SiO_4, $CaSiO_3$, SiO_2, diopside and anorthite, including a couple of hydrous liquids [33, 32, 15, 30, 69, 16, 31] results in over 500 data points. For completeness, we will also include data points from classical molecular dynamics simulations (e.g., [40, 50, 60, 2, 44]. More viscosity data points are expected during the time of this research. In this category, although data points are simulated, it has been shown that they agree well with experimental data. Since these data points are highly accurate, cover a large parameter space, and are available in large numbers and growing, they may be helpful in building better viscosity models.

Here, we have compiled melt viscosity data from both experimental and computational sources. Our database incorporates measured viscosity-temperature data at ambient pressure for melts of several compositions [24]. Next, calculated viscosity results have also been collected from sources using first-principles molecular dynamics simulation. There is much more viscosity data generated by other computational methods yet to be gathered. We anticipate eventually having several thousand records in the melt viscosity database.

Chapter 3. Enhanced Parallel Coordinates Plot (PCP)

To visually analyze the silicate melt viscosity data, we choose to adopt Parallel Coordinates Plot as this technique can address various visualization challenges related to large multivariate datasets. In essence, PCP maps all data items with respect to all variables/dimensions on a single display. The data polylines go across the display space sequentially through each dimensional axis.

Directly plotting the data on a standard PCP results in visual clutter because the data has a large number of dimensions and due to the complex relationship of viscosity with respect to its parameters. Problems such as the occlusion of polylines by others make it difficult to visualize the data fully. Therefore, we consider various standard and non-standard PCP features, falling broadly into two categories. First, the interaction with the data variables/dimensions is explored in detail with the derived axes, axis merging, and bi-scaling. Axis can also provide space to display additional information and overlays such as categorical bubbles or histograms. Second, we explore ways to interact with the data itself through polylines. One or more polylines can be selected from the entire data with appropriate color mapping and alpha blending. In this chapter[1], we discuss standard and some enhanced PCP features.

3.1. Standard PCP Features

A standard PCP contains data polylines going through a consecutive list of parallel vertical axes for a data record as shown in Figure 3.1. These lines intersect each axis at the corresponding scaled dimensional values. The axes thus divide a 2D drawing sur-

[1]Some parts of this chapter can be found in D. Bhattarai., J. Zhang., and B. B. Karki. Parallel coordinates-based visual analytics for materials property. In Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - IVAPP, pages 83–95. INSTICC, SciTePress, 2019. Copyright permission included in Appendix A.

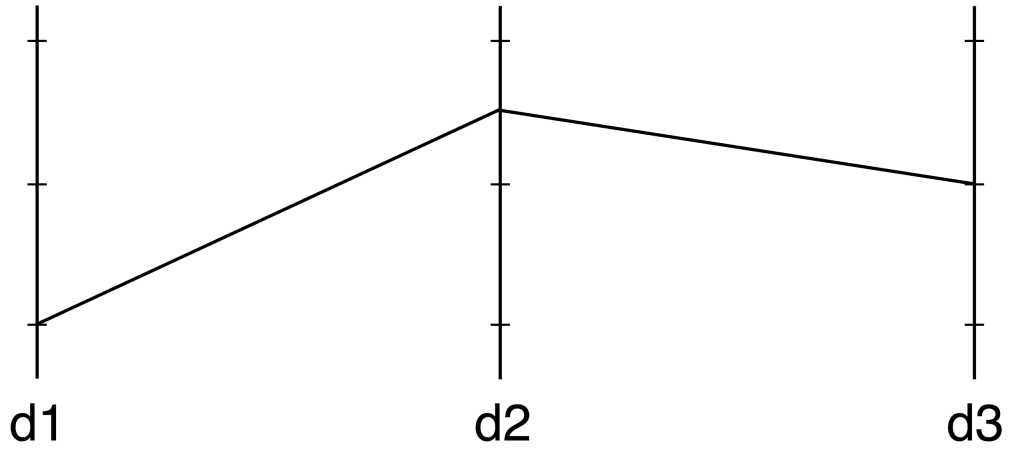


Figure 3.1. Basic Parallel Coordinates Plot with three dimensions $d1$, $d2$, and $d3$ with a single data point going across all axes.

face with respect to a k-dimensional data space onto k-1 sub-surfaces. Axes scaling is done with respect to maximum and minimum data values per dimension. Axes act as the main container for visual elements such as labels, markers, and overlays. These visual cues allow users to observe data space and read specific values. A linear (uniform) scale is used for most numerical data values. Categorical values are represented by transforming each category value to a point or bubble at the mapped axis location.

Plotting a large multi-dimensional dataset on a standard PCP may occlude some data. Data selection techniques such as axis-aligned brushing, probing, and pinching are used to filter the data. Brushing is useful along a single axis as well as when combined with brushes from multiple axes. Logical relations between dimensions such as *and* or *or* are used to construct higher-order brushes [67]. Similarly, pinching can be used to make data selections from the k-1 data sub-surfaces themselves. Polylines may also be given a discrete color map to differentiate between categories or a continuous color map to represent numerical values. In Figure 3.3, a discrete color map is used to differentiate between

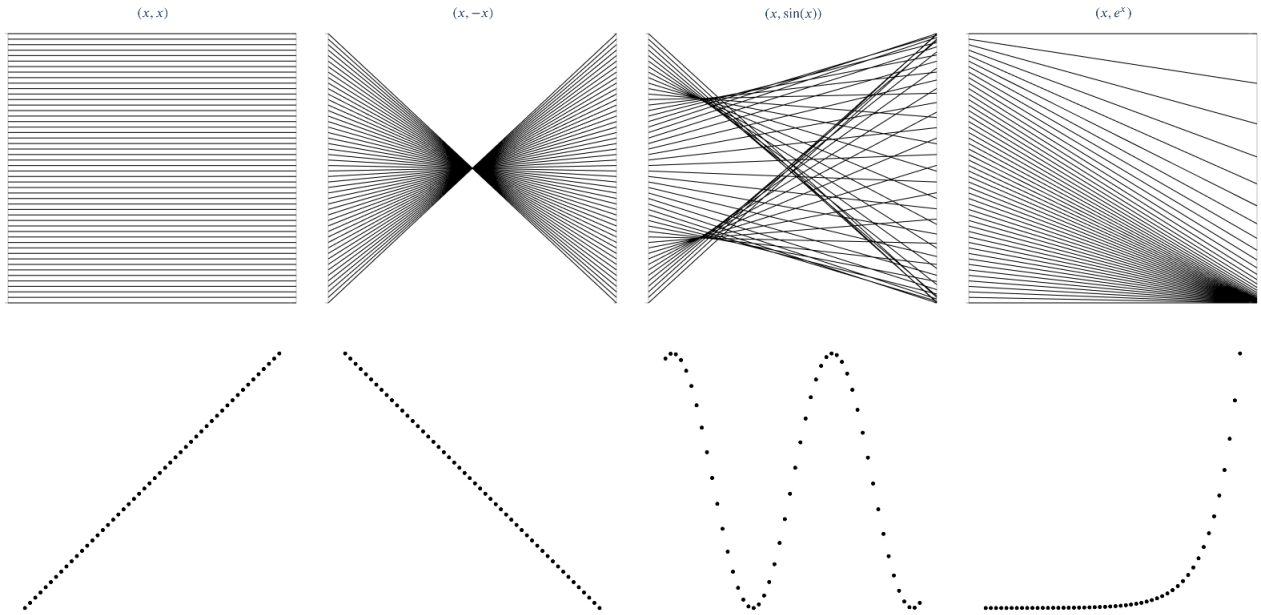


Figure 3.2. Common patterns in Cartesian coordinates (top) and their dual representation in parallel coordinates (bottom) (Heinrich et al. [22]).

experimental (red) and calculated (blue) data points.

Relationships between any two axes may emerge in the form of positive or negative correlation in any of the sub-surfaces between axes (i.e., between adjacent axes). Figure 3.2 compares common patterns when graphically plotted in PCP and cartesian space. The first two functions show highly positive and highly negative correlations. Negative correlation manifests itself with data lines crossing each other (perfectly negative correlation forms one point of intersection for all lines between the axes) while positive correlation forms parallel data lines going across the axes. Generalizing the above pattern, we can say that the lines will tend to cross each other between negatively correlated dimensions and the lines between positively correlated dimensions will tend to be parallel to each other. Other complex functions are also plotted for reference. For instance, we can observe the repeating patterns of a sin wave in the PCP space. We can observe these and many more

patterns by directly mapping the function parameter and values in the PCP space. Complex relations in a real physical system may contain a mixture of such patterns. In such cases, interactive plots may help in understanding the data. Different axis layouts may also help reveal previously hidden patterns. Axis reordering and flipping techniques are used to overcome the correlation identification problem. PCP systems usually also contain an augmented data table and scatter plot. We have added these features with additional interactivity between the plot and table.

3.2. Non-standard PCP Features

On initial load, our PCP system orders the axes by placing the most significant dimensions around the material property value dimension, which is viscosity in this study. It places pressure and temperature axes on the left side of the viscosity axis and places the composition axes (silica component followed by other components) on the right. The metadata axes are placed further away. Other axes orderings can be explored with drag-and-drop user interaction. Proper axis scaling is crucial for the true representation of data. While each categorical value is uniformly spread on its axis, the numerical values are dealt with differently.

In the standard PCP, the lowest domain values are mapped to the bottom while the highest ones are mapped to the top of the axis. This is true for all of our dimensions except the components. All components, regardless of their domain extents, are plotted from 0 to 100 (weight percentage) to show their proportions for a given data point effectively. Both linear and non-linear (e.g., logarithmic) scaling have been used for data representation. Since viscosity spans a large range, data patterns become difficult to observe

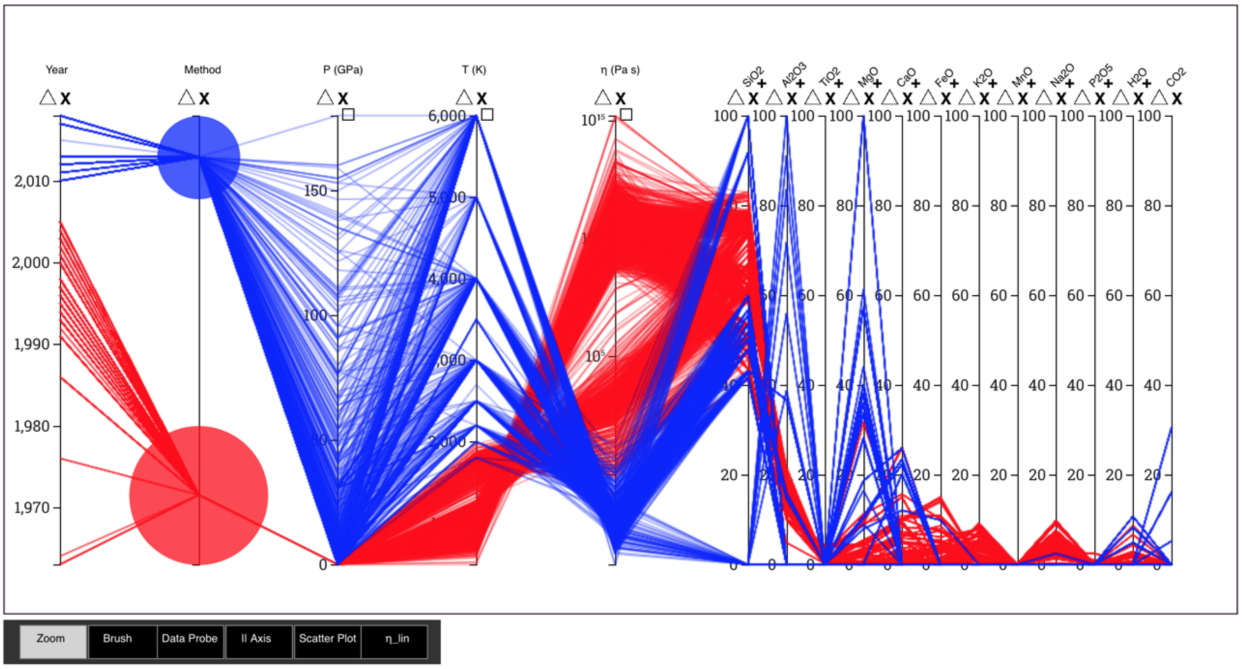


Figure 3.3. Parallel coordinates plot of 17-dimensional melt viscosity data. Experimental and calculated data are highlighted in red and blue colors, respectively. The plot shows numerical axes (viscosity, pressure, temperature, and others) and categorical axis (method). Linear and logarithmic scales are used, and the value range 0 to 100 (weight percentage) is used for all component axes. The triangle button inverts the axis, the cross button removes the selected dimension from the plot, and the square button performs binary axis scaling. The control panel at the bottom contains additional exploration tools such as Zoom, || Axis (derived axis), Data Probe, and others [6].

as the data size over the full range increases. A logarithmic scale is used for viscosity (Figure 3.3). Axes are usually laid out uniformly across the display space with axis spacing $\delta x = X_D/(k-1)$, where X_D represents the display width. This spacing layout provides equal significance to all dimensions. In the context of viscosity data, the users might be interested in exploring the relationships between the parameter space and viscosity values much more than the relationships between composition components. Therefore, to provide a larger space for interesting dimensions while also keeping the context of the overall data, our system uses the variable axial spacing technique (Figure 3.3) to separate the PCP into data and component regions. The data region can occupy one-half of the display space and contains the most important axes such as viscosity, temperature, pressure, method, and year. The component region is given the rest of the space with some padding. This region packs many (about a dozen) component axes.

3.2.1. Derived Axes/Sub-axes

Different data regimes for a dimension can be explored in large screen estate by augmenting a derived axis/sub-axis next to a primary axis. We can display the viscosity values using a linear scale on the derived axis alongside the logarithmic primary viscosity axis. Another scenario is that the user may want to focus on a sub-range of a variable. For example, the derived sub-axis maps filtered domain values (say, 1200 K $< T <$ 1600 K) to range $0 \leq r \leq y'$ using the same scale type as that of its primary axis (Figure 3.4). The data lines not falling in the chosen interval simply ignore the derived axis/sub-axis and continue along their paths to the next primary axis. This provides the context of overall data and details at a specific range. The minimum length of the derived sub-axis

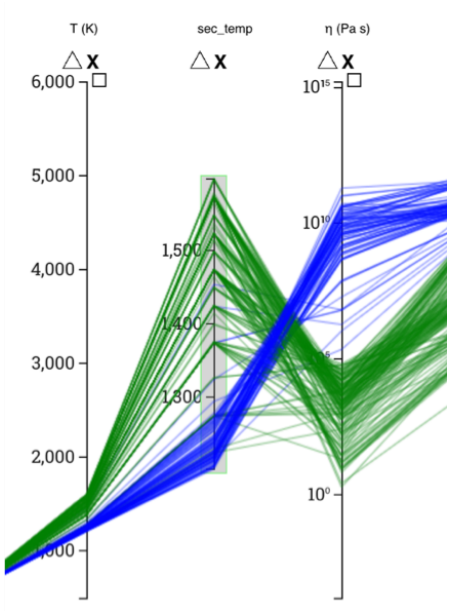


Figure 3.4. Derived temperature secondary axis showing a subset of temperature range between 1200 and 1600 K. Green represents viscosity values less than 104 Pa s and blue represents the rest [6].

is kept at half the length of its primary axis. However, if the chosen axes pair points contain more than 50% of the total data, the length is then made proportional to the number of points falling in the chosen dimensional domain range. Hence the length of the derived axis y':

$$y' = \begin{cases} n'(Y_D - p) & \text{if } n' > 0.5 \\ 0.5(Y_D - p) & \text{otherwise} \end{cases} \tag{3.1}$$

where n' is the ratio of the number of filtered data points to the total number of data points, Y_D is the vertical extent of the display, and p refers to the vertical padding value to accommodate dimension label and other controls on top of the axis. The horizontal position of all the axes is re-calculated taking the new axis into account. The derived axis is constructed such that a line through the middle point of both the primary

and the derived axis is orthogonal to both axes. This means translating the derived axis by $0.5(Y_D - p - y')$ in the vertical direction. Its position and height can be adjusted interactively. More derived axes are considered in the following sections.

3.2.2. Dimension Merging

Oftentimes, domain experts are interested in looking at various binary joins or ternary systems. For instance, MgO-SiO_2 join is considered to be the most important binary. The components CaO, FeO, and MnO can be treated on the same footing as MgO because these oxides play the role of structure modifiers and are highly mobile. They can be combined together and viewed as one compositional variable. On the other hand, SiO_2, Al_2O_3, and TiO_2 components together form a silicate polyhedral network and are mostly immobile. These oxides can be treated collectively as one compositional variable. In PCP, any two components can be merged together.

Merging components results in the insertion of a new derived axis which acts as an independent dimension itself. The data polylines are re-rendered by incorporating the new axis. Since the composition component is stored as a percentage, any two components can be directly summed together. Further, since a merged axis behaves as any other primary axes, it can itself be merged with other components. No component can be added twice so that the total mass of the composition for each data row is always at 100%.

3.2.3. Binary Scaling

Binary scaling contains two different scales on an axis divided at a user-chosen domain cut-off value. Let n_1 be the number of data points whose values are equal to or less than the cutoff and n_2 be the rest of the data points. The sub-lengths are assigned for the

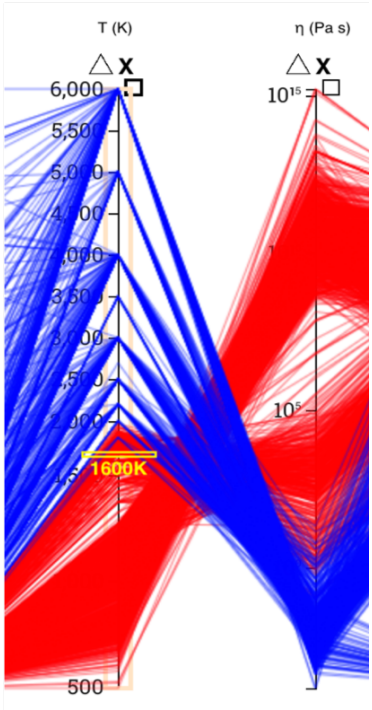


Figure 3.5. Binary scaling applied to the temperature axis with a cutoff point at 1600 K. Axis is scaled proportionately with respect to the number of data lines falling under and over the specified cut-off point [6].

two scales along the complete axis as follows:

$$l_1 = \frac{T_D - p}{\frac{n_2}{n_1} + 1} \tag{3.2}$$

$$l_2 = Y_D - p - l_1 \tag{3.3}$$

We can now create two scales of dimension-specific scale type with different domains and ranges: $0 \leq r_1 \leq l_1$ and $l_1 < r_2 \leq (Y_D - p)$. This scheme can be further extended by implementing more than two scales on the same axis. Figure 3.5 shows a bi-scaled temperature axis such that the lower range of 500 to 1600 K is stretched while the upper part is compressed.

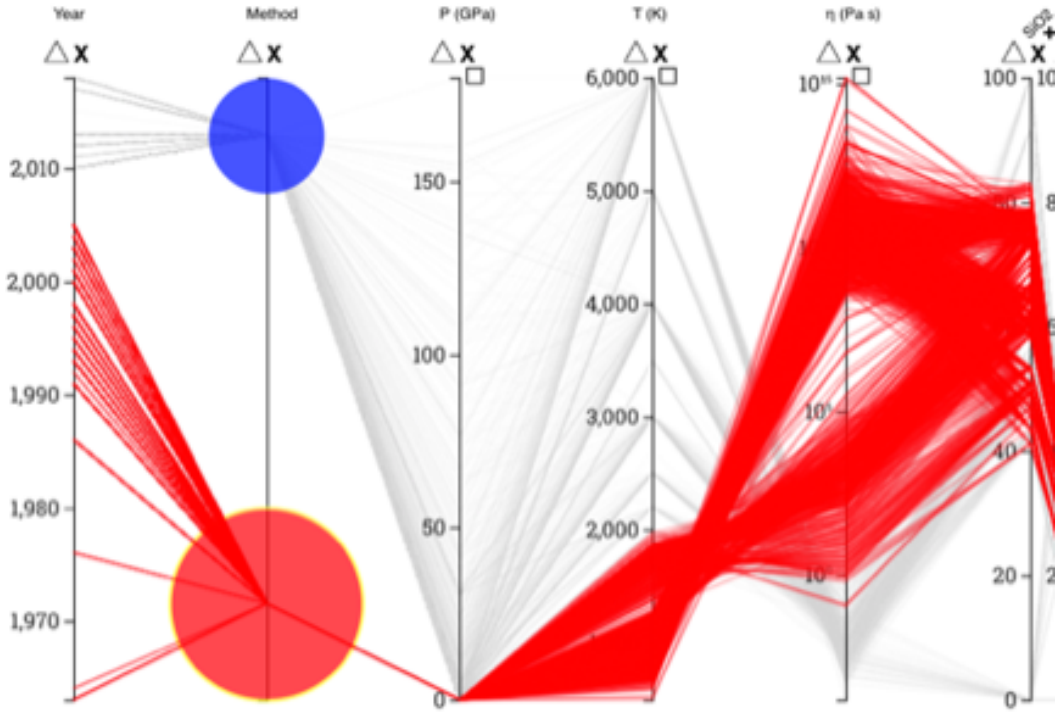


Figure 3.6. Experimental data selection by clicking categorical bubble (lower) shown in red while the calculated data are shown in the background (gray polylines) [6].

3.2.4. Categorical Bubbles

In PCP, it is desirable to treat numerical and categorical axes differently. Categorical variables consist of distinct categories which are difficult to map onto an axis directly. Therefore, a transformation to a metric scale must be done such that each category gets uniform space in the axis. One such data transformation can be done by overlaying circles or bubbles of varying radius on axes as category markers [66]. The radius of each bubble can be utilized to show different data properties. For instance, we can map the radius (r) with the frequency (f) of the data points falling under a category:

$$r = \sqrt{\frac{f}{f_{\max}}}\frac{\delta x}{2} \tag{3.4}$$

Here, $f_{\max}$ is the number of data rows for the category with maximum frequency. We utilize categorical bubbles to differentiate between experimental and calculated data

points. Bubbles can also be used for user interactions such as to hide and show data based on mouse clicks (Figure 3.6). Categorical bubbles can be further split into smaller bubbles with respect to data values from another axis (Figure 4.2). For instance, the experimental category can be split with respect to different viscosity regimes. The split bubbles are stacked on top of each other such that the sum of their radii equals the radius of the merged bubble. The radius of each split bubble is proportional to the frequency of the data row falling under both the original and split categories.

3.2.5. Nested PCP

A recent study has shown that using nested PCP to visualize model parameter correlation between different datasets is more useful than superimposed or juxtaposed PCP representations [70]. As we show, this technique can be further extended to analyze two or more subsets (groups) of the data corresponding to different intervals on selected numerical axis (for instance, low-pressure versus high-pressure regime) or different categorical values (for instance, experimental versus computational). Nested PCP resides symmetrically about the midline between two adjacent axes under consideration [70, 35]. The vertical space between two primary axes is divided into $\frac{Y}{n_s}$ uniform regions where n_s is the number of nested categories and $Y = Y_D - p$. The categories are sorted by the mean location of each polyline on either one or both of the primary axes. Each nested axes is then constructed symmetrically from the middle of its category region where the endpoints of the j^{th} nested axis ($j = 1, 2, 3, \cdots n_s$ counting from the bottom) are first constructed using $(j - 0.5)\frac{Y}{n_s} \pm \delta d$, where δd can vary between $\frac{0.2Y}{n_s}$ to $\frac{0.4Y}{n_s}$. A translation is then applied between 0 to $\pm\frac{0.1Y}{n_s}$ depending on the location of the maximum pixel value of the polylines

on the selected primary axis. The horizontal spacing of the i^{th} nested axis, h_i is given by $(i-0.5)\delta x \pm \delta h$ where $i = 1, 2, \cdots k-1$ and δh can vary between $0.1\delta x$ and $0.4\delta x$. This spacing scheme allows no overlap between the axes horizontally or vertically. A single line previously going between two primary axes is now replaced with two curves and a line. We use a cubic Bezier curve with control points' x location at $(i-1)\delta x \pm \alpha\delta n_x$ from primary axes where δn_x is the distance between the primary and nearest nested axis. Similarly, $(i-0.5)\delta x \pm \delta h \pm \alpha\delta n_x$ from nested axes where α can vary between 0.1 to 0.3 (Figure 3.7). The y values of these control points are the same as the y values of the data line in primary and nested axes, respectively. The curves join the primary axis with the nested axis and a straight line is drawn between the two nested axes for each data point. The nested axes display the full data extent range based on their subset's minimum and maximum domain extent (that is, a local scale is applied).

We use the nested PCP to compare the actual data and model-predicted values using the primary η axis, and the derived η_{model} axis (Figure 4.2). We consider two scenarios of the zero pressure $P-T-X$ model [34]. In one case, we apply the model to the pure MgO-SiO_2 binary system, where X represents the molar fraction of silica. In the other case, we apply the model to the multi-component system by taking X as the sum of SiO_2, Al_2O_3, and TiO_2 fractions. There are two nested plots. For model result comparison, using the same domain range for both axes in each nested plot category is desirable. Similarly, we use the nested plot to show the anomalous behavior of silicate-rich composition at 3000 K at two different pressure ranges (Figure 6.2). Here we use the local domain extent for two nested plots corresponding to viscosity and pressure.

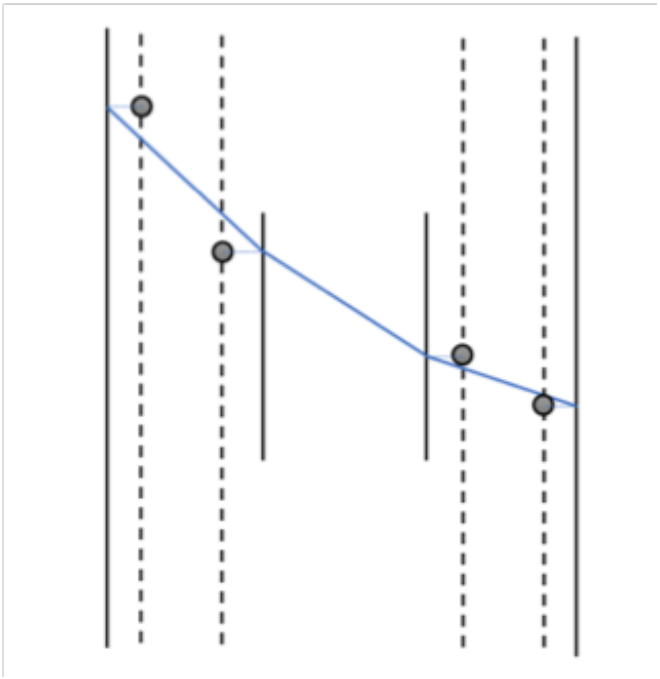


Figure 3.7. Nested PCP along with four control points for Bezier curves between nested and primary axes. The polylines connecting between the primary and nested axes are replaced by two curves [6].

Chapter 4. Viscosity Models

Our focus on the viscosity data has led us to some of the studies of viscosity models we plan to include in our platform. In this chapter[1], we will describe some viscosity models before discussing the modeling workflow details. Many models exist for representing the viscosity-temperature relationships for specific compositions and multi-component silicate melts (e.g., [46]). The earliest model adopted is the Arrhenius law:

$$\ln \eta = A + \frac{E_A}{RT} \tag{4.1}$$

where parameter A refers to the log of a notional value of viscosity at infinite temperature, E_A is the activation energy for viscous relaxation, R is the gas constant, and T is the temperature. This model was calibrated using the experimental data [8, 58]. For the first-principles results of seven liquids along the MgO-SiO_2 join, Karki et al. [34] derived a global Arrhenian model:

$$\ln \eta(T, X) = (A_0 + A_1 X^4) + \frac{E_{A0} + E_{A1} X^4}{RT} \tag{4.2}$$

where it is interesting to note that the pre-exponential factor and activation energy depend on composition as the 4^{th} power of the molar SiO_2 content (X). Therefore, implying a strong non-linear dependence of the configurational entropy on composition. In other words, the degree of polymerization controls the melt viscosity. This model works for a greater range of temperature (2000 to 8000 K) and composition (the entire possible range

[1]Some parts of this chapter can be found in D. Bhattarai., J. Zhang., and B. B. Karki. Parallel coordinates-based visual analytics for materials property. In Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - IVAPP, pages 83–95. INSTICC, SciTePress, 2019. Copyright permission included in Appendix A.

of NBO/T) compared to the previous models.

The original and modified Arrhenian models do not work for some silicate liquids that demonstrate non-Arrhenian behavior. This calls for the generalization of Equation 4.1 (e.g., [49, 23, 17, 72]). A commonly used equation is called the Vogel-Fulcher-Tamman (VFT) equation:

$$\ln \eta = A + \frac{B}{T - T_0} \tag{4.3}$$

where A, B, and T_0 are the adjustable parameters representing the pre-exponential factor, the pseudo-activation energy, and the VFT temperature of viscosity divergence, respectively. In one of the current multi-component viscosity models developed by Giordano, Russell, and Dingwell [18], A is assumed to be a constant, independent of composition so that B and T_0 capture all compositional effects. Their model consists of 18 coefficients to include the viscosity of fragile, strong liquids and incorporate the effects of volatile compounds.

Two other forms of a three-parameter viscosity model include the equation based on an atomic hopping approach [3]:

$$\ln \eta = A + \left(\frac{B}{T}\right)^{\alpha} \tag{4.4}$$

where, A, B, and α are the adjustable parameters. Recently, a new viscosity equation which is based on the temperature dependence of configurational entropy as required by the Adam-Gibbs relation is as follows:

$$\ln \eta = A + \left(\frac{B}{T}\right) exp\left(\frac{C}{T}\right) \tag{4.5}$$

where, A, B, and C are the adjustable parameters dependent on composition. These parameters in models can be related to two other physically meaningful quantities: the glass transition temperature and the fragility [18, 46]. Besides the above standard models, a four-parameter empirical relation was proposed by Hui and Zhang [24]:

$$\ln \eta = A + \frac{B}{T} + exp\left(C + \frac{D}{T}\right) \tag{4.6}$$

Here the fit parameters are expressed as linear functions of many oxide components and exponential dependence on H_2O component.

Although there remains some disagreement on which non-Arrhenian functions provide the best description of melts' behavior, the effects of temperatures are arguably the most studied on viscosity. Besides temperature, some success has been met in quantifying the effects of composition on viscosity (e.g.,[24, 18]). Additionally, the pressure effects have been examined qualitatively (e.g., [38, 57, 60, 30, 69]). These studies observed that the silicate melt viscosity increases with pressure in a depolymerized melt but decreases with pressure in a polymerized melt such as albite or jadeite melt in the low-pressure regime. Only a handful of quantitative studies on the effects of temperature, pressure, and compositions together on viscosity have been studied.

Model building starts with the functional form of the model (e.g., the variations of the VFT equation for viscosity). The parameters of the functional form are determined by fitting the model to the data. The fit results are then analyzed properly to ensure a

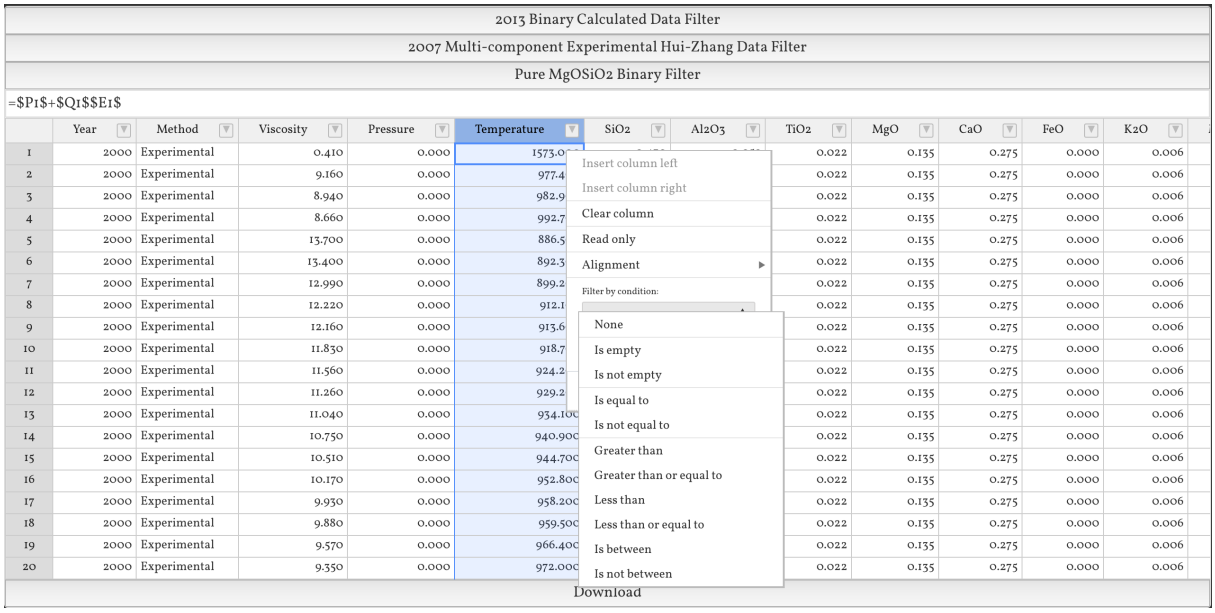


Figure 4.1. Filter buttons are placed above the table for easy data selection. An advanced filtering window is shown for the Temperature column where data selection can be made with complex logical and range operators such as *Is Not Empty* and *Is Between*.

proper fit of the data is established. New ideas may emerge from these analyses to make the model reflect the data better or use a different functional form, implying a different model. Therefore, model building is a process of exploration where researchers try to build, modify, and analyze models iteratively. Our software platform allows researchers to engage in this iterative process with minimal effort and consistent focus on the model rather than the low-level details of building one.

Model results can be plotted against currently available data. We have included two different models in our system - binary MgO-SiO_2 [34] developed using calculated data and zero-pressure multi-component composition [24] developed using experimental data. Both models require separate treatment of data as model input. For instance, a common zero-pressure data filter must be applied before calculating model results. For the binary model, we treat composition components in molar fraction; for the experimen-

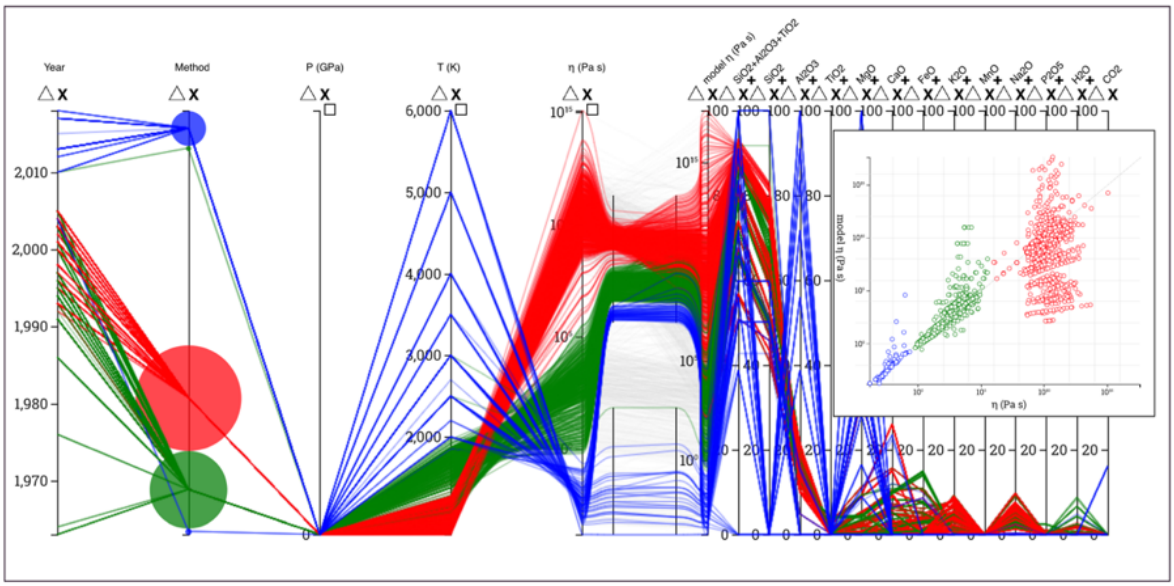


Figure 4.2. Model results plotted with discrete colormap to highlight three apparent clusters in the viscosity axis at zero pressure. The color also follows a positively correlated pattern in the model result axis. The lower nested axis shows a pure MgO-SiO_2 system, and the upper contains the rest of the compositions. The calculated categorical bubble is split into two, while the lower experimental bubble is split into three bubbles [6].

tal data model, they are used in weight percentage. More complex filters may need to be applied for proper data selection. Therefore, for ease of use to researchers, we provide several default filters in the system - the 2013 binary calculated data filter, the 2007 multi-component experimental Hui-Zhang data filter, and the pure MgO-SiO_2 binary filter. These filters can be applied by clicking the filter buttons found above the data table (Figure 4.1). Clicking filter buttons apply filters to all data in the database and subsequently also displays filtered data in the PCP.

4.1. Multi-component Experimental Data at Zero-pressure

An existing data filter is applied to select experimental data collection at zero pressure before proceeding with model result calculation. This filter selects only data used while optimizing the model given by Equation 4.7. The data selection contains all-natural

silicate melts collected via experimental methods. The compositions include anhydrous peridotite to rhyolite, anhydrous peralkaline to peraluminous melts, hydrous basalt to rhyolite, and hydrous peralkaline to peraluminous melts [24]. The viscosity range of such data selection is between 0.1 to 10^{15} Pa s while the temperature is between 573 to 1978 K. The pressure of the selected data ranges between 1 and 5 kbar, which are shown to be negligible [53, 72] and can all be treated as ambient pressure.

The model presented here is non-trivial as it involves pre-calculating certain entities like the value of Z, which depends on water. Water in hydrous oxides plays a significant role, and the effect on viscosity is non-linear. The form of Z addresses this non-linearity and uses other linear terms for water in mole fraction. During fitting viscosity data, the trial-and-error process constructed this water content raised to a power relation (where $e_1 = 185.797$).

$$
\begin{aligned}
\log \eta = \Big[& -6.83X_{\mathrm{SiO_2}} - 170.79X_{\mathrm{TiO_2}} - 14.71X_{\mathrm{A_2O_{3cx}}} - 18.01X_{\mathrm{MgO}} - 19.76X_{\mathrm{CaO}} \\
& + 34.31X_{\mathrm{(Na,K)_2O_{ex}}} - 140.38Z + 159.26X_{\mathrm{H_2O}} - 8.43X_{\mathrm{(Na,K)AlO_2}} \Big] \\
& + \Big[18.14X_{\mathrm{SiO_2}} + 248.93X_{\mathrm{TiO_2}} + 32.61X_{\mathrm{Al_2O_{3ex}}} + 25.96X_{\mathrm{MgO}} + 22.64X_{\mathrm{CaO}} \\
& - 68.29X_{\mathrm{(Na,K)_2O_{ex}}} + 38.84Z - 48.55X_{\mathrm{H_2O}} + 16.12X_{\mathrm{(Na,K)AlO_2}} \Big] 1000/T \\
& + \exp \Big\{ \big[21.73X_{\mathrm{Al_2O_{3ex}}} - 61.98X_{\mathrm{(Fe,Mn)O}} - 105.53X_{\mathrm{MgO}} - 69.92X_{\mathrm{CaO}} \\
& - 85.67X_{\mathrm{(Na,K)_2O_{ex}}} + 332.01Z - 432.22X_{\mathrm{H_2O}} - 3.16X_{\mathrm{(Na,K)AlO_2}} \big] + \big[2.16X_{\mathrm{SiO_2}} \\
& - 143.05X_{\mathrm{TiO_2}} - 22.10X_{\mathrm{Al_2O_{3cx}}} + 38.56X_{\mathrm{(Fe,Mn)O}} + 110.83X_{\mathrm{MgO}} \\
& + 67.12X_{\mathrm{CaO}} + 58.01X_{\mathrm{(Na_aK)_2O_{ex}}} + 384.77X_{\mathrm{P_2O_5}} \\
& - 404.97Z + 513.75\ \mathrm{X_{H_2O}} \big] 1000/T \Big\}
\end{aligned}
\tag{4.7}
$$

$\mathrm{Z}=(X_{\mathrm{H_2O}})^{\frac{1}{\left(1+\frac{e_1}{T}\right)}}$;

$(\mathrm{Fe,\ Mn})\mathrm{O} = \mathrm{X_{FeO}} + X_{\mathrm{MnO}}$;

$\mathrm{t} = \mathrm{Al_2}O_3 - (\mathrm{Na_2O} + \mathrm{K_2O})$;

$\mathrm{Al_2}O_{3ex} = 0; (Na, K)_2O_{ex} = 0$;

if *t is negative* **then**

$\mathrm{Al_2}O_{3ex} = t$;

$(\mathrm{Na,\ K})_2O_{ex} = 2 * (\mathrm{Na_2O} + \mathrm{K_2O})$;

else

$\mathrm{Al_2}O_{3ex} = (-1) * t$;

$(\mathrm{Na,\ K})_2O_{ex} = 2 * \mathrm{Al_2O_3}$;

end

Algorithm 1: Calculation of excess oxides and Z for Hui Zhang equation.

Oxides are treated linearly. However, data processing is needed before using them in the model (for example, calculating values for excess Al, Na, and K oxides). These calculated values are plugged into the model (Equation 4.7). The model is constructed using a combination of different standard models (Equation 4.6). The model assumes parameters A, B, C, and D as linear functions of oxide mole fractions. This assumption allows assigning different oxides weights, individually contributing their effects on viscosity. Since this model attempts to predict viscosity for naturally occurring melts, it also assumes both Ferric and Ferrous oxides to be the same oxide component. Even though the oxidation state of iron plays a significant role in melt viscosity, especially in the lower temperature regime [14, 42], often due to insufficient information in the dataset, all iron oxides are hence treated as FeO. FeO is combined with MnO to form (Fe, Mn)O. Melts containing Al oxides are also treated with particular specificity. Studies [55, 9] have shown that viscosity is higher in saturated Al melts compared to under-, or over-saturated Al melts. Since Al_2O_3 often combines with alkalis, forming (Na, K)AlO_2. After the combination, we are left with either excess alkali or aluminum oxide. The algorithm to calculate these excess oxides along with Z and (Fe, Mn)O are shown in Algorithm 1.

This model was trained using all available data which successfully reproduces natural silicate melts experimental data. Model predictions were also very close to the actual viscosity values when tested with data from the outside training set. This test data included hydrous and anhydrous compositions to test the wide compositional range properly. All oxides are given in mole fraction, and temperature is given in K. This 37-parameter model performs quite well with the data it was trained on with a 2σ deviation of 0.61 $\log_{10}\eta$ Pa s in the dataset $T - X$ range. A scatter plot of the predicted viscosity falls near

the actual values, and there are no significant outliers. The model behaves systematically and monotonically across the $P - T - X$ domain range. Further investigation of model results by composition presents minor uncertainty changes but has no systematic misfit or huge data scatter across the whole compositional range.

Although the model shows interesting predictability, it does come with certain constraints. As mentioned earlier, all oxides are taken in mole fraction, therefore, if composition component measurement is done in weight percentage (wt%), it would have to be converted into mole fraction first. Likewise, the temperature is in K. The model should not be extrapolated to binary systems, nor viscosities above 10^{15} Pa s, nor to a temperature below 573 K, nor to H_2O content above 5 wt% for melts other than rhyolite (for rhyolite, the highest H_2O content in the viscosity database is 12.3 wt% [24]). Another drawback of the model is a large number of parameters. Also, the model result uncertainty is still larger than the experimental data uncertainty, generally, 0.1 $\log_{10} \eta$ units or less. Besides the drawbacks, the model still reproduces viscosities with significant accuracy compared to its previous attempts. This model also considers the non-linear water dependence and the combined effects of ferric/ferrous and aluminum/alkaline oxides.

4.2. MgO-SiO_2 Binary at Zero-pressure

Going higher in $P - T$ regimes poses challenges in experimental data collection, leaving a large region of parametric conditions yet to be explored. Very few data points exist for these regions. Therefore, not many modeling efforts are made. However, melts at such a higher $P - T$ range may have been very widespread during the early formation stages of the Earth. It is also hypothesized that a giant impact might have caused the

whole mantle to melt and form a magma ocean across the whole planet with temperatures exceeding 10,000 K. Therefore, studying mantle melts is important to understand melt generation and dispersal [61]. Recent developments in first principles molecular dynamics simulations (FPMD) have shown promising results in modeling viscosity even at higher temperature and pressure regimes. Most relevant models, like the one shown in the previous section, consists of a narrow temperature range (less than 2000 K) that can only be used to explain shallow magma migration and volcanic eruption. Besides the $P - T$ range, since the data are generated from simulation, much fine-grain control over composition is also possible. For instance, in [34], MgO-SiO_2 binary composition is taken with varying mole fractions of each component. Since selected compositions are pure binary, it is modeled by a single term X representing the mole fraction of the silica component present in a chosen composition.

First, data selection is made using the 2013 binary calculated data filter. After filtering, we are left with only 50 rows of data (Figure 4.3) that were used to train the model. We can observe that the temperature ranges from 2000 - 8000 K, which is outside the experimentally collected data range of natural silicate melts. Other data with higher pressure are also included in the database but have not been included in this model evaluation since this model is valid only on zero pressure. We can also see the presence of some pure silica and some pure MgO in their respective axes. The compositional data varies uniformly in terms of mole fraction. This model gives out viscosity result values in natural base logarithm ($\ln \eta$), whereas the previous model works with base ten logarithm ($\log_{10}$).

We have used some new data to test further the model behavior outside the model optimized $P - T - X$ range. This selection includes data with zero pressure but with

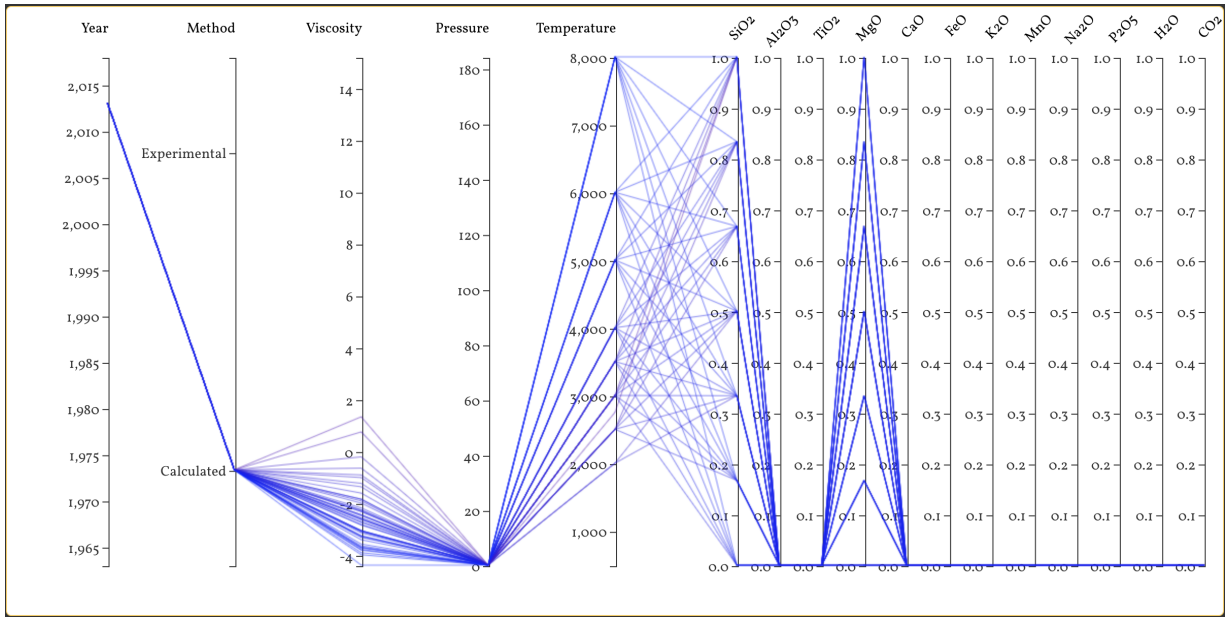


Figure 4.3. MgO-SiO_2 binary composition at zero-pressure calculated data filter applied to all data.

wide temperature and compositional range. We augment zero-pressure experimental data to test alongside calculated data at higher temperatures and pressure regimes. The non-binary composition may include many components, some containing water. Using results from Equation 4.2 (along with optimized parameter values), we make two types of model assessments and display the results using the nested plots for the η and η_{model} axes (Figure 4.2). The model works well for the MgO-SiO_2 binary at zero pressure, as shown by nearly parallel horizontal data lines between the two axes. To evaluate the model for the whole dataset, we take X as the sum of the molar fractions SiO_2, Al_2O_3, and TiO_2. Many data lines follow the horizontal trend, and the model data points show the cluster patterns found on the viscosity axis. These signify the dominant role of silica in controlling melt viscosity. However, on further exploration, several exceptions (e.g., polylines skewness, line crossing) are associated with other compositional factors. This means that the binary model is insufficient, and a general multi-component viscosity model is needed.

4.3. Model Comparison

Model results can be compared directly using actual values in the data table. A simple approach to understanding the difference between actual viscosity values and model results is to compare model root mean square error (RMSE). RMSE is one of the error measurement techniques, along with others such as mean square error and mean absolute error. In general, model results are first calculated and transformed into a common logarithmic base (or directly representing η) which is then subtracted from the actual viscosities. The square of difference for each data row is then averaged, and finally, the square root is taken to bring the mean squared error in the same scale as the actual value units. This value represents the error or distance between the actual viscosity and the model-generated one. Squaring the values eliminates the effects of positive or negative skewness of the predicted data. Generally, lower RMSE values represent fewer model result errors, but very low RMSE values may negatively impact the model by overfitting, which loses model generalizability for unseen data. Therefore, a balance between low RMSE such that the model generalizes well over the parametric region is sought after during model evaluation. However, RMSE is just an aggregate measure of model performance and hides much information about data and model behavior. For instance, to compare predicted values across different models per composition, researchers must select data relevant to the model and compositional constraints and can only calculate error measures.

The visual process can also aid during model evaluation. For example, plotting model results (in the y-axis) against actual viscosity values (in the x-axis) will show overall data scatter. If the predicted and actual values are close, the data will be scattered

around the $y = x$ line. Plotting both model results while coloring the data points by model results may help display the model's overall behavior. However, not all data points have the model results for all parametric conditions. For instance, the data trained on the binary model only takes the MgO-SiO_2 binary composition, while the experimental model can take any multi-component composition. This difference is more apparent while plotting data in PCP. Data lines falling outside model constraints are discontinuous at the model result dimension. Since losing continuity in PCP further complicates visual data understanding, we will add a dotted line with a light color to indicate discontinuous regions between dimensions due to lack of value. These dotted lines will be present by default, but their visibility can be toggled with the user interface. This way, we provide continuity in the data line for any data point, including those with missing model result values. It is also desirable to match the axis scale and corresponding tick marks and values while comparing two model results side-by-side. The default axis scale will match the larger domain range of the model results or actual viscosity values and uses it for all viscosity-related primary axes. Users can always use the zoom tool to change the resolution of the axis if they desire. It is also beneficial to place model results alongside actual viscosity values for the selected data points by dragging an axis to a new location. For a well-approximating model, data lines between the model result and actual viscosity dimensions should be parallel.

Further investigations on the sub-regions can be done by adding NPCP, secondary axes, and other non-standard features. Users can generate multiple scatter plots between any two dimensions at any point. These scatter plots will have the option to color by model results which can help evaluate model performance in different regions. Outliers can

be detected using scatter plots or by augmenting a new dimension in PCP, which holds the difference between the model result and the actual viscosity value for each selected data point. Brushing along the extremes (both top and bottom) can help reveal outlier model results. Users can construct histograms on the axis itself to view data distribution on this axis. Therefore, using the above workflow, PCP may help explore data models which would have been tedious to work with just data-table. Interactions such as brushing, dragging, zooming, and construction of scatter plots will also help users deep-dive into their data regions of choice.

For viscosity modeling, even a small number of model forms and constraints yields a large hypothesis space making it quite difficult to find an optimal model. Considering the large parameter space and its complex dependence on the materials' properties, it is necessary to find the best possible model to understand its behavior over all natural conditions. Manual search for model form and parameter values hence is infeasible. Therefore, automated model building is sought after for huge hypothesis space problems like these. However, even with ample computational resources, it might just take too long to finish searching. Therefore, we plan to automate an optimal model search process using machine learning.

4.4. Linear and Nonlinear Fitting

Fitting refers to finding parameters for a model such that the function results are very close to the corresponding data values. Here we will consider a comprehensive set of viscosity models under various pressure, temperature, and composition conditions. The most popular fitting method is to minimize the difference between the values predicted

by the model from the actual values in log space: $\sum_i \left(\left(\ln \eta^i, f(C^i, \theta) \right) / \sigma^i \right)^2$, where η^i is the measured and f is the calculated viscosity with respect to C^i which is the vector of all conditions, θ is the parameter vector, and σ^i is an error on the measured $\ln \eta^i$. If the model function f happens to be linear, we can treat this minimization problem as a standard linear least-square problem. However, model functions are usually not linear, and no analytic solution exists for non-linear minimization problems. Instead, numerical methods are used to solve them. A numerical method is an iterative approach to finding the solution to a minimization problem. The basic algorithm improves the parameters at each iteration to reduce the sum of the squared errors. One popular approach to solving non-linear minimization problems is the Gaussian-Newton method using improved algorithms such as the Levenberg-Marquardt algorithm [41, 45]. Several well-known packages already exist to solve these fitting problems; hence we will utilize existing optimization software to implement generic weighted least square to address the following specific issues:

Model and Parameter Space Constraints. Some parameters may require constraints to work properly within the model. One can specify a range of values for these parameters while constructing a model. For example, a term in some models may be a parameter serving as an exponent. Depending on the term's physical meaning, it may be required that the exponent be positive and have an upper bound. The numerical solution for our non-linear models should be able to satisfy this constraint. Therefore, we will include constraint fitting in our framework. This often leads to quadratic programming for the linear models, which can be solved using standard numeric methods. For non-linear models, Gaussian-Newton and the Levenberg-Marquardt algorithms can be used to find

solutions in the regions where the values satisfy all restrictions.

Local Minimum and Multiple Starting. Unlike linear models, non-linear models fitting may come across multiple local minimums. It is required to start numerical searches from multiple locations to improve the chances of finding the global minimum.

Modeling Different Parametric Regions. In some cases, multiple models may be needed to explain the viscosity across a wide range of parameters. For instance, models in low-pressure and high-pressure regions can be different. Different models can be built with the data falling in different pressure and temperature regions. While using multiple models, any two neighboring models should agree at the boundary. This task is non-trivial. Therefore, a single function mapping from the parametric space to viscosity is highly desirable.

4.4.1. Fitting Analysis and Model Selection

An essential step after the fitting process is to analyze and validate the results. If the model is perfect, all the points in the fitting results fall precisely on the data forming a $y = x$ line. Comparing fitting results to true values is one of the most common ways to analyze fitting results. Ability to analyze model continuity and derivative properties, such as glass transition temperature, give the model its usefulness.

To decide on the best from a collection of different regression models, the first metric to look out for is the root mean squared error, where lower values signify lower mistakes the model makes during prediction. However, in practice, this process may not pick the best model. Model building is a trade-off between the model's error measure and other properties. Among these properties, the following are commonly considered:

Simplicity (use fewer terms). Suppose x is a vector of the component fractions. A model may involve a polynomial x with n terms (the terms may include powers of each component and the cross products of two or more components). Among multiple potential polynomials, we may favor one with fewer terms (small n), even if with fewer terms, the sum of squared errors of the best-fit model becomes larger. This is to prevent overfitting. In principle, if the improvement in the sum of squared error by adding more terms is not significant, one may want to stop and choose a simpler model (fewer terms). A comparison among different model variants using cross-validation is needed to make this decision.

Smoothness (no oscillation). In many cases, the viscosity value is monotonic within a range of temperature, pressure, and composition. The viscosity value may increase or decrease in the range, but we don't expect it to oscillate (going up and down with maxima and minima). One may need to examine the fitted model to ensure it is the case. Our application will provide software components for performing such examinations through visualization.

Some previous efforts in online viscosity model building applications exist, such as the web viscosity calculator [20] and applications developed by Karki et al.[28]. The web viscosity calculator implements the viscosity model of Giordano et al. [18]. However, these applications are limited to a narrow $P - T - X$ range.

4.5. Regression Using Machine Learning

Machine learning is a data analysis method where computer models are improved with experience by iteratively going through each data point while minimizing errors computed against the expected value. Many successful applications such as recommendation

engines, outlier detection, self-driving vehicles, and countless others have shown their usefulness in today's world of high-performance computing. Machine learning is an amalgamation of fields like statistics, artificial intelligence, information theory, physical systems, etc. Due to advances in computation power and underlying theories of such automated systems, it has gained much popularity amongst researchers and practitioners.

4.5.1. Decision Trees

A decision tree (DT) is a supervised machine learning algorithm used in both classification and regression settings. The construction of DTs is done by splitting training data with respect to dimension at each tree node. Each node represents a test on the training data features, and its branches denote the output of the test. For instance, if we take temperature as a test feature, the decision tree might split all data into two parts: $T \leq 4000$ K and $T > 4000$ K. Each data split is further split until no further splits add value to the prediction accuracy. Each split makes the subset of the data homogeneous. After construction is completed, each leaf node will hold a single or collection of data points which are then used to either output a data class (in classification) or singular continuous value (in regression).

DTs for classification use information gain and entropy to split nodes. The target class is predicted by following a new data point through the tree until the leaf node. The class at the leaf node is output as the class prediction for the new data point. In the regression setting, DTs are generally constructed by reducing the mean square error of the predicted and actual target values. To predict the output of a new data point, we again follow the tree from the root to the leaf node. In contrast to the classification setting, the

aggregate of the target variable for all training data points is returned as the output prediction for regression.

Since the data are split recursively on the features at each level, the overall tree is easy to interpret. One can follow the splits through the depth of a trained DT to find out how a certain prediction was made. DTs can handle high-dimensional training features that may contain either numerical or categorical features. Further, DTs can be constructed with minimum pre-processing and without requiring domain knowledge of the data. A trained DT can also be used to analyze the feature importance of the data used during training.

DTs can be expensive to train for large data with many features since the algorithm needs to perform a top-down greedy search through all the possible data sub-spaces for each node. A fully grown DT also needs large memory to store. Overfitting is also an issue with large un-pruned DTs. Early-terminating the tree-growing process after it reaches a certain height may reduce overfitting of the model to training data. Additionally, the cardinality of data points in the leaf node can also be constrained to specific values to decrease the model complexity. Careful selection of these two parameters yields DTs that effectively learn from the training set while also being generalizable to unseen data.

Due to the inherent space-splitting nature of DTs, the model outputs are not smooth over its features. Since the target value output in regression DTs is the aggregate of target values of the data points in the leaf node, the output with respect to features may contain plateaus with large jumps between them. Therefore, this learning model cannot precisely assess the target variable value. This model can be used for exploratory

data analysis and pinpoint feature importances. Further, DTs can also serve as a baseline for comparison with other machine learning model results.

4.5.2. Random Forest

Random forest (RF) is an ensemble technique that uses several DTs to predict the output class or single value. It is a supervised machine learning algorithm where many trees are constructed, and the prediction is made by averaging the output of each tree for a given data point. Due to the simplicity of DTs, RFs are similarly easy to construct and interpret. Just like DTs, RFs can be used for both classification and regression.

RF is a powerful machine learning technique that can be used for various regression tasks. The algorithm can handle a large number of features and can also deal with non-linear relationships between features and targets. RFs can be especially helpful for small datasets since large data (and features) requires large DTs to model the data space. Model accuracy can be improved by tuning the hyperparameters, such as the number of trees in the forest, the number of features to use when bootstrapping, and the depth of each tree.

RF is relatively easy to construct and interpret and can provide good accuracy without extensive hyperparameter tuning. It can also handle missing data and is resistant to overfitting. Since we can look at the output predicted by each tree in the forest, they can be inspected, and the importance of each feature can be determined. Similar to DTs, this algorithm is a helpful tool for data exploration and understanding the relationships between features and targets.

4.5.3. XGBoost

XGBoost (XG) [12] implements gradient-boosted DTs designed to be highly efficient, scalable, and portable. Gradient Boosted Trees (GBT) are a type of ensemble machine learning algorithm that is used in both regression and classification problems. The algorithm is used sequentially, where each subsequent tree is trained on the residuals (errors) of the previous trees. The algorithm stops when the residuals are relatively small, or a pre-determined number of trees have been generated. GBTs can be used with different loss functions, like DTs and RFs, and they are also easy to interpret.

XG is an easy-to-use API that can handle large datasets with low training time. It is widely used in competitive data science, where it is shown to outperform many machine learning algorithms on various tasks, including regression. Since it is also a tree-based algorithm, the predictions contain plateaus with jumps like RFs or DTs on regression. RF and XG are ensemble techniques. However, RF is bagging, where independent tree outputs are averaged together, whereas XG is a boosting technique where trees are constructed sequentially. The first few trees during the training of XGBoost contain small DTs, which are refined further using subsequent trees. Therefore, XG predicts the target better than RFs since the algorithm constructs trees based on the residuals. Like other tree-based approaches, overfitting is an issue with XG, especially since it can create complex decision boundaries and relationships.

4.5.4. Artificial Neural Networks

Artificial neural networks are computational models inspired by the brain that approximate complex non-linearly separable functions. They are composed of a large number

of interconnected processing nodes that can be configured to perform specific tasks. Due to their flexibility, neural networks can be used in classification, regression, clustering, and data generation settings. The neural network is an algorithm under the subset of machine learning where the model is first trained on training data and evaluated on test (unseen) data.

A simple neural network (or multi-layer perceptron) contains groups of perceptrons arranged in one or more layers. Perceptrons are the unit of computation that accepts input data from either the input layer or other perceptrons from previous layers. Each input of the perceptron is multiplied by its weight. The sum of all incoming input values multiplied by their respective weights is then passed through an activation function. Activation function such as sigmoid or ReLU is the final output of the perceptron. Each perceptron's output for a layer is fed as an input to the next layer. The final layer perceptron typically does not contain any activation function for single output regression tasks. Therefore, the output of the final layer is directly taken as the target value given a particular input to the neural network.

The architecture of a simple neural network refers to the number of layers and the number of perceptrons in each layer. For instance, a 3-layer neural network can model XOR boolean expression. The input layer contains two nodes for each boolean variable. The hidden layer contains two nodes, and finally, the output layer contains one node, the network's output. This architecture can sufficiently produce the correct output for simple problems. However, modeling more complicated data and relationships requires larger or different architecture. Larger neural networks can have multiple hidden layers with more nodes in each layer. Convolutional neural networks and transformers have been shown

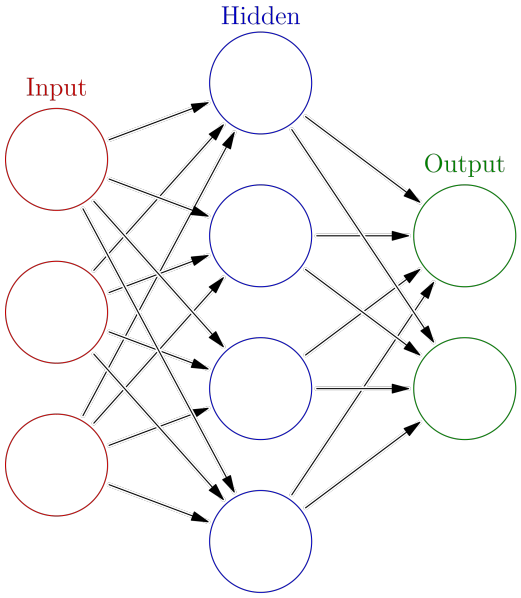


Figure 4.4. A simple neural network with three layers - input, hidden, and output. Each input, hidden, and output layers contain three, four, and two nodes, respectively. Each node from the previous layer is connected to every node in the next layer through weights.

to work more effectively to work with more complex data such as images and texts. Regardless of the architecture, the weights and biases are initialized and iteratively updated during the training process.

Typically the weights and biases of a neural network are initialized randomly within a certain range of values. Other initialization techniques include Xavier [19] and normalized Xavier. Training of a neural network is referred to as iteratively adjusting each weight and bias in the network such that the final output is close to the desired target value for a given input. A loss function such as RMSE (in regression) calculates the distance between model output and target value. Choosing the correct loss function that accurately represents the distance between actual and predicted values is crucial since this loss is used to adjust weights and biases of the neural network using the backpropagation algorithm. This algorithm calculates the derivative of the loss function for a given input

and the current weights and biases of the network. The learning rate hyper-parameter controls the magnitude of updates made to the weights and biases. A high learning rate may shorten training time, but it could also miss the minima where loss is the lowest. In contrast, training a network with a low learning rate might take a long time to complete, and with enough time, it will at least settle on local minima. The learning rate is usually kept constant throughout training, but sometimes schedulers are used to change it in complex ways as training progresses. As the network goes through each training data multiple times, the weights are updated iteratively until the changes to weights and biases with respect to the loss become minuscule or zero. At this point, the network is trained and can be used to assess its performance on unseen (test) data.

According to the universal approximation theorem, neural networks can approximate any continuous function to any desired accuracy, provided that the number of neurons in the hidden layer is sufficiently large. This theorem also holds for an arbitrarily large depth with a relatively small number of neurons in each layer. Therefore, neural networks are prone to overfitting the training data. During training, if the test error increases while the training error keeps decreasing, we can conclude that the model is more sensitive toward training data. Generally, we want the model to be robust for both seen and unseen data. To get this balance, we can stop training at the lowest test error or employ regularization, dropout, or weight decay during training. These methods help neural networks generalize while learning an adequate amount from the training data such that the model works for unseen data.

Chapter 5. Implementation Details

The overarching goal of this work is to develop a web-based data analysis application for materials physical properties of materials by using silicate melt viscosity as an example. Users from the geoscience field and, in general, diverse materials science communities may be interested in exploring these data. This work also describes a non-standard PCP implementation as a part of a web-based data analysis platform. As such, the web application was developed using a client-server model. This application can be accessed using any standard web browser. In this chapter[1], we describe different approaches and tools used to implement the three components of the system - database, web application, and modeling.

5.1. Database

The central component of our data-sharing platform is the database. The design of the database is done in such a way as to facilitate adding datapoints and providing fine-level search and selection of the data points. To provide this feature, we explain our data storage in tabular form using a relational database model.

The central table in our database is the *DataPoint* table. *DataPoint* contains information on each data point such as temperature, pressure, viscosity, meta-data, composition, and uncertainties. Since we are using a tabular representation of data, we can minimize data redundancy. For instance, the data provided by [24] consisted of a CSV file with a column of id, composition name, 11 columns of different components weight for each composition, temperature, calculated and measured viscosity, and reference. Al-

[1]Some parts of this chapter can be found in D. Bhattarai., J. Zhang., and B. B. Karki. Parallel coordinates-based visual analytics for materials property. In Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - IVAPP, pages 83–95. INSTICC, SciTePress, 2019. Copyright permission included in Appendix A.

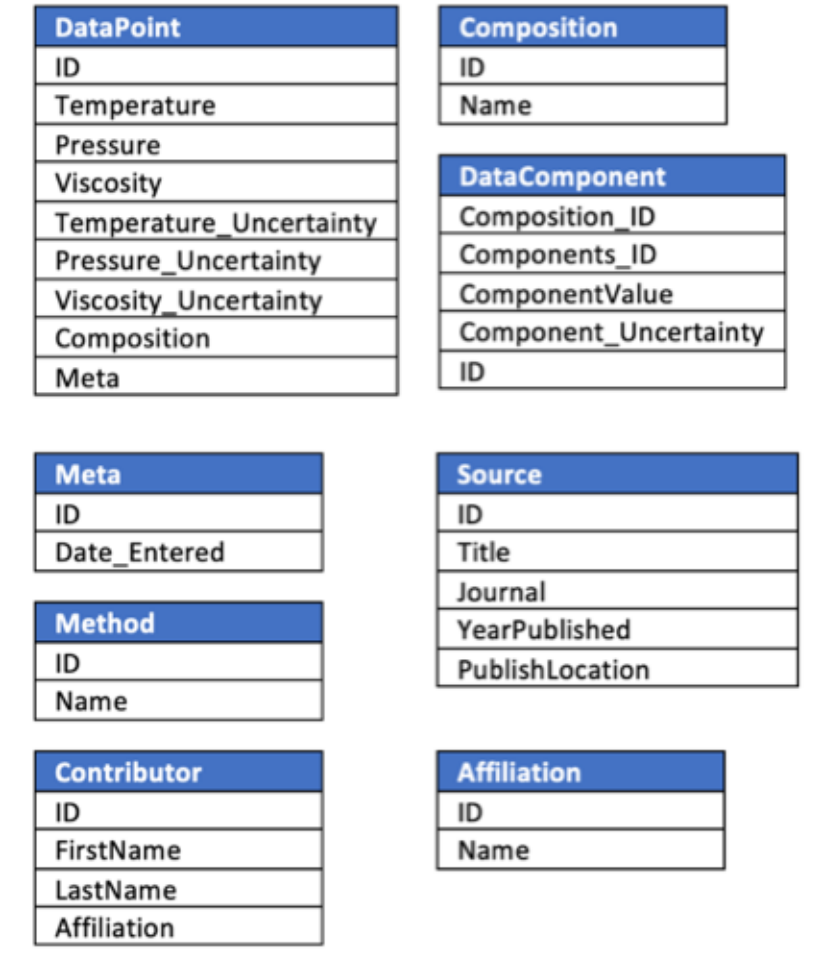


Figure 5.1. Database schema showing the central data tables [6].

though all data rows were unique, several pieces of data within them were repeated. One example would be for each composition there would be multiple measurements of temperature and viscosity. The columns corresponding to the weights of each component for a composition would repeat for different $P - T$ measurements. This is redundant and should be removed as much as possible. Removing redundancy keeps our database clean and improves data integrity. Therefore, we use a foreign key to *Composition* table in *DataPoint*'s *Composition* field. This way we remove the need to store repeated information in *DataPoint* table which would have made it unnecessarily large as the number of data points grows over time. In Figure 5.1 we can see the relation between *DataPoint* and *Composition* tables. Then, *DataComponent* table contains a value for each specific component along with uncertainties, and a foreign key to *Components* table which contains the name and ID of each component in a composition. Since *DataComponent* also contains a foreign key to *Composition*, each data point can be uniquely joined together when querying for specific data points. Similar to the relation with *Composition*, *DataPoint* only stores the value of the foreign key to the *Meta* table. *Meta* is related to other tables such as *Contributor*, *Source*, and *Method* which stores the contributor information, the source of publication, and the method of data collection. Any user information in the system can be saved in *User* table. This table contains only primary user information. Other user credentials such as password and login information are stored in another set of tables. On top of these tables, many junction tables like *DataComponent*, *MetaSource*, and others have also been designed to correctly map foreign key relationships.

Detailed information for each data point can be collected by joining together tables that are related to the data point of interest. Using this database model, we can also en-

able efficient fine-grain selection. For example, among viscosity data, one can search and obtain viscosity measures for a certain range of temperature and pressure, that are from simulations and are contributed by a certain research group.

Our database aims to compile all viscosity results available from various sources. These data will come from multiple research groups that are of both experimental and computational nature. As an example, a researcher may compare the viscosity of the same melt from different sources as well as methods of data collection. The modeling assessment will examine the data's consistency and smoothness by plotting viscosity as a function of pressure, temperature, and composition. When any inconsistencies are found, they are noted in the *Comments* and respective uncertainty fields in each database table. Note that this step does not change the value of the original data, but rather gives an opportunity to save uncertainties when the data is being used. Such decisions to include uncertainties in the data will be based on various considerations. An example could be if a group of data only reported melt composition as anorthite without reporting the actual analyses of the composition, a larger error would be assigned for the oxide concentrations or for viscosity. These assessments would help maintain consistency in the database and improve the chance of obtaining the best viscosity model.

This database model has an efficient structure to manage data. However, it is not intuitive for the end users. Therefore, an easy interface must be provided. Today, the most popular interface to interact with remote data is the web on the Internet. We have built a web application to facilitate researchers with an intuitive interface to search, select, and analyze data along with the ability to compare and share different models. Below we discuss in detail regarding the components of the web application currently in use today.

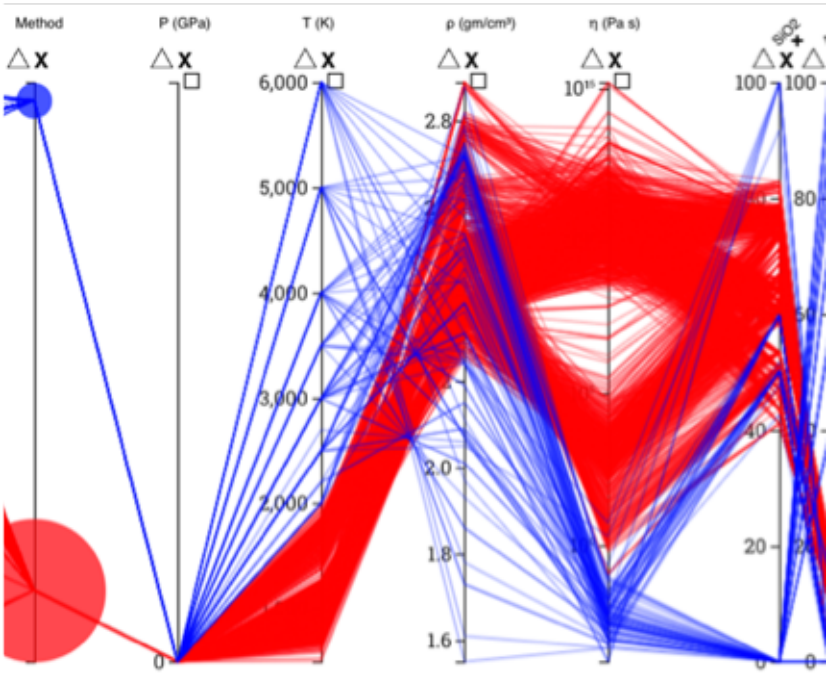


Figure 5.2. PCP showing two melt properties: viscosity (η) and density (ρ) [6].

Other material properties can be incorporated as new attributes and corresponding new axes in the plot as long as they are defined in the same parameter ($P - T - X$) space. For instance, to visualize the melt density ρ, the density value and uncertainty entries are made in the "DataPoint" table (Figure 5.3). The density data at zero pressure are estimated using the density model for multi-component melt systems. The density (ρ) axis appears in the PCP (Figure 5.2). The data for the computational category spreads more than those for the experimental category. Density takes smaller values at higher temperatures. The data lines between the density and viscosity axes show a positive correlation – the higher density, the higher viscosity.

5.2. Web Application

Once deployed to a public server, the web application can be accessed via any web browser. Our system has been developed using Angular, a popular web application frame-

work. The decision to utilize this framework was because of its stability which is backed by large open-source contributions. A simple API on the back end provides data and filtering access. We used D3 [7] for plotting on the front end. All these components will help us create the web application with the following functions:

Web Access to the Database. The primary purpose of the web application is to provide users convenient access to the data residing on the database. This application will allow users to explore and download data. Users will also have the capability to browse data with multiple fine-grain selection criteria.

Model Analysis, Visualization, and Comparison. After constructing single or several models, our application will also provide tools for calculation and plotting the model results. For the use case of viscosity, we will make several existing models available in the system. By using our application, researchers can easily:

- Calculate viscosity values following single or multiple models. This is particularly helpful for users interested in calculating viscosity for a specific temperature, pressure, and composition with multiple models. Similar work in the form of a web calculator [20] based on the multi-component viscosity model has been one of the primary works on the online platform.

- Perform visualization of the entire parametric space the model was optimized on. The current implementation contains Parallel Coordinates with interactive chart actions.

- Download and share model results.

DataPoint
ID
Temperature
Pressure
Viscosity
Density
Temperature_Uncertainty
Pressure_Uncertainty
Viscosity_Uncertainty
Density_Uncertainty
Composition
Meta

Figure 5.3. DataPoint table with added material property Density [6].

In order to implement these requirements as a cloud framework, we opted to choose a client-server model which consists of two remotely connected parts of the application - client and server side. The server side is what resides on the remote server and is responsible for facilitating data and responding to queries from clients. Client-side on the other hand runs on the user's computer and focuses on simple calculations and displaying data returned in response to queries from the server.

5.3. Modeling

We used Python and several packages to model our data in this study. These packages include PyTorch [52], sci-kit-learn [10], and NumPy [21]. Specifically, we trained tree-based models on the CPU using sci-kit-learn. On the other hand, we used PyTorch to run training and inference on the GPU for neural networks. To handle data storage, load, and transformations, we used sci-kit-learn and NumPy libraries. Finally, our web application uses Python to retrieve trained model predictions and sends them back to the requesting client on the server side.

Chapter 6. Results and Analysis

In this chapter[1], we describe analysis and modeling results using various datasets.

6.1. Visual Analytics

This section presents the details of our visual data analysis of the silicate melt viscosity database. Viewing the viscosity axis using a logarithmic scale, we realize that the viscosity values span the large range of orders of magnitudes: 10^{-4} to 10^{15}. We also notice that viscosity shows a bi-modal or tri-modal distribution (Figure 3.3). The outliers in the high viscosity region are from experimental sources and cover a narrow temperature range $<$ 1000 K and zero pressure. On the other hand, outliers in the low-viscosity region are mainly from computational sources at high temperatures (4000 - 6000 K) in the low-pressure regime (0 - 20 GPa).

The polylines across all axes are colored red or blue to represent the methodology categories (experimental or computational). On the *Method* axis, we see that the red circle is much larger than the blue circle Figure 3.3 (and Figure 3.6), which means that the majority of data are the measured values. Looking closely, we can also find that almost all experimental data are at ambient pressure (0 GPa) and low temperatures ($<$ 2000 K). In Figure 3.6, two groups can be seen in the viscosity axis for the experimental data. One group is characterized by super-high viscosity and low temperature, and the other is characterized by high viscosity and sub-low temperature.

The experimental data in our database come from publications as old as 1965. The metadata axis *Year* shows a steady increase in the experimental data since the 1990s (Fig-

[1]Some parts of this chapter can be found in D. Bhattarai., J. Zhang., and B. B. Karki. Parallel coordinates-based visual analytics for materials property. In Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - IVAPP, pages 83–95. INSTICC, SciTePress, 2019. Copyright permission included in Appendix A.

ure 3.3). The computational data, on the other hand, are relatively new and are available only from 2010 onwards. The calculated data cover a much wider range of temperature (2000 - 6000 K) and pressure (0 to over 150 GPa), but the majority are in the low-pressure regime. Even though we have not included all experimental data available from recent years, it seems that the broad ranges of temperature and pressure data for silicate melts were previously unattainable from just experimental sources.

The composition of magma includes several oxides along with volatiles H_2O and CO_2. The computational results offer a full range of MgO, SiO_2, and Al_2O_3 contents (0 to 100 wt% or mol% for each). Experimentally studied compositions fall in the silica range of 40 to 80 wt%. Pure silica or silica-rich melts tend to be highly viscous, as suggested by calculated data in the low pressure-low temperature regime. The SiO_2 as network former makes the melt highly polymerized and highly viscous. However, MgO as a structure modifier lowers the melt viscosity. Small amounts of volatiles can cause significant changes in melt viscosity. These dependencies can be observed by brushing along the merged dimension axis with controlled temperature and pressure ranges. The PCP method is considered to be highly effective in visually judging correlations between dimensions that are mapped to the adjacent axes. A negative correlation between the viscosity and temperature has manifested as data lines crossing each other. This can be further enhanced by constraining pressure and composition. Selecting the computational category at zero pressure (blue data lines), we find that the silica and MgO end members show the strongest and weakest negative correlation, respectively. Upon selecting experimental data by clicking the red circle in Figure 3.6, we can see that the viscosity varies more than six orders of magnitude for a relatively small temperature change, showing a negative correlation with each other.

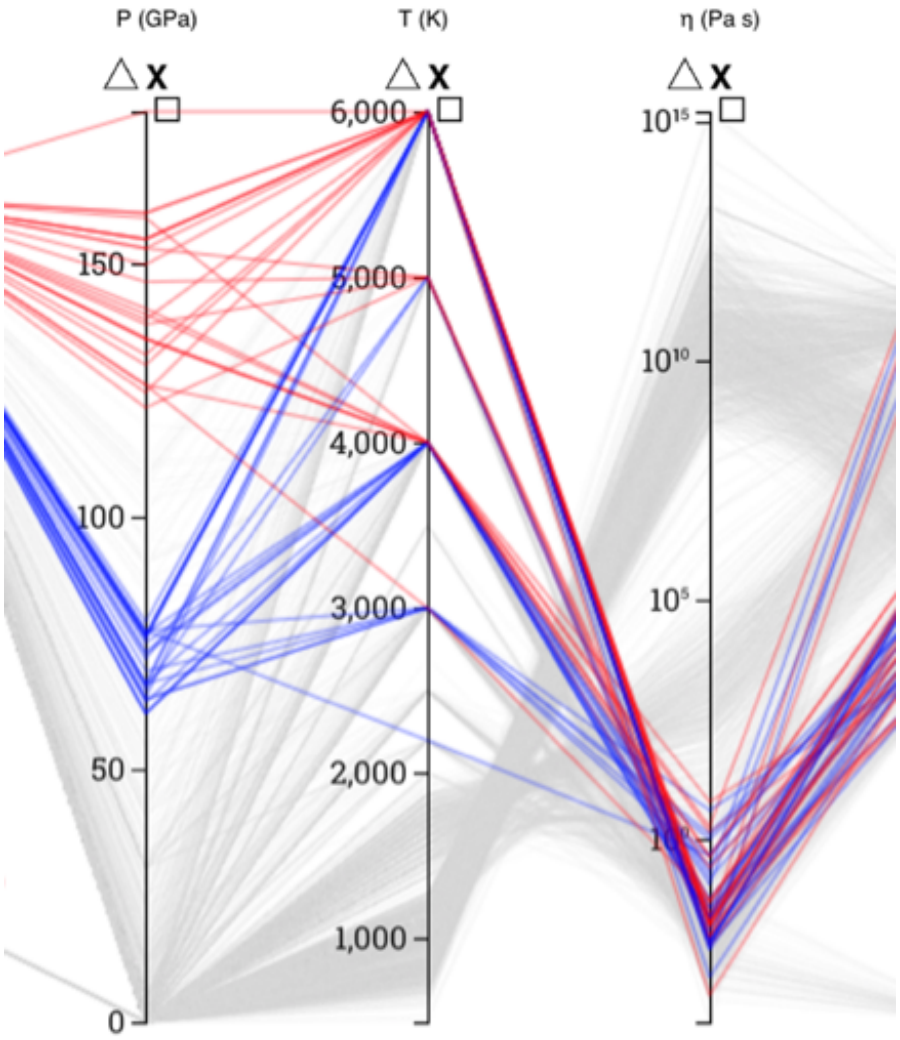


Figure 6.1. Two pressure regimes 60-80 GPa (blue) and above 120 GPa (red) are highlighted. Even with a wide high-pressure regime, the viscosity remains almost in the same region as relatively low pressure [6].

The viscosity-inverse temperature relationship appears to hold at all pressures and compositions. PCP has successfully captured this fundamental nature. Viscosity depends on the pressure component in a complicated way, however. Brushing the pressure axis in the range of 60 – 80 GPa, we can see that several calculations were performed at 3000, 4000, and 6000 K (Figure 6.1). However, the viscosity value does not change as much when translating the brush towards higher pressure regime of 120 – 180 GPa. Here we can see several data points at 4000 and 6000 K. The viscosity value is still found to be in the same lower region. These steps hint that the viscosity of silicate melts changes much more at low temperatures and pressures but not so much in the high $P - T$ regime. Interestingly, silica-rich melts display an anomalous behavior at 3000 K in that the viscosity first decreases and then increases as pressure increases. This can be observed in the two nested plots for the viscosity and pressure axes corresponding to low and high-pressure regimes (Figure 6.2). The data lines cross each other at low pressures (negative correlation), whereas they run parallel at high pressures (positive correlation).

6.2. Modeling

The viscosity of silicate melts is a complex function with respect to $P - T - X$. The data landscape requires careful thought while modeling. For instance, a simple linear regression model performs poorly over the whole region since it cannot accurately model the nuances of different data regimes. Generally, if no analytical solution exists, physical systems are expressed in equations with coefficients optimized from data for accurate predictions. Building these sets of equations is a complex task, often involving data selection, pre-processing, and choosing a model form. Data pre-processing such as combining oxides

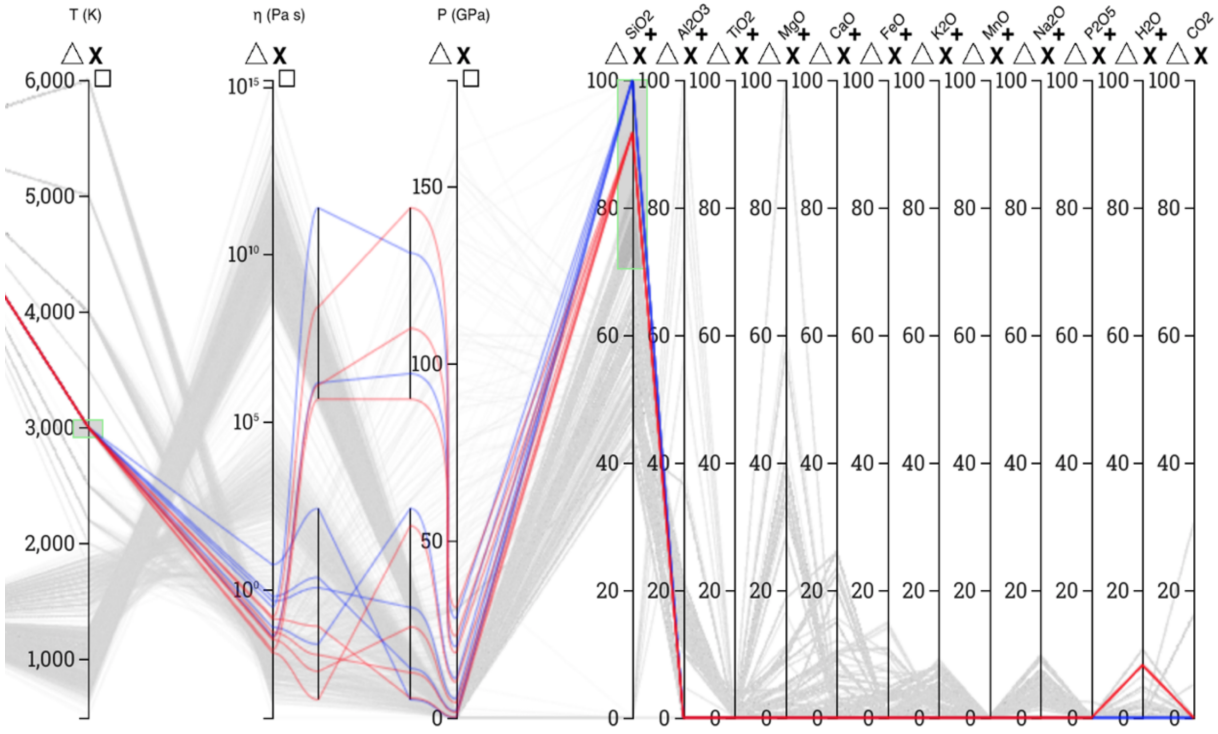


Figure 6.2. Data selection of silica-rich compounds at 3000 K showing anomalous behavior at low pressures below 15 GPa (lower nested plot) and normal behavior at higher pressures (upper nested plot). Blue and red represent pure and hydrous silica liquids, respectively [6].

into a new property, assigning exponential value for water content to amplify its effects, and others are usually performed to guide model accuracy better. These methods successfully model specific data regimes, but none are generalized over all possible parametric space. The main reasons for this are the lack of enough data throughout the region and the complex relationship of viscosity with respect to its parameters.

Model overfitting is also another concern. An overfitted model memorizes data space, producing accurate results for the training dataset but failing on unseen data. These models are not useful since they do not generalize well for data not used during optimization. Therefore, an ideal model should be able to generalize and not memorize over data space that it was trained on, is continuous, requires as few parameters as possible, and would be explainable. Physical models are generally continuous over specific regimes and have required only a few meaningful parameters. These models have worked quite well for different data regimes but do not generalize well for the whole data region. Designing such models require careful search over large hypothesis space, which might not always be feasible. Further, using multiple models may lose continuity at the joins on model boundaries.

Apart from describing the physical system manually, we can also use machine learning techniques to model data. The system is not explicitly defined or constrained through equations. Therefore, these models can explore larger hypothesis space. Since we have limited data, model optimization does not take a long time (less than 1 minute per model). But limited data also makes it difficult for the model to generalize over the intended data space. We understand this trade-off and perform experiments with models such as - tree-based, tree-based ensembles, and neural networks. Below we present an

analysis of models trained on subsets with data compared against baselines from previous works.

Train-test Partition. Machine learning algorithms require extensive testing to prevent both overfitting and underfitting. In the work of Hui et al. [24] (HZ), all available data points were used to optimize their model form coefficient. This model form was chosen according to the physical properties of silicate melts. The optimized model was tested against another set of relatively fewer data points not present during optimization. In the case of machine learning models, we have to partition all available data into train and test sets (we used 80-20% split) to properly assess model performance. We use a relatively large train set because machine learning models are not constrained by physical properties. Partitioning can be done where a certain percentage of data randomly goes to the training set and the rest to the testing set. A better partition strategy would be to partition data randomly per composition. This way, all compositions are represented in both the train and test sets.

Data Transformation. Since all the features and target variables are numerical quantities, only a few data transformation steps are required. We might notice that the components of each composition are already scaled from 0-1 (in mole fraction). On the other hand, the temperature values are orders of magnitude larger than the component values. Therefore, all train and test data features are scaled by standard scaler fit on the training set. Our target variable is the $\log_{10}$ viscosity values. Taking log compresses a large range of real viscosity values. This is already favorable for the learning models since the values they have to predict span a relatively smaller range since the exponential relationship of viscosity with respect to its parameters is reduced to a linear relationship.

Therefore, no transformation is performed on the viscosity values.

Learning Models. For this study, we use tree-based regression models such as Decision Tree Regressor (DT), Random Forest (RF), and XGBoost (XG). We also include a neural network model with two hidden layers and each layer containing 64 nodes with a ReLU activation function. The first layer takes all input features ($n = 12$). The last layer contains a single node with no activation function. This node predicts viscosity values in the $\log_{10}$ scale. We double the nodes and the number of hidden neural network layers for the complete dataset.

Hyper-parameter Search. Grid search with five-fold cross-validation was used to find optimal hyper-parameters for all tree-based models. The search spanned multiple depths (DT, RF, XG), learning rate (XG), as well as the number of estimators (XG). The best-found model was then trained with all training data. For the neural network, we fix the architecture and optimize its weights using train data. We train the neural network for at least 5000 epochs for each experiment using a constant learning rate of 0.00001 and AdamW optimizer with 1e-5 weight decay. Batch size 256 was a good point between model performance and training time. For the complete dataset, we use a larger neural network with batch size, learning rate, and weight decay of 1024, 0.0004, and 0.02, respectively. We follow [24] and use root means square error (RMSE) as our loss function. Below we also include a study on how changing the network architecture impacts overall model performance.

6.2.1. Cross-validation Study

Randomly partitioning data into different test and train sets to train various models is one of the first steps in judging model performance. However, this yields another problem of having either optimistic or pessimistic results depending on the partition. For instance, we might get optimum results with test data having only a subset of the distribution that the model was trained on. Perhaps the test partition did not contain data with complex behaviors. On the other hand, the training set might not have enough data to be adequately trained, which may result in less generalizability. This poses challenges in partitioning the data so that we can train models effectively and test the trained model.

One such way to examine this variability is using cross-validation, which is often used in machine learning and statistics. With cross-validation, we partition the data into multiple partitions of various sizes and test model result variance with each partition. One thing to note here is that cross-validation aims not to find the best model but to understand how well a model performs with unseen data. This method is computationally intensive as it requires training several hundred models. But in the end, it allows us to see how different partitions affect the model's performance. After we perform cross-validation, we can have at least an understanding of the model performance on unseen data. This step is crucial, especially when we do not have a large data to train on. Further, this allows us to observe model performance with different train and test distributions. Ideally, the train and test data distribution should be similar to end up on well-generalized models, but picking and judging that a particular partition is generalizable even for unseen data is a difficult task.

Cross-validation comes in different flavors. For instance, k-fold cross-validations where whole available data is partitioned into k-folds, trained on $(k-1)$ folds, and evaluated on the remaining k^{th} fold. For instance, if we pick $k = 5$, a model is trained on a total of 1^{st}, 2^{nd}, 3^{rd}, and 4^{th} partitions and evaluated on the 5^{th} partition. Partition 1 through 4 becomes our training data, and the 5^{th} partition becomes our testing data. However, what if the 3^{rd} partition (or any other partition for that matter) was more representative of test data? Therefore, we re-initialize the model, train it with the 1^{st}, 2^{nd}, 4^{th}, and 5^{th} partitions, and test it against the 3^{rd} partition. Therefore, for each value of k, we partition the data into $k-1$ for training and 1 for testing. We repeat this k times and report the mean, minimum, and maximum errors for each value of k. But this poses another question, what would be a good value of k? Generally, in practice, k is chosen to be 3, 5, or 10. However, we cannot rely on an arbitrary value of k; therefore, in this work, we have looked at k from 2 to 29 and observed the variance in the results with each k.

Furthermore, we also need to be sure which learning model might produce the smallest variance amongst all the folds for each k. Therefore, for each fold, we train four different models. Specifically, we trained Linear Regression, Decision Tree Regressor, XGBoost, and neural network. Next, we perform a grid search for each model for each fold of k to get the best-configured model for the specified training set. Finally, we kept the neural network with two hidden layers containing 64 nodes in each layer and optimized using AdamW optimizer with 1e-4 learning rate and 1e-5 weight decay, each trained for 5000 epochs.

Our approach to finding a good model requires good generalizability on each fold of the data partition. We iteratively go through different values of k, and for its every fold,

we reset the model and train them.

```
procedure EvaluateModels(train, test)
    allResults ← dict()
    k ← 2
    regressors ← [Linear, DT, XG, RF, NeuralNetwork]
    while k < 30 do
        folds ← Perform k partitions
        for train, test in folds do
            for regressor in regressors do
                Initialize regressor
                Search optimum parameters with subset of train
                Train regressor with tuned hyper-parameters using full train
                Record test, train RMSE in allResults
            Initialize neural network
            Train network on train
            Record test, train RMSE in allResults
        k ← k + 1
    return allResults
end procedure
```

Algorithm 2: Nested cross-validation to record train and test errors per-fold per-k.

Algorithm 2 shows a general procedure to record train and test errors for different partitions defined by k. For each value of k, there will be k different folds of data. To find their optimal parameters, we independently optimize three regressors - LinearRegressor, DecisionTreeRegressor, and XGBoost. The search for optimal parameters is also carried out using the best of 5 folds of inner cross-validation trials. Note that these inner cross-validation folds for model optimizations are performed using only the train data. Therefore, the optimization process is unaware of the test data defined by the fold in the outer cross-validation. Once we get the optimal model, we retrain it with complete training data of the current fold. Then we evaluate both test and train RMSE and record it in a dictionary variable. At the end of this procedure, we get RMSE for all four optimal regressors per fold per k.

After evaluating 28 different values of k, each k representing the number of partitions, we can now observe how the models perform on test data for each fold. The goal of the analysis now is to see with various partitions how much the models are generalizing. Below, Figure 6.3, 6.4, 6.5, and 6.6 are from the same graph except we sequentially turn off bad-performing models. For each value of k, we take the RMSE of each of its folds and report its mean (dots) and minimum and maximum deviations from the mean (error bars). This helps us to evaluate how much the model results fluctuate between the different folds of data.

In Figure 6.3, we can immediately notice that LinearRegressor performs badly with mean RMSE around $3\log_{10}$ Pa s. This is particularly bad because the actual viscosity value variation is quite large. Further, the variation is also generally increasing with k. On closer inspection, this model is significantly overestimating predictions. Therefore, this model is not practical for any predictability scenario.

In Figure 6.4, we display the results of models excluding LinearRegressors. These candidate models contain means at a modest place, but errors in test results are pretty different. We notice that, in particular, DecisionTreeRegressor has higher mean errors than the remaining models. The min and max errors are also quite large, with the model consistently underestimating at lower k. The mean error is between 0.4 and 0.6, which is relatively lower than LinearRegressors. The fluctuation in min and max errors does not provide confidence in having a good predictive capability.

This experiment's two remaining model candidates are XGBoost and ViscosityNet (neural network), as shown in Figure 6.5. While XGBoost's performance is significantly better than LinearRegressor or DecisionTreeRegressor, the mean error is still higher than

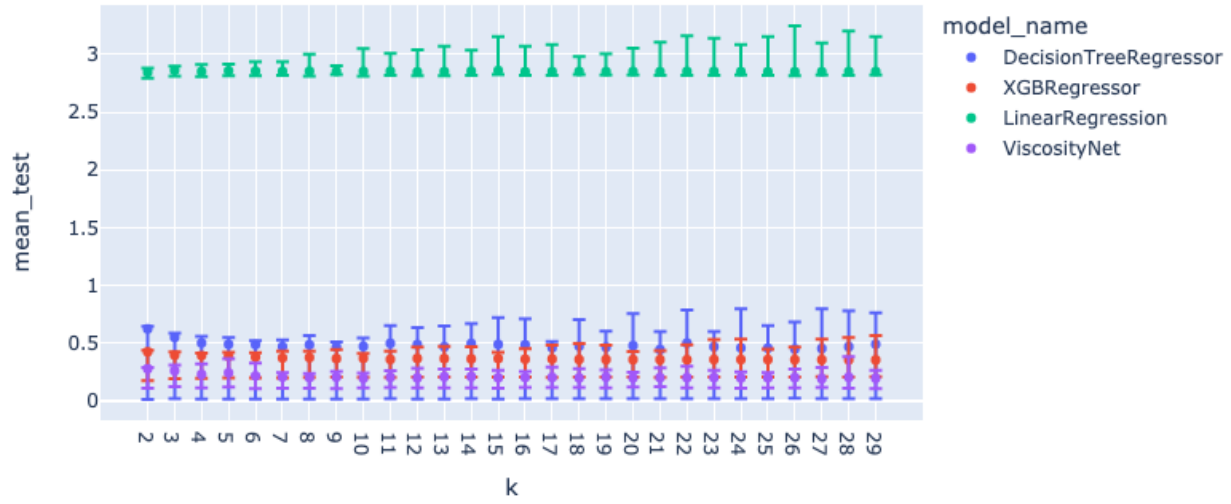


Figure 6.3. Test error mean (dots), minimum, and maximum (error bars) for each k is shown for four different regression models (less value represents better performance). The x-axis represents k-folds, and the y-axis represents errors in RMSE ($\log_{10}$). We observe that the linear model performs poorly, with errors in magnitude almost three times higher than other regressors.

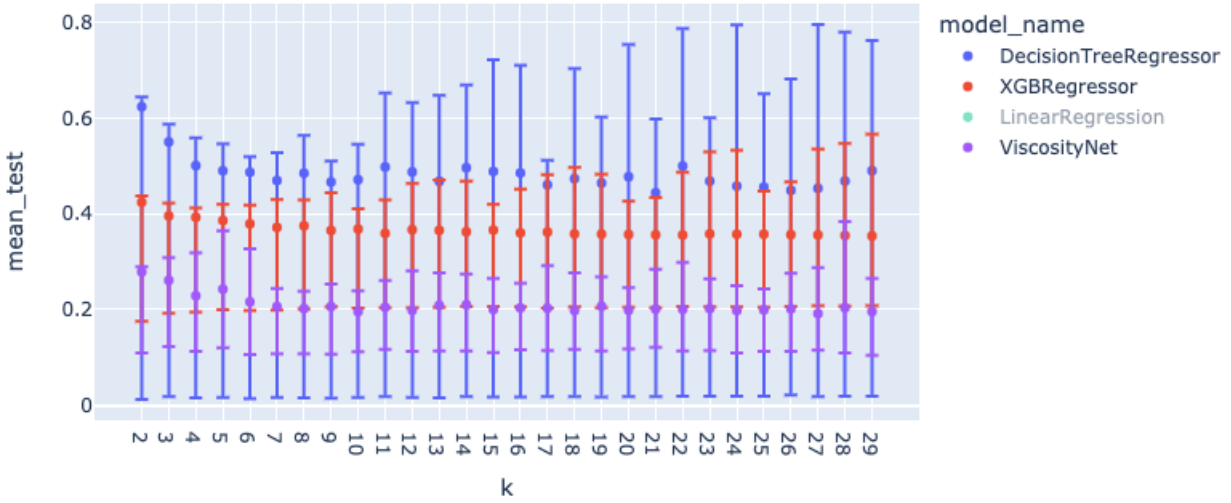


Figure 6.4. Test error plots without the LinearRegressor model give a much better view of competent models. Even amongst these models, the variance in test error for DecisionTreeRegressor is much higher than others.

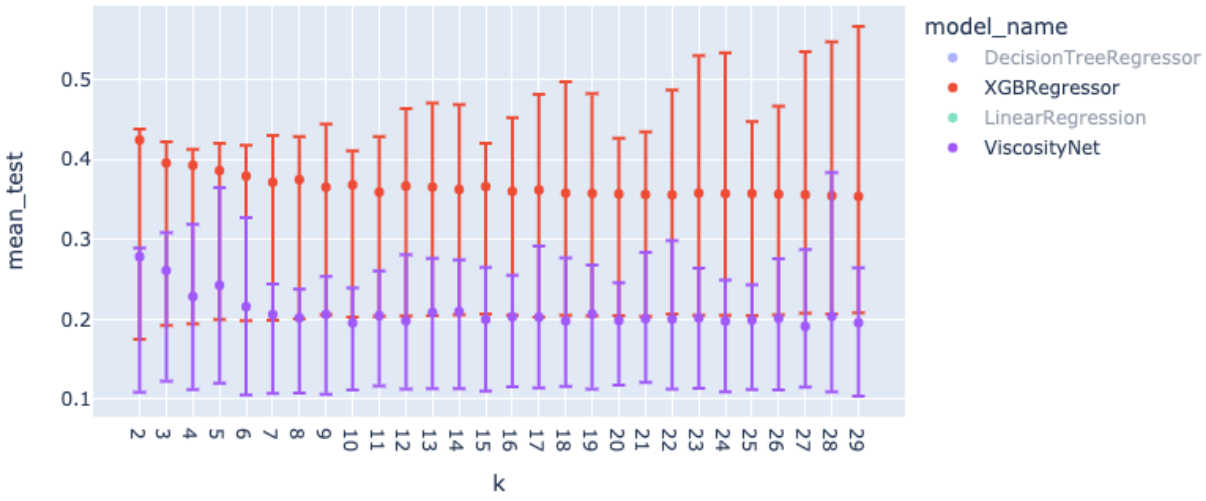


Figure 6.5. We only show errors and deviations of XGBRegressor and ViscosityNet (neural network). Neural network models achieve an overall lower test mean error for all folds.

the ones given by the neural network.

Overall, we observe that neural networks perform better than any models we compared against by judging the mean in test error and min and maximum deviations. For example, in Figure 6.6, we can see that the mean is around $0.2\log_{10}$ units for almost all folds (except for $k < 7$). We observe that at $k = 28$, the points contain the highest error. But apart from that, we can observe that the error deviations are mostly symmetric for almost all values of $k > 6$. This consistency and relatively fewer deviations of min and max error from mean per k suggest that it can generalize the data more than other models. This also suggests that the neural network can learn non-monotonic patterns well within the large parameter space of silicate melts.

It is also essential to ensure that the model is not just memorizing the training data (overfitting) but is generalizing the complex nature of melts. For example, neural networks are prone to overfitting partly due to the large parameter size. Other problems,

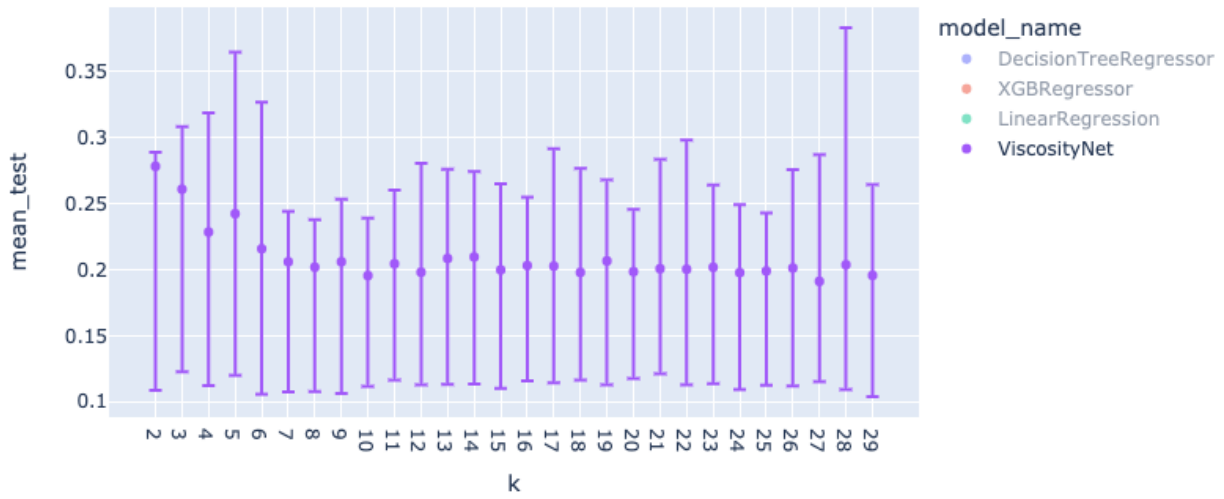
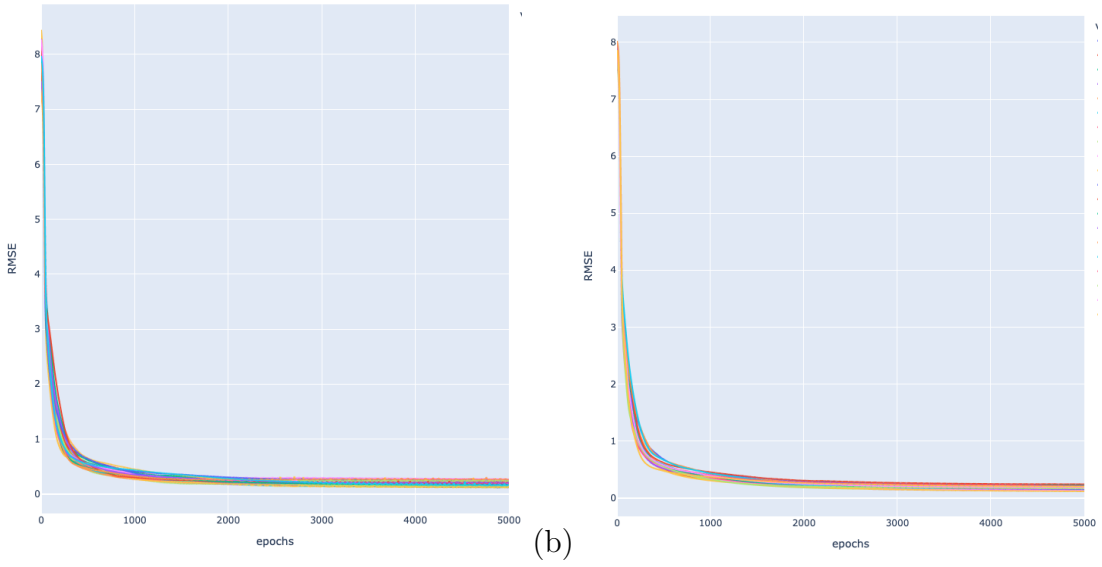


Figure 6.6. Neural network mean test error amongst all folds is generally around 0.2 ($\log_{10}$). Even with such diverse train and test data folds, the neural network can get a consistently lower mean for test data.

such as being locked into a local error minima during optimization, prevent it from being properly trained.

During the training process of neural networks, we update their weight and bias parameters by the overall loss. We perform backward propagation of errors for each batch to update the weights and biases. We iterate this process multiple times (commonly known as epochs) by letting our dataset through the same networks while gradually adjusting the values of weights. For example, we ran the same data multiple times (in this experiment, we chose 5000 epochs). Suppose the optimization process is not stuck at a local error minima region. In that case, the training error is expected to continue decreasing until it hits the global minima (if it ever can). Otherwise, we stop the training when an acceptable testing error is achieved. Finally, we expect training and testing errors to decrease or hover around some values on each epoch. This can also be another criterion to stop training because neither test nor train error will improve with more iterations.



(a) (b)

Figure 6.7. Test loss graph with cross-validation at $k = 28$ (left) and $k = 10$ (right).

In the case of neural network overfitting, the training error continues to decrease or plateau while the testing error suddenly increases. Generally, in this case, the training is stopped at the point of the lowest test error, even if the training error continues decreasing with more epochs. With overfitting, the model tries to memorize the training data reducing generalizability for unseen or test data. This behavior is akin to fitting a polynomial with a high degree for some scattered data points. For example, a higher-degree polynomial might be able to fit the data with low residuals. Still, it may not be necessary to go to such an extreme when a lower-degree polynomial can generalize well for unseen points during fitting. Therefore it is crucial to test our neural network and avoid overfitting.

For this reason, we perform tests where we plot (Figure 6.7) test errors from each fold per epoch during the training process. We do not observe any significant trend of overfitting. With each epoch, the test error continues to go down along with the training error.

6.2.2. HuiZhang (HZ) Dataset At Zero Pressure

The HZ dataset contains multi-component melt viscosity at varying temperatures and ambient pressure. These data points were published by various other researchers and were collected into a single data table [24]. The composition components in the original dataset are expressed in weight percentage, the temperature in K, and viscosity in $\log_{10}$ Pa s. Since all data points are at zero pressure, we do not include the pressure parameter during modeling. We reproduced the model by calculating viscosity values using the proposed equation and optimized coefficient from Hui et al. [24]. This equation takes multi-component composition in mole fraction and corresponding temperature as input producing viscosity value as output. We consider their published model and optimized coefficient as a baseline for comparing and selecting our models.

All learning models were trained using 1096 data points and analyzed for their performances on 355 test data points. We could observe that the neural network model yields the lowest test RMSE (Table 6.1). It is important to look at the train and test RMSE values to understand the model's predictive performance. In our case, the RMSE of XG on train data is quite low. However, for test data, it is slightly less performant when compared to the neural network. This means that the XG model is sensitive to training data (low train error) and cannot generalize well for unseen data (high test error). Contrasting this with our neural network model, we observed that both train and test loss go down as the number of epochs increases while training. This consequently means that the neural network model is learning training data distribution well while also being able to generalize for unseen data.

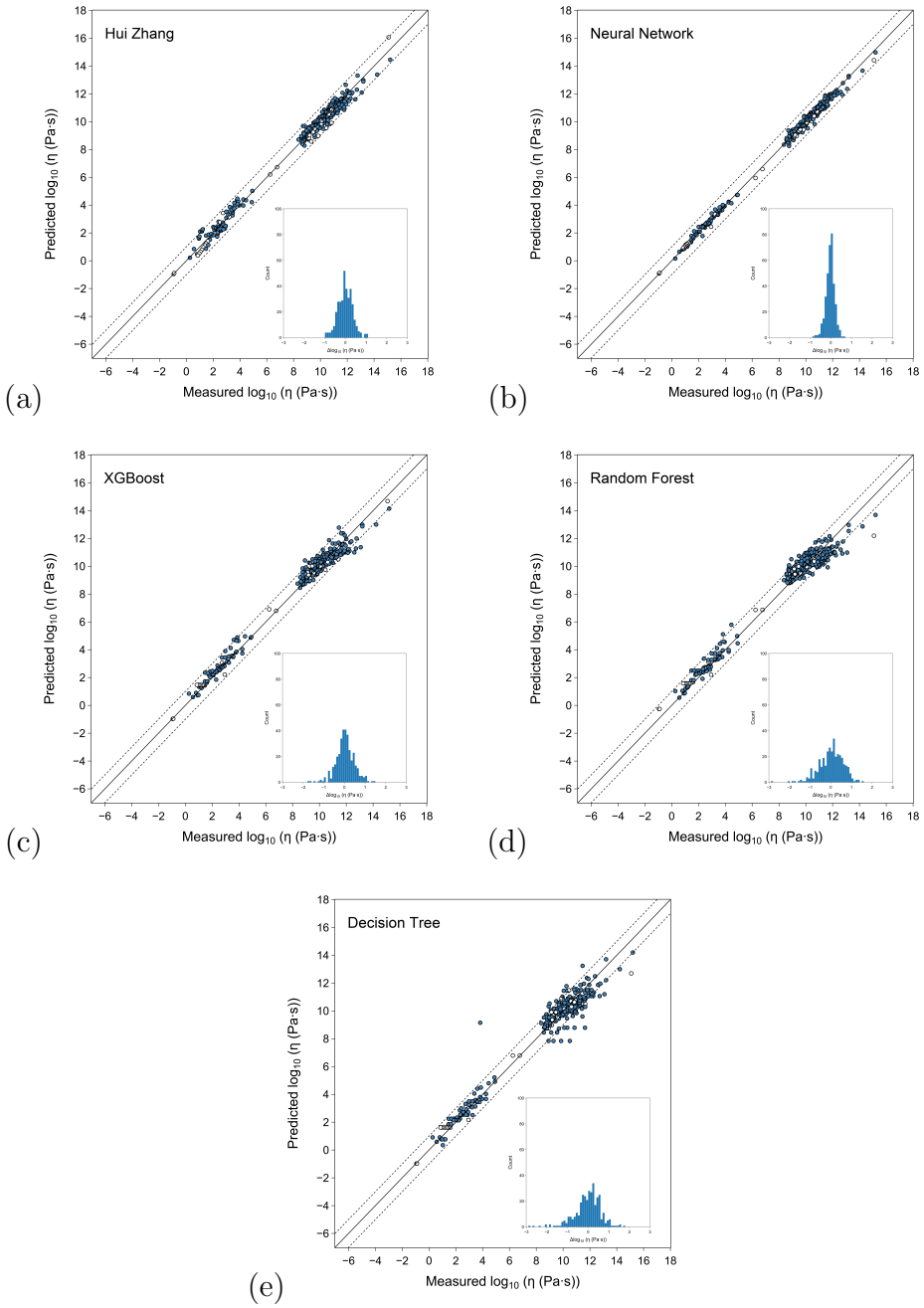


Figure 6.8. Model prediction scatter plots and error distributions produced by different models on HZ test data. Blue and white circles in scatter plots represent hydrous and anhydrous compositions, respectively.

Table 6.1. Train and test errors in RMSE produced by different models for the HZ dataset. The table also shows model prediction RMSE for the train-test data split using the equation and coefficients from the literature.

Model	Train	Test
Decision Tree	0.01	0.70
Random Forest	0.15	0.59
XGBoost	0.02	0.45
Neural Network	0.11	**0.22**
HuiZhang	0.28	0.36

To take a closer look at all the data lines along with model predictions, we plot measured vs. predicted values for the neural network as a scatter graph (Figure 6.8). The figures show $y = x$ perfect prediction solid line and ± 1 $\log_{10}$ units through dotted lines. We can see that most of the points fall close to the solid $y = x$ line for all models. HZ and neural network test model predictions stay within a log unit of actual values, whereas the points are more scattered for tree-based models. The error histogram on the same figure shows how an error is distributed across the viscosity range. Here, most data points lie in the 0 error region, a signified peak in the middle. However small, some data points also lie on the extremes of the error distribution. Compared to the HZ error distribution, the neural network's error distribution is narrower and contains fewer points at the extremes.

Similarly, compared with other tree-based models, the differences between extremes are much larger, with several points falling outside one log unit dotted lines. The worst case is for DT, where the model predictions are quite far from the actual value. An outlier test point is present in the scatter graph for DT. This datapoint contains composition with SiO_2, Al_2O_3, Na_2O, K_2O, and a small amount of water at 1571.15 K. This type of extreme outlier is not seen in other models' predictions.

For the neural network model, the data points that lie on the extremes of the er-

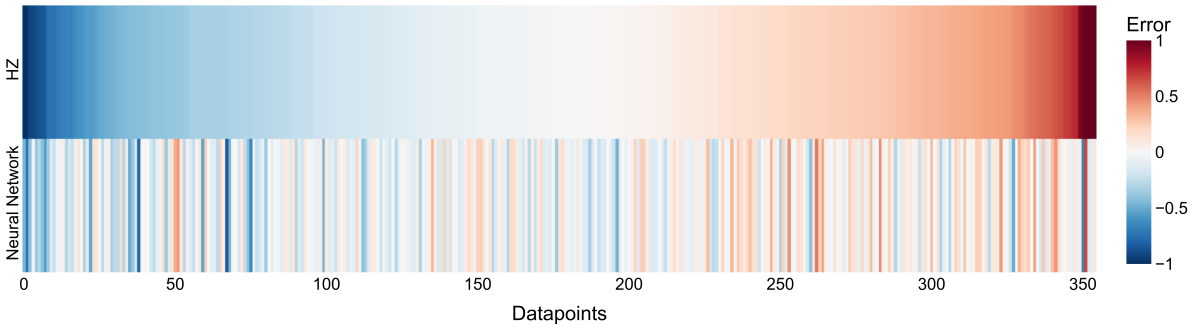


Figure 6.9. Heat map comparing prediction error by HZ and neural network sorted by HZ prediction error on the test dataset. Each data point is represented on the x-axis, and the y-axis shows errors produced by the models.

ror distribution plot (Figure 6.8) signify most underfit and overfit model predictions. The smallest and largest differences between neural network predictions and actual test points are -0.83 and $0.67\,\log_{10}$ Pa s, respectively. While the HZ model's smallest and largest errors (-0.99 and 1.07 $\log_{10}$ Pa s) extend beyond the neural network's differences. Since these errors are expressed in $\log_{10}$, it is important to note that an error of $1\,\log_{10}$ Pa s means that the actual viscosity values differ by order of magnitude. Data points with much larger errors exist on both positive and negative sides for HZ compared to the neural network. A test data point consisting of SiO_2, Al_2O_3, Na_2O, K_2O, and water at 524.95 K with the largest error ($0.67\,\log_{10}$ Pa s) produced by the neural network was also found to have produced a high error ($0.98\,\log_{10}$ Pa s) by the HZ model.

A heat map is used to compare predictive errors of HZ and neural network (Figure 6.9). Data in the heat map are first sorted by error produced by the HZ model and plotted alongside neural network error. Each vertical band in the plot represents a single data point, making it easier to compare model performance. We can pinpoint the data which produced the largest error on the right side (at data point 350 mark), indicated by red for

Table 6.2. Train and test errors in RMSE for different models for all available zero pressure data with temperature less than or equal to 4000 K. Numbers inside the parenthesis are RMSE after removing data points with large errors for HZ.

Model	Train	Test
Decision Tree	0.01	0.66
Random Forest	0.16	0.60
XGBoost	0.03	0.43
Neural Network	0.11	**0.25**
HuiZhang	144622.68(0.98)	565810.24(1.12)

both models. Both models have positive errors. Left of this data point, we can observe a flip in error compared to these two models. HZ produces a strong positive error, while the neural network model produces a strong negative error. Similarly, at around data point 50, we also have some points that produce inverse errors. All of these data fall below 1000 K. This temperature region of these data points lies near the crystallization interval inherent to the experimental procedures [56]. Since uniform data sampling is not possible in this interval region, a gap in the data can be noted in both the train and test datasets. The consistently low performance of all predictive models can be attributed to the lack of data on this composition (or near it) in the specific temperature region.

6.2.3. New Zero Pressure Data Along With HZ

In recent years, new data have been collected using both calculated and experimental methods. These data points were not included in the HZ dataset. Among these new data points, some calculated experiments are performed at very high temperatures ($T >$ 4000 K). These kinds of temperatures are rare to exist in an ambient pressure environment. Therefore, they have been filtered out. The remaining new data points were added to the HZ train and test datasets, totaling 1153 and 374, respectively. The modeling results are shown in Table 6.2.

For all machine learning models, the RMSE values follow a general trend where train errors are lower than test errors. The sensitivity of XG towards training data can again be observed here. The neural network model seems to have trained properly and produces the lowest test error among all models. The HZ model performs poorly, especially because it was not trained on the expanded data used in this experiment. This particularly high RMSE for HZ comes from a few data points of pure aluminum oxide at upper-temperature regimes ($T > 2000$ K). These data points were collected from experimental source [68]. The HZ model was not optimized for these data points, producing large errors for some of these data. After removing these data points, the model seems to have reasonable RMSE. However, the test RMSE is more than an order of magnitude, which makes it inferior to all other models.

The scatter plot in Figure 6.10(a) compares differences between predicted and measured values after removing two outlier data points produced the largest errors for the HZ model. Since the HZ model was originally trained using a subset of data used here, it works on most data points. However, some data points have more than two orders of magnitude difference between the predicted and actual value in $\log_{10}$. These high errors came from calculated data points in the upper-temperature regimes (2500 K $\leq T \leq$ 4000 K).

As seen in Figure 6.10(a), there are some data points where the neural network is quite off. However, the errors on either extreme on the positive or negative sides do not cross more than an order of magnitude. On the negative side, the difference is slightly higher with some quite off-model values. The data points with errors below -0.5 mostly came from lower temperature regimes (625 K $\leq T <$ 1580 K). These are mostly exper-

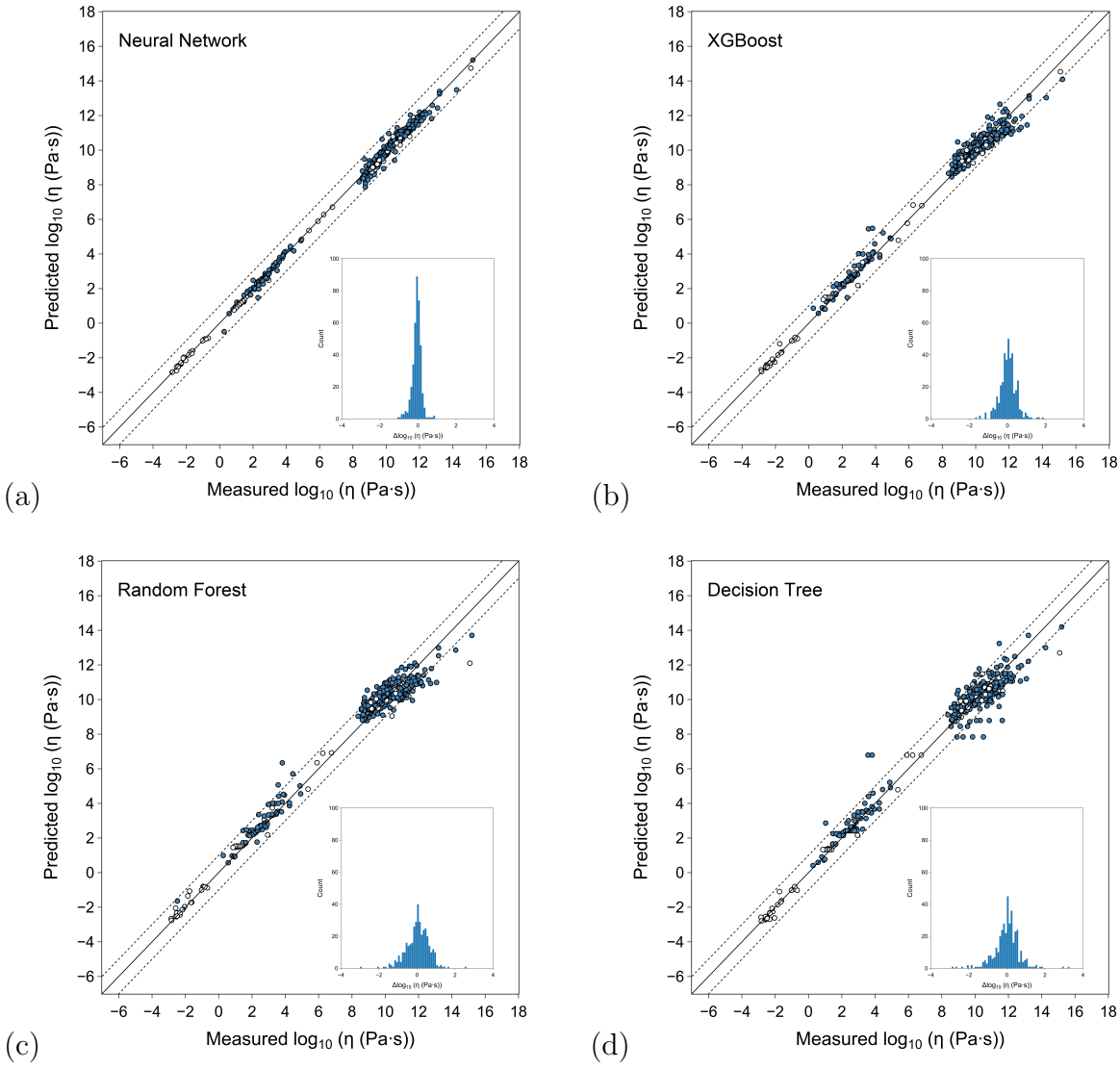


Figure 6.10. Test data model prediction scatter plots and error distributions for each model trained on zero pressure test data with $T \leq 4000K$. Blue and white circles in scatter plots represent hydrous and anhydrous compositions, respectively.

Table 6.3. Train-test RMSE for re-optimized HZ model. The coefficients were found using least squares while keeping the equation form the same as published. In cases where full datasets are optimized, the test RMSE and train RMSE columns show RMSE for only the test-split and train-split, respectively. The overall train RMSE column shows different values than the train RMSE column for full datasets because the training concatenates train and test splits before optimizing. In contrast, the train/test RMSE column shows errors for only the train and test splits.

Dataset	All Train RMSE	Train RMSE	Test RMSE
HZ (train-only)	0.28	0.28	0.40
HZ (full)	0.30	0.28	0.36
$P = 0, T \leq 4000$ (train-only)	0.35	0.35	0.44
$P = 0, T \leq 4000$ (full)	0.36	0.35	0.39

imentally measured high silica content (0.5 mole fraction) hydrous and anhydrous melts.

Similarly, data points with errors above 0.4 are coming from low-temperature regimes (600 K $< T <$ 1475 K).

$$
\begin{aligned}
\log \eta = \Big[& -6.65X_{\mathrm{SiO_2}} - 262.44X_{\mathrm{TiO_2}} - 7.04X_{\mathrm{A_2O_{3cx}}} - 3.006X_{\mathrm{MgO}} - 17.70X_{\mathrm{CaO}} \\
& + 32.66X_{\mathrm{(Na,K)_2O_{ex}}} - 139.82Z + 159.47X_{\mathrm{H_2O}} - 7.52X_{\mathrm{(Na,K)AlO_2}} \Big] \\
& + \Big[17.64X_{\mathrm{SiO_2}} + 396.95X_{\mathrm{TiO_2}} + 15.59X_{\mathrm{Al_2O_{3ex}}} + 0.27X_{\mathrm{MgO}} + 23.50X_{\mathrm{CaO}} \\
& - 69.52X_{\mathrm{(Na,K)_2O_{ex}}} + 47.02Z - 59.42X_{\mathrm{H_2O}} + 14.82X_{\mathrm{(Na,K)AlO_2}} \Big] 1000/T \\
& + \exp \Big\{ \big[-8.75X_{\mathrm{Al_2O_{3ex}}} - 48.99X_{\mathrm{(Fe,Mn)O}} - 56.50X_{\mathrm{MgO}} - 45.61X_{\mathrm{CaO}} \\
& - 62.93X_{\mathrm{(Na,K)_2O_{ex}}} + 81.59Z - 121.88X_{\mathrm{H_2O}} - 3.57X_{\mathrm{(Na,K)AlO_2}} \big] + \big[2.34X_{\mathrm{SiO_2}} \\
& - 111.90X_{\mathrm{TiO_2}} + 13.74X_{\mathrm{Al_2O_{3cx}}} + 30.13X_{\mathrm{(Fe,Mn)O}} + 63.13X_{\mathrm{MgO}} \\
& + 41.41X_{\mathrm{CaO}} + 40.86X_{\mathrm{(Na_aK)_2O_{ex}}} + 403.79X_{\mathrm{P_2O_5}} \\
& - 149.29Z + 200.66\ \mathrm{X_{H_2O}} \big] 1000/T \Big\}
\end{aligned}
\tag{6.1}
$$

Re-optimizing the HZ model using data from the train set would be a better

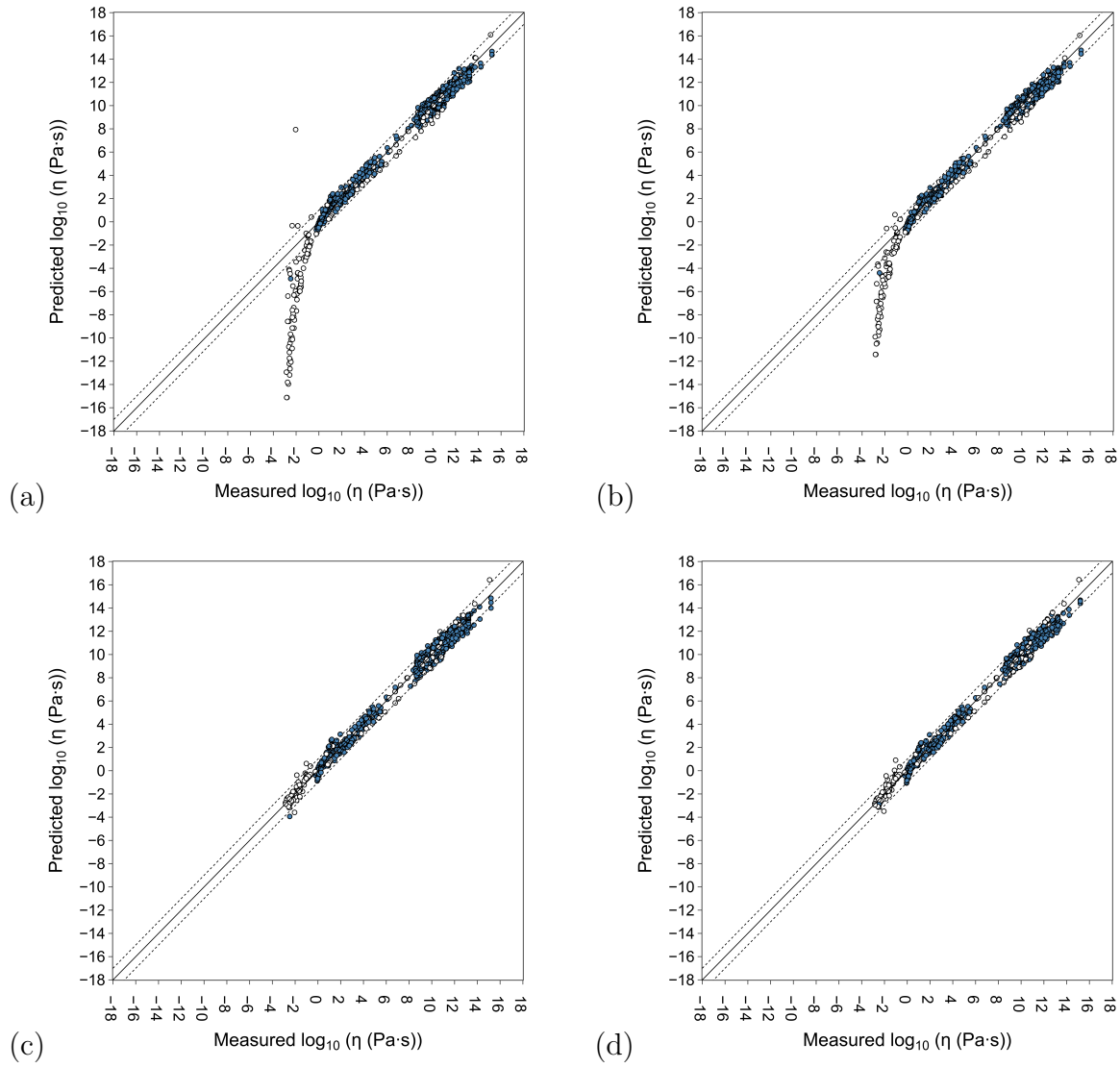


Figure 6.11. HZ model predictions for ambient pressure data with T $\leq$ 4000 K with theta obtained by (a) optimizing using train set from HZ dataset (b) from all HZ (c) optimizing using train set of all zero pressure data with T $\leq$ 4000 K (d) optimizing using all zero pressure data with T $\leq$ 4000 K. Since the original theta was obtained by optimizing mostly low-temperature (high-viscosity) data points, the model gives bad results (first row) in the high-temperature (low-viscosity) regime. Blue and white circles in scatter plots represent hydrous and anhydrous compositions, respectively.

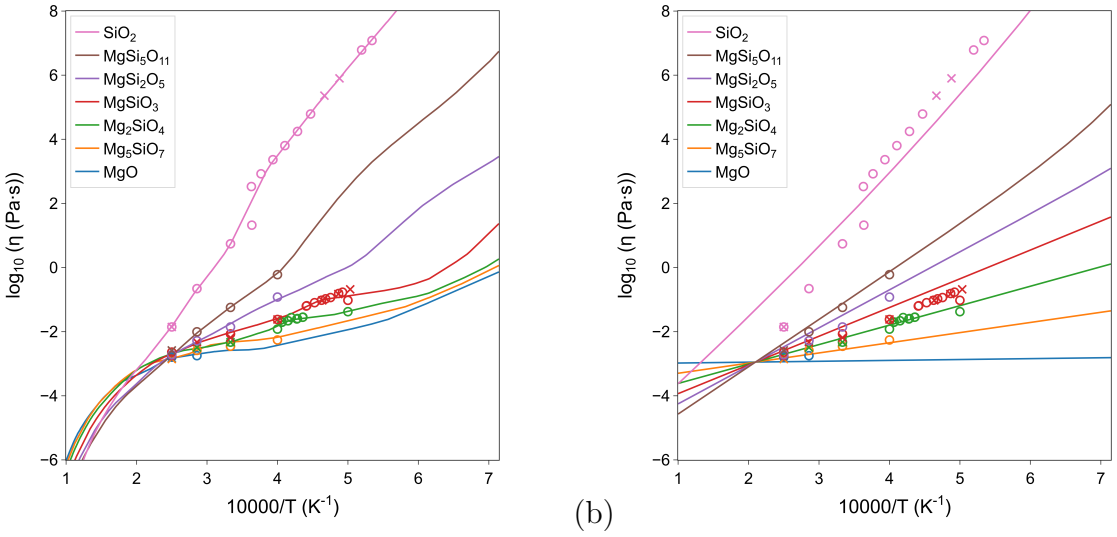


Figure 6.12. MgO-SiO_2 binary melts at different temperatures. Lines represent (a) neural network predictions and (b) reoptimized HZ using both train and test data. Circle (test) and cross (test) markers represent actual data points in the zero pressure and T $\leq$ 4000 K regime. The temperature axis is inverted; a low axis value means high temperature and vice versa. Colors show different compositions - magenta (SiO_2), brown ($MgSi_5O_{11}$), purple ($MgSi_2O_5$), red ($MgSiO_3$), green (Mg_2SiO_4), orange (Mg_5SiO_7), and blue (MgO).

comparison since the original model was optimized on a subset of data used here. To re-optimize HZ, we keep the original HZ equation form as is while using different data sets to get new equation coefficients. We follow the HZ method of optimizing equation coefficients using least squares for all of our tests. We got coefficients close to the published values with all HZ data. Then only the HZ train subset was used to find new coefficients. Since this subset contains fewer data than originally used by the authors, the predictive performance decreased slightly. As expected, these new model coefficients performed poorly for high-temperature data due to a lack of data during optimization. We then use the training set used to train machine learning models in this section to get a new set of equation coefficients. This training set contains high-temperature (low-viscosity) data therefore, it performs better than the original HZ model coefficients.

Table 6.4. Actual data for the MgO-SiO_2 binary system present in either training or testing dataset. MgO and SiO_2 components in weight percent (mole fraction in parenthesis), all other components have zero as value.

Composition	Name	SiO_2	MgO	Temperature range (K)	Data points
Pure MgO	Periclase	0 (0)	100 (1)	3000-4000	5
Mg_5SiO_7	Pe1	22.97 (0.16)	77.03 (0.83)	2500-4000	4
Mg_2SiO_4	Fosterite	42.71 (0.33)	57.29 (0.66)	2000-4000	16
$MgSiO_3$	Enstatite	59.85 (0.49)	40.15 (0.50)	1987-4000	21
$MgSi_2O_5$	Sil1	74.88 (0.66)	25.12 (0.33)	2500-4000	4
$MgSi_5O_{11}$	Sil2	88.17 (0.83)	11.83 (0.16)	2500-4000	4
Pure SiO_2	Silica	100 (1)	0 (0)	1523-4000	21

Further, the predictive performance increased as we added the training and testing sets during optimization. The temperature continuity of the neural network and XG model is shown in Figure 6.12 for compositions along the MgO-SiO_2 binary. Equation 6.1 shows the updated coefficients optimized using all zero-pressure data with $T \leq 4000$ K. Table 6.5 shows the derivative property glass transition temperature (T_g) calculated from various machine learning and HZ models.

6.2.4. Complete Dataset

In this section, we fit different models with all available data. This dataset contains all previously used parameters, with addition to CO_2 and P. All of our previously trained models contained data for ambient pressure. We now broaden our parameter space to include all data, including $P > 0$ GPa. Total data (2039) was then split into the train (1554) and test (485) sets.

This experiment used a bigger neural network architecture to incorporate the additional dimensions and data. Neural networks from previous experiments contained two hidden layers containing 64 nodes each. Here, they were doubled in size to four hidden layers containing 128 nodes in each layer. Further, we also updated the learning rate

Table 6.5. Actual and model predicted glass transition temperatures of four different compositions. Actual T_g^a refers to values obtained from Giordano et al. [18].

Components (wt%)	Rhyolite1	Rhyolite2	Basanite	Rhyolite3
SiO_2	76.38	76.38	41.17	76.29
TiO_2	0.06	0.06	2.74	0.14
Al_2O_3	11.59	11.59	12.10	12.04
FeO	1.03	1.03	10.10	1.37
MnO	0.05	0.05	0.18	0.08
MgO	0.36	0.36	11.24	0.04
CaO	3.25	3.25	15.66	0.30
Na_2O	2.44	2.44	2.76	3.39
K_2O	4.66	4.66	3.04	4.89
P_2O_5	0	0	1.02	0.01
H_2O	0	3	0	2
Actual T_g^a (K)	1037	739	938	770
Neural Network T_g (K)	1026	722	886	723
HZ T_g (K)	1047	698	935	741
HZ re-optimized T_g (K)	1056	694	934	741
XG T_g (K)	1036	658	905	741
RF T_g (K)	937	726	400	749
DT T_g (K)	912	730	949	755

(0.0004), weight decay (0.02), batch size (1024), and the number of epochs (50,000). Table 6.6 shows the average errors.

In this experiment, the test accuracy of the neural network model was once again lower than others. The addition of the pressure component did not affect the accuracy of the neural network model when compared to the experiments with ambient pressure data. The maximum difference between neural network predicted and actual values were within $\pm 1 \log_{10}$ orders of magnitude for both the train and test sets, while all other models produced even worse results. While XG errors were within $\pm 1.7 \log_{10}$, both RF and DT exceeded $\pm 2 \log_{10}$. Since most experimental data have the uncertainty of under a $\log_{10}$, any model that results in higher uncertainty produces unacceptable results. The neural network model is well within this range and is a candidate to explore further.

Neural networks are prone to overfitting. We examine if our trained network is

Table 6.6. Train and test errors in RMSE for different models trained using complete data including high-pressure data points.

Model	Train	Test
Decision Tree	0.01	0.44
Random Forest	0.15	0.36
XGBoost	0.04	0.29
Neural Network	0.06	**0.17**

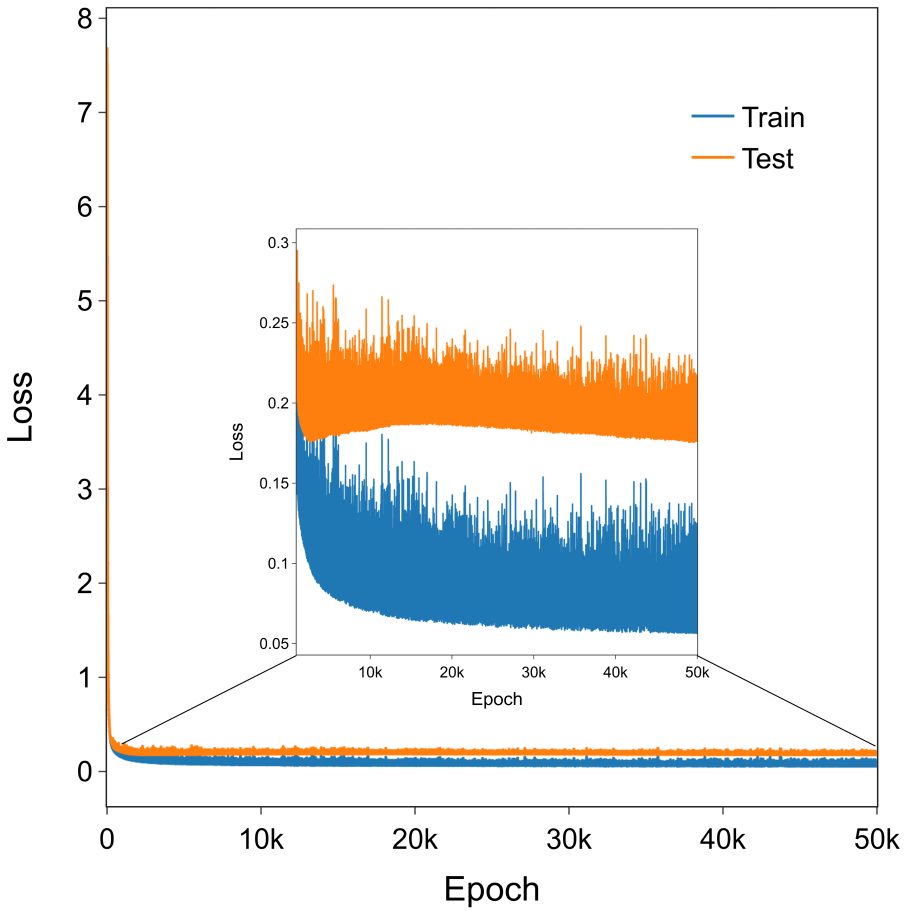


Figure 6.13. Neural network training progress for the complete dataset. Training and testing losses are shown for training epochs 1,000-50,000 in the inset.

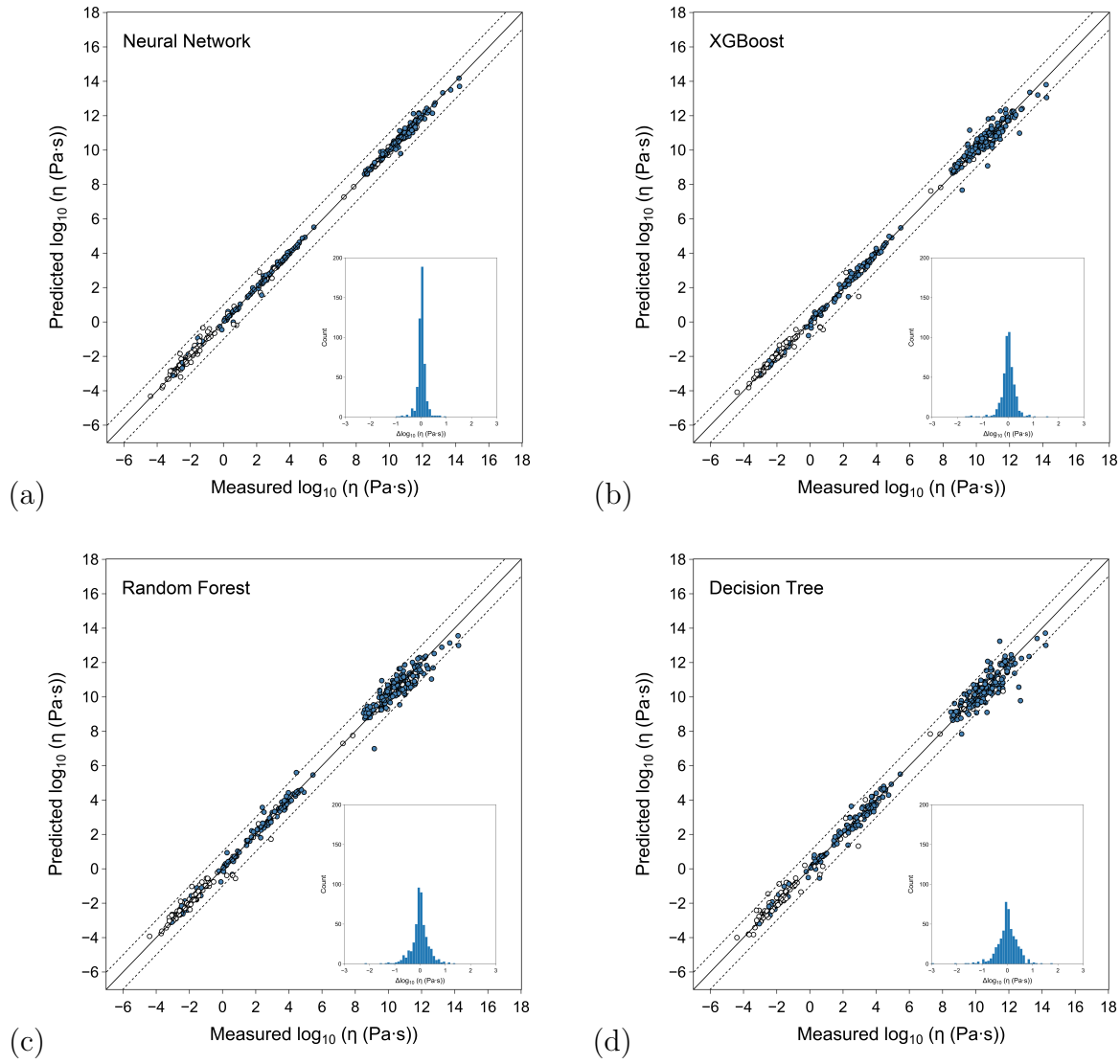


Figure 6.14. Scatter plot of measured vs. model predicted values on test data split from all available data. The center diagonal line represents $y = x$ while dotted lines on either side represent ± 1 $\log_{10}$ η. Colored circles represent hydrous compositions, and white-filled circles represent the rest of the data. Models are (a) neural network, (b) XG, (c) RF, and (d) DT. Error distributions for each model are shown as an inset within the scatter plot.

overfitting by looking at the training and test loss during the training progress. Figure 6.13 shows the training progress of the neural network from 600 - 50,000 epochs. The orange and blue lines show training and test accuracy at each epoch. Since neural networks are initialized randomly, the initial losses are high. These loss values are of little interest because they tend to decrease rapidly as the training progresses. We can observe in the figure that both the training and testing errors are decreasing, albeit slowly, with each epoch. Therefore, no sign of overfitting can be observed. However, both loss curves go up and down within a small range, but these fluctuations do not vary significantly. These can be generally tamed by adding stricter regularization (dropout, regularization on loss function, and others) on top of the weight decay.

The neural network predicted viscosity is close to the actual values. Most of the points on the scatter plot (Figure 6.14a) fall on the $y = x$ line and are far less spread out, decreasing the overall RMSE. The dotted lines in the figure show $\pm 1 \log_{10}$ boundaries. All test datapoints fall within these boundaries. The error histogram (Figure 6.14a, inset) shows that most predictions fall in the zero error region with no outliers. The boundary points fall in the outliers of natural $P - T - X$ conditions. For example, predictions for MgO-SiO_2 (0.3 and 0.6 mole fraction) binary compositions at 159, 67, and 0 GPa and 4000 and 3000 K produced the highest errors from the test set for all models. These calculated points do not contain oxides other than the pure binary. Only a handful of data are available for the binary compositions at these specific pressure and temperature points. The neural network produced prediction errors centered around zero for all other points in the test set, signifying a well-trained model.

Tree-based models' actual vs. predicted plots were spread out, with some points

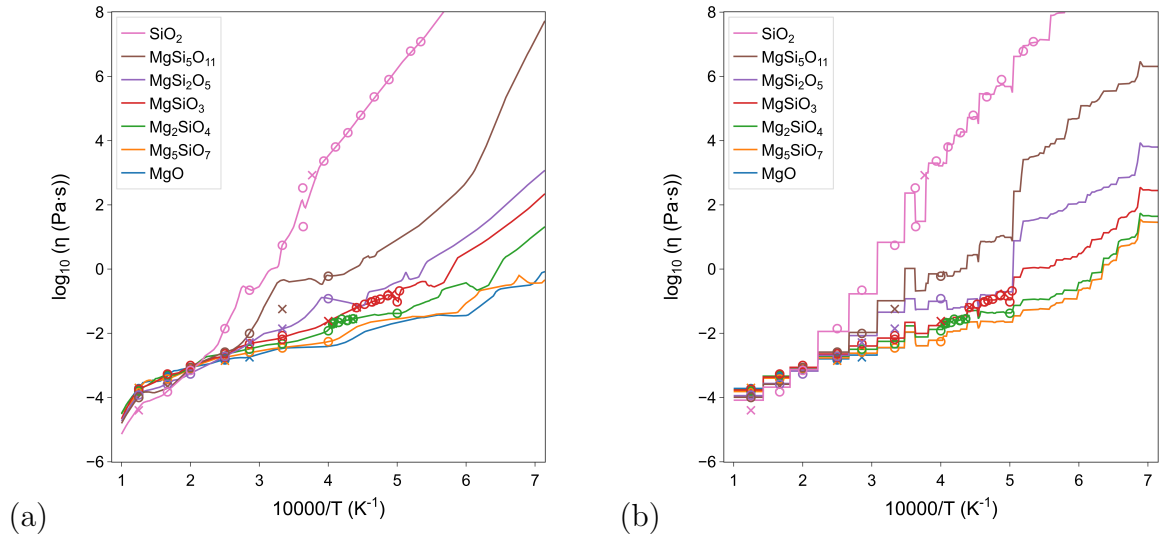


Figure 6.15. Model continuity with T shown in scatter plot using (a) neural network and (b) XG trained on all available data. Circles and crosses represent train and test data, respectively. Colors show different compositions - magenta (SiO_2), brown ($MgSi_5O_{11}$), purple ($MgSi_2O_5$), red ($MgSiO_3$), green (Mg_2SiO_4), orange (Mg_5SiO_7), and blue (MgO). The T axis is inverted, showing the axis line going from high-T to low-T regimes.

falling outside the dotted line boundaries. Some of these datapoints also produced high errors when evaluating with the neural network. Error histograms of these models were also centered around zero but were not as narrow as the neural network. Some outliers beyond the $\pm 1 \log_{10}$ are visible on either side of 0 on the histogram. Among the tree-based models, XG had the least RMSE and relatively closer predictions.

The continuity of trained models can be examined by generating points along its parameters and evaluating the predictions. Ideally, these predictions are smooth and contain fewer jumps. Some jumps are expected because they can be explained by data or model properties. The modeling goal is also to minimize expected and unexpected sudden changes for smoother predictions.

The T dependence for a few compositions along the MgO-SiO_2 binary is shown in Figure 6.15. The temperature axis has been inverted on both sub-figures such that the x-

axis goes from high to low temperature. Therefore, the inverse $\eta - T$ relationship is now seen as a positive relationship. Circles represent data used to train the respective models, while crosses represent testing data unseen during the training phase. Lines following these data points are model predictions.

The amount of silica (network former) positively affects the viscosity, while adding MgO (network modifier) has a negative relation. Therefore, silica-rich melts at similar temperatures have a viscosity higher than silica poor at similar temperatures. A comparison between the model-predicted values in Figure 6.15 shows that they are close to the existing data points. Further, these lines also show the inherent nature of each model. For instance, the prediction lines of XG are discontinuous with many sudden changes, which is in contrast to the relatively smoother neural network predictions. Tree-based models learn by dividing the training data space with rigid boundaries, while the neural network learns by optimizing its weights for minimum prediction error. For the regions within training data, neural network lines are less prone to sudden changes. The extrapolated regions, however, contain more spikes and valleys.

6.2.4.1. Glass Transition Temperature

The temperature at which silicate melts go through a phase change from liquid to glass or vice versa is known as the glass transition temperature (T_g). In normal conditions, silica glass forms a fully polymerized network structure. The random distribution of chemically ordered rings, where silicon atoms are linked by bridging oxygen, influences the structure of tetrahedral SiO_4. The structure of liquid silica, on the other hand, is different. The liquid is homogeneous in a large space-time. However, local ordering in liquids can in-

duce changes within the duration of their formation [51]. From the atomic point of view, these phase changes are not well understood, yet reliable prediction is essential in various geophysical processes. As a rule of thumb, the glass transition temperature is taken as the temperature of compositions when the viscosity value reaches 10^{12} Pa s.

Adding water is generally linked with an inverse relationship with T_g. We can study this relationship by using trained models. For example, to predict T_g as a function of water, we first linearly generate 10,000 temperature points between 400 - 1500 K. The generated temperature points are appended for each composition, totaling 10,000 input data points for the model. Since our trained neural network required data pre-processing, we transformed the data before predicting viscosity values corresponding to each data point using pre-trained models. Using the model-generated viscosity points, we pick the temperature as T_g that yields a viscosity close to 10^{12} Pa s.

We set P to zero for all input data points to the model to examine the glass transition temperature of specific compositions. The procedure from the previous section is repeated, and the results are shown in Table 6.7. Four compositions and their T_g from Giordano et al. [18] are compared to model-generated glass transition values. The highest absolute difference between predicted and actual T_g was within 100 K for neural network and XG, while RF and DT produced much larger errors.

For natural melts and pure SiO_2, the glass transition temperature decreases rapidly for a few percentages of water addition. In Figure 6.16, this behavior can be observed from both the neural network and XG predictions. Three compositions - pure SiO_2, Basanite, and Rhyolite1 were used to study the effect of increasing water content on T_g as predicted by different models. Each increment of the water content is subtracted from

Table 6.7. Actual and model predicted glass transition temperatures of four different compositions. The models were trained using the complete dataset. Actual T_g^a refers to values obtained from Giordano et al. [18].

Components (wt%)	Rhyolite1	Rhyolite2	Basanite	Rhyolite3
SiO_2	76.38	76.38	41.17	76.29
TiO_2	0.06	0.06	2.74	0.14
Al_2O_3	11.59	11.59	12.10	12.04
FeO	1.03	1.03	10.10	1.37
MnO	0.05	0.05	0.18	0.08
MgO	0.36	0.36	11.24	0.04
CaO	3.25	3.25	15.66	0.30
Na_2O	2.44	2.44	2.76	3.39
K_2O	4.66	4.66	3.04	4.89
P_2O_5	0	0	1.02	0.01
H_2O	0	3	0	2
Actual T_g^a (K)	1037	739	938	770
Neural Network T_g (K)	1038	821	946	787
XG T_g (K)	1033	645	909	728
RF T_g (K)	953	736	400	766
DT T_g (K)	1047	400	1129	744

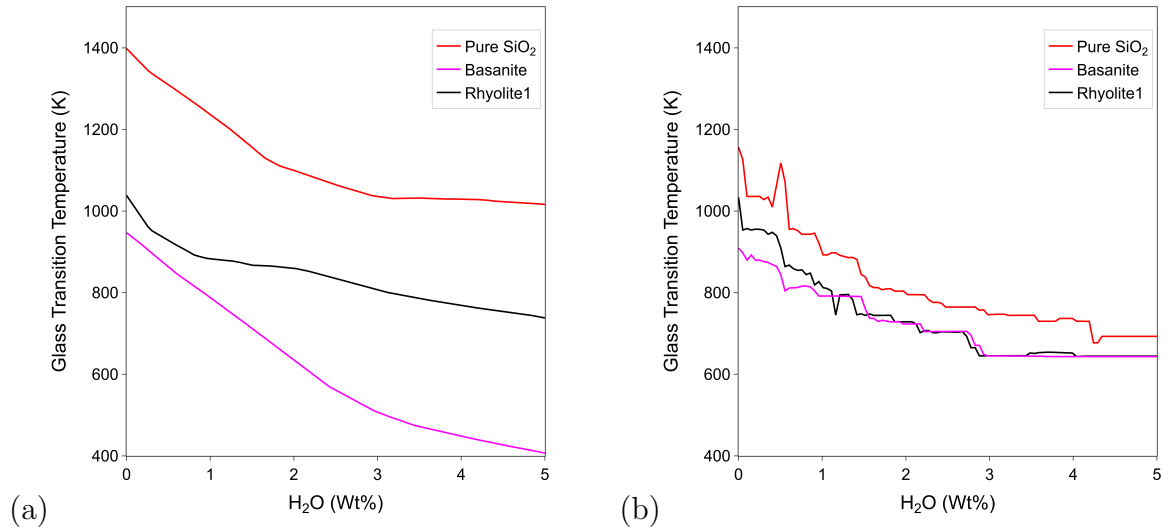


Figure 6.16. The T_g of a melt decreases with the addition of water content. Changes are rapid when adding the first few wt% of water. Models are (a) neural network and (b) XG. Colors represent compositions Rhyolite1 (black), Basanite (magenta), and pure SiO_2 (red).

the silica component of the composition. The model results agree with the general trend of decreasing viscosity with more water content.

6.2.4.2. Anomalous Behavior

Unlike the widely agreed inverse relationship of $T - \eta$, the P effect has a more complex behavior. For depolymerized melts such as $CaMgSi_2O_6$ [54] and peridotite [43], the viscosity increases with increasing pressure up to many gigapascals. An anomalous behavior is observed for polymerized melts like silica [32] and enstatite [33, 44] where the viscosity first decreases with compression up to a minimum and increases further after. Due to the difficulties in operating with elevated temperature and pressure regimes, only a few data points have been available. Recently with deep learning techniques, more data have been added to existing experimental and calculated data.

We generated 1000 datapoints each for wt% pure silica (100) and $MgSiO_3$(SiO_2 59.85 and MgO 40.15) to evaluate their predicted η over P by neural network and XG. The iso-therm curves for two melts shown in Figure 6.17 follow closely with existing datasets. For both melts, viscosity first decreases and then increases with the addition of pressure. Both neural network and XG models were able to replicate this behavior. Within the training data distribution (or interpolation region), a few test points (crosses) were not used during training. Qualitatively, the neural network predictions were closer to actual data points at even extreme conditions, for example, the high-P point at 4000 K for silica when compared with XG. The interpolation region of neural network was smoother than that of XG's. However, both models did succeed in capturing the general trend of data. Adding more data would help train both models to replicate this complex data

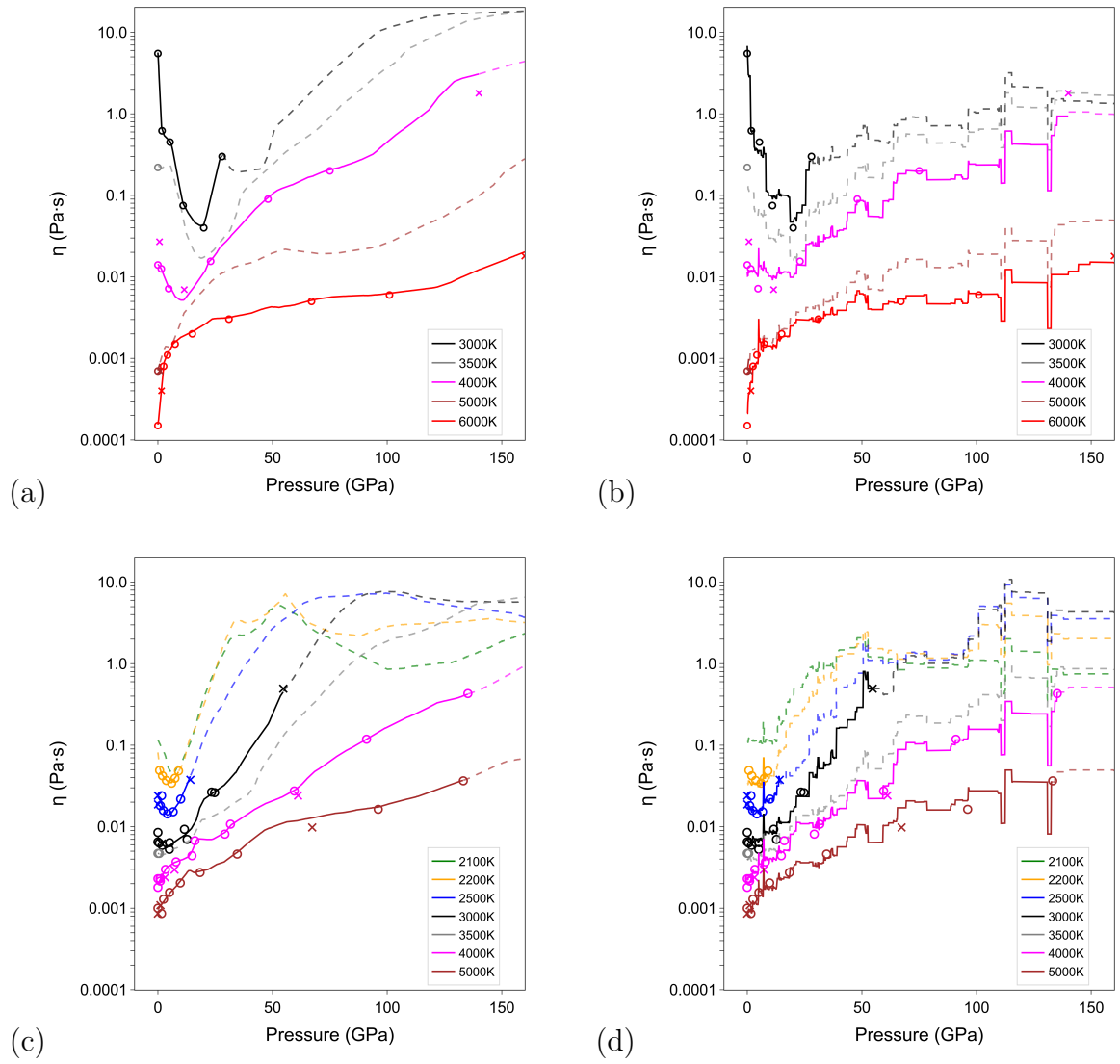


Figure 6.17. Anomalous and normal behavior of pure silica (first row) and $MgSiO_3$ (second row) respectively, at temperatures 3000 K (black), 3500 K (gray), 4000 K (magenta), 5000 K (brown), and 6000 K (red) with viscosity values generated by trained neural network (first column) and XG (second column) models. Actual data points from both train (open circles) and test (crosses) sets are overlayed on top of model predictions (solid and dashed lines). The dashed lines represent model extrapolation regions.

regime better.

6.2.5. Model Limitations

We have presented machine learning models that could follow the general trend of the melt viscosity data even at elevated $P - T - X$ conditions. Modeling this complex multivariate data poses several challenges due to the need to explore a large hypothesis space. Further, contrasting $P - \eta$ behaviors for different melts and temperature is difficult to model using single or many parametric equations.

Lack Of Data. At high $P - T$ conditions it becomes difficult to conduct experiments. FPMD simulations and other techniques have helped create new data that generally agree with the existing experimental results. Even at ambient pressure, there are few noticeable gaps, such as the crystallization interval. This interval is an experimental artifact and is recorded in several experiments. In addition, new techniques and technology in designing experiments are helping to fill this gap. In any case, adding new data that covers even more diverse and elevated $P - T - X$ conditions would help train better models.

Tree-based Models. The tree-based models perform well on medium and small-size data. Further, incorporating gradient boosting techniques while building trees can have a noticeable impact on model predictive performance. Tree-based models perform better on classification than regression tasks because tree construction algorithms divide data space with hard boundaries. Regression with these models requires grouping and aggregating the target value of data points in the leaf node. Aggregating data points, such as taking an average, typically results in information loss. Therefore these models often perform poorly on sparse and unstructured data.

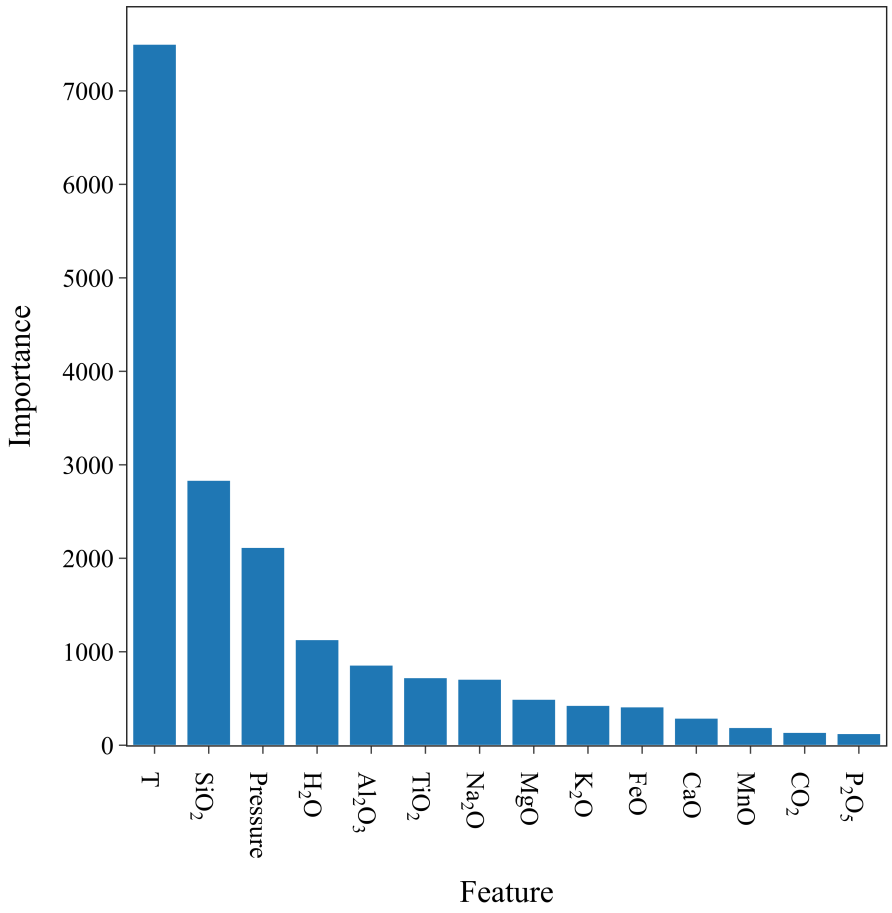


Figure 6.18. Feature importance by the number of times a feature was used to split data across all trees in XG trained with the complete dataset. T, SiO_2, and P play an important role in building a model to predict the viscosity of a composition.

Table 6.8. Parameter counts of different neural network architectures.

Architecture	Parameter Count
64x2	9345
128x4	68097

Tree depth and the number of trees (for ensemble models like RF and XG) provide flexibility in optimizing data. Careful tuning is required for both parameters, as small changes to them could result in large prediction deviations. Mainly, trees with large depths are prone to overfitting and require pruning. The hard boundaries setup by tree-constructing algorithms results in step-like discontinuous predictions with parameters that do not allow for reliable prediction even in the interpolation region. However, the same algorithm establishes a machine learning model baseline. It also enables feature importance comparison as shown in Figure 6.18 for the complete dataset with XG.

Neural Networks. Neural networks are a powerful tool for modeling complex data. Although they can take in any numerical values, our experiments found them to train well for normalized input data. For the tree-based models, input normalization made a minimal impact on model performance. Additionally, the random initialization of neural networks makes it challenging to maintain reproducibility across training sessions, even with the same architecture, training, and testing data. Therefore, using a single fixed seed for all random operations helps maintain a stable basis across experiments.

A significant disadvantage of neural network models is that they require many learning parameters. For example, a neural network with two hidden layers, each containing 64x2 nodes, contains a total of 9345 parameters. Similarly, when we double the size of the network to 4 hidden layers containing 128 nodes in each layer, the parameter count goes to 68097 ($\sim$7 times the smaller network). On the other hand, physics-based models

with known equation forms use a relatively small number of learnable coefficients to get the best fit for data. While a small number of parameters is desirable, it is challenging to implement in practice since it is difficult to construct a single equation that works in the vast parameter space while requiring only a handful of trainable parameters. Techniques such as knowledge distillation can help decrease the size of the trained neural network while keeping the same model performance.

In our experiments, we observed several small fluctuations in training and testing loss throughout the training of neural networks. These loss fluctuations become more prominent with the size of the network. Different values for weight decay and augmenting the loss function with additional regularization terms could help make these loss curves smoother. Additionally, with more extensive datasets, a network could model complex regions better.

Chapter 7. Conclusions and Future Works

Materials data collected using advanced experimental and calculated techniques have been gaining interest in recent years. Even for specific materials, such as silicate melts, geo-scientists have collected a significant amount of data for analysis. Silicate melt property such as viscosity is crucial to understanding Earth's geo-history. In this study, silicate melt viscosity data are collected and used for visualization and modeling. In addition, a web application is developed, which contains a database, visualization, and modeling modules. Users can filter, explore, and download the filtered data in a standard format such as *csv*. User interaction involves axes brushing, reordering, merging, scaling, and flipping in scatter and Parallel Coordinates Plots.

The viscosity of silicate melts exhibits complex variations in response to changes in pressure, temperature, and composition ($P - T - X$), making it a challenging aspect to model. Four machine learning models, Decision Trees, Random Forests, XGBoost, and neural networks, along with physical models, were evaluated for silicate melts viscosity data. Neural networks outperformed all models and could calculate complex derivative properties such as glass transition temperature and anomalous behavior. The trained model captures a wide range of $P - T - X$ conditions. Further, model-generated points were continuous over parameters within the training data region. Tree-based models, particularly XGBoost, were close to the neural network's predictions. However, these models' inherent nature of splitting the data space with decision boundaries makes the predictions step-like with possible significant changes. A particular challenge lies in modeling the P dependence on η where polymerized and depolymerized melts show opposite behaviors in the first few GPa. The neural network's generated points lie close to the actual points and

follow the anomalous behavior of silicate melts. Pre-trained models are available to explore and download in the web application.

Machine learning systems require a considerable amount of data for proper training. Therefore, more work on collecting new data would help improve model performance. Further exploring different data pre-processing techniques for each model may gain some improvements. Besides that, in the case of neural networks, a better loss function corresponding to the physical nature of melts would help steer the training progress in the correct direction. Additionally, generating synthetic points from a trained neural network, like an auto-encoder, to further use those points as training samples to train a bigger regression network than used in this study is a future avenue to explore. Finally, as we add even more data, transfer learning from other pre-trained large regression networks could also be helpful. Similar to the study presented here, we can use a similar workflow in visualizing and modeling other material properties.

Appendix A. Copyright Permission

Copyright permission from SCITEPRESS:

Dear Diwas,

We authorize the use of the paper for your thesis, as long as all the bibliographic information from its publication is there too, and it is properly cited and referenced. If you use any figures or tables from the original paper, please also cite the paper in the figure or table's caption.

--
Best regards,
Ana Rita Paciência
SCITEPRESS Team

SCITEPRESS Office
Avenida de S. Francisco Xavier Lote 7 Cv. C, 2900-616 Setubal - Portugal
Tel.: +351 265 520 184/5
Fax: +351 265 520 186
http://www.scitepress.org/

On Friday, January 13th 2023, 6:09 pm CET (+0100), dbhatt7 lsu.edu wrote:

> Dear Sir/Madam,
>
> I would like to request the use of a copyrighted published material, where I am the first author. I will be using the material for my PhD dissertation at Louisiana State University which will be uploaded to DigitalCommons. Paper details below:
>
> Bhattarai, D.; Zhang, J. and Karki, B. (2019). Parallel Coordinates-based Visual Analytics for Materials Property. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - IVAPP,* ISBN 978-989-758-354-4; ISSN 2184-4321, pages 83-95. DOI: 10.5220/0007375400830095
>
> Please grant me the copyright to use the material from above paper in my thesis.
>
> Thank you,
> Diwas

Bibliography

[1] Y. Abe. Thermal and chemical evolution of the terrestrial magma ocean. *Physics of Earth and Planetary Interiors*, 100:27 – 39, 1997.

[2] O. Adjaoud, G. Steinle-Neumann, and S. Jahn. Transport properties of Mg_2SiO_4 liquid at high pressure: Physical state of a magma ocean. *In Earth and Planetary Science Letters*, 312:463 – 470, 2011.

[3] I. Avramov and A. Milchev. Effect of disorder on diffusion and viscosity in condensed systems. *Journal of Non-Crystalline Solids*, 104(2):253 – 260, 1988.

[4] H. Behrens and F. Schulze. Pressure dependence of melt viscosity in the system $NaAlSi_3O_8$-$CaMgSi_2O_6$. *American Mineralogist*, 88(8/9):1351, 2003.

[5] D. Bhattarai. *Space-time multiresolution approach to atomistic visualization*. PhD dissertation, Louisiana State University, 2008.

[6] D. Bhattarai., J. Zhang., and B. B. Karki. Parallel coordinates-based visual analytics for materials property. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - IVAPP*, pages 83 – 95. INSTICC, SciTePress, 2019.

[7] M. Bostock, V. Ogievetsky, and J. Heer. D3: Data-driven documents. In *IEEE Trans. Visualization & Comp. Graphics (Proc. Infovis)*, 2011.

[8] Y. Bottinga and D. F. Weill. The viscosity of magmatic silicate liquids; a model calculation. *American Journal of Science*, 272(5):438 – 475, 1972.

[9] R. Bruckner. Structure and properties of silicate melts. *Bull. Mine*, (106), 1983.

[10] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108 – 122, 2013.

[11] R. M. Canup. Dynamics of lunar formation. *Annu. Rev. Astron. Astrophys*, 42:441 – 475, 2004.

[12] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785 – 794, New York, NY, USA, 2016.

[13] B. Cochain, C. Sanloup, C. Leroy, and Y. Kono. Viscosity of mafic magmas at high

pressures. *Geophysical Research Letters*, 44(2):818 – 826, 2017.

[14] D. B. Dingwell and D. Virgo. The effect of oxidation state on the viscosity of melts in the system Na_2O-FeO-Fe_2O_3-SiO_2. *Geochimica et Cosmochimica Acta*, 51(2):195 – 205, 1987.

[15] D. B. Ghosh and B. B. Karki. Diffusion and viscosity of Mg_2SiO_4 liquid at high pressure from first-principles simulations. *Geochimica et cosmochimica acta*, (16):4591 – 4600, 2011.

[16] D. B. Ghosh and B. B. Karki. Transport properties of carbonated silicate melt at high pressure. *In Science Advances*, 3:e1701840, 2017.

[17] D. Giordano and D. B. Dingwell. Non-arrhenian multicomponent melt viscosity: a model. *Earth and Planetary Science Letters*, 208(3):337 – 349, 2003.

[18] D. Giordano, J. K. Russell, and D. B. Dingwell. Viscosity of magmatic liquids: A model. *In Earth and Planetary Science Letters*, 27:123 – 134, 2008.

[19] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249 – 256, Chia Laguna Resort, Sardinia, Italy, 2010. PMLR.

[20] T. Gordon and K. Russell. *Silicate Melt Viscosity Calculator*, 2008.

[21] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357 – 362, 2020.

[22] J. Heinrich and D. Weiskopf. State of the art of parallel coordinates. *In STAR Proceedings of Eurographics*, pages 95 – 116, 2013.

[23] K.-U. Hess and D. B. Dingwell. Viscosities of hydrous leucogranitic melts: a non-arrhenian model. *In American Mineralogist*, 81:1297 – 1300, 1996.

[24] H. Hui and Y. Zhang. Toward a general viscosity equation for natural anhydrous and hydrous silicate melts. *In Geochimica et Cosmochimica Acta*, 71(2):403 – 416, 2007.

[25] K. ichi Funakoshi, A. Suzuki, and H. Terasaki. In situ viscosity measurements of albite melt under high pressure. *Journal of Physics: Condensed Matter*, 14(44):11343 –

11347, 2002.

[26] A. Inselberg. *Parallel coordinates: visual multidimensional geometry and its application*. Springer, New York, 2009.

[27] J. Johansson and C. Forsell. Evaluation of parallel coordinates: Overview, categorization and guidelines for future research. *IEEE Transactions on Visualization and Computer Graphics*, pages 579 – 588, 2016.

[28] B. B. Karki. *Viscosity Calculator for MgO-SiO_2 Melts*, 2013.

[29] B. B. Karki. First principles computation of mantle materials in crystalline and amorphous phases. *Physics of the Earth and Planetary Interiors*, 240:43 – 69, 2015.

[30] B. B. Karki, B. Bohara, and L. Stixrude. First-principles study of diffusion and viscosity of anorthite ($CaAl_2Si_2O_8$) liquid at high pressure. *American Minerologist*, 96(5-6):744 – 751, 2011.

[31] B. B. Karki, D. B. Ghosh, C. Maharjan, S.-i. Karato, and J. Park. Density-pressure profiles of Fe-bearing $MgSiO_3$ liquid; effects of valence and spin states, and implications for the chemical evolution of the lower mantle. *Geophysical Research Letters*, 45(9):3959 – 3966, 2018.

[32] B. B. Karki and L. Stixrude. First-principles study of enhancement of transport properties of silica melt by water. *Physical Review Letters*, 104(21):215901, 2010.

[33] B. B. Karki and L. Stixrude. Viscosity of $MgSiO_3$ liquid at earth's mantle conditions: Implications for an early magma ocean. *Science*, 328(5979):740, 2010.

[34] B. B. Karki, J. Zhang, and L. Stixrude. First-principles viscosity and derived models for MgO-SiO_2 melt system at high temperatures. *Geophysical Research Letters*, 40:94 – 99, 2013.

[35] G. Kaur and B. B. Karki. Bifocal parallel coordinates plot for multivariate data visualization. *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2018)*, 3:176 – 183, 2018.

[36] Y. Kono and C. Sanloup. *Magmas Under Pressure Advances in High-Pressure Experiments on Structure and Properties of Melts*. Elsevier, Inc, 2018.

[37] R. Kosara. Visualization criticism - the missing link between information visualization and art. In *2007 11th International Conference Information Visualization (IV '07)*, pages 631 – 636, 2007.

[38] I. Kushiro. Viscosity and structural changes of albite ($NaAlSi_3O_8$) melt at high pressures. *Earth and Planetary Science Letters*, 41(1):87 – 90, 1978.

[39] I. Kushiro, H. S. Yoder Jr., and B. O. Mysen. Viscosities of basalt and andesite melts at high pressures. *Journal of Geophysical Research (1896-1977)*, 81(35):6351 – 6356, 1976.

[40] D. J. Lacks, D. B. Rear, and J. A. Van Orman. Molecular dynamics investigation of viscosity, chemical diffusivities and partial molar volumes of liquids along the MgO-SiO_2 join as functions of pressure. *Geochimica et cosmochimica acta*, 71(5):1312 – 1323, 2007.

[41] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164 – 168, 1944.

[42] C. Liebske, H. Behrens, F. Holtz, and R. A. Lange. The influence of pressure and composition on the viscosity of andesitic melts. *Geochimica et Cosmochimica Acta*, 67(3):473 – 485, 2003.

[43] C. Liebske, B. Schmickler, H. Terasaki, B. T. Poe, A. Suzuki, K. ichi Funakoshi, R. Ando, and D. C. Rubie. Viscosity of peridotite liquid up to 13 GPa: Implications for magma ocean viscosities. *Earth and Planetary Science Letters*, 240(3):589 – 604, 2005.

[44] H. Luo, B. B. Karki, D. B. Ghosh, and H. Bao. Anomalous behavior of viscosity and electrical conductivity of $MgSiO_3$ melt at mantle conditions. *Geophysical Research Letters*, 48(13):e2021GL093573, 2021.

[45] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431 – 441, 1963.

[46] J. C. Mauro, Y. Yue, A. J. Ellison, P. K. Gupta, and D. C. Allan. Viscosity of glass-forming liquids. *Proceedings of National Academy of Sciences*, 106:19780 – 19784, 2009.

[47] G. H. Miller, E. M. Stolper, and T. J. Ahrens. The equation of state of a molten komatiite: 1 shock-wave compression to 36 GPa. *J. Geophys. Res. – Solid Earth and Planets*, 96:11831 – 11848, 1991.

[48] S. Mori, E. Ohtani, and A. Suzuki. Viscosity of the albite melt to 7 GPa at 2000 K. *Earth and Planetary Science Letters*, 175(1):87 – 92, 2000.

[49] D. R. Neuville, P. Courtial, D. B. Dingwell, and P. Richet. Thermodynamic and rheological properties of rhyolite and andesite melts. *Contributions to Mineralogy and Petrology*, 113(4):572 – 581, 1993.

[50] D. Nevins, F. J. Spera, and M. S. Ghiorso. Shear viscosity and diffusion in liquid $MgSiO_3$: Transport properties and implications for terrestrial planet magma oceans. *American Mineralogist*, 94(7):975 – 980, 2009.

[51] M. I. Ojovan and R. F. Tournier. On structural rearrangements near the glass transition temperature in amorphous silica. *Materials (Basel)*, 14(18):5235, 2021.

[52] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024 – 8035. Curran Associates, Inc., 2019.

[53] E. S. Persikov. Viscosities of model and magmatic melts at the pressures and temperatures of the earth's crust and upper mantle. *Russ. Geol. Geophys.*, 39:1780 – 1792, 1998.

[54] J. E. Reid, A. Suzuki, K.-I. Funakoshi, H. Terasaki, B. T. Poe, D. C. Rubie, and E. Ohtani. The viscosity of $CaMgSi_2O_6$ liquid at pressures up to 13 GPa. *Physics of the Earth and Planetary Interiors*, 139(1):45 – 54, 2003.

[55] E. F. Riebling. Structure of sodium aluminosilicate melts containing at least 50 mole % SiO_2 at 1500°C. *Journal of Chemical Physics*, 44(8):2857, 1966.

[56] J. K. Russell, K.-U. Hess, and D. B. Dingwell. Models for Viscosity of Geological Melts. *Reviews in Mineralogy and Geochemistry*, 87(1):841 – 885, 2022.

[57] C. Scarfe, B. Myson, and D. Virgo. Pressure dependence of the viscosity of silicate melts. *Mysen BO (ed) Magmatic processes: physicochemical principles, Geochemical Society*, 1:59 – 67, 1987.

[58] H. R. Shaw. Viscosities of magmatic silicate liquids: an empirical method of prediction. *In American Journal of Science*, 272:870 – 893, 1972.

[59] V. Solomatov. Grain size-dependent viscosity convection and the thermal evolution of the earth. *Earth and Planetary Science Letters*, 191(3):203 – 212, 2001.

[60] F. J. Spera, D. Nevins, M. Ghiorso, and I. Cutler. Structure, thermodynamic and transport properties of $CaAl_2Si_2O_8$ liquid. part I: Molecular dynamics simulations. *Geochimica et cosmochimica acta*, 73(22):6918 – 6936, 2009.

[61] J. G. Spray. Frictional melting processes in planetary materials: From hypervelocity impact to earthquakes. *Annual Review of Earth & Planetary Sciences*, 38(1):221 –

254, 2010.

[62] A. Suzuki, E. Ohtani, K. Funakoshi, H. Terasaki, and T. Kubo. Viscosity of albite melt at high pressure and high temperature. *Physics and Chemistry of Minerals*, 29(3):159 – 165, 2002.

[63] A. Suzuki, E. Ohtani, H. Terasaki, and K. Funakoshi. Viscosity of silicate melts in $CaMgSi_2O_6$-$NaAlSi_2O_6$ system at high pressure. *Physics and Chemistry of Minerals*, 32(2):140 – 145, 2005.

[64] A. Suzuki, E. Ohtani, H. Terasaki, K. Nishida, H. Hayashi, T. Sakamaki, Y. Shibazaki, and T. Kikegawa. Pressure and temperature dependence of the viscosity of a $NaAlSi_2O_6$ melt. *Physics and Chemistry of Minerals*, 38(1):59 – 64, 2011.

[65] D. Tinker, C. E. Lesher, G. M. Baxter, T. Uchida, and Y. Wang. High-pressure viscometry of polymerized silicate melts and limitations of the eyring equation. *American Mineralogist*, 89(11-12):1701 – 1708, 2004.

[66] R. Tuor, F. Evéquoz, and D. Lalanne. Parallel bubbles: Categorical data visualization in parallel coordinates. In *Actes de La 28ième Conference Francophone Sur l'Interaction Homme-Machine*, IHM '16, page 299 – 306, New York, NY, USA, 2016. Association for Computing Machinery.

[67] C. Turkay, P. Filzmoser, and H. Hauser. Brushing dimensions - a dual visual analysis model for high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 17:2591 – 2599, 2011.

[68] G. Urbain, Y. Bottinga, and P. Richet. Viscosity of liquid silica, silicates, and alumino-silicates. *Geochimica et Cosmochimica Acta*, 46:1061 – 1072, 1982.

[69] A. K. Verma and B. B. Karki. First-principles study of self-diffusion and viscous flow in diopside ($CaMgSi_2O_6$) liquid. *American Mineralogist*, 97(11-12):2049 – 2055, 2012.

[70] J. Wang, X. Liu, H. W. Shen, and G. Lin. Multiresolution climate ensemble parameter analysis with nested parallel coordinates plots. *IEEE Transactions on Visualization and Computer Graphics*, 23:81 – 90, 2016.

[71] Y. Wang, T. Sakamaki, L. B. Skinner, Z. Jing, T. Yu, Y. Kono, C. Park, G. Shen, M. L. Rivers, and S. R. Sutton. Atomistic insight into viscosity and density of silicate melts under pressure. *Nature Communications*, 5, 2014.

[72] Y. Zhang, Z. Xu, and Y. Liu. Viscosity of hydrous rhyolitic melts inferred from kinetic experiments, and a new viscosity model. *American Mineralogist*, 88:1741 – 1752, 2003.

[73] Y. Zhang, Z. Xu, M. Zhu, and H. Wang. Silicate melt properties and volcanic eruptions. *Reviews of Geophysics*, 45(4), 2007.

Vita

Diwas Bhattarai is originally from Kathmandu, Nepal. He earned a Bachelor's degree in Computer Science with a minor in Mathematics from Southeastern Louisiana University. Before enrolling in the graduate program, Diwas worked as a mobile and web developer at Amedisys, Inc.

Diwas is pursuing his Ph.D. in Computer Science under the supervision of Dr. Bijaya Karki at Louisiana State University. His research focuses on data visualization and modeling for material properties. During his internship at T. Baker Smith, LLC, he developed real-time alerts using IoT devices and performed data analysis for flood prevention systems across Louisiana. He anticipates receiving the Doctor of Philosophy degree in Computer Science in May 2023.